计 算 机 科 学 丛 书

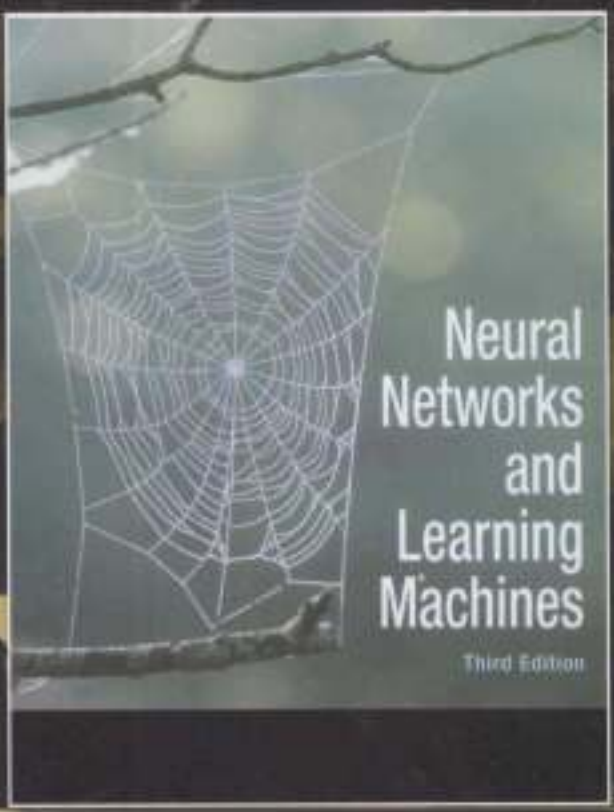
原书第3版

神经网络与机器学习

（加）Simon Haykin 著　申富饶 徐烨 郑俊 晁静 译

Neural Networks and Learning Machines

Third Edition



机械工业出版社
China Machine Press

神经网络与机器学习（原书第3版）

Neural Networks and Learning Machines Third Edition

神经网络是计算智能和机器学习的重要分支，在诸多领域都取得了很大的成功。在众多神经网络著作中，影响最为广泛的是Simon Haykin的《神经网络原理》（第3版更名为《神经网络与机器学习》）。在本书中，作者结合近年来神经网络和机器学习的最新进展，从理论和实际应用出发，全面、系统地介绍了神经网络的基本模型、方法和技术，并将神经网络和机器学习有机地结合在一起。

本书不但注重对数学分析方法和理论的探讨，而且也非常关注神经网络在模式识别、信号处理以及控制系统等实际工程问题中的应用。本书的可读性非常强，作者举重若轻地对神经网络的基本模型和主要学习理论进行了深入探讨和分析，通过大量的实验报告、例题和习题来帮助读者更好地学习神经网络。

本版在前一版的基础上进行了广泛修订，提供了神经网络和机器学习这两个越来越重要的学科的最新分析。

本书特色

- 基于随机梯度下降的在线学习算法：小规模和大规模学习问题。
- 核方法，包括支持向量机和表示定理。
- 信息论学习模型，包括独立分量分析（ICA）、相关独立分量分析和信息瓶颈等。
- 随机动态规划，包括逼近和神经动态规划。
- 逐次状态估计算法，包括卡尔曼和粒子滤波器。
- 利用逐次状态估计算法训练递归神经网络。
- 富有洞察力的面向计算机的实验。

作者简介



Simon Haykin 于1953年获得英国伯明翰大学博士学位，目前为加拿大McMaster大学电子与计算机工程系教授、通信研究实验室主任。他是国际电子电气工程界的著名学者，曾获得IEEE McNaughton金奖。他是加拿大皇家学会院士、IEEE会士，在神经网络、通信、自适应滤波器等领域成果颇丰，著有多部教材。



英文影印版
ISBN：978-7-111-26528-3
定价：69.00元

客服热线：(010) 88378991, 88361066
购书热线：(010) 68326294, 88379649, 68995259
投稿热线：(010) 88379604
读者信箱：hzjsj@hzbook.com

华章网站 http://www.hzbook.com

网上购书：www.china-pub.com

PEARSON Education

PEARSON

www.pearsonhighered.com

上架指导：计算机/神经网络
ISBN 978-7-111-32413-3
9 787111 324133

定价：79.00元

计 算 机 科 学 丛 书

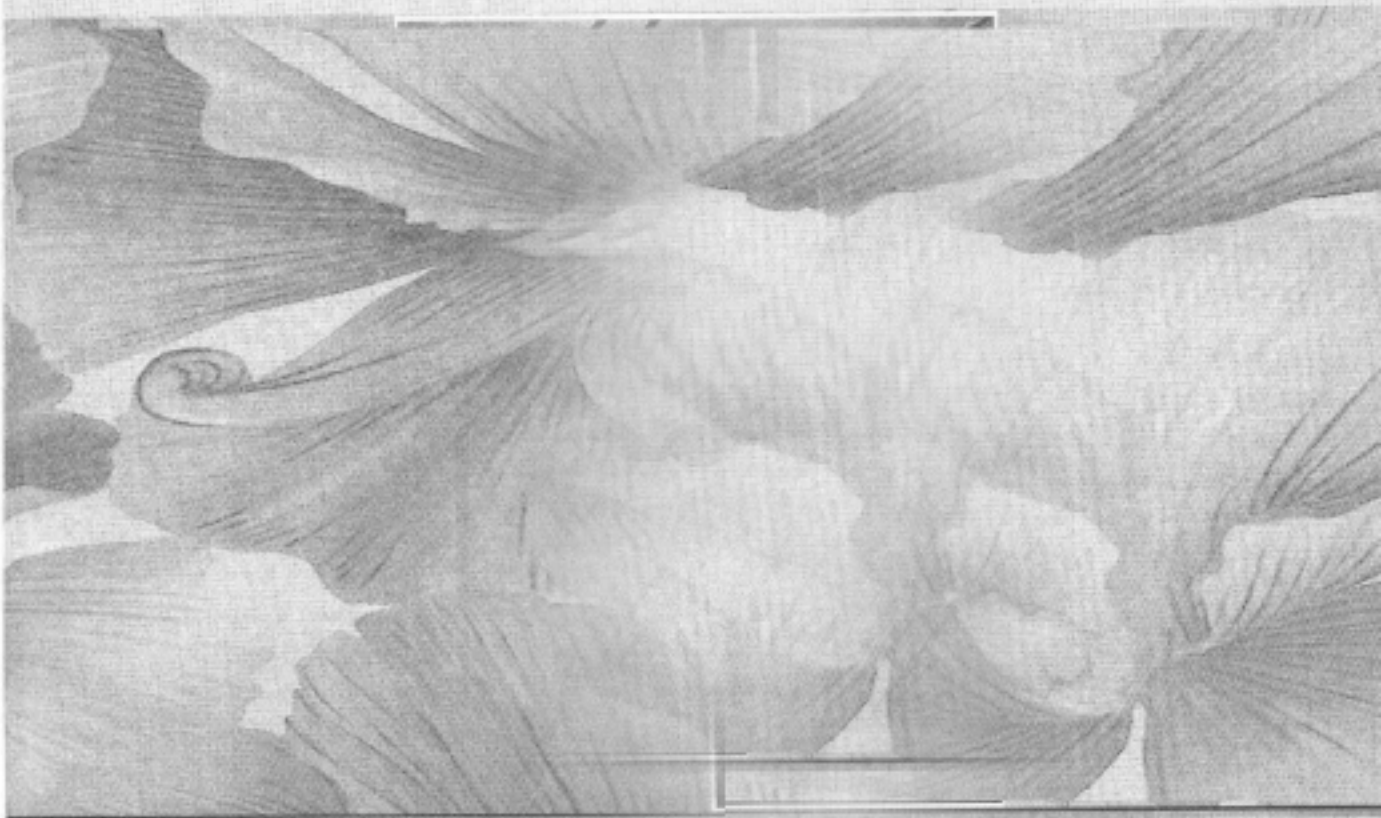
原书第3版

神经网络与机器学习

(加) Simon Haykin 著　申富饶 徐烨 郑俊 晁静 译

Neural Networks and Learning Machines

Third Edition



机械工业出版社
China Machine Press

本书是关于神经网络的全面的、彻底的、可读性很强的、最新的论述。全书共15章，主要内容包括Rosenblatt感知器、通过回归建立模型、最小均方算法、多层感知器、核方法和径向基函数网络、支持向量机、正则化理论、主分量分析、自组织映射、信息论学习模型、动态规划、神经动力学、动态系统状态估计的贝叶斯滤波等。

本书适合作为高等院校计算机相关专业研究生及本科生的教材，也可供相关领域的工程技术人员参考。

Simplified Chinese edition copyright © 2011 by Pearson Education Asia Limited and China Machine Press.

Original English language title: *Neural Networks and Learning Machines, Third Edition* (ISBN 978-0-13-147139-9) by Simon Haykin, Copyright © 2009.

All rights reserved.

Published by arrangement with the original publisher, Pearson Education, Inc., publishing as Prentice Hall.

本书封面贴有Pearson Education（培生教育出版集团）激光防伪标签，无标签者不得销售。

封底无防伪标均为盗版

版权所有，侵权必究

本书法律顾问　北京市展达律师事务所

本书版权登记号：图字：01-2009-1343

图书在版编目（CIP）数据

神经网络与机器学习（原书第3版）/（加）海金（Haykin, S.）著；申富饶等译. —北京：机械工业出版社，2011.1

（计算机科学丛书）

书名原文：Neural Networks and Learning Machines, Third Edition

ISBN 978-7-111-32413-3

Ⅰ. 神… Ⅱ. ①海… ②申… Ⅲ. ①人工神经-神经网络 ②机器学习 Ⅳ. TP18

中国版本图书馆CIP数据核字（2010）第213807号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）

责任编辑：迟振春

三河市明辉印装有限公司印刷

2011年3月第1版第1次印刷

185mm×260mm · 37.25印张

标准书号：ISBN 978-7-111-32413-3

定价：79.00元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

客服热线：（010）88378991；88361066

购书热线：（010）68326294；88379649；68995259

投稿热线：（010）88379604

读者信箱：hzjsj@hzbook.com

文艺复兴以降，源远流长的科学精神和逐步形成的学术规范，使西方国家在自然科学的各个领域取得了垄断性的优势；也正是这样的传统，使美国在信息技术发展的六十多年间名家辈出、独领风骚。在商业化的进程中，美国的产业界与教育界越来越紧密地结合，计算机学科中的许多泰山北斗同时身处科研和教学的最前线，由此而产生的经典科学著作，不仅擘划了研究的范畴，还揭示了学术的源变，既遵循学术规范，又自有学者个性，其价值并不会因年月的流逝而减退。

近年，在全球信息化大潮的推动下，我国的计算机产业发展迅猛，对专业人才的需求日益迫切。这对计算机教育界和出版界都既是机遇，也是挑战；而专业教材的建设在教育战略上显得举足轻重。在我国信息技术发展时间较短的现状下，美国等发达国家在其计算机科学发展的几十年间积淀和发展的经典教材仍有许多值得借鉴之处。因此，引进一批国外优秀计算机教材将对我国计算机教育事业的发展起到积极的推动作用，也是与世界接轨、建设真正的世界一流大学的必由之路。

机械工业出版社华章公司较早意识到“出版要为教育服务”。自 1998 年开始，我们就将工作重点放在了遴选、移译国外优秀教材上。经过多年的不懈努力，我们与 Pearson，McGraw-Hill，Elsevier，MIT，John Wiley & Sons，Cengage 等世界著名出版公司建立了良好的合作关系，从他们现有的数百种教材中甄选出 Andrew S. Tanenbaum，Bjarne Stroustrup，Brain W. Kernighan，Dennis Ritchie，Jim Gray，Afred V. Aho，John E. Hopcroft，Jeffrey D. Ullman，Abraham Silberschatz，William Stallings，Donald E. Knuth，John L. Hennessy，Larry L. Peterson 等大师名家的一批经典作品，以“计算机科学丛书”为总称出版，供读者学习、研究及珍藏。大理石纹理的封面，也正体现了这套丛书的品位和格调。

“计算机科学丛书”的出版工作得到了国内外学者的鼎力襄助，国内的专家不仅提供了中肯的选题指导，还不辞劳苦地担任了翻译和审校的工作；而原书的作者也相当关注其作品在中国的传播，有的还专程为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近两百个品种，这些书籍在读者中树立了良好的口碑，并被许多高校采用为正式教材和参考书籍。其影印版“经典原版书库”作为姊妹篇也被越来越多实施双语教学的学校所采用。

权威的作者、经典的教材、一流的译者、严格的审校、精细的编辑，这些因素使我们的图书有了质量的保证。随着计算机科学与技术专业学科建设的不断完善和教材改革的逐渐深化，教育界对国外计算机教材的需求和应用都将步入一个新的阶段，我们的目标是尽善尽美，而反馈的意见正是我们达到这一终极目标的重要帮助。华章公司欢迎老师和读者对我们的工作提出建议或给予指正，我们的联系方法如下：

华章网站：www. hzbook. com
电子邮件：hzjsj@hzbook. com
联系电话：(010) 88379604
联系地址：北京市西城区百万庄南街 1 号
邮政编码：100037



华章教育
华章科技图书出版中心

译者序

Neural Networks and Learning Machines, 3E

从20世纪40年代M-P神经元模型的提出开始，神经网络的发展过程可谓是一波三折。1965年M. Minsky和S. Papert的《感知机》使得神经网络的研究停滞了超过10年，直到20世纪80年代初Hopfield网络和误差反向传播算法等的提出，神经网络的研究才步入恢复期。时至今日，神经网络系统研究的重要意义已经得到广泛承认，在模式识别、人工智能、通信、控制、金融、机器人、生物信息学等许多领域都有广泛应用。可以说神经网络作为目前非线性科学和计算智能研究的主要内容之一，已经成为解决很多实际问题的一种必要的技术手段。

本书作者Simon Haykin长期从事神经网络的研究，其关于神经网络的系列教材是国际上最有影响力的教材之一。本书是其经典教材《Neural Networks：A Comprehensive Foundation》的第3版。正如本书的题目所示，这一版对神经网络和机器学习这两个密切相关的分支进行了全面分析，在前一版的基础上作了广泛修订，提供了神经网络和机器学习这两个重要性持续增长的学科的最新分析。本书全面、系统地介绍了神经网络的基本模型、基本方法，对神经网络的基本模型和主要学习理论作了深入研究，对神经网络的最新发展趋势和主要研究方向进行了全面而综合的介绍。

在翻译过程中，译者常常为本书作者严谨的治学态度及本书博大精深的内容而赞叹不已。本书综合了诸多神经网络和机器学习的最新研究，在翻译过程中虽然力求准确地反映原著内容，但由于译者水平有限，翻译中如有错漏之处，恳请读者批评指正。

本书的翻译得到了国家自然科学基金的资助（项目编号60975047），特此表示感谢。同时，感谢参与本书翻译的全体人员，没有他们的辛勤工作，本书的中文译本是无法顺利完成的；感谢本书第2版（《神经网络原理》）的译者，在翻译过程中我们大量参考了第2版中文译本的内容；还要感谢南京大学计算机软件新技术国家重点实验室的支持。

南京大学计算机科学与技术系
计算机软件新技术国家重点实验室
申富饶

2010年10月于南京

在写这本经典书籍第 3 版的时候，我遵循了本书第 1 版的基本原则：写一本关于神经网络的全面的、彻底的、可读性很强的、最新的论述。

新版本更名为《神经网络与机器学习》，主要是为了反映以下两个事实：

1. 感知器、多层感知器、自组织映射及神经动力学，以及其他一些通常被看成是神经网络一部分的主题，这些内容源自人类大脑所激发的灵感。

2. 核方法，例如支持向量机和核主分量分析，这些内容源自统计学习理论。

虽然这两者之间的确有一些共同的基本概念和应用，但是在神经网络和机器学习的具体操作上存在一些微妙的差别。因而，如果将这两者放在同一个体系下共同研究，一些潜在的主题会变得更丰富，特别体现在以下方面：

- 将神经网络和机器学习的思想综合起来以完成更难的学习任务。这些学习任务往往是神经网络或者机器学习自身所无法解决的。
- 源自人类大脑的灵感往往会引起新的特别重要的新发现。

除此之外，本书的范围也有所扩大，提供了详细的动态规划和逐次状态估计，这两者各自都能够在一些重要方面影响强化学习和监督学习。

本书的组织

本书从导言部分开始，导言主要讲述了编写本书的动机，可作为后续章节的阅读基础。本书包括以下 6 个部分：

1. 第 1～4 章构成了本书的第一部分，主要介绍监督学习的一些经典方法。具体介绍如下：

- 第 1 章描述 Rosenblatt 感知器，重点介绍感知器收敛定理，以及在高斯环境下感知器和贝叶斯分类器的关系。
- 第 2 章讲述作为模型建立基础的最小二乘法，建立了在特定的高斯环境下这一方法和贝叶斯推理之间的关系。这一章还讨论了用于模式选择的最小描述长度（MDL）算法。
- 第 3 章讲述最小均方（LMS）算法及其收敛分析。其理论框架的分析揭示出两个原理：Kushner 直接法和朗之万（Langevin）方程（在非平衡态热力学中很著名）。

这三章通过对不同概念的介绍揭示了其共同特点：它们都是基于一个计算单元。更为重要的是，它们从各自的角度深入、细致地讨论了学习过程的深层知识——这一特征将在后续章节中进一步探讨。

第 4 章是关于多层感知器的，是 Rosenblatt 感知器的广义版本。这一相对比较长的章节包含如下主题：

- 反向传播算法、其优点和局限性，以及将其作为一个最优化方法来计算偏导数。
- 学习率的最优退火和自适应控制。
- 交叉验证。

- 卷积网络，来自于 Hubel 和 Wiesel 在视觉系统方面的开拓性研究。
- 将监督学习视为最优化问题，集中讨论共轭梯度法、拟牛顿法以及 Marquardt-Levenberg 算法。
- 非线性滤波。
- 最后，对于小规模和大规模学习问题作了对比。

2. 第二部分包括第 5 章和第 6 章，讨论了基于径向基函数（RBF）网络的核方法。

从某种意义上来说，第 5 章可以看做是对核方法的深入介绍。具体来说，这一章包括如下几个方面。

- 介绍 Cover 定理来作为对 RBF 网络的构造结构的理论证明。
- 描述相对简单的用于监督学习的两阶段混合过程，第一阶段基于聚类思想（即 K-均值算法）来计算隐藏层，第二阶段利用 LMS 或者最小二乘法来计算网络的线性输出层。
- 介绍核回归及其与 RBF 网络的关系。

第 6 章介绍支持向量机（SVM），通常这一方法被认为是一种监督学习方法。本质上 SVM 是一个两类分类器，本章中将包括如下几个主题：

- 定义在一对线性可分的两类之间最大分离边缘的条件。
- 当两个类是线性可分或者不可分时用来寻找最优超平面的二次最优化。
- 将 SVM 视为核机器，包含关于核欺骗和 Mercer 定理的讨论。
- SVM 的设计原理。
- ε-不敏感损失函数及其在回归问题最优化中的作用。
- 表示定理及希尔伯特空间构想和再生核希尔伯特空间构想（RKHS）的作用。

根据以上描述，很明显支持向量机的基本理论是建立在很强的数学背景之上的，因而 SVM 可以作为监督学习的一个具有强大计算能力的、一流的工具。

3. 本书第三部分只有一章——第 7 章。这一章介绍作为机器学习核心的正则化理论。本章将详细探讨如下几个主题：

- 建立在第 6 章讨论过的 RKHS 基础之上的 Tikhonov 经典正则化理论。这一理论隐含了一些深奥的数学概念：Tikhonov 泛函的 Fréchet 微分、Riesz 表示定理、Euler-Lagrange方程、Green 函数，以及多变量高斯函数。
- 广义 RBF 网络及其计算精确性的修正。
- 正则最小二乘估计，根据表示定理的再讨论。
- 正则化参数估计，利用 Wahba 的广义交叉验证概念。
- 半监督学习，利用有标签和无标签样本。
- 可微流形及其在流形正则化中的作用——设计半监督学习机的基础。
- 寻找用于半监督学习的 RBF 网络中高斯核函数的光谱图理论。
- 处理半监督核机器的广义表示定理。
- 用于计算 RBF 网络线性输出层的拉普拉斯正则最小二乘（LapRLS）算法。这里需要说明的是，当内在正则化参数（对应于无标签数据）衰减为 0 的时候，算法相应地衰减为通常的最小二乘法。

这一高度理论化的章节具有非常实际的重要意义。首先，它提供了关于监督学习机的正则化基础。其次，它打下了设计正则化半监督学习机的基础。

4. 第 8～11 章构成本书的第四部分，讨论非监督学习。从第 8 章开始介绍由神经生物学研究直接激发的自组织的四个原则。

1）自增强学习的 Hebb 假定。

2）单个神经元或者一组神经元的突触连接为了有限的资源而进行的竞争。

3）在胜利神经元及其邻居间的合作。

4）包含于输入数据中的结构信息（如冗余）。

这一章的主要主题包括三个方面：

- 原则 1）、2）和 4）应用于单个神经元，最大特征滤波的 Oja 规则来源于这些原则；通过自组织获得的结果是值得注意的，它包含了自底向上和自顶向下学习。其次，最大特征滤波思想被推广到主分量分析（PCA）中，用来对输入数据进行维数削减，其所得算法称为广义 Hebb 算法（GHA）。
- 本质上 PCA 是线性方法，因而其计算能力局限于二阶统计量。为了处理高阶统计量，核方法以类似于第 6 章支持向量机的相似方式应用于 PCA，但是和 SVM 的根本上的不同在于，核 PCA 是非监督方式。
- 遗憾的是，在处理自然图像的时候，核 PCA 从计算的角度变得很难操控。为了克服这一计算局限性，把 GHA 和核 PCA 结合起来组成一个新的在线非监督学习算法，称为核 Hebb 算法（KHA），这一方法可以用于图像去噪。

KHA 的产生是一个将机器学习的想法和来源于神经网络的补充想法结合起来的杰出例子，结合所产生的新算法克服了它们各自的实际局限性。

第 9 章介绍自组织映射（SOM），对自组织映射的开发遵从第 8 章介绍的自组织原则。从计算角度来说，自组织映射是一个简单的算法，而且具有内在的构造拓扑映射的强大能力，它包括如下一些有用的特性：

- 从空间上离散逼近输入空间，负责数据生成。
- 拓扑次序，在某种意义上神经元的空间位置在拓扑图上对应于输入空间中的特定特征。
- 输入输出密度匹配。
- 输入数据特征选择。

SOM 在实际中被广泛应用，构造上下文映射和分层次矢量量化被作为 SOM 运算能力的两个有说服力的例子。事实上，令人惊异的是，尽管 SOM 展示了多个有趣的特性并且能够解决很难的计算任务，但它依然缺少一个能用来最优化的目标函数。为了填补这一缺口，以提供改进拓扑映射的可能性，自组织映射采用了核方法。这一改进是通过引入一个熵函数作为目标函数并且最大化这个函数来实现的。我们再次看到了将来自于神经网络的思想和补充的核理论思想结合所带来的实际好处。

第 10 章探讨如何将来自于香农（Shannon）信息论的原则作为工具来实现非监督学习。这一个相对较长的章节从回顾香农信息论开始，重点讨论了熵、互信息、相对熵（KLD）等概念。这一回顾也包括系词（copula）的概念，遗憾的是这一概念几十年来没有被注意到。更重要的是，系词提供了对一对相关随机变量之间统计相关性的测量。在任何事件中，集中于将互信息作为目标函数，这一章建立了如下原则：

- 最大互信息原则，最大化神经系统的输入和输出之间的互信息；最大互信息和冗余减少之间有着很紧密的关系。
- Imax 原则，最大化由相关输入驱动的神经系统对的单一输出之间的互信息。
- Imin 原则，以一种和 Imax 原则相似的方式操作，但这里是最小化输出随机变量对之间的互信息。
- 独立分量分析（ICA）原则，提供一种很强的工具用于盲分离来自统计独立源信号的隐藏集合。当满足一定的操作条件时，ICA 原则将提供对源信号进行恢复的起源程序基础，用于恢复的信号来自于对源信号的线性混合变形的相应的观察集合。这里将介

绍两个特别的 ICA 算法。

1）自然梯度学习算法，除了拉伸和排列之外，通过最小化参数概率密度函数和相应的阶乘分布之间的 KLD 来解决 ICA 问题。

2）最大熵学习算法，最大化反混合输出的非线性变换版本的熵；这一算法通常被认为是 ICA 的最大化信息算法，也表现出拉伸和排列性质。

第 10 章还描述了另一个称为快速 ICA（FastICA）的重要的 ICA 算法，这一算法正如其名字那样，计算速度快。这一算法基于负熵的概念最大化对比函数，对比函数提供了对于随机变量的非高斯分布程度的测量。作为 ICA 的延续，本章继续描述了一种称为相关 ICA 的新算法，其开发是根据最大化信息和 Imax 原则的融合并经由连接函数的运用来完成的；相关 ICA 在采集调幅信号的混合物的包迹时非常有用。最后，第 10 章介绍了另一个来自于香农信息论的称为速率失真理论的概念，这一理论被用来开发这一章的最后一个概念：信息瓶颈。给定关于输入向量和（有关的）输出向量的连接分布，这一方法通过如下方式被构造为约束最优化问题：在两个信息量之间做一个权衡，一个信息量是关于输入的瓶颈向量中包含的信息，另一个信息量是关于输出的瓶颈向量中所包含的信息。这一章将利用信息瓶颈法来寻找数据表达的最优流形。

第 11 章讲述非监督学习的最后途径，利用源自统计力学的随机方法来实现。统计力学的研究和信息论密切相关。这一章从回顾 Helmholtz 自由能和熵概念（从统计力学意义上）开始，紧接着介绍马尔可夫链。然后介绍用于产生马尔可夫链的 Metropolis 算法，其转移概率将收敛到唯一的、稳定的分布。接下来以两个方面作为随机方法讨论的结束：一是用于全局最优化的模拟退火，二是 Gibbs 抽样，它可以作为 Metropolis 算法的特殊形式。有了手头这些统计力学的背景知识，就可以讲述 Boltzmann 机了，Boltzmann 机从历史上来说是文献中讨论的第一个多层学习机器。遗憾的是，Boltzmann 机的学习过程非常慢，特别是当隐藏神经元的数目很大的时候，因而其实用性是最主要的缺陷。人们提出了很多变种方法来克服 Boltzmann 机的缺点。其中到目前为止最成功的创新方法是深度信度网络，它明智地把下面的两个功能组合起来形成了一个高效的机器：

- 生成模型，无监督地一层一层自底向上学习所得结果。
- 推论，自顶向下学习所得结果。

最后，第 11 章讲述确定性退火来克服模拟退火极端的计算需求问题；确定性退火的问题在于其可能陷入局部极小点。

5. 到目前为止，本书集中精力讲述了构造用于监督学习、半监督学习和非监督学习的算法。第 12 章，作为本书下一个部分，是关于强化学习的。强化学习以一种在线方式发生，作为智能体（如机器人）与其周围的环境相互作用的结果。实际上，动态规划是强化学习的核心。相应地，第 15 章的前面部分用来介绍 Bellman 动态规划方法，然后用来证明两个广泛使用的强化学习方法：时序差分学习（TD）和 Q-学习，这两种方法能通过作为动态规划的特例推导得出。TD 学习和 Q-学习都是相对比较简单的在线强化学习算法，无需转移概率知识。然而，其实际应用局限于状态空间的维数处于中等程度的情况。在大规模动态系统中，维数灾难变得非常严重，使得不仅仅是动态规划，也包括其近似形式的 TD 学习和 Q-学习变得难以计算。为了克服这一严重的局限性，这一章描述了两个逼近动态规划的非直接方法：

- 线性方法，称为最小二乘策略评估（LSPV）算法。
- 非线性方法，利用神经网络（如多层感知器）作为通用逼近器。

6. 本书最后一部分包括第 13、14 和 15 章，讨论非线性反馈系统，特别强调递归神经网络：

1）第 13 章研究神经动力学，对稳定性问题给予了特别的关注。这一章介绍了 Lyapunov 直接法，这个方法包含两个定理，一个用来处理系统稳定性，另一个用来处理渐近稳定性。这一方法的核心是 Lyapunov 函数，通常来说能量函数就能满足这一函数的要求。有了这样的背景知识，就可以引出两种联想记忆模型：

- Hopfield 模型，这一模型的操作说明一个复杂的系统是能够产生简单的突现行为的。
- 盒中脑状态模型，它是聚类的基础。

第 13 章还讨论了混沌过程的特性及其动态重构的正则化过程。

2）第 14 章是关于贝叶斯滤波器的，贝叶斯滤波器至少从概念意义上提供了逐次状态估计算法的统一基础。这一章的发现总结为以下几点：

- 经典的线性高斯环境下的卡尔曼滤波器可以通过利用最小均方差准则来推导；在这一章最后的一个习题中，证明这样推导的卡尔曼滤波器是贝叶斯滤波器的特例。
- 平方根滤波用来克服卡尔曼滤波在实际应用中遇到的发散现象。
- 扩展卡尔曼滤波（EKF）用来解决动力系统中非线性属于软排序的情况；保持高斯假设。
- 以一个新的称为数值积分卡尔曼滤波器（CKF）的滤波器为例来证明贝叶斯滤波器的直接逼近形式。这里再次强调了保持高斯假设。
- 以粒子滤波器为例来证明贝叶斯滤波器的非直接逼近形式，粒子滤波器的实现能够调节非线性程度和非高斯程度。

卡尔曼滤波本质上是预测-改正机制，第 14 章接着描述“类卡尔曼滤波”在人类大脑的一定区域的可能作用。

本书第 15 章研究动态驱动的递归神经网络。这一章的开始部分讨论不同的递归网络结构（模型）及其计算能力，紧接着介绍训练递归网络的两个算法：通过时间的反向传播和实时递归学习。

遗憾的是，这两个方法都是基于梯度的，容易遭遇所谓的消失梯度（vanishing-gradient）问题。为减轻这一问题，本书较详细地讨论了利用非线性逐次状态估计，采用全新的方式来对递归网络进行监督训练。这里，对于扩展卡尔曼滤波器（简单，但是导数依赖）以及数值积分卡尔曼滤波器（导数自由，但是数学上更加复杂）作为监督学习的逐次状态估计器的优缺点进行了讨论。此外，还讨论了对递归网络来说唯一的自适应行为的出现以及利用自适应技巧来增强递归网络性能的潜在好处。

在本书不同部分出现的一个重要的主题是，将监督学习和半监督学习应用于大规模问题。这包括本书评论中所指出的这一主题还处于发展的初期阶段；更重要的是，本书还为这一问题的未来发展描述了四阶段过程。

本书特色

本书完整、详尽地讨论了各个主题，除此之外，本书还有以下几个截然不同的特色：

1. 第 1～7 章以及第 10 章包含计算机实验，涉及双月形态，为两类分类问题产生数据。实验涵盖了从简单的线性可分模式例子到困难的不可分模式例子。作为运行例子的双月形态，被用于第 1～7 章以及第 10 章，因而提供了一个用于研究和比较这 8 章中描述的算法的实验途径。

2. 针对第 8 章的主分量分析、第 9 章的 SOM 和核 SOM，以及第 15 章的利用 EKF 和 CKF 算法对 Mackay-Glass 吸引子进行动态重构等，也进行了计算机实验。

3. 给出了几个利用现实数据进行研究的例子：

- 第 7 章讨论了利用拉普拉斯 RLS 算法对美国邮政服务（USPS）数据进行半监督学习。
- 第 8 章讨论了如何将 PCA 应用于手写数字数据，并描述了如何对图像进行编码和去噪。
- 第 10 章利用稀疏传感编码和 ICA 对自然图像进行分析。
- 第 13 章利用正则 RBF 网络将动态重构应用于 Lorenz 吸引子。

第 15 章也包含了一节关于模型参照自适应控制系统的案例研究。

4. 每一章的最后都有注释和参考文献用于进一步学习，每章末尾还提供了习题，用来练习并丰富读者的专业知识。

本书的“术语”表也进行了扩充，包含了用于处理矩阵分析和概率论问题的方法学解释。

5. 本书所有图和表格的 PowerPoint 文件都可以提供给教师，可到华章网站（www.hzbook.com）下载。

我们尽了最大努力来使本书不犯错误，更重要的是，我们也尽力提高它的可读性。

Simon Haykin

于 Ancaster，Ontario

缩写

AR	autoregressive	自回归
BBTT	back propagation through time	通过时间的反向传播
BM	Boltzmann machine	Boltzmann 机
BP	back propagation	反向传播
b/s	bits per second	每秒比特率
BSB	brain-state-in-a-box	盒中脑状态
BSS	Blind source (signal) separation	盲源（信号）分离
cmm	correlation matrix memory	相关矩阵记忆
CV	cross-validation	交叉验证
DFA	deterministic finite-state automata	确定性有限状态自动机
EKF	extended Kalman filter	扩展卡尔曼滤波器
EM	expectation-maximization	期望最大化
FIR	finite-duration impulse response	有限时间冲击响应
FM	frequency-modulated (signal)	频率调制（信号）
GCV	generalized cross-validation	广义交叉验证
GHA	generalized Hebbian algorithm	广义 Hebb 算法
GSLC	generalized sidelobe canceler	广义旁瓣消除器
Hz	hertz	赫兹
ICA	independent-components analysis	独立分量分析
Infomax	maximum mutual information	最大互信息
Imax	variant of Infomax	最大互信息的变体
Imin	another variant of Infomax	最大互信息的另一个变体
KSOM	kernel self-organizing map	核自组织映射
KHA	kernel Hebbian algorithm	核 Hebb 算法
LMS	least-mean-square	最小均方
LR	likelihood ratio	似然比
LS	Least-squares	最小二乘

LS-TD	Least-squares, temporal-difference	最小二乘，时序差分
LTP	long-term potentiation	长期增强
LTD	lone-term depression	长期衰减
LRT	Likelihood ratio test	似然比测试
MAP	Maximum a posteriori	最大后验估计
MCA	minor-components analysis	次分量分析
MCMC	Markov Chain Monte Carlo	马尔可夫链蒙特卡罗
MDL	minimum description length	最小描述长度
MIMO	multiple input-multiple output	多输入多输出
ML	maximum likelihood	最大似然
MLP	multilayer perceptron	多层感知器
MRC	model reference control	模型参考控制
NARMA	nonlinear autoregressive moving average	非线性自回归滑动平均
NARX	nonlinear autoregressive with exogenous inputs	具有外部输入的非线性自回归
NDP	neuro-dynamic programming	神经动态规划
NW	Nadaraya-Watson (estimator)	Nadaraya-Watson（估计器）
NWKR	Nadaraya-Watson kernel regression	Nadaraya-Watson 核回归
OBD	optimal brain damage	最优脑损伤
OBS	optimal brain surgeon	最优脑外科
OCR	optical character recognition	光学字符识别
PAC	probably approximately correct	可能近似正确
PCA	principal-components analysis	主分量分析
PF	Particle Filter	粒子滤波器
pdf	probability density function	概率密度函数
pmf	probability mass function	概率质量函数
QP	quadratic programming	二次规划
RBF	radial basis function	径向基函数
RLS	recursive least-squares	递归最小二乘
RLS	regularized least-squares	正则最小二乘
RMLP	recurrent multilayer perceptron	递归多层感知器
RTRL	real-time recurrent learning	实时递归学习
SIMO	single input-multiple output	单输入多输出
SIR	sequential importance resampling	逐次重要重采样
SIS	sequential important sampling	逐次重要采样
SISO	single input-single output	单输入单输出
SNR	signal-to-noise ratio	信噪比
SOM	self-organizing map	自组织映射
SRN	simple recurrent network (also referred to as Elman's recurrent network)	简单递归网络（也称为 Elman 递归网络）

SVD	singular value decomposition	奇异值分解
SVM	support vector machine	支持向量机
TD	temporal difference	时序差分
TDNN	time-delay neural network	时延神经网络
TLFN	time-lagged feedforward network	时间滞后前馈网络
VC	Vapnik-Chervononkis (dimension)	Vapnik-Chervononkis (维数)
VLSI	very-large-scale integration	超大规模集成
XOR	exclusive OR	异或

重要的符号

a	action	动作
$\mathbf{a}^T\mathbf{b}$	inner product of vectors **a** and **b**	向量 **a** 和 **b** 的内积
$\mathbf{a}\mathbf{b}^T$	outer product of vectors **a** and **b**	向量 **a** 和 **b** 的外积
$\binom{l}{m}$	binomial coefficient	二项式系数
$A \cup B$	unions of A and B	A 和 B 的并集
B	inverse of temperature	温度的逆
b_k	bias applied to neuron k	神经元 k 的偏置
$\cos(\mathbf{a},\mathbf{b})$	cosine of the angle between vectors **a** and **b**	向量 **a** 和 **b** 夹角的余弦
$c_{u,v}(u,v)$	probability density function of copula	系词的概率密度函数
D	depth of memory	记忆深度
$D_{f\|g}$	Kullback-Leibler divergence between probability density functions f and g	概率密度函数 f 和 g 之间的 Kullback-Leibler 散度
$\tilde{\mathbf{D}}$	adjoint of operator **D**	算子 **D** 的伴随矩阵
E	energy function	能量函数
E_i	energy of state i in statistical mechanics	统计力学中状态 i 的能量
$\mathbb{E}$	statistical expectation operator	统计期望算子
$\langle E \rangle$	average energy	平均能量
exp	exponential	指数
$\mathscr{E}_{av}$	average squared error, or sum of squared errors	平均平方误差或平方误差和
$\mathscr{E}(n)$	instantaneous value of the sum of squared errors	平方误差和的瞬时值
$\mathscr{E}_{total}$	total sum of error squares	总平方误差和
F	free energy	自由能量
$\mathscr{F}^*$	subset (network) with minimum empirical risk	经验风险最小的子集 (网络)
$\mathbf{H}$	Hessian (matrix)	Hessian 矩阵
$\mathbf{H}^{-1}$	inverse of Hessian **H**	Hessian 矩阵 **H** 的逆
i	square root of -1, also denoted by j	-1 的平方根，亦记作 j
$\mathbf{I}$	identity matrix	单位矩阵
$\mathbf{I}$	Fisher's information matrix	Fisher 信息矩阵
J	mean-square error	均方误差

$\mathbf{J}$	Jacobian (matrix)	Jacobi 矩阵
$\mathbf{P}^{1/2}$	square root of matrix $\mathbf{P}$	矩阵 $\mathbf{P}$ 的方根
$\mathbf{P}^{T/2}$	transpose of square root of matrix $\mathbf{P}$	矩阵 $\mathbf{P}$ 的方根的转置
$\mathbf{P}n,n-1$	error covariance matrix in Kalman filter theory	卡尔曼滤波理论中的误差协方差矩阵
k_B	Boltzmann constant	Boltzmann 常数
log	logarithm	对数
$L(\mathbf{w})$	log-likelihood function of weight vector $\mathbf{w}$	权值向量 $\mathbf{w}$ 的对数似然函数
$\mathscr{L}(\mathbf{w})$	log-likelihood function of weight vector $\mathbf{w}$ based on a single example	单样本的权值向量 $\mathbf{w}$ 的对数似然函数
$\mathbf{M}_c$	controllability matrix	可控矩阵
$\mathbf{M}_o$	observability matrix	可观察矩阵
n	discrete time	离散时间
p_i	probability of state i in statistical mechanics	统计力学中状态 i 的概率
p_{ij}	transition probability from state i to state j	从状态 i 到状态 j 的转移概率
$\mathbf{P}$	stochastic matrix	随机矩阵
$P(e\|\mathscr{C})$	conditional probability of error e given that the input is drawn from class $\mathscr{C}$	从类$\mathscr{C}$中输入时误差 e 的条件概率
P_α^+	probability that the visible neurons of a Boltzmann machine are in state α, given that the network is in its clamped condition (i. e. , positive phase)	假设网络处于钳制条件（即正向阶段）时，Boltzmann 机的可见神经元状态为 α 的概率
P_α^-	probability that the visible neurons of a Boltzmann machine are in state α, given that the network is in its free-running condition (i. e. , negative phase)	假设网络处于自由运行条件（即负向阶段）时，Boltzmann 机的可见神经元状态为 α 的概率
$\hat{r}_x(j,k;n)$	estimate of autocorrelation function of $x_j(n)$ and $x_k(n)$	$x_j(n)$和 $x_k(n)$的自相关函数估计
$\hat{r}_{dx}(k;n)$	estimate of cross-correlation function of $d(n)$ and $x_k(n)$	$d(n)$和 $x_k(n)$的交叉相关函数估计
$\mathbf{R}$	correlation matrix of an input vector	输入向量的相关矩阵
t	continuous time	连续时间
T	temperature	温度
$\mathscr{T}$	training set (sample)	训练集（样本）
tr	operator denoting the trace of a matrix	表示矩阵迹的算子
var	variance operator	方差算子
$V(\mathbf{x})$	Lyapunov function of state vector $\mathbf{x}$	状态向量 $\mathbf{x}$ 的 Lyapunov 函数
v_j	induced local field or activation potential of neuron j	神经元 j 的诱导局部域或激活位势
$\mathbf{w}_o$	optimum value of synaptic weight vector	突触权值向量的最优值
w_{kj}	weight of synapse j belonging to neuron k	属于神经元 k 的突触 j 的突触权值
$\mathbf{w}^*$	optimum weight vector	最优权值向量
$\bar{\mathbf{x}}$	equilibrium value of state vector $\mathbf{x}$	状态向量 $\mathbf{x}$ 的平衡值
$\langle x_j \rangle$	average of state x_j in a "thermal" sense	"热" 意义下状态 x_j 的平均

$\hat{x}$	estimate of x, signified by the use of a caret (hat)	x 的估计，用加字符号^（帽符号）表示
$\|x\|$	absolute value (magnitude) of x	x 的绝对值（幅度）
x^*	complex conjugate of x, signified by asterisk as superscript	状态 x 的复共轭，用星号 * 作上标
$\|\mathbf{x}\|$	Euclidean norm (length) of vector $\mathbf{x}$	向量 $\mathbf{x}$ 的欧几里得范数（长度）
$\mathbf{x}^T$	transpose of vector $\mathbf{x}$, signified by the superscript T	向量 $\mathbf{x}$ 的转置，用上标 T 表示
z^{-1}	unit-time delay operator	单位时间延迟算子
Z	partition function	剖分函数
$\delta_j(n)$	local gradient of neuron j at time n	神经元 j 在时刻 n 的局部梯度
Δw	small change applied to weight w	权值 w 的微小改变
∇	gradient operator	梯度算子
∇^2	Laplacian operator	拉普拉斯算子
$\nabla_w J$	gradient of J with respect to w	J 关于 w 的梯度
$\nabla \cdot \mathbf{F}$	divergence of vector $\mathbf{F}$	向量 $\mathbf{F}$ 的散度
η	learning-rate parameter	学习率参数
κ	cumulant	累积量
μ	policy	策略
θ_k	threshold applied to neuron k (i. e., negative of bias b_k)	神经元 k 的阈值（即偏置 b_k 的负值）
λ	regularization parameter	正则化参数
λ_k	kth eigenvalue of a square matrix	方阵的第 k 个特征值
$\varphi_k(\cdot)$	nonlinear activation function of neuron k	神经元 k 的非线性激活函数
$\in$	symbol for "belongs to"	"属于" 符号
$\cup$	symbol for "union of"	"并" 符号
$\cap$	symbol for "intersection of"	"交" 符号
$*$	symbol for convolution	卷积符号
$+$	superscript symbol for pseudoinverse of a matrix	矩阵伪逆的上标符号
$+$	superscript symbol for updated estimate	更新估计的上标符号

开区间和闭区间

- 变量 x 的开区间 (a, b) 表示 $a<x<b$。
- 变量 x 的闭区间 $[a, b]$ 表示 $a\leqslant x\leqslant b$。
- 变量 x 的半闭半开区间 $[a, b)$ 表示 $a\leqslant x<b$；类似地，变量 x 的半开半闭区间 $(a, b]$ 表示 $a<x\leqslant b$。

最小和最大

- 符号 $\arg\min_{\mathbf{w}} f(\mathbf{w})$ 表示函数 $f(\mathbf{w})$ 关于变元向量 $\mathbf{w}$ 的最小值。
- 符号 $\arg\max_{\mathbf{w}} f(\mathbf{w})$ 表示函数 $f(\mathbf{w})$ 关于变元向量 $\mathbf{w}$ 的最大值。

术 语

Neural Networks and Learning Machines, 3E

记号Ⅰ：矩阵分析

标量：用小写斜体符号表示标量。

向量：用小写粗体符号表示向量。

向量被定义为一列标量。因而 m 维向量 $\mathbf{x}$ 和 $\mathbf{y}$ 的内积可以写成

$$\mathbf{x}^T\mathbf{y} = [x_1, x_2, \cdots, x_m]\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \sum_{i=1}^{m} x_i y_i$$

其中，上标 T 用来表示矩阵转置。其内积为标量，因而我们有

$$\mathbf{y}^T\mathbf{x} = \mathbf{x}^T\mathbf{y}$$

矩阵：用大写粗体符号表示矩阵。

矩阵相乘是通过行和列的相乘来计算的。为了说明这一点，考虑 $m \times k$ 的矩阵 $\mathbf{X}$ 和 $k \times l$ 的矩阵 $\mathbf{Y}$。这两个矩阵的乘积产生一个 $m \times l$ 的矩阵

$$\mathbf{Z} = \mathbf{XY}$$

更具体地说，矩阵 $\mathbf{Z}$ 的第 ij 个分量是通过矩阵 $\mathbf{X}$ 的第 i 行和矩阵 $\mathbf{Y}$ 的第 j 列相乘而得到的，这两者都由 k 个标量组成。

一对 m 维向量 $\mathbf{x}$ 和 $\mathbf{y}$ 的外积写成 $\mathbf{x}\mathbf{y}^T$，是一个 $m \times m$ 的矩阵。

记号Ⅱ：概率论

随机变量：用大写的斜体符号来表示随机变量。随机变量的样本值（即单次实现）用相应的小写斜体符号来表示。例如，我们用 X 来表示随机向量，而用 x 来表示其样本值。

随机向量：用大写的粗体符号来表示随机向量。相似地，随机向量的样本值用相应的小写粗体符号来表示。例如，我们用 $\mathbf{X}$ 来表示随机向量，而用 $\mathbf{x}$ 来表示其样本值。

随机变量 $\mathbf{X}$ 的概率密度函数（pdf）由 $p_{\mathbf{X}}(\mathbf{x})$来表示，这是关于样本值 $\mathbf{x}$ 的函数；其下标 $\mathbf{X}$ 是用来提示 pdf 是关于随机向量 $\mathbf{X}$ 的。

出版者的话
译者序
前言
缩写和符号
术语

第 0 章 导言 …… 1

0.1 什么是神经网络 …… 1
0.2 人类大脑 …… 4
0.3 神经元模型 …… 7
0.4 被看作有向图的神经网络 …… 10
0.5 反馈 …… 11
0.6 网络结构 …… 13
0.7 知识表示 …… 14
0.8 学习过程 …… 20
0.9 学习任务 …… 22
0.10 结束语 …… 27
注释和参考文献 …… 27

第 1 章 Rosenblatt 感知器 …… 28

1.1 引言 …… 28
1.2 感知器 …… 28
1.3 感知器收敛定理 …… 29
1.4 高斯环境下感知器与贝叶斯分类器的关系 …… 33
1.5 计算机实验：模式分类 …… 36
1.6 批量感知器算法 …… 38
1.7 小结和讨论 …… 39
注释和参考文献 …… 39
习题 …… 40

第 2 章 通过回归建立模型 …… 28

2.1 引言 …… 41
2.2 线性回归模型：初步考虑 …… 41
2.3 参数向量的最大后验估计 …… 42
2.4 正则最小二乘估计和 MAP 估计之间的关系 …… 46
2.5 计算机实验：模式分类 …… 47
2.6 最小描述长度原则 …… 48
2.7 固定样本大小考虑 …… 50
2.8 工具变量方法 …… 53
2.9 小结和讨论 …… 54
注释和参考文献 …… 54
习题 …… 55

第 3 章 最小均方算法 …… 56

3.1 引言 …… 56
3.2 LMS 算法的滤波结构 …… 56
3.3 无约束最优化：回顾 …… 58
3.4 维纳滤波器 …… 61
3.5 最小均方算法 …… 63
3.6 用马尔可夫模型来描画 LMS 算法和维纳滤波器的偏差 …… 64
3.7 朗之万方程：布朗运动的特点 …… 65
3.8 Kushner 直接平均法 …… 66
3.9 小学习率参数下统计 LMS 学习理论 …… 67
3.10 计算机实验Ⅰ：线性预测 …… 68
3.11 计算机实验Ⅱ：模式分类 …… 69
3.12 LMS 算法的优点和局限 …… 71
3.13 学习率退火方案 …… 72
3.14 小结和讨论 …… 73
注释和参考文献 …… 74

习题 …… 74

第 4 章　多层感知器 …… 77

4.1　引言 …… 77
4.2　一些预备知识 …… 78
4.3　批量学习和在线学习 …… 79
4.4　反向传播算法 …… 81
4.5　异或问题 …… 89
4.6　改善反向传播算法性能的试探法 … 90
4.7　计算机实验：模式分类 …… 94
4.8　反向传播和微分 …… 95
4.9　Hessian 矩阵及其在在线学习中的规则 …… 96
4.10　学习率的最优退火和自适应控制 … 98
4.11　泛化 …… 102
4.12　函数逼近 …… 104
4.13　交叉验证 …… 107
4.14　复杂度正则化和网络修剪 …… 109
4.15　反向传播学习的优点和局限 …… 113
4.16　作为最优化问题看待的监督学习 …… 117
4.17　卷积网络 …… 126
4.18　非线性滤波 …… 127
4.19　小规模和大规模学习问题 …… 131
4.20　小结和讨论 …… 136
注释和参考文献 …… 137
习题 …… 138

第 5 章　核方法和径向基函数网络 … 144

5.1　引言 …… 144
5.2　模式可分性的 Cover 定理 …… 144
5.3　插值问题 …… 148
5.4　径向基函数网络 …… 150
5.5　K-均值聚类 …… 152
5.6　权向量的递归最小二乘估计 …… 153
5.7　RBF 网络的混合学习过程 …… 156
5.8　计算机实验：模式分类 …… 157
5.9　高斯隐藏单元的解释 …… 158
5.10　核回归及其与 RBF 网络的关系 … 160
5.11　小结和讨论 …… 162
注释和参考文献 …… 164
习题 …… 165

第 6 章　支持向量机 …… 168

6.1　引言 …… 168
6.2　线性可分模式的最优超平面 …… 168
6.3　不可分模式的最优超平面 …… 173
6.4　使用核方法的支持向量机 …… 176
6.5　支持向量机的设计 …… 178
6.6　XOR 问题 …… 179
6.7　计算机实验：模式分类 …… 181
6.8　回归：鲁棒性考虑 …… 184
6.9　线性回归问题的最优化解 …… 184
6.10　表示定理和相关问题 …… 187
6.11　小结和讨论 …… 191
注释和参考文献 …… 192
习题 …… 193

第 7 章　正则化理论 …… 197

7.1　引言 …… 197
7.2　良态问题的 Hadamard 条件 …… 198
7.3　Tikhonov 正则化理论 …… 198
7.4　正则化网络 …… 205
7.5　广义径向基函数网络 …… 206
7.6　再论正则化最小二乘估计 …… 209
7.7　对正则化的附加要点 …… 211
7.8　正则化参数估计 …… 212
7.9　半监督学习 …… 215
7.10　流形正则化：初步的考虑 …… 216
7.11　可微流形 …… 217
7.12　广义正则化理论 …… 220
7.13　光谱图理论 …… 221
7.14　广义表示定理 …… 222
7.15　拉普拉斯正则化最小二乘算法 … 223
7.16　用半监督学习对模式分类的实验 …… 225
7.17　小结和讨论 …… 227
注释和参考文献 …… 228
习题 …… 229

第 8 章　主分量分析 …… 232

8.1　引言 …… 232

8.2 自组织原则 …… 232
8.3 自组织的特征分析 …… 235
8.4 主分量分析：扰动理论 …… 235
8.5 基于 Hebb 的最大特征滤波器…… 241
8.6 基于 Hebb 的主分量分析 …… 247
8.7 计算机实验：图像编码 …… 251
8.8 核主分量分析 …… 252
8.9 自然图像编码中的基本问题 …… 256
8.10 核 Hebb 算法 …… 257
8.11 小结和讨论 …… 260
注释和参考文献 …… 262
习题 …… 264

第 9 章 自组织映射 …… 268

9.1 引言 …… 268
9.2 两个基本的特征映射模型 …… 269
9.3 自组织映射 …… 270
9.4 特征映射的性质 …… 275
9.5 计算机实验Ⅰ：利用 SOM 解网格动力学问题 …… 280
9.6 上下文映射 …… 281
9.7 分层向量量化 …… 283
9.8 核自组织映射 …… 285
9.9 计算机实验Ⅱ：利用核 SOM 解点阵动力学问题 …… 290
9.10 核 SOM 和相对熵之间的关系 … 291
9.11 小结和讨论 …… 293
注释和参考文献 …… 294
习题 …… 295

第 10 章 信息论学习模型 …… 299

10.1 引言 …… 299
10.2 熵 …… 300
10.3 最大熵原则 …… 302
10.4 互信息 …… 304
10.5 相对熵 …… 306
10.6 系词 …… 308
10.7 互信息作为最优化的目标函数 … 310
10.8 最大互信息原则 …… 311
10.9 最大互信息和冗余减少 …… 314
10.10 空间相干特征 …… 316
10.11 空间非相干特征 …… 318
10.12 独立分量分析 …… 320
10.13 自然图像的稀疏编码以及与 ICA 编码的比较 …… 324
10.14 独立分量分析的自然梯度学习 …… 326
10.15 独立分量分析的最大似然估计 …… 332
10.16 盲源分离的最大熵学习 …… 334
10.17 独立分量分析的负熵最大化 … 337
10.18 相关独立分量分析 …… 342
10.19 速率失真理论和信息瓶颈 …… 347
10.20 数据的最优流形表达 …… 350
10.21 计算机实验：模式分类 …… 354
10.22 小结和讨论 …… 354
注释和参考文献 …… 356
习题 …… 361

第 11 章 植根于统计力学的随机方法 …… 366

11.1 引言 …… 366
11.2 统计力学 …… 367
11.3 马尔可夫链 …… 368
11.4 Metropolis 算法 …… 374
11.5 模拟退火 …… 375
11.6 Gibbs 抽样 …… 377
11.7 Boltzmann 机 …… 378
11.8 logistic 信度网络 …… 382
11.9 深度信度网络…… 383
11.10 确定性退火 …… 385
11.11 和 EM 算法的类比 …… 389
11.12 小结和讨论 …… 390
注释和参考文献 …… 390
习题 …… 392

第 12 章 动态规划 …… 396

12.1 引言 …… 396
12.2 马尔可夫决策过程 …… 397
12.3 Bellman 最优准则 …… 399

12.4 策略迭代 …… 401
12.5 值迭代 …… 402
12.6 逼近动态规划：直接法 …… 406
12.7 时序差分学习 …… 406
12.8 Q-学习 …… 410
12.9 逼近动态规划：非直接法 …… 412
12.10 最小二乘策略评估 …… 414
12.11 逼近策略迭代 …… 417
12.12 小结和讨论 …… 419
注释和参考文献 …… 421
习题 …… 422

第 13 章 神经动力学 …… 425

13.1 引言 …… 425
13.2 动态系统 …… 426
13.3 平衡状态的稳定性 …… 428
13.4 吸引子 …… 432
13.5 神经动态模型 …… 433
13.6 作为递归网络范例的吸引子操作 …… 435
13.7 Hopfield 模型 …… 435
13.8 Cohen-Grossberg 定理 …… 443
13.9 盒中脑状态模型 …… 445
13.10 奇异吸引子和混沌 …… 448
13.11 混沌过程的动态重构 …… 452
13.12 小结和讨论 …… 455
注释和参考文献 …… 457
习题 …… 458

第 14 章 动态系统状态估计的贝叶斯滤波 …… 461

14.1 引言 …… 461
14.2 状态空间模型 …… 462
14.3 卡尔曼滤波器 …… 464
14.4 发散现象及平方根滤波 …… 469
14.5 扩展的卡尔曼滤波器 …… 474
14.6 贝叶斯滤波器 …… 477
14.7 数值积分卡尔曼滤波器：基于卡尔曼滤波器 …… 480
14.8 粒子滤波器 …… 484
14.9 计算机实验：扩展的卡尔曼滤波器和粒子滤波器对比评价 …… 490
14.10 大脑功能建模中的卡尔曼滤波 …… 493
14.11 小结和讨论 …… 494
注释和参考文献 …… 496
习题 …… 497

第 15 章 动态驱动递归网络 …… 501

15.1 引言 …… 501
15.2 递归网络体系结构 …… 502
15.3 通用逼近定理 …… 505
15.4 可控性和可观测性 …… 507
15.5 递归网络的计算能力 …… 510
15.6 学习算法 …… 511
15.7 通过时间的反向传播 …… 512
15.8 实时递归学习 …… 515
15.9 递归网络的消失梯度 …… 519
15.10 利用非线性逐次状态估计的递归网络监督学习框架 …… 521
15.11 计算机实验：Mackay-Glass 吸引子的动态重构 …… 526
15.12 自适应考虑 …… 527
15.13 实例学习：应用于神经控制的模型参考 …… 529
15.14 小结和讨论 …… 530
注释和参考文献 …… 533
习题 …… 534

参考文献 …… 538

导　言

0.1　什么是神经网络

自从认识到人脑计算与传统的数字计算机相比是完全不同的方式开始，关于人工神经网络（一般称为“神经网络”（neural network））的研究工作就开始了。人脑是一个高度复杂的、非线性的和并行的计算机器（信息处理系统）。人脑能够组织它的组成成分，即神经元，以比今天已有的最快的计算机还要快许多倍的速度进行特定的计算（如模式识别、感知和发动机控制）。例如，考虑人类视觉，这是一个信息处理任务。视觉系统的功能是为我们提供一个关于周围环境的表示，并且更重要的是提供我们与环境交互（interact）所需的信息。具体来说，完成一个感知识别任务（例如识别一张被嵌入陌生场景的熟悉的脸）人脑大概需要100～200毫秒，而一台高效的计算机却要花费比人脑多很多的时间才能完成一个相对简单的任务。

再举一个例子：考虑一只蝙蝠的声呐。声呐就是一个活动回声定位系统。除了提供目标（例如飞行的昆虫）有多远的信息外，蝙蝠的声呐可以搜集目标的相对速度、目标大小、目标不同特征的大小以及它的方位角和仰角的信息。所有这些信息都从目标回声中提取，而所有需要的复杂神经计算只在李子般大小的脑中完成。事实上，一只回声定位的蝙蝠可以灵巧地以很高的成功率追逐和捕捉目标，这一点足以使雷达或声呐工程师们自叹弗如。

那么，人脑或蝙蝠的脑是如何做到这一点的呢？脑在出生的时候就有很复杂的构造和具有通过我们通常称为的“经验”来建立它自己规则的能力。确实，经验是经过时间积累的，人脑在出生后头两年内发生了非常大的进化（即硬接线），但是进化将超越这个阶段并继续进行。

一个“进化中”的神经系统是与可塑的大脑同义的。可塑性（plasticity）允许进化中的神经系统适应（adapt）其周边环境。可塑性似乎是人类大脑中作为信息处理单元的神经元功能的关键，同样，它在人工神经元组成的神经网络中亦是如此。最普通形式的神经网络，就是对人脑完成特定任务或感兴趣功能所采用的方法进行建模的机器。网络一般用电子元件实现或者用软件在数字计算机上模拟。在本书中，我们集中介绍一类重要的神经网络，这类网络通过学习过程来实现有用的计算。为了获得良好性能，神经网络使用一个很庞大的简单计算单元间的相互连接，这些简单计算单元称为“神经元”或者“处理单元”。据此我们给出将神经网络看作一种自适应机器的定义[1]：

神经网络是由简单处理单元构成的大规模并行分布式处理器，天然地具有存储经验知识和使之可用的特性。神经网络在两个方面与大脑相似：

1. 神经网络是通过学习过程从外界环境中获取知识的。
2. 互连神经元的连接强度，即突触权值，用于存储获取的知识。

用于完成学习过程的程序称为学习算法，其功能是以有序的方式改变网络的突触权值以获得想要的设计目标。

对突触权值的修改提供了神经网络设计的传统方法。这种方法和线性自适应滤波器理论很接近，而滤波器理论已经很好地建立起来并被成功地应用在很多领域（Widrow and Stearns，1985；Haykin，2002）。但是，受人脑的神经元会死亡以及新的突触连接会生长的事实所启发，神经网络修改它自身的拓扑结构也是可能的。

神经网络的优点

很明显，神经网络的计算能力可通过以下两点得到体现：第一，神经网络的大规模并行分布式结构；第二，神经网络的学习能力以及由此而来的泛化能力。*泛化*（generalization）是指神经网络对未在训练（学习）过程中遇到的数据可以得到合理的输出。这两种信息处理能力让神经网络可以找到一些当前难以处理的复杂（大规模）问题的好的近似解。但是在实践中，神经网络不能单独做出解答，它们需要被整合在一个协调一致的系统工程方法中。具体来说，一个复杂问题往往被分解成若干个相对简单的任务，而神经网络处理与其能力相符的子任务。但是，我们在建立一个可以模拟人脑的计算机结构（如果可能）之前还有很长的路要走，认识这一点是很重要的。

神经网络具有下列有用的性质和能力：

1. *非线性*（nonlinearity）：人工神经元可以是线性或者非线性的。由非线性神经元互相连接而成的神经网络自身是非线性的，并且从某种特别意义上来说非线性是分布于整个网络中的。非线性是一个非常重要的特性，特别是当产生输入信号（如语音信号）的内部物理机制是天生非线性的时候。

2. *输入输出映射*（input-output mapping）：称之为*有教师学习*（learning with a teacher）或*监督学习*（supervised learning）的关于学习的流行方法。它使用带标号的*训练样例*（training example）或*任务样例*（task example）对神经网络的突触权值进行修改。每个样例由一个唯一的*输入信号*（input signal）和相应的*期望*（*目标*）*响应*（desired（target）response）组成。从一个训练集中随机选取一个样例提供给网络，网络就调整它的突触权值（自由参数），以最小化期望响应和由输入信号以适当的统计准则产生的网络实际响应之间的差别。使用训练集中的很多样例来重复训练神经网络，直到网络达到对突触权值没有显著修正的稳定状态为止。先前已经使用过的训练样例可能还要在训练期间以不同顺序重复使用。因此对当前问题来说，神经网络是通过建立输入输出映射来从样例中学习的。这样的方法使人想起了*非参数统计推断*（nonparametric statistical inference）的研究，它是非模型估计统计处理的一个分支，或者从生物学角度看，称为*白板学习*（tabula rasa learning，Geman 等，1992）。这里使用“非参数”这一术语表示的一个事实是，没有对输入数据的统计模型作任何先验假设。比如，考虑一个*模式分类*（pattern classification）任务，这里的要求是把代表具体物体或事件的输入信号分类到几个预先分好的类中去。关于这一问题的非参数方法中，要求利用样本集“估计”输入信号空间中模式分类任务的任意决策边界，并且不使用概率分布模型。而监督学习方法也隐含了类似的观点，这就提示在神经网络的输入输出映射和非参数统计推断之间存在相近的类比。

3. *自适应性*（adaptivity）：神经网络具有调整自身突触权值以适应外界环境变化的固有能力。特别是，一个在特定运行环境下接受训练的神经网络，在环境条件变化不大的时候可以很容易地进行重新训练。而且，当它在一个*不稳定*（nonstationary）环境（即它的统计特性随时间变化）中运行时，可以设计神经网络使得其突触权值随时间实时变化。用于模式分类、信号处理和控制的神经网络与它的自适应能力相耦合，就可以变成能进行自适应模式分类、自适应信号处理和自适应控制的有效工具。作为一般规则，在保证系统保持稳定时，一个系统的自适应性越好，它被要求在一个不稳定环境下运行时其性能就越具鲁棒性。但是，需要强调的是，自适应性不一定总能导致鲁棒性，实际还可能导致相反结果。比如，一个短时常数自适应系统可能变化过快，以至于对干扰扰动有所反应，从而引起系统性能的急剧恶化。为了获得自适应性的最大好处，系统的主要时间常数应该长到可以忽略干扰扰动，却依然足够短以能反应环境的重要变化。这一问题通常被称为*稳定性-可塑性困境*（Grossberg，1988）。

4. *证据响应*（evidential response）：在模式分类问题中，神经网络可以设计成不仅提供选

择哪一个特定模式的信息，还提供关于决策的置信度信息。后者可以用来拒判那些可能出现的过于模糊的模式，从而进一步改善网络的分类性能。

5. *上下文信息*（contextual information）：神经网络的特定结构和激发状态代表知识。网络中每一个神经元都受网络中所有其他神经元全局活动的潜在影响。因此，神经网络将很自然地能够处理上下文信息。

6. *容错性*（fault tolerance）：一个以硬件形式实现的神经网络具有天生的容错性，或者说具有鲁棒计算的能力，在这种意义上其性能在不利的运行条件下是逐渐下降的。比如，一个神经元或它的连接损坏了，存储模式的记忆性在质量上会被削弱。但是，由于网络信息存储的分布特性，在网络的总体响应严重恶化之前这种损坏是分散的。因此，原则上，神经网络从性能上显示了一个缓慢恶化的过程而不是灾难性的失败。有一些关于鲁棒性计算的经验证据，但通常它是不可控的。为了确保网络事实上的容错性，有必要在设计训练网络的算法时采用正确的度量（Kerlirzin and Vallet，1993）。

7. *VLSI 实现*（VLSI implementability）：神经网络的大规模并行性使它具有快速处理某些任务的潜在能力。这一特性使得神经网络很适合使用超大规模集成（very-large-scale-integrated，VLSI）技术来实现。VLSI 的一个特殊优点是可以提供一个以高度分层的方式来捕捉真实复杂行为的方法（Mead，1989）。

8. *分析和设计的一致性*：基本上，神经网络作为信息处理器具有通用性。我们这样说是因为涉及神经网络应用的所有领域都使用同样的记号。这一特征以不同的方式表现出来：

- 神经元，不管形式如何，在所有的神经网络中都代表一种相同成分。
- 这种共性使得在不同应用中的神经网络共享相同的理论和学习算法成为可能。
- 模块化网络可以用*模块的无缝集成*来实现。

9. *神经生物类比*：神经网络的设计是由与人脑的类比引发的，人脑是一个容错的并行处理的实例，说明这种处理不仅在物理上是可实现的，而且还是快速、高效的。神经生物学家将（人工）神经网络看作是一个解释神经生物现象的研究工具。另一方面，工程师对神经生物学的关注在于将其作为解决复杂问题的新思路，这些问题比基于常规的硬件线路设计技术所能解决的问题更复杂。下面两个例子说明了这两种观点：

- 在 Anastasio（1993）中，将*前庭视觉反射*（vestibulo-ocular reflex，VOR）的线性系统模型和基于在 0.6 节描述及第 15 章中详细讲述的*递归网络*的神经网络模型进行了比较。前庭视觉反射是眼球运动系统的一部分，其作用是让眼球向与头转动方向相反的方向运动，以维持视觉（视网膜）图像的稳定性。VOR 由前庭核酸的前端神经元调节，前端神经元从前庭感知神经元中接受头部旋转信息并加以处理，将结果告知眼球肌肉的动作神经元。输入（头部旋转信息）和输出（眼球旋转）可以精确确定，因此 VOR 很适合建模。另外，它是比较简单的反射作用，并且其组成神经元的神经生理学的内容已经被很好地阐述过了。在三种神经类型中，前端神经元（反射内层神经元）在前庭核酸中是最复杂、也是最引人注意的。VOR 以前已经用集块线性系统描述器和控制理论模型化了。这些模型对解释 VOR 的整体性质有一些作用，但是对了解其组成神经元特性却用处不大。这种情况通过建模神经网络已经被大大改善了。VOR 的递归网络模型（使用第 15 章描述的实时递归学习算法设计）能通过调节 VOR 的神经元（特别是前庭核酸神经元）重现和解释处理信号时的静态、动态、非线性和分布式等多方面特性。
- *视网膜*不同于人脑的其他任何部分，是我们开始将外部环境的物理图像投射到一行接收器上形成的视觉表示和第一个神经图像相结合的地方。它是眼球后部的神经组织薄

层，其功能是将光学图像转换成神经图像并沿光神经传输给大量的视觉中枢以便进一步处理。这是一个复杂的工作，可以从视网膜的突触组织得到证明。在脊椎动物的视网膜中，光图像转化成神经图像的过程由三个阶段组成（Sterling，1990）。

1）受体神经元层的图像传导。

2）结果信号（产生于对光刺激的反应）由化学性突触传输给一层双极细胞。

3）同样，由化学性突触把结果信号传给神经节细胞的输出神经元。

在两个突触阶段（即从受体到双极细胞和从双极细胞到神经节细胞），有专门侧向连接的神经元，分别称为*水平细胞的神经元*和*无长突细胞的神经元*。这些神经元的工作是修改突触层之间的传输。另外还有称为*中间网状细胞的离心元素*，它们的工作是将信号从内部突触层传到外部突触层。一些研究人员已经建立了模拟视网膜结构的电子芯片。这些电子芯片称为*神经形态*（neuromorphic）集成电路，这个术语由 Mead（1989）所创造。神经形态的图像传感器是由一排感光器与每个图形元素（像素）的模拟回路结合而成的。它能模拟视网膜适应局部的亮度变化、检测边缘以及检测运动。神经生物学模拟（例如神经形态集成电路）有另一个重要的应用：它提供了一种希望和信念，并在一定程度上提供一种存在性证明，即对神经生物结构的物理上的了解对电子学工艺和超大规模集成电路技术有多方面的影响。

有了神经生物学的启示，我们对人脑及其组织的结构层次作简要的考察看来是合适的。[2]

0.2 人类大脑

人的神经系统可看作三阶段系统，如同图 1（Arbib，1987）所描绘的框图所示。系统的中央是人脑，由神经网络表示，它持续地接收信息，感知它并做出适当的决定。图中有两组箭头，从左到右的箭头表示携带信息的信号通过系统向前传输，从右到左的箭头表示系统中的反馈。感受器把来自人体或外界环境的刺激转换成电冲击，对神经网络（大脑）传送信息。神经网络的效应器会将神经网络产生的电冲击转换为可识别的响应从而作为系统的输出。

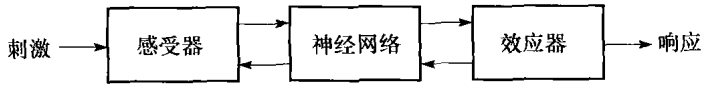


图 1 神经系统的框图

在 Ramón y Cajál（1911）的开创性工作中，引入神经元作为人脑结构成分的思想，从而使得人们理解人脑的努力变得简单多了。通常，神经元比硅逻辑门要慢 5 到 6 个数量级；硅逻辑门中的事件发生在纳秒级，而在神经中的事件发生在毫秒级。但是人脑是由运行速度相对较慢的神经元所构成的，神经元（神经细胞）的数目非常惊人，而且它们之间具有大量的互连。据估计人的大脑皮层中有大约 100 亿个神经元和大约 60 万亿个突触或连接（Shepherd and Koch，1990）。这些数据说明大脑拥有非常高效的结构。具体来说，脑的能量效率为每秒每个操作大约为 10^{-16} 焦耳，而今天所用的最好计算机的相应值则远远大于人脑。

突触（synapse）或称之为*神经末梢*（nerve ending），是调节神经元之间相互作用的基本结构和功能单位。最普通的一类突触是*化学突触*，它是这样运行的：前突触过程释放*发送器物质*，扩散到神经元之间的突触连接，然后作用于后突触过程。这样突触就完成了突触前端的电信号向化学信号的转换，然后转换回突触后端电信号（Shepherd and Koch，1990）。用电学术语来说，这样的元素称为*非互逆的两端口设备*。在传统的神经组织描述中，仅假设突触是一个简单的连接，能施加*兴奋*或*抑制*，但不同时作用在接受神经元。

我们曾提到过，可塑性允许进化神经系统以适应周边环境（Eggermont，1990；Churchland and Sejnowski，1992）。在成年人的大脑中，可塑性可以解释两个机能：创建神经元间的

新连接以及修改已有的连接。轴突（即传导线路）和树突（即接受区域）组成两种细胞长纤维，它们在形态上互相区别。轴突有光滑的表面、较少的分支、比较长，而树突正相反（之所以这样称呼是因为它和树相似），它有不规则的表面和更多的分支（Freeman，1975）。脑中的不同部分有很多种形状和大小不同的神经元。图 2 是一种锥形细胞，它在大脑皮层中最常见。与其他许多神经元一样，它从树突刺接收大部分输入信号；可以从图 2 中看到树突片段的细节。锥形细胞可以有一万个或更多的突触与其他细胞连接，它可以投射到数以千计的目标细胞。

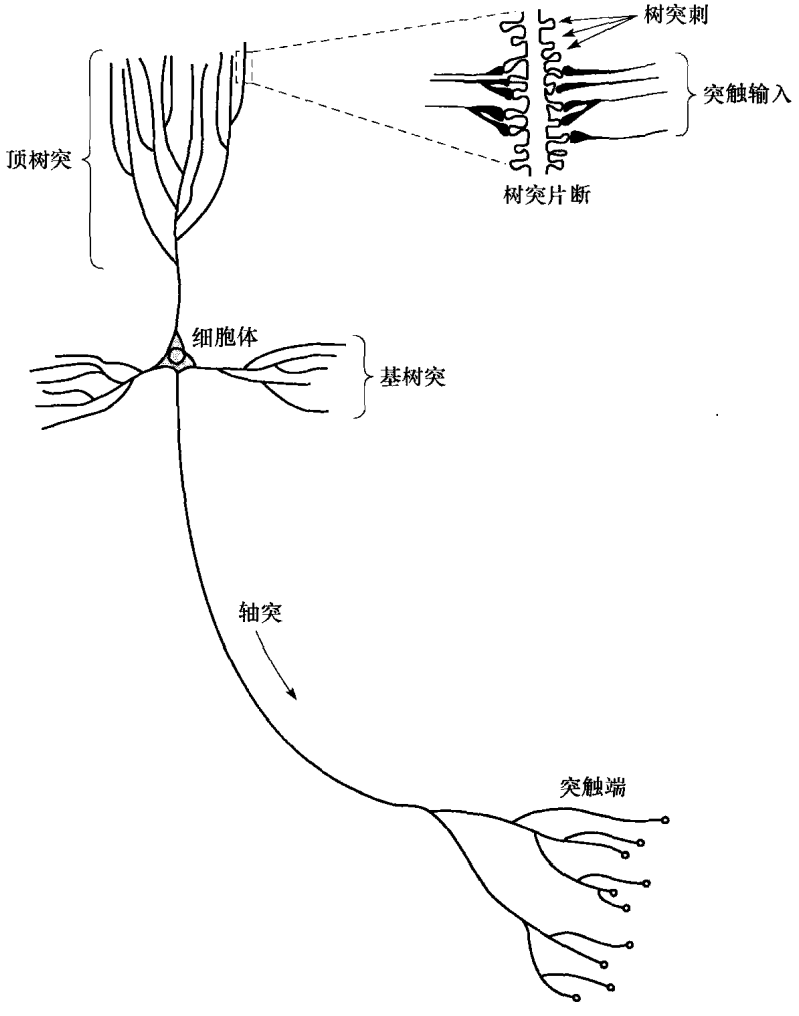


图 2 锥形细胞

大多数神经元把它们的输出转化成一系列简短的电压脉冲编码。这些脉冲，一般称为动作电位或尖峰[3]，产生于神经元细胞体或其附近并以恒定的电压和振幅穿越个体神经元。使用神经元间的动作电位通信是由轴突的物理性质决定的。轴突很长很细，有很高的电阻和非常大的电容，这两者分布于轴突中。因此可以用 RC 传输线路来建模，用“电缆方程”这个术语来描述轴突中的信号传播。对传播机制的分析揭示了电压在传输中随距离呈指数衰减，在到达另一端时会变得很小。动作电位提供了克服这个问题的方法（Anderson，1995）。

在人脑中，有小规模和大规模解剖组织之分，在底层和高层会发生不同的机能。图 3 显示了脑组织各种级别交织的层次结构，这已经在广泛的关于脑局部区域的分析工作中显现出来（Shepherd and Koch，1990；Churchland and Sejnowski，1992）。突触表示最基本的层次，其

活动依赖于分子和离子。其后的层次有神经微电路、树突树以及神经元。神经微电路指突触集成，组织成可以产生所需的功能操作的连接模式。它就像一个由晶体管集成的硅片，最小的尺寸用微米（μm）度量，最快的操作速度用毫秒（mm）度量。神经微电路被组织成属于神经元个体的树突树的树突子单元。整个神经元大约为 100μm 大小，包含几个树突子单元。局部电路（大约 1mm 大小）处在其次的复杂性水平，由具有相似或不同性质的神经元组成，这些神经元集成完成脑局部区域的特征操作。接下来是区域间电路，由通路、柱子和局部解剖图组成，牵涉脑中不同部分的多个区域。

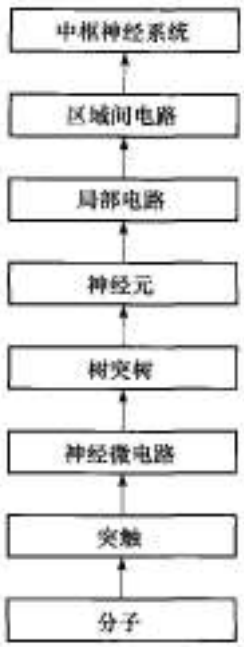


图 3 大脑的分层结构组织

局部解剖图（topographic map）被组织成用来响应输入的感知信息。它们经常被组织成片束状，如同在上丘中一样。上丘中视觉、听觉和人体触觉区以层邻接的方式放置，使得空间中相应点的刺激处于各层的下面或上面。图 4 表示由 Brodmann（Brodal，1981）做出的大脑皮层的细胞结构图。图中清晰表明不同的感知输入（运动、触觉、视觉、听觉等）被有序地映射到大脑皮层的相应位置。在复杂性的最后一级，局部解剖图和其他的区域间电路成为中央神经系统传递特定行为的媒介。

认识到在这里描绘的结构分层组织是大脑的独有特征非常重要。我们在数字计算机中找不到这种结构，在人工神经网络中也无法近似地重构它们。但是，我们仍在向图 3 中描述的类似的分级计算层状结构缓慢推进。用以构造神经网络的人工神经元和人脑中的神经元相比确实比较初级，我们目前能设计的网络和人脑中初级的局部电路和区域间电路相当，但是，真正令人满意的是我们已经在许多前沿有了显著进步。以神经生物类比作为灵感的源泉，加上我们具有的理论和技术工具等财富，逐步地，我们对人工神经网络及其应用的理解一定会更加深入和宽广。

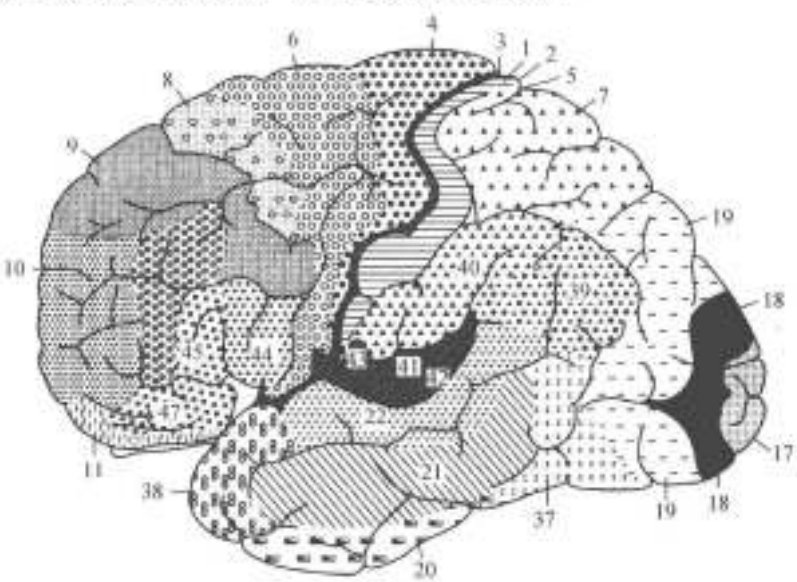


图 4 大脑皮层细胞结构图。不同区域由它们的层厚度及其内部细胞类型标示。一些最重要的感知区域如下。运动皮层：运动区，区域 4；前运动区，区域 6；前端眼球区，区域 8。人体触觉皮层：区域 3，1，2。视觉皮层：区域 17，18，19。听觉皮层：区域 41，42（摘自 A. Brodal，1981；经 Oxford University Press 许可）

0.3 神经元模型

神经元是神经网络操作的基本信息处理单位。图5给出了神经元的模型，它是后续章节中将要探讨的设计（人工）神经网络大家庭的基础。我们在这里给出神经元模型的三种基本元素：

1. *突触或连接链集*，每一个都由其权值或者强度作为特征。具体来说，在连到神经元 k 的突触 j 上的输入信号 x_j 被乘以 k 的突触权值 w_{kj}。注意突触权值 w_{kj} 下标的写法很重要。第一个下标指正在研究的这个神经元，第二个下标指权值所在的突触的输入端。和人脑中的突触不一样，人工神经元的突触权值有一个范围，可以取正值也可以取负值。

2. *加法器*，用于求输入信号被神经元的相应突触加权的和。这个操作构成一个线性组合器。

3. *激活函数*，用来限制神经元输出振幅。由于它将输出信号压制（限制）到允许范围之内的一定值，故而激活函数也称为*压制函数*。通常，一个神经元输出的正常幅度范围可写成单位闭区间 [0，1] 或者另一种区间 [−1，+1]。

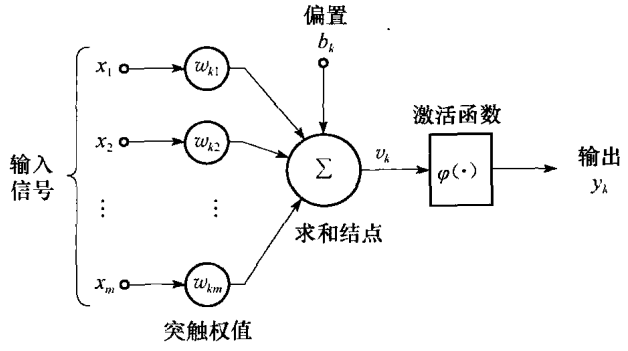


图5 神经元的非线性模型，标记为第 k 个神经元

图5的神经元模型也包括一个外部偏置（bias），记为 b_k。偏置 b_k 的作用是根据其为正或为负，相应地增加或降低激活函数的网络输入。

用数学术语来表示，我们可以用如下一对方程描述图5中的神经元 k：

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \tag{1}$$

$$y_k = \varphi(u_k + b_k) \tag{2}$$

其中 $x_1, x_2, \cdots, x_m$ 是输入信号，$w_{k1}, w_{k2}, \cdots, w_{km}$ 是神经元 k 的突触权值，u_k（没有在图5中标出）是输入信号的*线性组合器的输出*，b_k 为偏置，*激活函数*为 $\varphi(\cdot)$，y_k 是神经元输出信号。偏置 b_k 的作用是对图5模型中的线性组合器的输出 u_k 作*仿射变换*（affine transformation），如下所示：

$$v_k = u_k + b_k \tag{3}$$

特别地，根据偏置 b_k 取正或取负，神经元 k 的诱导局部域（induced local field）或*激活电位*（activation potential）v_k 和线性组合器输出 u_k 的关系如图6所示。以后我们将把“诱导局部域”和“激活电位”这两个术语交替使用。注意到由于这个仿射变换的作用，v_k 与 u_k 的图形不再经过原点。

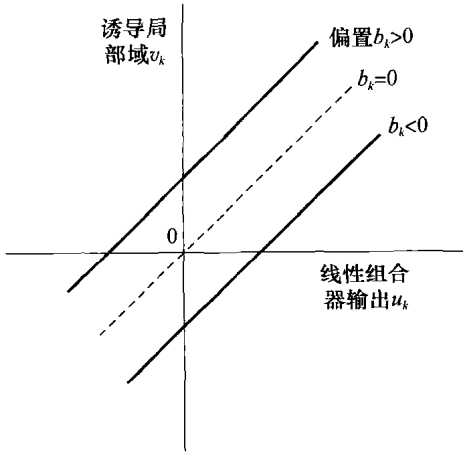


图6 偏置产生的仿射变换，注意 $u_k=0$ 时 $v_k=b_k$

偏置 b_k 是人工神经元 k 的外部参数。我们可以像在式(2)中一样考虑它。同样，可以结合式(1)和式(3)得到如下公式：

$$v_k = \sum_{j=0}^{m} w_{kj} x_j \tag{4}$$

$$y_k = \varphi(v_k) \tag{5}$$

在式(4)中，我们加上一个新的突触，其输入是

$$x_0 = +1 \tag{6}$$

权值是

$$w_{k0} = b_k \tag{7}$$

因此得到了神经元 k 的新模型，如图 7 所示。在这个图中，偏置起两种作用：(1) 添加新的固定输入+1；(2) 添加新的等于偏置 b_k 的突触权值。虽然形式上图 5 和图 7 的模型不相同，但在数学上它们是等价的。

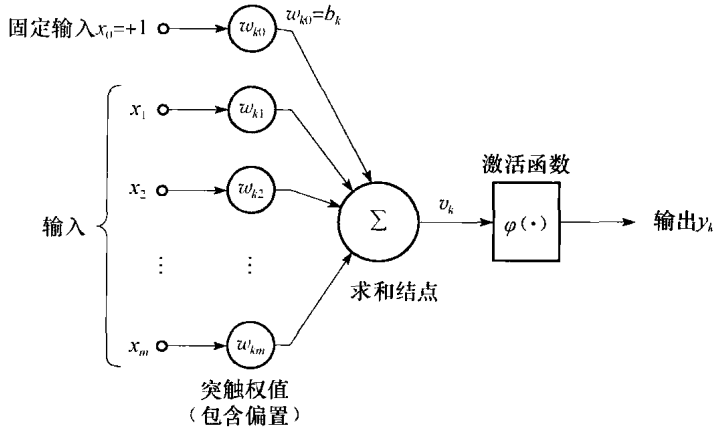


图 7 神经元的另一个非线性模型，w_{k0} 代替了偏置 b_k

激活函数的类型

激活函数，记为 $\varphi(v)$，通过诱导局部域 v 定义神经元输出。这里我们给出两种基本的激活函数：

1. *阈值函数*。这种激活函数如图 8a 所示，可写为：

$$\varphi(v) = \begin{cases} 1 & \text{如果 } v \geqslant 0 \\ 0 & \text{如果 } v < 0 \end{cases} \tag{8}$$

在工程文献中，这种函数一般称为 Heaviside 函数。相应地，在神经元 k 上使用这种阈值函数，其输出可表示为

$$y_k = \begin{cases} 1 & \text{如果 } v_k \geqslant 0 \\ 0 & \text{如果 } v_k < 0 \end{cases} \tag{9}$$

其中 v_k 是神经元的诱导局部域，即

$$v_k = \sum_{j=1}^{m} w_{kj} x_j + b_k \tag{10}$$

在神经计算中，这样的神经元在文献中称为 McCulloch-Pitts 模型，以纪念 McCulloch and Pitts (1943) 的开拓性工作。在模型中，如果神经元的诱导局部域非负，则输出为 1，否则为 0。这描述了 McCulloch-Pitts 模型的*皆有或者皆无特性* (all-or-none property)。

2. *sigmoid 函数*。[4] 此函数的图形是“S”形的，在构造人工神经网络中是最常用的激活函

数。它是严格的递增函数，在线性和非线性行为之间显现出较好的平衡。sigmoid 函数的一个例子是 logistic 函数[5]，定义如下：

$$\varphi(v) = \frac{1}{1+\exp(-av)} \tag{11}$$

其中 a 是 sigmoid 函数的倾斜参数。修改参数 a 就可以改变倾斜程度，如图 8b 所示。实际上，在原点的斜度等于 $a/4$。在极限情况下，倾斜参数趋于无穷，sigmoid 就变成了简单的阈值函数。阈值函数仅取值 0 或 1，而 sigmoid 的值域是 0 到 1 的连续区间。还要注意到 sigmoid 函数是可微分的，而阈值函数不是。(如第 4 章所述，可微性是神经网络理论的一个重要特征。)

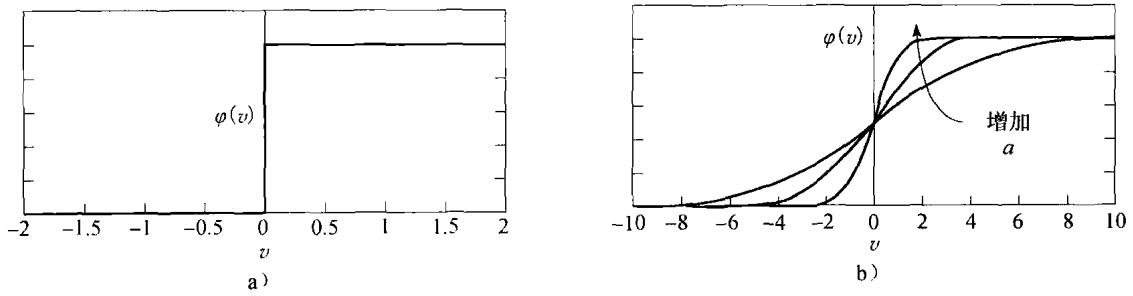


图 8 a) 阈值函数；b) 具有不同倾斜参数 a 的 sigmoid 函数

在式(8)、(11)中定义的激活函数的值域是 0 到+1。有时也期望激活函数的值域是−1 到+1，这种情况下激活函数是诱导局部域的奇函数。具体来说，阈值函数(8)的另一种形式是

$$\varphi(v) = \begin{cases} 1 & \text{如果 } v > 0 \\ 0 & \text{如果 } v = 0 \\ -1 & \text{如果 } v < 0 \end{cases} \tag{12}$$

通常称之为 signum 函数。为了与 sigmoid 函数相对应，我们可以使用双曲正切函数

$$\varphi(v) = \tanh(v) \tag{13}$$

如式(13)所示，它允许 sigmoid 型的激活函数取负值，这有时候会产生比式(11)的 logistic 函数更好的实际利益。

神经元的统计模型

图 7 的神经元模型是确定性的，它的输入输出行为对所有的输入精确定义。但在一些神经网络的应用中，基于随机神经模型的分析更符合需要。使用一些解析处理方法，McCulloch-Pitts 模型的激活函数用概率分布来实现。具体来说，一个神经元允许有两个可能的状态值+1 或−1。一个神经元激发（即它的状态开关从“关”到“开”）是随机决定的。用 x 表示神经元的状态，$P(v)$表示激发的概率，其中 v 是诱导局部域。我们可以设定

$$x = \begin{cases} +1 & \text{概率为 } P(v) \\ -1 & \text{概率为 } 1-P(v) \end{cases} \tag{14}$$

一个标准选择是 sigmoid 型的函数：

$$P(v) = \frac{1}{1+\exp(-v/T)} \tag{15}$$

其中 T 是伪温度（pseudotemperature），用来控制激发中的噪声水平即不确定性（Little, 1974)。但是，不管神经网络是生物的或人工的，T 都不是神经网络的物理温度，认识到这一点很重要。进一步，正如所说明的一样，我们仅仅将 T 看作是一个控制表示突触噪声效果的热波动参数。注意当 T 趋于 0 时，式(14)和式(15)所描述的随机神经元就变为无噪声（即确

定性）形式，也就是 McCulloch-Pitts 模型。

0.4 被看作有向图的神经网络

图 5 或图 7 的方框图提供了构成人工神经元模型各个要素的功能描述。我们可以在不牺牲模型功能细节的条件下用信号流图来简化模型外观。Mason（1953，1956）开发了线性网络的一套信号流图，并带有定义好的规则。神经元的非线性限制了它们在神经网络中的应用范围。不过，信号流图在描述神经网络信号流时为我们提供了简洁的方法，我们在本节进行讨论。

信号流图是一个由在一些特定的称为节点的点之间相连的有向连接（分支）组成的网络。一个典型的节点 j 有一个相应的节点信号 x_j。一个典型的有向连接从节点 j 开始，到 k 节点结束。它有相应的传递函数或传递系数以确定节点 k 的信号 y_k 依赖于节点 j 的信号 x_j 的方式。图形中各部分的信号流动遵循三条基本规则。

规则 1 信号仅仅沿着定义好的箭头方向在连接上流动。

两种不同类型的连接可以区别开来：

- 突触连接，它的行为由线性输入输出关系决定。具体来说，如图 9a 所示，节点信号 y_k 由节点信号 x_j 乘以突触权值 w_{kj} 产生。
- 激活连接，它的行为一般由非线性输入输出关系决定。如图 9b 所示，其中 $\varphi(\cdot)$ 为非线性激活函数。

规则 2 节点信号等于经由连接进入的有关节点的所有信号的代数和。

这个规则通过如图 9c 所示的突触会聚或扇入的情形来说明。

规则 3 节点信号沿每个外向连接向外传递，此时传递的信号完全独立于外向连接的传递函数。

第三个规则通过如图 9d 所示的突触散发或扇出的情形来说明。

比如，利用这些规则，我们可以制作出对应于图 7 的信号流图 10 来作为神经元模型。可以看出，图 10 要比图 7 的形式更简单，但是它包含了后者描绘的所有功能细节。注意，在两个图中，输入 $x_0=+1$ 和相关的突触权值 $w_{k0}=b_k$，其中 b_k 是神经元 k 的偏置。

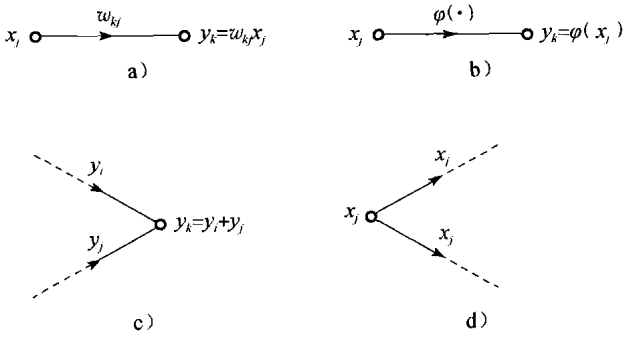


图 9 用于构造信号流图的基本规则图示

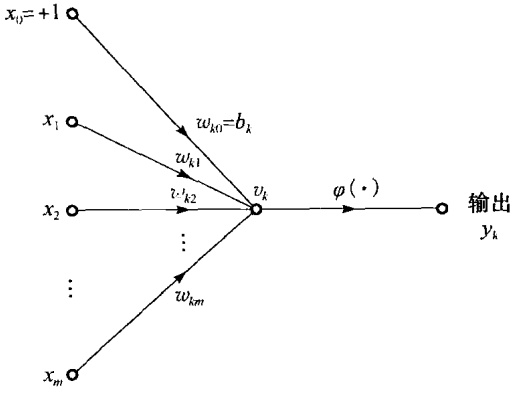


图 10 神经元的信号流图

确实，根据图 10 的信号流图所显示的神经元模型，我们可以给出一个神经网络的下列数学定义：

神经网络是由具有互相连接的突触节点和激活连接构成的有向图，具有 4 个主要特征：

1. 每个神经元可表示为一组线性的突触连接，一个外部应用偏置，以及可能的非线性激活连接。偏置由和一个固定为 $+1$ 的输入连接的突触连接表示。
2. 神经元的突触连接给它们相应的输入信号加权。

3. 输入信号的加权和构成该神经元的诱导局部域。

4. 激活连接压制神经元的诱导局部域产生输出。

一个如此定义的有向图是完全的，这是指它不仅仅描述了神经元间的信号流，也描述了每个神经元内部的信号流。但是当我们的注意集中在神经元之间的信号流上时，可以使用这个图的一个简略形式，它省略神经元内部的信号流的细节。这样的有向图是局部完全的，它的特征是：

1. 源节点向图提供输入信号。

2. 每个神经元由称为计算节点的单个节点表示。

3. 联结图中源节点和计算节点之间的通信连接没有权值，它们仅仅提供图中信号流的方向。

这样定义的一个局部完全的有向图就是所谓的神经网络结构图，描述神经网络的布局。图 11 给出了具有 m 个源节点和一个用于偏置的、固定为+1 的节点的单一神经元的简单情况。注意表示该神经元的计算节点以阴影显示，而源节点用小方块显示。在本书中，我们都遵循这里的表示方法。在 0.6 节有更精巧的布局结构图的例子。

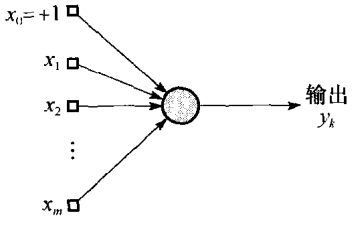


图 11 神经元的结构图

总的来说，我们有三种神经网络的图形表示方法：

- 方框图，提供网络的功能描述；
- 结构图，描述网络布局；
- 信号流图，提供网络中完全的信号流描述。

0.5 反馈

当系统中一个元素的输出能够部分地影响作用于该元素的输入，从而造成一个或多个围绕该系统进行信号传输的封闭路径时，我们说动态系统中存在着反馈（feedback）。实际上，反馈存在于所有动物神经系统的几乎每一部分中（Freeman，1975）。并且，在一类特殊的神经网络——递归网络的研究中扮演着重要角色。图 12 表示单环反馈系统的信号流图，输入信号 $x_j(n)$、内部信号 $x_j'(n)$和输出信号 $y_k(n)$是离散时间变量 n 的函数。这个系统被假定为线性的，由“算子” **A** 表示的前向通道和“算子” **B** 表示的反馈通道组成。特别地，前向通道的输出通过反馈通道来部分地影响自己的输出。由图 12 可以很容易得到这样的输入输出关系：

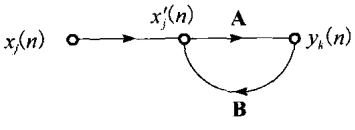


图 12 单环反馈系统的信号流图

$$y_k(n) = \mathbf{A}[x_j'(n)] \tag{16}$$

$$x_j'(n) = x_j(n) + \mathbf{B}[y_k(n)] \tag{17}$$

其中方括号是为了强调 **A** 和 **B** 是扮演着算子的角色。在式(16)和式(17)中消去 $x_j'(n)$，得到

$$y_k(n) = \frac{\mathbf{A}}{1-\mathbf{AB}}[x_j(n)] \tag{18}$$

我们把 **A**/(1−**AB**)称为系统的闭环算子，**AB** 称为开环算子。通常，开环算子没有交换性，即 **BA**≠**AB**。

例如，考虑图 13a 中的单环反馈系统。**A** 是一个固定的权值 w，**B** 是单位时间延迟算子 z^{-1}，其输出是输入延迟一个时间单位的结果。我们可以将这个系统的闭环算子表示为

$$\frac{\mathbf{A}}{1-\mathbf{AB}} = \frac{w}{1-wz^{-1}} = w(1-wz^{-1})^{-1}$$

将 $(1-wz^{-1})^{-1}$二项式展开，可以把系统的闭环算子重写为

$$\frac{\mathbf{A}}{1-\mathbf{AB}} = w\sum_{l=0}^{\infty} w^l z^{-l} \tag{19}$$

因此，将式(19)代入式(18)，我们有

$$y_k(n) = w\sum_{l=0}^{\infty} w^l z^{-l}[x_j(n)] \tag{20}$$

其中，再次用方括号强调 z^{-1} 是算子的事实。特别地，由 z^{-1} 的定义我们有

$$z^{-l}[x_j(n)] = x_j(n-l) \tag{21}$$

其中 $x_j(n-l)$ 是输入信号延迟 l 个时间单位的样本。因此，可以用输入 $x_j(n)$ 现在和过去所有样本的无限加权和来表示输出 $y_k(n)$：

$$y_k(n) = \sum_{l=0}^{\infty} w^{l+1} x_j(n-l) \tag{22}$$

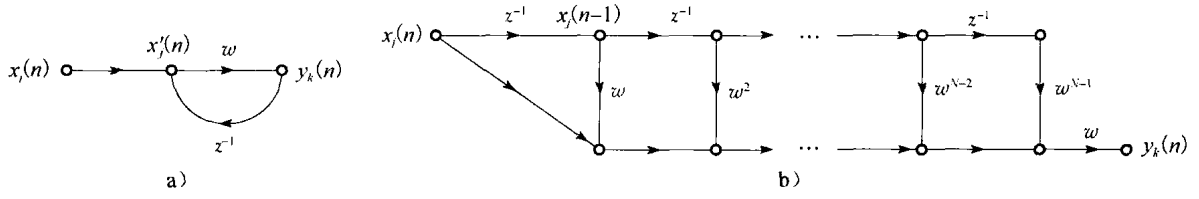


图 13 a) 一阶无限冲击响应（IIR）滤波器的信号流图；b) 图中 a) 部分的前馈近似，通过切断式(20)得到

我们现在清楚地看到由图 13 的信号流图表示的反馈系统的动态行为是由权值 w 控制的。特别是，我们可以识别两种特殊情况：

1. $|w|<1$，此时输出信号 $y_k(n)$ 以指数收敛；也就是说，系统是稳定的。如图 14a 对一个正 w 值的情况所示。

2. $|w|\geqslant 1$，此时输出信号 $y_k(n)$ 发散；也就是说，系统是不稳定的。图 14b 是 $|w|=1$ 的情况，发散是线性的；图 14c 是 $|w|>1$ 的情况，发散是指数的。

稳定性是闭环反馈系统研究中的突出特征。

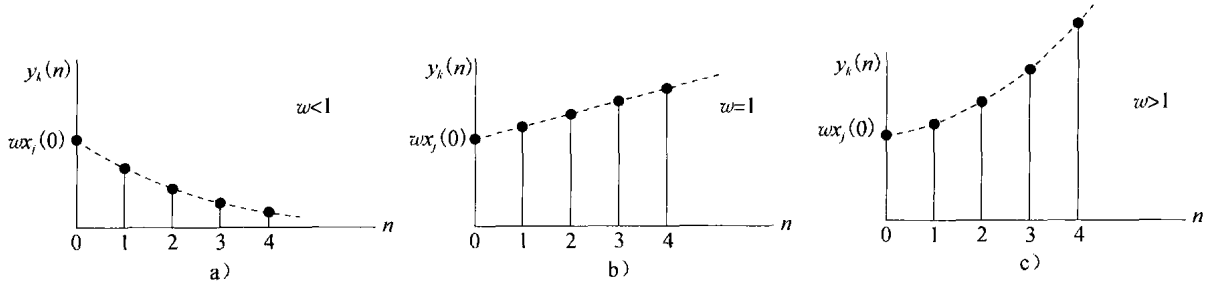


图 14 图 13 中前馈权值 w 的三种不同值的时间响应：a) 稳定；b) 线性发散；c) 指数发散

$|w|<1$ 的情况对应于具有无限记忆的系统，这是指系统的输出依赖于无限过去的输入样本。并且，过去的样本对记忆的影响是随时间 n 呈指数衰减的。假设对任意的幂 N，$|w|$ 相对于数 1 足够小以保证对任何实际目的来说 w^N 是可以忽略的。在这种情况下，可以通过下面的有限和来逼近输出 y_k：

$$y_k(n) \approx \sum_{l=0}^{N-1} w^{l+1} x_j(n-l) = wx_j(n) + w^2 x_j(n-1) + w^3 x_j(n-2) + \cdots + w^N x_j(n-N+1)$$

相应地，可以利用图 13b 所示的前馈信号流图作为图 13a 的反馈信号流图的逼近。在实现这样的逼近的时候，我们称为反馈系统的“伸展”。然而，必须说明的是，仅在反馈系统稳定的时候伸展操作才有实际价值。

由于用于构造神经网络的处理单元通常是非线性的，因此它所涉及的反馈应用的动态行为

分析都很复杂。这一点在本书后面会给出进一步分析。

0.6 网络结构

神经网络中神经元的构造方式与用于训练网络的学习算法有着紧密的联系。因此，我们可以说，用于神经网络设计的学习算法（规则）是被构造的。我们将在0.8节讨论学习算法的分类。这一节我们专注于网络的体系结构。

一般说来，我们可以区分三种基本不同的网络结构。

单层前馈网络

在分层网络中，神经元以层的形式组织。在最简单的分层网络中，源节点构成输入层，直接投射到神经元输出层（计算节点）上，反之则不然。也就是说，这个网络是严格前馈的。如图15所示，输出输入层各有4个节点。这样的网络称为单层网。单层指的是计算节点（神经元）输出层。我们不把源节点的输入层计算在内，因为在这一层没有进行计算。

多层前馈网络

前馈神经网络的第二种网络有一层或多层隐藏层，相应的计算节点称为隐藏神经元或隐藏单元。隐藏是指神经网络的这一部分无论从网络的输入端或者输出端都不能直接看到。隐藏神经元的功能是以某种有用的方式介入外部输入和网络输出之中。通过增加一个或多个隐藏层，网络可以根据其输入引出高阶统计特性。即使网络为局部连接，由于额外的突触连接和额外的神经交互作用，也可以使网络在不十分严格的意义下获得一个全局关系（Churchland and Sejnowski，1992）。

网络输入层的源节点提供激活模式的元素（输入向量），组成第二层（第一隐藏层）神经元（计算节点）的输入信号。第二层的输出信号作为第三层输入，这样一直传递下去。通常，每一层的输入都是上一层的输出，最后的输出层给出相对于源节点的激活模式的网络输出。结构图如图16所示，图中只有一个隐藏层以简化神经网络的布局。这是一个10－4－2网络，其中有10个源节点，4个隐藏神经元，2个输出神经元。作为另外一个例子，具有 m 个源节点的前馈网络，第一个隐藏层有 h_1 个神经元，第二个隐藏层有 h_2 个神经元，输出层有 q 个神经元，可以称为 $m-h_1-h_2-q$ 网络。

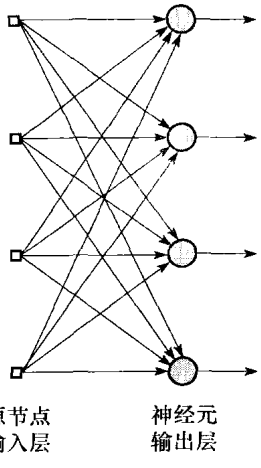


图15 单层神经元前馈网络

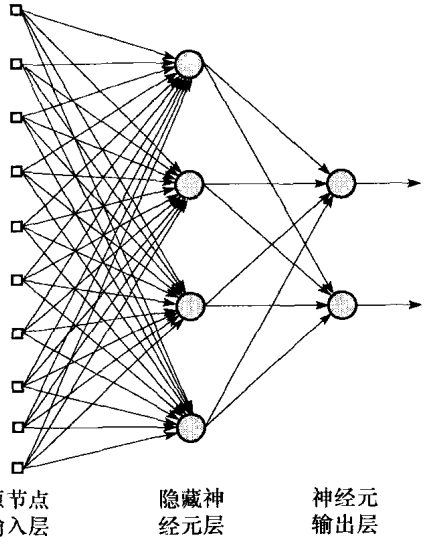


图16 具有一个隐藏层和输出层的全连接前馈网络

图 16 的网络也可以称为完全连接网络，这是指相邻层的任意一对节点都有连接。如果不是这样，我们称之为部分连接网络。

递归网络

递归网络和前馈网络的区别在于它至少有一个反馈环。如图 17 所示，递归网络可以由单层神经元组成，单层网络的每一个神经元的输出都反馈到所有其他神经元的输入中。这个图中描绘的结构没有自反馈环；自反馈环表示神经元的输出反馈到它自己的输入上。图 17 也没有隐藏层。

图 18 所示是带有隐藏神经元的另一类递归网络，反馈连接的起点包括隐藏层神经元和输出神经元。

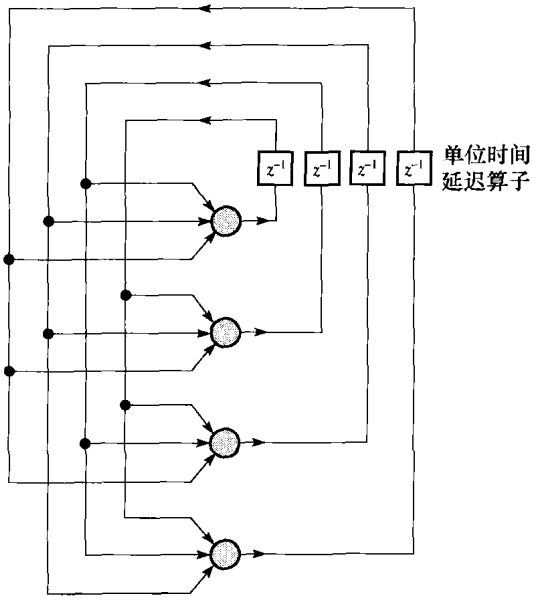


图 17 无自反馈环和隐藏神经元的递归网络

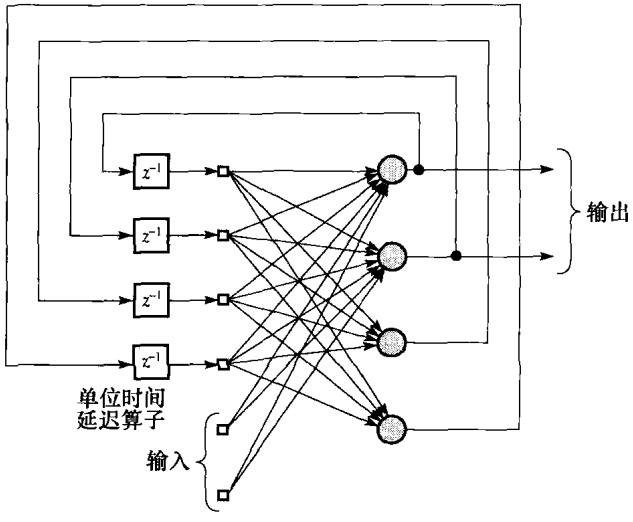


图 18 有隐藏神经元的递归网络

无论是在图 17 或图 18 的递归结构中，反馈环的存在对网络的学习能力和它的性能都有深刻的影响。并且，由于反馈环涉及使用单位时间延迟元素（记为 z^{-1}）构成的特殊分支，假如神经网络包含非线性单元，将导致非线性的动态行为。

0.7 知识表示

0.1 节中用到了“知识”这个术语，我们用它来定义神经网络的时候没有对它的涵义作明确的表述。现在来关注这一点，并在下面给出关于知识的一般性定义（Fischler and Firschein，1987）：

知识就是人或机器储存起来以备使用的信息或模型，用来对外部世界作出解释、预测和适当的反应。

知识表示的主要特征有两个方面：(1) 什么信息是明确表述的；(2) 物理上信息是如何被编码和使用的。按知识表示的本性，它是目标导向的。在“智能”机器的现实应用中，可以说好的方案取决于好的知识表示（Woods，1986）。代表一类特殊智能机器的神经网络也是如此。但是，典型地，从输入到内部网络参数的可能表现形式是高度多样性的，这就导致基于神经网络的对满意解的求解成为一个具有挑战性的设计。

神经网络的一个主要任务是学习它所依存的外部世界（环境）模型，并且保持该模型和真实世界足够兼容，使之能够实现感兴趣应用的特定目标。有关世界的知识由两类信息组成。

1. 已知世界的状态，由“什么是”事实和“什么是已知道的”事实所表示；这种形式的知识称为*先验信息*（priori information）。

2. 对世界的观察（测量），由神经网络中被设计用于探测环境的传感器获得。一般说来，这些观察是带有固有噪声的，这是由于传感器的噪声和系统的不完善而产生的误差。不管怎样，这样得到的观察会提供一个信息池，从中提取样例来训练神经网络。

样例可以是*有标号的*，也可以是*无标号的*。对于带标号样例来说，每个样例的输入信号有相应的与之配对的期望响应（即目标输出）。另一方面，无标号的样例包括输入信号自身的不同实现。不管怎样，一组样例，无论有无标号，都代表了神经网络通过训练可以学习的环境知识。但是，要说明的是，带标号样例的采集可能代价较高，因为它们需要“教师”来对每个带标号样例提供需要的响应。与之相反，通常无标号样例数目是足够的，因为对无标号样例来说不需要教师。

一组由输入信号和相应的期望响应所组成的输入输出对称为*训练数据集*（set of training data）或简单称为*训练样本*（training sample）。为了说明怎样使用这样的数据集，我们以手写数字识别问题为例。在这个问题中，输入信号是一幅黑白图像，每幅图像代表可以从背景中明显区分出的十个数字之一。期望的响应就是“确定”网络的输入信号代表哪个数字。通常训练样本就是手写体数字的大量变形，这代表了真实世界的情形。有了这些样本，可以用如下的办法设计神经网络：

- 为神经网络选择一个合适的结构，输入层的源节点数和输入图像的像素数一样，而输出层包含10个神经元（每个数字对应一个神经元）。利用合适的算法，以样本的一个子集来训练网络。这个网络设计阶段称为*学习*。
- 用陌生样本来测试已训练网络的识别性能。具体来说，呈现给网络一幅输入图像时并不告诉它这幅图像属于哪个数字。网络的性能就用网络报告的数字类别和输入图像的实际类别的差异来衡量。网络运行的这个第二个阶段叫做*测试*，对测试模式而言的成功性叫做*泛化*，这是借用了心理学的术语。

这里神经网络的设计与传统信息处理对应部分（模式分类器）的设计有着根本的差别。对后一种情况来说，首先我们通常设计一个观测环境的数学模型，并利用真实数据来验证这个模型，再以此模型为基础来设计。相反，神经网络的设计直接基于实际数据，“*让数据自己说话*”。因此神经网络不但提供了其内嵌于环境的隐含模型，也实现了感兴趣的信息处理功能。

用于训练神经网络的例子可以由*正例*和*反例*组成。比如，在被动声呐探测问题上，正例是有关包括感兴趣的目标（如潜艇）的输入训练数据。在被动声呐环境下，测试数据中可能存在的海洋生物经常造成虚警。为了缓解这个问题，可以把反例（如海洋生物的回声）包括在训练集中从而教会网络不要混淆海洋生物和目标。

在神经网络的独特结构中，周围环境的知识表示是由网络的自由参数（即突触权值和偏置）的取值所定义的。这种知识表示的形式构成神经网络的设计本身，因此，也是网络性能的关键。

知识表示的规则

然而，在人工网络中知识的表示是非常复杂的。这里有关于知识表示的通用的4条规则，如下所述。

规则1 *相似类别中的相似输入通常应产生网络中相似的表示，因此，可以归入同一类中。*

测量输入相似性有很多方法。常用的*测量方法*是利用欧几里得距离的概念。具体来说，令$\mathbf{x}_i$定义一个$m\times 1$的向量，

$$\mathbf{x}_i = [x_{i1}, x_{i2}, \cdots, x_{im}]^T$$

所有的元素都是实值；上标 T 表示矩阵转置。向量 x_i 就是 m 维空间（称为欧几里得空间）的一个点，记为 $\mathbb{R}^m$。如图 19 所示，两个 $m\times 1$ 向量 $\mathbf{x}_i$，$\mathbf{x}_j$ 之间的欧几里得距离定义为

$$d(\mathbf{x}_i, \mathbf{x}_j) = \| \mathbf{x}_i - \mathbf{x}_j \| = \left[\sum_{k=1}^{m} (x_{ik} - x_{jk})^2 \right]^{1/2} \tag{23}$$

其中 x_{ik}，x_{jk} 分别是输入向量 $\mathbf{x}_i$，$\mathbf{x}_j$ 的第 k 个分量。相应地，由向量 $\mathbf{x}_i$，$\mathbf{x}_j$ 表示的两个输入的相似性就定义为欧几里得距离 $d(\mathbf{x}_i, \mathbf{x}_j)$。输入向量 $\mathbf{x}_i$ 和 $\mathbf{x}_j$ 相距越近，欧几里得距离 $d(\mathbf{x}_i, \mathbf{x}_j)$就越小，相似性就越大。规则 1 说明，如果两个向量是相似的，就将它们归入同一类。

另一个相似性测量方法是基于点积或内积，它也是借用了矩阵代数的概念。给定一对相同维数的向量 $\mathbf{x}_i$，$\mathbf{x}_j$，它们的内积是 $\mathbf{x}_i^T\mathbf{x}_j$，定义为向量 $\mathbf{x}_i$ 对向量 $\mathbf{x}_j$ 的投影，如图 19 所示。可展开如下：

$$(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T\mathbf{x}_j = \sum_{k=1}^{m} x_{ik}x_{jk} \tag{24}$$

内积$(\mathbf{x}_i, \mathbf{x}_j)$除以范数积 $\| \mathbf{x}_i \| \cdot \| \mathbf{x}_j \|$，就是两个向量 $\mathbf{x}_i$，$\mathbf{x}_j$ 的夹角的余弦。

这里定义的两种相似性度量有密切的关系，如图 19 所示。图 19 清楚地表明欧几里得距离 $\| \mathbf{x}_i - \mathbf{x}_j \|$ 越小，向量 $\mathbf{x}_i$ 和 $\mathbf{x}_j$ 越相似，内积 $\mathbf{x}_i^T\mathbf{x}_j$ 越大。

为了把这种关系置于形式化基础之上，首先将向量 $\mathbf{x}_i$ 和 $\mathbf{x}_j$ 归一化，即

$$\| \mathbf{x}_i \| = \| \mathbf{x}_j \| = 1$$

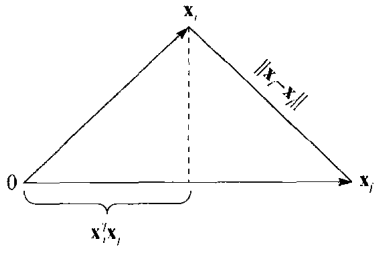


图 19 图解内积以及作为模式相似性度量的欧几里得距离之间的关系

我们就可以将式(23)写成

$$d^2(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j) = 2 - 2\mathbf{x}_i^T\mathbf{x}_j \tag{25}$$

式(25)表明最小化欧几里得距离 $d(\mathbf{x}_i, \mathbf{x}_j)$就对应于最大化内积$(\mathbf{x}_i, \mathbf{x}_j)$，因而，也对应于最大化 $\mathbf{x}_i$ 和 $\mathbf{x}_j$ 之间的相似性。

这里的欧几里得距离和内积的定义都是用确定性的术语定义的。如果向量 $\mathbf{x}_i$ 和 $\mathbf{x}_j$ 是“随机的”，从不同数据群体或集合中得来的，又该怎样定义相似性呢？具体来说，假设两个群体的差异仅在它们的均值向量。令 $\boldsymbol{\mu}_i$ 和 $\boldsymbol{\mu}_j$ 分别表示向量 $\mathbf{x}_i$ 和 $\mathbf{x}_j$ 的均值。也就是说，

$$\boldsymbol{\mu}_i = \mathbb{E}[\mathbf{x}_i] \tag{26}$$

其中$\mathbb{E}$是数据向量 $\mathbf{x}_i$ 的集合体（ensemble）的统计期望算子（statistical expectation operator）。用同样的方法定义均值向量 $\boldsymbol{\mu}_j$。为了度量这两个群体的距离，可以用 Mahalanobis 距离来衡量，记为 d_{ij}。从 $\mathbf{x}_i$ 到 $\mathbf{x}_j$ 的距离的平方值定义为：

$$d_{ij}^2 = (\mathbf{x}_i - \boldsymbol{\mu}_i)^T\mathbf{C}^{-1}(\mathbf{x}_j - \boldsymbol{\mu}_j) \tag{27}$$

其中 $\mathbf{C}^{-1}$是协方差矩阵 $\mathbf{C}$ 的逆矩阵。假设两个群体的协方差矩阵是一样的，表示如下：

$$\mathbf{C} = \mathbb{E}[(\mathbf{x}_i - \boldsymbol{\mu}_i)(\mathbf{x}_i - \boldsymbol{\mu}_i)^T] = \mathbb{E}[(\mathbf{x}_j - \boldsymbol{\mu}_j)(\mathbf{x}_j - \boldsymbol{\mu}_j)^T] \tag{28}$$

则对于给定的 $\mathbf{C}$ 来说，距离 d_{ij} 越小，向量 $\mathbf{x}_i$ 和 $\mathbf{x}_j$ 越相似。

当 $\mathbf{x}_i = \mathbf{x}_j$，$\boldsymbol{\mu}_i = \boldsymbol{\mu}_j = \boldsymbol{\mu}$ 且 $\mathbf{C} = \mathbf{I}$ 时（$\mathbf{I}$ 为单位矩阵），Mahalanobis 距离变为样本向量 $\mathbf{x}_j$ 和均值向量 $\boldsymbol{\mu}$ 间的欧几里得距离。

无论数据向量 $\mathbf{x}_i$ 和 $\mathbf{x}_j$ 是确定的还是随机的，规则 1 讨论了这两个向量之间是如何彼此相关的。相关性不仅仅在人类大脑中起着关键的作用，同样对多种信号处理系统来说也是如此(Chen et al., 2007)。

规则 2 网络对可分离为不同种类的输入向量给出差别很大的表示。

根据规则1，从一个特定的类中取得的模式之间有一个很小的代数测量值（如欧几里得距离）。另一方面，从不同类中取得的模式之间的代数测量值必须很大。因而，我们说规则2与规则1正相反。

规则3 *如果某个特征很重要，那么网络表示这个向量将涉及大量神经元。*

比如，考虑用雷达来探测在混杂状态（即雷达从不期望的目标如建筑物、树木和云层的反射）下的目标（如航空器）。这样的雷达系统的探测性能由下面两种概率形式来衡量：

- *探测概率*，就是目标存在时系统判断目标出现的概率。
- *虚警概率*，就是目标不存在时系统判断目标出现的概率。

根据Neyman-Pearson准则，在虚警概率不超过预先指定值的限制下，探测概率达到最大值（Van Trees，1968）。在这种应用中，接收到的信号中目标的实际出现代表着输入信号中的重要特征。实际上，规则3意味着在真实目标存在的时候应该有大量神经元参与判决该目标出现。同理，仅当混杂状态实际存在的时候才应该有大量神经元参与判决该混杂状态的出现。在这两种情形下，大量的神经元保证了判决的高度准确性和对错误神经元的容错性。

规则4 *如果存在先验信息和不变性，应该将其附加在网络设计中，这样就不必学习这些信息而简化网络设计。*

规则4特别重要，因为真正坚持这一规则就会使网络具有特定结构。这一点正是我们需要的，原因如下：

1. 已知生物视觉和听觉网络是非常特别的。

2. 相对于完全连接网络，特定网络用于调节的自由参数是较少的。因此，特定网络所需的训练数据更少，学习更快而且常常泛化性能更强。

3. 能够加快通过特定网络的信息传输速率（即网络的吞吐量）。

4. 和全连接网络相比特定网络的建设成本比较低，因为其规模较小。

然而，要说明的是，将先验知识结合进神经网络的设计中会限制神经网络仅能应用于根据某些感兴趣的知识来解决特定问题。

怎样在神经网络设计中加入先验信息

当然，怎样在神经网络设计中建立先验信息，以此建立一种特定的网络结构，是必须考虑的重要问题。遗憾的是，现在还没有一种有效的规则来实现这一目的；目前我们更多的是通过某些特别的过程来实现，并已知可以产生一些有用的结果。特别是我们使用下面两种技术的结合：

1. 通过使用称为接收域（receptive field）[6]的局部连接，*限制网络结构。*

2. 通过使用权值共享（weight-sharing）[7]，*限制突触权值的选择。*

这两种方法，特别是后一种，有很好的附带效益，它能使网络自由参数的数量显著下降。

作为特例，考虑一个如图20所示的部分连接前馈网络。这个网络构造具有带限制的结构。顶部6个源节点组成隐藏神经元1的接收域，网络其余隐藏神经元类推。一个神经元的接收域被定义为输入域区域，其输入刺激能够影响该神经元产生的输

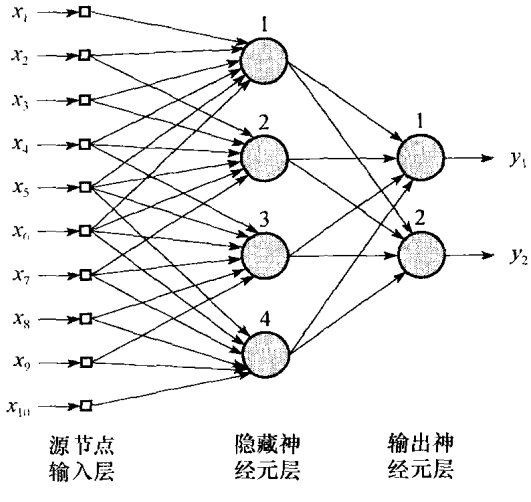


图20 联合利用接收域和权值共享的图例。所有四个隐神经元共享它们突触连接的相同权值集

出信号。接收域的绘制是关于该神经元行为以及其输出的有效而快速的描述。

为满足权值共享限制，我们对网络隐藏层中的每个神经元都使用同一组突触权值。这样，在图 20 所示的例子中，每个隐藏神经元有 6 个局部连接，共有 4 个隐藏神经元，我们可以表示每个隐藏神经元的诱导局部域如下：

$$v_j = \sum_{i=1}^{6} w_i x_{i+j-1}, \quad j = 1,2,3,4 \tag{29}$$

其中 $\{w_i\}_{i=1}^{6}$ 构成所有四个隐藏神经元共享的同一权值集，x_k 为从源节点 $k = i + j - 1$ 挑选的信号。式(29)为卷积和的形式。这里描述的前馈网络使用局部连接和权值共享的方式，我们称这样的前馈网络为卷积网络（LeCun and Bengio, 2003）。

在神经网络的设计中建立先验信息的问题是属于规则 4 的一部分；该规则的剩余部分涉及不变性问题，下面进一步讨论。

如何在网络设计中建立不变性

考虑下列物理现象：

- 当感兴趣的目标旋转时，观察者感知到的目标图像通常会产生相应的变化。
- 在一个提供它周围环境的幅度和相位信息的相干雷达中，由于目标相对于雷达射线运动造成的多普勒效应（Doppler effect），活动目标的回声在频率上会产生偏移。
- 人说话的语调会有高低快慢的变化。

为了分别建立一个对象识别系统、一个雷达目标识别系统和一个语音识别系统来处理这些现象，系统必须可以应付一定范围内观察信号的变换（transformation）。相应地，一个模式识别问题的主要任务就是设计对这些变换不变（invariant）的分类器。也就是说，分类器输出的类别估计不受分类器输入观察信号变换的影响。

至少可用三种技术使得分类器类型的神经网络对变换不变（Barnard and Casasent, 1991）：

1. 结构不变性（invariance by structure）。适当地组织神经网络的设计，在神经网络中加进不变性。具体来说，在建立网络的神经元突触连接时要求同一输入变换后必须得到同样的输出。例如考虑利用神经网络对输入图像的分类问题，要求神经网络不受图像关于中心的平面旋转的影响。我们可以在网络中强制加上旋转不变性如下：令 w_{ji} 表示神经元 j 和输入图像的像素 i 的连接权重。如果对所有两个到图像中心距离相等的像素 i 和 k 强制 $w_{ji} = w_{jk}$，那么神经网络对平面内的旋转不变。但是为了保持旋转不变性，对从原点出发的相同半径距离上输入图像的每个像素必须复制突触权值 w_{ji}。这说明了结构不变性的一个缺点：神经网络即使在处理中等大小的图像时，网络中的连接数目也会变得非常大。

2. 训练不变性（invariance by training）。神经网络有天生的模式分类能力。利用这种能力可以直接得到下面的变换不变性：用一些来自同一目标的不同样本来训练网络，这些样本代表目标的不同变换（即目标的不同方面）。假设样本足够大且训练后的网络已经学会分辨目标的不同方面，我们就可以期望训练后的网络能对已出现目标的不同变换做出正确的泛化。但是从工程的角度看，训练不变性有两方面不足：第一，如果一个神经网络训练后对已知变换的目标有不变性，不一定能保证它对其他类型目标的变换也有不变性。第二，网络的计算要求可能会很难达到，特别在高维特征空间尤其如此。

3. 不变特征空间（invariant feature space）。第三种建立神经网络不变性分类器的技术如图 21 所示。它依赖于这样的前提条件，即能提取表示输入数据本质信息内容特性的特征，并且这些特征对输入的变换保持不变。如果使用这样的特征，那么分类神经网就可以从刻画具有复杂决

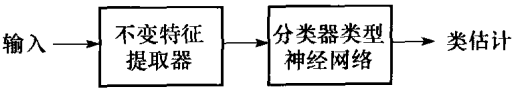


图 21 不变特征空间类型系统方框图

策边界的目标变换范围的负担中解脱出来。确实，同一目标的不同事例的差异仅仅在于噪声和偶发事件等不可避免因素的影响。特征空间不变性提供了三个明显的好处：第一，适用于网络的特征数可以降低到理想的水平。第二，网络设计的要求放宽了。第三，所有目标的已知变换的不变性都得到保证。

例 1 自回归模型

为了描述不变特征空间思想，考虑一个用于空中监控相干雷达系统的例子，其感兴趣的目标可能包括航空器、天气、迁移鸟群以及地面目标。这些目标的雷达回声具有特有的谱特征。并且，实验研究表明这样的雷达信号容易用中等大小阶的自回归（autoregressive，AR）过程模型来建模（Haykin and Deng，1991）。AR 模型是如下对复数数据定义的回归模型的特殊形式：

$$x(n) = \sum_{i=1}^{M} a_i^* x(n-i) + e(n) \tag{30}$$

其中 $\{a_i\}_{i=1}^{M}$ 为 AR 系数，M 为模型阶，$x(n)$ 为输入，$e(n)$ 为用白噪声表示的误差。基本上，式(30)的 AR 模型由带状延迟线滤波器表示，如图 22a 中 $M=2$ 的情形所示。同样，它可由图 22b 所示的网格滤波器表示，它的系数称为反射系数。图 22a 中模型的 AR 系数和图 22b 中模型的反射系数一一对应。所描绘的两个模型都假设输入 $x(n)$ 是复数，因为在相干雷达的情形下，AR 系数和反射系数都为复数。在式(30)和图 22 中的星号表示复共轭。现在可以说相干雷达数据可以用一组自回归系数来描述，或者由一组相应的反射系数来描述。后一组系数有计算上的优点，已存在有效的算法从输入数据直接计算。但是，特征提取问题是很复杂的，因为活动物体产生不同的多普勒频率，这取决于测得的物体相对于雷达的径向速度，并且作为特征判别式的反射系数的谱分布会产生模糊。为了克服这种困难，必须建立反射系数计算中的多普勒不变性。第一个反射系数的相位角结果与雷达信号的多普勒频率相等。相应地，归一化多普勒频率可以去掉多普勒位移的均值。这些可以通过从输入数据计算得到的常规反射系数 $\{\kappa_m\}$ 定义新的反射系数 $\{\kappa'_m\}$ 来实现：

$$\kappa'_m = \kappa_m e^{-jm\theta} \quad m = 1,2,\cdots,M \tag{31}$$

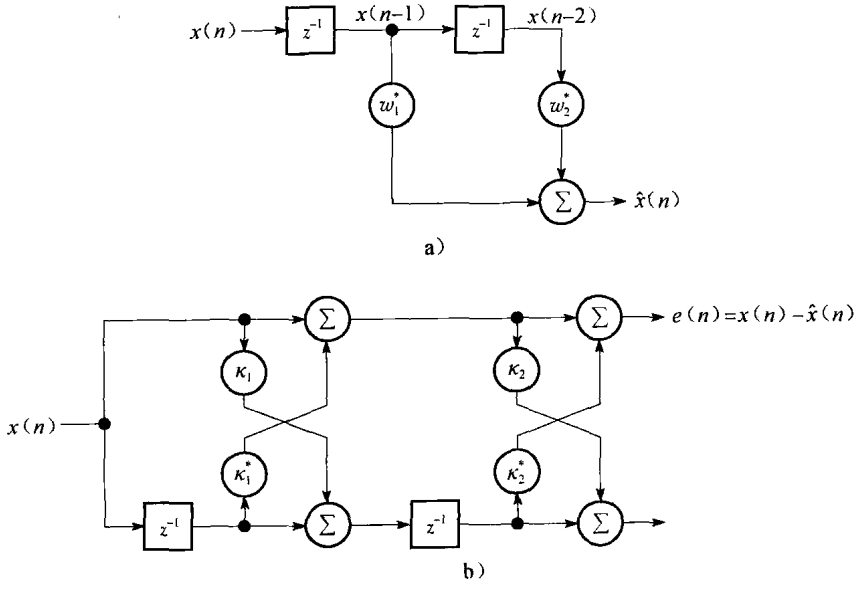


图 22 二阶自回归模型：a) 带状延迟线模型；b) 网格滤波器模型（星号表示复共轭）

其中 θ 为第一反射系数的相位角。式(31)描述的运算称为外差法。一组多普勒不变雷达特征可由归一化的发射系数 κ_1'，κ_2'，…，κ_M' 表示，κ_1' 为唯一的实系数。我们说过，空中监控的雷达目标主要可归类为天气、鸟群、航空器和地面，前三类目标都是动的，后一种则是不动的。地面回声混频后的谱参数和航空器类似，但因为其小的多普勒位移，地面回声可以和飞机相区别。相应地，雷达分类器包括一个如图 23 所示的后处理器，用来操作分类结果（编码标号）以识别地面类（Haykin and Deng，1991）。这样，在图 23 中的预处理器处理从分类器输入中抽取的多普勒位移不变特征，而后处理器利用存储的多普勒特征区分返回的地面和航空器信号。 ■

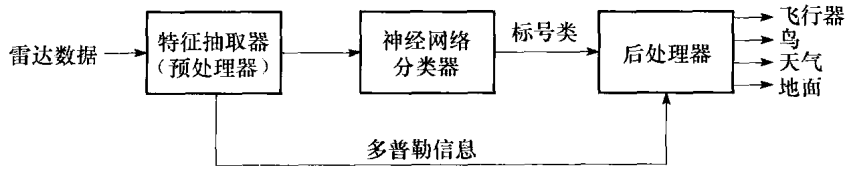


图 23 雷达信号的多普勒位移不变分类器

例 2 回声定位蝙蝠

神经网络知识表示的一个更有趣的例子是蝙蝠的生物回声定位声呐系统。为了声音映射，大多数蝙蝠使用频率调制（FM 或“chirp”）信号，在 FM 信号中，信号的瞬时频率随时间变化。具体来说，蝙蝠用口发出短时 FM 声呐信号，用听觉系统来作接收器。对于感兴趣目标的回声在听觉系统中选用声音参数不同组合的神经元活动来表达。蝙蝠的听觉表达有三个主要的神经维数（Simmons 等，1992）：

- *回声频率*，在耳蜗频率图中通过“位置”发信来编码；通过整个听觉系统的通路保存，该通路是调制成不同频率的一定神经元的有序排列。
- *回声幅度*，由其他具有不同动态范围的神经元编码；它被表示成幅度调制和每个刺激的放电次数。
- *回声延迟*，通过神经计算来编码（基于互相关）并产生延迟选择响应。它被表示成目标范围调制。

用于图像形成的目标回声的两个主要特点是目标“形状”的谱和目标范围的“延迟”。利用目标不同反射面回声（反射）的到达时间，蝙蝠感知“形状”。为此目的，回声谱的频率信息被转换为目标的时间结构估计。由 Simmons 及其合作者对棕色大蝙蝠（Eptesicus fuscus）进行的实验严格验证了这个转换过程，它的组成包括并行时域转换和频率对时域转换构成，它的收敛输出产生目标感知图像范围轴上的共同延迟。虽然最初执行的回声延迟的听觉时间表示和回声谱的频率表示方法不同，但看起来蝙蝠的感知协调性来自于变换自身的一些性质。并且特征不变性被嵌入声呐图像形成过程，所以它本质上独立于目标运动和蝙蝠自己的运动。

一些最终评论

神经网络中的知识表示和网络结构有着直接关系。遗憾的是，还没有成功的理论可以根据环境来优化神经网络结构，或者评价修改网络结构对网络内部知识表示的影响。实际上，对这些问题的满意结果经常要对感兴趣的具体应用进行彻底的实验研究才能得到，而神经网络的设计者也成为结构学习环中的关键部分。 ■

0.8 学习过程

和我们自己能够通过多种不同的方法从周围环境中学习一样，神经网络也有多种不同的学习方法。广义上讲，我们可以通过神经网络的功能来对其学习过程进行如下分类：有教师学习

和无教师学习。按照同样的标准，后者又可以分为无监督学习和强化学习两个子类。这些应用于神经网络的不同形式是和人类学习的形式相似的。

有教师学习

有教师学习也称为监督学习。图 24 是说明这种学习方式的方框图。从概念上讲，我们可以认为教师具有对周围环境的知识，这些知识被表达为一系列的输入－输出样本。然而神经网络对环境却一无所知。现在假设给教师和神经网络提供从同样环境中提取出来的训练向量（即样例）。教师可以根据自身掌握的一些知识为神经网络提供对训练向量的期望响应。事实上，期望响应一般都代表着神经网络完成的最优动作。神经网络的参数可以在训练向量和误差信号的综合影响下进行调整。误差信号可以定义为神经网络的实际响应与预期响应之差。这种调整可以逐步而又反复地进行，其最终目的就是要让神经网络模拟（emulate）教师；在某种统计的意义下，可以认为这种模拟是最优的。利用这种手段，教师所掌握的关于环境的知识就可以通过训练过程最大限度地传授给神经网络。当条件成熟的时候，就可以将教师排除在外，让神经网络完全自主地应对环境。

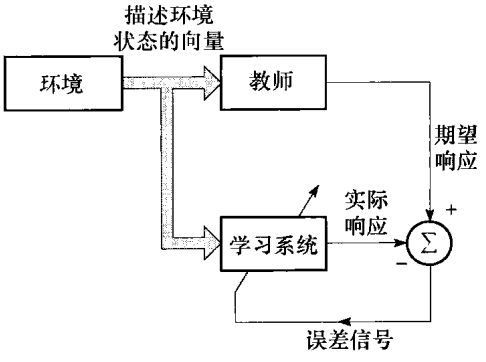


图 24 有教师学习方框图

我们刚刚描述的监督学习形式是误差-修正学习（error-correction learning）的基础。由图 24 可知，监督学习系统构成一个闭环反馈系统，但未知的环境不包含在循环中。我们可以采用训练样本的均方误差（mean-square error）或平方误差和（sum of squared errors）作为系统性能的测试手段，它可以定义为系统的一个关于自由参数（即突触权值）的函数。该函数可以看作一个多维误差-性能曲面（error-performance surface），或者简称误差曲面（error surface），其中自由参数作为坐标轴。实际误差曲面是在所有可能的输入输出样例上的平均。任何一个在教师监督下的系统给定的操作都表示误差曲面上的一个点。该系统要随时间而提高性能，就必须向教师学习，操作点必须要向误差曲面的最小点逐渐下降，误差极小点可能是局部最小，也可能是全局最小。监督学习系统可以根据系统当前的行为计算出误差曲面的梯度，然后利用梯度这一有用信息来求得误差极小点。误差曲面上任何一点的梯度是指向最快速下降方向的向量。实际上，通过样本进行监督学习，系统可以采用梯度向量的“瞬时估计”（instantaneous estimate），这时将样例的索引假定为访问的时间。采取这种估计一般会导致在误差曲面上操作点的运动轨迹经常以“随机行走”的形式出现。然而，如果我们能给定一个设计好的算法来使代价函数最小，而且有足够的输入/输出样本集和充裕的训练时间，那么监督学习系统往往能够较好地逼近一个未知的输入－输出映射。

无教师学习

在监督学习中，学习过程是在教师的监督下进行的。然而，在无教师学习范例中，顾名思义，没有教师监视学习过程。也就是说，没有任何带标号的样例可以供神经网络学习。在无教师学习范例下，又有如下的两个子类。

强化学习

在强化学习（reinforcement learning）中，输入输出映射的学习是通过与环境的不断交互完成的，目的是使一个标量性能指标达到最小。图 25 显示的是强化学习系统的方框图。这种学习系统建立在一个评价的基础上，评价将从周围环境中接收到的原始强化信号转换成一种称为启迪强化信号的高质量强化信号，两者都是标量输入（Barto 等，1983）。设计该系统的目的

是为了适应延迟强化情况下的学习，即意味着系统观察从环境接收的一个时序刺激，它们最终产生启发式的强化信号。

强化学习的目标是将 cost-to-go 函数最小化，cost-to-go 函数定义为采取一系列步骤动作的代价累积期望值，而不是简单的直接代价。可以证明：在时间序列上早期采取的动作事实上是整个系统最好的决定。学习系统的功能就是用来发现这些动作并将它们反馈给环境。

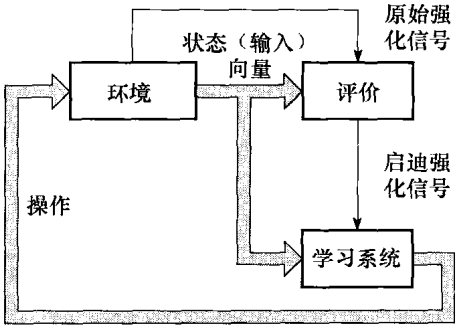


图 25 强化学习方框图；学习系统和环境都在反馈环内部

基于如下两个原因延迟强化学习系统很难完成：

- 在学习过程中的每个步骤，没有教师提供一个期望的响应。
- 生成原始强化信号时的延迟意味着学习机必须解决时间信任赋值问题。也就是说，对将导致最终结果的时间序列步中的每一个动作，学习机必须各自独立地对信任和责任赋值，而原始强化可能仅评价最终结果。

尽管存在这些困难，但延迟强化学习还是非常有吸引力的。它提供系统与周围环境交互的基础，因此可以仅仅在这种与环境交互获得经验结果的基础上，发展学习能力来完成指定任务。

无监督学习

如图 26 所示，在无监督或自组织学习系统中，没有外部的教师或者评价来监督学习的过程。而且，必须提供任务独立度量(task-independent measure) 来度量网络的表达质量，让网络学习该度量而且将根据这个度量来最优化网络自由参数。对一个特定的任务独立度量来说，一旦神经网络能够和输入数据的统计规律相一致，那么网络将会发展其形成输入数据编码特征的内部表示的能力，从而自动创造新的类别(Becker，1991)。

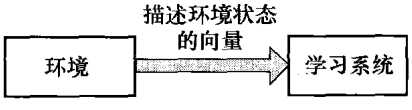


图 26 无监督学习方框图

为了完成无监督学习，我们可以使用竞争性学习规则。例如，可以采用包含两层的神经网络：输入层和竞争层。输入层接收有效数据。竞争层由相互竞争（根据一定的学习规则）的神经元组成，它们力图获得响应包含在输入数据中的特征的“机会”。最简单的形式就是神经网络采用“胜者全得”的策略。在这种策略中具有最大总输入的神经元赢得竞争而被激活，而其他所有的神经元被关掉。

0.9 学习任务

本章前面几节讨论了不同的学习范例。本节将描述一些基本的学习任务。对特定学习规则的选择与神经网络需要完成的学习任务密切相关，而学习任务的多样性正是神经网络通用性的证明。

模式联想

联想记忆是与大脑相似的依靠联想学习的分布式记忆。自从亚里士多德时代起，联想就被看作是人类记忆的一个显著特征，而且认知的所有模型都以各种形式使用联想作为其基本行为(Anderson，1995)。

联想有一种或两种形式：自联想与异联想。在自联想方式中，神经网络被要求通过不断出示一系列模式（向量）给网络而存储这些模式。其后将某已存模式的部分描述或畸变（噪声）形式出示给网络，而网络的任务就是检索（回忆）出已存储的该模式。异联想与自联想的不同

之处就在于一个任意的输入模式集合与另一个输出模式集合配对。自联想需要使用无监督学习方式，而异联想采用监督学习方式。

设 $\mathbf{x}_k$ 表示在联想记忆中的关键模式（向量）而 $\mathbf{y}_k$ 表示存储模式（向量）。网络完成的模式联想由下式表示：

$$\mathbf{x}_k \rightarrow \mathbf{y}_k, \quad k = 1,2,\cdots,q \tag{32}$$

其中 q 是存储在网络中的模式数。关键模式 $\mathbf{x}_k$ 作为输入，不仅决定存储模式 $\mathbf{y}_k$ 的存储位置，同时也拥有恢复该模式的键码。

在自联想记忆模式中：$\mathbf{y}_k = \mathbf{x}_k$，所以输入输出数据的空间维数相同。在异联想记忆模式中：$\mathbf{y}_k \neq \mathbf{x}_k$；因此，第二种情况的输出空间维数可能与输入数据空间维数相同，也可能不同。

联想记忆模式的操作一般包括两个阶段：

- *存储阶段*，指的是根据式(32)对网络进行训练。
- *回忆阶段*，网络根据所呈现的有噪声的或畸变的关键模式恢复对应的存储模式。

令刺激（输入）$\mathbf{x}$ 表示关键模式 $\mathbf{x}_j$ 的有噪声或畸变形式。如图 27 所示，这个刺激产生响应（输出）$\mathbf{y}$。对理想的回忆来说，我们有 $\mathbf{y}=\mathbf{y}_j$，其中 $\mathbf{y}_j$ 为由关键模式 $\mathbf{x}_j$ 联想的记忆模式。如果对 $\mathbf{x}=\mathbf{x}_j$ 有 $\mathbf{y}\neq\mathbf{y}_j$，就说联想记忆有*回忆错误*。

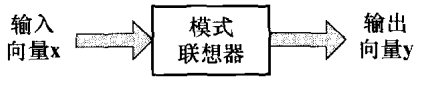


图 27 模式联想器输入输出关系

联想记忆中存储的模式数目 q 提供网络存储能力的一个直接度量。在设计联想记忆时，问题就是使*存储能力* q（表示为与构建网络的神经元总数 N 的百分比）尽量大，并且保持记忆中的大部分模式能正确回忆。

模式识别

人类非常擅长模式识别。通过感官，我们可以从周围的世界接收到数据，并且可以识别出数据源。我们往往是瞬间完成，几乎毫不费力。例如，我们能够识别出任何一张熟悉的脸，即使我们和这个人已经多年未曾谋面。无论电话线路如何差劲，我们还是可以迅速地根据他或者她的声音很快地甄别出你的熟人。仅仅闻一下，就能分辨出一个煮鸡蛋是否变坏。人类是通过学习过程来成功地实现模式识别的，神经网络也是如此。

*模式识别被形式地定义为一个过程，由这个过程将接收到的模式或信号确定为一些指定类别中的一个类。*神经网络要实现模式识别需要先经过一个训练的过程，在此过程中网络需要不断地接受一个模式集合以及每个特定模式所属的类别；然后，把一个以前没有见过但属于用于训练网络的同一模式总体的新模式呈现给神经网络。神经网络可以根据从训练数据中提取的信息识别特定模式的类别。神经网络的模式识别本质上是基于统计特性的，各个模式可以表示成多维决策空间的一些点。决策空间被划分为不同的区域，每个区域对应于一个模式类。决策边界由训练过程决定。我们可以根据各个模式类内部以及它们之间的固有可变性用统计方式来确定边界。

一般而论，采用神经网络的模式识别机分为如下两种形式：

- 如图 28a 的混合系统所示，识别机分为两部分，用来作特征提取的无监督网络和作分类的监督网络。这种方法遵循传统的统计特性模式识别方法（Fukunaga，1990；Duda 等，2001；Theodoridis and Koutroumbas，2003）。用概念术语来表示，一个模式是一个 m 维的可观测的数据，即 m 维观测（数据）空间集中的一个点 $\mathbf{x}$。如图 28b 所示，特征提取被描述为一个变换，它将点 $\mathbf{x}$ 映射成一个 q 维特征空间相对应的中间点 $\mathbf{y}(q<m)$。这种变换可看作是维数缩减（即数据压缩），这种做法主要是基于简化分类任务的考虑。分类本身可描述为一个变换，它将中间点 $\mathbf{y}$ 映射为 r 维决策空间上的一个类，其中 r 是要区分的类别数。

- 识别机设计成一个采用监督学习算法的前馈网络。在这第二个方法中，特征提取由网络隐藏层中的计算单元执行。

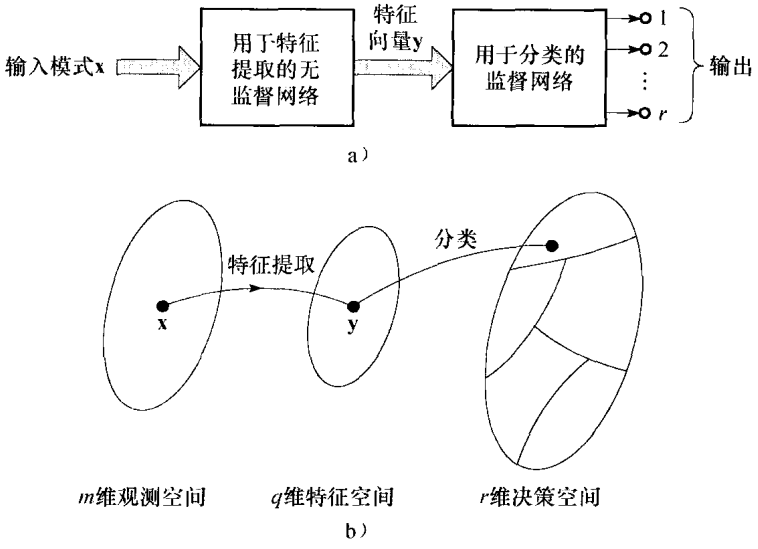


图 28 模式分类的经典方法图解

函数逼近

第三个感兴趣的学习任务是函数逼近。考虑由函数关系

$$\mathbf{d} = \mathbf{f}(\mathbf{x}) \tag{33}$$

描述的一个非线性输入输出映射，其中向量 $\mathbf{x}$ 是输入，向量 $\mathbf{d}$ 为输出。向量值函数 $\mathbf{f}(\cdot)$ 假定为未知。为了弥补函数 $\mathbf{f}(\cdot)$ 知识的缺乏，我们假定有如下的带标号样例集合：

$$\mathcal{T} = \{(\mathbf{x}_i, \mathbf{d}_i)\}_{i=1}^{N} \tag{34}$$

我们的要求是设计一个神经网络来逼近未知函数 $\mathbf{f}(\cdot)$，使由网络实际实现的描述输入-输出映射的函数 $\mathbf{F}(\cdot)$ 在欧几里得距离的意义下与 $\mathbf{f}(\cdot)$ 足够接近，即

$$\| \mathbf{F}(\mathbf{x}) - \mathbf{f}(\mathbf{x}) \| < \varepsilon, \text{对于所有的 } \mathbf{x} \tag{35}$$

其中 ε 是一个很小的正数。假定训练集 $\mathcal{T}$ 的样本数目 N 足够大，神经网络也有适当数目的自由参数，那么对于特定的任务逼近误差 ε 应当是足够小的。

在这里，逼近问题其实是一个很完整的监督学习，其中 $\mathbf{x}_i$ 是输入向量，而 $\mathbf{d}_i$ 是期望的响应。我们可以换一个角度思考这种问题，将监督学习看成是一个逼近问题。

神经网络逼近一个未知输入-输出映射的能力可以从两个重要途径加以利用：

1) *系统辨识*。令式(33)描述一个未知无记忆多输入-多输出 (multiple input-multiple output, MIMO) 系统的输入输出关系；所谓“无记忆”系统，是指时间不变的系统。然后利用式(34)中的标定的样例集合将神经网络作为系统的一个模型进行训练。假定 $\mathbf{y}_i$ 表示神经网络中对输入向量 $\mathbf{x}_i$ 产生的实际输出。正如图 29 所示，在 $\mathbf{d}_i$ (与 $\mathbf{x}_i$ 相对应的期望响应) 与输出 $\mathbf{y}_i$ 之间产生一个误差信号 $\mathbf{e}_i$，这个误差信号接着用来调节网络的自由参数，最终使未知系统的输出和神经网络输出在整个训练集 $\mathcal{T}$ 上的平方差在统计意义上达到最小。

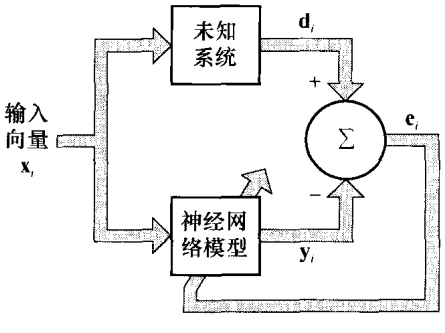


图 29 系统辨识方框图：实现辨识的神经网络是反馈环的一部分

2) *逆模型*。下一步假定我们给定一个已知无记忆 MIMO 系统，其输入输出关系如式(33)所示。在这种情况下的要求是如何构造一个逆模型，针对向量 $\mathbf{d}$ 产生向量 $\mathbf{x}$。逆系统可以由：

$$\mathbf{x} = \mathbf{f}^{-1}(\mathbf{d}) \tag{36}$$

描述，其中向量值函数 $\mathbf{f}^{-1}(\cdot)$表示 $\mathbf{f}(\cdot)$的反函数。注意，$\mathbf{f}^{-1}(\cdot)$不是 $\mathbf{f}(\cdot)$的倒数，上标-1仅仅是反函数的标志而已。在实际遇到的很多问题中，向量值函数 $\mathbf{f}(\cdot)$过于复杂，从而限制了求出反函数 $\mathbf{f}^{-1}(\cdot)$的直接公式。给定如式(34)的一些标定样例集，我们可以通过采取图 30 所示的方案构造一个神经网络来逼近函数 $\mathbf{f}^{-1}(\cdot)$。在这里描述的情况中，$\mathbf{x}_i$ 和 $\mathbf{d}_i$ 的作用交换了位置：向量 $\mathbf{d}_i$ 作为输入，向量 $\mathbf{x}_i$ 作为期望的响应。假定误差信号向量 $\mathbf{e}_i$ 表示 $\mathbf{x}_i$ 与神经网络针对 $\mathbf{d}_i$ 的实际输出 $\mathbf{y}_i$ 之间的差。与系统辨识问题类似，利用误差信号向量来调节网络的自由参数，最终使未知逆系统的输出和神经网络输出在整个训练样例集上的平方差在统计意义上达到最小。特别地，逆模型是比系统辨识更困难的学习任务，因为对它的解可能不是唯一的。

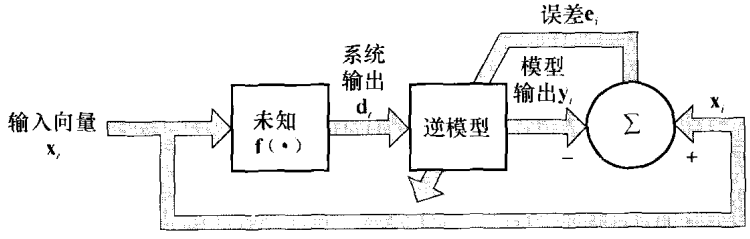


图 30 逆系统模型方框图。作为逆模型的神经网络是反馈环的一部分

控制

神经网络可以完成的另外一个学习任务是对设备进行控制操作。设备是指一个过程或者是可以在被控条件下维持运转的系统的一个关键部分。学习和控制相关其实不是一件什么值得大惊小怪的事情，毕竟我们人脑就是一个计算机（即信息处理器），作为整个系统的输出是实际的动作。在控制的这种意义下，人脑就是一个活生生的例子，它证明可以建立一个广义控制器，充分利用并行分布式硬件，能够并行控制成千上万的制动器（如肌肉神经纤维），能够处理非线性性和噪声，并且可以在长期计划水平上进行优化（Werbos，1992）。

考虑如图 31 所示的反馈控制系统。该系统涉及利用被控设备的单元反馈，即设备的输出直接反馈给输入。因此设备的输出 $\mathbf{y}$ 减去从外部信息源提供的参考信号 $\mathbf{d}$，产生误差信号 $\mathbf{e}$ 并将之应用到神经控制器以便调节它的自由参数。控制器的主要功能就是为设备提供相应的输入，从而使它的输出 $\mathbf{y}$ 跟踪参考信号 $\mathbf{d}$。换句话说，就是控制器不得不对设备的输入输出行为进行转换。

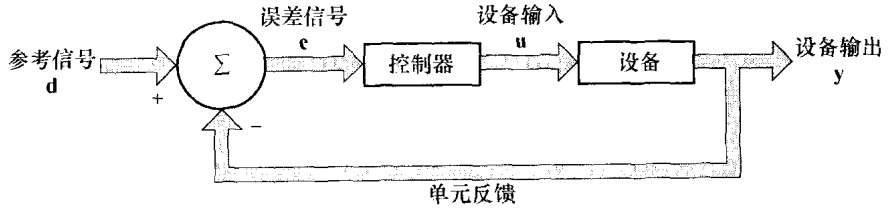


图 31 反馈控制系统方框图

我们注意到在图 31 中误差信号 $\mathbf{e}$ 在到达设备之前先通过神经控制器。结果，根据误差—修正学习算法，为了实现对设备自由参数的调节，我们必须知道 Jacobi 矩阵

$$\mathbf{J} = \left\{\frac{\partial y_k}{\partial u_j}\right\}_{j,k} \tag{37}$$

其中 y_k 是设备输出 $\mathbf{y}$ 的一个元素，而 u_j 是设备输入 $\mathbf{u}$ 的一个元素。遗憾的是偏导数 $\partial y_k/\partial u_j$ 对于不同的 k，j 依赖于设备的运行点，因而是未知的。我们可以采用下面两种方法之一来近

似计算该偏导数：

1）*间接学习*。利用设备的实际输入-输出测量值，首先构造神经网络模型产生一个它的复制品。接着利用这个复制品提供 Jacobi 矩阵 **J** 的一个估计值。随之把构成 Jacobi 矩阵 **J** 的偏导数用于误差-修正学习算法，以便计算对神经控制器的自由参数的调节（Nguyen and Widrow，1989；Suykens 等，1996；Widrow and Walach，1996）。

2）*直接学习*。偏导数 $\partial y_k/\partial u_j$ 的符号通常是知道的，而且在设备的动态区域内一般是不变的。这意味着我们可以通过各自的符号来逼近这些偏导数。它们的绝对值由神经控制器自由参数的一种分布式表示给出（Saerens and Soquet，1991；Schiffman and Geffers，1993）。因此，神经控制器能够直接从设备学习如何调节它的自由参数。

波束形成

波束形成是用来区分目标信号和背景噪声之间的空间性质的。用于实现波束形成的设备称为*波束形成器*（bemformer）。

波束形成适合利用于比如蝙蝠回声定位听觉系统皮质层的特征映射这样的任务（Suga，1990a；Simmons 等，1992）。蝙蝠的回声定位由发送短时频率调制（frequency-modulated，FM）声呐信号来了解周围环境，然后利用它的听觉系统（包括一对耳朵）集中注意于它的猎物（如飞行的昆虫）。蝙蝠的耳朵提供波束形成能力，听觉系统利用它产生*注意选择性*（attentional selectivity）。

波束形成通常用于雷达和声呐系统，它们的基本任务是在接收器噪声和干扰信号（如人为干扰）出现的情况下探测和跟踪感兴趣的目标。两个因素使这个任务复杂化：

- 目标信号源自未知的方向。
- 干扰信号无可用的先验信息。

处理这种情况的一种方法是使用*广义旁瓣消除器*（generalized sidelobe canceller，GSLC），图 32 显示的是它的方框图。这个系统由以下组件组成（Griffiths and Jim，1982；Haykin，2002）：

- *一个天线元阵列*，它提供对空间中离散点上被观察信号取样的手段。
- *一个线性组合器*，它是由固定权重集合 $\{w_i\}_{i=1}^{m}$ 定义的，其输出就是期望的响应。这个线性组合器的作用就像一个“空间滤波器”，它由一个辐射模式刻画（即一个天线输出振幅与输入信号入射角的极坐标图）。辐射模式的主瓣指向规定的方向。因此 GSLC 受它约束而产生一个无畸变的响应。线性组合器的输出记为 $d(n)$，它对波束形成器提供期望的响应。
- *一个信号阻塞矩阵* $\mathbf{C}_a$，它的功能是删除干扰，这种干扰是通过代表线性组合器的空间

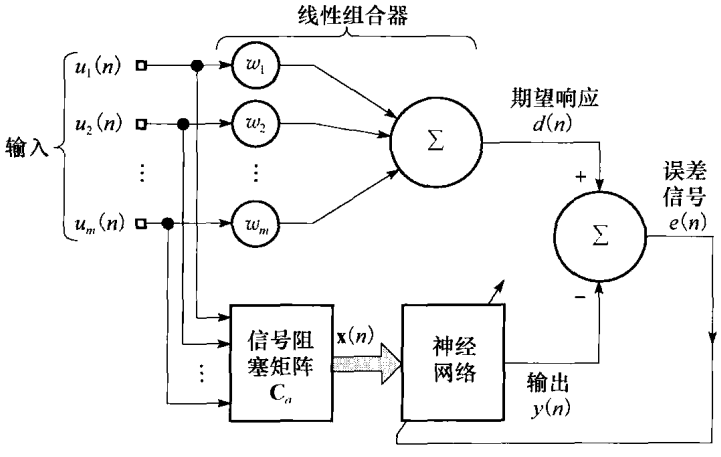


图 32 广义旁瓣消除器方框图

滤波器辐射模式的旁瓣而泄漏的。

- 一个具有可调参数的神经网络，它被设计成能适应干扰信号的统计变化。

神经网络自由参数的调节是由一个在误差信号 $e(n)$ 上操作的误差修正学习算法完成的，$e(n)$ 由线性组合器的输出 $d(n)$ 和神经网络的实际输出 $y(n)$ 之间的差确定。从而 GSLC 在线性组合器的监督下操作，线性组合器担当着“教师”的角色。作为普通的监督学习时，注意线性组合器是在神经网络的反馈环之外的。一个使用神经网络来学习的波束形成器称为神经-波束形成器（neuron-beamformer）。这类学习机可归入注意性神经计算机（attentional neurocomputers）的范围（Hecht-Nielsen，1990）。

0.10 结束语

在本导言章节中，我们把注意力集中于神经网络，而关于神经网络的研究是由人脑所启发的。神经网络的一个突出的重要性质是“学习”，而学习可以分为以下几个类别。

1）监督学习，通过最小化感兴趣的代价函数来实现特定的输入-输出映射，需要提供目标或者期望的响应。

2）无监督学习，其执行依赖于提供网络在自组织方式下学习所需要的对表示质量的“任务独立度量”。

3）强化学习，学习系统通过持续地与其环境的交互来最小化一个标量性能指标，从而实现输入-输出映射。

监督学习依赖于带标号样例（labeled example）的训练样本，每个样例由一个输入信号（刺激）以及相应的期望（目标）响应组成。实际上，我们发现收集带标号样例是费时而昂贵的任务，在处理大规模学习问题时尤其如此。因而我们发现带标号样例是短缺的。另一方面，无监督学习仅仅依赖于无标号样例，样例仅简单地由输入信号或者刺激组成，因而通常无标号样例的供应很充分。根据这样的事实，另一种学习的分支引起了广泛的兴趣：半监督学习。半监督学习的训练数据采用有标号和无标号的样例。如后续章节所讨论的，半监督学习最大的挑战在于当处理大规模模式分类问题时如何设计学习系统，使其运行过程是实际可行的。

强化学习处于监督学习和无监督学习之间。它通过学习系统和环境之间的持续交互而工作。学习系统提供行动并且从环境对该行动的反应中学习。例如，从效果上讲，监督学习中教师的角色在这里被一个评价值所取代，而这个评价值被综合进了机器学习中。

注释和参考文献

1. 神经网络的定义来自 Aleksander and Morton（1990）。
2. 有关大脑计算方面可读性的材料可参看 Churchland and Sejnowski（1992）。更详细的讲述可参看 Kandel 等（1991），Shepherd（1990），Kuffler 等（1984）和 Freeman（1975）。
3. 关于尖峰和尖峰神经元的细节可参看 Rieke 等（1997）。关于单个神经元的计算和信息处理能力的生物物理学观点，可参看 Koch（1999）。
4. 关于 sigmoid 函数和相关问题的全面叙述可参看 Mennon 等（1996）。
5. logistic 函数，或更精确地说 logistic 分布函数，其命名来自见于大量文献中的深奥的“logistic 增长律”。利用适当的度量单位，假定所有的增长过程可表示为 logistic 分布函数

$$F(t)=\frac{1}{1+e^{\alpha t-\beta}}$$

其中 t 代表时间，α 和 β 为常数。

6. 根据 Kuffler 等（1984），“接受域”（receptive field）这个术语最早是由 Sherrington（1906）创造的，并被 Hartline（1940）重新引入。在视觉系统环境下，神经元的接受域是指视网膜曲面上由光所引起的神经元放电的限制区域。
7. 权值共享技术最早在 Rumelhart 等（1986b）中描述。

第1章
Neural Networks and Learning Machines, 3E

Rosenblatt感知器

本章组织

感知器在神经网络发展的历史上占据着特殊位置：它是第一个从算法上完整描述的神经网络。它的发明者 Rosenblatt 是一位心理学家。在 20 世纪 60 年代和 70 年代，受感知器的启发，工程师、物理学家以及数学家们纷纷投身于神经网络各个不同方面的研究。更值得一提的是，尽管在 1958 年 Rosenblatt 关于感知器的论文就首次发表了，感知器（以本章所讲述的最基本形式）在今天依然是有效的。

本章分为如下部分：

1.1 节详述神经网络的形成阶段，追溯 1943 年 McCulloch 和 Pitts 的开创性工作。

1.2 节介绍 Rossenblatt 感知器的最基本形式。然后在 1.3 节讨论感知器收敛定理。这一定理证明了当感知器作为线性可分模式分类器的时候在有限数目时间步下是收敛的。

1.4 节建立高斯环境下感知器和贝叶斯分类器之间的关系。

1.5 节通过实验来说明感知器的模式分类能力。

1.6 节引入感知器代价函数，在此基础上展开讨论，为推导感知器收敛定理的批量版本铺路。

本章以 1.7 节的总结和讨论作为结束。

1.1 引言

在神经网络的形成阶段（1943—1958），一些研究者做出了开拓性的贡献：

- McCulloch and Pitts（1943）引入神经网络的概念作为计算工具。
- Hebb（1949）提出自组织学习的第一个规则。
- Rosenblatt（1958）提出感知器作为有教师学习（即监督学习）的第一个模型。

McCulloch-Pitts 关于神经网络的论文所产生的重要影响在导言中已经做了充分阐述。Hebb 学习的概念在第 8 章中将会做比较详细的讨论。在本章中我们讨论 Rosenblatt 感知器。

感知器是用于线性可分模式（即模式分别位于超平面所分隔开的两边）分类的最简单的神经网络模型。基本上它由一个具有可调突触权值和偏置的神经元组成。用来调整这个神经网络中自由参数的算法最早出现在 Rosenblatt（1958，1962）提出的用于其脑感知模型的一个学习过程中[1]。事实上，Rosenblatt 证明了当用来训练感知器的模式（向量）取自两个线性可分的类时，感知器算法是收敛的，并且决策面是位于两类之间的超平面。算法的收敛性证明称为*感知器收敛定理*。

建立在一个神经元上感知器的模式分类被限制为只能完成两类（假设）的模式分类。通过扩展感知器的输出（计算）层可以使感知器包括不止一个神经元，相应地可以进行多于两类的分类。但是，只有这些类是线性可分时感知器才能正常工作。重要的是，当感知器的基本理论用于模式分类器时，只需考虑单个神经元的情况。将这个理论推广到多个神经元是不重要的。

1.2 感知器

Rosenblatt 感知器建立在一个非线性神经元上，即神经元的 McCulloch-Pitts 模型。回忆一下，导言里讲过这种神经元模型由一个线性组合器和随后的硬限幅器（执行一个符号函数）组成，如图 1.1 所示。神经元模型的求和节点计算作用于突触上的输入的线性组合，同时也合

并外部作用的偏置。求和节点计算得到的结果，也就是诱导局部域，被作用于硬限幅器。相应地，当硬限幅器输入为正时，神经元输出 +1，反之则输出 −1。

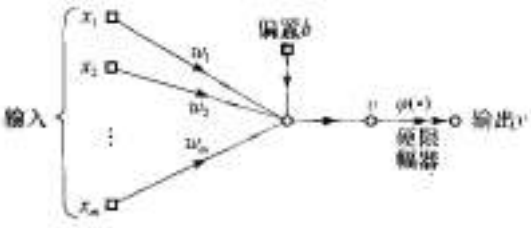


图 1.1 感知器的符号流图

在图 1.1 的符号流图模型中，感知器的突触权值记为 $w_1, w_2, \cdots, w_m$。相应地，作用于感知器的输入记为 $x_1, x_2, \cdots, x_m$。外部作用偏置记为 b。从这个模型我们发现硬限幅器输入或神经元的诱导局部域是

$$v = \sum_{i=1}^{m} w_i x_i + b \tag{1.1}$$

感知器的目的是把外部作用刺激 $x_1, x_2, \cdots, x_m$ 正确分为$\mathscr{C}_1$和$\mathscr{C}_2$两类。分类规则是：如果感知器输出 y 是 +1 就将 $x_1, x_2, \cdots, x_m$ 表示的点分配给类$\mathscr{C}_1$，如果感知器输出 y 是 −1 则分配给类$\mathscr{C}_2$。

为了进一步观察模式分类器的行为，一般要在 m 维信号空间中画出决策区域图，这个空间是由 m 个输入变量 $x_1, x_2, \cdots, x_m$ 所张成的。在最简单的感知器中存在被一个超平面分开的两个决策区域，此超平面定义为

$$\sum_{i=1}^{m} w_i x_i + b = 0 \tag{1.2}$$

对两个输入变量 x_1 和 x_2 的情形已在图 1.2 中做了说明，图中的决策边界是直线。位于边界线上方的点 (x_1, x_2) 分入$\mathscr{C}_1$类，位于边界线下方的点 (x_1, x_2) 分入$\mathscr{C}_2$类。注意这里偏置 b 的作用仅仅是把决策边界从原点移开。

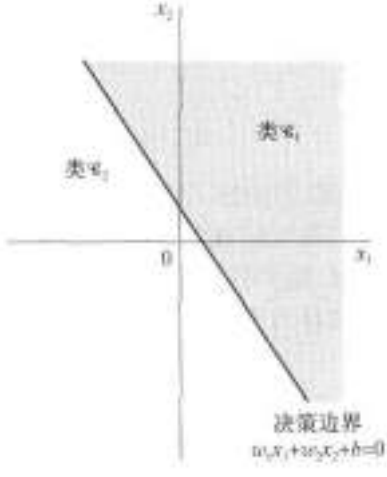


图 1.2 两维两类模式分类问题决策边界超平面的实例（在这个例子中超平面是一条直线）

感知器的突触权值 $w_1, w_2, \cdots, w_m$ 可以通过多次迭代来调整。对于自适应性可以使用通称为感知器收敛算法的误差修正规则，下面会讨论。

1.3 感知器收敛定理

为了导出感知器的误差修正学习算法，我们发现利用图 1.3 中的修正信号流图更方便。在这个与图 1.1 中的模型等价的模型中，偏置 $b(n)$ 被当作一个等于 +1 的固定输入量所驱动的突触权值。我们因此定义 $(m+1)\times 1$ 个输入向量

$$\mathbf{x}(n) = [+1, x_1(n), x_2(n), \cdots, x_m(n)]^T$$

这里 n 表示使用算法时的迭代步数。相应地定义 $(m+1)\times 1$ 个权值向量

$$\mathbf{w}(n) = [b, w_1(n), w_2(n), \cdots, w_m(n)]^T$$

因此，线性组合器的输出可以写成紧凑形式

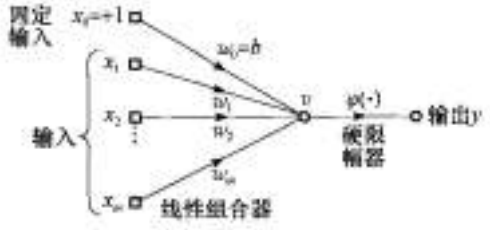


图 1.3 等价的感知器信号流图；为清楚起见省略了对时间的依赖性

$$v(n) = \sum_{i=0}^{m} w_i(n) x_i(n) = \mathbf{w}^T(n)\mathbf{x}(n) \tag{1.3}$$

这里，第一行中的 $w_0(n)$ 对应于 $i=0$，表示偏置 b。对于固定的 n，等式 $\mathbf{w}^T\mathbf{x}=0$ 在以 $x_1, x_2, \cdots,$

x_m 为坐标的 m 维空间中（对某些给定的偏置）所作的图定义了一个超平面，它就是两个不同输入类之间的决策面。

为了使感知器能够正确地工作，$\mathscr{C}_1$ 和 $\mathscr{C}_2$ 两个类必须是线性可分的。这意味着待分类模式必须分离得足够开以保证决策平面是超平面。这个要求对两维感知器的情形如图 1.4 所示。在图 1.4a 中两个类 $\mathscr{C}_1$ 和 $\mathscr{C}_2$ 分离得足够开，使得我们能画一个超平面（在此例中是一条直线）作为决策边界。但是，假如允许两个类 $\mathscr{C}_1$ 和 $\mathscr{C}_2$ 靠得太近，如图 1.4b 所示，它们就变成非线性可分的，这种情况就超出了感知器的计算能力。

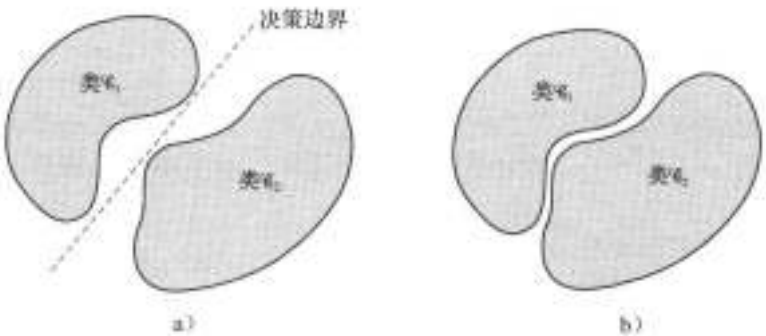


图 1.4 a) 一对线性可分离模式；b) 一对非线性可分离模式

假设感知器的输入变量来源于两个线性可分类。设 $\mathscr{H}_1$ 为训练向量 $\mathbf{x}_1(1)$，$\mathbf{x}_1(2)$，…中属于类 $\mathscr{C}_1$ 的向量所组成的子集，$\mathscr{H}_2$ 表示训练向量 $\mathbf{x}_2(1)$，$\mathbf{x}_2(2)$，…属于类 $\mathscr{C}_2$ 的向量所组成的子集。$\mathscr{H}_1$ 和 $\mathscr{H}_2$ 的并是整个训练集 $\mathscr{H}$。给定向量集 $\mathscr{H}_1$ 和 $\mathscr{H}_2$ 来训练分类器，训练过程涉及对权值向量 $\mathbf{w}$ 的调整使得两个类 $\mathscr{C}_1$ 和 $\mathscr{C}_2$ 线性可分。也就是说，存在一个权值向量 $\mathbf{w}$ 具有以下性质

$$\begin{aligned}\mathbf{w}^T\mathbf{x} &> 0 \quad \text{对属于类 } \mathscr{C}_1 \text{ 的每个输入向量 } \mathbf{x}\\ \mathbf{w}^T\mathbf{x} &\leqslant 0 \quad \text{对属于类 } \mathscr{C}_2 \text{ 的每个输入向量 } \mathbf{x}\end{aligned} \tag{1.4}$$

在式(1.4)的第二行中当 $\mathbf{w}^T\mathbf{x}=0$ 时我们随意地选择输入向量 $\mathbf{x}$ 属于类 $\mathscr{C}_2$。给定训练向量子集 $\mathscr{H}_1$ 和 $\mathscr{H}_2$，感知器的训练问题就是找到一个权值向量 $\mathbf{w}$ 满足式(1.4)中的两个不等式。

使基本感知器的权值向量自适应的算法现在可以用以下公式来表述：

1. 假如训练集合的第 n 个成员 $\mathbf{x}(n)$ 根据算法中的第 n 次迭代的权值向量 $\mathbf{w}(n)$ 能正确分类，那么感知器的权值向量按下述规则不做修改：

$$\begin{aligned}\mathbf{w}(n+1) &= \mathbf{w}(n) \quad \text{假如 } \mathbf{w}^T\mathbf{x}(n) > 0 \text{ 且 } \mathbf{x}(n) \text{ 属于类 } \mathscr{C}_1\\ \mathbf{w}(n+1) &= \mathbf{w}(n) \quad \text{假如 } \mathbf{w}^T\mathbf{x}(n) \leqslant 0 \text{ 且 } \mathbf{x}(n) \text{ 属于类 } \mathscr{C}_2\end{aligned} \tag{1.5}$$

2. 否则，感知器的权值向量根据以下规则进行修改：

$$\begin{aligned}\mathbf{w}(n+1) &= \mathbf{w}(n) - \eta(n)\mathbf{x}(n) \quad \text{假如 } \mathbf{w}^T(n)\mathbf{x}(n) > 0 \text{ 且 } \mathbf{x}(n) \text{ 属于类 } \mathscr{C}_2\\ \mathbf{w}(n+1) &= \mathbf{w}(n) + \eta(n)\mathbf{x}(n) \quad \text{假如 } \mathbf{w}^T(n)\mathbf{x}(n) \leqslant 0 \text{ 且 } \mathbf{x}(n) \text{ 属于类 } \mathscr{C}_1\end{aligned} \tag{1.6}$$

这里学习率参数 $\eta(n)$ 控制着第 n 次迭代中作用于权值向量的调节。

假如 $\eta(n)=\eta>0$，这里 η 是与迭代次数 n 无关的常数，我们有一个感知器的固定增量自适应规则（fixed-increment adaptation rule）。

接下来首先证明当 $\eta=1$ 时固定增量自适应规则的收敛性。很明显 η 的具体值并不重要，只要它是正的。$\eta\neq1$ 时的值不影响模式可分性而仅仅改变模式向量的大小。对于可变 $\eta(n)$ 的情况稍后考虑。

感知器收敛定理[2] 的证明针对初始条件 $\mathbf{w}(0)=0$。假设对 $n=1,2,\cdots,\mathbf{w}^T(n)\mathbf{x}(n)<0$，且输入向量 $\mathbf{x}(n)$ 属于子集 $\mathscr{H}_1$。即，因为式(1.4)的第二个的条件不满足，感知器就不能正确地对向

量 $\mathbf{x}(1)$，$\mathbf{x}(2)$，…进行分类。在常量 $\eta(n)=1$ 的情况下，可以利用式(1.6)的第二行，有

$$\mathbf{w}(n+1)=\mathbf{w}(n)+\mathbf{x}(n)\quad \text{对 } \mathbf{x}(n) \text{ 属于类}\mathscr{C}_1 \tag{1.7}$$

给定初始条件 $\mathbf{w}(0)=\mathbf{0}$，可以迭代求解这个关于 $\mathbf{w}(n+1)$ 的方程而得到结果

$$\mathbf{w}(n+1)=\mathbf{x}(1)+\mathbf{x}(2)+\cdots+\mathbf{x}(n) \tag{1.8}$$

因为假设类 $\mathscr{C}_1$ 和 $\mathscr{C}_2$ 为线性可分的，因此对属于子集 $\mathscr{H}_1$ 的向量 $\mathbf{x}(1),\cdots,\mathbf{x}(n)$ 的不等式方程 $\mathbf{w}^T\mathbf{x}(n)>0$ 存在一个解 $\mathbf{w}_o$。对固定解 $\mathbf{w}_o$，可以定义一个正数 α，

$$\alpha=\min_{\mathbf{x}(n)\in\mathscr{H}_1}\mathbf{w}_o^T\mathbf{x}(n) \tag{1.9}$$

因此，在式(1.8)两边同时乘以行向量 $\mathbf{w}_o^T$，我们有

$$\mathbf{w}_o^T\mathbf{w}(n+1)=\mathbf{w}_o^T\mathbf{x}(1)+\mathbf{w}_o^T\mathbf{x}(2)+\cdots+\mathbf{w}_o^T\mathbf{x}(n)$$

所以，依据等式(1.9)中的定义，我们有

$$\mathbf{w}_o^T\mathbf{w}(n+1)\geqslant n\alpha \tag{1.10}$$

下面利用众所周知的 Cauchy-Schwarz 不等式。给定两个向量 $\mathbf{w}_o$ 和 $\mathbf{w}(n+1)$，Cauchy-Schwarz 不等式表述为

$$\|\mathbf{w}_o\|^2\|\mathbf{w}(n+1)\|^2\geqslant[\mathbf{w}_o^T\mathbf{w}(n+1)]^2 \tag{1.11}$$

这里 $\|\cdot\|$ 表示所包含变元向量的欧几里得范数，内积 $\mathbf{w}_o^T\mathbf{w}(n+1)$ 是标量。从式(1.10)得到 $[\mathbf{w}_o^T\mathbf{w}(n+1)]^2$ 大于或等于 $n^2\alpha^2$。从式(1.11)我们注意到 $\|\mathbf{w}_o\|^2\|\mathbf{w}(n+1)\|^2$ 大于或等于 $[\mathbf{w}_o^T\mathbf{w}(n+1)]^2$。这样就有

$$\|\mathbf{w}_o\|^2\|\mathbf{w}(n+1)\|^2\geqslant n^2\alpha^2$$

或等价地有

$$\|\mathbf{w}(n+1)\|^2\geqslant\frac{n^2\alpha^2}{\|\mathbf{w}_o\|^2} \tag{1.12}$$

下面我们遵循另一种发展路线。特别地，可以把式(1.7)改写为如下形式

$$\mathbf{w}(k+1)=\mathbf{w}(k)+\mathbf{x}(k),\quad k=1,\cdots,n,\quad \mathbf{x}(k)\in\mathscr{H}_1 \tag{1.13}$$

通过对式(1.13)两边同取欧几里得范数的平方，得到

$$\|\mathbf{w}(k+1)\|^2=\|\mathbf{w}(k)\|^2+\|\mathbf{x}(k)\|^2+2\mathbf{w}^T(k)\mathbf{x}(k) \tag{1.14}$$

但是，$\mathbf{w}^T(k)\mathbf{x}(k)\leqslant 0$。因此从等式(1.14)中得到

$$\|\mathbf{w}(k+1)\|^2\leqslant\|\mathbf{w}(k)\|^2+\|\mathbf{x}(k)\|^2$$

或等价于

$$\|\mathbf{w}(k+1)\|^2-\|\mathbf{w}(k)\|^2\leqslant\|\mathbf{x}(k)\|^2,\quad k=1,\cdots,n \tag{1.15}$$

把 $k=1,\cdots,n$ 情况下的这些不等式相加，结合所假设的初始条件 $\mathbf{w}(0)=\mathbf{0}$，我们得到不等式

$$\|\mathbf{w}(n+1)\|^2\leqslant\sum_{k=1}^{n}\|\mathbf{x}(k)\|^2\leqslant n\beta \tag{1.16}$$

这里 β 是一个正数，定义为

$$\beta=\max_{\mathbf{x}(k)\in\mathscr{H}_1}\|\mathbf{x}(k)\|^2 \tag{1.17}$$

式(1.16)表明权值向量 $\mathbf{w}(n+1)$ 的欧几里得范数平方的增长至多只能和迭代次数 n 是线性关系。

当 n 有足够大的值时，式(1.16)的第二个结果显然与式(1.12)的结果相矛盾。实际上，我们可以说 n 不能大于某个值 $n_{\max}$，值 $n_{\max}$ 使得式(1.12)和式(1.16)的等号都成立。这里，$n_{\max}$ 是下面方程的解：

$$\frac{n_{\max}^2\alpha^2}{\|\mathbf{w}_o\|^2}=n_{\max}\beta$$

给定解向量 $\mathbf{w}_o$，解出 $n_{\max}$，我们求出

$$n_{\max} = \frac{\beta \| \mathbf{w}_o \|^2}{\alpha^2} \tag{1.18}$$

这样我们证明了对所有的 n，$\eta(n)=1$，且 $\mathbf{w}(0)=\mathbf{0}$，如果解向量 $\mathbf{w}_o$ 存在，那么感知器权值的适应过程最多在 $n_{\max}$ 次迭代后终止。从式(1.9)、(1.17) 和 (1.18) 注意到 $\mathbf{w}_o$ 或 $n_{\max}$ 的解并不唯一。

现在可以叙述感知器的固定增量收敛定理 (Rosenblatt，1962)：

设训练向量的子集 $\mathcal{H}_1$ 和 $\mathcal{H}_2$ 是线性可分的，感知器的输入来自这两个子集。感知器在 n_0 次迭代后在如下意义下收敛：

$$\mathbf{w}(n_o) = \mathbf{w}(n_o+1) = \mathbf{w}(n_o+2) = \cdots$$

是对 $n_0 \leqslant n_{\max}$ 的一个解向量。

下面考虑当 $\eta(n)$ 变化时，单层感知器自适应的绝对误差修正过程。特别地，设 $\eta(n)$ 是满足下式的最小整数：

$$\eta(n)\mathbf{x}^T(n)\mathbf{x}(n) > |\mathbf{w}^T(n)\mathbf{x}(n)|$$

利用这个过程我们发现如果第 n 次迭代时的内积 $\mathbf{w}^T(n)\mathbf{x}(n)$ 存在符号错误，那么第 $n+1$ 次迭代中 $\mathbf{w}^T(n+1)\mathbf{x}(n)$ 符号就会是正确的。这说明如果在第 n 次迭代 $\mathbf{w}^T(n)\mathbf{x}(n)$ 有符号错误，可以通过设 $\mathbf{x}(n+1)=\mathbf{x}(n)$ 来改变第 $n+1$ 次迭代时的训练次序。换句话说，将每个模式重复呈现给感知器直到模式被正确分类。

注意当 $\mathbf{w}(0)$ 的初始值不为零时，仅仅是导致收敛需要的迭代次数或增加或减少，这依赖于 $\mathbf{w}(0)$ 与解 $\mathbf{w}_0$ 的相关程度。无论 $\mathbf{w}(0)$ 的值是多少，感知器都可以保证是收敛的。

在表 1.1 中我们对感知器收敛算法做出概述 (Lippmann，1987)。在此表第三步计算感知器的实际响应中使用的记号 sgn(·)，表示符号函数 (signum function)：

$$\text{sgn}(v) = \begin{cases} +1 & \text{如果 } v > 0 \\ -1 & \text{如果 } v < 0 \end{cases} \tag{1.19}$$

这样可以把感知器的量化响应 $y(n)$ 表示为以下的简洁形式：

$$y(n) = \text{sgn}[\mathbf{w}^T(n)\mathbf{x}(n)] \tag{1.20}$$

表 1.1 感知器收敛算法概述

变量和参数：

$\mathbf{x}(n) = m+1$ 维输入向量
$= [+1, x_1(n), x_2(n), \cdots, x_m(n)]^T$

$\mathbf{w}(n) = m+1$ 维权值向量
$= [b, w_1(n), w_2(n), \cdots, w_m(n)]^T$

$b =$ 偏置

$y(n) =$ 实际响应(量化的)

$d(n) =$ 期望响应

$\eta =$ 学习率参数，一个比 1 小的正常数

1. *初始化。* 设 $\mathbf{w}(0)=\mathbf{0}$。对时间步 $n=1, 2, \cdots$ 执行下列计算。
2. *激活。* 在时间步 n，通过提供连续值输入向量 $\mathbf{x}(n)$ 和期望响应 $d(n)$ 来激活感知器。
3. *计算实际响应。* 计算感知器的实际响应：
$$y(n) = \text{sgn}[\mathbf{w}^T(n)\mathbf{x}(n)]$$
这里 sgn(·) 是符号函数。
4. *权值向量的自适应。* 更新感知器的权值向量：
$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta[d(n) - y(n)]\mathbf{x}(n)$$
这里
$$d(n) = \begin{cases} +1 & \text{若 } \mathbf{x}(n) \text{ 属于类} \mathscr{C}_1 \\ -1 & \text{若 } \mathbf{x}(n) \text{ 属于类} \mathscr{C}_2 \end{cases}$$
5. *继续。* 时间步 n 增加 1，返回第 2 步。

注意输入向量 $\mathbf{x}(n)$ 是 $(m+1)\times 1$ 向量，它的第一个元素在整个计算中固定为 $+1$。相应地，权值向量 $\mathbf{w}(n)$ 是 $(m+1)\times 1$ 向量，它的第一个元素等于偏置 b。表 1.1 中的另一个要点是：我们引入一个量化期望响应 $d(n)$，定义为

$$d(n)=\begin{cases}+1 & \text{若 } \mathbf{x}(n) \text{ 属于类}\mathscr{C}_1\\ -1 & \text{若 } \mathbf{x}(n) \text{ 属于类}\mathscr{C}_2\end{cases} \tag{1.21}$$

因此，权值向量 $\mathbf{w}(n)$ 的自适应是以*误差修正学习规则*（error-correction learning rule）形式下的累加：

$$\mathbf{w}(n+1)=\mathbf{w}(n)+\eta[d(n)-y(n)]\mathbf{x}(n) \tag{1.22}$$

这里 η 是*学习率参数*，差 $d(n)-y(n)$ 起误差信号的作用。学习率参数是正常数，且 $0<\eta\leqslant 1$。当在这个区间里给 η 赋一个值时，必须记住两个互相冲突的需求（Lippmann，1987）：

- *平均*，过去输入的平均值提供一个稳定的权值估计，这需要一个较小的 η。
- *快速自适应*，相对于产生输入向量 $\mathbf{x}$ 的过程的固有分布的实时变化，快速自适应需要较大的 η。

1.4　高斯环境下感知器与贝叶斯分类器的关系

感知器与一类通称为贝叶斯分类器的经典模式分类器具有一定联系。在高斯环境下，贝叶斯分类器退化为线性分类器。这与感知器采用的形式是一样的。但是，感知器的线性特性并不是由于高斯假设而具有的。这一节我们研究这种联系，并借此深入研究感知器的运行。首先简单复习一下贝叶斯分类器。

贝叶斯分类器

在贝叶斯分类器和贝叶斯假设检验过程中，我们最小化平均风险（记为$\mathscr{R}$）。对两类问题（记为类$\mathscr{C}_1$ 和类$\mathscr{C}_2$），Van Trees（1968）定义的平均风险为：

$$\begin{aligned}\mathscr{R}=&c_{11}p_1\int_{\mathscr{H}_1}p_{\mathbf{X}}(\mathbf{x}|\mathscr{C}_1)\mathrm{d}\mathbf{x}+c_{22}p_2\int_{\mathscr{H}_2}p_{\mathbf{X}}(\mathbf{x}|\mathscr{C}_2)\mathrm{d}\mathbf{x}\\&+c_{21}p_1\int_{\mathscr{H}_2}p_{\mathbf{X}}(\mathbf{x}|\mathscr{C}_1)\mathrm{d}\mathbf{x}+c_{12}p_2\int_{\mathscr{H}_1}p_{\mathbf{X}}(\mathbf{x}|\mathscr{C}_2)\mathrm{d}\mathbf{x}\end{aligned} \tag{1.23}$$

这里各项的定义如下：

p_i = 观察向量 $\mathbf{x}$（表示随机向量 $\mathbf{X}$ 的实现值）取自子空间 $\mathscr{H}_i$ 的先验概率，这里 $i=1,2$ 且 $p_1+p_2=1$。

c_{ij} = 当类 $\mathscr{C}_j$ 是真实的类（即观察向量 x 是取自子空间 $\mathscr{H}_j$）时决策为由子空间 $\mathscr{H}_i$ 代表的类 $\mathscr{C}_i$ 的代价，$(i,j)=1,2$。

$p_{\mathbf{X}}(\mathbf{x}\mid\mathscr{C}_i)$ = 随机向量 $\mathbf{X}$ 的条件概率密度函数，假设观察向量 $\mathbf{x}$ 取自子空间 $\mathscr{H}_i$，$i=1,2$。

式(1.23)右边的头两项表示正确决策（即正确分类），后面两项代表不正确决策（即错误分类）。每个决策通过两个因子乘积加权：作出决策的代价和发生的相对频率（即先验概率）。

我们的目的是确定一个最小化平均风险的策略。因为需要作出这样的决策，在全部观察空间$\mathscr{X}$中每个观察向量 $\mathbf{x}$ 必须被设定或者属于$\mathscr{X}_1$ 或者属于$\mathscr{X}_2$。因此

$$\mathscr{X}=\mathscr{X}_1+\mathscr{X}_2 \tag{1.24}$$

相应地，可以把式(1.23)改写为等价的形式：

$$\begin{aligned}\mathscr{R}=&c_{11}p_1\int_{\mathscr{X}_1}p_{\mathbf{X}}(\mathbf{x}|\mathscr{C}_1)\mathrm{d}\mathbf{x}+c_{22}p_2\int_{\mathscr{X}-\mathscr{X}_1}p_{\mathbf{X}}(\mathbf{x}|\mathscr{C}_2)\mathrm{d}\mathbf{x}\\&+c_{21}p_1\int_{\mathscr{X}-\mathscr{X}_1}p_{\mathbf{X}}(\mathbf{x}|\mathscr{C}_1)\mathrm{d}\mathbf{x}+c_{12}p_2\int_{\mathscr{X}_1}p_{\mathbf{X}}(\mathbf{x}|\mathscr{C}_2)\mathrm{d}\mathbf{x}\end{aligned} \tag{1.25}$$

这里 $c_{11}<c_{21}$ 且 $c_{22}<c_{12}$。现在注意到下述事实：

$$\int_{\mathscr{X}} p_{\mathbf{X}}(\mathbf{x}|\mathscr{C}_1)\mathrm{d}\mathbf{x} = \int_{\mathscr{X}} p_{\mathbf{X}}(\mathbf{x}|\mathscr{C}_2)\mathrm{d}\mathbf{x} = 1 \tag{1.26}$$

因此，式(1.25)简化为：

$$\mathscr{R} = c_{21}p_1 + c_{22}p_2 + \int_{\mathscr{X}_1} [p_2(c_{12}-c_{22})p_{\mathbf{X}}(\mathbf{x}|\mathscr{C}_2) - p_1(c_{21}-c_{11})p_{\mathbf{X}}(\mathbf{x}|\mathscr{C}_1)]\mathrm{d}\mathbf{x} \tag{1.27}$$

式(1.27)右边的头两项代表一个固定代价。因为需要最小化平均风险$\mathscr{R}$，我们从式(1.27)得到以下最优分类的策略：

1. 所有使被积函数（即方括号里的表达式）为负的观察向量 x 的值都归于子空间$\mathscr{X}_1$（即类$\mathscr{C}_1$），因为此时积分对风险$\mathscr{R}$有一个负的贡献。

2. 所有使被积函数为正的观察向量 $\mathbf{x}$ 的值都必须从子空间$\mathscr{X}_1$ 中排除（即分配给类$\mathscr{C}_2$），因为此时积分对风险$\mathscr{R}$有一个正的贡献。

3. 使被积函数为零的 x 的值对平均风险$\mathscr{R}$没有影响，因此可以任意分配。假设这些点分配给子空间$\mathscr{X}_2$（即类$\mathscr{C}_2$）。

在这个基础上，写出贝叶斯分类器公式：

假如条件

$$p_1(c_{21}-c_{11})p_{\mathbf{X}}(\mathbf{x}|\mathscr{C}_1) > p_2(c_{12}-c_{22})p_{\mathbf{X}}(\mathbf{x}|\mathscr{C}_2)$$

满足，把观察向量 $\mathbf{x}$ 分配给子空间$\mathscr{X}_1$（即类$\mathscr{C}_1$）。否则把 $\mathbf{x}$ 分配给$\mathscr{X}_2$（即类$\mathscr{C}_2$）。

为了简化起见，定义

$$\Lambda(\mathbf{x}) = \frac{p_{\mathbf{X}}(\mathbf{x}|\mathscr{C}_1)}{p_{\mathbf{X}}(\mathbf{x}|\mathscr{C}_2)} \tag{1.28}$$

和

$$\xi = \frac{p_2(c_{12}-c_{22})}{p_1(c_{21}-c_{11})} \tag{1.29}$$

量 $\Lambda(\mathbf{x})$是两个条件概率密度函数的比，被称为*似然比*（likelihood ratio）。量 ξ 称为检验的阈值。注意 $\Lambda(\mathbf{x})$和 ξ 都是恒正的。根据这两个量，可以把贝叶斯分类重新表述为：

假如对一个观察向量 $\mathbf{x}$，其似然比 $\Lambda(\mathbf{x})$比阈值 ξ 大，就把 $\mathbf{x}$ 分配给类$\mathscr{C}_1$，反之，分配给类$\mathscr{C}_2$。

图 1.5a 是一个描绘贝叶斯分类器的模块图。此模块图的要点是两方面的：

1. 进行贝叶斯分类器设计的数据处理被完全限制在似然比 $\Lambda(\mathbf{x})$的计算中。

2. 此计算与分配给先验概率的值和决策过程中的代价是完全无关的。这两个量仅仅影响阈值 ξ。

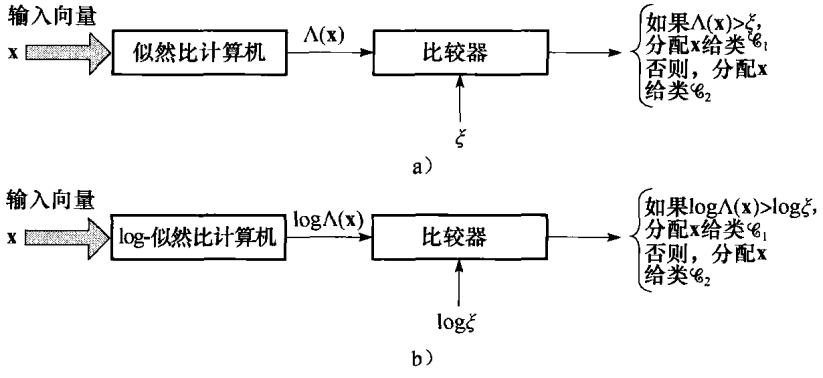


图 1.5 贝叶斯分类器的两个等价模型：a）似然比检验；b）对数似然比检验

从计算的观点来看，我们发现使用似然比的对数比使用似然比自身方便得多。允许这样做有两个理由。首先，对数是单调函数。其次，似然比 $\Lambda(\mathbf{x})$ 和阈值 ξ 都是正的。因此，贝叶斯分类器可以用如图 1.5b 所示的等价形式来实现。很显然，第二个图中嵌入的检验被称为对数似然比检验。

高斯分布下的贝叶斯分类器

现在考虑一个在高斯分布下两类问题的特殊情形。随机向量 $\mathbf{X}$ 的均值依赖于 $\mathbf{X}$ 是属于类$\mathscr{C}_1$还是$\mathscr{C}_2$，但 $\mathbf{X}$ 的协方差阵对两类都是一样的。也就是说：

$$\begin{aligned}
\text{类}\mathscr{C}_1:\quad & \mathbb{E}[\mathbf{X}]=\boldsymbol{\mu}_1\\
& \mathbb{E}[(\mathbf{X}-\boldsymbol{\mu}_1)(\mathbf{X}-\boldsymbol{\mu}_1)^T]=\mathbf{C}\\
\text{类}\mathscr{C}_2:\quad & \mathbb{E}[\mathbf{X}]=\boldsymbol{\mu}_2\\
& \mathbb{E}[(\mathbf{X}-\boldsymbol{\mu}_2)(\mathbf{X}-\boldsymbol{\mu}_2)^T]=\mathbf{C}
\end{aligned}$$

协方差矩阵 $\mathbf{C}$ 是非对角的，这意味着取自类$\mathscr{C}_1$和类$\mathscr{C}_2$的样本是相关的。假设 $\mathbf{C}$ 是非奇异的，这样它的逆矩阵 $\mathbf{C}^{-1}$ 存在。

在这个背景下可以把 $\mathbf{X}$ 的条件概率密度函数表示为多变量高斯分布：

$$p_{\mathbf{X}}(\mathbf{x}|\mathscr{C}_i)=\frac{1}{(2\pi)^{m/2}(\det(\mathbf{C}))^{1/2}}\exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T\mathbf{C}^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)\right),\ i=1,2 \tag{1.30}$$

这里 m 是观察向量 $\mathbf{x}$ 的维数。

进一步假设：

1. 两类$\mathscr{C}_1$和$\mathscr{C}_2$的概率相同：

$$p_1=p_2=\frac{1}{2} \tag{1.31}$$

2. 错误分类造成同样的代价，正确分类的代价为零：

$$c_{21}=c_{12}\quad\text{和}\quad c_{11}=c_{22}=0 \tag{1.32}$$

我们现在有了对两类问题设计贝叶斯分类器的信息。具体地讲，将式(1.30)代入式(1.28)并取自然对数，我们得到（简化后）：

$$\begin{aligned}
\log\Lambda(\mathbf{x}) &= -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_1)^T\mathbf{C}^{-1}(\mathbf{x}-\boldsymbol{\mu}_1)+\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_2)^T\mathbf{C}^{-1}(\mathbf{x}-\boldsymbol{\mu}_2)\\
&= (\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)^T\mathbf{C}^{-1}\mathbf{x}+\frac{1}{2}(\boldsymbol{\mu}_2^T\mathbf{C}^{-1}\boldsymbol{\mu}_2-\boldsymbol{\mu}_1^T\mathbf{C}^{-1}\boldsymbol{\mu}_1)
\end{aligned} \tag{1.33}$$

把式(1.31)和式(1.32)代入式(1.29)并取自然对数，得到

$$\log\xi=0 \tag{1.34}$$

式(1.33)和式(1.34)表明当前问题的贝叶斯分类器是线性分类器，如关系式

$$y=\mathbf{w}^T\mathbf{x}+b \tag{1.35}$$

所示，这里

$$y=\log\Lambda(\mathbf{x}) \tag{1.36}$$

$$\mathbf{w}=\mathbf{C}^{-1}(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2) \tag{1.37}$$

$$b=\frac{1}{2}(\boldsymbol{\mu}_2^T\mathbf{C}^{-1}\boldsymbol{\mu}_2-\boldsymbol{\mu}_1^T\mathbf{C}^{-1}\boldsymbol{\mu}_1) \tag{1.38}$$

更进一步，分类器由一个权值向量 $\mathbf{w}$ 和偏置 b 的线性组合器构成，如图 1.6 所示。

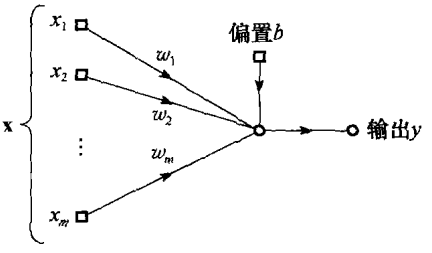


图 1.6　高斯分类器的信号流图

在式(1.35)的基础上，可以把对两类问题的对数似然比检验描述如下：

假如线性组合器（包括偏置 b）的输出是正的，把观察向量 $\mathbf{x}$ 分配给类 $\mathscr{C}_1$。否则，把它分配给类 $\mathscr{C}_2$。

这里描述的高斯环境下贝叶斯分类器的运行与感知器是类似的，因为它们都是线性分类器；请见式(1.1)和式(1.35)。但是，在它们之间还存在一些需要仔细检查的细微而重要的区别（Lippmann，1987）：

- 感知器运行的前提是待分模式是线性可分的。导出贝叶斯分类器过程中所假设的两个高斯分布的模式当然是互相重叠的，因此它们不是可分的。重叠的程度是由均值向量 $\boldsymbol{\mu}_1$ 和 $\boldsymbol{\mu}_2$ 以及协方差矩阵 $\mathbf{C}$ 所决定。重叠的性质如图 1.7 所示，这是一个随机标量的特殊情况（即维数 $m=1$）。当输入如图所示是不可分且其分布是重叠的时候，感知器收敛算法会出现问题，因为两类间的决策边界可能会持续振荡。
- 贝叶斯分类器最小化分类误差的概率。这个最小化与高斯分布下两类之间的重叠无关。例如，在图 1.7 的特例中，贝叶斯分类使决策边界总是位于高斯分布下两类 $\mathscr{C}_1$ 和 $\mathscr{C}_2$ 的交叉点上。
- 感知器收敛算法是非参数的，这指它没有关于固有分布形式的假设。它通过关注误差来运行，这些误差出现在分布重叠的地方。当输入由非线性物理机制产生同时它们的分布是严重偏离而且非高斯分布的时候，算法将可能工作得很好。相反，贝叶斯分类器是参数化的；它的导出是建立在高斯分布的假设上的，这可能会限制它的适用范围。
- 感知器收敛算法是自适应的且实现简单；它的存储需求仅限于权值集合和偏置。另一方面，贝叶斯分类器设计是固定的；可以使它变成自适应的，但代价是增加存储量和更高的计算复杂性。

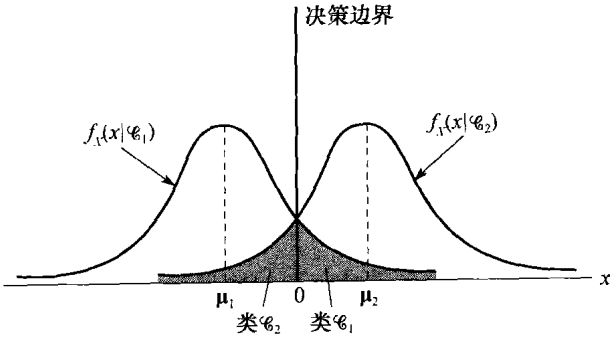


图 1.7 两个重叠的一维高斯分布

1.5 计算机实验：模式分类

本计算机实验的目的包括两个方面：

1）给出双月分类问题的详细说明，这一问题将成为本书处理模式分类实验部分的基本原型；

2）说明 Rosenblatt 感知器算法对线性可分模式正确分类的能力，并说明当线性可分性不满足时 Rosenblatt 感知器会崩溃。

分类问题详细说明

图 1.8 给出了一对非对称的面对面的“月亮”。被标志为“区域 A”的月亮是关于 y-轴对称的，而被标志为“区域 B”的月亮被安置在 y-轴右边距离半径 r 以及 x-轴下面距离 d 的地方。这两个月亮具有相同的参数：

$$\text{每个月亮的半径}, r = 10$$

$$\text{每个月亮的宽度}, w = 6$$

将两个月亮分开的垂直距离 d 是可调的，它是根据 x-轴来测量的，如图 1.8 所示。

- 增加 d 的正值意味着增加两个月亮之间相互分离；
- 增加 d 的负值意味着两个月亮会相互靠近。

训练样本集 $\mathcal{T}$ 是由 1 000 对数据点所组成，每对数据点的其中一个是取自区域 A，另一个取自

区域 B，两者都是随机选取的。测试样本集是由 2 000 对数据点组成的，也是以随机方式选取的。

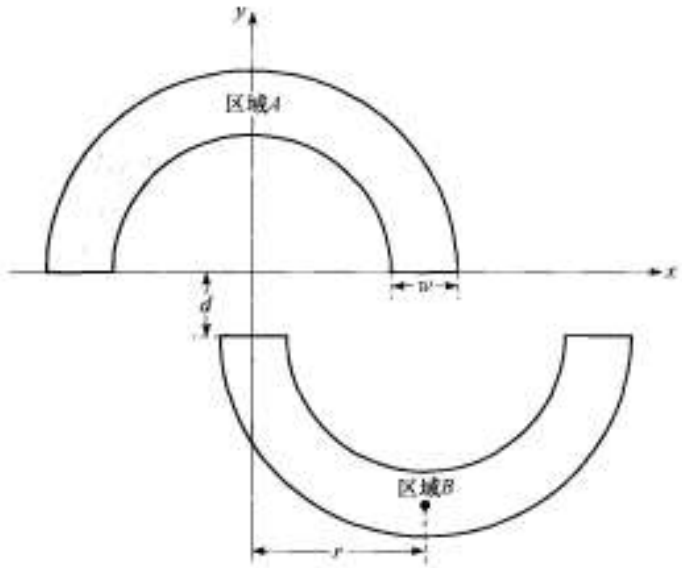


图 1.8　双月分类问题

实验

这里的实验所要采用的感知器参数如下所示：

$$\text{输入层大小} = 2$$
$$\text{权向量大小 } m = 20$$
$$\beta = 50;\text{参照式(1.17)}$$

学习率参数 η 线性地从 10^{-1} 下降到 10^{-5}。

权重被初始化为 0。

图 1.9 给出了 $d=1$ 时的实验结果，这相应于具有良好线性可分性的情况。图 1.9a 是学习曲线，描画了均方误差（MSE）和迭代次数之间的关系；该图显示出经过三步迭代算法就收敛了。图 1.9b 画出了经感知器算法训练后计算得到的决策边界，展示了对 2 000 个测试点良好的可分离性。

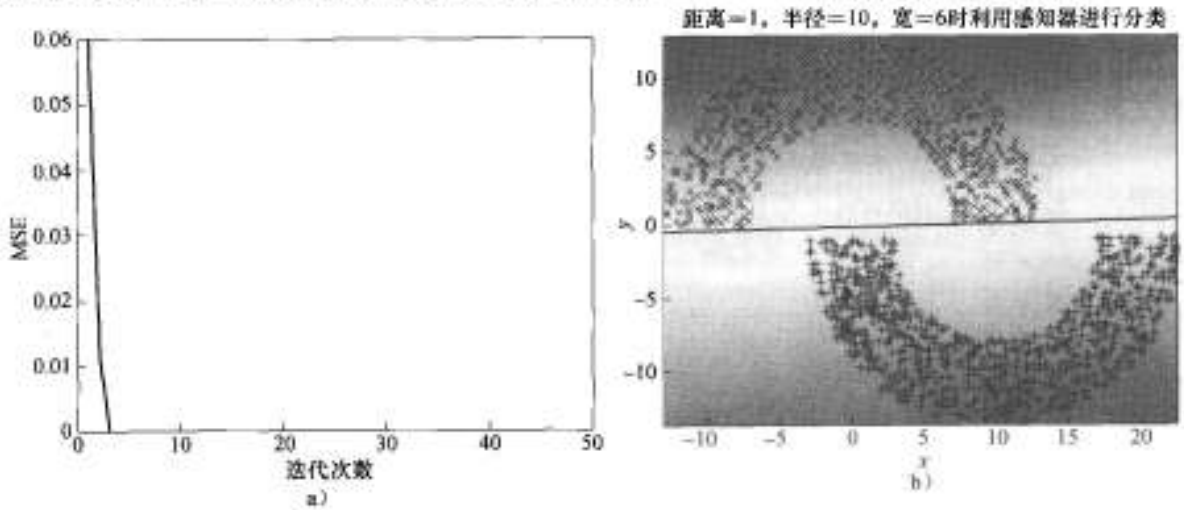


图 1.9　在距离 $d=1$ 时解双月集合的感知器：a）学习曲线；b）测试结果

在图 1.10 中，两个月亮之间的分离度被设为 $d=-4$，这个条件破坏了线性可分离性。图 1.10a 给出了学习曲线，从学习曲线的波动性可知感知器算法会持续波动，意味着算法的崩溃。这一结果也从图 1.10b 的图中得到了验证，其决策边界（通过训练得到的）和两个月亮相交，其误识别率为 $(186/2\,000)\times 100\%=9.3\%$。

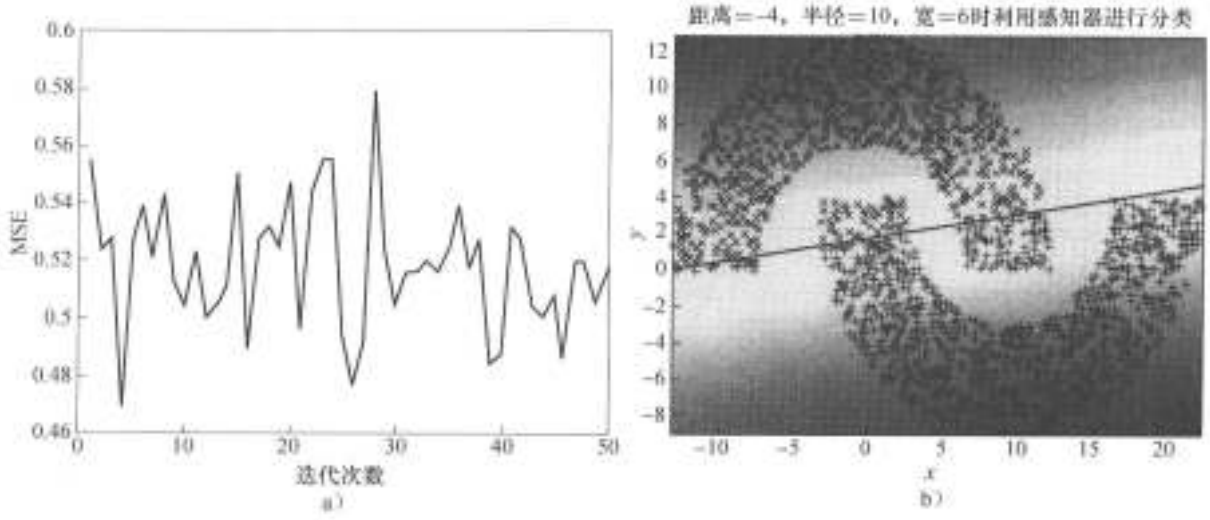


图 1.10 在距离 $d=-4$ 时解双月集合的感知器：a) 学习曲线；b) 测试结果

1.6 批量感知器算法

表 1.1 所总结的感知器收敛算法的推导没有考虑代价函数，而且，这一推导集中于单样本修正。本节我们将做如下两件事：

1. 介绍感知器代价函数的广义形式。
2. 利用代价函数来构成感知器收敛算法的批量版本。

我们想到的代价函数是允许应用梯度搜索的函数。具体而言，我们定义如下的感知器代价函数：

$$J(\mathbf{w})=\sum_{\mathbf{x}\in\mathcal{Y}}(-\mathbf{w}^{T}\mathbf{x}) \tag{1.39}$$

其中$\mathcal{Y}$是利用 $\mathbf{w}$ 作为其权值向量的感知器误识别的样本 $\mathbf{x}$ 的集合（Duda 等，2001）。如果所有样本都被正确识别，那么$\mathcal{Y}$为空，这种情况下代价函数 $J(\mathbf{w})$为 0。无论如何，代价函数 $J(\mathbf{w})$的一个优异特点是这一函数是关于权值向量 $\mathbf{w}$ 可微的。因而将 $J(\mathbf{w})$关于 $\mathbf{w}$ 微分可以产生梯度向量：

$$\nabla J(\mathbf{w})=\sum_{\mathbf{x}\in\mathcal{Y}}(-\mathbf{x}) \tag{1.40}$$

其中梯度算子为：

$$\nabla=\left[\frac{\partial}{\partial w_1},\frac{\partial}{\partial w_2},\cdots,\frac{\partial}{\partial w_m}\right]^{T} \tag{1.41}$$

在最速下降法中，算法的每一个时间步对权值向量 $\mathbf{w}$ 的修正都是在梯度向量$\nabla J(\mathbf{w})$的反方向作用的。相应地，算法具有如下的形式：

$$\mathbf{w}(n+1)=\mathbf{w}(n)-\eta(n)\nabla J(\mathbf{w})=\mathbf{w}(n)+\eta(n)\sum_{\mathbf{x}\in\mathcal{Y}}\mathbf{x} \tag{1.42}$$

这包括了感知器收敛算法的单样本修正版本作为其特殊情况。而且，式(1.42)包含了给定样本

集 $\mathbf{x}(1)$，$\mathbf{x}(2)$，…来计算权值向量的批量感知器算法。特别地，在时间步 $n+1$ 对权值向量的修正是通过根据权值向量 $\mathbf{w}(n)$ 而误识别的所有样本的和而来的，而这个和经由学习率参数 $\eta(n)$ 的调整。这一算法被称为“批量”是由于在算法的每一个时间步，一批误识别样本被用来计算权值向量的修正。

1.7　小结和讨论

感知器是一个单层神经网络，其操作是基于误差修正学习的。术语“单层”用在这里是为了表示网络计算层是由单个神经元组成的用于解决两类的分类问题。模式分类的学习过程需要经过一定次数的迭代然后终止。然而，为了成功实现分类，这些模式必须是线性可分的。

感知器的神经元使用 McCulloch-Pitts 模型。我们很容易提出这样一个问题：如果用一个 sigmoid 型非线性限制代替硬限幅器，感知器会不会有更好的表现？结果是不管我们使用硬限幅还是软限幅作为神经元模型中的非线性源，感知器稳定状态的决策特征基本不变（Shynk，1990；Shynk and Bershad，1991）。因此我们可以正式地说只要限制为由线性组合器和随后一个非线性元素组成的神经元模型，不管非线性使用什么形式，一个单层感知器都只能在线性可分模式上进行模式分类。

对于 Rosenblatt 感知器的第一个真正的批评是由 Minsky and Selfridge（1961）提出的。Minsky 和 Selfridge 指出，Rosenblatt 定义的感知器甚至都不能推广到二进制数的奇偶校验对的情况，更不用说完成一般的抽象。Rosenblatt 感知器的计算局限后来又在 Minsky 和 Papert 的名著《感知器》中得到了严格的数学证明（1969，1988）。在给出一些出色的和非常详细的对感知器的数学分析以后，Minsky 和 Papert 证明，建立在局部学习例子基础上的 Rosenblatt 感知器从本质上无法进行全局的泛化。在他们著作的最后一章，Minsky 和 Papert 推测他们发现的 Rosenblatt 感知器的局限性对它的变形——更具体说是多层神经网络也是成立的。下文摘录自他们著作（1969）的第 13.2 节：

> 尽管（甚至因为！）它具有严重的局限性，感知器仍然展示了其自身的研究价值。它有很多值得注意的特点：它的线性性；它迷人的学习理论；它清楚地作为一类并行计算范例的简单性。没有任何理由认为多层感知器仍然具有这些优点。靠直觉判断向多层系统推广是不会有好结果的，然而，证明（或否定）这一点仍是一个很重要的需要研究的问题。

这个结论在很大程度上导致了一直持续到 20 世纪 80 年代中期的对不仅是感知器也包括一般神经网络计算能力的严重怀疑。

但是历史已经证明，Minsky 和 Papert 作出的推测似乎是不太公正的，因为我们现在已经有很多神经网络和机器学习的高级形式，它们的计算能力比 Rosenblatt 感知器强得多。例如，第 4 章讨论的反向传播算法训练的多层感知器，第 5 章讨论的径向基函数网络，第 6 章讨论的支持向量机等，都以它们各自的方法克服了单层感知器的计算局限性。

在结束关于感知器的讨论时，我们可以断定感知器是一个用来对线性可分模式进行分类的精致的神经网络。其重要性不仅仅在于其历史价值，也在于其在线性可分模式分类方面的实际价值。

注释和参考文献

1. Rosenblatt 预想的原始感知器模型的网络组织（1962）有三种类型的单元：感知单元、联想单元和响应单元。感知单元和联想单元之间的连接有固定的权值，而联想单元和响应单元之间的连接具有变化的权值。联想单元被设计成一个从环境输入中抽取模式的预处理器。就仅关心可变权值而论，Rosenblatt 的原始感知器的运行与只有一个响应单元（即单个神经元）的情况是基本一致的。

2. 第 1.3 节关于感知器收敛算法的证明遵循 Nilsson（1965）的经典图书。

习题

1.1 证明总结感知器收敛算法的式(1.19)至式(1.22)是与式(1.5)和式(1.6)一致的。

1.2 假设图 1.1 中的感知器信号流图的硬限幅器被如下的 sigmoid 非线性函数所替代：

$$\varphi(v) = \tanh\left(\frac{v}{2}\right)$$

这里 v 是诱导局部域。感知器的分类决策定义如下：

如果输出 $y>\xi$ 则观察向量 $\mathbf{x}$ 属于类 $\mathscr{C}_1$，这里 ξ 是阈值；反之，$\mathbf{x}$ 属于类 $\mathscr{C}_2$。

证明如此构造的决策边界是一个超平面。

1.3 (a) 感知器可以用来执行很多逻辑函数。说明它对二进制逻辑函数与（AND）、或（OR）和非（COMPLEMENT）的实现过程。

(b) 感知器的一个基本局限是不能执行异或（EXCLUSIVE OR）函数。解释造成这个局限的原因。

1.4 考虑两个一维高斯分布类 $\mathscr{C}_1$ 和 $\mathscr{C}_2$，它们的方差都为 1。它们的均值为

$$\boldsymbol{\mu}_1 = -10$$

$$\boldsymbol{\mu}_2 = +10$$

这两个类本质上是线性可分的。设计一个分类器来分离这两个类。

1.5 式(1.37)和式(1.38)定义贝叶斯分类器在高斯环境下的权值向量和偏置。当协方差矩阵 $\mathbf{C}$ 由

$$\mathbf{C} = \sigma^2\mathbf{I}$$

定义时，求此分类器的构成。这里 σ^2 是常数，$\mathbf{I}$ 是单位矩阵。

计算机实验

1.6 重复 1.5 节的计算机实验，但是这一次将图 1.8 的两个月亮放到分隔边界处，即 $d=0$。计算在 2 000 个测试数据点上由算法产生的误识别率。

通过回归建立模型

本章组织

本章的主题是如何应用线性回归这一函数逼近的特殊形式对给定的随机变量集合建模。

本章的组织如下：

2.1 节是引言，2.2 节通过描述线性回归模型的数学框架建立本章余下部分的基础。

2.3 节导出线性回归模型参数向量的最大后验（maximum a posteriori，MAP）估计。

2.4 节是利用最小二乘法处理参数估计问题，并讨论这一方法和贝叶斯方法之间的关系。

2.5 节再次讨论第 1 章中讨论过的模式分类实验，这一次利用最小二乘法。

2.6 节讨论模型阶的选择问题。

2.7 节讨论参数估计中固定样本容量的推论，包括偏置-方差困境。

2.8 节介绍用工具变量（instrumental variable）概念来处理变量误差（errors-in-variables）问题。

最后是 2.9 节的小结和讨论。

2.1 引言

建模的思想在需要处理统计数据分析的每一个学科中都很有用。例如，假设给定一个随机变量集，要完成的任务是找到可能存在于这些变量之间的关系，如果这种关系存在的话。作为函数逼近的一种特殊形式，回归的典型方案如下：

- 对随机变量中的一个变量有着特别的兴趣；这一随机变量被称为依赖变量，或者响应（response）。
- 剩下的随机变量称为独立变量，或者回归量（regressor）。它们的作用是用来解释或者预测响应的统计行为。
- 响应对回归量的依赖还包括一个附加的误差项，用来说明在对依赖程度公式化时候的不确定性；误差项称为期望误差（expectational error）或解释误差（explanational error），这两个称呼是可以相互替换的。

这样的模型称为回归模型（regression model）[1]。

回归模型有两类：线性回归模型和非线性回归模型。在线性回归模型中，响应对回归量的依赖是通过线性函数定义的，这使得其统计分析从数学上来说是易处理的。另一方面，在非线性回归模型中，依赖性是由非线性函数定义的，因而其数学分析过程是困难的。本章将注意力集中于线性回归模型。在后续章节中将学习非线性回归模型。

本章对线性回归模型在数学上的易处理性是通过两个途径来说明的。首先，我们利用贝叶斯理论[2]来推导线性回归模型参数向量的最大后验估计。然后，使用另一个称为最小二乘法的方法来解决参数估计问题；而这一方法是由高斯在 19 世纪早期导出的。接着我们说明在高斯环境这一特殊情况下这两个方法之间的等价性。

2.2 线性回归模型：初步考虑

考虑图 2.1a 所示的情况，这里主要关注未知随机环境（unknown stochastic environment）。通过应用一组输入来探究这一环境，构成回归量：

$$\mathbf{x} = [x_1, x_2, \cdots, x_M]^T \tag{2.1}$$

其中上标 T 表示矩阵的转置。环境的输出结果用 d 来表示，构成了相应的响应，而响应被假设为标量，这仅是为了表述的方便。通常，我们不知道响应 d 和回归量 $\mathbf{x}$ 之间的函数依赖关系，因而我们提出一个线性回归模型，参数化为：

$$d = \sum_{j=1}^{M} w_j x_j + \varepsilon \tag{2.2}$$

其中 $w_1, w_2, \cdots, w_M$ 定义一组固定的但未知的参数，意味着环境是稳定的（stationary）。附加项 ε 表示模型的期望误差，表明对环境的未知量。图 2.1b 是关于式(2.2)所描述模型的输入-输出行为的信号流图。

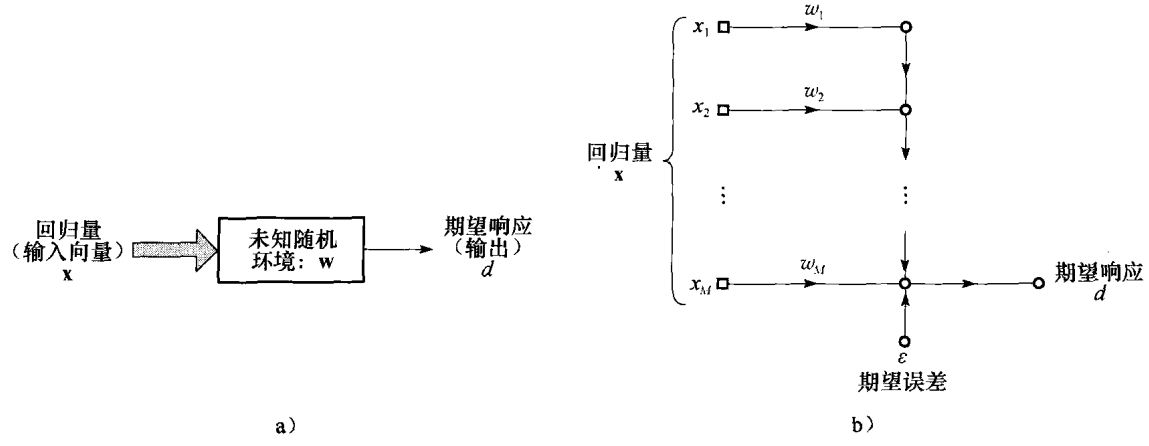


图 2.1　a）未知稳定随机环境；b）环境的线性回归模型

利用矩阵记号，将式(2.2)重写为下面的紧凑形式：

$$d = \mathbf{w}^T\mathbf{x} + \varepsilon \tag{2.3}$$

其中回归量 $\mathbf{x}$ 由式(2.1)中的元素来定义。相应地，参数向量 $\mathbf{w}$ 定义为

$$\mathbf{w} = [w_1, w_2, \cdots, w_M]^T \tag{2.4}$$

其维数和回归量 $\mathbf{x}$ 的维数相同；这一共同的维数称为模型阶（model order）。矩阵项 $\mathbf{w}^T\mathbf{x}$ 是向量 $\mathbf{w}$ 和 $\mathbf{x}$ 的内积。

由于是随机环境，回归量 $\mathbf{x}$、响应 d 以及期望误差 ε 是相应的随机向量 $\mathbf{X}$、随机变量 D 以及随机变量 E 的样本值（即单点实现）。有了这些随机集作为背景，感兴趣的问题可以像下面这样描述：

给定回归量 $\mathbf{X}$ 和相应的响应 D 的联合统计量，估计未知的参数向量 $\mathbf{w}$。

我们这里所说的联合统计量是指下面的统计参数集合：

- 回归量 $\mathbf{X}$ 的相关矩阵
- 期望响应 D 的方差
- 回归量 $\mathbf{X}$ 和期望响应 D 的互相关向量

假设 $\mathbf{X}$ 和 D 的均值都为 0。

第 1 章中讨论了贝叶斯推论的一个用于模式分类的重要方面。本章将讨论贝叶斯推论的另一个用于参数估计的方面。

2.3　参数向量的最大后验估计

贝叶斯方法提供了对式(2.3)的线性回归模型中参数向量 $\mathbf{w}$ 的选择过程中的非确定性进行

量化的一种高效方法。关于这一线性回归模型，应注意以下两点：

1. 回归量 $\mathbf{X}$ 充当“刺激”的角色，和参数向量 $\mathbf{w}$ 没有任何关系。

2. 关于未知参数向量 $\mathbf{W}$ 的信息仅仅包含在期望响应 D 中，而期望响应 D 扮演着环境“可观测量”的角色。

相应地，我们的注意力集中于 $\mathbf{W}$ 和 D 的联合概率分布密度函数，$\mathbf{X}$ 为条件。

将密度函数记为 $p_{\mathbf{W},D|\mathbf{X}}(\mathbf{w},d|\mathbf{x})$。由概率理论，我们知道密度函数可以表示为：

$$p_{\mathbf{W},D|\mathbf{X}}(\mathbf{w},d|\mathbf{x}) = p_{\mathbf{W}|D,\mathbf{X}}(\mathbf{w}|d,\mathbf{x})p_D(d) \tag{2.5}$$

也可以将之表达为等价形式：

$$p_{\mathbf{W},D|\mathbf{X}}(\mathbf{w},d|\mathbf{x}) = p_{D|\mathbf{W},\mathbf{X}}(d|\mathbf{w},\mathbf{x})p_{\mathbf{W}}(\mathbf{w}) \tag{2.6}$$

根据这两个等式，可以得到：

$$p_{\mathbf{W}|D,\mathbf{X}}(\mathbf{w}|d,\mathbf{x}) = \frac{p_{D|\mathbf{W},\mathbf{X}}(d|\mathbf{w},\mathbf{x})p_{\mathbf{W}}(\mathbf{w})}{p_D(d)} \tag{2.7}$$

其中 $p_D(d)\neq 0$。式(2.7)是贝叶斯定理的特殊形式；其中隐含了 4 个密度函数，如下所示：

1. 观测密度 (observation density)：这代表条件概率密度函数 $p_{D|\mathbf{W},\mathbf{X}}(d|\mathbf{w},\mathbf{x})$，表示给定参数向量 $\mathbf{w}$，由回归量 $\mathbf{x}$ 对环境响应 d 的“观测”。

2. 先验 (prior)：这代表概率密度函数 $p_{\mathbf{W}}(\mathbf{w})$，表示先验于环境观测量的参数向量 $\mathbf{w}$ 的信息。此后，先验被简单地记为 $\pi(\mathbf{w})$。

3. 后验密度 (posterior density)：这代表条件概率密度函数 $p_{\mathbf{W}|D,\mathbf{X}}(\mathbf{w}|d,\mathbf{x})$，表示对环境的观测完成之后的参数向量 $\mathbf{w}$。此后，后验密度记为 $\pi(\mathbf{w}|d,\mathbf{x})$。作为条件的响应-回归对 $(\mathbf{x},d)$ 是“观测模型”，包括由回归量 $\mathbf{x}$ 得到的环境的响应 d。

4. 证据 (evidence)：这代表概率密度函数 $p_D(d)$，表示用于统计分析的包含于响应 d 中的“信息”。

观测密度 $p_{D|\mathbf{W},\mathbf{X}}(d|\mathbf{w},\mathbf{x})$在数学上通常以似然函数的形式来表示，定义为：

$$l(\mathbf{w}|d,\mathbf{x}) = p_{D|\mathbf{W},\mathbf{X}}(d|\mathbf{w},\mathbf{x}) \tag{2.8}$$

而且，在所关心的参数向量 $\mathbf{w}$ 的估计的范围内，在式(2.7)右边的证据项 $p_D(d)$仅仅扮演着归一化常量的角色。于是，可以用如下语句来描述式(2.7)：

参数化回归模型的向量 w 的后验密度与似然函数及先验之间的积成正比。

即

$$\pi(\mathbf{w}|d,\mathbf{x}) \propto l(\mathbf{w}|d,\mathbf{x})\pi(\mathbf{w}) \tag{2.9}$$

其中符号$\propto$表示正比。

似然函数 $l(\mathbf{w}|d,\mathbf{x})$，从其自身考虑，提供了对于参数向量 $\mathbf{w}$ 的最大似然 (ML) 估计，如下式所示：

$$\mathbf{w}_{ML} = \arg\max_{\mathbf{w}} l(\mathbf{w}|d,\mathbf{x}) \tag{2.10}$$

然而，对于参数向量 $\mathbf{w}$ 的更深层次的估计，考虑后验密度 $\pi(\mathbf{w}|d,\mathbf{x})$。具体来说，我们通过下式来定义参数向量 $\mathbf{w}$ 的最大后验 (MAP) 估计：

$$\mathbf{w}_{MAP} = \arg\max_{\mathbf{w}} \pi(\mathbf{w}|d,\mathbf{x}) \tag{2.11}$$

我们说 MAP 估计器是比 ML 估计器更深层次的估计，这是基于如下两个原因：

1. 用于参数估计的贝叶斯范式，是根植于式(2.7)所示的贝叶斯定理，并通过式(2.11)的 MAP 估计器证明的，它采用了关于参数向量 $\mathbf{w}$ 的所有可能信息。与之相反，式(2.10)的 ML 估计器基于贝叶斯范式的极端 (fringe)，忽略了先验信息。

2. ML 估计器仅仅依赖于观测模型 $(d,\mathbf{x})$，因而可能导致非唯一解。为了加强解的唯一性

和稳定性，先验 $\pi(\mathbf{w})$必须被合并到估计器的规划中，这正是 MAP 估计器要做的。

当然，应用 MAP 估计器的难点在于如何找到合适的先验信息，这使得 MAP 比 ML 需要更多的计算量。

以下是最后的讨论。从计算的角度讲，我们发现往往采用后验密度的对数会比直接采用后验密度要方便。而对数是关于其自变量的单调递增函数，因而允许我们采取这一方案。相应地，可以将 MAP 估计器写成如下所期望的形式：

$$\mathbf{w}_{\text{MAP}} = \arg\max_{\mathbf{w}} \log(\pi(\mathbf{w}|d,\mathbf{x})) \tag{2.12}$$

其中 log 表示自然对数。对于 ML 估计器来说可采用相似的方法。

高斯环境下的参数估计

令 $\mathbf{x}_i$ 和 d_i 表示应用于环境的回归量和结果响应，相对应于作用于环境的第 i 次试验。令这一试验重复 N 次。然后将可用于参数估计的训练样本表示为：

$$\mathcal{T} = \{\mathbf{x}_i, d_i\}_{i=1}^{N} \tag{2.13}$$

为了实现参数估计任务，我们做如下假设：

假设 1：统计独立与同分布

构成训练样本的 N 个样例是统计独立与同分布的（independent and identically distributed，iid）。

假设 2：高斯性

产生训练样本$\mathcal{T}$的环境服从高斯分布。

更具体来说，式(2.3)中的线性回归模型的期望误差由均值为 0 及共同方差为 σ^2 的高斯密度函数来描述，如下式所示：

$$p_E(\varepsilon_i) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{\varepsilon_i^2}{2\sigma^2}\right), \quad i = 1,2,\cdots,N \tag{2.14}$$

假设 3：稳定性

环境是稳定的，这意味着在全部 N 次试验中参数向量 $\mathbf{w}$ 是固定的，但是未知的。

更具体来说，权值向量 $\mathbf{w}$ 的 M 个元素被假设为 iid 的，每一个元素由均值为 0 及共同方差为 σ_w^2 的高斯密度函数所决定。因而可以将参数向量 $\mathbf{w}$ 的第 k 个元素的先验表示为

$$\pi(w_k) = \frac{1}{\sqrt{2\pi}\sigma_w}\exp\left(-\frac{w_k^2}{2\sigma_w^2}\right), \quad k = 1,2,\cdots,M \tag{2.15}$$

对作用于环境的第 i 次试验重写式(2.3)，我们有

$$d_i = \mathbf{w}^T\mathbf{x}_i + \varepsilon_i, \quad i = 1,2,\cdots,N \tag{2.16}$$

其中 d_i，$\mathbf{x}_i$ 和 ε_i 分别是随机变量 D、随机向量 $\mathbf{X}$ 和随机变量 E 的相应的样本值（即单点实现）。

令$\mathbb{E}$定义统计期望算子。根据假设 2，我们有

$$\mathbb{E}[E_i] = 0, \quad \text{对于所有的 } i \tag{2.17}$$

和

$$\text{var}[E_i] = \mathbb{E}[E_i^2] = \sigma^2, \quad \text{对于所有的 } i \tag{2.18}$$

根据式(2.16)，对于给定的回归量 $\mathbf{x}_i$，

$$\mathbb{E}[D_i] = \mathbf{w}^T\mathbf{x}_i, \quad i = 1,2,\cdots,N \tag{2.19}$$

$$\text{var}[D_i] = \mathbb{E}[(D_i - \mathbb{E}[D_i])^2] = \mathbb{E}[E_i^2] = \sigma^2 \tag{2.20}$$

然后根据假设 2 的高斯含义，由式(2.14)，我们表述第 i 次试验的似然函数为

$$l(\mathbf{w}|d_i,\mathbf{x}_i) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{1}{2\sigma^2}(d_i - \mathbf{w}^T\mathbf{x}_i)^2\right), \quad i = 1,2,\cdots,N \tag{2.21}$$

然后，由假设 1，对于环境的 N 次试验具有 iid 特性，我们将试验中所有的似然函数表述为

$$l(\mathbf{w}|d,\mathbf{x}) = \prod_{i=1}^{N} l(\mathbf{w}|d_i,\mathbf{x}_i) = \frac{1}{(\sqrt{2\pi}\sigma)^N}\prod_{i=1}^{N}\exp\left(-\frac{1}{2\sigma^2}(d_i-\mathbf{w}^T\mathbf{x}_i)^2\right)$$

$$= \frac{1}{(\sqrt{2\pi}\sigma)^N}\exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{N}(d_i-\mathbf{w}^T\mathbf{x}_i)^2\right) \tag{2.22}$$

这表示包含于式(2.13)中的训练样本$\mathcal{T}$的关于权值向量 $\mathbf{w}$ 的全部试验知识。

仅剩的需要考虑的另一个信息源是包含于先验 $\pi(\mathbf{w})$中的信息。根据式(2.15)描述的 $\mathbf{w}$ 的第 k 个元素的 0 -均值高斯特性，以及在假设 3 下 $\mathbf{w}$ 的 M 个元素的 iid 特性，我们有

$$\pi(\mathbf{w}) = \prod_{k=1}^{M}\pi(w_k) = \frac{1}{(\sqrt{2\pi}\sigma_w)^M}\prod_{k=1}^{M}\exp\left(-\frac{w_k^2}{2\sigma_w^2}\right)$$

$$= \frac{1}{(\sqrt{2\pi}\sigma_w)^M}\exp\left(-\frac{1}{2\sigma_w^2}\sum_{i=1}^{M}w_k^2\right) = \frac{1}{(\sqrt{2\pi}\sigma_w)^M}\exp\left(-\frac{1}{2\sigma_w^2}\|\mathbf{w}\|^2\right) \tag{2.23}$$

其中 $\|\mathbf{w}\|$ 是未知参数向量 $\mathbf{w}$ 的欧几里得范数，由下式定义：

$$\|\mathbf{w}\| = \left(\sum_{k=1}^{M}w_k^2\right)^{1/2} \tag{2.24}$$

因而，将式(2.22)和式(2.23)代入式(2.9)，然后简化其结果，得到后验密度：

$$\pi(\mathbf{w}|d,\mathbf{x}) \propto \exp\left[-\frac{1}{2\sigma^2}\sum_{i=1}^{N}(d_i-\mathbf{w}^T\mathbf{x}_i)^2 - \frac{1}{2\sigma_w^2}\|\mathbf{w}\|^2\right] \tag{2.25}$$

现在可以应用式(2.12)的 MAP 公式来解决手头的估计问题。具体来说，将式(2.25)代入这一公式，我们有

$$\hat{\mathbf{w}}_{\text{MAP}}(N) = \max_{\mathbf{w}}\left[-\frac{1}{2}\sum_{i=1}^{N}(d_i-\mathbf{w}^T\mathbf{x}_i)^2 - \frac{\lambda}{2}\|\mathbf{w}\|^2\right] \tag{2.26}$$

其中引入了一个新的参数：

$$\lambda = \frac{\sigma^2}{\sigma_w^2} \tag{2.27}$$

现在定义二次函数：

$$\mathscr{C}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}(d_i-\mathbf{w}^T\mathbf{x}_i)^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2 \tag{2.28}$$

显然，最大化式(2.26)中相应于 $\mathbf{w}$ 的参数等价于最小化二次函数$\mathscr{C}(\mathbf{w})$。相应地，最优估计$\hat{\mathbf{w}}_{\text{MAP}}$可以通过将函数$\mathscr{C}(\mathbf{w})$对 $\mathbf{w}$ 微分并令其结果为 0 来获得。这样，可以获得如下的 $M\times 1$ 参数向量的期望 MAP 估计：

$$\hat{\mathbf{w}}_{\text{MAP}}(N) = [\mathbf{R}_{xx}(N)+\lambda\mathbf{I}]^{-1}\mathbf{r}_{dx}(N) \tag{2.29}$$

这里引入了两个矩阵和一个向量。

1. 回归量 $\mathbf{x}$ 的时间平均 $M\times M$ 相关矩阵，由下式定义

$$\hat{\mathbf{R}}_{xx}(N) = -\sum_{i=1}^{N}\sum_{j=1}^{N}\mathbf{x}_i\mathbf{x}_j^T \tag{2.30}$$

其中 $\mathbf{x}_i\mathbf{x}_j^T$ 是回归量 $\mathbf{x}_i$ 和 $\mathbf{x}_j$ 的外积，应用于环境的第 i 和第 j 次试验。

2. $M\times M$ 的单位矩阵 $\mathbf{I}$，其 M 个对角元素为 1，其他元素为 0。

3. 回归量 $\mathbf{x}$ 和期望响应 d 的时间平均 $M\times 1$ 互相关向量，由下式定义

$$\hat{\mathbf{r}}_{dx}(N) = -\sum_{j=1}^{N}\mathbf{x}_i d_i \tag{2.31}$$

相关$\hat{\mathbf{R}}_{xx}(N)$和$\hat{\mathbf{r}}_{dx}(N)$都是在整个训练样本$\mathcal{T}$的所有 N 个样例上的平均，因而这里使用了术语

“时间平均”。

假设我们给方差 σ_w^2 分配一个大的值，这样做的隐含效果是参数向量 $\mathbf{w}$ 的每个元素的先验分布，在可能的取值范围内，从本质上来说是一致的。在这一条件下，参数 λ 实质上是 0，式(2.29)退化为 ML 估计：

$$\hat{\mathbf{w}}_{\mathrm{ML}}(N) = \hat{\mathbf{R}}_{xx}^{-1}(N)\,\hat{\mathbf{r}}_{dx}(N) \tag{2.32}$$

这支持我们早先提出的观点：ML 估计器仅仅依赖于以训练样本$\mathcal{T}$为示例的观测模型，从线性回归的统计学的观点而言，方程

$$\hat{\mathbf{R}}_{xx}(N)\,\hat{\mathbf{w}}_{\mathrm{ML}}(N) = \hat{\mathbf{r}}_{dx}(N) \tag{2.33}$$

通常称为*法方程*（normal equation），ML 估计$\hat{\mathbf{w}}_{\mathrm{ML}}$当然是该方程的解。另一个感兴趣的话题是 ML 估计器是*无偏估计器*，对于一个无限大的训练样本集$\mathcal{T}$，我们发现从极限角度来说，如果回归量 $\mathbf{x}(n)$和响应 $d(n)$是从*联合遍历过程*（jointly ergodic processes）中取得的，那么 $\mathbf{w}_{\mathrm{ML}}$收敛于未知随机环境的参数向量 $\mathbf{w}$。这时候时间平均可以用总体平均来代替。在这一条件下，习题 4 中证明了

$$\lim_{N\to\infty} \hat{\mathbf{w}}_{\mathrm{ML}}(N) = \mathbf{w}$$

作为对比，式(2.29)所示的 MAP 估计器是有偏估计器，因而提醒我们做出如下陈述：

通过利用正则化（即引入先验知识）来改进最大似然估计器的稳定性，其最大后验估计器的结果是有偏的。

简单来说，在稳定性和偏差之间需要做出一个权衡。

2.4　正则最小二乘估计和 MAP 估计之间的关系

我们可以通过另一条途径来估计参数向量 $\mathbf{w}$，即关注*代价函数*$\mathscr{E}_0(\mathbf{w})$，该函数定义为对环境的 N 次试验的期望误差的平方和。具体来说，我们令

$$\mathscr{E}_0(\mathbf{w}) = \sum_{i=1}^{N} \varepsilon_i^2(\mathbf{w})$$

其中我们已经将 $\mathbf{w}$ 作为 ε_i 的自变量来强调这样的事实，即回归模型的不确定性是由向量 $\mathbf{w}$ 引起的。重新组织式(2.16)中的各项，我们有

$$\varepsilon_i(\mathbf{w}) = d_i - \mathbf{w}^T\mathbf{x}_i, \quad i = 1,2,\cdots,N \tag{2.34}$$

将这一等式代入$\mathscr{E}_0(\mathbf{w})$的表达式中得到

$$\mathscr{E}_0(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}(d_i - \mathbf{w}^T\mathbf{x}_i)^2 \tag{2.35}$$

这仅仅依赖于训练样本集$\mathcal{T}$。对 $\mathbf{w}$ 最小化代价函数将产生*一般最小二乘估计器*（ordinary least-squares estimator）的规则，和式(2.32)的最大似然估计是等价的，因而，明显有得到缺乏唯一性和稳定性解的可能性。

为了克服这一严重问题，习惯的做法是通过增加如下新的项对式(2.35)定义的代价函数进行扩展：

$$\mathscr{E}(\mathbf{w}) = \mathscr{E}_0(\mathbf{w}) + \frac{\lambda}{2}\|\mathbf{w}\|^2 = \frac{1}{2}\sum_{i=1}^{N}(d_i - \mathbf{w}^T\mathbf{x}_i)^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2 \tag{2.36}$$

这一表达式等价于由式(2.28)定义的函数。其中包含的欧几里得范数的平方 $\|\mathbf{w}\|^2$ 称为*结构正则化*（structural regularization）。相应地，标量 λ 称为*正则化参数*（regularization parameter）。

当 $\lambda=0$ 时，意味着我们对于由训练样本集$\mathcal{T}$所例证的观测模型有完全的信心。另一种极端

情况是 $\lambda=\infty$，意味着我们对于观测模型没有信心。实际上，正则化参数 λ 在这两个极端情况之间作出选择。

在多数情况下，对于预先给定的正则化参数 λ，正则最小二乘法的解是通过最小化式(2.36)定义的关于参数向量 w 的正则代价函数而得到的，它等价于式(2.29)的 MAP 估计。这个解称为正则最小二乘解（regularized least-squares (RLS) solution)。

2.5　计算机实验：模式分类

本节中，我们重复在第 1 章中学习过的模式分类问题的计算机实验，第 1 章中采用了感知器算法。如前所示，图 1.8 给出了提供训练数据和测试数据的双月结构。而这里采用最小二乘法来实现分类。

图 2.2 给出了在两月之间的分隔距离为 $d=1$ 时最小二乘算法的训练结果。图中给出了双月之间构造的决策边界。相应地由感知器算法在相同的设置 $d=1$ 时的结果在图 1.9 中给出。比较这两个图，可以观察到如下有趣的结果：

1. 这两个算法所构造的决策边界都是线性的，从直观上来说是令人满意的。最小二乘算法揭示了双月的位置之间彼此相关的不对称方式，像图 2.2 中正倾斜的决策边界那样。有趣的是，感知器算法完全忽视了这种不对称性，构造了和 x 轴平行的决策边界。

2. 对于分隔距离 $d=1$，双月是线性可分的。感知器算法对这个设置完美地完成了任务；而最小二乘法虽然发现了双月图的非对称特征，但对测试数据产生了误分类，带来了 0.8% 的分类误差。

3. 和感知器不同，最小二乘法一次性地计算决策边界。

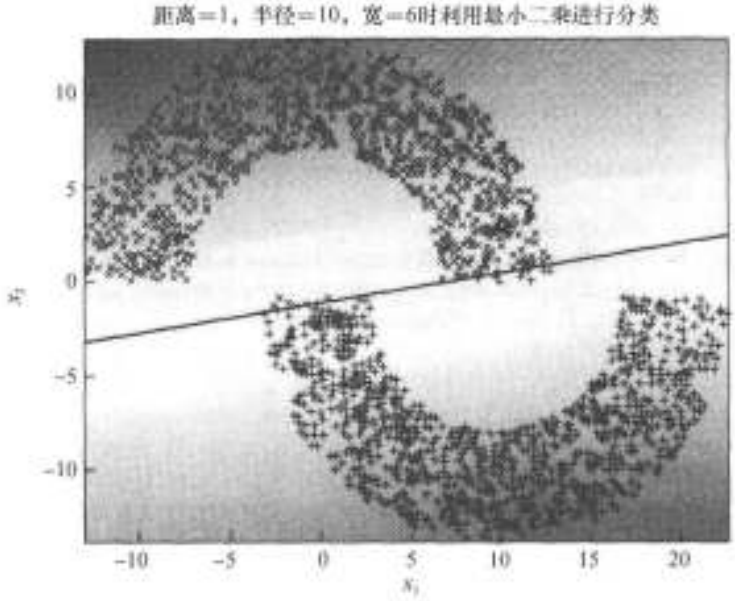


图 2.2　距离 $d=1$ 时对图 1.8 的双月的最小二乘分类

图 2.3 是对分隔距离为 $d=-4$ 时最小二乘法作用于双月模式的实验结果。如预料的那样，现在分类误差显著增加，达到了 9.5%。对于同样的设置，根据图 1.10 所报告的感知器算法存在 9.3% 的分类误差，两者比较，我们看到最小二乘法的分类性能略弱于感知器算法。

由 1.5 节和 2.5 节的模式分类计算机实验可以得到如下的重要结论：

尽管感知器和最小二乘算法都是线性的，它们在实现模式分类任务的时候其运行是不

同的。

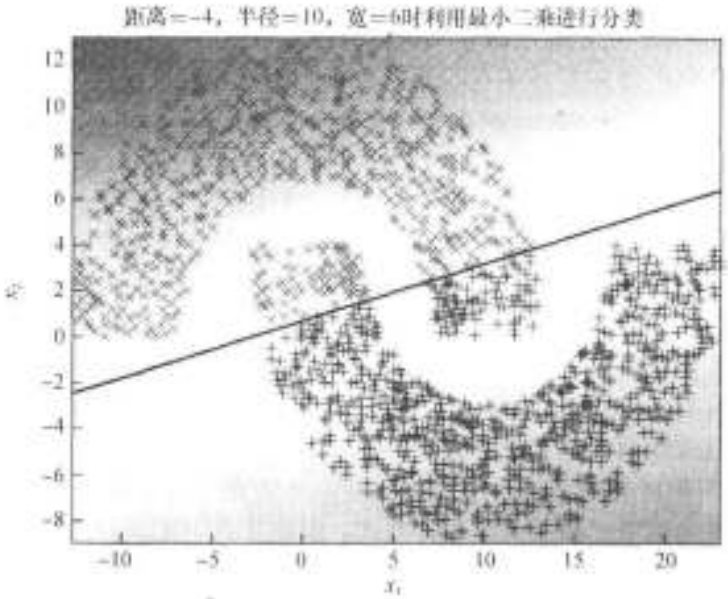


图 2.3 距离 $d=-4$ 时对图 1.8 的双月的最小二乘分类

2.6 最小描述长度原则

通过线性模型对随机过程进行描述可以用于合成和分析。对于合成（synthesis），我们通过给模型参数分配给定集的数值来产生希望的时间序列，并将之和均值为 0 方差预先给定的白噪（white noise）结合起来；这样获得的模型可以称为生成模型（generative model）。对于分析（analysis），从另一方面，我们对于固定长度的给定时间序列进行处理以估计模型参数，可以利用贝叶斯方法或者正则最小二乘法。由于估计是在统计的范围内，我们需要在模型和观测数据的匹配性上给出一个适当的测量。我们将这第二种问题称为模型选择（model selection）。例如，我们可能想要估计模型自由度（即可调整的参数）个数，甚至估计模型的一般结构。

统计学文献中提出了很多用于模型选择的方法，其中每个方法都有其自身的目标。由于这些方法的目标各不相同，因而在应用于同样的数据集合时，不同的方法会产生很大的不同是不奇怪的（Grünwald，2007）。

本节中，我们介绍一个充分证明了行之有效的方法，称为模型选择的最小描述长度（minimum-description-length，MDL）原则，这是由 Rissanen（1978）所做出的开创性工作。

MDL 原则发现的灵感可以追溯到柯尔莫哥洛夫复杂性理论（Kolmogorov complexity theory）。在这一值得注意的理论中，伟大的数学家柯尔莫哥洛夫定义了如下的复杂性（Kolmogorov，1965；Li and Vitányi，1993；Cover and Thomas，2006；Grünwald，2007）：

数据序列的算法（描述）复杂度是用于打印出这个序列然后终止的最短二进计算机程序的长度。

令人惊讶的是对于复杂度的定义没有采用概率分布的记号作为其基础，而是考虑计算机这一最通常形式的数据压缩机。

利用柯尔莫哥洛夫复杂度的基本概念，我们可以详尽阐述理想归纳推理理论（theory of idealized inductive inference），其目标是找到给定数据序列的“规律”（regularity）。将学习视

为尝试寻找“规律”的思想为 Rissanen 在表述 MDL 原则时提供了第一个观察。Rissanen 利用的第二个观察是规律本身可以等同于“压缩能力”。

因此，MDL 原则将两个观察组合在一起，一个观察是规律，另一个观察是压缩能力，从而将学习视为数据压缩，这反过来提示我们如下内容：

给定一个假设集合$\mathcal{H}$，一个数据序列 d，我们尝试寻找$\mathcal{H}$中的特定的假设或者$\mathcal{H}$中某些假设的组合来最大化地压缩数据序列 d。

这段话很简洁地总结了 MDL 原则是什么。这里的用于标识序列的符号 d 不要和前面用于表示期望响应的符号 d 相混淆。

文献中已经有 MDL 原则的多个版本。我们将集中讨论最古老的但却最简单、最著名的版本，称为*概率模型的简单两部分编码* MDL *原则*（simplistic two-part code MDL principle）。术语“简单”（simplistic）意思是所考虑的编码长度不是由优化方式决定。这里采用的术语“编码”（code）和“编码长度”（codelengths）是关于对数据序列按照最短或最小冗余（least redundant）方式编码的过程。

假设给定一个候选模型或者模型类$\mathcal{M}$。$\mathcal{M}$的所有元素都是概率源，后文将用 p 来表示点假设而不是用$\mathcal{H}$。特别地，我们寻找能够最好解释给定数据序列 d 的概率密度函数 $p\in\mathcal{M}$。两部分编码 MDL 原则告诉我们，寻找（点）假设 $p\in\mathcal{M}$使其最小化 p 的描述长度，我们将 p 的描述长度记为$L_1(p)$，在 p 的帮助下编码后的数据序列d 的描述长度记为$L_2(d|p)$。我们有和的形式：

$$L_{12}(p,d)=L_1(p)+L_2(d|p)$$

选择特别的点假设 $p\in\mathcal{M}$最小化 $L_{12}(p,\ d)$。

重要的是这里 p 本身也被编码。因而，在寻找最大地压缩数据序列 d 的假设的时候，必须按照这样的途径来编码（描述或压缩）数据，即解码器能够在甚至事先不知道假设的情况下恢复数据。这可以通过明确的编码假设来实现，如前述的两部分编码原则所述；这也可以通过完全不同的途径来实现——例如通过对假设进行平均（Grünwald，2007）。

模型阶选择

令 $\mathcal{M}^{(1)},\mathcal{M}^{(2)},\cdots,\mathcal{M}^{(k)},\cdots$，定义一组线性回归模型，其相应的参数向量为 $\mathbf{w}^k\in\mathcal{W}^k$，其中模型阶 $k=1,2,\cdots$；即权空间 $\mathcal{W}^{(1)},\mathcal{W}^{(2)},\cdots,\mathcal{W}^{(k)},\cdots$，是维数增加的。感兴趣的问题是确定能最好解释未知环境的模型，训练样本集 $\{\mathbf{x}_i,\ d_i\}_{i=1}^N$是产生于这个环境的，其中 $\mathbf{x}_i$ 为刺激，d_i 为相应的响应。我们刚刚描述的是模型阶选择问题（model-order selection problem）。

通过对组合长度 $L_{12}(p,\ d)$ 的统计特性的工作，两部分编码 MDL 原则告诉我们选择使下式最小的第 k 个模型：

$$\min_k\left\{\overbrace{-\log p(d_i|\mathbf{w}^{(k)})\pi\mathbf{w}^{(k)}}^{\text{误差项}}+\overbrace{\frac{k}{2}\log(N)+O(k)}^{\text{复杂度项}}\right\},\quad\begin{matrix}k=1,2,\cdots\\ i=1,2,\cdots,N\end{matrix}\tag{2.37}$$

其中 $\pi(\mathbf{w}^{(k)})$为参数向量 $\mathbf{w}^{(k)}$ 的先验分布，上式的最后一项是关于模型阶 k 的阶（Rissanen，1989；Grünwald，2007）。对于大的样本集大小 N，最后一项会被表达式的第二项$\frac{k}{2}\log(N)$所覆盖。式(2.37)的表达式通常分解为两项：

- *误差项*，记为$-\log(p(d_i|\mathbf{w}^{(k)})\pi(\mathbf{w}^{(k)})$，与模型以及数据有关。
- *复杂度项*，记为$\frac{k}{2}\log(N)+O(k)$，仅仅和模型有关。

实际上，对于不同的结果，应用式(2.37)时经常省略 $O(k)$项以简化问题。产生不同结果

的原因是 O(k)项可能相当大。然而对于线性回归模型，它能够明确有效地计算出来，其结果在实际中能够工作得非常好。

注意在式(2.37)的表达式中不采用先验分布 $\pi(\mathbf{w}^{(k)})$ 是由 Rissanen (1978) 首先提出的。

如果对于式(2.37)的表达式有不止一个最小化值，那么我们选择具有最小假设复杂项的那个模型。如果这样做仍然留下了多个候选模型，将不做额外的选择而只是用其中的一个来工作(Grünwald, 2007)。

MDL 原则的贡献

模型选择的 MDL 原则提供了两个重要的贡献 (Grünwald, 2007)：

1. 当有两个模型对于给定的数据序列匹配得一样好的时候，MDL 原则选择“最简单的”那个，即它允许利用对数据更短的描述。换句话说，MDL 原则实现了*奥卡姆剃刀* (Occam's razor) 的精确形式，奥卡姆剃刀说明了对简单理论的优先选择：

接受匹配数据的最简单解释

2. MDL 原则是*一致的* (consistent) 模型选择估计器，随着样本个数的增加，它收敛于真的模型阶。

也许最值得注意的是，包括 MDL 原则在内的几乎所有的应用，在文献中极少有关于不良特性的反常结果或模型的记录。

2.7　固定样本大小考虑

对于参数估计的最大似然估计或一般最小二乘法来说，其最大的局限在于解的非唯一性和不稳定性，这是由于完全依赖于观测模型（即训练样本$\mathcal{T}$）所导致的；刻画解的非唯一性和不稳定性在文献中也被称为*过拟合* (overfitting)。为了对这一实际问题进行更深入的探讨，考虑如下的一般回归模型：

$$d = f(\mathbf{x},\mathbf{w}) + \varepsilon \tag{2.38}$$

其中 $f(\mathbf{x},\mathbf{w})$ 为关于回归量 $\mathbf{x}$ 和模型参数 $\mathbf{w}$ 的确定函数，ε 是期望误差。由图 2.4a 所示的这一模型，是随机环境的数学描述，其目的在于解释或者预测由回归量 $\mathbf{x}$ 产生的响应 d。

图 2.4b 是环境的相应物理模型，其中$\hat{\mathbf{w}}$记为未知参数向量 $\mathbf{w}$ 的一个估计。第二个模型的目的是编码由训练样本$\mathcal{T}$表示的试验知识，如下式所示：

$$\mathcal{T} \rightarrow \hat{\mathbf{w}} \tag{2.39}$$

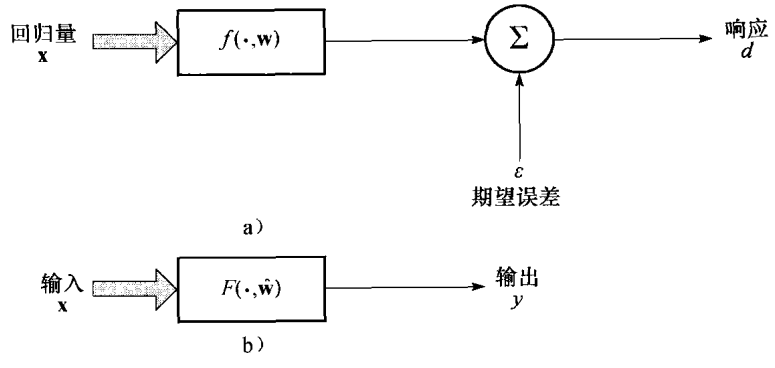


图 2.4　a) 随机环境的数学模型，其参数为向量 $\mathbf{w}$；b) 环境的物理模型，其中$\hat{\mathbf{w}}$是未知参数向量 $\mathbf{w}$ 的估计

实际上，物理模型提供了图 2.4a 所示的回归模型的一个*逼近* (approximation)。将响应于输入向量 $\mathbf{x}$ 所产生的物理模型的实际响应记为：

$$y = F(\mathbf{x},\hat{\mathbf{w}}) \tag{2.40}$$

其中 $F(\cdot,\hat{\mathbf{w}})$为由物理模型实现的输入输出函数；式(2.40)中的 y 是随机变量 Y 的一个样本值。给定式(2.39)的训练样本$\mathcal{T}$，估计器$\hat{\mathbf{w}}$是如下的代价函数的最小化值：

$$\mathscr{E}(\hat{\mathbf{w}}) = \frac{1}{2}\sum_{i=1}^{N}(d_i - F(\mathbf{x}_i,\hat{\mathbf{w}}))^2 \tag{2.41}$$

其中因子 1/2 是为了和前面的记号保持一致。除了尺度因子 1/2，代价函数$\mathscr{E}(\hat{\mathbf{w}})$ 是环境（期望）响应 d 和物理模型的实际响应 y 之间差的平方，在整个训练样本集$\mathcal{T}$上计算。

令符号$\mathbb{E}_{\mathcal{T}}$表示在整个训练样本集$\mathcal{T}$上所取的*平均算子* (average operator)。在平均算子$\mathbb{E}_{\mathcal{T}}$下的变量或其函数由 $\mathbf{x}$ 和 d 表示，$(\mathbf{x},d)$对表示训练样本$\mathcal{T}$中的一个例子。与之相比，统计期望算子$\mathbb{E}$作用于整个 $\mathbf{x}$ 和 b 的总体上，$\mathcal{T}$作为一个子集也包括在内。在下面要特别注意算子$\mathbb{E}$和$\mathbb{E}_{\mathcal{T}}$之间的区别。

受式(2.39)所示的变换的启发，可以将 $F(\mathbf{x},\hat{\mathbf{w}})$ 和 $F(\mathbf{x},\mathcal{T})$ 互换使用，从而将式(2.41)写成如下等价形式：

$$\mathscr{E}(\hat{\mathbf{w}}) = \frac{1}{2}\mathbb{E}_{\mathcal{T}}[(d - F(\mathbf{x},\mathcal{T}))^2] \tag{2.42}$$

通过对项 $(d-F(\mathbf{x},\mathcal{T})$ 增加然后减去 $f(\mathbf{x},\mathbf{w})$，然后利用式(2.38)，我们有

$$d - f(\mathbf{x},\mathcal{T}) = [d - f(\mathbf{x},\mathbf{w})] + [f(\mathbf{x},\mathbf{w}) - F(\mathbf{x},\mathcal{T})] = \varepsilon + [f(\mathbf{x},\mathbf{w}) - F(\mathbf{x},\mathcal{T})]$$

将这一表达式代入式(2.42)然后扩展开来，可以重写代价函数$\mathscr{E}(\hat{\mathbf{w}})$为如下的等价形式：

$$\mathscr{E}(\hat{\mathbf{w}}) = \frac{1}{2}\mathbb{E}_{\mathcal{T}}[\varepsilon^2] + \frac{1}{2}\mathbb{E}_{\mathcal{T}}[(f(\mathbf{x},\mathbf{w}) - F(\mathbf{x},\mathcal{T}))^2] + \mathbb{E}_{\mathcal{T}}[\varepsilon f(\mathbf{x},\mathbf{w}) - \varepsilon F(\mathbf{x},\mathcal{T})] \tag{2.43}$$

然而，式(2.43)右边最后的期望项是 0，基于如下两个原因：

- 期望误差 ε 和回归函数 $f(\mathbf{x},\mathbf{w})$之间是不相关的。
- 期望误差 ε 属于图 2.4a 所示的回归模型，而逼近函数 $F(\mathbf{x},\hat{\mathbf{w}})$属于图 2.4b 所示的物理模型。

相应地，式(2.43)简化为

$$\mathscr{E}(\hat{\mathbf{w}}) = \frac{1}{2}\mathbb{E}_{\mathcal{T}}[\varepsilon^2] + \frac{1}{2}\mathbb{E}_{\mathcal{T}}[(f(\mathbf{x},\mathbf{w}) - F(\mathbf{x},\mathcal{T}))^2] \tag{2.44}$$

式(2.44)右端的项$\mathbb{E}_{\mathcal{T}}[\varepsilon^2]$是期望（回归建模）误差 ε 的方差 (variance)，在整个训练样本集$\mathcal{T}$上评估；这里假设 ε 具有 0-均值。方差表示*固有误差* (intrinsic error)，因为它独立于估计$\hat{\mathbf{w}}$。因此，估计量$\hat{\mathbf{w}}$（代价函数$\mathscr{E}(\hat{\mathbf{w}})$的最小化值）也将最小化回归函数 $f(\mathbf{x},\mathbf{w})$和逼近函数 $F(\mathbf{x},\hat{\mathbf{w}})$之间距离平方的总体平均。换句话说，对于 $F(\mathbf{x},\hat{\mathbf{w}})$的效果的*自然测度* (natural measure) 是作为期望响应 d 的预测值，由下式定义（忽略了尺度因子 1/2）：

$$L_{av}(f(\mathbf{x},\mathbf{w}),F(\mathbf{x},\hat{\mathbf{w}})) = \mathbb{E}_{\mathcal{T}}[(f(\mathbf{x},\mathbf{w}) - F(\mathbf{x},\mathcal{T}))^2] \tag{2.45}$$

自然测度从根本上是非常重要的，因为它提供了在偏置和方差之间取得平衡的数学基础，而这两者是由利用 $F(\mathbf{x},\hat{\mathbf{w}})$ 作为 $f(\mathbf{x},\mathbf{w})$ 的逼近而产生的。

偏置-方差困境

由式(2.38)我们发现函数 $f(\mathbf{x},\mathbf{w})$ 等于条件期望$\mathbb{E}(d|\mathbf{x})$。因而可以将 $f(\mathbf{x})$和 $F(\mathbf{x},\hat{\mathbf{w}})$之间距离的平方重新定义如下

$$L_{av}(f(\mathbf{x},\mathbf{w}),F(\mathbf{x},\hat{\mathbf{w}})) = \mathbb{E}_{\mathcal{T}}[(\mathbb{E}[d|\mathbf{x}] - F(\mathbf{x},\mathcal{T}))^2] \tag{2.46}$$

这一表达式可以看成是在回归函数 $f(\mathbf{x},\mathbf{w})=\mathbb{E}[d|\mathbf{x}]$以及逼近函数 $F(\mathbf{x},\hat{\mathbf{w}})$之间估计误差的平均值，在整个训练样本集$\mathcal{T}$上评估。注意条件均值$\mathbb{E}[d|\mathbf{x}]$对训练样本集$\mathcal{T}$具有常期望。下面我们有

$$\mathbb{E}[d|\mathbf{x}]-F(\mathbf{x},\mathcal{T})=(\mathbb{E}[d|\mathbf{x}]-\mathbb{E}_{\mathcal{T}}[F(\mathbf{x},\mathcal{T})]+(\mathbb{E}_{\mathcal{T}}[F(\mathbf{x},\mathcal{T})])-F(\mathbf{x},\mathcal{T}))$$

这里只是简单地加上然后减去平均$\mathbb{E}_{\mathcal{T}}[F(\mathbf{x},\mathcal{T})]$。通过与根据式(2.42)导出式(2.43)相似的方式，我们将式(2.46)重新表示成两项之和的形式（见习题 2.5）：

$$L_{av}(f(\mathbf{x}),F(\mathbf{x},\mathcal{T}))=B^2(\hat{\mathbf{w}})+V(\hat{\mathbf{w}}) \tag{2.47}$$

其中 $B(\hat{\mathbf{w}})$和 $V(\hat{\mathbf{w}})$被分别定义为

$$B(\hat{\mathbf{w}})=\mathbb{E}_{\mathcal{T}}[F(\mathbf{x},\mathcal{T})]-\mathbb{E}[d|\mathbf{x}] \tag{2.48}$$

和

$$V(\hat{\mathbf{w}})=\mathbb{E}_{\mathcal{T}}[(F(\mathbf{x},\mathcal{T})-\mathbb{E}_{\mathcal{T}}[F(\mathbf{x},\mathcal{T})])^2] \tag{2.49}$$

现在可观察到两个重要结果：

1. 第一项 $B(\hat{\mathbf{w}})$是逼近函数 $F(\mathbf{x},\mathcal{T})$的平均值的偏置，根据回归函数 $f(\mathbf{x},\mathbf{w})=\mathbb{E}[d|\mathbf{x}]$来测量。因而，$B(\hat{\mathbf{w}})$表示由函数 $F(x,\hat{\mathbf{w}})$定义的物理模型不能够精确逼近回归函数 $f(\mathbf{x},\mathbf{w})=\mathbb{E}[d|\mathbf{x}]$。因此可以将偏置 $B(\hat{\mathbf{w}})$看成是逼近误差（approximation error）。

2. 第二项 $V(\hat{\mathbf{w}})$是逼近函数 $F(\mathbf{x},\mathcal{T})$ 的离散，在整个训练样本集$\mathcal{T}$上测量。因而，$V(\hat{\mathbf{w}})$表示关于回归函数 $f(\mathbf{x},\mathbf{w})$的包含于训练样本集$\mathcal{T}$中的试验知识的不充分性。因而可以将 $V(\hat{\mathbf{w}})$看成估计误差（estimation error）的显性表示。

图 2.5 图示了目标（期望）和逼近函数之间的关系，它说明估计误差（即偏置和离散）是如何累积的。为了达到好的总体性能，逼近函数 $F(\mathbf{x},\hat{\mathbf{w}})=F(\mathbf{x},\mathcal{T})$的偏置 $B(\hat{\mathbf{w}})$和离散 $V(\hat{\mathbf{w}})$都必须很小。

遗憾的是，我们发现在对有限容量的训练样本通过样例进行学习的复杂物理模型中，获取小偏置的代价是大的离散。对于任何物理模型而言，仅仅在训练样本容量无限大的时候，才有希望同时消除偏置和离散。相应地就有偏置-离散困境（bias-variance dilemma），其结果就是过慢的收敛速度（Geman 等，1992）。偏置-离散困境可以通过有目的地引入偏置来规避，这将使得消除或者显著减少离散成为可能。当然，我们必须保证在物理模型的设计中引入的偏置是无害的。例如从模式分类的角度来说，偏置是无害的是指仅仅在我们试图推断未预期类的回归时它对于均方误差有显著贡献。

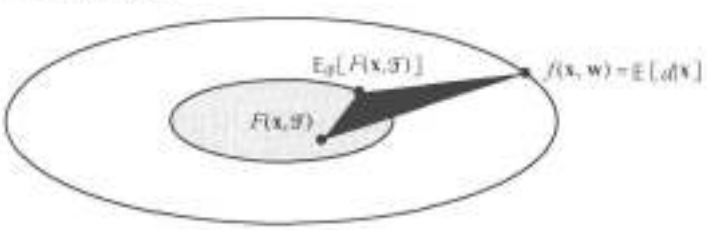


图 2.5　对于线性回归模型，由式(2.46)定义的自然测度 $L_{av}(f(\mathbf{x},\mathbf{w}),F(\mathbf{x},\hat{\mathbf{w}}))$被分解为偏置和离散项

关于图 2.5 的解释如下：

1. 图中带阴影的内部空间是外部空间的子集：
   外部空间表示回归函数 $f(\cdot,\mathbf{w})$的总体。
   内部空间表示逼近函数 $F(\cdot,\hat{\mathbf{w}})$的总体。
2. 图中有三个点，两个是固定的，一个是随机的：
   $\mathbb{E}[d|\mathbf{x}]$，固定点，外部空间上的平均
   $\mathbb{E}_{\mathcal{T}}[F(\mathbf{x},\mathcal{T})]$，第二个固定点，内部空间上的平均
   $F(\mathbf{x},\mathcal{T})$ 随机分布在内部空间内
3. 统计参数，包含在图中：

$B(\mathbf{w})$ = 偏置，表示$\mathbb{E}[d|\mathbf{x}]$和$\mathbb{E}_{\mathcal{T}}[F(\mathbf{x},\mathcal{T})]$之间的距离。

$V(\mathbf{w})$ = 离散，表示$F(\mathbf{x},\mathcal{T})$和$\mathbb{E}_{\mathcal{T}}[F(\mathbf{x},\mathcal{T})]$之间距离的平方，在训练样本集$\mathcal{T}$上平均。

$B^2(\mathbf{w})+V(\mathbf{w})$ = $F(\mathbf{x},\mathcal{T})$和$\mathbb{E}[d|\mathbf{x}]$之间距离的平方，在训练样本集$\mathcal{T}$上平均。

通常来说，必须对每个具体的应用设计偏置。达成这一目标的一个实际途径是利用约束(constrained)网络结构，这将比通用结构表现更优。

2.8　工具变量方法

在学习线性回归模型的时候，我们首先在 2.3 节从贝叶斯理论的观点作了讨论，然后在 2.4 节从最小二乘法的观点作了讨论。我们指出，两种方法都能产生图 2.1 所示的未知随机环境的参数向量 $\mathbf{w}$ 的相同解，即作为正则线性回归模型的式(2.29)和非正则版本的式(2.32)。这些公式在回归量（即输入信号）$\mathbf{x}$ 和期望响应 d 都无噪声的前提下根据高斯环境导出。然而，如果回归量 $\mathbf{x}$ 仅能在加性噪声的环境下观测，实际中会发生什么？也就是说，现在噪声回归量被定义为

$$\mathbf{z}_i = \mathbf{x}_i + \boldsymbol{v}_i \tag{2.50}$$

其中$\boldsymbol{v}_i$是伴随着训练样本集$\mathcal{T}$中第 i 次实现的观测 $\mathbf{x}_i$ 的噪声的测量。如果应用式(2.32)的非正则公式，将获得未知随机环境的参数向量 $\mathbf{w}$ 的修正解：

$$\hat{\mathbf{w}}_{\mathrm{ML}} = \hat{\mathbf{R}}_{zz}^{-1}\,\hat{\mathbf{r}}_{dz} \tag{2.51}$$

其中$\hat{\mathbf{R}}_{zz}$是噪声回归量 $\mathbf{z}$ 的时间平均相关函数，$\hat{\mathbf{r}}_{dz}$是相应的期望响应 d 和 $\mathbf{z}$ 的时间平均互相关函数。为了简化问题，我们忽略了这两个相关函数对于训练样本容量的依赖性。假设测量噪声向量 $\mathbf{v}$ 是白噪声，其均值为 0 且相关矩阵为 $\sigma_v^2\mathbf{I}$，其中 $\mathbf{I}$ 是单位矩阵，我们得到下面的相关函数：

$$\hat{\mathbf{R}}_{zz} = \hat{\mathbf{R}}_{xx} + \sigma_v^2\mathbf{I}$$

和

$$\hat{\mathbf{r}}_{dz} = \hat{\mathbf{r}}_{dx}$$

相应地，最大似然估计器假设为下面的新形式

$$\hat{\mathbf{w}}_{\mathrm{ML}} = (\hat{\mathbf{R}}_{xx} + \sigma_v^2\mathbf{I})^{-1}\,\hat{\mathbf{r}}_{dx} \tag{2.52}$$

从数学上来说，这个式子等价于式(2.29)的 MAP 公式，其正则化参数 λ 被设置为等于噪声方差 σ_v^2。这一观察使得我们可以作如下的陈述：

在回归量 $\mathbf{z}$ 中存在的加性噪声（具有合适的噪声方差）具有稳定最大似然估计器的有益效果，但是以给解引入偏置为代价。

这是个很具有讽刺意味的陈述：附加的噪声扮演了正则器（稳定器）的角色！

然而，假设需要的是对未知参数向量 $\mathbf{w}$ 产生的解是*渐近无偏的*（asymptotically unbiased）。在这种情形下，我们可以求助于*工具变量方法*（Young，1984）。这种方法依赖于引入工具变量集，表示为向量$\hat{\mathbf{x}}$，和噪声回归量 $\mathbf{z}$ 具有相同的维数，且满足下述两个性质：

性质 1　工具向量$\hat{\mathbf{x}}$和无噪回归量 $\mathbf{x}$ 之间是高度相关的，如下式表示：

$$\mathbb{E}[x_j\hat{x}_k] \neq 0 \quad ,\text{对所有 } j \text{ 和 } k \tag{2.53}$$

其中，x_j 是无噪回归量 $\mathbf{x}$ 的第 j 个元素，$\hat{x}_k$ 是工具向量$\hat{\mathbf{x}}$的第 k 个元素。

性质 2　工具向量$\hat{\mathbf{x}}$和测量噪声向量 $\mathbf{v}$ 是统计独立的，如下式所示：

$$\mathbb{E}[v_j\hat{x}_k] = 0 \quad ,\text{对所有 } j \text{ 和 } k \tag{2.54}$$

有了满足上面两个性质的工具向量$\hat{\mathbf{x}}$，我们来计算下面的相关函数：

1. 噪声回归量 $\mathbf{z}$ 和工具向量$\hat{\mathbf{x}}$是相关的，得到互相关矩阵：

$$\hat{\mathbf{R}}_{z\hat{x}} = \sum_{i=1}^{N} \hat{\mathbf{x}}_i \mathbf{z}_i^T \tag{2.55}$$

其中 $\mathbf{z}_i$ 是噪声训练样本 $\{\mathbf{z}_i, d_i\}_{i=1}^N$ 中的第 i 个回归量，$\hat{\mathbf{x}}_i$ 是相应的工具向量。

2. 期望的响应 d 和工具向量$\hat{\mathbf{x}}$是相关的，得到互相关向量：

$$\hat{\mathbf{r}}_{d\hat{x}} = \sum_{i=1}^{N} \hat{\mathbf{x}}_i d_i \tag{2.56}$$

给定这两个相关度量，我们利用修正公式：

$$\hat{\mathbf{w}}(N) = \mathbf{R}_{z\hat{x}}^{-1} \mathbf{r}_{d\hat{x}} = \left(\sum_{i=1}^{N} \hat{\mathbf{x}}_i \mathbf{z}_i^T \right)^{-1} \left(\sum_{i=1}^{N} \hat{\mathbf{x}}_i d_i \right) \tag{2.57}$$

来计算对于未知参数向量 $\mathbf{w}$ 的一个估计（Young，1984）。和式(2.51)的 ML 解不同，式(2.57)的修正公式基于工具变量方法，对未知参数向量 $\mathbf{w}$ 提供了一个渐近无偏估计；参考习题 2.7。

然而，在利用工具变量方法的时候，关键问题是如何获取或者产生满足性质 1 和 2 的变量。结果在时间序列分析中，关于这一问题的解非常直接，这有些出乎意料。

2.9　小结和讨论

本章学习了在统计学文献中已经很好地建立起来的线性回归的最小二乘法。我们是从两个不同但互补的观点展开学习的：

- *贝叶斯理论*，这里感兴趣的目标是对一组未知的参数作最大后验估计。这一参数估计方法需要对于未知参数的先验分布知识。这里是关于高斯环境的说明。
- *正则理论*，这里用于最小化的对未知参数的代价函数包含两部分：在整个训练数据上的平方解释误差和由参数向量的平方欧几里得范数定义的正则项。

对于如下特殊环境，即未知参数的先验分布服从均值为 0，方差为 σ_w^2 的高斯分布，正则参数 λ 和 σ_w^2 是成反比的。这意味着当 σ_w^2 很大的时候（即未知参数在很广的范围内一致分布），用于寻找参数向量 $\mathbf{w}$ 的估计的公式可以由法方程（normal equation）定义：

$$\hat{\mathbf{w}} = \hat{\mathbf{R}}_{xx}^{-1} \hat{\mathbf{r}}_{dx}$$

其中$\hat{\mathbf{R}}_{xx}$是关于向量 $\mathbf{x}$ 的时间平均相关矩阵，$\hat{\mathbf{r}}_{dx}$是相应的输入向量 $\mathbf{x}$ 和期望响应 d 之间的时间平均互相关向量。两个相关参数都是利用训练样本 $\{\mathbf{x}_i, d_i\}_{i=1}^N$ 来计算，因而依赖于样本容量 N。进一步说，如果假设其先验是一致分布的话，这一公式和最大似然估计法获得的解是等价的。

我们还讨论了其他三个重要的问题：

- 用于模型阶选择（即线性回归模型中未知参数向量的大小）的最小描述长度（MDL）准则。
- 偏置-离散困境，这意味着在参数估计（包含利用有限样本容量）时不可避免地会遇到在估计的离散和偏置之间寻找平衡的任务；偏置定义为参数估计的期望值和实际值之间的偏差，而离散是对期望值四周估计的“变更度”（volatility）的度量。
- 工具变量方法，当训练样本观测是有噪声的时候就需要用到这一方法；在实际中已知会遇到这样的情况。

注释和参考文献

1. 回归模型可以是线性的也可以是非线性的。在 Rao（1973）的经典图书中对线性回归模型进行了深入讨论。

Seber 和 Wild（1989）讨论了非线性回归模型。

2. 具有高度可读性的贝叶斯理论方面的资料，参考 Robert（2001）。
3. 对于最小二乘法的细节方面的讨论，参考 Haykin（2002）的第 8 章。

习题

2.1 讨论线性回归模型中参数向量的最大后验估计和最大似然估计之间的基本差别。

2.2 从式(2.36)的代价函数$\mathscr{E}(\mathbf{w})$开始，通过对未知参数向量 $\mathbf{w}$ 最小化代价函数来推导式(2.29)。

2.3 基于图 2.1 的线性回归模型给出最小二乘估计器的性质：

性质 1　最小二乘估计

$$\hat{\mathbf{w}} = \mathbf{R}_{xx}^{-1}\mathbf{r}_{dx}$$

是无偏的，如果图 2.1 的线性回归模型中期望误差 ε 具有 0 -均值。

性质 2　当期望误差 ε 是从一个均值为 0 方差为 σ^2 的白噪中产生的话，最小二乘估计$\hat{\mathbf{w}}$的协方差矩阵等于

$$\sigma^2\,\hat{\mathbf{R}}_{xx}^{-1}.$$

性质 3　估计误差

$$e_o = d - \hat{\mathbf{w}}^T\mathbf{x}$$

产生于最小二乘优化方法，和期望响应的估计（表示为 d）是正交的；这一性质是正交性原理（principle of orthogonality）的必然结果。如果采用 d，$\hat{d}$，e_o 的几何表示，我们将发现表示 e_o 的向量，是垂直于（即法于）表示 d 的向量的。正是受这一几何表示的启发，下面的式子才称为法方程：

$$\hat{\mathbf{R}}_{xx}\,\hat{\mathbf{w}} = \hat{\mathbf{r}}_{dx}$$

从法方程开始，在$\hat{\mathbf{R}}_{xx}$和$\hat{\mathbf{r}}_{dx}$为时间平均相关函数的前提下证明这三个性质。

2.4 令 $\mathbf{R}_{xx}$ 表示回归量 $\mathbf{x}$ 的总体平均相关函数，且令 $\mathbf{r}_{dx}$ 表示相应的回归量 $\mathbf{x}$ 和响应 d 之间的总体平均互相关向量；即

$$\mathbf{R}_{xx} = \mathbb{E}[\mathbf{x}\mathbf{x}^T]$$
$$\mathbf{r}_{dx} = \mathbb{E}[d\mathbf{x}]$$

参考式(2.3)的线性回归模型，证明最小化均方误差

$$\mathbf{J}(\mathbf{w}) = \mathbb{E}[\varepsilon^2]$$

导致 Wiener-Hopf 方程

$$\mathbf{R}_{xx}\mathbf{w} = \mathbf{r}_{dx}$$

其中 $\mathbf{w}$ 是回归模型的参数向量。比较这一方程和式(2.23)的法方程。

2.5 式(2.47)表示逼近函数 $F(\mathbf{x},\hat{\mathbf{w}})$ 作为期望响应 d 的预测器的有效性的自然测度。这一表达式由两个分量组成，一个定义平方偏置，另一个定义离散。从式(2.46)推导这一表达式。

2.6 详细阐述下面的陈述：

通过编入先验知识而约束的网络结构，以增加偏置为代价来减少离散，从而处理偏置-离散困境。

2.7 式(2.57)描述的工具变量法提供了对未知参数向量$\hat{\mathbf{w}}(N)$的渐近无偏估计；即

$$\lim_{N\to\infty}\hat{\mathbf{w}}(N) = \mathbf{w}$$

证明这句话的正确性，假设回归量 $\mathbf{x}$ 和响应 d 是联合遍历的。

计算机实验

2.8 重复 2.5 节中所描述的模式分类实验，这一次将两个月亮设为恰好线性可分，即 $d=0$。对你的结果作评论，并且将之和习题 1.6 中用感知器所获得的结果相比较。

2.9 在 2.5 节和习题 2.8 的实验中，没有对最小二乘法进行正则化。如果采用正则化的话会不会对最小二乘法的性能产生影响？

为了证实你对这个问题的回答，重复习题 2.8 的实验，这一次利用正则最小二乘法来做。

第3章
Neural Networks and Learning Machines, 3E

最小均方算法

本章组织

本章介绍一个非常流行的在线学习算法，称之为最小均方（least-mean-square，LMS）算法，它是由 Widrow 和 Hoff 在 1960 年提出的。

本章的组织如下：

3.1 节的引言部分以及随后的 3.2 节通过关于有限脉冲响应的线性离散时间滤波器的讲述，为本章剩余部分建立了基础。

3.3 节回顾两个无约束最优化技术：最速下降法和牛顿法。

3.4 节介绍维纳（Wiener）滤波器，在最小均方误差意义下它是最优的。一般来说，LMS 算法的平均性能是通过维纳滤波器来判断的。

3.5 节介绍 LMS 算法的推导。3.6 节提供一个作为马尔可夫模型的 LMS 算法的修正形式。然后，为研究 LMS 算法的收敛行为作准备，3.7 节介绍来自于非稳定热力学的朗之万（Langevin）方程。算法的收敛分析所必需的另一个工具是关于直接平均的 Kushner 方法；这一方法在 3.8 节中讨论。3.9 节中介绍算法的详细的统计分析；更重要的是，能够证明算法的统计行为（使用小的学习率参数）实际上是朗之万方程的离散时间版本。

3.10 节用计算机实验来评估 LMS 算法的小学习率理论。3.11 节重复 1.5 节利用感知器进行的模式分类实验，而这一节采用的是 LMS 算法。

3.12 节讨论 LMS 算法的优点和局限性。3.13 节讨论关于学习率退火方案的相关问题。

3.1 引言

第 1 章所讨论的 Rosenblatt 感知器是解决线性可分模式分类问题的第一个学习算法。而由 Widrow 和 Hoff(1960) 提出的最小均方算法（LMS）是第一个解决如预测和信道均等化等问题的线性自适应滤波算法。LMS 算法的提出是受到了感知器的启发。尽管从应用上来说有所不同，这两个算法之间有一个共同的特征：它们都用到了线性组合器（linear combiner），因而其名称是“线性”的。

- 令人惊奇的是，LMS 算法自身不仅仅可以作为自适应滤波应用机器，它还可以作为其他自适应滤波算法的评价准则。这里面的原因是多方面的：
- 从计算复杂度来说，对于可调参数而言 LMS 算法的复杂度是线性的，这使得算法是计算高效（computationally efficient）的，而算法从性能上来说依然是有效的。
- 算法可以简单地用代码来实现，因而是容易建立的。
- 最重要的是，对于外部扰动来说，算法是鲁棒的。

从工程的角度来说，上述性能都是非常需要的。因而 LMS 算法能够经受住时间的考验就一点也不奇怪了。

本章中，我们推导 LMS 算法的最基本形式并讨论其优点和局限。更重要的是，这里所讨论的素材为下一章将要讨论的反向传播算法提供了基础素材。

3.2 LMS 算法的滤波结构

图 3.1 是一个未知动态系统的方框图，由包含元素 $x_1(i), x_2(i), \cdots, x_M(i)$ 的输入向量所刺激，其中 i 是刺激（激励）应用于系统时的瞬间时间。时间索引 $i = 1, 2, \cdots, n$。作为对刺激的

响应，系统产生一个输出 $y(i)$作为响应。因此，此系统的外部行为由下述数据集描述：

$$\mathcal{T}:\{\mathbf{x}(i),d(i);i=1,2,\cdots,n,\cdots\} \tag{3.1}$$

其中

$$\mathbf{x}(i)=[x_1(i),x_2(i),\cdots x_M(i)]^T \tag{3.2}$$

组成$\mathcal{T}$的样本对根据一个未知概率法则是同分布的。输入向量 $\mathbf{x}(i)$的维数 M 称为输入空间的维数（dimensionality of the input space），或简称为输入维数（input dimensionality）。

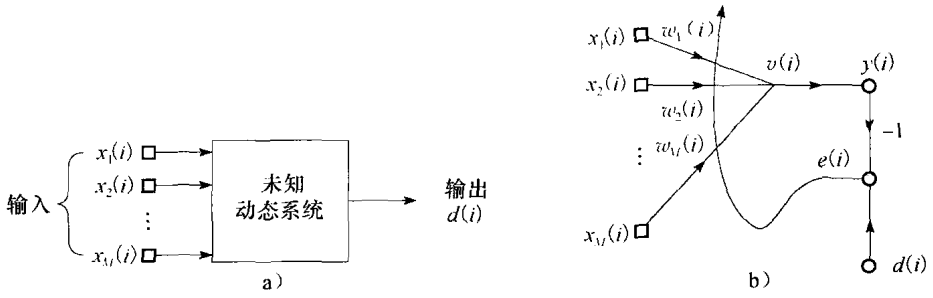


图 3.1　a）未知动态系统；b）系统自适应模型的信号流图

刺激向量 $\mathbf{x}(i)$能够以两种根本不同的方式出现，一种是空间的，另一种是时间的：

- $\mathbf{x}(i)$的 M 个元素代表空间中的不同点，在这种情况下我们称 $\mathbf{x}(i)$为数据的瞬像（snapshot）。
- $\mathbf{x}(i)$的 M 个元素代表在时间上均匀分布的某个刺激的现在和（$M-1$）个过去的值组成的集合。

我们面对的问题是如何通过建立一个简单线性神经元来设计未知动态系统的一个多输入-单输出模型。这个神经元模型是在一个算法的影响下运行的，此算法控制对神经元的突触权值的必要调整，同时记住以下要点：

- 此算法从任意设定的一个神经元突触权值开始。
- 为响应系统行为的统计变化，突触权值的调整是建立在连续的基础之上的（即把时间加进算法中）。
- 调整突触权值的计算在长度为一个采样周期的时间段里完成。

这样描述的神经元模型称为自适应滤波器（adaptive filter）。虽然是在作为系统辨识（system identification）的任务背景下给出的描述，但自适应滤波器的特征还是具有很广泛的应用。

图 3.1b 是一个自适应滤波器的示意图，它的运行由两个连续过程组成：

1. 过滤过程，涉及两个信号计算：
   - 一个输出，记为 $y(i)$，它被产生以响应刺激向量 $\mathbf{x}(i)$的 M 个元素，即 $x_1(i),x_2(i),\cdots,x_M(i)$。
   - 一个误差信号，记为 $e(i)$，它是通过比较输出 $y(i)$和未知系统的相应输出 $d(i)$而获得的。从效果上讲，$d(i)$可作为一个期望响应信号（desired response）或者目标（target）信号。
2. 自适应过程，包括根据误差 $e(i)$对神经元突触权值的自动调整。

因此，这两个共同运作过程的组合构成了一个围绕神经元运作的反馈环（feedback loop），如图 3.1b 所示。

因为神经元是线性的，输出 $y(i)$恰为诱导局部域 $v(i)$，即

$$y(i)=v(i)=\sum_{k=1}^{M}w_k(i)x_k(i) \tag{3.3}$$

其中 $w_1(i), w_2(i), \cdots, w_M(i)$ 表示在时刻 i 神经元的 M 个突触权值。利用矩阵形式可以把 $y(i)$ 表示为向量 $\mathbf{x}(i)$ 和 $\mathbf{w}(i)$ 的内积形式：

$$y(i) = \mathbf{x}^T(i)\mathbf{w}(i) \tag{3.4}$$

这里

$$\mathbf{w}(i) = [w_1(i), w_2(i), \cdots w_M(i)]^T$$

注意这里突触权值的记号已被简化，它不包括附加的标识神经元的下标，因为我们只需要处理单个神经元。当只需要考虑单个神经元时，全书都采用这种记号。神经元的输出 $y(i)$ 要与未知系统在时刻 i 的相应输出 $d(i)$ 作比较。通常，$y(i)$ 与 $d(i)$ 不等；因此它们的比较结果得到了误差信号：

$$e(i) = d(i) - y(i) \tag{3.5}$$

误差信号 $e(i)$ 用来对神经元突触权值的调整进行控制的方式是由用于导出自适应滤波算法的代价函数决定的。这个问题与最优化紧密相关。因此可以回顾一下无约束最优化方法。这些材料不仅可以应用在线性自适应滤波器上，还可以应用在一般的神经网络上。

3.3　无约束最优化：回顾

考虑代价函数 $\mathscr{E}(\mathbf{w})$，它是一个对未知权值（参数）向量 $\mathbf{w}$ 连续可微（continuously differentiable）的函数。函数 $\mathscr{E}(\mathbf{w})$ 映射 $\mathbf{w}$ 的元素为实数。它是一种度量，用来选择自适应滤波算法的权值（参数）向量 $\mathbf{w}$ 使得它以最优方式运行。我们想找到一个最优解 $\mathbf{w}^*$ 满足条件

$$\mathscr{E}(\mathbf{w}^*) \leqslant \mathscr{E}(\mathbf{w}) \tag{3.6}$$

也就是说，需要解决一个无约束的优化问题，即

选择适当的权值向量 $\mathbf{w}$ 最小化代价函数 $\mathscr{E}(\mathbf{w})$。

最优性的必要条件是：

$$\nabla\mathscr{E}(\mathbf{w}^*) = \mathbf{0} \tag{3.7}$$

这里 ∇ 是*梯度算子*（gradient operator），

$$\nabla = \left[\frac{\partial}{\partial w_1}, \frac{\partial}{\partial w_2}, \cdots, \frac{\partial}{\partial w_M}\right]^T \tag{3.8}$$

同时 $\nabla\mathscr{E}(\mathbf{w})$ 是代价函数的*梯度向量*（gradient vector），

$$\nabla\mathscr{E}(\mathbf{w}) = \left[\frac{\partial\mathscr{E}}{\partial w_1}, \frac{\partial\mathscr{E}}{\partial w_2}, \cdots, \frac{\partial\mathscr{E}}{\partial w_M}\right]^T \tag{3.9}$$

（对于向量的微分在本章结束部分的注释 1 中讨论）。

一类特别适合自适应滤波器设计的无约束最优化算法是以局部*迭代下降*（iterative descent）思想为基础的：

从一个初始估计值 $\mathbf{w}(0)$ 开始，产生一系列权值向量 $\mathbf{w}(1), \mathbf{w}(2), \cdots$，使得代价函数 $\mathscr{E}(\mathbf{w})$ 在算法的每次迭代中都要下降，即

$$\mathscr{E}(\mathbf{w}(n+1)) < \mathscr{E}(\mathbf{w}(n)) \tag{3.10}$$

这里 $\mathbf{w}(n)$ 是权值向量的旧值而 $\mathbf{w}(n+1)$ 是它的更新值。

我们希望算法最终收敛到最优解 $\mathbf{w}^*$。我们说“希望”是因为除非采取特别的预防措施，算法有可能发散（即变得不稳定）。

在这一节我们描述三种以迭代下降思想这种或那种形式为基础的无约束最优化方法（Bertsekas，1995）。

最速下降法

在最速下降法中，对权值向量 $\mathbf{w}$ 的连续调整是在最速下降的方向进行的，即它是与梯度向量$\nabla\mathscr{E}(\mathbf{w})$方向相反的。为了表示的方便，记为

$$\mathbf{g} = \nabla\mathscr{E}(\mathbf{w}) \tag{3.11}$$

因此，最速下降法一般表示为

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \eta\mathbf{g}(n) \tag{3.12}$$

这里 η 是一个正常数，称为步长 (stepsize) 或学习率参数 (learning-rate parameter)，$\mathbf{g}(n)$ 是在 $\mathbf{w}(n)$处的梯度向量值。在从迭代 n 到 $n+1$ 的过程中算法应用修正：

$$\Delta\mathbf{w}(n) = \mathbf{w}(n+1) - \mathbf{w}(n) = -\eta\mathbf{g}(n) \tag{3.13}$$

式(3.13)实际上是导言中描述过的误差修正公式的标准形式。

为了证明最速下降法的公式满足式(3.10)的迭代下降条件，我们用 $\mathbf{w}(n)$附近的一阶泰勒 (Taylor) 级数展开来逼近$\mathscr{E}(\mathbf{w}(n+1))$，即

$$\mathscr{E}(\mathbf{w}(n+1)) \approx \mathscr{E}(\mathbf{w}(n)) + \mathbf{g}^T(n)\Delta\mathbf{w}(n)$$

上式对小的 η 是适用的。在这个近似关系式中代入式(3.13)得到：

$$\mathscr{E}(\mathbf{w}(n+1)) \approx \mathscr{E}(\mathbf{w}(n)) - \eta\mathbf{g}^T(n)\mathbf{g}(n) = \mathscr{E}(\mathbf{w}(n)) - \eta \| \mathbf{g}(n) \|^2$$

上式表明，对正的学习率参数 η，代价函数每次迭代都是下降的。但这里提供的推导是近似的，只有当学习率足够小时才是正确的。

最速下降法收敛到最优解 $\mathbf{w}^*$ 的速度是很慢的。此外，学习率参数 η 对收敛行为有重要影响：

- 当 η 小的时候，算法的瞬时响应是平缓的 (overdamped)，这是由于 $\mathbf{w}(n)$的轨迹是$\mathcal{W}$平面的一个光滑曲线，如图 3.2a 所示。
- 当 η 大的时候，算法的瞬时响应是剧烈的 (underdamped)，这是由于 $\mathbf{w}(n)$的轨迹是锯齿（振荡）形的，如图 3.2b 所示。
- 当 η 超过了某一临界值时，算法是不稳定的（即不收敛）。

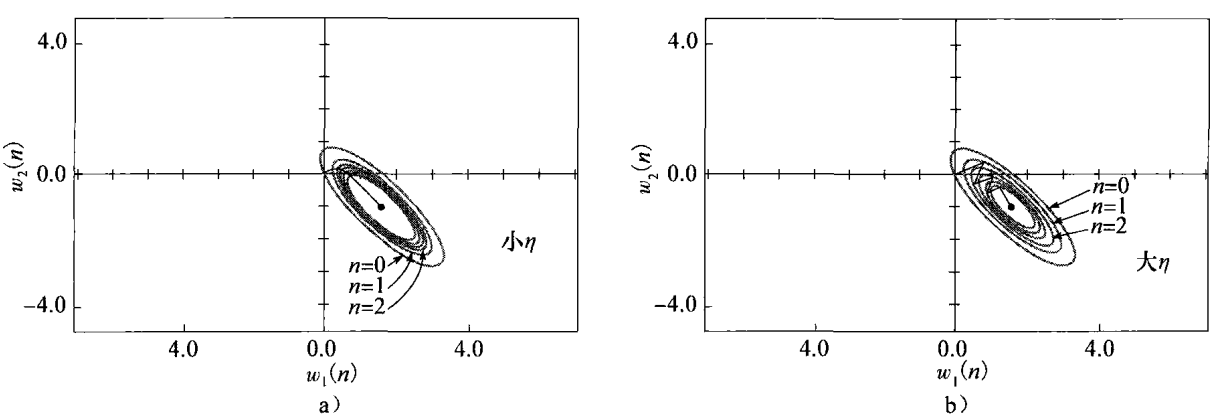


图 3.2　最速下降法关于学习率参数的不同值在二维空间的轨迹。a) 小的 η；b) 大的 η。坐标 $\mathbf{w}_1$ 和 $\mathbf{w}_2$ 是权值向量 $\mathbf{w}$ 的元素；它们都位于$\mathcal{W}$平面中

牛顿法

下面介绍牛顿法 (Newton's method)，这是更复杂的最优化技术。牛顿法的基本思想是最小化代价函数$\mathscr{E}(\mathbf{w})$在当前点 $\mathbf{w}(n)$周围的二次近似值；最小化在算法的每次迭代中都要进行。具体来说，利用代价函数在点 $\mathbf{w}(n)$周围的二次泰勒级数展开式，我们得到：

$$\Delta\mathscr{E}(\mathbf{w}(n)) = \mathscr{E}(\mathbf{w}(n+1)) - \mathscr{E}(\mathbf{w}(n)) \approx \mathbf{g}^T(n)\Delta\mathbf{w}(n) + \frac{1}{2}\Delta\mathbf{w}^T(n)\mathbf{H}(n)\Delta\mathbf{w}(n) \tag{3.14}$$

和以前一样，$\mathbf{g}(n)$是代价函数$\mathscr{E}(\mathbf{w})$在点 $\mathbf{w}(n)$处的 $M\times 1$ 梯度向量。矩阵 $\mathbf{H}(n)$是$\mathscr{E}(\mathbf{w})$在 $\mathbf{w}(n)$的 m 行 m 列 Hessian 矩阵。$\mathscr{E}(\mathbf{w})$的 Hessian 矩阵定义为：

$$\mathbf{H}=\nabla^2\mathscr{E}(\mathbf{w})=\begin{bmatrix}\dfrac{\partial^2\mathscr{E}}{\partial w_1^2} & \dfrac{\partial^2\mathscr{E}}{\partial w_1\partial w_2} & \cdots & \dfrac{\partial^2\mathscr{E}}{\partial w_1\partial w_M}\\ \dfrac{\partial^2\mathscr{E}}{\partial w_2\partial w_1} & \dfrac{\partial^2\mathscr{E}}{\partial w_2^2} & \cdots & \dfrac{\partial^2\mathscr{E}}{\partial w_2\partial w_M}\\ \vdots & \vdots & \vdots & \vdots\\ \dfrac{\partial^2\mathscr{E}}{\partial w_M\partial w_1} & \dfrac{\partial^2\mathscr{E}}{\partial w_M\partial w_2} & \cdots & \dfrac{\partial^2\mathscr{E}}{\partial w_M^2}\end{bmatrix} \tag{3.15}$$

式(3.15)需要代价函数$\mathscr{E}(\mathbf{w})$关于 $\mathbf{w}$ 的元素二阶连续可微。将式(3.14)对 $\Delta\mathbf{w}$ 微分[1]，当

$$\mathbf{g}(n)+\mathbf{H}(n)\Delta\mathbf{w}(n)=\mathbf{0}$$

时，我们最小化了改变量 $\Delta\mathscr{E}(\mathbf{w})$。解上式有关 $\Delta\mathbf{w}(n)$的方程得到：

$$\Delta\mathbf{w}(n)=-\mathbf{H}^{-1}(n)\mathbf{g}(n)$$

也就是

$$\mathbf{w}(n+1)=\mathbf{w}(n)+\Delta\mathbf{w}(n)=\mathbf{w}(n)-\mathbf{H}^{-1}(n)\mathbf{g}(n) \tag{3.16}$$

这里 $\mathbf{H}^{-1}(n)$是$\mathscr{E}(\mathbf{w})$的 Hessian 矩阵的逆。

一般来说，牛顿法收敛得很快，而且不会出现最速下降法有时会出现的锯齿形情况。但是，应用牛顿法时，Hessian 矩阵必须对每个 n 都是正定矩阵。遗憾的是，一般不能保证在算法的每次迭代中 $\mathbf{H}(n)$都是正定矩阵。假如 Hessian 矩阵 $\mathbf{H}(n)$不正定，对牛顿法进行修正就有必要(Powell，1987；Bertsekas，1995)。在很多时候，牛顿法的最主要局限在于其计算复杂度。

Gauss-Newton 法

为了处理牛顿法的计算复杂度而不对其收敛行为做太严重的让步，可以使用 Gauss-Newton 法。为了应用这一方法，我们采用表示为误差平方和的代价函数。令

$$\mathscr{E}(\mathbf{w})=\frac{1}{2}\sum_{i=1}^{n}e^2(i) \tag{3.17}$$

这里尺度因子 1/2 是为了简化下面的分析。此公式中所有的误差项都是以权值向量 $\mathbf{w}$ 为基础计算得来的，这里 $\mathbf{w}$ 在遍及 $1\leqslant i\leqslant n$ 的全部观察区间内固定。

误差信号 $e(i)$是可调权值向量 $\mathbf{w}$ 的函数。给定操作点 $\mathbf{w}(n)$，通过引入下面新的项来线性化 $e(i)$对 $\mathbf{w}$ 的依赖性：

$$e'(i,\mathbf{w})=e(i)+\left[\frac{\partial e(i)}{\partial\mathbf{w}}\right]^T_{\mathbf{w}=\mathbf{w}(n)}\times(\mathbf{w}-\mathbf{w}(n)),\quad i=1,2,\cdots,n$$

用矩阵记号可写成等价的形式：

$$\mathbf{e}'(n,\mathbf{w})=\mathbf{e}(n)+\mathbf{J}(n)(\mathbf{w}-\mathbf{w}(n)) \tag{3.18}$$

其中 $\mathbf{e}(n)$是误差向量

$$\mathbf{e}(n)=[e(1),e(2),\cdots,e(n)]^T$$

$\mathbf{J}(n)$是 $\mathbf{e}(n)$的 $n\times m$ Jacobi 矩阵：

$$\mathbf{J}(n)=\begin{bmatrix}\dfrac{\partial e(1)}{\partial w_1} & \dfrac{\partial e(1)}{\partial w_2} & \cdots & \dfrac{\partial e(1)}{\partial w_M}\\ \dfrac{\partial e(2)}{\partial w_1} & \dfrac{\partial e(2)}{\partial w_2} & \cdots & \dfrac{\partial e(2)}{\partial w_M}\\ \vdots & \vdots & & \vdots\\ \dfrac{\partial e(n)}{\partial w_1} & \dfrac{\partial e(n)}{\partial w_2} & \cdots & \dfrac{\partial e(n)}{\partial w_M}\end{bmatrix}_{\mathbf{w}=\mathbf{w}(n)} \tag{3.19}$$

Jacobi 矩阵 $\mathbf{J}(n)$是 $m\times n$ 梯度矩阵$\nabla \mathbf{e}(n)$的转置，这里

$$\nabla \mathbf{e}(n) = [\nabla e(1), \nabla e(2), \cdots, \nabla e(n)]$$

现在更新后的权值向量 $\mathbf{w}(n+1)$定义为

$$\mathbf{w}(n+1) = \arg\min_{\mathbf{w}} \left\{ \frac{1}{2} \| \mathbf{e}'(n,\mathbf{w}) \|^2 \right. \tag{3.20}$$

利用式(3.18)来评估 $\mathbf{e}'(n,\mathbf{w})$的欧几里得范数的平方，得到

$$\frac{1}{2} \| \mathbf{e}'(n,\mathbf{w}) \|^2 = \frac{1}{2} \| \mathbf{e}(n) \|^2 + \mathbf{e}^T(n)\mathbf{J}(n)(\mathbf{w}-\mathbf{w}(n)) + \frac{1}{2}(\mathbf{w}-\mathbf{w}(n))^T\mathbf{J}^T(n)\mathbf{J}(n)(\mathbf{w}-\mathbf{w}(n))$$

因此，将以上表达式对 $\mathbf{w}$ 求微分并设结果为零，得到

$$\mathbf{J}^T(n)\mathbf{e}(n) + \mathbf{J}^T(n)\mathbf{J}(n)(\mathbf{w}-\mathbf{w}(n)) = \mathbf{0}$$

从这个方程中解出 $\mathbf{w}$，考虑到式(3.20)，可写为：

$$\mathbf{w}(n+1) = \mathbf{w}(n) - (\mathbf{J}^T(n)\mathbf{J}(n))^{-1}\mathbf{J}^T(n)\mathbf{e}(n) \tag{3.21}$$

上式描述了 Gauss-Newton 方法的纯粹形式。

不像牛顿法必须知道代价函数$\mathscr{E}(n)$的 Hessian 矩阵，Gauss-Newton 法只需要知道误差向量 $\mathbf{e}(n)$ 的 Jacobi 矩阵。但是，为了使 Gauss-Newton 迭代可计算，矩阵乘积 $\mathbf{J}^T(n)\mathbf{J}(n)$必须是非奇异的。

关于后一点，我们认识到 $\mathbf{J}^T(n)\mathbf{J}(n)$总是非负定的。为了保证它是非奇异的，Jacobi 矩阵 $\mathbf{J}(n)$ 的行秩必须是 n；也就是说，式(3.19)中 $\mathbf{J}(n)$的 n 行必须是线性无关的。遗憾的是，我们并不能保证这个条件总能满足。为了防止 $\mathbf{J}(n)$的秩亏损，通常的办法是给矩阵$\mathbf{J}^T(n)\mathbf{J}(n)$加一个对角矩阵 $\delta\mathbf{I}$，其中 $\mathbf{I}$ 是单位矩阵。参数 δ 是一个小的正常数，它的选择必须保证

$$\mathbf{J}^T(n)\mathbf{J}(n) + \delta\mathbf{I} \text{ 对所有 } n \text{ 都是正定的}$$

在这个基础上，Gauss-Newton 法以下面的微小修正形式来实现：

$$\mathbf{w}(n+1) = \mathbf{w}(n) - (\mathbf{J}^T(n)\mathbf{J}(n) + \delta\mathbf{I})^{-1}\mathbf{J}^T(n)\mathbf{e}(n) \tag{3.22}$$

当迭代次数 n 不断增大时，增加项 $\delta\mathbf{I}$ 的影响是逐渐减少的。同时注意递归式(3.22)是修正代价函数

$$\mathscr{E}(\mathbf{w}) = \frac{1}{2}\left\{ \sum_{i=1}^{n} e^2(i) + \delta \| \mathbf{w}-\mathbf{w}(n) \|^2 \right\} \tag{3.23}$$

的解，其中 $\mathbf{w}(n)$是权值向量 $\mathbf{w}(i)$的当前值。

在信号处理的文献中，式(3.22)中的增加项 $\delta\mathbf{I}$ 称为对角加载 (diagonal loading)。这一项的增加是为了将代价函数$\mathscr{E}(\mathbf{w})$扩展为式(3.23)的方式，这里我们就有两个项（忽略尺度因子 1/2)：

- 第一项 $\sum_{i=1}^{n} e^2(i)$是误差平方的和，依赖于训练数据。
- 第二项包含欧几里得范数的平方 $\| \mathbf{w}-\mathbf{w}(n) \|^2$，依赖于滤波器结构。实际上，这一项相当于稳定器 (stabilizer)。

尺度因子 δ 通常称为正则参数 (regularization parameter)，代价函数的结果修正相应地称为结构正则化 (structural regularization)。正则化问题将在第 7 章详细讨论。

3.4　维纳滤波器

第 2 章讨论了通常的最小二乘估计器，那里利用极小化的传统方法来从环境的观测模型中找到最小二乘解。为了和本章采用的术语相一致，我们将之称为最小二乘滤波器 (least-squares filter)。而且，我们将利用 Gauss-Newton 法来重新推导这个滤波器的公式。

我们利用式(3.3)和式(3.4)来定义如下的误差向量：

$$\mathbf{e}(n) = \mathbf{d}(n) - [\mathbf{x}(1), \mathbf{x}(2), \cdots, \mathbf{x}(n)]^T \mathbf{w}(n) = \mathbf{d}(n) - \mathbf{X}(n)\mathbf{w}(n) \tag{3.24}$$

其中 $\mathbf{d}(n)$ 是 $n \times 1$ 的期望响应向量，

$$\mathbf{d}(n) = [d(1), d(2), \cdots, d(n)]^T$$

$\mathbf{X}(n)$ 是 $n \times M$ 的数据矩阵，

$$\mathbf{X}(n) = [\mathbf{x}(n), \mathbf{x}(2), \cdots, \mathbf{x}(n)]^T$$

误差向量 $\mathbf{e}(n)$ 对 $\mathbf{w}(n)$ 取微分得到梯度矩阵：

$$\nabla \mathbf{e}(n) = -\mathbf{X}^T(n)$$

相应地，$\mathbf{e}(n)$ 的 Jacobi 矩阵是

$$\mathbf{J}(n) = -\mathbf{X}(n) \tag{3.25}$$

因为误差式(3.18)对权值向量 $\mathbf{w}(n)$ 已经是线性的，如下所示的 Gauss-Newton 法在一次迭代后收敛。将式(3.24)和式(3.25)代入式(3.21)得到

$$\begin{aligned}\mathbf{w}(n+1) &= \mathbf{w}(n) + (\mathbf{X}^T(n)\mathbf{X}(n))^{-1}\mathbf{X}^T(n)(\mathbf{d}(n) - \mathbf{X}(n)\mathbf{w}(n)) \\ &= (\mathbf{X}^T(n)\mathbf{X}(n))^{-1}\mathbf{X}^T(n)\mathbf{d}(n)\end{aligned} \tag{3.26}$$

项 $(\mathbf{X}^T(n)\mathbf{X}(n))^{-1}\mathbf{X}^T(n)$ 被看作是数据矩阵 $\mathbf{X}(n)$ 的伪逆 (pseudoinverse)，即[2]

$$\mathbf{X}^+(n) = (\mathbf{X}^T(n)\mathbf{X}(n))^{-1}\mathbf{X}^T(n) \tag{3.27}$$

因此，可以把式(3.26)改写为紧凑的形式：

$$\mathbf{w}(n+1) = \mathbf{X}^+(n)\mathbf{d}(n) \tag{3.28}$$

这个公式表示了下面所陈述的一个简便途径：

权值向量 $\mathbf{w}(n+1)$ 求解定义在持续时间 n 的一个观察区间上的线性最小二乘问题，是如下两项的乘积：伪逆 $\mathbf{X}^+(n)$ 和期望的响应向量 $\mathbf{d}(n)$。

Wiener 滤波器：遍历环境下线性最小二乘滤波器的极限形式

令 $\mathbf{w}_o$ 表示线性最小二乘滤波器关于观测数 n 的极限形式，允许 n 趋于无穷。可以利用式(3.26)得到：

$$\begin{aligned}\mathbf{w}_o &= \lim_{n\to\infty}\mathbf{w}(n+1) = \lim_{n\to\infty}(\mathbf{X}^T(n)\mathbf{X}(n))^{-1}X^T(n)d(n) \\ &= \lim_{n\to\infty}\left(\frac{1}{n}\mathbf{X}^T(n)\mathbf{X}(n)\right)^{-1} \times \lim_{n\to\infty}\frac{1}{n}\mathbf{X}^T(n)\mathbf{d}(n)\end{aligned} \tag{3.29}$$

现在假设输入向量 $\mathbf{x}(i)$ 和相应的期望响应 $d(i)$ 来自于联合遍历 (ergodic) 平稳环境。我们可以用时间均值来代替总体均值。由定义，输入向量 $\mathbf{x}(i)$ 的相关矩阵 (correlation matrix) 的总体平均形式是

$$\mathbf{R}_{xx} = \mathbb{E}[\mathbf{x}(i)\mathbf{x}^T(i)] \tag{3.30}$$

并且，相应地，输入向量 $\mathbf{x}(i)$ 和期望响应 $d(i)$ 之间的互相关向量 (cross-correlation vector) 的总体平均形式是

$$\mathbf{r}_{dx}\ \mathbb{E}[\mathbf{x}(i)d(i)] \tag{3.31}$$

其中 $\mathbb{E}$ 表示期望算子。从而，在遍历假设下，有

$$\mathbf{R}_{xx} = \lim_{n\to\infty}\frac{1}{n}\mathbf{X}(n)\mathbf{X}^T(n)$$

和

$$\mathbf{r}_{dx} = \lim_{n\to\infty}\mathbf{X}^T(n)\mathbf{d}(n)$$

相应地，可以把式(3.29)改写为用总体平均相关参数来表示：

$$\mathbf{w}_o = \mathbf{R}_{xx}^{-1}\mathbf{r}_{dx} \tag{3.32}$$

这里 $\mathbf{R}_{xx}^{-1}$ 是相关矩阵 $\mathbf{R}_{xx}$ 的逆。式(3.32)是由式(2.32)定义的最小二乘解的总体平均版本。

权值向量 $\mathbf{w}_o$ 称为线性最优滤波问题的维纳解（Widrow and Stearns，1985；Haykin，2002）。因此，我们可以做以下的陈述：

对一个遍历过程，当观察样本数趋于无穷时，线性最小二乘滤波器渐进趋于维纳滤波器。

设计维纳滤波器需要二阶统计量的知识：输入向量 $\mathbf{x}(n)$ 的相关矩阵 $\mathbf{R}_{xx}$ 和 $\mathbf{x}(n)$ 与期望响应 $d(n)$ 的互相关向量 $\mathbf{r}_{xd}$。但是，在实际遇到的很多环境下，这些信息都是未知的。我们可以利用线性自适应滤波器（linear adaptive filter）来处理未知的环境，自适应在这里的意思是滤波器能够调整自己的自由参数来响应环境的统计变化。在连续时间基础上做这类调整的一个流行的算法是最小均方算法，下面来讨论这一算法。

3.5　最小均方算法

最小均方（least mean square，LMS）算法的建立是极小化代价函数的瞬时值，代价函数为

$$\mathscr{E}(\hat{\mathbf{w}}) = \frac{1}{2}e^2(n) \tag{3.33}$$

这里 $e(n)$ 是 n 时刻测得的误差信号。把 $\mathscr{E}(\hat{\mathbf{w}})$ 对权值向量 $\mathbf{w}$ 求微分得到

$$\frac{\partial \mathscr{E}(\hat{\mathbf{w}})}{\partial \hat{\mathbf{w}}} = e(n)\frac{\partial e(n)}{\partial \mathbf{w}} \tag{3.34}$$

如同在最小二乘滤波器上一样，LMS 算法运行在一个线性神经元上，可以把误差信号表示为：

$$e(n) = d(n) - \mathbf{x}^T(n)\hat{\mathbf{w}}(n) \tag{3.35}$$

因此

$$\frac{\partial e(n)}{\partial \hat{\mathbf{w}}(n)} = -\mathbf{x}(n)$$

和

$$\frac{\partial \mathscr{E}(\hat{\mathbf{w}})}{\partial \hat{\mathbf{w}}(n)} = -\mathbf{x}(n)e(n)$$

把后者作为梯度向量的一种瞬间估计（instantaneous estimate），可以记

$$\hat{\mathbf{g}}(n) = -\mathbf{x}(n)e(n) \tag{3.36}$$

最后，利用式(3.36)作为式(3.12)中的最速下降法的梯度向量，可以写出 LMS 算法公式：

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \eta\mathbf{x}(n)e(n) \tag{3.37}$$

这里值得注意的是学习率参数 η 的倒数可以用于度量 LMS 算法的记忆（memory）：给 η 赋的值越小，LMS 算法将记忆的过去数据就越多。因此，η 值小的话，LMS 算法执行得更精确，但算法的收敛速度慢。

在式(3.37)的推导中，我们用 $\hat{\mathbf{w}}(n)$ 代替 $\mathbf{w}(n)$ 来强调这样一个事实：利用最速下降法可以得到一个权值向量，而 LMS 算法产生该权值向量的一个瞬时估计值。所以，使用 LMS 算法时我们牺牲掉最速下降法的一个明显特征。在最速下降法中，对一个给定的 η，权值向量 $\mathbf{w}(n)$ 在权值空间中遵循一个明确定义的轨迹。对比之下，在 LMS 算法中，权值向量 $\hat{\mathbf{w}}(n)$ 则跟踪一个随机的轨迹。由于这个原因，LMS 算法有时也被称为“随机梯度算法”。当 LMS 算法的迭代次数趋于无限时，$\mathbf{w}(n)$ 在维纳解 $\mathbf{w}_o$ 周围随机行走（布朗运动）。一个重要的事实是，与最速下降法不同，LMS 算法不需要知道环境的统计特性。从实际角度来看，LMS 的这一特征是非常重要的。

表 3.1　LMS 算法小结

训练样本：输入信号向量 $=\mathbf{x}(n)$ 　　期望响应 $=d(n)$
用户选择参数：η
初始化：设 $\hat{\mathbf{w}}(0)=\mathbf{0}$
计算：当 $n=1, 2, \cdots$，计算 　　$e(n)=d(n)-\hat{\mathbf{w}}^T(n)\mathbf{x}(n)$ 　　$\hat{\mathbf{w}}(n+1)=\hat{\mathbf{w}}(n)+\eta\mathbf{x}(n)e(n)$

表 3.1 基于式(3.35)和式(3.37)对 LMS

算法做了小结，它清楚表明这种算法的简单性。如该表所示，对于算法的初始化，可以简单地设算法中的权值向量初始值为$\hat{\mathbf{w}}(0)=\mathbf{0}$。

LMS 算法的信号流图表示

结合式(3.35)和式(3.37)，可以把 LMS 算法中的权值向量演变过程表示如下：

$$\begin{aligned}\hat{\mathbf{w}}(n+1) &= \hat{\mathbf{w}}(n)+\eta\mathbf{x}(n)[d(n)-\mathbf{x}^T(n)\,\hat{\mathbf{w}}(n)]\\ &= [\mathbf{I}-\eta\mathbf{x}(n)\mathbf{x}^T(n)]\,\hat{\mathbf{w}}(n)+\eta\mathbf{x}(n)d(n)\end{aligned} \tag{3.38}$$

这里 I 是单位矩阵。通过运用 LMS 算法，我们认识到

$$\hat{\mathbf{w}}(n) = z^{-1}[\hat{\mathbf{w}}(n+1)] \tag{3.39}$$

这里z^{-1}是单位时间延迟算子（unit-time delay operator），意味着存储。利用式(3.38)和式(3.39)，我们就可以用图 3.3 描绘的信号流图来表示 LMS 算法。这个信号流图揭示 LMS 算法是随机反馈系统的一个实例。反馈的出现对 LMS 算法的收敛行为有重要影响。

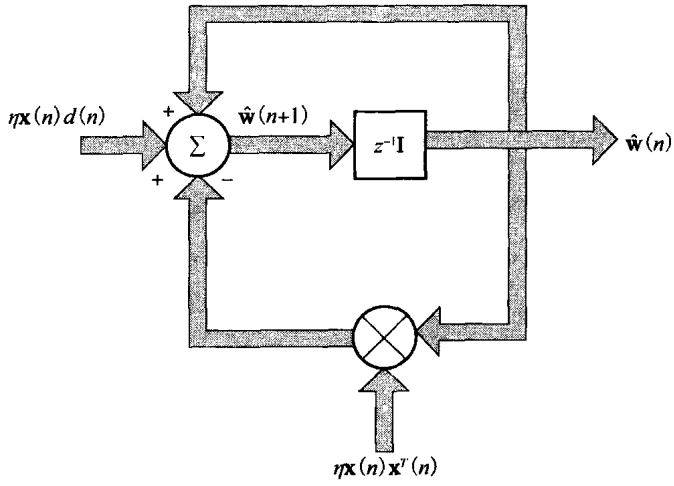


图 3.3　LMS 算法的信号流图表示

3.6　用马尔可夫模型来描画 LMS 算法和维纳滤波器的偏差

为了给 LMS 算法提供一种统计分析，我们发现利用下式定义的权值误差向量（weight-error vector）更加方便：

$$\boldsymbol{\varepsilon}(n) = \mathbf{w}_o - \hat{\mathbf{w}}(n) \tag{3.40}$$

其中$\mathbf{w}_o$是由式(3.32)定义的最优维纳解，$\hat{\mathbf{w}}(n)$是相应的由 LMS 算法计算的权值向量的估计。因此，利用术语$\boldsymbol{\varepsilon}(n)$，假设其为一个状态（state），可以将式(3.38)重写为紧凑形式：

$$\boldsymbol{\varepsilon}(n+1) = \mathbf{A}(n)\boldsymbol{\varepsilon}(n)+\mathbf{f}(n) \tag{3.41}$$

这里，我们有

$$\mathbf{A}(n) = \mathbf{I}-\eta\mathbf{x}(n)\mathbf{x}^T(n) \tag{3.42}$$

其中 I 是单位矩阵。式(3.41)右边附加的噪声项由下式定义：

$$\mathbf{f}(n) = -\eta\mathbf{x}(n)e_o(n) \tag{3.43}$$

其中

$$e_o(n) = d(n)-\mathbf{w}_o^T\mathbf{x}(n) \tag{3.44}$$

是由维纳滤波产生的估计误差。

式(3.41)表示 LMS 算法的马尔可夫模型（Markov model），这一模型的特征如下所示：

- 模型的更新状态（updated state），由向量$\boldsymbol{\varepsilon}(n+1)$定义，依赖于老的状态$\boldsymbol{\varepsilon}(n)$，其自

依赖性由转移矩阵（transition matrix）$\mathbf{A}(n)$定义。

- 在时间 n 上状态的演化被内部所产生的噪声 $\mathbf{f}(n)$所扰动，这一噪声扮演着“驱动力”的角色。

图 3.4 给出了表示这一模型的向量值信号流图。标志为 $z^{-1}\mathbf{I}$ 的分支表示模型的记忆，z^{-1} 作为单位时间延迟算子，由下式定义：

$$z^{-1}[\boldsymbol{\varepsilon}(n+1)] = \boldsymbol{\varepsilon}(n) \tag{3.45}$$

和图 3.3 相比，这个图中用紧凑形式重点强调了 LMS 算法中的反馈过程。

图 3.4 的信号流图以及相应的方程提供了在小学习率参数 η 的假设下 LMS 算法收敛性分析的框架。然而，在进行这一分析之前，我们简要地介绍实现这一目标所需的两个基础知识：在 3.7 节中介绍的朗之万方程，以及随后的 3.8 节中的 Kushner 直接平均法。有了这两个基础知识，我们将在 3.9 节中继续学习 LMS 算法的收敛分析。

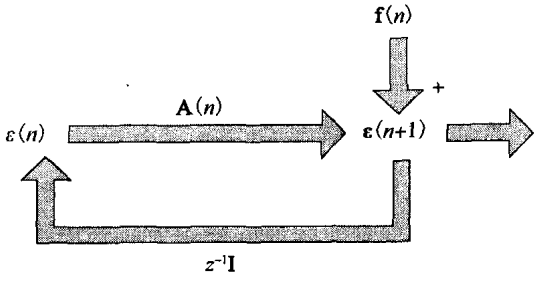


图 3.4　表示由式(3.41)所描述的马尔可夫模型的信号流图

3.7　朗之万方程：布朗运动的特点

对 3.5 节结束部分的评论用更精确的术语重新叙述，考虑稳定性和收敛性，我们可以说 LMS 算法（对足够小的 η）从未达到完美的稳定或者收敛条件。而且，在大量的迭代时间步 n 之后，算法到达“伪平衡”条件，这从定性上讲，可由算法围绕着维纳解执行布朗运动来描述。这一类统计行为可通过非平衡热力学[3] 的朗之万方程（Langevin equation）来很好地解释。因而，我们岔开一下简要地介绍这一重要方程。

令 $v(t)$定义质量为 m 的宏观粒子陷入粘滞铃的速度。假设粒子足够小以使其由热起伏而来的速度被视为重要的。然后，由均分热力学原理（equipartition law of thermodynamics），粒子的平均能量由下式定义：

$$\frac{1}{2}\mathbb{E}[v^2(t)] = \frac{1}{2}k_B T \qquad \text{对所有连续时间 } t \tag{3.46}$$

其中 k_B 为 Boltzmann 常数，T 为绝对温度（absolute temperature）。粘滞铃中分子作用于粒子的总驱动由两部分组成：

1. 依据 Stoke 定律（Stoke's law）的等于$-\alpha v(t)$的阻尼力（damping force），其中 α 是摩擦系数；

2. 涨落力（fluctuating force）$F_f(t)$，其性质是平均指定的。

粒子的运动方程在缺少外部驱动时由下式给定：

$$m\frac{\mathrm{d}v}{\mathrm{d}t} = -\alpha v(t) + F_f(t)$$

两边同时除以 m，我们有

$$\frac{\mathrm{d}v}{\mathrm{d}t} = -\gamma v(t) + \Gamma(t) \tag{3.47}$$

其中

$$\gamma = \frac{\alpha}{m} \tag{3.48}$$

和

$$\Gamma(t) = \frac{F_f(t)}{m} \tag{3.49}$$

项 $\Gamma(t)$是*每单位质量涨落力*（fluctuating force per unit mass）；因为它依赖于组成粒子的极为大量的原子数的位置，所以它是一个统计驱动力，它处于常的不规则运动状态。式(3.47)称为*朗之万方程*（Langevin equation），$\Gamma(t)$称为*朗之万力*（Langevin force）。朗之万方程描述了在粘滞铃中所有时间下粒子的运动（如果其初始条件是指定的），它是描述非平衡热力学的第一个数学公式。

在 3.9 节中，我们将证明 LMS 算法的一个变换版本具有和朗之万方程的离散时间版本相同的数学形式。但在证明之前，需要给出下一个基础知识。

3.8　Kushner 直接平均法

式(3.41)的马尔可夫模型是*非线性随机差分方程*（nonlinear stochastic difference equation）。这一方程是非线性的是因为转移矩阵 $\mathbf{A}(n)$依赖于输入向量 $\mathbf{x}(n)$的外积 $\mathbf{x}(n)\mathbf{x}^T(n)$。因此，权值误差向量 $\boldsymbol{\varepsilon}(n+1)$对于 $\mathbf{x}(n)$的依赖性和叠加原则相冲突，而这一原则是线性的需要。而且，方程是随机的是因为训练样本 $\{\mathbf{x}(n), d(n)\}$ 是从随机环境中取得的。有了这两个事实，我们发现对 LMS 算法作严格的统计分析是很困难的任务。

然而，在一定的条件下，LMS 算法的统计分析能够通过将 Kushner *直接平均法*（Kushner's direct-averaging method）应用于式(3.41)的模型而显著地简化。对这一方法的正规陈述，我们做如下说明（Kushner，1984）：

考虑由马尔可夫模型所描述的随机学习系统：

$$\boldsymbol{\varepsilon}(n+1) = \mathbf{A}(n)\boldsymbol{\varepsilon}(n) + \mathbf{f}(n)$$

其中，对一些输入向量 $\mathbf{x}(n)$，我们有

$$\mathbf{A}(n) = \mathbf{I} - \eta\mathbf{x}(n)\mathbf{x}^T(n)$$

而且附加噪声 $\mathbf{f}(n)$是由学习率参数 η 线性拉伸的。有

- *学习率参数 η 是充分小的。*
- *附加噪声 $\mathbf{f}(n)$ 本质上独立于状态 $\boldsymbol{\varepsilon}(n)$，修正马尔可夫模型的状态演化由下面两个公式来描述：*

$$\boldsymbol{\varepsilon}_0(n+1) = \bar{\mathbf{A}}(n)\boldsymbol{\varepsilon}_0(n) + \mathbf{f}_0(n) \tag{3.50}$$

$$\bar{\mathbf{A}}(n) = \mathbf{I} - \eta E[\mathbf{x}(n)\mathbf{x}^T(n)] \tag{3.51}$$

实际上，对于所有的 n 和原始的马尔可夫模型是一样的。

式(3.51)的确定矩阵 $\bar{\mathbf{A}}(n)$是修正马尔可夫模型的转移矩阵，我们用 $\boldsymbol{\varepsilon}_0(n)$来表示修正马尔可夫模型的状态，用来强调这一模型随时间的演化仅在微小的学习率参数 η 的有限情况下等同于原始马尔可夫模型。

式(3.50)和式(3.51)的证明在习题 3.7 中给出，假设遍历性（即用时间平均来代替总体平均）。由这里的讨论可以充分说明：

1. 如前所述，当学习率参数 η 小的时候，LMS 算法具有*长记忆*（long memory）。因此，更新状态 $\boldsymbol{\varepsilon}_0(n+1)$的演化可以通过时间步一步一步追踪所有的路径直到初始条件 $\boldsymbol{\varepsilon}(0)$。

2. 当 η 小的时候，可以在 $\boldsymbol{\varepsilon}_0(n+1)$的展开式序列中忽略二阶和高阶项。

3. 最后，式(3.50)和式(3.51)中的陈述可以通过调用遍历性来得到，此时总体平均为时间平均所替代。

3.9　小学习率参数下统计 LMS 学习理论

现在我们已经有了 Kushner 直接平均法，该到建立 LMS 算法的统计分析原则的阶段了。我们做三个合理的假设：

假设 Ⅰ：学习率参数 η 是小的

通过这一假设，我们证明可以应用 Kushner 直接平均法—因此采用式(3.50)和式(3.51)的修正马尔可夫模型作为 LMS 算法的统计分析的基础。

从实际角度，η 的小的选择也是有意义的。特别是，当 η 小时，LMS 算法对于外部扰动是鲁棒的；鲁棒性问题将在 3.12 节讨论。

假设 Ⅱ：维纳滤波器产生的估计误差 $e_o(n)$ 是白噪

如果期望响应的产生是由如下的线性回归模型（linear regression model）所描述的，这个假设就会满足：

$$d(n) = \mathbf{w}_o^T \mathbf{x}(n) + e_o(n) \tag{3.52}$$

式(3.52)是式(3.44)的简单的重写，这实际上说明了维纳滤波的权值向量和描述感兴趣随机环境的回归模型的权值向量是匹配的。

假设 Ⅲ：输入向量 $\mathbf{x}(n)$ 和期望响应 $d(n)$ 是联合高斯分布

由物理现象产生的随机过程频繁地出现使得高斯模型是适当的—因此第三个假设得到了验证。

不需要对 LMS 算法的统计分析作更多的假设（Haykin，2002，2006）。下面我们讲述这一分析的精简版本。

LMS 算法的固有模式

令 $\mathbf{R}_{xx}$ 定义输入向量 $\mathbf{x}(n)$ 的总体平均相关矩阵，$\mathbf{x}(n)$ 由稳定过程产生；即

$$\mathbf{R}_{xx} = \mathbb{E}[\mathbf{x}(n)\mathbf{x}^T(n)] \tag{3.53}$$

相应地，可以将式(3.51)的平均转移矩阵表达为修正马尔可夫模型：

$$\begin{aligned}\overline{\mathbf{A}} &= \mathbb{E}[\mathbf{I} - \eta\,\mathbf{x}(n)\mathbf{x}^T(n)] \\ &= [\mathbf{I} - \eta\,\mathbf{R}_{xx}]\end{aligned} \tag{3.54}$$

然后将式(3.50)展开为下面的形式：

$$\boldsymbol{\varepsilon}_0(n+1) = (\mathbf{I} - \eta\,\mathbf{R}_{xx})\boldsymbol{\varepsilon}_0(n) + \mathbf{f}_0(n) \tag{3.55}$$

其中 $\mathbf{f}_o(n)$ 是附加噪声。今后，式(3.55)将作为 LMS 算法统计分析的基础公式。

LMS 算法的固有模式

应用矩阵理论[4] 中对相关矩阵 $\mathbf{R}_{xx}$ 的正交变换，我们有

$$\mathbf{Q}^T\mathbf{R}_{xx}\mathbf{Q} = \boldsymbol{\Lambda} \tag{3.56}$$

其中 $\mathbf{Q}$ 是正交矩阵，其列是 $\mathbf{R}_{xx}$ 的特征向量，且 $\boldsymbol{\Lambda}$ 是对角矩阵且其对角元素是相应的特征值(eigenvalue)。将这一变换引申到式(3.55)的差分方程产生相应的解耦一阶方程系统（system of decoupled first-order equations）（Haykin，2002，2006）：

$$v_k(n+1) = (1 - \eta\lambda_k)v_k(n) + \phi_k(n), k = 1,2,\cdots,M \tag{3.57}$$

其中 M 是权值向量 $\hat{\mathbf{w}}(n)$ 的维数。此外，$v_k(n)$ 是变换后权值误差向量的第 k 个元素：

$$\mathbf{v}(n) = \mathbf{Q}^T\boldsymbol{\varepsilon}_0(n) \tag{3.58}$$

而且，相应地，$\phi_k(n)$ 是变换后噪声向量的第 k 个元素：

$$\boldsymbol{\phi}(n) = \mathbf{Q}^T\mathbf{f}_0(n) \tag{3.59}$$

更具体来说，$\phi_k(n)$是均值为 0 方差为 $\mu^2 J_{\min}\lambda_k$ 的白噪过程的样本函数，其中 $J_{\min}$为由维纳滤波器产生的最小均方误差。实际上，式(3.57)的第 k 个差分方程的 0 -均值驱动力的方差和相关矩阵 $\mathbf{R}_{xx}$的第 k 个特征值 λ_k 成正比。

定义差分：

$$\Delta v_k(n) = v_k(n+1) - v_k(n) \qquad k = 1,2,\cdots,M \tag{3.60}$$

可以将式(3.57)写为如下形式：

$$\Delta v_k(n) = -\eta\lambda_k v_k(n) + \phi_k(n) \qquad k = 1,2,\cdots,M \tag{3.61}$$

随机方程 (3.61) 可以认为是式(3.47)的朗之万方程的离散时间版本。特别地，我们一项项比较这两个公式，可以给出如表 3.2 所列出的类比关系。受这个表的启发，我们可以给出如下重要陈述：

差分方程 (3.55) 的正交变换的应用结果所得到的 LMS 滤波器的收敛行为，可以通过具有 M 个解耦朗之万方程的系统来描述。其第 k 个分量的特点如下所示：

- *阻尼力由 $\eta\lambda_k v_k(n)$定义；*
- *朗之万力 $\phi_k(n)$由均值为 0 方差为 $\eta^2 J_{\min}\lambda_k$ 的白噪过程描述。*

更重要的是，朗之万力 $\phi_k(n)$对于 LMS 算法的非平衡行为负责，它证明了自身在大量的足够多的迭代次数 n 的时候，算法在最优维纳解的周围进行布朗运动 (Brownian motion)。然而，需要强调的是，在表 3.2 中所总结的发现以及前述的陈述是基于学习率参数 η 为小的前提之下。

表 3.2　朗之万方程（连续时间）和变换后的 LMS 演化（离散时间）之间的比较

朗之万方程 (3.47)	LMS 演化 (3.61)
$\frac{dv(t)}{dt}$加速度	$\Delta v_k(n)$
$\gamma v(t)$阻尼力	$\eta\lambda_k v_k(n)$
$\Gamma(t)$随机驱动力	$\phi_k(n)$

LMS 算法的学习曲线

通过解式(3.57)的变换差分方程，我们得到由 Haykin(2002，2006) 所描述的 LMS 学习曲线，

$$J(n) = J_{\min} + \eta J_{\min}\sum_{k=1}^{M}\frac{\lambda_k}{2-\eta\lambda_k} + \sum_{k=1}^{M}\lambda_k\left(|v_k(0)|^2 - \frac{\eta J_{\min}}{2-\eta\lambda_k}\right)(1-\eta\lambda_k)^{2n} \tag{3.62}$$

其中

$$J(n) = \mathbb{E}[|e(n)|^2]$$

为均方误差，$v_k(0)$是变换向量 $\mathbf{v}(n)$的第 k 个元素。在学习率参数 η 小的假设下，式(3.62)简化为

$$J(n) \approx J_{\min} + \frac{\eta J_{\min}}{2}\sum_{k=1}^{M}\lambda_k + \sum_{k=1}^{M}\lambda_k\left(|v_k(0)|^2 - \frac{\eta J_{\min}}{2}\right)(1-\eta\lambda_k)^{2n} \tag{3.63}$$

本节中小学习率参数理论的实际评估在下面的计算机实验部分讲述。

3.10　计算机实验 Ⅰ：线性预测

本实验的目的是证明 3.9 节中所讲的 LMS 算法的统计学习理论，假定一个小的学习率参数 η。

对这一实验，我们考虑一个一般模型，由下式定义

$$x(n) = ax(n-1) + \varepsilon(n) \tag{3.64}$$

这表示了一阶自回归 (AR) 过程。这个模型是一阶的，a 是模型中唯一的参数。解释误差 $\varepsilon(n)$由均值为 0 方差为 σ_ε^2 的白噪过程产生。模型的参数如下所示：

$$a = 0.99$$
$$\sigma_\varepsilon^2 = 0.02$$
$$\sigma_x^2 = 0.995$$

为了估计模型参数 a，我们利用学习率参数为 $\eta=0.001$ 的 LMS 算法。开始的初始条件为 $\hat{\mathbf{w}}(0)=0$，我们应用式(3.35)的标量版本，其中估计误差为：

$$e(n) = x(n) - \hat{a}(n)x(n-1)$$

这里 $\hat{a}(n)$ 是由 LMS 算法在第 n 时间步所产生的 a 的估计。然后，做 100 次统计独立的 LMS 算法的应用，画出算法的总体平均学习曲线。图 3.5 中 5 000 次迭代所画的实心（随机变化）曲线是总体平均操作的结果。

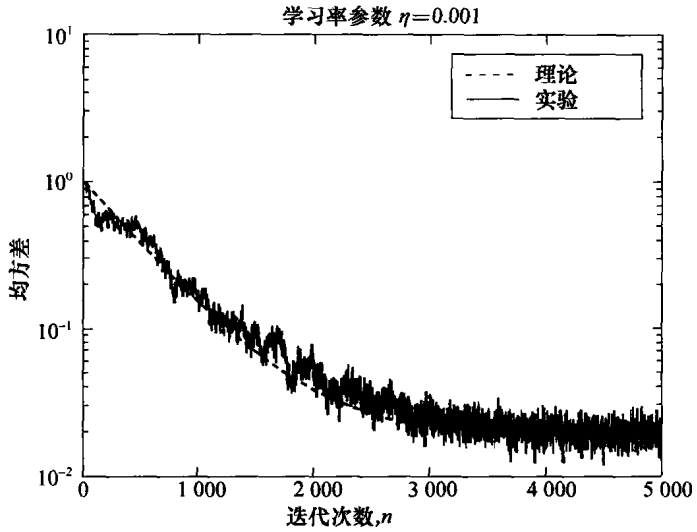


图 3.5　LMS 算法应用于一阶自回归过程的小学习率参数理论的实验验证

在图 3.5 中已经包含了计算总体平均学习曲线的结果，这是通过利用式(3.63)的理论推导公式来实现的，在假设小的 η 的前提下。值得注意的是图 3.5 证明了理论和实际之间完美的一致性。更加地，这一值得注意的一致性可以看成是两个重要的理论原则的确认：

1. 在小学习率参数假设下，Kushner 方法可以用于处理 LMS 学习行为的理论分析。
2. LMS 算法的学习行为可以解释为朗之万方程的一个例子。

3.11　计算机实验 Ⅱ：模式分类

对于 LMS 算法的第二个实验，我们研究将这一算法应用于如图 1.8 所示的双月结构。更具体来说，通过对双月结构的两个设置来评估算法的性能：

(1) $d=1$，相应于线形可分

(2) $d=-4$，相应于线性不可分

实际上，我们重复了第 2.5 节的实验，那时候采用的是最小二乘，这一次采用 LMS 算法。

对于两个 d 值，实验结果分别在图 3.6 和图 3.7 中给出。将这两个图和图 2.2 以及图 2.3 相比较，有以下结果：

(a) 对所有实际目的来说，在所考虑的识别性能范围内，最小二乘和 LMS 算法产生的结果是等同的。

(b) 从收敛性的角度来说，LMS 算法比最小二乘法慢很多。这个差别是因为 LMS 算法事实上是递归的，而最小二乘法是按批量模式运行，包括在一个时间步内进行矩阵求逆。

作为一个有趣的问题，第 5 章中将给出最小二乘法的递归执行方法。由于采用二阶信息，最小二乘法的递归执行仍然比 LMS 算法的收敛行为来得快。

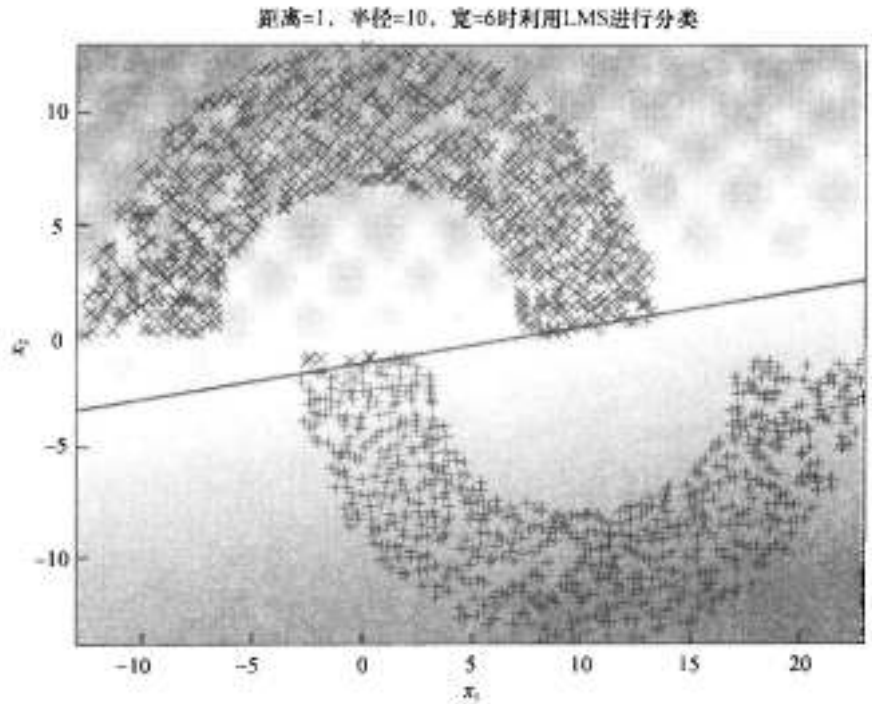


图 3.6　在距离为 1 时的 LMS 分类，基于图 1.8 所示的双月结构

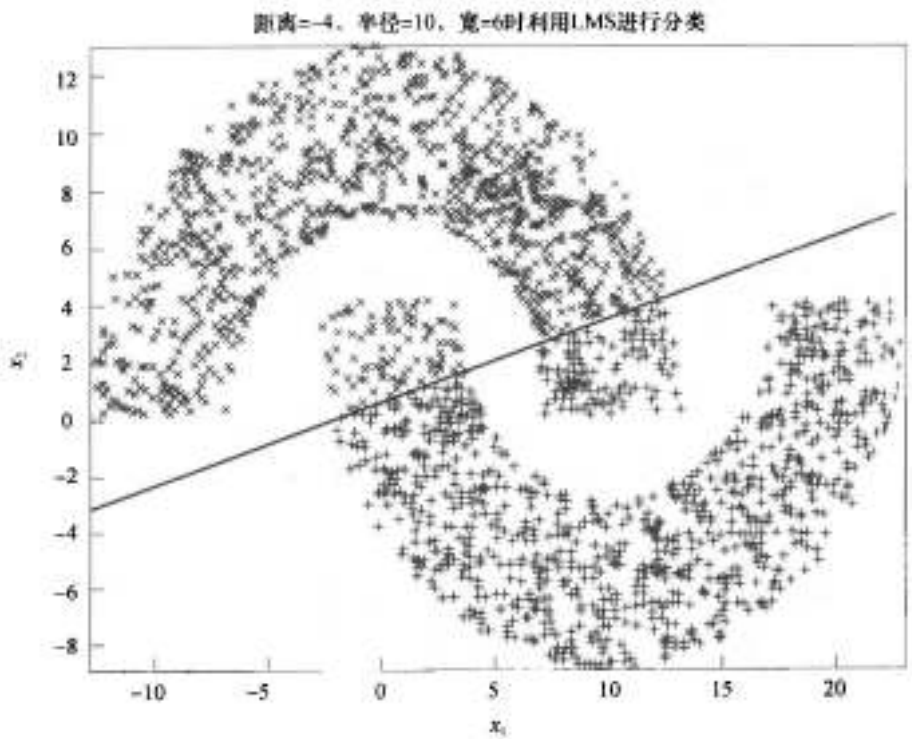


图 3.7　在距离为 −4 时的 LMS 分类，基于图 1.8 所示的双月结构

3.12　LMS 算法的优点和局限

计算简单且有效

LMS 算法的两个优点是计算的简单性和有效性，这两点都可以通过表 3.1 中对算法的总结来得到验证：

- 对于算法的编程仅由 2～3 行组成，这已经是简单得不能再简单了。
- 算法的计算复杂度对于可调整参数个数来说是线性的。

从实际角度来看，这些都是非常重要的优点。

鲁棒性

LMS 算法的另一个重要的优点是它是模型独立的，因而对于扰动来说是鲁棒的。为了解释这里鲁棒性的意义，考虑图 3.8 的情形，那里一个转移算子T将一些扰动从输入端映射到输出端的“一般的”估计误差。具体来说，在输入端，我们有如下项：

- 由下式定义的*初始权值误差向量*

$$\delta\mathbf{w}(0) = \mathbf{w} - \hat{\mathbf{w}}(0) \tag{3.65}$$

其中 $\mathbf{w}$ 是未知的参数向量且$\hat{\mathbf{w}}(0)$是在时间步 $n=0$ 时的“建议”初始估计。在 LMS 算法中，一般，我们设 $\mathbf{w}(0)=\mathbf{0}$，这在某种程度上是对这个算法的最坏的可能初始化条件。

- 回到式(2.3)回归模型中的*解释误差* ε，这里重写这一误差是为了讲述的方便，d 是响应于回归 x 的模型输出：

$$d = \mathbf{w}^T\mathbf{x} + \varepsilon \tag{3.66}$$

自然地，算子T是用于构造估计$\hat{\mathbf{w}}(n)$的方案（例如，LMS 算法）的函数。现在可以引入如下定义：

估计器的能量增益可以定义为算子T的输出的误差能量和输入的总扰动能量之间的比。

为了消除这样的依赖性从而使得估计器是“模型独立”的，我们考虑具有作用于估计器输入的“所有可能扰动序列之上的最大能量增益”（largest possible energy gain over all conceivable disturbance sequences）的情景。这样做的时候，我们定义了转移算子T的 H^∞ 范数。

有了这样简要的背景，现在可以给出转移算子T的 H^∞ 范数：

寻找一个使得T的 H^∞ 范数最小的因果估计器，其中T是将扰动映射到估计误差的转移算子。

和 H^∞ 准则相应的最优估计器是属于极大极小（minimax)种类的。更具体来说，我们可以将 H^∞ 最优估计问题看成是如下意义下的“对策论问题”（game-theoretic problem)：自然，作为“敌对者”，因具有未知扰动，因而能最大化能量增益。另一方面，估计策略的“设计者”具有寻找因果算法的任务以使得误差能量最小化。注意，在图 3.8 中我们介绍 H^∞ 准则思想的时候对于输入的扰动没有做任何假设。因此我们可以说按照 H^∞ 准则设计的估计器是*最坏情况估计器*(worst-case estimator)。

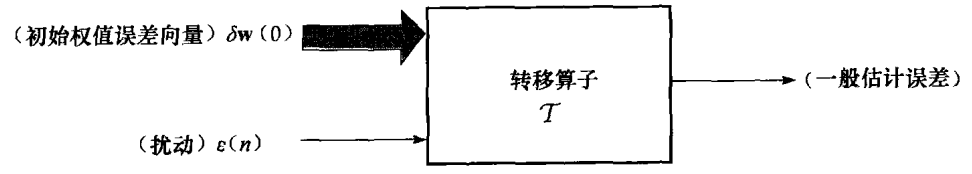


图 3.8　最优 H^∞ 估计问题的构成。转移算子输出端的一般估计误差可以是权值误差向量、解释误差等

以精确的数学术语来说，LMS算法按照 H^{∞} 准则（或极大极小准则）是最优的[5]。在 H^{∞} 意义下最优性的基本原理要处理最坏情况：

如果你不知道你面对的是什么，计划最坏的情况并优化它。

长期以来LMS算法被当作梯度下降法的瞬时逼近。但是，LMS算法的 H^{∞} 最优性为这个广泛应用的算法提供了严格的基础。而且，LMS算法的 H^{∞} 理论说明当学习率参数 η 被赋予一个小的值时算法获得最大的鲁棒特性。

LMS算法的模型独立行为也解释了算法在稳定和不稳定环境下令人满意的工作能力。这里“不稳定”环境是指统计特性随时间变化的环境。在这样一个环境下，最优的维纳解随时间变化，LMS算法有了一个附加任务——跟踪维纳滤波器最小均方误差的变化。

限制LMS性能的因素

LMS算法的主要局限性是收敛速度较慢，并且对输入特征结构的变化是敏感的(Haykin, 1996)。LMS算法一般需要输入空间维数10倍的迭代次数才能达到稳定状态。当输入空间维数较高时缓慢的收敛速度会变得特别严重。

至于对环境条件变化的敏感性，LMS算法的收敛行为对输入向量 x 的相关矩阵 $\mathbf{R}_{xx}$ 的*条件数*（condition number）或*特征值散布*（eigenvalue spread）的变化特别敏感。$\mathbf{R}_{xx}$ 的条件数记为 $\chi(\mathbf{R})$，定义如下：

$$\chi(\mathbf{R}) = \frac{\lambda_{\max}}{\lambda_{\min}} \tag{3.67}$$

这里 $\lambda_{\max}$ 和 $\lambda_{\min}$ 分别是相关矩阵 R_{xx} 的最大和最小特征值。当输入向量 $\mathbf{x}(n)$ 所属的训练样本是*坏条件*(ill conditioned)时，也就是当LMS算法的条件数很大时，LMS算法对条件数 $\chi(\mathbf{R})$ 变化的敏感性变得特别严重。[6]

3.13 学习率退火方案

LMS算法遇到的慢速率收敛可归因于学习率参数在计算过程中保持在某个值 η_0 不变的事实，表示为

$$\eta(n) = \eta_0 \quad \text{对所有 } n \tag{3.68}$$

这只是学习率参数能够假设的最简单的可能形式。相反，在Robbins和Monro有关随机逼近的经典论文中（1951），学习率参数是随时间而改变的。在随机逼近文献中最常用到的学习率参数随时间变化的形式是

$$\eta(n) = \frac{c}{n} \tag{3.69}$$

这里 c 是常数。这样一个选择确实足够保证随机逼近算法的收敛性（Kushner and Clark, 1978）。但是，当常数 c 较大时，对于较小的 n 有可能出现参数放大的危险。

作为式(3.68)和式(3.69)式的替代，可以使用Darken and Moody(1992)定义的*搜索然后收敛方案*(search-then-converge schedule)

$$\eta(n) = \frac{\eta_0}{1 + (n/\tau)} \tag{3.70}$$

这里 η_0 和 τ 是由用户选择的常数。在自适应的早期阶段，即迭代次数 n 相对*搜索时间常数* τ 较小时，学习率参数 $\eta(n)$ 近似等于 η_0，算法运行实际上也是与“标准”LMS算法一样的，如图3.9所示。因此，通过在允许范围内选择一个较大的 η_0，我们希望对滤波器的可调权值能找到一组较好的值并在其中上下浮动。然后，当迭代次数 n 比搜索时间常数 τ 大时，学习率参数

$\eta(n)$近似为 c/n，这里$c=\tau\eta_0$，如图 3.9 所示。算法现在以一个传统的随机逼近算法运行，且权值收敛到它们的最优值。因此，*搜索然后收敛方案*具有把标准 LMS 算法的期望特征和传统随机逼近理论结合起来的潜力。

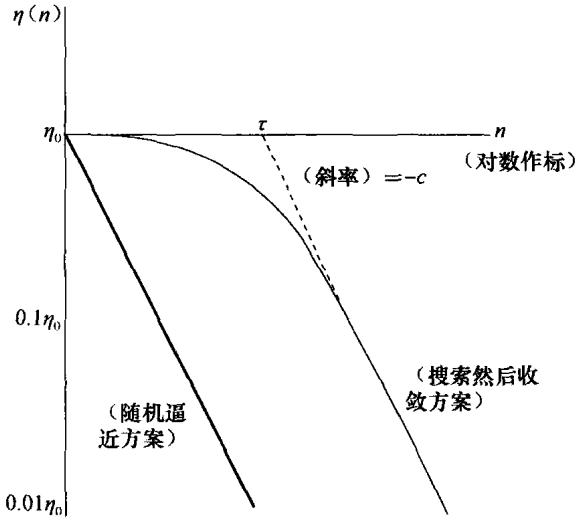


图 3.9　学习率退火方案：横轴表示标准 LMS 算法

3.14　小结和讨论

本章中，我们讨论了著名的最小均方（LMS）算法，它是由 Widrow 和 Hoff 在 1960 年提出的。从这个方法的提出开始，由于以下多个实际原因这一算法经受住了时间的考验：

1. 算法的公式简单而且执行简单，无论是以硬件或者软件形式。
2. 尽管这一算法是简单的，但其性能很高。
3. 从计算的角度来说，算法是*非常高效*的，其复杂度对于可调整参数的个数而言是线性的。
4. 最后也是很重要的一点，算法是模型独立的因而对扰动而言是鲁棒的。

在学习率参数 η 是小的正数的假设下，有了 Kushner *直接平均法*，LMS 算法的收敛行为（通常难以分析）变得数学易处理的。这一方法的理论优点在于当 η 小的时候，描述 LMS 算法收敛行为的非线性“随机”差分方程被原始方程的非线性“确定性”版本所代替。而且，通过灵活运用特征分解，所得到的非线性确定性方程的解被一个解耦一阶差分方程系统所代替。这里要注意的要点是这样推导而来的一阶差分方程从数学上等价于非平衡热力学的朗之万方程的离散时间版本。这一等价性解释了 LMS 算法在大量迭代之后围绕着维纳解进行的布朗运动。在 3.10 节中的计算机实验以及在 Haykin(2006)中的其他计算机实验证实了式(3.63)的有效性，这一公式描述了 LMS 算法的总体平均学习曲线。

值得注意的是当学习率参数 η 小的时候 LMS 算法表现了最佳鲁棒性能。然而，为了这一实际中的重要性能而付出的代价是相应的慢速收敛。在某种程度上，LMS 算法的这一局限可以通过利用学习率退火来缓和，如 3.13 节所讲的那样。

作为最后的评论，本章我们集中讨论了普通的 LMS 算法。无需赘言，这一算法具有多个变形，每个变形都提供了各自的实际优点；对于这些变形的细节，有兴趣的读者可以参考（Haykin，2002）。

注释和参考文献

1. **对一个向量的微分**

设 $f(\mathbf{w})$表示参数向量 $\mathbf{w}$ 的一个实值函数。$f(\mathbf{w})$ 对 $\mathbf{w}$ 的导数定义为如下向量：

$$\frac{\partial f}{\partial \mathbf{w}} = \left[\frac{\partial f}{\partial w_1}, \frac{\partial f}{\partial w_2}, \cdots \frac{\partial f}{\partial w_m}\right]^T$$

这里 m 是向量 $\mathbf{w}$ 的维数。我们对下面的两种情形很感兴趣：

情形 1　函数 $f(\mathbf{w})$定义为内积：

$$f(\mathbf{w}) = \mathbf{x}^T\mathbf{w} = \sum_{i=1}^{m} x_i w_i$$

因此，

$$\frac{\partial f}{\partial w_i} = x_i, i = 1, 2\cdots, m$$

或等价地，以矩阵形式表示：

$$\frac{\partial f}{\partial \mathbf{w}} = \mathbf{x} \tag{3.71}$$

情形 2　函数 $f(\mathbf{w})$定义为二次型：

$$f(\mathbf{w}) = \mathbf{w}^T\mathbf{R}\mathbf{w} = \sum_{i=1}^{m}\sum_{j=1}^{m} w_i r_{ij} w_j$$

这里 r_{ij} 是 $m \times m$ 矩阵 $\mathbf{R}$ 的第 ij 个元素。因此，

$$\frac{\partial f}{\partial w_i} = 2\sum_{j=1}^{m} r_{ij} w_j, i = 1, 2, \cdots, m$$

或等价地，以矩阵形式表示：

$$\frac{\partial f}{\partial \mathbf{w}} = 2\,\mathbf{R}\mathbf{w} \tag{3.72}$$

式(3.71)和式(3.72)为向量的实值函数的微分提供了两个有用的规则。

2. 矩形矩阵的伪逆在 Golub and Van Loan(1996)中进行了讨论；也可参照 Haykin(2002)的第 8 章。
3. 朗之万方程在 Reif(1965)中进行了讨论。关于朗之万方程迷人的历史报告，参照 Cohen(2005)关于噪声的辅导论文。
4. 式(3.56)的正交变换由方阵的特征分解而来。这一主题将在第 8 章中详细讲解。
5. 对于早期的(可能是第一个)关于 H^∞ 控制的诱发性讨论，可以参考 Zames(1981)。

在 H^∞ 意义下关于 LMS 算法最优性的第一个探索是 Hassibi 等(1993)。Hassibi 等 (1999)从估计或者自适应滤波的角度论述 H^∞ 理论。Hassibi 也在 H^∞ 意义下给出了关于 LMS 算法的鲁棒性的精简讨论，参考 Haykin and Widrow(2005)的第 8 章。

从控制的角度来看 H^∞ 理论的书，可以参照 Zhou and Doyle(1998)以及 Green and Limebeer(1995)。

6. LMS 算法的收敛行为关于记为 $\chi(\mathbf{R})$的相关矩阵 $\mathbf{R}_{xx}$ 的条件数变化的敏感性，在 Haykin(2002)的 5.7 节通过实验作了描述。在 Haykin(2002)的第 9 章中，处理最小二乘法的递归执行，也证明了算法的收敛行为本质上独立于条件数 $\chi(\mathbf{R})$。

习题

3.1　(a) 令 $\mathbf{m}(n)$表示 LMS 算法在第 n 次迭代的平均权值向量，即

$$\mathbf{m}(n) = \mathbb{E}[\hat{\mathbf{w}}(n)]$$

利用 3.9 节的小学习率参数理论，证明

$$\mathbf{m}(n) = (\mathbf{I} - \eta\mathbf{R}_{xx})^n[\mathbf{m}(0) - \mathbf{m}(\infty)] + \mathbf{m}(\infty)$$

其中 η 是学习率参数，$\mathbf{R}_{xx}$ 是输入向量 $\mathbf{x}(n)$的相关矩阵，$\mathbf{m}(0)$和 $\mathbf{m}(\infty)$为 $\mathbf{m}(n)$相应的初始和最终值。

(b) 证明对于 LMS 算法在平均意义下收敛，学习率参数 η 必须满足条件：

$$O < \eta < \frac{2}{\lambda_{max}}$$

这里 λ_{max} 是相关矩阵 $\mathbf{R}_{xx}$ 的最大特征值。

3.2 继续习题 3.1，讨论为什么 LMS 算法在平均意义下收敛不是实际中收敛的充分准则。

3.3 考虑将均值为 0 方差为 σ^2 的高斯白噪序列作为 LMS 算法的输入。给出均方意义下算法的收敛条件。

3.4 在 LMS 算法的一个称为漏 LMS 算法(leaky LMS algorithm)的变形中，用于极小化的代价函数定义为

$$\mathscr{E}(n) = \frac{1}{2}|e(n)|^2 + \frac{1}{2}\lambda \|\mathbf{w}(n)\|^2$$

其中 $\mathbf{w}(n)$ 是参数向量，$e(n)$ 是估计误差，λ 是常数。如普通 LMS 算法，我们有

$$e(n) = d(n) - \mathbf{w}^T(n)\mathbf{x}(n)$$

其中 $d(n)$ 是相应于输入向量 $\mathbf{x}(n)$ 的期望响应。

(a) 证明每个时间步对漏 LMS 算法的参数向量的更新由下式定义：

$$\hat{\mathbf{w}}(n+1) = (1-\eta\lambda)\hat{\mathbf{w}}(n) + \eta\mathbf{x}(n)e(n)$$

普通 LMS 算法是一个特例。

(b) 利用 3.9 节的小学习率参数理论，证明

$$\lim_{n\to\infty} \mathbb{E}[\hat{\mathbf{w}}(n)] = (\mathbf{R}_{xx} + \lambda\mathbf{I})^{-1}\mathbf{r}_{dx}$$

其中 $\mathbf{R}_{xx}$ 是 $\mathbf{x}(n)$ 的相关矩阵，$\mathbf{I}$ 是单位矩阵，$\mathbf{r}_{dx}$ 是 $\mathbf{x}(n)$ 和 $d(n)$ 的互相关向量。

3.5 继续习题 3.4，证明漏 LMS 算法可以通过在输入向量 $\mathbf{x}(n)$ 中加入白噪来“模拟”。

(a) 噪声的方差是多少才能使得习题 3.4 中的 (b) 条件得到保持?

(b) 什么时候模拟算法实际上具有和漏 LMS 算法相同的形式? 证明你的答案。

3.6 我们有时在文献中发现学习曲线的均方误差(MSE) 公式被替代为均方偏差(mean-square deviation, MSD) 学习曲线。定义权值误差向量

$$\boldsymbol{\varepsilon}(n) = \mathbf{w} - \hat{\mathbf{w}}(n)$$

这里 $\mathbf{w}$ 是提供期望响应的回归模型的参数向量。这第二个学习曲线是通过对迭代次数 n 计算一块 MSD 而获得的：

$$D(n) = \mathbb{E}[\|\boldsymbol{\varepsilon}(n)\|^2]$$

利用 3.9 节的小学习率参数理论，证明

$$D(\infty) = \lim_{n\to\infty} D(n) = \frac{1}{2}\eta M J_{min}$$

其中 η 是学习率参数，M 是参数向量 $\hat{\mathbf{w}}$ 的大小，J_{min} 是 LMS 算法的最小均方误差。

3.7 在本习题中我们考虑证明直接平均法，假设遍历性。

从式(3.41)开始，它通过转移矩阵 $\mathbf{A}(n)$ 和驱动力 $\mathbf{f}(n)$ 定义权值误差向量 $\boldsymbol{\varepsilon}(n)$。而 $\mathbf{A}(n)$ 和 $\mathbf{f}(n)$ 分别通过输入向量 $\mathbf{x}(n)$ 在式(3.42)和式(3.43)中定义；然后做如下过程：

- 令 $n=0$，评估 $\boldsymbol{\varepsilon}(1)$。
- 令 $n=1$，评估 $\boldsymbol{\varepsilon}(2)$。
- 对于少量的更多次迭代继续这一方式。

有了这些关于 $\boldsymbol{\varepsilon}(n)$ 的迭代值之后，推导转移矩阵 $\mathbf{A}(n)$ 的公式。

下面假设学习率参数 η 足够小以验证仅保持对于 η 是线性的项。因此，假设遍历性，证明

$$\overline{\mathbf{A}}(n) = I - \eta\sum_{i=1}^{n}\mathbf{x}(i)\mathbf{x}^T(i)$$

具有下面的形式：

$$\overline{\mathbf{A}}(n) = \mathbf{I} - \mu\mathbf{R}_{xx}$$

3.8 当学习率参数 η 小的时候，LMS 算法像“小截至频率低通滤波器”(low-pass filter with a small cutoff frequency)一样工作。这样的滤波器产生的输出和输入信号的平均成正比。

利用式(3.41)，通过考虑利用单一参数算法的简单例子讲述 LMS 算法的性质。

3.9 对于小学习率参数从式(3.55)开始，证明在稳定状态条件下，保持 Lyapunov 方程

$$\mathbf{R}\mathbf{P}_0(n)+\mathbf{P}_0(n)\mathbf{R}=\eta\sum_{i=0}^{\infty}J_{\min}^{(i)}\mathbf{R}^{(i)}$$

其中我们有

$$\mathbf{J}_{\min}^{(i)}=\mathbb{E}[e_o(n)e_o(n-i)]$$

且

$$\mathbf{R}^{(i)}=\mathbb{E}[\mathbf{x}(n)\mathbf{x}^T(n-i)]$$

对于 $i=0,1,2,\cdots$ 矩阵 $\mathbf{P}_0$ 通过$\mathbb{E}[\varepsilon_o(n)\varepsilon_o^T(n)]$来定义，$e_o(n)$是由维纳滤波器产生的不可削减的估计误差。

计算机实验

3.10　对于下面的学习率参数值重复 3.10 节关于线性预测的计算机实验：

(1) $\eta=0.002$；

(2) $\eta=0.01$；

(3) $\eta=0.02$。

对于每个 η 值，根据 LMS 算法的小学习率参数理论的适用性对你的发现做出评论。

3.11　将图 1.8 中的双月间的分隔距离设为 $d=0$，重复 3.11 节中模式分类的计算机实验。

和习题 1.6 关于感知器的实验以及习题 2.7 关于最小二乘法的试验比较你的实验结果。

3.12　利用下面的分隔距离，画出将 LMS 算法应用于图 1.8 的双月结构的模式分类学习曲线：

$d=1$

$d=0$

$d=-4$

将这一实验结果与第 1 章利用 Rosenblatt 感知器的实验结果进行比较。

多层感知器

本章组织

在本章中，我们将从多个方面来学习多层感知器，多层感知器代表一类具有一层或多层隐藏层的神经网络。在4.1节介绍了引言素材之后，我们按如下步骤学习：

4.2～4.7节讨论关于反向传播的知识。4.2节中介绍一些预备知识来为反向传播算法的推导做准备。这一节也包含了关于信用分配问题的讨论。4.3节介绍两种学习方法：批量和在线学习。4.4节介绍反向传播算法的推导细节，利用了微积分学的链式规则。推导过程中采用了传统方法。4.5节通过解XOR问题来说明反向传播算法的应用。XOR是一个无法用Rosenblatt感知器来解决的有趣的问题。4.6节给出了一些使得反向传播算法更好实现的启发式方法和实际的指导方针。4.7节给出了一个关于多层感知器的模式分类实验，这一感知器通过反向传播算法来训练。

4.8节和4.9节处理误差曲面。4.8节讨论反向传播学习在计算网络逼近函数的偏导数中的基础规则。4.9节讨论关于误差曲面的Hessian矩阵的计算问题。4.10节讨论两个问题：如何实现最优退火以及如何使得学习率参数自适应。

4.11～4.14节集中讨论用反向传播算法训练的多层感知器性能方面的多个问题。4.11节讨论泛化问题——这是关于学习的一个非常本质的问题。4.12节中讨论通过多层感知器来实现连续函数的逼近问题。在4.13节中将交叉验证作为统计设计工具来讨论。4.14节讨论复杂度正则化问题以及网络修剪技术。

4.15节总结了反向传播学习的优点和局限性。

学习完反向传播学习之后，4.16节从不同角度来将监督学习看作为最优化问题进行讨论。

在4.17节讲述一类重要的神经网络：卷积多层感知器（convolutional multilayer perceptron）。这一网络已经在解困难模式识别问题时得到了成功的应用。

4.18节处理非线性滤波，这里时间扮演着关键角色。这一讨论从短时记忆结构开始，为通用短视映射定理（universal myopic mapping theorem）建立了基础。

4.19节讨论小规模和大规模学习问题。

最后是4.20节的小结和讨论。

4.1 引言

在第1章中，我们学习了Rosenblatt感知器，它本质上是一个单层神经网络。该章证明了这一网络局限于线性可分模式的分类问题。然后，在第3章中，我们学习了自适应滤波，采用了Widrow和Hoff的LMS算法。这一算法也是基于权值可调的单个线性神经元，这也限制了这一算法的计算能力。为了克服感知器和LMS算法的实际局限，我们考虑所熟知的多层感知器这一神经网络结构。

下面的三点揭示了多层感知器的基本特征：

- 网络中每个神经元模型包含一个可微的非线性激活函数。
- 网络中包括一个或多个隐藏在输入和输出神经节点之间的层。
- 网络展示出高度的连接性，其强度是由网络的突触权值决定的。

然而，同样这些特性也导致了现阶段关于网络行为知识的缺乏。首先，由于非线性分布式的存在和网络的高度连接性使得多层感知器的理论分析难于进行。第二，隐藏层的使用使得学

习过程变得更难。这暗示着学习过程必须决定输入模式的哪些特征应该由隐藏层神经元表示出来。学习过程因此变得更困难了，因为不得不在大得多的可能函数空间中搜索，同时必须在输入模式的不同表示中进行选择。

训练多层感知器的一个流行方法是反向传播算法，这包含 LMS 算法作为一个特例。训练分为如下的两个阶段：

1. 前向阶段，网络的突触权值是固定的，输入信号在网络中一层一层传播，直到到达输出端。因此，在这一阶段，输入信号的影响限制在网络中激活隐藏神经元和输出神经元上。

2. 反向阶段，通过比较网络的输出和期望输出产生一个误差信号。得到的误差信号再次通过网络一层一层传播，但是这一次传播是在反向方向进行的。在这第二阶段，对于网络的突触权值进行不断的修正。对于输出层权值的修正计算是直接的，但是对于隐藏层来说则更有挑战性。

“反向传播”这个词的使用出现在 1985 年后，而它的广泛使用是在《Parallel Distributed Processing》(Rumelhart and McClelland，1986) 这本书出版以后。

20 世纪 80 年代中期反向传播算法的提出是神经网络发展史上的一个里程碑，因为它为训练多层感知器提供了一个高效的计算方法，它使多层感知器的学习不再像 Minsky 和 Papert 在其 1969 年所著的书中所暗示的那样悲观。

4.2　一些预备知识

图 4.1 表示一个具有两个隐藏层和一个输出层的多层感知器的结构图。为了构筑多层感知器一般形式的描述平台，这里说的网络是全连接的 (fully connected)。这就是说在任意层上的一个神经元与它之前的层上的所有节点/神经元都连接起来。信号一层接一层地逐步流过，方向是向前的，从左到右。

图 4.2 描绘了多层感知器的一部分。在这个网络中，两种信号都能被识别：

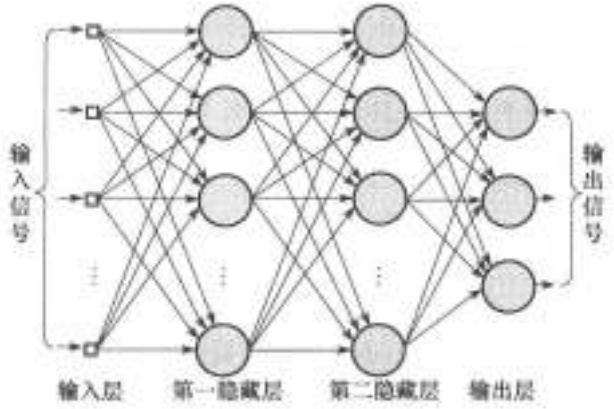


图 4.1　具有两个隐藏层的多层感知器结构图

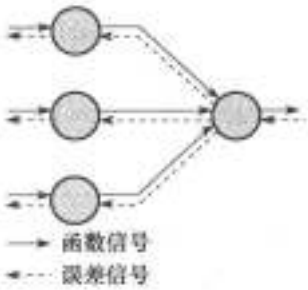


图 4.2　多层感知器中两个基本信号流的方向图示：函数信号的前向传播和误差信号的反向传播

1. 函数信号。函数信号是从网络输入端而来的一个输入信号（刺激），通过网络（一个神经元接一个神经元）向前传播，到达网络输出端即成为一个输出信号。我们把这样一个信号称为“函数信号”有两个原因。首先，在网络输出端时假设它表现为有用的函数。第二，在函数信号通过网络上每一个神经元处，该处信号都被当成输入以及与该神经元有关的权值的一个函数来计算的。函数信号也被认为是输入信号。

2. 误差信号。一个误差信号产生于网络的一个输出神经元，并通过网络（一层接一层）反向传播。我们称之为“误差信号”是因为网络的每一个神经元对它的计算都以这种或那种形

式涉及误差依赖函数。

输出神经元构成网络的输出层。余下的神经元构成网络的隐藏层。因此隐藏层单元并不是网络输出或输入的一部分——因此它们被称为“隐藏”的。第一隐藏层的信号是从由传感单元（源节点）构成的输入层馈给的；而第一隐藏层的输出结果又应用于下一个隐藏层；网络的其余部分依此类推。

多层感知器每一个隐藏层或输出层神经元的作用是进行两种计算：

1. 计算一个神经元的输出处出现的函数信号，它表现为关于输入信号以及与该神经元相关联的突触权值的一个连续非线性函数。

2. 计算梯度向量（即误差曲面对连接于一个神经元输入的权值的梯度）的一个估计，它需要反向通过网络。

隐藏神经元的功能

隐藏神经元扮演着特征检测算子（feature detector）的角色；它们在多层感知器的运转中起着决定性作用。随着学习过程通过多层感知器不断进行，隐藏神经元开始逐步“发现”刻画训练数据的突出特征。它们是通过将输入数据非线性变换到新的称为特征空间的空间而实现的。例如，在模式分类问题中，感兴趣的类在这个新的空间中可能比原始输入数据空间中更容易分隔开。甚至，正是通过监督学习形成的这一特征空间将多层感知器和 Rosenblatt 感知器区别开来。

信用分配问题

当学习如图 4.1 所示的分布式系统的学习算法时，注意信用分配（credit assignment）的概念是有益的。基本上，信用分配问题是分配总体结果的信用或者责任（blame）给每一个由分布式学习系统的隐藏计算单元所产生的内部决策（internal decision），首先要注意的是那些决策将决定总体结果。

在利用误差相关学习（error-correlation learning）的多层感知器中，会发生信用分配问题。这是因为网络中每一个隐藏神经元和每一个输出神经元的操作，对于网络感兴趣的学习任务的总体正确行为而言都是重要的。也就是说，为了解决给定的任务，网络必须通过特定的误差修正学习算法给它的所有神经元分配某种形式的行为。在这一背景下，考虑图 4.1 所示的多层感知器。因为每一个输出神经元对于外部世界来说是可见的，我们可以提供一个期望响应来指导这些神经元的行为。因此，一旦考虑了输出神经元，就可以直接通过误差修正算法来修正每个输出神经元的突触权值。但是，当误差修正学习算法被用来修正隐藏神经元的突触权值时，如何给隐藏神经元的行为分配信用或者责任呢？对这一基本问题的答案需要比输出神经元的情形给出更细节的关注。

在本章后续的部分，我们给出反向传播算法，它是多层感知器训练的基础算法。反向传播算法以一种精致的方式解决了信用分配问题。但是在介绍反向传播算法之前，我们在下一节中讲述监督学习的两种基本方法。

4.3　批量学习和在线学习

考虑具有一个由源节点组成的输入层、一个或多个隐藏层、由一个或者多个神经元组成的输出层的多层感知器，如图 4.1 所示。令

$$\mathcal{T} = \{\mathbf{x}(n), \mathbf{d}(n)\}_{n=1}^{N} \tag{4.1}$$

用于训练网络的训练样本采用有监督方式。令 $y_j(n)$ 记为在输出层第 j 个神经元输出产生的函数信号，这一函数信号是由作用在输入层的刺激 $\mathbf{x}(n)$ 所产生的。相应地，神经元 j 的输出所产生的误差信号定义为：

$$e_j(n) = d_j(n) - y_j(n) \tag{4.2}$$

其中 $d_j(n)$是期望响应向量 $\mathbf{d}(n)$的第 j 个元素。根据在第 3 章学习过的 LMS 算法的术语，神经元 j 的瞬时误差能量（instantaneous error energy）定义为

$$\mathscr{E}_j(n) = \frac{1}{2}e_j^2(n) \tag{4.3}$$

将所有输出层神经元的误差能量相加，得到整个网络的全部瞬时误差能量（total instantaneous error energy）：

$$\mathscr{E}(n) = \sum_{j\in C}\mathscr{E}_j(n) = \frac{1}{2}\sum_{j\in C}e_j^2(n) \tag{4.4}$$

其中集合 C 包括输出层的所有神经元。设训练样本中包含 N 个样例，训练样本上的平均误差能量（error energy averaged over the training sample）或者说经验风险（empirical risk）定义为：

$$\mathscr{E}_{av}(N) = \frac{1}{N}\sum_{n=1}^{N}\mathscr{E}(n) = \frac{1}{2N}\sum_{n=1}^{N}\sum_{j\in C}e_j^2(n) \tag{4.5}$$

自然，瞬时误差能量以及平均误差能量都是多层感知器的所有可调突触权值（即自由参数）的函数。这一函数依赖性没有包含在$\mathscr{E}$(n)和$\mathscr{E}_{av}(N)$的公式中，这仅仅是为了简化表达。

根据多层感知器监督学习的实际执行方式，可以给出两种不同的方法——即批量学习和在线学习，在下面梯度下降的讨论中将对此进行探讨。

批量学习

在监督学习的批量方法中，多层感知器的突触权值的调整在训练样本集合$\mathscr{T}$的所有 N 个样例都出现后进行，这构成了训练的一个回合（epoch）。换句话说，批量学习的代价函数是由平均误差能量$\mathscr{E}_{av}$定义的。多层感知器的突触权值的调整是以回合-回合为基础的（epoch-by-epoch basis）。相应地，学习曲线的一种实现方式是通过描画$\mathscr{E}_{av}$对回合数的图形而得到，对于训练的每一个回合，训练样本集$\mathscr{T}$的样例是随机选取的（randomly shuffled）。学习曲线通过对足够大量的这样实现的总体平均（ensemble averaging）来计算，这里每次实现是在随机选取不同初始条件下完成的。

用梯度下降法来实现训练时，批量学习的优点在于：

- 对梯度向量（即代价函数$\mathscr{E}_{av}$对权值向量 $\mathbf{w}$ 的导数）的精确估计，因此，在简单条件下，保证了这一方法最速下降到局部极小点的收敛性。
- 学习过程的并行性。

然而，从实际观点看，批量学习有着存储需求（storage requirement）。

从统计的角度看，批量学习可以看成是某种形式的统计推断（statistical inference）。因此它很适合于解非线性回归问题。

在线学习

在监督学习的在线方法下，对于多层感知器突触权值的调整是以样例-样例为基础的（example-by-example basis）。用来最小化的代价函数是全体瞬时误差能量$\mathscr{E}(n)$。

考虑由 N 个训练样本构成的一个回合，样本的顺序是 $\{\mathbf{x}(1),\mathbf{d}(1)\}$，$\{\mathbf{x}(2),\mathbf{d}(2)\}$，…，$\{\mathbf{x}(N),\mathbf{d}(N)\}$。回合中第一个样例对 $\{\mathbf{x}(1),\mathbf{d}(1)\}$ 输入给网络时，梯度下降法被用来调整权值。然后回合中第二个样本 $\{\mathbf{x}(2),\mathbf{d}(2)\}$ 输入给网络，这导致对网络权值的进一步调整。这一过程不断持续直到最后一个样例 $\{\mathbf{x}(N),\mathbf{d}(N)\}$。遗憾的是，这样的过程违反了在线学习的并行性。

对于给定的初始条件集合，学习曲线的一种实现是靠以下方式得到的，对训练过程中的回

合数，描画最终值$\mathscr{E}(N)$，这里和前面一样。训练样例是在每个回合后随机选取的。和批量学习一样，在线学习的学习曲线是通过对足够大量的随机选取的初始条件上的总体平均来计算的。自然地，对于给定的网络结构，在线学习下获得的学习曲线和批量学习下获得的学习曲线有着很大的不同。

给定训练样本以随机的方式呈现给网络，在线学习的使用使得在多维权值空间中的搜索事实上是随机的；正是由于这个原因，在线学习方法有时被称为随机方法。这一随机性具有所希望的学习过程不容易陷入局部极值点的效果，这是在线学习好于批量学习的明确意义所在。在线学习的另一个优点在于它比批量学习需要的存储量要少得多。

而且，如果训练数据是冗余的（即训练样本集$\mathscr{T}$包含同一个样例的多个复制），我们发现，和批量学习不同，在线学习能够从冗余性中获益，因为在一次学习中样例只出现一个。

在线学习的另一个有用的性质是它能够追踪训练数据的小的改变，尤其是产生数据的环境是不稳定的情况下。

总之，尽管在线学习有一些缺点，但它在解决模式分类问题时仍然是流行的方法，原因有以下两点：

- 在线学习容易执行。
- 对于大规模和困难模式分类问题它提供有效解。

正是由于这两个原因，本章中大量的内容都是关于在线学习的。

4.4　反向传播算法

多层感知器监督训练在线学习的流行由于反向传播算法的提出而得到了加强。为了描述这一算法，考虑图 4.3，它描绘神经元 j 被它左边的一层神经元产生的一组函数信号所馈给。因此，在神经元 j 的激活函数输入处产生的诱导局部域 $v_j(n)$ 是：

$$v_j(n) = \sum_{i=0}^{m} w_{ji}(n) y_i(n) \tag{4.6}$$

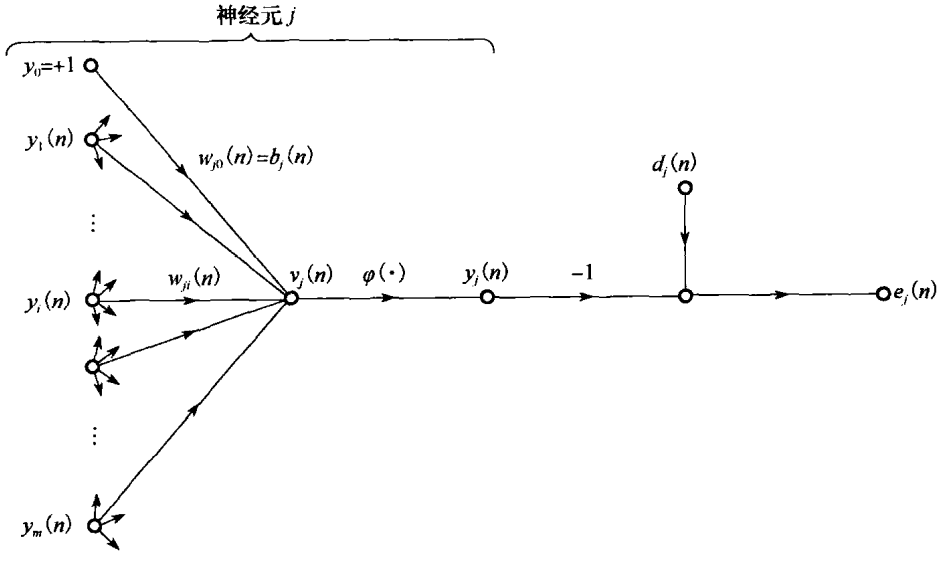


图 4.3　显现输出神经元 j 细节的信号流图

这里 m 是作用于神经元 j 的所有输入（不包括偏置）个数。突触权值 w_{j0}（对应于固定输入 $y_0=+1$）等于神经元 j 的偏置 b_j。所以迭代 n 时出现在神经元 j 输出处的函数信号 $y_j(n)$ 是

$$y_j(n) = \varphi_j(v_j(n)) \tag{4.7}$$

反向传播算法以与第 3 章学习过的 LMS 算法类似的方式对突触权值 $w_{ji}(n)$ 应用一个修正值 $\Delta w_{ji}(n)$，它正比于偏导数 $\partial \mathscr{E}(n)/\partial w_{ji}(n)$。根据微分的链式规则，可以将这个梯度表示为：

$$\frac{\partial \mathscr{E}(n)}{\partial w_{ji}(n)} = \frac{\partial \mathscr{E}(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial w_{ji}(n)} \tag{4.8}$$

偏导数 $\partial \mathscr{E}(n)/\partial w_{ji}(n)$ 代表一个敏感因子，决定突触权值 w_{ji} 在权值空间的搜索方向。

在式(4.4)两边对 $e_j(n)$ 取微分，得到：

$$\frac{\partial \mathscr{E}(n)}{\partial e_j(n)} = e_j(n) \tag{4.9}$$

在式(4.2)两边对 $y_j(n)$ 取微分，得到：

$$\frac{\partial e_j(n)}{\partial y_j(n)} = -1 \tag{4.10}$$

接着，在式(4.7)两边对 $v_j(n)$ 取微分，得到：

$$\frac{\partial y_j(n)}{\partial v_j(n)} = \varphi_j'(v_j(n)) \tag{4.11}$$

这里，导数符号（等式右边）的使用强调了对于自变量的微分。最后，在式(4.6)两边对 $w_{ji}(n)$ 取微分，得到：

$$\frac{\partial v_j(n)}{\partial w_{ji}(n)} = y_i(n) \tag{4.12}$$

将式(4.9)至式(4.12)代入式(4.8)，得到：

$$\frac{\partial \mathscr{E}(n)}{\partial w_{ji}(n)} = -e_j(n)\varphi_j'(v_j(n))y_i(n) \tag{4.13}$$

应用于 $w_{ji}(n)$ 的修正 $\Delta w_{ji}(n)$ 由 delta 法则定义为

$$\Delta w_{ji}(n) = -\eta \frac{\partial \mathscr{E}(n)}{\partial w_{ji}(n)} \tag{4.14}$$

其中 η 是反向传播算法的学习率参数。式(4.14)中负号的使用意味着在权空间中梯度下降（即寻找一个使得 $\mathscr{E}(n)$ 值下降的权值改变的方向）。于是将式(4.13)代入式(4.14)中得到：

$$\Delta w_{ji}(n) = \eta \delta_j(n) y_i(n) \tag{4.15}$$

这里局域梯度 $\delta_j(n)$ 定义为：

$$\delta_j(n) = \frac{\partial \mathscr{E}(n)}{\partial v_j(n)} = \frac{\partial \mathscr{E}(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} = e_j(n)\varphi_j'(v_j(n)) \tag{4.16}$$

局域梯度指明突触权值所需要的变化。根据式(4.16)，输出神经元 j 的局域梯度 $\delta_j(n)$ 等于该神经元相应误差信号 $e_j(n)$ 和相应激活函数的导数 $\varphi_j'(v_j(n))$ 的乘积。

从式(4.15)和式(4.16)我们注意到，权值调整 $\Delta w_{ji}(n)$ 计算所涉及的一个关键因子是神经元 j 输出端的误差信号 $e_j(n)$。在这种情况下，我们要根据神经元 j 的不同位置来区别两种不同的情况。第一种情况，神经元 j 是输出节点。这种情况的处理很简单，因为网络的每一个输出节点都提供自己期望的反应信号，使得计算误差信号变得非常简单。在第二种情况下，神经元 j 是隐藏层节点。虽然隐藏层神经元不能直接访问，但是它们分担对网络输出的误差的责任。然而，问题是要知道对隐藏层神经元这种共担的责任如何进行惩罚或奖赏。这就是在第 4.2 节中讨论过的*信用分配问题*。

情况 1　神经元 j 是输出节点

当神经元 j 位于网络的输出层时，给它提供自己的一个期望响应。我们可以用式(4.2)来计算这个神经元的误差信号 $e_j(n)$；参看图 4.3。当 $e_j(n)$ 确定以后，用式(4.16)来计算局域梯

度 $\delta_j(n)$是很直接的。

情况 2　神经元 j 是隐藏层节点

当神经元 j 位于网络的隐藏层时，就没有对该输入神经元的指定期望响应。因此，隐藏层的误差信号要根据所有与隐藏层神经元直接相连的神经元的误差信号来向后递归决定。这就是为什么反向传播算法的提出变得很复杂的原因。考虑在图 4.4 中所描绘的情况，它描绘的神经元 j 就是一个网络隐藏层节点。根据式(4.16)，可把隐藏层神经元的局域梯度 $\delta_j(n)$重新定义为：

$$\delta_j(n) = -\frac{\partial \mathscr{E}(n)}{\partial y_j(n)}\frac{\partial y_j(n)}{\partial v_j(n)} = -\frac{\partial \mathscr{E}(n)}{\partial y_j(n)}\varphi_j'(v_j(n)), \text{神经元 } j \text{ 是隐藏的} \tag{4.17}$$

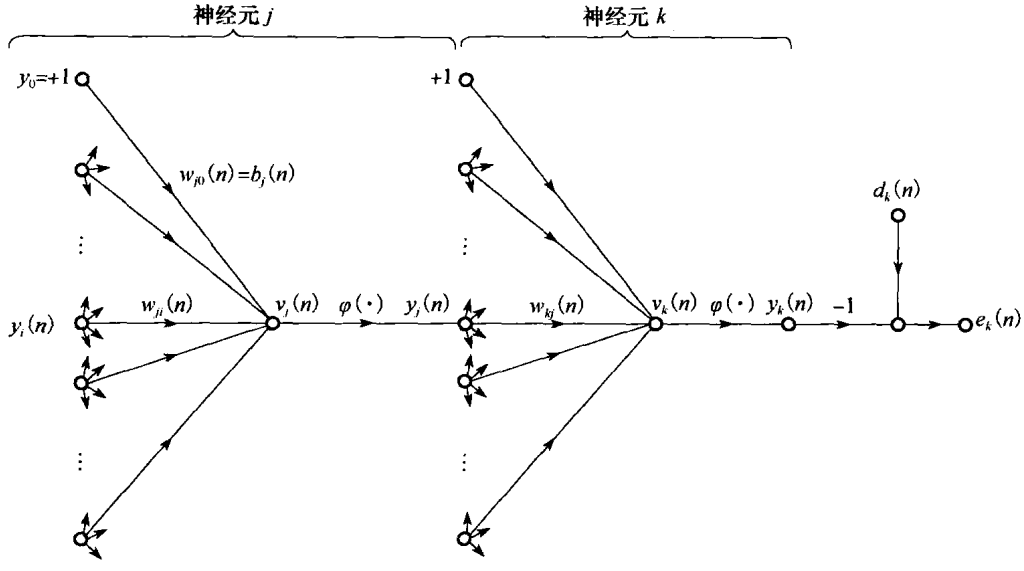


图 4.4　显现输出神经元 k 连接到隐藏神经元 j 的信号流图

公式的第二行用到了式(4.11)。要计算偏导 $\partial \mathscr{E}(n)/\partial y_j(n)$我们进行如下处理。从图 4.4可以看到：

$$\mathscr{E}(n) = \frac{1}{2}\sum_{k\in C} e_k^2(n), \text{神经元 } k \text{ 是输出节点} \tag{4.18}$$

这就是对式(4.4)用下标 k 替代下标 j。这么写是为了避免与在情况 2 使用下标 j 表示一个隐藏神经元相混淆。在式(4.18)两边对函数信号 $y_j(n)$求偏导，得到：

$$\frac{\partial \mathscr{E}(n)}{\partial y_j(n)} = \sum_k e_k \frac{\partial e_k(n)}{\partial y_j(n)} \tag{4.19}$$

接着对偏导数 $\partial E_k(n)/\partial y_j(n)$使用链式规则，重写式 (4.19) 为等价形式：

$$\frac{\partial \mathscr{E}(n)}{\partial y_j(n)} = \sum_k e_k(n)\frac{\partial e_k(n)}{\partial v_k(n)}\frac{\partial v_k(n)}{\partial y_j(n)} \tag{4.20}$$

然而，从图 4.4 我们注意到：

$$e_k(n) = d_k(n) - y_k(n) = d_k(n) - \varphi_k(v_k(n)), \quad \text{神经元 } k \text{ 为输出节点} \tag{4.21}$$

因此

$$\frac{\partial e_k(n)}{\partial v_k(n)} = -\varphi_k'(v_k(n)) \tag{4.22}$$

我们从图 4.4 也要注意到对神经元 k 来说，诱导局部域是：

$$v_k(n) = \sum_{j=0}^{m} w_{kj}(n) y_j(n) \tag{4.23}$$

这里 m 是神经元 k 的所有输入的个数（不包括偏置）。同样在这里突触权值 $w_{k0}(n)$ 等于作用于神经元 k 的偏置 $b_k(n)$，相应的输入是固定在值 +1 处的。求式(4.23)对 $y_j(n)$ 的微分得到：

$$\frac{\partial v_k(n)}{\partial y_j(n)} = w_{kj}(n) \tag{4.24}$$

用式(4.22)和式(4.24)代入式(4.20)，得到期望的偏微分：

$$\frac{\partial \mathscr{E}(n)}{\partial y_j(n)} = -\sum_k e_k(n)\varphi_k'(v_k(n))w_{kj}(n) = -\sum_k \delta_k(n)w_{kj}(n) \tag{4.25}$$

在第二行用到局域梯度 $\delta_k(n)$ 的定义，它由式(4.16)给出，其中用下标 k 替代 j。

最后，用式(4.25)代入式(4.17)，得到关于局域梯度 $\delta_j(n)$ 的反向传播公式：

$$\delta_j(n) = \varphi_j'(v_j(n))\sum_k \delta_k(n)w_{kj}(n), \qquad \text{神经元 } j \text{ 为隐藏单元} \tag{4.26}$$

图 4.5 代表式(4.26)的信号流图，假设输出层有 m_L 个神经元。

在式 (4.26) 中与局域梯度 $\delta_j(n)$ 的计算有关的因子 $\varphi_j'(v_j(n))$仅仅依赖于隐藏层神经元 j 的激活函数。这个计算涉及的其余因子，也就是所有神经元 k 的和，依赖于两组项。第一项的集合需要以下神经元的误差信号 $e_k(n)$的知识，这些神经元紧接隐藏层神经元 j 右端，且直接与神经元 j 相连；参看图 4.4。第二项的集合 $w_{kj}(n)$是由所有这些连接的突触权值组成的。

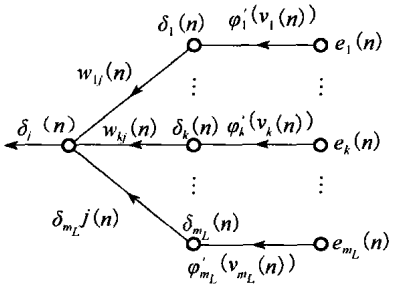


图 4.5　误差信号反向转播伴随系统的部分信号流图

现在，我们总结一下反向传播算法的导出。首先，由神经元 i 连接到神经元 j 的突触权值的校正值 $\Delta w_{ji}(n)$由 delta 规则定义如下：

$$\begin{pmatrix}\text{权值}\\ \text{校正}\\ \Delta w_{ji}(n)\end{pmatrix} = \begin{pmatrix}\text{学习率参数}\\ \eta\end{pmatrix} \times \begin{pmatrix}\text{局部梯度}\\ \delta_j(n)\end{pmatrix} \times \begin{pmatrix}\text{神经元 } j \text{ 的输入信号}\\ y_i(n)\end{pmatrix} \tag{4.27}$$

其次，局域梯度 $\delta_j(n)$取决于神经元 j 是一个输出节点还是一个隐藏层节点：

1. 如果神经元 j 是一个输出节点，$\delta_j(n)$等于导数 $\varphi_j'(v_j(n))$和误差信号 $e_j(n)$的乘积，它们都和神经元 j 相关联；参看式 (4.16)。

2. 如果神经元 j 是隐藏层节点，$\delta_j(n)$等于相应导数 $\varphi_j'(v_j(n))$和 δ_j 的加权和的乘积，这些 δ_j 是对与神经元 j 相连的下一个隐藏层或输出层中的神经元计算得到的；参看式(4.26)。

计算的两次通过

在反向传播算法的应用中，计算有两种截然不同的通过。第一个通过是指前向通过，而第二个是指反向通过。

在前向通过中，经过网络时突触权值保持不变，而网络的函数信号在一个神经元接一个神经元基础上计算。出现在神经元 j 输出处的函数信号计算为：

$$y_j(n) = \varphi(v_j(n)) \tag{4.28}$$

其中 $v_j(n)$是神经元 j 的诱导局部域，定义为：

$$v_j(n) = \sum_{i=0}^{m} w_{ji}(n)y_i(n) \tag{4.29}$$

这里，m 是神经元 j 的所有输入的数量（不包括偏置），而 $w_{ji}(n)$是连接神经元 i 和神经元 j 的突触权值，$y_i(n)$是指神经元 j 的输入信号或是出现在神经元 i 的输出端的函数信号。如果神经元 j 在网络的第一隐藏层，则 $m=m_0$ 且下标 i 是指网络的第 i 个输入端点，我们写作：

$$y_i(n) = x_i(n) \tag{4.30}$$

这里 $x_i(n)$是指输入向量（模式）的第 i 个元素。在另一方面，如果神经元 j 在网络的输出层，则 $m=m_L$，并且下标 j 是指网络的第 j 个输出端点，我们写作：

$$y_j(n) = o_j(n) \tag{4.31}$$

这里 $o_j(n)$是指多层感知器输出向量的第 j 个元素。这个输出和期望响应 $d_j(n)$相比较，得到第 j 个输出神经元的误差信号 $e_j(n)$。因此，计算的前向阶段由输入向量馈给的第一个隐藏层开始，以输出层计算该层的每一个神经元的误差信号而结束。

另一方面，反向通过从输出层开始，误差信号向左经过网络一层一层传播，并且递归计算每一个神经元的 δ（即局部梯度）。该递归过程允许突触权值根据式(4.27)的 delta 规则变化。对于位于输出层的神经元，δ 简单地等于这个神经元的误差信号乘以它的非线性一次导数。因此，我们使用式(4.27)来计算所有馈入输出层的连接的权值变化。给出输出层神经元的 δ，接着用式(4.26)来计算倒数第二层的所有神经元的 δ 和所有馈入该层的连接的权值变化。通过传播这个变化给网络的所有突触权值，一层接一层连续递归计算。

注意，由于每给出一个训练例子，其输入模式在整个往返过程中是固定的（钳制的），这个往返过程包括前向通过和随后的反向通过。

激活函数

计算多层感知器每一个神经元的 δ 需要神经元的激活函数 $\varphi(\cdot)$的导数知识。导数存在的条件是函数 $\varphi(\cdot)$连续。从根本上讲，激活函数必需满足的要求是可微性。通常用于多层感知器的连续可微非线性激活函数的一个例子是 sigmoid 非线性性[1]；这里有两种形式要说一下：

1. logistic 函数。这种 sigmoid 非线性性的一般形式由

$$\varphi_j(v_j(n)) = \frac{1}{1+\exp(-av_j(n))}, \quad a>0 \tag{4.32}$$

定义，这里 $v_j(n)$ 是神经元 j 的诱导局部域。根据这种非线性性，输出的范围位于 $0 \leqslant y_j \leqslant 1$ 之内。对式(4.32)取 $v_j(n)$的微分，得到

$$\varphi_j'(v_j(n)) = \frac{a\exp(-av_j(n))}{[1+\exp(-av_j(n))]^2} \tag{4.33}$$

由于 $y_j(n)=\varphi_j(v_j(n))$，我们可以从式(4.33)中消去指数项 $\exp(-av_j(n))$，所以导数 $\varphi_j'(v_j(n))$可以表示为：

$$\varphi_j'(v_j(n)) = ay_j(n)[1-y_j(n)] \tag{4.34}$$

因为神经元 j 位于输出层，所以 $y_j(n)=o_j(n)$。因此可以将神经元 j 的局域梯度表示为

$$\delta_j(n) = e_j(n)\varphi_j'(v_j(n)) = a[d_j(n)-o_j(n)]o_j(n)[1-o_j(n)], j \text{ 输出节点} \tag{4.35}$$

这里的 $o_j(n)$是神经元 j 输出端的函数信号，而 $d_j(n)$是它的期望响应。另一方面，对任意的一个隐藏层神经元 j，可以将局域梯度表示为：

$$\begin{aligned}\delta_j(n) &= \varphi_j'(v_j(n))\sum_k \delta_k(n)w_{kj}(n) \\ &= ay_j(n)[1-y_j(n)]\sum_k \delta_k(n)w_{kj}(n), \quad j \text{ 为隐藏神经元}\end{aligned} \tag{4.36}$$

从式 (4.34) 可以看出，导数 $\varphi_j'(v_j(n))$ 当 $y_j(n)=0.5$ 时取最大值，当 $y_j(n)=0$ 或 $y_j(n)=1$ 时取它的最小值 (0)。既然网络的一个突触权值的变化总量与导数 $\varphi_j'(v_j(n))$ 成比例，因此对于一个 sigmoid 激活函数来说，突触权值改变最多的神经元是那些函数信号在它们的中间范围之内的网络的神经元。根据 Rumelhart 等 (1986a)，正是反向传播学习的这个特点导致它作为学习算法的稳定性。

2. 双曲正切函数。另外一个经常使用的 sigmoid 非线性形式是双曲正切函数，它的最通用

的形式由

$$\varphi_j(v_j(n)) = a\tanh(bv_j(n)) \tag{4.37}$$

定义，这里 a 和 b 是正常数。事实上，双曲正切函数只是伸缩和平移的 logistic 函数。它对 $v_j(n)$ 的导数如下：

$$\varphi_j'(v_j(n)) = ab\,\mathrm{sech}^2(bv_j(n)) = ab(1-\tanh^2(bv_j(n))) = \frac{b}{a}[a-y_j(n)][a+y_j(n)] \tag{4.38}$$

如果神经元 j 位于输出层，它的局域梯度是：

$$\begin{aligned}\delta_j(n) &= e_j(n)\varphi_j'(v_j(n)) \\ &= \frac{b}{a}[d_j(n)-o_j(n)][a-o_j(n)][a+o_j(n)]\end{aligned} \tag{4.39}$$

如果神经元 j 位于隐藏层，我们有

$$\begin{aligned}\delta_j(n) &= \varphi_j'(v_j(n))\sum_k \delta_k(n)w_{kj}(n) \\ &= \frac{b}{a}[a-y_j(n)][a+y_j(n)]\sum_k \delta_k(n)w_{kj}(n)\text{，神经元 } j \text{ 为隐藏神经元}\end{aligned} \tag{4.40}$$

对 logistic 函数使用式(4.35)和式(4.36)以及对双曲正切函数使用式(4.39)和式(4.40)，不需要激活函数的具体信息就可以计算局域梯度 δ_j。

学习率

反向传播算法提供使用最速下降方法在权空间计算得到的轨迹的一种近似。使用的学习率参数 η 越小，从一次迭代到下一次迭代的网络突触权值的变化量就越小，轨迹在权值空间就越光滑。然而，这种改进是以减慢学习速度为代价的。另一方面，如果让 η 的值太大以加快学习速度的话，结果就有可能使网络的突触权值的变化量不稳定(即振荡)。一个既要加快学习速度又要保持稳定的简单方法是修改式(4.15)的 delta 法则，使它包括动量项，表示为

$$\Delta w_{ji}(n) = \alpha\Delta w_{ji}(n-1) + \eta\delta_j(n)y_i(n) \tag{4.41}$$

这里 α 是动量常数，通常是正数。它控制围绕 $\Delta w_{ji}(n)$ 的反馈环路，如图 4.6 所示，其中 z^{-1} 表示单位时间延迟操作符。式(4.41)被称之为广义 delta 规则[2]；它包括式(4.15)的 delta 规则作为特殊情况（即$\alpha=0$）。

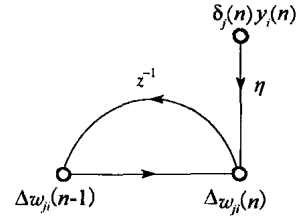


图 4.6　说明动量常数 α 作用的信号流图，它位于反馈环内部

为了观察动量常数 α 在一系列模式呈现上对突触权值的影响，我们将式(4.41)重新写为带下标 t 的一个时间序列。索引 t 从初始时刻 0 到当前时刻 n。式(4.41)可被视为权值修正量 $\Delta w_{ji}(n)$ 的一阶差分方程。解这个关于 $\Delta w_{ji}(n)$ 的方程得到

$$\Delta w_{ji}(n) = \eta\sum_{t=0}^{n}\alpha^{n-t}\delta_j(t)y_i(t) \tag{4.42}$$

它代表一个长度为 $n+1$ 的时间序列。从式(4.13)和式(4.16)，可知 $\delta_j(n)y_j(n)$ 等于 $-\partial\mathscr{E}(n)/\partial w_{ji}(n)$。因此将方程(4.42)重写为等价形式：

$$\Delta w_{ji}(n) = -\eta\sum_{t=0}^{n}\alpha^{n-t}\frac{\partial\mathscr{E}(t)}{\partial w_{ji}(t)} \tag{4.43}$$

在这个关系的基础上，我们来做以下深入观察：

1. 当前修正值 $\Delta w_{ji}(n)$ 代表指数加权的时间序列的和。欲使时间序列收敛，动量常数必须限制在 $0\leqslant|\alpha|<1$ 范围内。当 α 等于 0 时，反向传播算法运行起来没有动量。虽然在实际中

动量常数 α 不大可能是负的，但它还是可正可负。

2. 当偏导数 $\partial \mathscr{E}(t)/\partial w_{ji}(t)$ 在连续迭代中有相同的代数符号，指数加权和 $\Delta w_{ji}(n)$ 在数量上增加，所以，权值 $\Delta w_{ji}(n)$ 被大幅度调整。在反向传播算法中包含动量趋于在稳定的下降方向上加速下降。

3. 当偏导数 $\partial \mathscr{E}(t)/\partial w_{ji}(t)$ 在连续迭代中有相反的代数符号，指数加权和 $\Delta w_{ji}(n)$ 在数量上减少，所以，权值 $w_{ji}(n)$ 调整不大。在反向传播算法中包含动量具有符号正负摆动方向的稳定效果。

在反向传播算法中，动量的使用对更新权值来说是一个较小的修改，而它对算法的学习可能会有一些有利的影响。动量项可能也有益于防止学习过程停止在误差曲面上的局部最小值。

在导出反向传播算法时假设学习率参数 η 是一个常数。然而，事实上它应该被定义为 η_{ji}；也就是说，学习率参数应该是*连接依赖*（connection dependent）的。确实，在网络的不同地方使用不同的学习率参数会发生很多有趣的事情。关于这一点在后续节中我们会给出详细描述。

同样值得注意的是，我们在反向传播算法的应用中可以选择使所有突触权值都是可调整的，或者在自适应过程中可能限制网络中某些权值使其保持固定。对于后者，误差信号是以通常的方式通过网络反向传播的；然而，固定的突触权值是不改变的。这一点可以简单通过使突触权值的学习率参数 η_{ji} 等于 0 来做到。

停止准则

通常，不能证明反向传播算法是收敛的，并且没有明确定义的算法停止准则。相反，仅有一些合理的准则，它们每个都有自己的实际用处，这些准则可以用于终止权值的调整。要提出这样一个准则，考虑关于误差曲面[3]的局部或全局最小的特殊性质是符合逻辑的。将权值向量 $\mathbf{w}^*$ 标记为局部或全局最小点。要使 $\mathbf{w}^*$ 成为最小点的一个必要条件是误差曲面对权值向量 $\mathbf{w}$ 的梯度向量 $\mathbf{g}(\mathbf{w})$（即一阶偏导数）在 $\mathbf{w}=\mathbf{w}^*$ 处等于 0。因此，我们可以提出反向传播学习的一个合理的收敛准则（Kramer and Sangiovanni-Vincentelli，1989）：

当梯度向量的欧几里得范数达到一个充分小的梯度阈值时，我们认为反向传播算法已经收敛。

这个收敛准则的缺点是，为了成功试验，学习时间可能会很长。同时它需要计算梯度向量 $\mathbf{g}(\mathbf{w})$。

另一个我们能够使用的最小点的特殊性质是代价函数或误差量度 $\mathscr{E}_{av}(\mathbf{w})$ 在 $\mathbf{w}=\mathbf{w}^*$ 处是平稳的。因此，我们可以建议一个不同的收敛准则：

当每一个回合的均方误差变化的绝对速率足够小时，我们认为反向传播算法已经收敛。

均方误差变化的速率如果每个回合是在 0.1%～1%，一般认为它足够小。有时，每一个回合都会小到 0.01%这样的值。不幸的是，这个准则可能会导致学习过程的过早终止。

还存在另一个有用的且有理论支持的收敛准则。在每一步学习迭代之后，都要检查网络的泛化性能。当泛化性能是适当的，或泛化性能明显达到峰值时，学习过程被终止：第 4.13 节将介绍更多细节。

反向传播算法小结

图 4.1 给出了一个多层感知器的结构布局。图 4.7 给出了 $L=2$ 和 $m_0=m_1=m_2=3$ 的情况下反向传播学习的相应的信号流图，包括学习过程计算的前向和反向阶段。信号流图的上面一部分是说明前向通过的。信号流图的下面一部分是说明反向通过的，这也称为在反向传播算法中计算局域梯度的*灵敏图*（sensitivity graph）（Narendra and Parthasarathy，1990）。

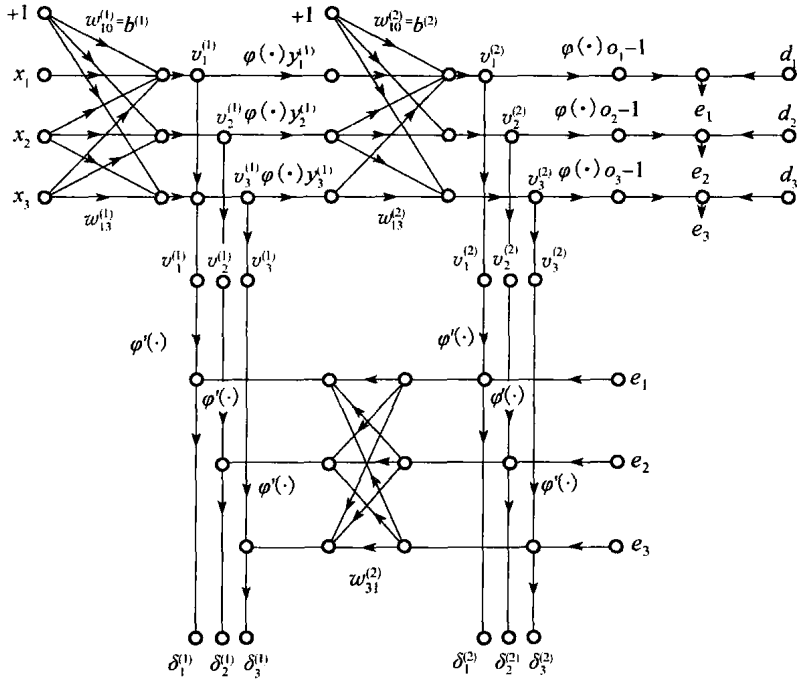


图 4.7　反向传播学习信号流图小结。图顶部：前向通过。图底部：反向通过

前面我们提到权值的串行更新是反向传播算法的在线实现的更好方法。对这种运行方式，算法通过训练样本 $\{(\mathbf{x}(n),\mathbf{d}(n))\}_{n=1}^{N}$ 进行如下循环：

1. *初始化*。假设没有先验知识可用，我们以一个一致分布来随机地挑选突触权值和阈值，这个分布选择为均值等于 0 的均匀分布，它的方差的选择应该使得神经元的诱导局部域的标准偏差位于 sigmoid 激活函数的线形部分与饱和部分过渡处。

2. *训练样本的呈现*。呈现训练样本的一个回合给网络。对训练集中以某种形式排序的每个样本，依次进行下面的第 3 点和第 4 点中所描述的前向和反向计算。

3. *前向计算*。在该回合中设一个训练样本是 $(\mathbf{x}(n),\mathbf{d}(n))$，输入向量 $\mathbf{x}(n)$作用于感知节点的输入层，期望响应向量 $\mathbf{d}(n)$指向计算节点的输出层。不断经由网络一层一层地前进，可以计算网络的诱导局部域和函数信号。在层 l 的神经元 j 的诱导局部域$v_j^{(l)}(n)$为

$$v_j^{(l)}(n) = \sum_i w_{ji}^{(l)}(n) y_i^{(l-1)}(n) \tag{4.44}$$

这里 $y_i^{(l-1)}(n)$是迭代 n 时前面第 $l-1$ 层的神经元 i 的输出（函数）信号，而 $w_{ji}^{(l)}(n)$是从第 $l-1$ 层的神经元 i 指向第 l 层的神经元 j 的权值。对 $i=0$，我们有 $y_0^{(l-1)}(n)=+1$，并且 $w_{j0}^{(l)}(n)=b_j^{(l)}(n)$是第 l 层的神经元 j 的偏置。假设使用一个 sigmoid 函数，则第 l 层的神经元 j 的输出信号是：

$$y_j^{(l)} = \varphi_j(v_j(n))$$

如果神经元 j 是在第一隐藏层（即 $l=1$），置

$$y_j^{(0)}(n) = x_j(n)$$

这里 $x_j(n)$是输入向量 $x(n)$的第 j 个元素。如果神经元 j 在输出层（即 $l=L$，这里的 L 称为网络的深度），令

$$y_j^{(L)} = o_j(n)$$

计算误差信号

$$e_j(n) = d_j(n) - o_j(n) \tag{4.45}$$

这里 $d_j(n)$是期望响应向量 $\mathbf{d}(n)$的第 j 个元素。

4. 反向计算。计算网络的 δ（即局域梯度），定义为：

$$\delta_j^{(l)}(n)=\begin{cases}e_j^{(L)}(n)\varphi_j'(v_j^{(L)}(n)), & \text{对输出层 } L \text{ 的神经元 } j\\ \varphi_j'(v_j^{(l)}(n))\sum_k \delta_k^{(l+1)}(n)w_{kj}^{(l+1)}(n), & \text{对隐藏层 } l \text{ 的神经元 } j\end{cases} \tag{4.46}$$

这里 $\varphi_j'(\cdot)$是指对自变量的微分。根据广义 delta 规则调节网络第 l 层的突触权值：

$$w_{ji}^{(l)}(n+1)=w_{ji}^{(l)}(n)+\alpha[w_{ji}^{(l)}(n-1)]+\eta\delta_j^{(l)}(n)y_i^{(l-1)}(n) \tag{4.47}$$

这里 η 为学习率参数，α 为动量常数。

5. 迭代。通过呈现新的一回合样本给网络并根据第 3 和第 4 进行前向和反向迭代计算，直到满足停止准则。

注意：训练样本的呈现顺序从一个回合到另一个回合必须是随机的。动量和学习率参数随着训练迭代次数的增加而调整（通常是减少的）。以后会给出这些注意点的理由。

4.5　异或问题

在 Rosenblatt 单层感知器中，没有隐藏神经元。因此，它不能对非线性可分的输入模式进行分类。然而，非线性可分模式却是普遍存在的。例如，对异或（XOR）问题就遇到这种情形，它可以看作在单位超立方体中更一般的点分类问题的特例。在超立方体中的每个点不是属于类 0 就是属于类 1。但是对异或问题特殊情形，我们仅考虑单位正方形的四个角，相应的输入模式为(0，0)，(0，1)，(1，0) 和(1，1)。第一个和第三个输入模式属于类 0，即

$$0\oplus 0=0$$

和

$$1\oplus 1=0$$

这里⊕指的是异或布尔函数运算符。输入模式（0，0）和（1，1）是单位正方形的两个相对的角，但它们产生相同的结果是 0。另一方面，输入模式（0，1）和（1，0）是单位正方形的另一对相对的角，但是它们属于类 1，即

$$0\oplus 1=1$$

和

$$1\oplus 0=1$$

首先我们知道有两个输入的单个神经元的使用得到的决策边界是输入空间的一条直线。在这条直线的一边的所有的点，神经元输出 1；而在这条直线的另一边的点，神经元输出 0。在输入空间中，这条直线的位置和方向由与两个输入节点相连的神经元的突触权值和它的偏置决定。由于输入模式（0，0）和（1，1）是位于单位正方形的相对的两个角，输入模式（0，1）和（1，0）也一样，很明显我们做不出这样一条直线作为决策边界可以使（0，0）和（1，1）在一个区域，而（1，0）和（0，1）在另一区域。换句话说，一个单层感知器不能解决 XOR 问题。

然而，如图 4.8a 中所示，我们可以使用一层有两个神经元的隐藏层来解决异或问题（Touretzky and Pomerleau，1989）。网络的信号流图在图 4.8b 中给出。这里做以下假设：

- 每一个神经元都由一个 McCulloch-Pitts 模型表示，使用阈值函数作为它的激活函数。
- 比特符号 0 和 1 分别由水平 0 和+1 表示。

隐藏层顶部的神经元标记为“神经元 1”，有

$$w_{11}=w_{12}=+1 \quad b_1=-\frac{3}{2}$$

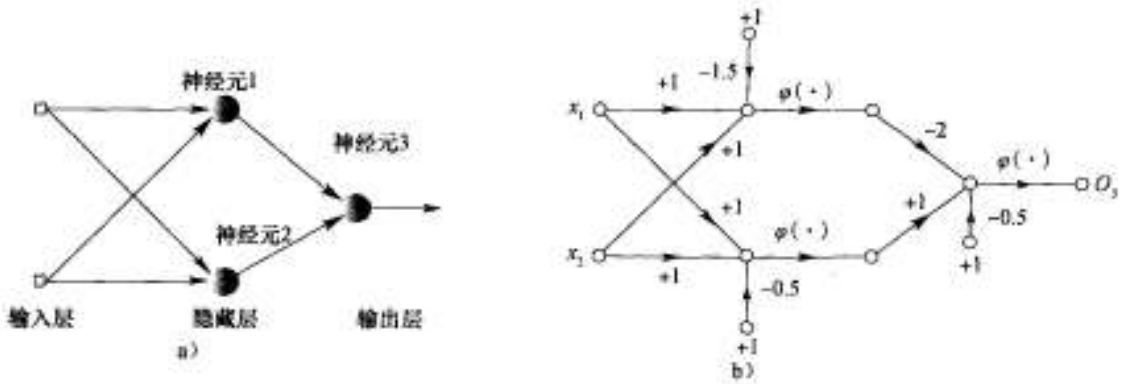


图 4.8　a) 解决 XOR 问题的网络结构图；b) 网络信号流图

该隐藏神经元构造的决策边界的斜率等于 −1，在图 4.9a 中给出其位置。在隐藏层的底部神经元标记为“神经元 2”，有

$$w_{21} = w_{22} = +1 \quad b_2 = -\frac{1}{2}$$

第二隐藏神经元构造的决策边界的方向和位置由图 4.9b 给出。

图 4.8a 的标记为“神经元 3”的输出神经元定义为

$$w_{31} = -2 \quad w_{32} = +1 \quad b_3 = -\frac{1}{2}$$

输出神经元的功能是对两个隐藏神经元形成的决策边界构造线性组合。这个计算结果表示在图 4.9c中。底部隐藏神经元由一个兴奋（正）连接到输出神经元，而顶部隐藏神经元由一个更强的抑制（负）连接到输出神经元。当两个隐藏神经元都断开时，这种情况当输入信号是 (0，0) 时发生，输出神经元保持断开。当两个隐藏神经元都接通时，这种情况当输入模式是 (1，1) 时发生，输出神经元也保持断开，因为由连向顶部隐藏神经元负权值产生的抑制效果超过由连向底部隐藏神经元正权值产生的兴奋效果。当顶部隐藏神经元是断开的而底部隐藏神经元是接通的，即输入模式是 (0，1) 或 (1，0) 时，输出神经元是接通的，因为正的权值连向了底部隐藏神经元。因此图 4.8a 确实解决了异或问题。

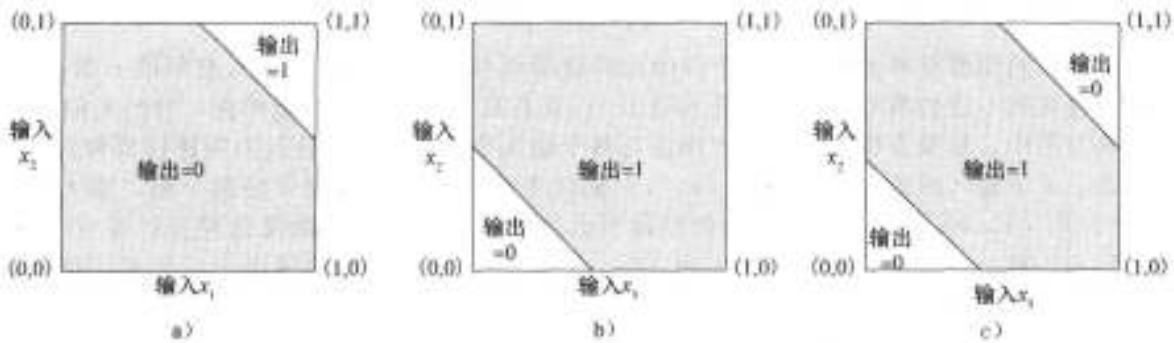


图 4.9　a) 在图 4.8 中的网络隐藏神经元 1 所构造的决策边界；b) 网络隐藏神经元 2 所构造的决策边界；c) 整个网络所构造的决策边界

4.6　改善反向传播算法性能的试探法

人们常说，用于反向传播算法的神经网络的设计与其说是科学，不如说更像一门艺术，因

为这个设计中的很多数值因素依赖于个人自己的经验。从某种意义上讲这个论断是正确的。但是，也有些方法能对反向传播算法有显著提高，如下所述：

1. *随机和批量方式更新*。如前面已经提到过的，反向传播学习的随机（串行）方式（涉及一个模式接一个模式的更新）要比批量方式的计算快。特别是当训练数据集很大且高度冗余时，更是如此。（高度冗余的数据对批量方式更新所需要的 Jacobi 矩阵的估计提出了计算上的问题。）

2. *最大信息内容*。作为一个基本的规则，对呈现给反向传播算法的每一个训练样本的挑选必须建立在其信息内容对解决问题有最大可能的基础上（LeCun，1993）。达到这个目标的两种方法是：

- 使用训练误差最大的样本。
- 使用的样本要与以前使用的有根本区别。

这两个试探方法起因于对权空间进行更多搜索的愿望。

在模式分类的任务中使用串行反向传播学习，经常使用的一个简单技巧是将样本的每个回合呈现给多层感知器的顺序随机化（即弄乱）。理想情况下，随机化可以确保一个回合中的相继的样本很少属于同一类。

3. *激活函数*。在考虑学习速度的情况下，较好的选择是采用关于其自变量为奇函数的 sigmoid激活函数，即

$$\varphi(-v) = -\varphi(-v)$$

如下的双曲函数是满足这个条件的

$$\varphi(v) = a\tanh(bv)$$

如图 4.10 所示，但是 logistic 函数不满足这个条件。在 $\varphi(v)$中系统规定参数 a 和 b 的合适的值是（LeCun，1989，1993）

$$a = 1.715\,9$$

和

$$b = \frac{2}{3}$$

图 4.10 的双曲正切函数有如下有用的性质：

- $\varphi(1)=1$ 和 $\varphi(-1)=-1$。
- 在原点激活函数的倾斜度（即有效增益）接近于 1，如下所示：

$$\varphi(0) = ab = 1.715\,9\left(\frac{2}{3}\right) = 1.142\,4$$

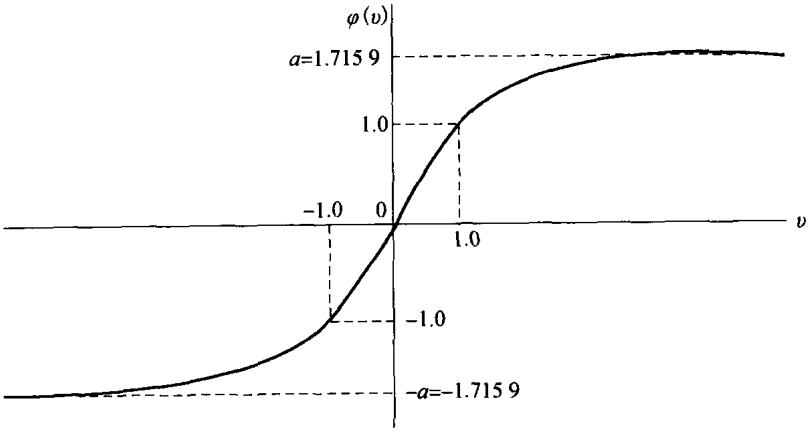


图 4.10　在 $a=1.715\,9$ 和 $b=2/3$ 时的双曲正切函数 $\varphi(v)=a\tanh(bv)$的图形。推荐的目标值是+1 和−1

- $\varphi(v)$ 的二阶导数在 $v=1$时达到最大。

4. *目标值*。在 sigmoid 激活函数的范围内选择目标值（期望响应）是很重要的。更具体来说，多层感知器输出层的神经元 j 的期望响应 d_j 必须与 sigmoid 激活函数的极限值偏离某个 ε 值，具体取决于极限值是正或负。否则反向传播算法会使网络的自由参数趋向于无穷大，驱使隐藏神经元达到饱和从而减慢学习过程。具体讲，考虑图 4.10 所示的双曲正切函数。对于极限值$+a$，我们令

$$d_j = a - \varepsilon$$

对于有限值$-a$，我们令

$$d_j = -a + \varepsilon$$

这里 ε 是一个合适的正常数。对图 4.10 中选择的 $a=1.7159$，可以令 $\varepsilon=0.7159$，这样，目标值 d_j 可以方便地选为± 1，正如图所示的那样。

5. *输入的标准化*。每一个输入变量都需要预处理，使得它关于整个训练集求平均的均值接近 0，或者与标准偏差相比是比较小的（LeCun，1993）。为评价这个规则的实际意义，我们考虑输入恒正的极端情况。在这种情况下，第一隐藏层的一个神经元的所有突触权值只能同时增加或同时减少。所以，如果这个神经元权值向量改变方向，则它的误差曲面的路径变成锯齿形的，这会使收敛速率变慢，因此应该避免。

要加速反向传播学习的过程，输入变量的标准化必须包括下面两个步骤（LeCun，1993）：

- 训练集包含的输入变量应该是不相关的；这可以通过第 8 章提到的主分量分析法来做到。
- 去相关后的输入变量应调整其长度使得它们的协方差近似相等，因此可以保证网络中的不同突触权值以大约相等的速度进行学习。

图 4.11 说明依次执行三个标准化步骤的结果：消除均值、去相关性以及协方差均衡。

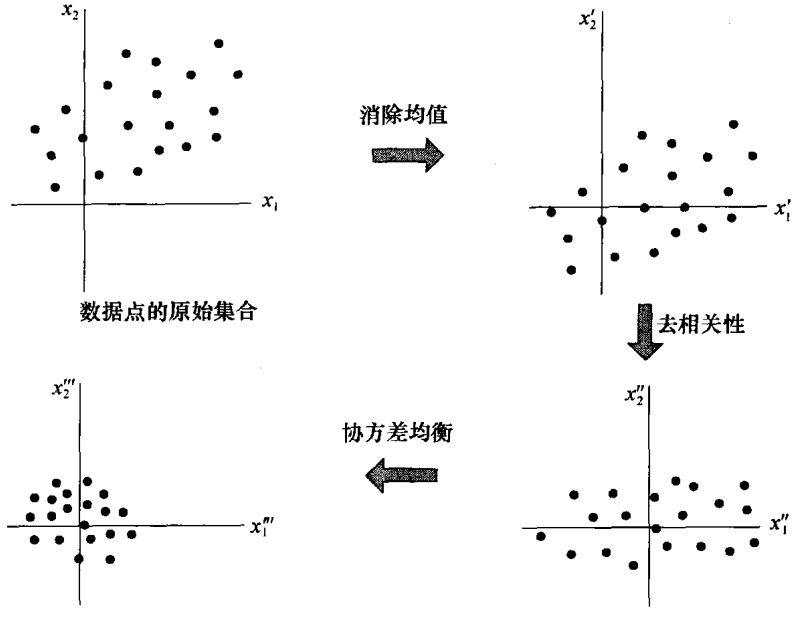


图 4.11　二维输入空间的消除均值、去相关性以及协方差均衡运算的图示

另一个有趣的现象是当通过图 4.11 的方式对输入进行变换，并将之和图 4.10 所示的双曲正切函数结合起来时，多层感知器中各个神经元的输出的方差接近于 1（Orr and Muller，1998）。这一声明的基本原理在于在有用范围内有效获得的 sigmoid 函数是大体上为 1 的。

6. *初始化*。网络的突触权值和阈值初值的一个较好的选择对一个成功的网络设计会有巨大帮助。关键问题是：什么是好的选择？

当突触权值被赋予一个较大的初始值时，网络的神经元很可能会趋于饱和。如果发生这种情况，反向传播算法中的局域梯度呈现出一个很小的值，结果导致反向传播学习过程很缓慢。然而，如果突触权值被赋予一个较小的初始值，反向传播算法可能就在误差曲面的原点的一个非常平缓的区域内进行；特别对于如双曲正切函数这样的 sigmoid 函数时，这种可能性就更大。不幸的是，原点是一个鞍点，这个鞍点是一个稳定点，在该点处与鞍正交的误差曲面的曲率为负，而沿着鞍方向为正。由于这些原因，使用过大或过小值初始化突触权值都应该避免。恰当的初始化选择位于这两种极端之间。

具体地说，考虑将一个双曲正切函数作为激活函数的多层感知器。设网络的每一个神经元的偏置为 0。我们将神经元 j 的诱导局部域表示为

$$v_j = \sum_{i=1}^{m} w_{ji} y_i$$

假设网络的每一个神经元的输入的均值为 0 方差为 1，表示为

$$\mu_y = \mathbb{E}[y_i] = 0 \qquad \text{对所有神经元 } i$$

和

$$\sigma_y^2 = \mathbb{E}[(y_i - \mu_i)^2] = \mathbb{E}[y_i^2] = 1 \qquad \text{对所有神经元 } i$$

进一步，假设输入值都是不相关的，即

$$\mathbb{E}[y_i y_k] = \begin{cases} 1 & \text{对 } k = i \\ 0 & \text{对 } k \neq i \end{cases}$$

并且设突触权值的值是以均值为 0 的均匀分布抽取的一组数，即

$$\mu_w = \mathbb{E}[w_{ji}] = 0 \qquad \text{对所有}(j,i)\text{对}$$

和方差

$$\sigma_w^2 = \mathbb{E}[(w_{ji} - \mu_w)^2] = \mathbb{E}[w_{ji}^2] \qquad \text{对所有}(j,i)\text{对}$$

因此可以将诱导局部域 v_j 的均值和方差表示为

$$\mu_v = \mathbb{E}[v_j] = \mathbb{E}\Big[\sum_{i=1}^{m} w_{ji} y_i\Big] = \sum_{i=1}^{m} \mathbb{E}[w_{ji}]\mathbb{E}[y_i] = 0$$

和

$$\sigma_v^2 = \mathbb{E}[(v_j - \mu_v)^2] = \mathbb{E}[v_j^2] = \mathbb{E}\Big[\sum_{i=1}^{m}\sum_{k=1}^{m} w_{ji} w_{jk} y_i y_k\Big] = \sum_{i=1}^{m}\sum_{k=1}^{m} \mathbb{E}[w_{ji} w_{jk}]\mathbb{E}[y_i y_k] = \sum_{i=1}^{m} \mathbb{E}[w_{ji}^2] = m\sigma_w^2$$

这里 m 是一个神经元的突触连接的数目。

根据上述结果，我们可以得到一个如何初始化突触权值的一个好策略，使得神经元诱导局部域的标准偏差位于它的 sigmoid 激活函数的线性部分和饱和部分的过渡区域。例如，如图 4.10 所示的参数 a 和 b 所设值的双曲正切函数，当上式中的 $\sigma_v = 1$ 时可以满足这个目标，这样得到（LeCun，1993）：

$$\sigma_w = m^{-1/2} \tag{4.48}$$

因此，对于一个均匀分布，它需要其均值为 0 而方差将与神经元的突触连接的数目成反比，从而以这个分布来选择突触权值的值。

7. 从提示中学习。从一组未知的训练例子中学习意味着处理未知的输入—输出映射函数 $f(\cdot)$。事实上，学习过程利用函数 $f(\cdot)$例子所包含的信息来推断它的逼近实现。从例子中学习的过程可以推广为包括从提示中学习，这可以通过在学习过程中加入函数 $f(\cdot)$的先验知识来实现（Abu-Mostafa，1995）。这些知识包括不变性、对称性或关于函数 $f(\cdot)$的其他知识，它们可以用来加速实现 $f(\cdot)$的逼近的搜索，而且更重要的是，会提高最后估计的质量。式(4.48)的使用就是如何从提示中学习的例子。

8. 学习率。多层感知器的所有神经元理论上应以同一速率进行学习。网络最后一层的局域梯度通常比别的层大。因此，最后一层的学习率参数 η 应设得比别的层小。输入较多的神经元的学习率参数应比输入较少的神经元小。LeCun(1993) 中提到，对一个给定的神经元，其学习率应与该神经元的突触连接的平方根成反比。

4.7　计算机实验：模式分类

在本节的计算机实验中，我们回顾模式分类实验的序列，首先在第 1 章中利用 Rosenblatt 感知器，然后在第 2 章中利用了最小二乘法。对上述的两个实验，我们都采用图 1.8 所示的双月结构来随机产生训练和测试数据样本。在上述的每个实验中，我们都考虑了两种情形，一种是线性可分模式，另一种是非线性可分模式。感知器对于 $d=1$ 时的线性可分情形工作得非常好，但是最小二乘法需要在两个月亮之间更大的分隔度以便得到好的分类。在两个方法下，他们对于 $d=-4$ 的非线性可分的情形都失败了。

这里的计算机实验的目的包括两方面：

1. 用来说明通过反向传播算法训练的多层感知器，能够分类非线性可分测试数据。

2. 找到更困难的非线性可分的情形，这时候多层感知器对于双月分类测试来说失败了。

实验中使用的多层感知器的具体情况如下所示：

输入层大小：$m_0=2$

隐藏层（仅有的）大小：$m_1=20$

输出层大小：$m_2=1$

激活函数：双曲正切函数 $\varphi(v)=\dfrac{1-\exp(-2v)}{1+\exp(-2v)}$

阈值设置：0

学习率参数 η：从 10^{-1} 下降到 10^{-5} 的线性退火

实验分为两部分，一部分相应于垂直可分的 $d=-4$，另一部分相应于 $d=-5$。

(a) 垂直分隔 $d=-4$

图 4.12 是两月之间长度 $d=-4$ 时候的 MLP 实验的结果。图 4.12a 是训练阶段所产生的学习曲线。我们看到在训练了大约 15 个回合时学习曲线有效收敛。图 4.12b 显示了 MLP 计算的最优非线性决策边界。更重要的是，实现了这两种模式的良好分类，没有分类误差。这一完美性能的实现应归因于 MLP 的隐藏层。

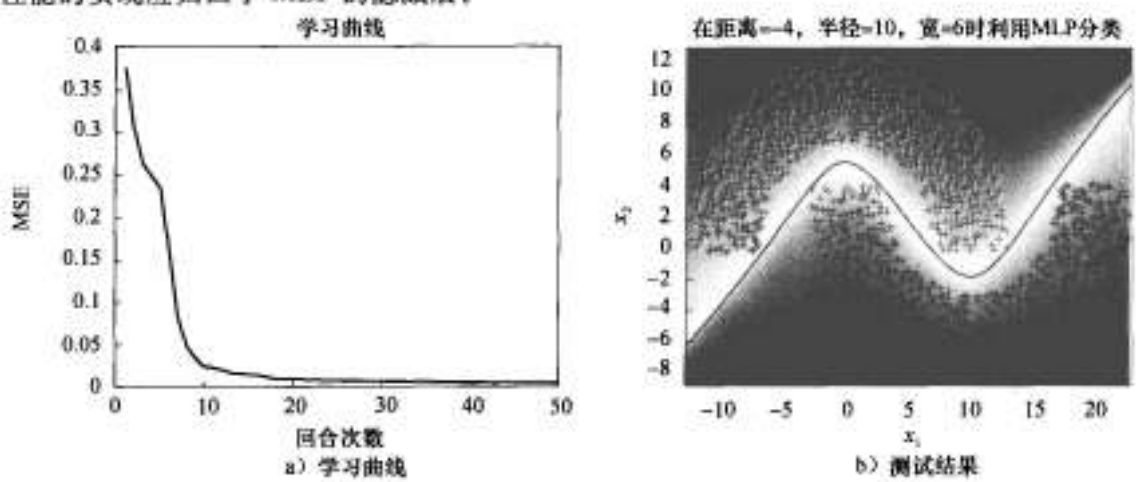


图 4.12　距离 $d=-4$ 时反向传播算法作用于 MLP 的计算机实验结果，MSE 是指均方误差

(b) 垂直分隔 $d=-5$

为了用更加困难的模式分类任务来挑战多层感知器，我们在两月之间减少垂直可分性，令 $d=-5$。实验第二部分的结果如图 4.13 所示。反向传播算法的学习曲线在图 a 部分画出，说明了较慢的收敛速度，大概是容易情形 $d=-4$ 的三倍左右。而且，在图 b 部分所给出的测试结果揭示了在 2 000 个数据点组成的测试集中有三个分类错误，表示了 0.15%的误差率。

决策边界是通过寻找属于输入向量 $\mathbf{x}$ 的坐标 x_1 和 x_2 来计算的，对于它来说，在实验的两个类是等可能的假设下，输出神经元的响应是 0。相应地，当超过阈值 0 时，做出它属于某个类的决策；反之，给出决策属于另一个类。这一过程在本书中报告的所有关于双月分类实验的报告中都将继续。

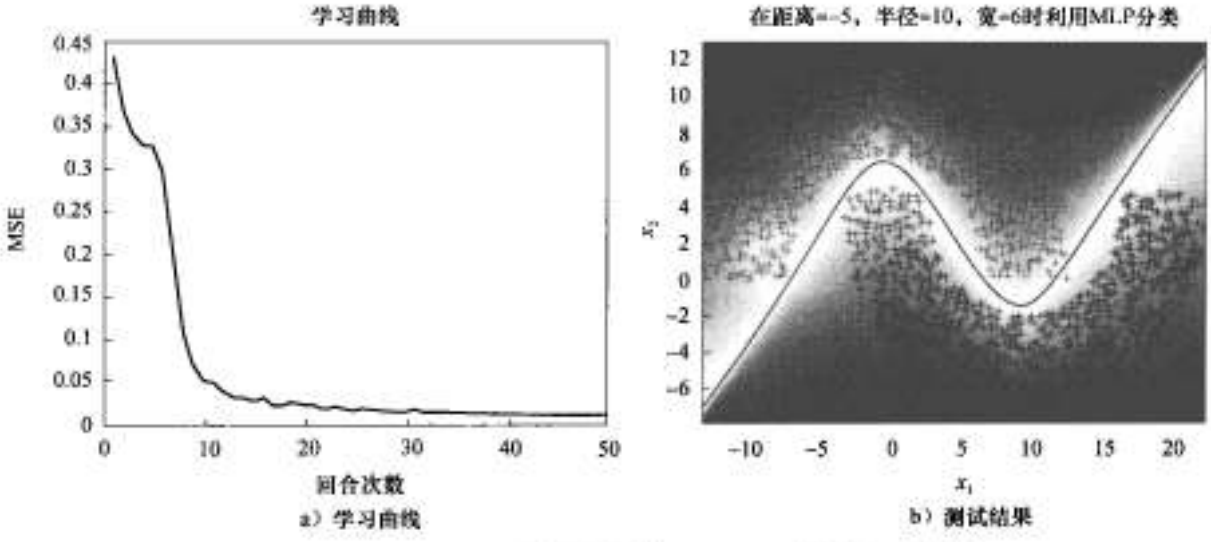


图 4.13 距离 $d=-5$ 时反向传播算法作用于 MLP 的计算机实验结果

4.8 反向传播和微分

反向传播是用于在多层前馈网络的权值空间中实现梯度下降的一种特殊技巧。其基本思想是有效计算一个近似函数 $F(\mathbf{w},\mathbf{x})$ 的偏导数，对于给定输入向量 $\mathbf{x}$ 的值近似函数 $F(\mathbf{w},\mathbf{x})$ 由网络根据可调整权值向量 $\mathbf{w}$ 的所有元素实现。这一点决定了反向传播算法的计算能力[4]。

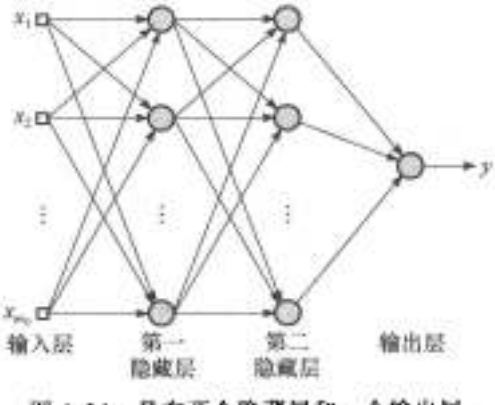


图 4.14 具有两个隐藏层和一个输出层的多层感知器

具体来说，假定一个多层感知器有一个 m_0 个节点的输入层，两个隐藏层，以及一个单一的输出神经元，如图 4.14 所示。权值向量 $\mathbf{w}$ 的元素根据层数(从第一个隐藏层开始)，然后根据层内的神经元，最后根据神经元中突触的数目来排序。令 $w_{ij}^{(l)}$ 表示从神经元 i 到层 $l=0,1,2,\cdots$ 中的神经元 j 的突触权值。对于 $l=1$，对应于第一个隐藏层，序号 i 表示一个源结点而不是一个神经元；对于 $l=3$，对应于图 4.14 的输出层，我们有 $j=1$。对于一个特定的输入向量 $\mathbf{x}=[x_1,x_2,\cdots,x_{m_0}]^T$，我们希望计算函数 $F(\mathbf{w},\mathbf{x})$ 对向量 $\mathbf{w}$ 的所有元素的导数值。将权值向量 $\mathbf{w}$ 作为函数 F 的变量，并将注意力放在其上。例如，对于 $l=2$(即一个单一隐藏层和一个线性输出层)，我们有：

$$F(\mathbf{w},\mathbf{x}) = \sum_{j=0}^{m_1} w_{oj}\varphi\Big(\sum_{i=0}^{m_0} w_{ji}x_i\Big) \tag{4.49}$$

其中 $\mathbf{w}$ 是排序后的权值向量，$\mathbf{x}$ 是输入向量。

图 4.14 的多层感知器被结构$\mathscr{A}$(表示一个离散参数) 和一个权值向量 $\mathbf{w}$(由连续的元素组成) 参数化。令$\mathscr{A}_j^{(l)}$ 表示从输入层 ($l=0$) 到层 $l=1,2,3$ 内的节点 j 所扩展成的部分结构。因此，我们可以写成：

$$F(\mathbf{w},\mathbf{x}) = \varphi(\mathscr{A}_1^{(3)}) \tag{4.50}$$

这里 φ 是激活函数。然而，$\mathscr{A}_1^{(3)}$ 仅仅被认为是一个结构符号而不是一个变量，因此，改写式(4.2)、式(4.4)、式(4.13)和式(4.25)使之在这种情况下可用，得到如下结果：

$$\frac{\partial F(\mathbf{w},\mathbf{x})}{\partial w_{1k}^{(3)}} = \varphi'(\mathscr{A}_1^{(3)})\varphi(\mathscr{A}_k^{(2)}) \tag{4.51}$$

$$\frac{\partial F(\mathbf{w},\mathbf{x})}{\partial w_{kj}^{(2)}} = \varphi'(\mathscr{A}_1^{(3)})\varphi'(\mathscr{A}_k^{(2)})\varphi(\mathscr{A}_j^{(1)})w_{1k}^{(3)} \tag{4.52}$$

$$\frac{\partial F(\mathbf{w},\mathbf{x})}{\partial w_{ji}^{(1)}} = \varphi'(\mathscr{A}_1^{(3)})\varphi'(\mathscr{A}_j^{(1)})x_i\Big[\sum_k w_{1k}^{(3)}\varphi'(\mathscr{A}_k^{(2)})w_{kj}^{(2)}\Big] \tag{4.53}$$

这里 φ'是非线性 φ 关于其输入的偏导数，x_i 是输入向量 x 的第 i 个元素。用相似的方法可以得到一般的具有更多的隐藏层和在输出层上有更多神经元的网络的偏导等式。

对于计算网络函数 $F(\mathbf{w},\mathbf{x})$关于权值向量 $\mathbf{w}$ 的元素变化的灵敏度，式(4.51)至式(4.53)提供了基础。令 ω 表示权值向量 $\mathbf{w}$ 的元素，$F(\mathbf{w},\mathbf{x})$关于 ω 的灵敏度定义为

$$S_\omega^F = \frac{\partial F/F}{\partial \omega/\omega}$$

由于这个原因我们把图 4.7 中信号流图的较低部分称为“灵敏度图”。

Jacobi 矩阵

令 W 表示一个多层感知器自由参数（即突触权值和偏置）的总数，参数按形成权值向量 $\mathbf{w}$ 的方式排序。令 N 表示用于训练网络的样本总数。对于训练集中的给定样本 $\mathbf{x}(n)$，利用反向传播可以计算近似函数 $F[\mathbf{w},\mathbf{x}(n)]$对权值向量 $\mathbf{w}$ 元素的偏导数。对于 $n=1,2,\cdots,N$ 重复上述计算，最后得到一个 $N\times W$ 的偏导数矩阵。这个矩阵被称为多层感知器的在 $\mathbf{x}(n)$处的 Jacobi 矩阵 $\mathbf{J}$。Jacobi 矩阵每列对应于训练集中的一个样本。

实验证据显示许多神经网络训练问题是内在“坏条件的”(ill conditioned)，导致 Jacobi 矩阵 $\mathbf{J}$ 几乎总是秩亏损的 (Saarinen 等，1991)。矩阵的秩是矩阵的列或行的线性无关组的数目中最小的一个。假如秩小于 $\min(N, W)$，我们说 Jacobi 矩阵 $\mathbf{J}$ 是秩亏损的。在 Jacobi 矩阵中任何的秩亏损导致反向传播算法仅仅得到可能搜寻方向上的部分信息，从而导致训练时间过长。

4.9　Hessian 矩阵及其在在线学习中的规则

代价函数$\mathscr{E}_{av}(\mathbf{w})$的 Hessian 矩阵用 $\mathbf{H}$ 表示，定义为$\mathscr{E}_{av}(\mathbf{w})$对权值向量 $\mathbf{w}$ 的二阶导数，显示为

$$\mathbf{H} = \frac{\partial^2 \mathscr{E}_{av}(\mathbf{w})}{\partial \mathbf{w}^2} \tag{4.54}$$

Hessian 矩阵在研究神经网络中起着重要作用；尤其要提出以下几点[5]：

1. Hessian 矩阵的特征值对反向传播学习动力学有着深远的影响；

2. Hessian 矩阵的逆为从一个多层感知器中修剪（即删除）不重要的突触权值提供基础，这一问题将在 4.14 节中讨论；

3. Hessian 矩阵是形成二阶优化方法的基础，二阶优化方法可作为反向传播学习的替代，这将在 4.16 节讨论。

本节将注意力放在第 1 点。

第 3 章说明了 Hessian 矩阵的特征结构对 LMS 算法的收敛性质有重大影响。它对反向传播算法也一样，但是更为复杂。一般情况下，用反向传播算法来训练的多层感知器，其误差曲面的 Hessian 矩阵有如下的特征值组合（LeCun 等，1998）：

- 小特征值的数目较少。
- 中等大小的特征值的数目很多。
- 大特征值的数目较少。

因此 Hessian 矩阵的特征值伸展范围较广。

影响特征值组合的因素可分组如下：

- 非零均值的输入信号或非零均值的神经元诱导输出信号。
- 输入信号向量的元素之间的相关性和神经元诱导输出信号之间的相关性。
- 代价函数对于网络中神经元突触权值的二阶导数随着从一层到下一层进行处理有很宽的变化范围。在较低的层中二阶导数通常更小，这样突触权值在第一隐藏层的学习很慢，但在后面的层就学习较快。

避免非 0 均值输入

回顾一下第 3 章，我们讲过 LMS 算法的学习时间对条件数 $\lambda_{max}/\lambda_{min}$ 的变化很灵敏，这里 λ_{max} 是 Hessian 矩阵最大的特征值，而 λ_{min} 是 Hessian 矩阵最小的非 0 特征值。实验结果显示反向传播算法有着相似的结果，反向传播算法是 LMS 算法的一个推广。对于非零均值的输入，它的比值 $\lambda_{max}/\lambda_{min}$ 比相应的零均值输入的比值要大：输入的均值越大，比值 $\lambda_{max}/\lambda_{min}$ 越大。这个结果对反向传播学习动力学有着重要意义。

为了使学习时间最小化，应避免使用非零均值的输入。现在，考虑将单个向量 $\mathbf{x}$ 应用于一个多层感知器的第一隐藏层的神经元这种情况，$\mathbf{x}$ 应用于网络之前先对它的每个元素减去平均值是很容易的。但是将信号应用到剩下的隐藏层和输出层中的神经元情况又会如何呢？这个问题的答案在于网络中使用的激活函数的类型。在采用 logistic 函数的情形下，每个神经元的输出界于 [0,1] 区间。这样的选择为那些位于网络中第一隐藏层之后的神经元带来了一个系统偏差源。为了克服这一问题，我们需要利用一个如同双曲正切函数的奇对称函数。对于后一种选择，每个神经元的输出可以是区间 [−1,1] 中的任何正值和负值，在这种情况下，它的均值可能为 0。假如网络连接数很大，用奇对称激活函数的反向传播学习可能比一个使用非对称激活函数的相似过程有着更快的收敛。这为 4.6 节描述的启发 3 提供了合理性依据。

在线学习的渐进行为

为了更好地理解在线学习，我们需要知道总体-平均学习曲线是如何随着时间演化的。和 LMS 算法不同，很遗憾这样的计算是很难实现的。一般来说，因为网络的对称性误差性能曲面可能有以指数方式存在的多个局部最小点和若干全局最小点。令人惊讶的是，误差性能曲面的这一特性可能反过来从以下意义上说是有用的特征：假设在网络训练中采用了早期停止方法（参照 4.13 节）或者网络是正则的（参照 4.14 节），我们几乎总是发现我们“靠近了”局部最小点。

在很多情况下，由于误差性能曲面的复杂性，我们从文献中发现，学习曲线的统计分析限定在局部最小点邻域的渐进行为上。这里重点介绍这一渐进行为的几个重要方面如下，假设学

习率参数是固定的：

（1）学习曲线包含三项：

- 最小损失，由最优参数 $\mathbf{w}^*$ 决定，它属于局部或全局最小点。
- 附加损失，由权值向量估计 $w(n)$ 在均值附近的波动引起：

$$\lim_{n\to\infty}\mathbb{E}[\hat{\mathbf{w}}(n)] = \mathbf{w}^*$$

- 时间依赖项，描述算法性能的误差收敛降速效应。

（2）为了保证在线学习算法的稳定性，学习率参数 η 必须被赋予一个小于 Hessian 矩阵最大特征值倒数 $1/\lambda_{max}$ 的值。另一方面，算法的收敛速度是由 Hessian 矩阵的最小特征值 λ_{min} 来支配的。

（3）粗略地说，如果学习率参数 η 被赋予一个大的值，收敛速度是快的，但是在局部或者全局最小点附近会有大的波动，甚至迭代次数 n 趋于无穷大时也是如此。相反，如果 η 赋予小的值，波动程度会变小，但收敛速度也会变慢。

4.10　学习率的最优退火和自适应控制

在 4.2 节中，我们强调了在线学习的流行有两个原因：

（1）算法简单，其执行只需要极少量的存储，存储量仅仅用来存放从一次迭代到下一次迭代估计权值向量的旧值。

（2）在每一个时间步每一个样本 $\{\mathbf{x},\mathbf{d}\}$ 仅仅使用一次，在线学习的学习率比批量学习的学习率有着更加重要的作用，因为在线学习算法具有追踪用来产生训练集样本的环境的统计变化的内在能力。

Amari(1967) 和最近的 Opper(1996) 中证明了具有最优退火的在线学习能够在渐进意义下和批量学习运行得一样快。下面的内容探讨了这一问题。

学习率的最优退火

令 $\mathbf{w}$ 记为网络的突触权值向量，在某种排序方式下堆叠。$\hat{\mathbf{w}}(n)$ 记为权值向量 $\mathbf{w}$ 在时间步 n 的老的估计，令 $\hat{\mathbf{w}}(n+1)$ 记为在接收到“输入-期望”样本 $\{\mathbf{x}(n+1),\mathbf{d}(n+1)\}$ 后 $\mathbf{w}$ 的更新估计。相应地，令 $\mathbf{F}(\mathbf{x}(n+1);\hat{\mathbf{w}}(n))$ 记为网络对于输入 $\mathbf{x}(n+1)$ 所产生的向量值输出；自然地，函数 $\mathbf{F}$ 的维数必须与期望响应向量 $\mathbf{d}(n)$ 相同。根据式(4.3)的定义公式，可以将瞬时能量表示为估计误差的平方欧几里得范数，如下式所示：

$$\mathscr{E}(\mathbf{x}(n),\mathbf{d}(n);\mathbf{w}) = \frac{1}{2}\|\mathbf{d}(n)-\mathbf{F}(\mathbf{x}(n);\mathbf{w})\|^2 \tag{4.55}$$

在线学习问题的均方误差或期望风险定义为：

$$J(\mathbf{w}) = \mathbb{E}_{\mathbf{x},\mathbf{d}}[\mathscr{E}(\mathbf{x},\mathbf{d};\mathbf{w})] \tag{4.56}$$

其中 $\mathbb{E}_{\mathbf{x},\mathbf{d}}$ 是作用于样本 $\{\mathbf{x},\mathbf{d}\}$ 上的期望算子。解

$$\mathbf{w}^* = \arg\min_{\mathbf{w}}[J(\mathbf{w})] \tag{4.57}$$

定义了最优参数向量。

学习过程的瞬时梯度向量定义为：

$$\mathbf{g}(\mathbf{x}(n),\mathbf{d}(n);\mathbf{w}) = \frac{\partial}{\partial\mathbf{w}}\mathscr{E}(\mathbf{x}(n),\mathbf{d}(n);\mathbf{w}) = -(\mathbf{d}(n)-\mathbf{F}(\mathbf{x}(n);\mathbf{w})\mathbf{F}'(\mathbf{x}(n);\mathbf{w}) \tag{4.58}$$

其中

$$\mathbf{F}'(\mathbf{x};\mathbf{w}) = \frac{\partial}{\partial\mathbf{w}}\mathbf{F}(\mathbf{x};\mathbf{w}) \tag{4.59}$$

有了刚刚定义的梯度向量，现在可以将在线学习算法表示为：

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) - \eta(n)\mathbf{g}(\mathbf{x}(n+1),\mathbf{d}(n+1);\hat{\mathbf{w}}(n)) \tag{4.60}$$

或者等价地，

$$\underbrace{\hat{\mathbf{w}}(n+1)}_{\text{更新估计}} = \underbrace{\hat{\mathbf{w}}(n)}_{\substack{\text{老的}\\\text{估计}}} + \underbrace{\eta(n)}_{\substack{\text{学习率}\\\text{参数}}} \underbrace{[\mathbf{d}(n+1) - \mathbf{F}(\mathbf{x}(n+1);\hat{\mathbf{w}}(n))]}_{\text{误差信号}} \underbrace{\mathbf{F}'(\mathbf{x}(n+1);\hat{\mathbf{w}}(n))}_{\text{网络函数}\mathbf{F}\text{的偏导数}} \tag{4.61}$$

有了这个差分方程，我们可以继续通过如下的连续微分方程来描述权值向量 $\mathbf{w}$ 在最优参数 $\mathbf{w}^*$ 的邻域的总体-平均动力学：

$$\frac{\mathrm{d}}{\mathrm{d}t}\hat{\mathbf{w}}(t) = -\eta(t)\,\mathbb{E}_{\mathbf{x},\mathbf{d}}[\mathbf{g}(\mathbf{x}(t),\mathbf{d}(t);\hat{\mathbf{w}}(t))] \tag{4.62}$$

其中 t 表示连续时间。根据 Murata(1998)，梯度向量的期望值通过下式来近似：

$$\mathbb{E}_{\mathbf{x},\mathbf{d}}[\mathbf{g}(\mathbf{x},\mathbf{d};\hat{\mathbf{w}}(t))] \approx -\mathbf{K}^*(\mathbf{w}^* - \hat{\mathbf{w}}(t)) \tag{4.63}$$

其中总体平均矩阵 $\mathbf{K}^*$ 定义为：

$$\mathbf{K}^* = \mathbb{E}_{\mathbf{x},\mathbf{d}}\left[\frac{\partial}{\partial \mathbf{w}}\mathbf{g}(\mathbf{x},\mathbf{d};\mathbf{w})\right] = \mathbb{E}_{\mathbf{x},\mathbf{d}}\left[\frac{\partial^2}{\partial \mathbf{w}^2}\mathscr{E}(\mathbf{x},\mathbf{d};\mathbf{w})\right] \tag{4.64}$$

新的 Hessian 矩阵 $\mathbf{K}^*$ 是一个正定矩阵，和式(4.54)定义的 Hessian 矩阵 $\mathbf{H}$ 是不同的。然而，如果产生训练样本 $\{\mathbf{x},\mathbf{d}\}$ 的环境是遍历的，则可以用基于时间平均的 Hessian 矩阵 $\mathbf{H}$ 来替代基于总体平均的 Hessian 矩阵 $\mathbf{K}^*$。在任何情况下，将式(4.63)代入式(4.62)，我们发现描述估计 $\hat{\mathbf{w}}(t)$ 演化的连续微分方程可以通过下式逼近：

$$\frac{\mathrm{d}}{\mathrm{d}t}\hat{\mathbf{w}}(t) \approx -\eta(t)\mathbf{K}^*(\mathbf{w}^* - \hat{\mathbf{w}}(t)) \tag{4.65}$$

令向量 $\mathbf{q}$ 表示 $\mathbf{K}^*$ 矩阵的特征向量，如下面的定义公式所示：

$$\mathbf{K}^*\mathbf{q} = \lambda\mathbf{q} \tag{4.66}$$

其中 λ 是对应于特征向量 $\mathbf{q}$ 的特征值。则可以引入新的函数

$$\xi(t) = \mathbb{E}_{\mathbf{x},\mathbf{d}}[\mathbf{q}^T\mathbf{g}(\mathbf{x},\mathbf{d};\hat{\mathbf{w}}(t))] \tag{4.67}$$

由式(4.63)，这可以近似表示为

$$\xi(t) \approx -\mathbf{q}^T\mathbf{K}^*(\mathbf{w}^* - \hat{\mathbf{w}}(t)) = -\lambda\mathbf{q}^T(\mathbf{w}^* - \hat{\mathbf{w}}(t)) \tag{4.68}$$

在每一个瞬时 t，函数 $\xi(t)$ 为一个标量值，这可以看成是两个于特征向量 $\mathbf{q}$ 上的投影之间的欧几里得距离的近似测量，一个是最优参数 $\mathbf{w}^*$，另一个是估计 $\hat{\mathbf{w}}(t)$。当估计 $\hat{\mathbf{w}}(t)$ 收敛到 $\mathbf{w}^*$ 时，$\xi(t)$ 的值减为 0。

由式(4.65)、式(4.66)和式(4.68)，我们发现函数 $\eta(t)$ 与随时间变化的学习率参数 η 有关：

$$\frac{\mathrm{d}}{\mathrm{d}t}\xi(t) = -\lambda\eta(t)\xi(t) \tag{4.69}$$

解该微分方程产生：

$$\xi(t) = c\exp\left(-\lambda\int\eta(t)\,dt\right) \tag{4.70}$$

其中 c 是正的积分常数。

根据 Darken and Moody(1991) 的退火方案，这已经在第 3 章中关于 LMS 算法时讨论过了，令公式

$$\eta(t) = \frac{\tau}{t+\tau}\eta_0 \tag{4.71}$$

说明学习率对时间 t 的依赖性，其中 τ 和 η_0 为正的调谐参数。然后，将这一公式代入式(4.70)，我们发现相应的 $\xi(t)$ 函数为：

$$\xi(t) = c(t+\tau)^{-\lambda\tau\eta_0} \tag{4.72}$$

为了使当时间 t 趋于无穷时 $\xi(t)$ 成为 0，指数部分的乘积项 $\lambda\tau\eta_0$ 必须大于 1，这可以通过对正的 α 令 $\eta_0=\alpha/\lambda$ 来满足。

现在，仅剩的问题是如何选取特征向量 $\mathbf{q}$。前一节讲过，学习曲线的收敛速度由 Hessian 矩阵 $\mathbf{H}$ 的最小特征值 λ_{min} 支配。由于 Hessian 矩阵和新的 Hessian 矩阵 $\mathbf{H}^*$ 倾向于相似的行为，一个聪明的选择是假设对于充分大的迭代数，估计器 $\hat{\mathbf{w}}(t)$ 关于时间 t 的演化可以考虑为一维过程，对于和最小特征值 λ_{min} 相关联的 Hessian 矩阵 $\mathbf{K}^*$ “几乎平行”地运行，如图 4.15 所示。因此可以令：

$$\mathbf{q}=\frac{\mathbb{E}_{\mathbf{x},\mathbf{d}}[\mathbf{g}(\mathbf{x},\mathbf{d};\hat{\mathbf{w}})]}{\|\mathbb{E}_{\mathbf{x},\mathbf{d}}[\mathbf{g}(\mathbf{x},\mathbf{d};\hat{\mathbf{w}})]\|} \tag{4.73}$$

其中引入了正规化来假设特征向量 $\mathbf{q}$ 为单位欧几里得长度。相应地，式(4.67)的运用产生了

$$\xi(t)=\|\mathbb{E}_{\mathbf{x},\mathbf{d}}[\mathbf{g}(\mathbf{x},\mathbf{d};\hat{\mathbf{w}}(t))]\| \tag{4.74}$$

现在可以把本节讨论过的结果总结如下：

1. 由式(4.71)所描述的退火方案的选择满足两个条件：

$$\sum_t\eta(t)\to\infty \quad 和 \quad \sum_t\eta^2(t)>\infty, \text{当 } t\to\infty \tag{4.75}$$

换句话说，$\eta(t)$ 满足随机逼近理论（Robbins and Monro，1951）的需要。

2. 在时间 t 趋于无穷时，函数 $\xi(t)$ 渐进地趋于 0。相应于式(4.68)，紧接着有 t 趋于无穷时估计器 $\hat{\mathbf{w}}(t)$ 趋于最优估计 $\mathbf{w}^*$。

3. 在足够大量的迭代次数之后，估计器 $\hat{\mathbf{w}}(t)$ 的总体平均轨线几乎平行于和最小特征值 λ_{min} 相关联的 Hessian 矩阵 $\mathbf{K}^*$ 的特征向量。

4. 由权重向量 w 刻画的网络的最优退火在线学习算法可以通过下面的三个公式来共同描述

$$\left.\begin{aligned}
&\underbrace{\hat{\mathbf{w}}(n+1)}_{\text{更新估计}}=\underbrace{\hat{\mathbf{w}}(n)}_{\substack{\text{老的}\\\text{估计}}}+\underbrace{\eta(n)}_{\substack{\text{学习率}\\\text{参数}}}\underbrace{(\mathbf{d}(n+1)-\mathbf{F}(\mathbf{x}(n)+1;\hat{\mathbf{w}}(n))}_{\text{误差信号}}\underbrace{\mathbf{F}'(\mathbf{x}(n+1);\hat{\mathbf{w}}(n))}_{\text{网络函数}\mathbf{F}\text{的偏导数}}\\
&\eta(n)=\frac{n_{\text{switch}}}{n+n_{\text{switch}}\eta_0}\\
&\eta_0=\frac{\alpha}{\lambda_{\min}},\quad \alpha=\text{正常数}
\end{aligned}\right\} \tag{4.76}$$

这里，假设产生训练样本 $\{\mathbf{x},\mathbf{d}\}$ 的相应的环境是遍历的，因此假设总体平均 Hessian 矩阵 $\mathbf{K}^*$ 和时间平均 Hessian 矩阵 $\mathbf{H}$ 同样的值。

5. 当基于随机梯度下降的在线学习中的学习率参数 η_0 固定时，算法的稳定性需要我们选择 $\eta_0<1/\lambda_{max}$，其中 λ_{max} 是 Hessian 矩阵 $\mathbf{H}$ 的最大特征值，在最优退火随机梯度下降的情形下，相应于式(4.76)的第三行，选择是 $\eta_0<1/\lambda_{min}$，其中 λ_{min} 是 $\mathbf{H}$ 的最小特征值。

6. 时间常数 n_{switch} 是一个正整数，定义了从固定的 η_0 状态转换为退火状态，其中时间变化学习率参数 $\eta(n)$ 假设为期望形式 c/n，其中 c 是常数，对应于随机逼近理论。

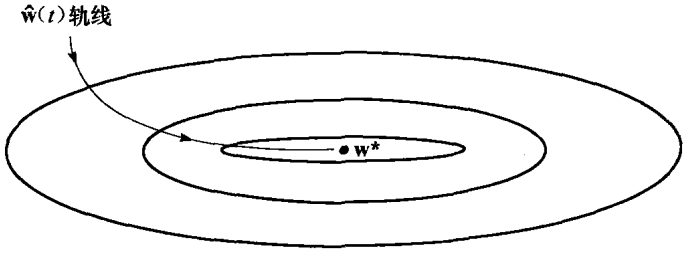


图 4.15　估计 $\hat{\mathbf{w}}(t)$ 在时间 t 上的演化。椭圆表示对于 $\mathbf{w}$ 的变化值的期望风险的轮廓，假设为二维的

学习率的自适应控制

式(4.76)的第二行所描述的最优退火方案提供了在线学习，为促进在线学习的应用迈出了重要一步。然而，这一退火方案的实际局限在于需要知道的时间常量 η_{switch} 为先验的。考虑到实际问题，事实上，当在不稳定的环境中建立在线学习感兴趣的应用时，训练序列的统计性质从一个样例到下一个样例会发生改变，利用一个预先给定的时间常量 n_{switch} 可能不再是一个现实的选择。这一类情形在实际中经常发生，因而在线学习算法需要装备内在机制用于学习率的自适应控制。这样的机制在文献中由 Murata(1998) 首次提出，那里对称为*学习算法的学习*(Sompolinsky 等，1995) 作了适当修正。

Murata 的自适应算法被配置来达到两个目的：

1. *自动调整学习率*，用来处理产生训练序列样例的环境的统计特性有变化的情形。
2. *在线学习算法的泛化*，通过避免预定义代价函数的需要使其适用性更广。

具体来说，由式(4.62)定义的权值向量 $\mathbf{w}$ 的总体-平均动力学，现在可以写为[6]

$$\frac{\mathrm{d}}{\mathrm{d}t}\hat{\mathbf{w}}(t) = -\eta(t)\,\mathbb{E}_{\mathbf{x},\mathbf{d}}[\mathbf{f}(\mathbf{x}(t),\mathbf{d}(t);\hat{\mathbf{w}}(t))] \tag{4.77}$$

这里向量值函数 $f(\cdot,\cdot;\cdot)$ 定义了决定作用于估计 $\hat{\mathbf{w}}(t)$ 相应于进入样例 $\{\mathbf{x}(t),\mathbf{d}(t)\}$ 上的变化的流程（flow)。流程 $\mathbf{f}$ 需要满足条件

$$\mathbb{E}_{\mathbf{x},\mathbf{d}}[\mathbf{f}(\mathbf{x},\mathbf{d};\mathbf{w}^*)] = \mathbf{0} \tag{4.78}$$

其中 $\mathbf{w}^*$ 是权值向量 $\mathbf{w}$ 的最优值，如前面式(4.57)所定义。换句话说，流程 $\mathbf{f}$ 必须渐进地收敛于通过时间 $\mathbf{t}$ 的最优参数 $\mathbf{w}^*$。而且，在稳定性方面，我们也需要 $\mathbf{f}$ 的梯度为正定矩阵。流程 $\mathbf{f}$ 包含了式(4.62)的梯度向量 $\mathbf{g}$ 作为一个特例。

前面从式(4.63)到式(4.69)所定义的公式可以很好地等价应用于 Murata 的算法中。然而在此之后，所做的假设是学习率 $\eta(t)$ 通过时间 t 的演化由如下的一对微分方程构成的动力系统所决定：

$$\frac{\mathrm{d}}{\mathrm{d}t}\xi(t) = -\lambda\eta(t)\xi(t) \tag{4.79}$$

和

$$\frac{\mathrm{d}}{\mathrm{d}t}\eta(t) = \alpha\eta(t)(\beta\xi(t) - \eta(t)) \tag{4.80}$$

这里需要注意的是，$\xi(t)$ 总是正的，α 和 β 是正的常数。这一动态系统的第一个方程是式(4.69)的重复。系统的第二个方程是受相应的微分方程启发得到的，该微分方程位于对 Sompolinsky 等 (1995)[7] 所描述的学习算法的学习中。

如前所述，式(4.79)中的 λ 是相应于 Hessian 矩阵 $\mathbf{K}^*$ 的特征向量 $\mathbf{q}$ 的特征值。而且，假设 $\mathbf{q}$ 被选择为对应于最小特征值 $\lambda_{\min}$ 的特定的特征向量。这就意味着总体-平均流程 $\mathbf{f}$ 以和前面如图 4.15 所描述的相似的方式收敛于最优参数 $\mathbf{w}^*$。式(4.79)和式(4.80)所描述的动态系统的渐进行为通过相应的方程对给出：

$$\xi(t) = \frac{1}{\beta}\left(\frac{1}{\lambda} - \frac{1}{\alpha}\right)\frac{1}{t}, \quad \alpha > \lambda \tag{4.81}$$

和

$$\eta(t) = \frac{c}{t}, \quad c = \lambda^{-1} \tag{4.82}$$

这里需要注意的要点是这一新的动态系统展示了学习率 $\eta(t)$ 的期望退火，即当 t 很大时的 c/t 的值，这对于任意收敛于 $\mathbf{w}^*$ 的估计 $\hat{\mathbf{w}}(t)$ 是最优的，如前所讨论的那样。

根据上面的讨论，现在可以正式地描述离散时间下在线学习的 Murata 自适应算法如下

(Murata，1998；Muller 等，1998)：

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) - \eta(n)\mathbf{f}(\mathbf{x}(n+1),\mathbf{d}(n+1);\hat{\mathbf{w}}(n)) \tag{4.83}$$

$$\mathbf{r}(n+1) = \mathbf{r}(n) + \delta\mathbf{f}(\mathbf{x}(n+1),\mathbf{d}(n+1);\hat{\mathbf{w}}(n)), \quad 0 < \delta < 1 \tag{4.84}$$

$$\eta(n+1) = \eta(n) + \alpha\eta(n)(\beta \| \mathbf{r}(n+1) \| - \eta(n)) \tag{4.85}$$

以下是这一离散时间系统方程中值得注意的点：

- 式(4.83)是简单的式(4.77)的微分方程的瞬时离散时间版本。
- 式(4.84)包含了辅助向量 $\mathbf{r}(n)$，这是被引入来说明连续时间函数 $\xi_{\chi(t)}$。而且，Murata 自适应算法的第二个方程包含一个漏损因子 (leakage factor)，其值 δ 控制了流程 $\mathbf{f}$ 的流动平均。
- 式(4.85)是微分方程(4.80)的离散时间版本。式(4.85)中更新的辅助向量 $\mathbf{r}(n+1)$ 将它和式(4.84)联系起来；在这样做的过程中，允许将式(4.79)和式(4.80)分别定义的连续时间函数 $\xi(t)$ 和 $\eta(t)$ 结合起来。

与式(4.79)和式(4.80)描述的连续时间动力系统不同，式(4.85)的学习率参数 $\eta(t)$ 的渐进行为在迭代次数 n 趋于无穷时不收敛于 0，因此违反了最优退火的需要。相应地，在最优退火参数 $\mathbf{w}^*$ 的邻域中，我们发现对于 Murata 自适应算法有：

$$\lim_{n\to\infty} \hat{\mathbf{w}}(n) \neq \mathbf{w}^* \tag{4.86}$$

这一渐进行为和式(4.76)的最优退火在线学习算法是不同的。基本上，对于最优退火的背离是归因于式(4.77)中流程的流出平均的应用，包含了这一应用是由于需要处理算法无法预先定义代价函数的情形，正如导出最优退火在线学习算法式(4.76)的情形。

当最优解 $\hat{\mathbf{w}}^*$ 随时间 n 缓慢变化时（即产生样例的环境是不稳定的）或者突然改变时，学习规则的学习是有用的。另一方面，$1/n$ 规则在这样的环境下不是一个好的选择，因为 η_n 对于很大的 n 来说变得很小，导致 $1/n$ 规则失去其学习能力。基本上，式(4.76)的最优退火在线学习算法和式(4.83)到式(4.85)的在线学习算法之间的不同是，后者有一个内在的机制用于自适应地控制学习率——因而它能够追踪最优解 $\hat{\mathbf{w}}^*$ 的变化。

最后的评论是：尽管 Murata 自适应算法在所考虑的学习率参数的退火范围内实际上是次优的，其重要的优点在于扩大了在线学习在实际执行方式上的适用性。

4.11　泛化

在反向传播学习中，我们一般从一个训练样本开始，而且通过向网络中装载（编码）尽可能多的训练样本来使用反向传播算法计算一个多层感知器的突触权值。希望这样设计的神经网络可以很好地泛化（推广）。对于从未在生成或训练网络时使用过的测试数据，若网络计算的输入-输出映射对它们来说是正确（或接近于正确）的，我们就认为网络的泛化是很好的；术语“泛化”是从心理学中借用来的。这里假定测试数据是从用于生成训练数据的相同数据集抽取出来的。

学习过程（即神经网络的训练）可以看作是一个“曲线拟合”的问题。网络本身可以被简单地认为是一个非线性输入-输出映射。这个观点允许我们不再把神经网络的泛化看作是它的一个神秘的特性，而是作为相当简单的关于输入数据非线性插值的结果。这种网络之所以能够完成有意义的插值过程，主要是因为具有连续激活函数的多层感知器导致输出函数同样也是连续的。

图 4.16a 表明一个假定的网络是如何进行泛化的。图中描绘的曲线所代表的非线性输入/输出映射是由网络通过对标有“训练数据”的点进行学习的结果来计算的。曲线上标有“泛化”的点就是由这个网络完成的插值结果。

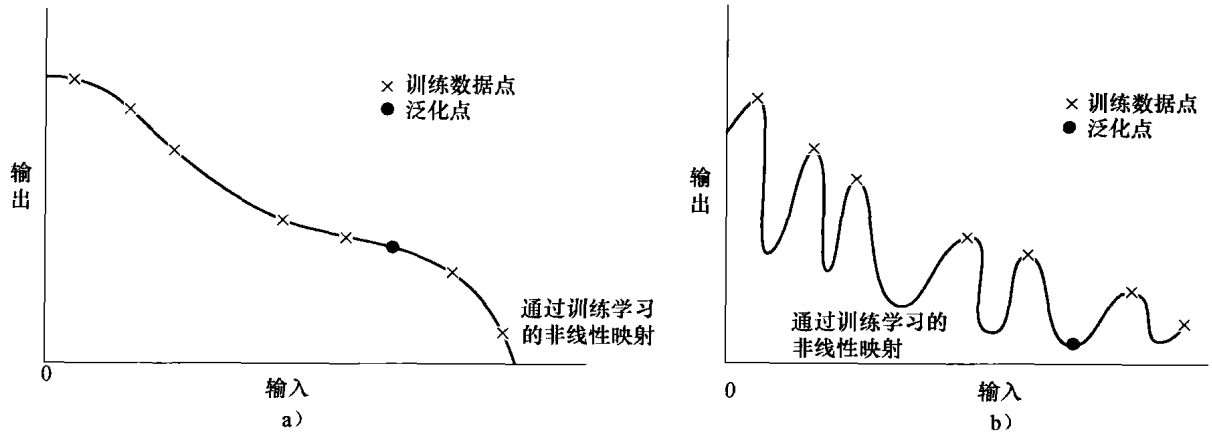


图 4.16　a）良好泛化的恰当拟合非线性映射；b）差的泛化的过拟合非线性映射

一个具有良好泛化能力的神经网络，即使当输入数据与训练样本稍有不同时，也能产生正确的输入-输出映射，如图中所示。然而，当神经网络学习太多的输入-输出样本时，它可能会完成训练数据的记忆。这可能在以下情况中出现，找到一个存在于训练数据中的特征（如由于噪声），但对于将要建模的固有函数却为假。这种现象称为“过拟合”或者“过训练”。当网络被过训练的时候，它就失去了在相近输入/输出模式之间进行泛化的能力。

通常，用这种方法把数据装载到多层感知器要求使用比实际需要更多的隐藏层神经元，结果导致在网络的突触权值中存储了输入空间中由于噪声引起的非期望因素。例如，在图 4.16a 相同的数据条件下，图 4.16b 显示由于神经网络中的记忆导致泛化不佳是如何出现的例子。“记忆”本质上是一个“查询表”，这意味着由神经网络计算的输入/输出映射是非光滑的。正如在 Poggio and Girosi(1990a) 文章中指出的那样，输入/输出映射的光滑性与如 Occam 剃刀之类的模型选择标准紧密相关，在没有相反的先验知识情况下，它的核心本质是选择“最简单”函数。针对于我们给出的讨论，最简单函数是指在给定的误差标准下逼近一个给定映射的函数中最光滑的函数，因为这个选择总体上要求最少的计算资源。依赖于研究现象的规模范围，光滑性在许多应用上同样是自然的。因而为不恰当的输入/输出关系寻找一个光滑的非线性映射是重要的，使得网络能够根据训练模式将新模式正确地分类（Wieland and Leighton，1987）。

为有效的泛化给出充分的训练集大小

下面的三个因素对泛化产生影响：（1）训练集的大小，以及它如何表示感兴趣的环境；（2）神经网络的体系结构；（3）当前问题的物理复杂度。无疑，我们无法对后者进行控制。在另外的两个因素中，我们可以从两个不同的方面考察泛化问题：

- 网络的体系结构是固定的（可期望与固有问题的物理复杂度一致），需要解决的问题是决定一个产生好的泛化必需的训练集的大小。
- 训练集的大小是固定的，感兴趣的问题是决定最好的网络体系结构使得具有好的泛化。

在它们各自的方法里这两种观点都是合理的。

在实践中，看起来对一个好的泛化而言，事实上我们所需要的全部是训练集的大小 N 满足条件

$$N = O\left(\frac{W}{\varepsilon}\right) \tag{4.87}$$

这里 W 是指网络中自由参数（即突触权值和偏置）的总数，ε 表示测试数据中容许分类误差的

部分（正如在模式分类中一样）。$O(\cdot)$表示所包含的量的阶数。例如，具有 10%误差的所需训练样本数量应该是网络中自由参数数量的 10 倍。

式(4.85)与用于 LMS 算法的 Widrow 经验方法是一致的，后者指出线性自适应时间滤波的适应迟滞时间，近似等于自适应抽头延迟线滤波器的记忆范围除以误调节得到的商（Widrow and Stearns，1985；Haykin，2002）。LMS 算法中的误调节扮演的角色与式(4.87)中的误差 ε 有某些相似。这个经验规则的进一步理由将在下一节中介绍。

4.12　函数逼近

一个由反向传播算法训练的多层感知器可以被看作一个实现一般性质的非线性输入-输出映射的实际工具。具体地讲，令 m_0 表示多层感知器的输入（源）节点的数目，令 $M=m_L$ 表示网络中输出层神经元的数目。网络的输入-输出关系定义一个从 m_0 维欧几里得输入空间到 M 维欧几里得输出空间的映射，当激活函数是无限连续可微的时候，这个映射也是无限连续可微的。在用这种输入-输出映射观点来评价多层感知器能力的过程中，提出了下面基本的问题：

一个多层感知器的输入-输出映射能够提供任何一个连续映射的近似实现，它的隐藏层层数的最小数目是多少？

通用逼近定理

这个问题可以用一个非线性输入-输出映射的通用逼近定理[8]来回答，该定理如下：

令 $\varphi(\cdot)$是一个非常数的、有界的和单调增的连续函数。令 I_{m_0} 表示 m_0 维单位超立方体 $[0,1]^{m_0}$。I_{m_0} 上连续函数空间用 $C(I_{m_0})$表示。那么，给定任何函数 $f\ni C(I_{m_0})$和 $\varepsilon>0$，存在这样的一个整数 m_1 和实常数 α_i，b_i 和 w_{ij}，其中 $i=1,\cdots,m_1$，$j=1,\cdots,m_0$，使我们可以定义

$$F(x_1,\cdots,x_{m_0})=\sum_{i=1}^{m_1}\alpha_i\varphi\Big(\sum_{j=1}^{m_0}w_{ij}x_j+b_i\Big) \tag{4.88}$$

作为 $f(\cdot)$函数的一个近似实现；也就是说，

$$|F(x_1,\cdots,x_{m_0})-f(x_1,\cdots,x_{m_0})|<\varepsilon$$

对存在于输入空间中的所有 $x_1,x_2,\cdots,x_{m_0}$ 均成立。

通用逼近定理可直接用于多层感知器。我们首先注意到在一个作为多层感知器结构的神经元模型中作为非线性部分的双曲正切函数是一个真正非常数的、有界的和单调递增的函数；因此它满足函数 $\varphi(\cdot)$的上述条件。下一步，注意式(4.88)表达如下所述的多层感知器的输出：

1. 网络具有 m_0 个输入节点和单个由 m_1 个神经元组成的隐藏层；输入由 $x_1,\cdots,x_{m_0}$ 表示。
2. 隐藏神经元 i 具有突触权值 $w_{i_1},\cdots,w_{m_0}$和偏置 b_i。
3. 网络的输出是隐藏层的线性组合，带有定义输出层突触权值的 $\alpha_1,\cdots,\alpha_{m_1}$。

通用逼近定理是存在性定理，它与精确表示相反，为任意连续函数的逼近提供数学上的基础。作为定理的本质，式(4.88)仅仅是推广有限 Fourier 级数逼近。事实上，这个定理说明，对于多层感知器计算一个由输入 $x_1,\cdots,x_{m_0}$和期望（目标）输出 $f(x_1,\cdots,x_{m_0})$表示的给定训练集的一致 ε 逼近来说，单个隐藏层是足够的。然而，此定理并没有说明单个隐藏层在学习时间、实现的难易程度或者（更重要的）泛化意义上是最优的。

逼近误差的界

假定网络使用 sigmoid 函数的单层隐藏神经元和线性输出神经元，Barron(1993) 建立了多层感知器的逼近性质。网络通过使用反向传播算法训练，然后用新的数据测试。在训练过程

中，网络根据训练数据学习目标函数 f 中的特殊点，从而产生由式(4.88)中定义的逼近函数 F。当网络遇到以前没有见过的测试数据的时候，网络函数 F 就充当目标函数中新的点的估计器；即 $F=\hat{f}$。

目标函数的光滑度属性用它的 Fourier(变换) 来表达。特别地，用 Fourier 幅度分布加权后的频率向量的范数的平均值作为函数 f 振荡的度量标准。令 $\tilde{f}(\boldsymbol{\omega})$表示函数 $f(\mathbf{x})$的多维 Fourier 变换，$\mathbf{x}\in\mathbb{R}^{m_0}$：$m_0\times 1$ 向量 $\boldsymbol{\omega}$ 为频率向量。函数 $f(x)$由关于它的 Fourier 变换函数 $\tilde{f}(\boldsymbol{\omega})$ 的反变换公式定义如下：

$$f(x)=\int_{\mathbb{R}^{m_0}}\tilde{f}(\boldsymbol{\omega})\exp(j\boldsymbol{\omega}^{\mathrm{T}}\mathbf{x})d\boldsymbol{\omega} \tag{4.89}$$

这里 $j=\sqrt{-1}$。对于复值函数 $\tilde{f}(\boldsymbol{\omega})$，由于 $\boldsymbol{\omega}\tilde{f}(\boldsymbol{\omega})$ 是可积的，我们定义函数 f 的 Fourier 幅度分布的一阶绝对动量如下：

$$C_f=\int_{\mathbb{R}^{m_0}}|\tilde{f}(\boldsymbol{\omega})|\times\|\boldsymbol{\omega}\|^{1/2}d\boldsymbol{\omega} \tag{4.90}$$

其中，$\|\boldsymbol{\omega}\|$ 为 $\boldsymbol{\omega}$ 的欧几里得范数，$|\tilde{f}(\boldsymbol{\omega})|$ 为 $\tilde{f}(\boldsymbol{\omega})$ 的绝对值。一阶绝对动量 C_f 量化函数 f 的光滑度。

一阶绝对动量 C_f 为使用以式(4.88)中输入-输出映射函数 $F(\mathbf{x})$为表示的多层感知器来逼近 $f(\mathbf{x})$而导致的误差范围的界提供了基础。逼近误差可以用与一个半径 $r>0$ 的球体 $B_r=\{\mathbf{x}:\|\mathbf{x}\|\leqslant r\}$ 中任意可能的概率测度 μ 相关的积分平方误差来衡量。在这个基础上我们可以对 Barron(1993) 提出的逼近误差范围的界提出如下命题：

对于每个具有有限一阶绝对动量 C_f 的连续函数 $f(\mathbf{x})$，以及每个 $m_1\geqslant 1$，存在一个由式(4.88)定义的 sigmoid 函数的线性组合 $F(\mathbf{x})$，使得当在严格属于球体内部的输入向量 $\mathbf{x}$ 的值集合 $\{\mathbf{x}_i\}_{i=1}^N$ 上观察函数 $f(\mathbf{x})$的时候，命题的结果对经验风险提供如下的界：

$$\mathscr{E}_{\mathrm{av}}(N)=\frac{1}{N}\sum_{i=1}^{N}(f(\mathbf{x}_i)-F(\mathbf{x}_i))^2\leqslant\frac{C_f'}{m_1} \tag{4.91}$$

其中 $C_f'=(2rC_f)^2$。

在 Barron(1992) 中，利用式(4.91)的逼近结果表示使用具有 m_0 个输入节点和 m_1 个隐藏神经元的多层感知器而导致的风险$\mathscr{E}_{\mathrm{av}}(N)$的界如下：

$$\mathscr{E}_{\mathrm{av}}(N)\leqslant O\left(\frac{C_f^2}{m_1}\right)+O\left(\frac{m_0m_1}{N}\log N\right) \tag{4.92}$$

风险$\mathscr{E}_{\mathrm{av}}(N)$的界中的两项表达了对隐藏层大小的两种矛盾要求的权衡：

1. *最佳逼近的精确度。*为了满足这个要求，根据通用逼近定理隐藏层的大小 m_1 必须足够大；

2. *逼近的经验拟合精确度。*为了满足第二个要求，必须使用一个小的比值 m_1/N。对于训练集的固定的大小 N，隐藏层的大小 m_1 应该保持较小，这跟第一个要求是矛盾的。

式(4.92)描述的风险$\mathscr{E}_{\mathrm{av}}(N)$的界具有另外一个有趣的含义。特别地，我们看到假如一阶绝对动量 C_f 仍是有限的话，相对于输入空间维数 m_0，一个指数规模的大样本集对于得到一个目标函数精确的估算并不是必须的。这个结果使得多层感知器作为通用逼近器在实际条件下甚至显得更重要。

经验拟合和最佳逼近之间的误差可以看作是估计误差。令 ε_0 表示*估计误差*的均方值。然后忽略式(4.92)中表达式的第二项的对数因子 $\log N$，我们可以推断出一个好的泛化所需的训练集大小 N 大约是 m_0m_1/ε_0。这个结果具有与经验公式(4.87)相似的数学结构，记住 m_0m_1 等

于网络中自由参数 W 的总数。换句话说，我们可以从总体上说为了得到好的泛化，训练样本的数目 N 应该大于网络中自由参数总数和估计误差均方值之比。

维数灾难

出现在式(4.92)所描述的界中另一个有趣的结果是，当对隐藏层的大小通过设定

$$m_1 \simeq C_f\left(\frac{N}{m_0 \log N}\right)^{1/2}$$

进行优化（也就是风险$\mathscr{E}_{av}(N)$关于 N 最小化）的时候，风险$\mathscr{E}_{av}(N)$由 $O(C_f\sqrt{m_0(\log N/N)})$ 限定。这个结果的一个令人惊奇的方面是根据风险$\mathscr{E}_{av}(N)$的一阶行为，以训练集大小 N 的函数表达的收敛速率的阶为 $(1/N)^{1/2}$(乘以一个对数因子)。另一方面，对传统的光滑函数（例如多项式和三角函数）我们有不同的行为。令 s 表示光滑度的一种度量，定义为函数具有连续导数的阶数。那么，对于传统光滑函数，我们发现总风险$\mathscr{E}_{av}(N)$的极小极大的收敛速率的阶为 $(1/N)^{2s/(2s+m_0)}$。这个收敛速率对输入空间维数 m_0 的依赖就是维数灾难，这严重地制约了这些函数的实际应用。使用多层感知器进行函数逼近看起来提供超越于传统光滑函数的优势；但是，这个优势受限于一阶绝对动量 C_f 保持有限的条件；这是一个光滑度约束。

Richard Bellman 在对自适应控制过程（Bellman，1961）的研究中介绍了维数灾难。为了从几何上解释这个概念，令 $\mathbf{x}$ 表示一个 m_0 维的输入向量，$\{(\mathbf{x}_i,d_i)\}$，$i=1,2,\cdots,N$ 表示训练样本。采样密度与 N^{1/m_0} 成正比。令函数 $f(\mathbf{x})$代表一个位于 m_0 维输入空间的曲面，它近似通过点 $\{(\mathbf{x}_i,d_i)\}_{i=1}^{N}$。现在，如果函数 $f(\mathbf{x})$是任意复杂并且（对绝大部分）是完全未知的，我们需要密集的样本（数据）来进行很好的学习。不幸的是，密集样本在“高维”中是很难找到的，因此产生了维数灾难。特别地，维数增加的结果导致复杂度呈指数增长，从而引起高维空间中一致随机分布点的空间填充性质退化。维数灾难的基本原因如下（Friedman，1995）：

定义在高维空间的函数很可能远远比定义在低维空间上的函数复杂得多，并且这些复杂的东西更难以区分。

基本上，仅有两个途径可以减轻维数灾难问题：

1. 结合关于要逼近的未知函数的一些*先验知识*。这些先验知识是在训练数据之上提供的。自然，这些知识的获得是依赖于问题的。例如在模式分类中可以通过理解输入数据的相关的类(种类）来获得知识。

2. 设计网络使之随着输入维数的增加而增加未知函数的*光滑度*。

可行性考虑

从理论的角度来看，通用逼近定理是重要的，因为对具有单个隐藏层的前馈神经网络作为一类逼近器的可行性，该定理提供了*必要的数学工具*。如果没有这样一个理论，我们可能在盲目寻找那些并不存在的方法。然而，这个理论并不是构造性的，即它实际上并不能具体实现如何由陈述的逼近性质决定一个多层感知器。

通用逼近定理假设被逼近的连续函数是给定的并且可用一个神经元数目无限制的隐藏层来逼近。这两个假设在多层感知器的绝大多数实际应用中都是不成立的。

使用单个隐藏层的多层感知器的问题是隐藏层的神经元倾向于全局地相互作用。在复杂情形下这种相互作用使得在一点提高它的逼近的同时又很难不恶化它在另外点上的逼近。另一方面，在具有两个隐藏层的情况下逼近（曲线拟合）过程变得更容易协调。具体地，我们可以进行如下处理（Funahashi，1989；Chester，1990）：

1. 从第一个隐藏层中抽取*局部特征*。特别地，利用在第一个隐藏层中的一些神经元将输

入空间分割成区域，这层中另外的神经元学习表征这些区域特点的局部特征。

2. 从第二个隐藏层中抽取全局特征。特别地，在第二隐藏层中的一个神经元组合在输入空间特定区域操作的第一个隐藏层的各神经元的输出，从而学习该区域的全局特征并且在别处的输出为零。

Sontag(1992) 为在逆问题中两个隐藏层的使用提供了进一步理由。

4.13 交叉验证

反向传播学习的本质是把输入/输出映射（由标定的一组训练样本表示）编码为一个多层感知器的突触权值和阈值。我们希望的是，网络通过良好的训练，使得它充分地学习过去的数据，从而对未来有良好的泛化能力。从这个观点来看，学习过程意味着对给定的数据集合给出网络参数化的一个选择。具体地，我们可以把网络选择问题看作是从一组候选模型结构（参数）集合中选择符合某个标准的“最好”的一个。

在这种意义下，统计学中一个名为交叉验证的标准工具提供了一个有吸引力的指导原则[9] (Stone，1974，1978)。已有的可用数据集首先被随机分割成一个训练集和一个测试集。这个训练集被进一步细分为两个不相交子集：

- 估计子集，用来选择模型。
- 验证子集，用来测试或者验证模型。

这里的动机是用一个与参数估计数据集不同的数据集来验证模型。用这个办法可以用训练集来估计不同候选模型的性能，进而选择“最好”的一个。然而，这样选出的具有最优表现的参数值的模型，很可能会导致对验证子集的过度拟合。为了防止这种情况出现，使用测试集来衡量被选模型的泛化性能，测试集是与验证子集不同的集合。

当我们不得不以设计一个具有好的泛化性能的大型神经网络作为目标的时候，交叉验证的使用是特别吸引人的。例如，我们可以使用交叉验证确定具有最优隐藏神经元数目的多层感知器，以及最好在何时停止它的训练，正如在下面两小节中所述的那样。

模型选择

根据交叉验证选择模型的思想，考虑如下表示的布尔函数类的嵌入结构：

$$\mathcal{F}_1 \subset \mathcal{F}_2 \subset \cdots \subset \mathcal{F}_n \tag{4.93}$$

$$\mathcal{F}_k = \{F_k\} = \{F(\mathbf{x},\mathbf{w});\mathbf{w} \in \mathcal{W}_k\}, \quad k = 1,2,\cdots,n$$

也就是说，第 k 个函数类 $\mathcal{F}_k$ 包含一簇具有相似体系结构的多层感知器，其权值向量 $\mathbf{w}$ 从一个多维权值空间 $\mathcal{W}_k$ 中抽出。以函数或者假设 $F_k = F(\mathbf{x},\mathbf{w})$，$\mathbf{w} \in \mathcal{W}_k$ 为特征的类的一个成员把输入向量 $\mathbf{x}$ 映射到 $\{0,1\}$，这里 $\mathbf{x}$ 是以某未知概率 P 从输入空间 $\mathcal{X}$ 中抽取出来的。在所述结构中每个多层感知器都是由反向传播算法训练的，该算法负责多层感知器参数的训练。模型选择问题本质是选择具有最好的自由参数（即突触权值和阈值）数目 $\mathbf{w}$ 值的多层感知器。更精确地讲，假设对输入向量 $\mathbf{x}$ 的期望响应标量是 $d = \{0, 1\}$，我们定义泛化误差如下：

$$\varepsilon_g(F) = P(F(\mathbf{x}) \neq d) \qquad \text{对 } \mathbf{x} \in \mathcal{X}$$

给出一个标定的训练样本集

$$\mathcal{T} = \{\mathbf{x}_i, d_i\}_{i=1}^{N}$$

我们的目标是选择特定的假设 $F(\mathbf{x},\mathbf{w})$，当从测试集中给定输入时它最小化所得泛化误差 $\varepsilon_g(F)$。

下面假设由式(4.93)表达的结构具有这样的性质，即对于任意大小的 N 都可以找到一个具有数量足够多的自由参数的数目 $W_{\max}(N)$ 的多层感知器，使得训练数据集 $\mathcal{T}$ 可以被合适地拟合。这只不过重申 4.12 节的通用逼近定理。我们把 $W_{\max}(N)$ 称为拟合数。$W_{\max}(N)$ 的意义在于，一个合理的模型选择程序应该选择一个满足 $W \leqslant W_{\max}(N)$ 的假设 $F(\mathbf{x},\mathbf{w})$；否则网络复杂

度将会增加。

令一个位于 0 和 1 之间的参数 r 决定估计子集和验证子集之间的训练数据集$\mathcal{T}$的划分。$\mathcal{T}$由 N 个样本组成，$(1-r)N$ 个样本分配给估计子集，剩下的 rN 个样本分配给验证子集。估计子集用$\mathcal{T}'$表示，它用于训练多层感知器的一个嵌套序列，嵌套结构导致复杂度递增的假设 $\mathcal{F}_1$，$\mathcal{F}_2,\cdots,\mathcal{F}_n$。由于$\mathcal{T}'$由 $(1-r)N$ 个样本组成，我们认为 W 的值小于或者等于相应的拟合数 $W_{\max}((1-r)N)$。

交叉验证方法的使用导致选择

$$\mathcal{F}_{cv} = \min_{k=1,2,\cdots,v} \{e''_t(\mathcal{F}_k)\} \tag{4.94}$$

其中 v 对应于 $W_v \leqslant W_{\max}((1-r)N)$，$e''_t(\mathcal{F}_k)$是在由 rN 个样本组成的验证子集$\mathcal{T}''$上测试时由假设$\mathcal{F}_k$ 产生的分类误差。

关键问题是如何具体确定参数 r 以决定训练集$\mathcal{T}$在估计子集$\mathcal{T}'$和验证子集$\mathcal{T}''$之间的划分。在 Kearns(1996) 描述的研究中，对该论题进行分析处理和具体的计算机仿真支持，确定了最优 r 的几个定性特点：

- 当定义输入向量 $\mathbf{x}$ 的期望响应 d 的目标函数的复杂度相对于样本大小的 N 是很小的时候，交叉验证的性能对 r 的选择相对不灵敏。
- 随着目标函数相对于样本大小 N 变得更复杂，最优 r 的选择在交叉验证性能上具有更重要的影响，并且目标函数自身的值减小。
- r 的一个单一固定的值在目标函数复杂度的一个相当大的范围内保持近乎最佳。

根据 Kearns(1996) 报告的结果，r 等于 0.2 的一个固定值看来是一个合理的选择，这意味着训练集$\mathcal{T}$的 80%被指定为估计子集，剩下的 20%被指定为验证子集[10]。

训练的早期停止方法

通常，用反向传播算法训练的多层感知器分阶段地进行学习，随训练过程的进行实现相当简单的映射函数到更复杂的映射函数。这通过在一个典型情形下在训练中均方误差随着训练回合的增加而减少的例子来证明：均方误差从一个很大的值开始，然后迅速地减小，最后随着网络在误差曲面接近局部最小值的时候缓慢地减小。目标函数的复杂度，依据输入向量 $\mathbf{x}$ 定义需要的响应 d，当该复杂度小于样本大小 N 时，以好的泛化能力为目标，如果我们通过观察自身训练得到的学习曲线，很难断定何时停止训练最好。特别地，根据 4.11 节关于泛化的内容，如果训练未在合适的时间停止，那么网络可能过拟合训练数据。

我们可以通过交叉验证来标记过拟合的发生，为此训练数据被分成估计子集和验证子集。使用样本的估计子集以通常方法训练网络，但有较小的修改：训练时间被周期性地停止（即每一个周期都有许多训练回合），并且在每个训练周期之后都由验证子集测试网络。具体地讲，周期性的估计伴随确认（estimation-followed-by-validation）的过程是如下进行的：

- 经过一个估计（训练）周期之后——例如每五个回合——多层感知器的突触权值和偏置都已经固定，网络是在它的前向方式下运作的。从而对验证子集中的每个样本测定验证误差。
- 当验证阶段完成的时候，估计（训练）重新开始另一个周期，这个过程被重复。

这个过程称作训练的早期停止方法，这是易于理解的从而在实际中广泛使用。

图 4.17 显示了两种学习曲线的概念形式，一个属于估计子集上的测定误差，另一个属于验证子集。通常，模型在验证子集上的表现并不像它在估计子集上的表现那么出色，它的设计是基于估计子集的。估计学习曲线在一般情况下随训练回合数目的增加而单调地减小。与此相对，验证学习曲线单调地递减到一个最小值，然后开始随训练的继续而递增。当仅观察估计学

习曲线的时候，很明显通过越过验证学习曲线上的最小点可以得到它的更小的值。然而实际上，网络在越过该点学习到的主要是包含在训练数据中的噪声。这种启发方法意味着验证学习曲线上的最小点可用于停止训练过程的合理准则。

然而，这里有一点要当心。在实际中，验证样本误差在训练回合数上的演化并不能和图 4.17 所示的理想曲线一样平滑。验证样本误差更可能在随着回合数的增加之前本身呈现少数局部极小点。在这样的情形下，必须在系统方式下选取终止准则。Prechelt(1998)提出的多层感知器中的实验调查从实验上表明，事实上，在训练时间和泛化能力上存在着折中。在 1296 个训练集、12 个不同的问题、24 个不同的网络结构所获得训练结果的基础上，得到的结论是存在两个或更多局部极小点的情形下，“较慢”的停止准则（即一个比其他准则较后停止的准则）的选取在花费更长的训练时间（典型地，大约平均 4 倍）下获得了泛化性能的小的改善（大约平均 4%）。

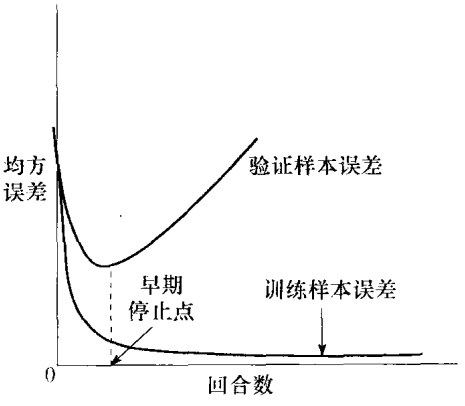


图 4.17　基于交叉验证的早期停止准则示意图

交叉验证的变体

上述交叉验证的方法称为*坚持到底方法*（hold out method）。在实际中还有另外一些能找到它们自身应用的交叉验证的变体，特别是在缺乏标定样本的时候。在这样的情况下可以通过把 N 个样本的可用集合分割为 K 个子集来使用多重交叉验证方法，$K>1$；这里假设 N 对 K 是可除的。这个模型在除了一个子集之外的其他子集上进行训练，验证误差通过剩下子集上的测试来测量。这个过程总共被重复 K 次试验，每次使用一个不同的子集进行验证，如图 4.18 所示 $K=4$ 的情形。模型性能的评估是通过求实验中所有的实验的验证平方误差的平均值来进行的。多重交叉验证存在--个缺点：因为模型必须训练 K 次，它可能需要过多的计算量，这里 $1<K\leqslant N$。

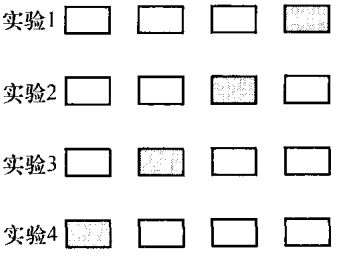


图 4.18　多重交叉验证的示意图。对一给定的实验，带阴影的数据集用来验证模型，而剩下的数据用来训练模型

当可用的标定样本的数目 N 被严格限制的时候，我们可以使用被称为*留一方法*（leave-one-out method）的多重交叉验证的极端形式。在这种方法中，$N-1$ 个样本用来训练模型，并且这个模型通过剩下的一个样本的测试来验证。这个实验总共被重复 N 次，每次留出一个不同的样本来进行验证。然后通过验证的平方误差在 N 次实验上求平均。

4.14　复杂度正则化和网络修剪

无论用何种方式设计一个多层感知器，实际上都是对生成用于训练网络的输入输出样本的物理现象建立一个非线性模型。就网络的设计而论在本质上还是统计的，我们需要在训练数据的可靠性和模型的适应度之间寻找一个适当的平衡（即在第 2 章中解决偏置方差困境的方法）。在反向传播学习的背景下，或者任何其他的监督学习过程而言，我们都可能通过最小化表述如下的总量风险以实现折中：

$$R(\mathbf{w}) = \mathscr{E}_{av}(\mathbf{w}) + \lambda \mathscr{E}_c(\mathbf{w}) \tag{4.95}$$

第一项$\mathscr{E}_{av}(\mathbf{w})$是标准的性能度量，它同时依赖于网络（模型）和输入数据。在反向传播学习中，它被典型地定义为均方误差，该误差的计算扩展到网络输出神经元，并且它在每一回合的

基础上对所有训练样本来完成，参考式(4.5)。第二项$\mathscr{E}_c(\mathbf{w})$是复杂度惩罚，复杂度仅依赖于网络（模型）；它所包含的内容利用我们可能具有的关于所考虑模型的解的先验知识。对于当前的讨论，把λ看作正则化参数就足够了，它代表着关于性能度量项的复杂度惩罚项的相对重要性。当λ为零的时候，反向传播学习过程是无约束的，网络由训练样本完全确定。另一方面，当λ趋于无穷大的时候，这意味着由复杂度惩罚所得到的约束自身就可以具体确定网络，用另一种说法就是训练样本是不可靠的。在复杂度正则化的实际应用中，正则化参数λ被赋予两个极端情形之间的某个位置的值。第 7 章将讨论正则化理论的细节。

权值衰减过程

在一个简单但有效的称为权值衰减过程（Hinton，1989）的复杂度正则化中，复杂度惩罚项被定义为网络中权值向量$\mathbf{w}$(即所有的自由参数）的平方范数，表示为

$$\mathscr{E}_c(\mathbf{w}) = \|\mathbf{w}\|^2 = \sum_{i\in\mathscr{C}_{\text{total}}} w_i^2 \tag{4.96}$$

其中集合$\mathscr{C}_{\text{total}}$是指网络中所有的突触权值。这个过程是通过强迫网络中的一些突触权值取近似于零的值来进行的，而允许其他的权值保持它们相对大的值。所以，网络的权值大致分为两个类：

1）对网络性能具有很大影响的权值。

2）对网络性能具有很少或者根本没有影响的权值。

在后一类中的权值称为多余权值。在不进行复杂度正则化的情况下，这些权值很可能取完全任意的数值，或为了得到训练误差上的轻微减少而促使网络过度拟合训练数据，从而导致很差的泛化性能（Hush and Horne，1993）。复杂度正则化的使用鼓励多余权值取得接近于零的数值，因而提高泛化能力。

基于 Hessian 矩阵的网络修剪：最佳脑外科医生

网络修剪解析方法的基本思想是利用误差曲面的二次导数信息得到网络复杂度和训练误差性能之间的折中方案。特别地，构造误差曲面的一个局部模型，解析地预测突触权值的扰动所造成的影响。构造这样一个模型结构的出发点是在运行点附近使用 Taylor 级数给出代价函数$\mathscr{E}_{\text{av}}$的局部逼近，描述如下：

$$\mathscr{E}_{\text{av}}(\mathbf{w}+\Delta\mathbf{w}) = \mathscr{E}_{\text{av}}(\mathbf{w}) + \mathbf{g}^T(\mathbf{w})\Delta\mathbf{w} + \frac{1}{2}\Delta\mathbf{w}^T\mathbf{H}\Delta\mathbf{w} + O(\|\Delta\mathbf{w}\|^3) \tag{4.97}$$

其中$\Delta\mathbf{w}$是运行点$\mathbf{w}$的扰动，$\mathbf{g}(\mathbf{w})$是在$\mathbf{w}$处的梯度向量。Hessian 矩阵同样在$\mathbf{w}$点进行计算，因而，为了准确起见，我们用$\mathbf{H}(\mathbf{w})$来表示它。在式(4.97)中并没有这么做仅仅是因为简化记号。

要求确认一组参数使得从多层感知器上删除它们而代价函数$\mathscr{E}_{\text{av}}$的值增长最小。为了实际解决这个问题，我们进行如下逼近：

1. 极值逼近。我们假设参数仅在训练过程收敛（即网络被完全训练）之后才被从网络中删除。这个假设的含义就是参数的取值为误差曲面上一个局部最小或者全局最小。在这种情况下，梯度向量$\mathbf{g}$可以设为零因而可以忽略式(4.97)右边的$\mathbf{g}^T\Delta\mathbf{w}$项。否则显著性度量（将在后边定义）将对当前问题无效。

2. 二次逼近。我们假设局部最小或者全局最小周围的误差曲面是近似“二次的”。因此同样可以忽略式(4.97)中的更高次项。

在这两个假设之下，式(4.97)被简单近似为：

$$\Delta\mathscr{E}_{\text{av}} = \mathscr{E}(\mathbf{w}+\Delta\mathbf{w}) - \mathscr{E}(\mathbf{w}) = \frac{1}{2}\Delta\mathbf{w}^T\mathbf{H}\Delta\mathbf{w} \tag{4.98}$$

式(4.98)提供了称为最优脑外科（optimal brain surgeon，OBS）的修剪过程，这是根据 Hassibi and Stork(1993) 而来的。

OBS 的目标是置一个突触权值为零使得式(4.98)中给出的$\mathscr{E}_{av}$的递增增量最小化。令 $w_i(n)$ 表示这个特别的突触权值。这个权值的删除等价于条件：

$$\mathbf{1}_i^T \Delta \mathbf{w} + w_i = 0 \tag{4.99}$$

成立，其中 $\mathbf{1}_i$ 是除了第 i 个元素等于单位 1 之外其他所有元素均为零的单位向量。现在可以重申 OBS 的目标如下：

对权值向量增长变化 $\Delta\mathbf{w}$ 最小化二次型 $\frac{1}{2}\Delta\mathbf{w}^T\mathbf{H}\Delta\mathbf{w}$，使它满足约束条件 $1_i^T\Delta\mathbf{w}+w_i$ 为零，然后关于下标 i 求最小化。

这里进行两个层次上的最小化。一个最小化是当第 i 个权值向量置零后对仍保留的突触权值向量进行的；第二个最小化是对特定被修剪的向量进行的。

为了解决这个约束最优化问题，首先构建一个拉格朗日算子

$$S = \frac{1}{2}\Delta\mathbf{w}^T\mathbf{H}\Delta\mathbf{w} - \lambda(\mathbf{1}_i^T\Delta\mathbf{w} + w_i) \tag{4.100}$$

其中 λ 是拉格朗日乘子。然后求拉格朗日函数 S 对 $\Delta\mathbf{w}$ 的导数，应用式(4.99)的约束条件，并且利用矩阵的逆，我们发现权值向量 $\mathbf{w}$ 中的最佳变化是

$$\Delta\mathbf{w} = -\frac{w_i}{[\mathbf{H}^{-1}]_{i,i}}\mathbf{H}^{-1}\mathbf{1}_i \tag{4.101}$$

拉格朗日算子 S 对元素 $\mathbf{w}_i$ 的相应最优值是

$$S_i = \frac{w_i^2}{2[\mathbf{H}^{-1}]_{i,i}} \tag{4.102}$$

其中 $\mathbf{H}^{-1}$ 是 Hessian 矩阵 $\mathbf{H}$ 的逆，$[\mathbf{H}^{-1}]_{i,i}$ 是这个逆矩阵的第 (i,i) 个元素。假设第 i 个突触权值 w_i 被删除，对 $\Delta\mathbf{w}$ 进行优化而得到的拉格朗日算子 S_i 称为 w_i 的显著性（saliency）。事实上，显著性 S_i 代表由于 w_i 的删除而导致的均方误差（性能标准）中的增长。注意显著性 S_i 是与 w_i^2 成正比的。这样小的权值在均方误差上具有小的影响。然而，从式(4.102)中可以看到显著性 S_i 同样与逆 Hessian 矩阵的对角元素成反比。这样如果 $[\mathbf{H}^{-1}]_{i,i}$ 是小的，那么甚至小的权值也可能对均方误差有实质性的影响。

在 OBS 过程中，相应于最小特征值的权值被选为删除的权值。此外，剩余权值的最佳变化由公式(4.101)给出，这说明它们可以沿逆 Hessian 矩阵的第 i 列方向被校正。

据 Hassibi 等人发表的关于一些基准问题的内容，OBS 过程产生的网络比其他通过权值衰减的过程得到的网络更小。同时报告 OBS 过程应用于包含单个隐藏层和 18 000 个权值的多层感知器 NETtalk 的结果，网络被修剪到仅有 1 560 个权值，这在网络的大小上有急剧的减少。归功于 Sejnowski 和 Rosenberg(1987) 的 NE Ttalk，将在 4.18 节中讲述。

计算 Hessian 矩阵的逆。Hessian 矩阵的逆 $\mathbf{H}^{-1}$ 是 OBS 过程的公式基础。当网络中自由参数 $\mathbf{W}$ 的数目很大的时候，计算 $\mathbf{H}^{-1}$ 的问题可能是难以处理的。设多层感知器被完全训练到误差曲面上的局部最小，下面我们描述一个计算 $\mathbf{H}^{-1}$ 的可控过程（Hassibi and Stork，1993）。

为了简化表达，假设多层感知器具有单个输出神经元。然后对一个给定的训练集可以把式(4.5)的代价函数表示为：

$$\mathscr{E}_{av}(\mathbf{w}) = \frac{1}{2N}\sum_{n=1}^{N}(d(n) - o(n))^2$$

其中 $o(n)$ 是第 n 个样本输入时网络的实际输出，$d(n)$ 是相应的期望响应，N 是训练集中样本

的总数。输出 o(n)本身可以表示为：

$$o(n) = F(\mathbf{w},\mathbf{x})$$

其中 F 是多层感知器实现的输入输出映射函数，$\mathbf{x}$ 是输入向量，$\mathbf{w}$ 是网络的突触权值向量。因此$\mathscr{E}_{av}$对 $\mathbf{w}$ 的一阶导数为：

$$\frac{\partial \mathscr{E}_{av}}{\partial \mathbf{w}} = -\frac{1}{N}\sum_{n=1}^{N}\frac{\partial F(\mathbf{w},\mathbf{x}(n))}{\partial \mathbf{w}}(d(n)-o(n)) \tag{4.103}$$

$\mathscr{E}_{av}$对 $\mathbf{w}$ 的二阶导数或者 Hessian 矩阵是：

$$\mathbf{H}(N) = \frac{\partial^2 \mathscr{E}_{av}}{\partial \mathbf{w}^2} = \frac{1}{N}\sum_{n=1}^{N}\left\{\left(\frac{\partial F(\mathbf{w},\mathbf{x}(n))}{\partial \mathbf{w}}\right)\left(\frac{\partial F(\mathbf{w},\mathbf{x}(n))}{\partial \mathbf{w}}\right)^T - \frac{\partial^2 F(\mathbf{w},\mathbf{x}(n))}{\partial \mathbf{w}^2}(d(n)-o(n))\right\} \tag{4.104}$$

这里我们强调了 Hessian 矩阵对训练样本大小 N 的依赖性。

在网络是被完全训练的假设下，即代价函数$\mathscr{E}_{av}$被调整到误差曲面的一个局部最小值，说 $o(n)$近似于 $d(n)$是合理的。在这个条件下我们可以忽略第二项，这样式(4.104)的逼近为：

$$\mathbf{H}(N) \approx \frac{1}{N}\sum_{n=1}^{N}\left(\frac{\partial F(\mathbf{w},\mathbf{x}(n))}{\partial \mathbf{w}}\right)\left(\frac{\partial F(\mathbf{w},\mathbf{x}(n))}{\partial \mathbf{w}}\right)^T \tag{4.105}$$

为了简化符号，定义 $W\times 1$ 向量：

$$\boldsymbol{\xi}(n) = \frac{1}{\sqrt{N}}\frac{\partial F(\mathbf{w},\mathbf{x}(n))}{\partial \mathbf{w}} \tag{4.106}$$

它可以通过 4.8 节所述的过程来计算。然后我们就可以用递归的形式重写式(4.105)如下：

$$\mathbf{H}(n) = \sum_{k=1}^{n}\boldsymbol{\xi}(k)\boldsymbol{\xi}^T(k) = \mathbf{H}(n-1)+\boldsymbol{\xi}(n)\boldsymbol{\xi}^T(n),\quad n = 1,2,\cdots,N \tag{4.107}$$

这个递归正是所谓的*矩阵逆引理*应用的正确形式，它也称为 Woodbury 等式。

令 $\mathbf{A}$ 和 $\mathbf{B}$ 表示由关系

$$\mathbf{A} = \mathbf{B}^{-1} + \mathbf{CDC}^T$$

定义的正定矩阵，其中 $\mathbf{C}$ 和 $\mathbf{D}$ 是另外两个矩阵。根据矩阵逆引理，矩阵 $\mathbf{A}$ 的逆定义为

$$\mathbf{A}^{-1} = \mathbf{B} - \mathbf{BC}(\mathbf{D}+\mathbf{C}^T\mathbf{BC})^{-1}\mathbf{C}^T\mathbf{B}$$

对于式(4.107)中所述的问题我们有

$$\mathbf{A} = \mathbf{H}(n)$$
$$\mathbf{B}^{-1} = \mathbf{H}(n-1)$$
$$\mathbf{C} = \boldsymbol{\xi}(n)$$
$$\mathbf{D} = 1$$

应用矩阵逆引理得到对于 Hessian 矩阵求逆的递归计算公式：

$$\mathbf{H}^{-1}(n) = \mathbf{H}^{-1}(n-1) - \frac{\mathbf{H}^{-1}(n-1)\boldsymbol{\xi}(n)\boldsymbol{\xi}^T(n)\mathbf{H}^{-1}(n-1)}{1+\boldsymbol{\xi}^T(n)\mathbf{H}^{-1}(n-1)\boldsymbol{\xi}(n)} \tag{4.108}$$

注意式(4.108)中的分母是一个标量；因此直接计算它的倒数。这样，给定 Hessian 矩阵的逆过去的值 $\mathbf{H}^{-1}(n-1)$，我们就可以计算它由向量 $\boldsymbol{\xi}(n)$表示的第 n 个样本呈现后的更新值 $\mathbf{H}^{-1}(n)$。这个递归计算将继续到 N 个样本的整个集合被计算为止。为了初始化这个算法，我们需要使 $\mathbf{H}^{-1}(0)$很大，因为根据式(4.108)它是持续地减少的。这个要求可以通过如下设定来满足：

$$\mathbf{H}^{-1}(0) = \delta^{-1}\mathbf{I}$$

其中 δ 是一个小的正数，$\mathbf{I}$ 是单位矩阵。这个初始化的形式保证 $\mathbf{H}^{-1}(n)$总是正定的。δ 的影响随着越来越多的样本出现在网络中而变得逐渐减少。

表 4.1 是最优脑外科算法的小结。

表 4.1　最优脑外科算法小结

1. 训练给定多层感知器至最小均方误差。
2. 利用 4.8 节所述过程计算向量

$$\boldsymbol{\xi}(n)=\frac{1}{\sqrt{N}}\frac{\partial F(\mathbf{w},\mathbf{x}(n))}{\partial \mathbf{w}}$$

其中 $F(\mathbf{w},\mathbf{x}(n))$是由具有全部权值向量 $\mathbf{w}$ 的多层感知器实现的输入输出映射，$\mathbf{x}(n)$是输入向量。

3. 利用递归式(4.108)计算 Hessian 矩阵的逆 $\mathbf{H}^{-1}$。
4. 寻找相应于最小显著性的 i：

$$S_i=\frac{w_i^2}{2[\mathbf{H}^{-1}]_{i,i}}$$

其中 $[\mathbf{H}^{-1}]_{i,i}$是 $\mathbf{H}^{-1}$的第 (i,i) 个元素。如果显著性 S_i 远小于均方误差$\mathscr{E}_{av}$，那么删除突触权值 w_i，并且执行第 5 步。否则，转第 6 步。

5. 通过应用如下的调整来校正网络中所有的突触权值：

$$\Delta\mathbf{w}=-\frac{w_i}{[\mathbf{H}^{-1}]_{i,i}}\mathbf{H}^{-1}\mathbf{1}_i$$

转第 2 步。

6. 当不再有权值可以因为网络中均方误差没有大的增加而被删除的时候停止计算。(也许期望在该点重新训练网络。)

4.15　反向传播学习的优点和局限

首先最重要的是，需要理解反向传播算法不是试图实现最优设计的多层感知器的算法。描述它的正确途径应该是：

反向传播算法是一个对于计算代价函数$\mathscr{E}(w)$的梯度（即一阶导数）计算有效的技术。代价函数由刻画多层感知器的可调参数（突触权值和偏置）的函数来表示。

算法的计算能力是由两个明显的性质而导出的：

1. 反向传播算法是局部计算简单的。
2. 当算法是在线（逐次）学习时它实现权值空间的随机梯度下降。

连接机制

反向传播算法是依靠局部计算来发现神经网络信息处理能力的一个连接论者范例的例子。计算限制的这种形式称为局部约束，它是指单个神经元实现的计算仅受那些与它有物理接触的神经元的影响。在（人工）神经网络的设计中提倡利用局部计算通常有三个主要的理由：

1. 实现局部计算的神经网络常常被作为生物神经网络的类比来推举。
2. 由于使用局部计算允许由于硬件错误引起的平稳的性能下降，因此为容错网络设计提供基础。
3. 局部计算支持作为神经网络实现有效方法的并行体系结构。

复制器（恒等）映射

通过反向传播算法训练的多层感知器的隐藏神经元作为特征检测器扮演着重要的角色。利用多层感知器的这个重要性质的一个新方法是使用它作为复制器或者恒等映射（Rumelhart 等，1986b；Cottrel 等，1987）。图 4.19 表明对于使用单个隐藏层的多层感知器情况下这是如何完成的。网络构形满足如下的结构要求，正如图 4.19a 所示的那样：

- 输入和输出层神经元数目具有相同的大小 m。
- 隐藏层的神经元个数 M 小于 m。
- 网络是完全连接的。

一个给定的模式 $\mathbf{x}$ 同时作为输入层的刺激和输出层的期望响应。输出层的实际响应$\hat{\mathbf{x}}$是打算用作 $\mathbf{x}$ 的“估计”。通过常用的方法使用反向传播算法训练网络，估计误差向量（$\mathbf{x}-\hat{\mathbf{x}}$）作

为误差信号处理，如图 4.19b 所示。这个训练是在无监督情形下完成的（即不需要教师）。借助多层感知器的设计所建立的特殊结构这一优点，通过它的隐藏层约束网络以实现恒等映射。输入模式的一个编码形式，用 s 表示，它是在隐藏层的输出中产生的，如图 4.19a 所示。事实上，完全训练的多层感知器充当了“编码器”的角色。为了重构初始输入模式 $\mathbf{x}$ 的估计 $\hat{\mathbf{x}}$（即实现解码），我们将编码信号应用于复制器网络隐藏层，如图 4.19c 所示。事实上，后面的网络扮演了“解码器”的角色。如果我们使得隐藏层的大小 M 与输入/输出层大小 m 相比越小，那么图 4.19a 的结构作为一个数据压缩系统的作用就越大[10]。

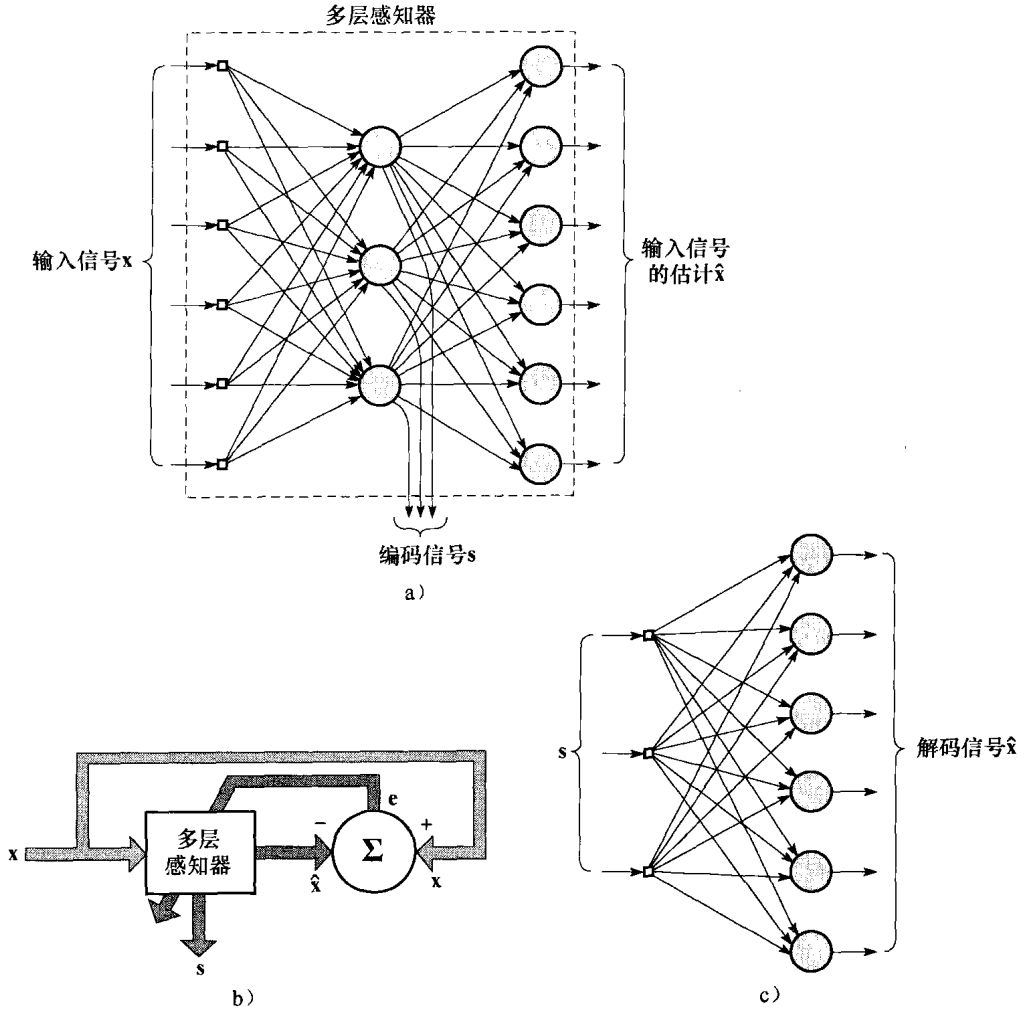


图 4.19　a）具有一个隐藏层的作为编码器的复制器网络（恒等映射）；b）复制器网络监督训练的方框图；c）作为解码器的复制器网络部分

函数逼近

通过反向传播算法训练的多层感知器自身表明是一个嵌套 sigmoid 函数结构，在单个输出的情形下用紧凑形式写为：

$$F(\mathbf{x},\mathbf{w}) = \varphi\Big(\sum_{k} w_{ok}\varphi\Big(\sum_{j} w_{kj}\varphi\Big(\cdots\varphi\Big(\sum_{i} w_{ji}x_i\Big)\Big)\Big)\Big) \tag{4.109}$$

其中 $\varphi(\cdot)$ 是 sigmoid 激活函数，w_{ok} 是从最后一个隐藏层的神经元 k 到单个输出神经元 o 的突触权值，依此类推得到其他突触权值，x_i 是输入向量 $\mathbf{x}$ 的第 i 个元素。权值向量 $\mathbf{w}$ 表示突触权

值的完整集合，其排列顺序首先按层，然后按每层中的神经元，最后按神经元中的突触。式(4.109)中嵌入非线性函数的设计在经典逼近论中是不常见的。正如第 4.12 节讨论的那样，它是一个通用逼近器。

计算的效率

算法的计算复杂度通常是用乘法、加法的次数和它的实现所涉及的存储量来衡量的。一个学习算法从一次迭代到下一次迭代，若它的计算复杂度对于要更新的可调整参数的数目而言是多项式的，我们就说这个算法是计算有效的。在这个基础上，也可以说反向传播算法是计算有效的，正如本节的开始部分所总结的那样。特别地，在使用这一算法进行包含全部的突触权值 W(包括偏置) 的多层感知器的训练中，它的计算复杂度在 W 中是线性的。反向传播算法的这个重要性质可以通过检查如 4.4 节所述的完成前向通过和反向通过所涉及的计算而容易得到证明。在前向通过中，计算涉及的突触权值是那些网络中不同神经元的诱导局部域所属的权值。这里我们从式(4.44)看到这些计算对网络的突触权值是线性的。在反向通过中，涉及突触权值的仅有的计算是那些分别由式(4.46)和式(4.47)所述的属于 (1) 隐藏神经元的局部梯度和 (2) 突触权值自身的更新。这里同样可以看到这些计算对网络的突触权值全部是线性的。因此得出结论，反向传播算法的计算复杂度对 W 是线性的，即它是 $O(W)$。

灵敏度分析

从使用反向传播学习中得到的另一个计算上的好处是它提供了一个有效的方法，通过它可以进行由这个算法实现的输入输出映射的灵敏度分析。输入输出映射函数 F 关于函数的一个参数的灵敏度，以 ω 表示，定义为：

$$S_{\omega}^{F}=\frac{\partial F/F}{\partial \omega/\omega} \tag{4.110}$$

然后考虑一个经过反向传播算法训练的多层感知器。令函数 $F(\mathbf{w})$为网络实现的输入输出映射；$\mathbf{w}$ 表示网络中包含的所有突触权值（包括偏置）向量。在 4.8 节中我们证明了函数 $F(\mathbf{w})$对权值向量 $\mathbf{w}$ 中所有元素的偏导数是可以进行有效计算的。特别地，我们知道这些偏导数计算所涉及的复杂度对网络包含权值的总数 W 是线性的。这种线性关系与问题的突触权值在计算链中出现的位置无关。

鲁棒性

在第 3 章中我们指出，LMS 算法中能量小的扰动只会引起小的估计误差，从这个角度来看它是鲁棒的。如果固有的观察模型是线性的，LMS 算法是一个 $\mathbf{H}^{\infty}$ 最优滤波器（Hassibi 等，1993，1996)。这意味着 LMS 算法最小化由估计误差的扰动带来的最大能量增益。

从另一方面来看，如果固有的观察模型是非线性的，Hassibi 和 Kailath(1995) 证明了反向传播算法是一个局部 $\mathbf{H}^{\infty}$ 最优滤波器。这里使用的“局部”术语是指反向传播算法中使用的权值向量初始值充分靠近权值向量的最优值 $\mathbf{w}^{*}$，以确保该算法不陷入一个坏的局部最小中。用概念性的说法，看到 LMS 和反向传播算法属于同一类型的 $\mathbf{H}^{\infty}$ 最优滤波器是令人满意的。

收敛性

反向传播算法在权值空间中对于误差曲面上的梯度使用“瞬时估计”。因此该算法在本质上是随机的；也就是说，它在误差曲面上具有通过在真实方向附近的锯齿形路线趋于最小点的倾向。其实，反向传播学习是最初由 Robbins and Monro(1951) 提出的所谓随机逼近的统计学方法的一个应用。因此，它倾向于缓慢收敛。我们可以验明这个性质的两个基本原因 (Jacobs，1988)：

1. 误差曲面沿着一个权值方向是相当平坦的，这意味着误差曲面对这个权值的导数在数

量上是很小的。在这种情况下，应用于这个权值的调整是很小的，因此在网络误差性能上产生重大的降低可能要求这个算法的多次迭代。或另一方面，误差曲面沿着一个权值方向是高度弯曲的，在这种情形下误差曲面对该权值的导数在数量上是很大的。在这第二种情况下，应用于该权值的调整是很大的，这可能会导致该算法越过误差曲面的最小点。

2. 负梯度向量的方向（即代价函数对权值向量的负导数）可能指向远离误差曲面的最小值：因此应用于权值的调整可能导致算法往错误的方向进行。

为了避免用于训练多层感知器的误差反向传播算法的慢速收敛，我们可以选择如 4.10 节所描述的最优退火在线学习算法。

局部最小值

对反向传播算法性能造成影响的误差曲面的另一个特点是除了全局最小值之外的局部最小值（即孤立凹槽）的出现。通常，很难确定有多少个局部和全局最小值。由于反向传播学习基本上是一个爬山技术，因此它存在陷入局部最小值的危险，此处突触权值的每个微小变化都会引起代价函数的增长。但在权值空间的别的某个地方存在另外一个突触权值的集合，它的代价函数的值比在网络被停止处的局部最小值更小。我们显然不希望学习进程停止在局部最小值，特别是当它处于远高于全局最小值的位置。

规模

原则上，诸如由反向传播算法训练的多层感知器之类的神经网络具有成为通用计算机器的潜在可能。然而，要充分实现这种潜能，必须克服规模（scaling）问题，它是指随计算任务在大小和复杂性上的增加网络表现的优劣（如由训练所需时间和可得到的最优泛化性能来衡量）的问题。在度量计算任务大小和复杂度的许多可能的办法中，由 Minsky and Papert(1969, 1988) 定义的谓词阶（predicate order）提供了最有用也是最重要的标准。

为了解释一个谓词意味着什么，令 $\psi(X)$表示一个只能有两个取值的函数。通常取 $\psi(X)$ 的两个值为 0 和 1。但通过取值为假（FALSE）或真（TRUE），可以认为 $\psi(X)$是一个谓词，即一个可变的陈述，其真和假依赖于变量 X 的选择。例如，我们可以写出

$$\psi_{\text{CIRCLE}}(X)=\begin{cases}1 & \text{若图形 } X \text{ 是一个圆}\\ 0 & \text{若图形 } X \text{ 不是一个圆}\end{cases}$$

使用谓词的思想，Tesauro and Janssens(1988) 进行实证研究，使用反向传播算法训练多层感知器来学习计算奇偶函数。奇偶函数是如下定义的布尔谓词：

$$\psi_{\text{PARITY}}(X)=\begin{cases}1 & \text{若 } |X| \text{ 是奇数}\\ 0 & \text{否则}\end{cases}$$

它的阶数等于输入的个数。Tesauro 和 Janssens 进行的这个实验显示，网络学习计算奇偶函数所需的时间与输入个数（即计算的谓词阶数）呈指数关系，并且使用反向传播算法学习任意复杂的函数的计划可能是过于乐观的。

一般认为对一个多层感知器进行完全连接是失策的。因此，在此背景下，我们可以提出如下问题：给定一个不应被完全连接的多层感知器，网络的突触连接将如何分配？这个问题在小规模的应用情况并不是主要考虑的问题，但它对利用反向传播学习解决现实世界中大规模的问题的成功应用是至关重要的。

减轻规模问题的一个有效办法是发展对当前问题的认识（可能是通过神经生物学的类比）并利用它增加多层感知器体系结构设计的灵活性。具体地讲，网络体系结构和加于网络突触权值上的约束应该这样设计，以使关于任务的先验知识合并到网络的组成中去。这种设计策略在第 4.17 节中在关于光学字符识别的问题中说明。

4.16　作为最优化问题看待的监督学习

本节用一种与前面几节讨论有很大不同的关于监督学习的观点。具体地讲，我们把多层感知器的监督训练看作是一个数值最优化问题。在这个背景下我们首先指出使用监督学习的多层感知器的误差曲面是突触权值向量 $\mathbf{w}$ 的高度非线性函数；在多层感知器的情形下，$\mathbf{w}$ 表示网络中以某种顺序排列的突触权值。令 $\mathscr{E}_{av}(\mathbf{w})$ 表示在训练样本上的平均代价函数。使用 Taylor 级数，在误差曲面当前运行点附近我们可以如式(4.97)那样展开 $\mathscr{E}_{av}(\mathbf{w})$，这里重写为：

$$\begin{aligned}\mathscr{E}_{av}(\mathbf{w}(n)+\Delta\mathbf{w}(n)) &= \mathscr{E}_{av}(\mathbf{w}(n))+\mathbf{g}^T(n)\Delta\mathbf{w}(n)\\ &\quad+\frac{1}{2}\Delta\mathbf{w}^T(n)\mathbf{H}(n)\Delta\mathbf{w}(n)+(\text{三次和更高次项})\end{aligned} \tag{4.111}$$

其中 $\mathbf{g}(n)$ 是局部梯度向量，定义为：

$$\mathbf{g}(n)=\left.\frac{\partial\mathscr{E}_{av}(\mathbf{w})}{\partial\mathbf{w}}\right|_{\mathbf{w}=\mathbf{w}(n)} \tag{4.112}$$

$\mathbf{H}(n)$ 是局部 Hessian 矩阵，表示误差性能曲面的“曲率”，定义为：

$$\mathbf{H}(n)=\left.\frac{\partial^2\mathscr{E}_{av}(\mathbf{w})}{\partial\mathbf{w}^2}\right|_{\mathbf{w}=\mathbf{w}(n)} \tag{4.113}$$

总体-平均代价函数 $\mathscr{E}_{av}(\mathbf{w})$ 的使用预设了批量学习。

在以反向传播算法为例的最速下降法中，应用于突触权值向量 $\mathbf{w}(n)$ 的调节量 $\Delta\mathbf{w}(n)$ 定义为：

$$\Delta\mathbf{w}(n)=-\eta\mathbf{g}(n) \tag{4.114}$$

其中 η 为固定的学习率参数。事实上，最速下降法是在运行点 $\mathbf{w}(n)$ 的局部邻域对代价函数的线性逼近基础上进行计算的。在这样的处理中，它依赖于作为误差曲面局部信息唯一来源的梯度向量 $\mathbf{g}(n)$。这个限制具有一个有利的效果：实现的简单性。不幸的是，它同样具有一个不利的影响：缓慢的收敛速度，特别是在大规模问题的情形下这是令人烦恼的。在权值更新的公式中包含动量项是使用误差曲面二阶信息的大胆尝试，这样做有一些帮助。然而，由于必须在由设计者“调整”的参数列表中增加一项，它的使用使得训练过程的管理更费时间。

为了使多层感知器的收敛性能有显著的改善（与反向传播学习相比），必须使用训练过程的高阶信息。我们可以通过调用误差曲面在当前点 $\mathbf{w}(n)$ 周围的二次逼近来实现。然后从式(4.111)可以发现应用于突触权值向量 $\mathbf{w}(n)$ 的调整量的最优值 $\Delta\mathbf{w}(n)$ 由下式给出：

$$\Delta\mathbf{w}^*(n)=\mathbf{H}^{-1}(n)\mathbf{g}(n) \tag{4.115}$$

其中 $\mathbf{H}^{-1}(n)$ 是 Hessian 矩阵 $\mathbf{H}(n)$ 的逆，假设它是存在的。式(4.115)是牛顿法的核心。如果代价函数 $\mathscr{E}_{av}(\mathbf{w})$ 是二次的（式(4.109)中的三次和更高次项为零），那么牛顿法一次迭代后收敛到最优值位置。然而，牛顿法对多层感知器的有监督训练的实际应用受到三个因素的阻碍：

- 牛顿法要求计算 Hessian 矩阵的逆 $\mathbf{H}^{-1}(n)$，这可能在计算上需要很大的开销。
- 为了使 $\mathbf{H}^{-1}(n)$ 是可计算的，$\mathbf{H}(n)$ 必须是非奇异的。在 $\mathbf{H}(n)$ 为正定的情况下，当前点 $\mathbf{w}(n)$ 周围的误差曲面可以描述为“凸碗状”。遗憾的是，并不能保证多层感知器误差曲面的 Hessian 矩阵总是符合这样的描述。而且，还有 Hessian 矩阵秩亏损的潜在问题（即并不是所有的 $\mathbf{H}$ 的列都线性无关），这是由于网络训练问题中固有的病态性所造成的（Saarinen 等，1992）；这只会使得计算任务更加困难。
- 当代价函数 $\mathscr{E}_{av}(\mathbf{w})$ 是非二次的时候，牛顿法的收敛性得不到保证，这使得它不适合于训练多层感知器。

为了克服其中某些困难，我们可以使用拟牛顿法，它仅仅要求梯度向量 $\mathbf{g}$ 的一个估计值。这种牛顿法的修正不经过计算矩阵的逆而直接得到逆矩阵 $\mathbf{H}^{-1}$ 的正定估计。通过使用这样的估计，拟牛顿法保证在误差曲面上是下降的。然而，我们仍然有一个 $O(W^2)$ 的计算复杂度，其中 W 是权值向量 $\mathbf{w}$ 的大小。因此拟牛顿法在计算上是不可行的，除非对一个非常小规模的神经网络进行训练。关于拟牛顿法将在本节后面讨论。

另一类型的二阶最优化方法包括共轭梯度方法，它被认为是一种介于最速下降法和牛顿法之间的方法。使用共轭梯度方法的动机是期望加速在最速下降法中特别缓慢的收敛速度，同时避免在牛顿法中要求对 Hessian 矩阵的估值、存储和求逆。

共轭梯度方法[11]

共轭梯度方法属于人所共知的共轭方向方法的二阶最优化方法的一类。我们通过考虑二次函数：

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} - \mathbf{b}^T\mathbf{x} + c \tag{4.116}$$

的最小化来开始这些方法的讨论，其中 x 是一个 $W\times 1$ 参数向量，$\mathbf{A}$ 是 $W\times W$ 对称正定矩阵，$\mathbf{b}$ 是 $W\times 1$ 向量，c 是标量。二次函数 $f(\mathbf{x})$ 的最小化是通过赋予 x 如下唯一值得到的：

$$\mathbf{x}^* = \mathbf{A}^{-1}\mathbf{b} \tag{4.117}$$

这样 $f(\mathbf{x})$ 的最小化和求解方程 $\mathbf{A}\mathbf{x}^* = \mathbf{b}$ 的线性系统就是等价问题。

给定矩阵 $\mathbf{A}$，如果满足下述条件，则称非零向量 $\mathbf{s}(0),\mathbf{s}(1),\cdots,\mathbf{s}(W-1)$ 的集合是 $\mathbf{A}$-共轭的（即在矩阵 $\mathbf{A}$ 下互不干扰）：

$$\mathbf{s}^T(n)\mathbf{A}\mathbf{s}(j) = 0 \qquad \text{对所有 } n \neq j \tag{4.118}$$

如果 $\mathbf{A}$ 等于单位矩阵，共轭就等同于通常的正交性概念。

例 1　A-共轭向量的解释

为了解释 $\mathbf{A}$-共轭向量，考虑图 4.20a 所示的属于二维问题的情形。图中所示椭圆轨迹对应于方程（4.116）在

$$\mathbf{x} = [x_0, x_1]^T$$

对二次函数 $f(\mathbf{x})$ 指定的某个常数值的图形。图 4.20a 也包括一对关于矩阵 $\mathbf{A}$ 共轭的方向向量。假定我们通过变换

$$\mathbf{v} = \mathbf{A}^{1/2}\mathbf{x}$$

定义一个新的与 $\mathbf{x}$ 相关的参数向量 $\mathbf{v}$，其中 $\mathbf{A}^{1/2}$ 是 $\mathbf{A}$ 的平方根。这样图 4.20a 中椭圆轨迹就被变换为图 4.20b 所示的圆形轨迹，相应地，图 4.20a 中 $\mathbf{A}$-共轭的方向向量对也被转换为图 4.20b 中的一对正交方向向量。 ■

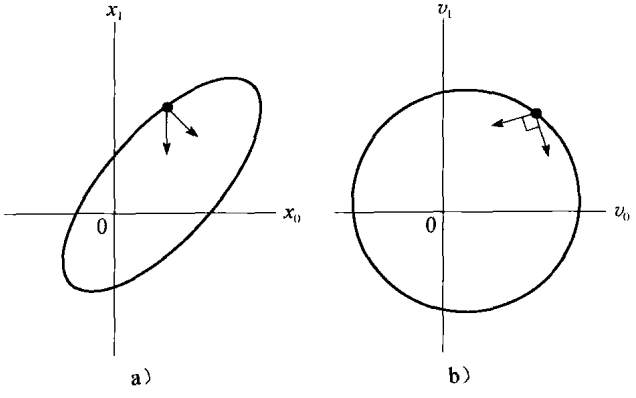


图 4.20　A-共轭向量解释。a）二维权值空间的椭圆轨迹；b）椭圆轨迹到圆形轨迹的变换

关于 $\mathbf{A}$-共轭向量的一个重要性质是它们是线性无关的。我们可以用反证法证明这个性质。令这些向量的其中之一，比如 $\mathbf{s}(0)$，用其余 $W-1$ 个向量的线性组合表示如下：

$$\mathbf{s}(0) = \sum_{j=1}^{W-1} \alpha_j \mathbf{s}(j)$$

两边乘以 $\mathbf{A}$ 并用 $\mathbf{A}\mathbf{s}(0)$ 和 $\mathbf{s}(0)$ 作内积得到

$$\mathbf{s}^T(0)\mathbf{A}\mathbf{s}(0) = \sum_{j=1}^{W-1} \alpha_j \mathbf{s}^T(0)\mathbf{A}\mathbf{s}(j) = 0$$

然而，有两个原因使得二次型 $\mathbf{s}^T(0)\mathbf{A}\mathbf{s}(0)$ 不可能为零：矩阵 $\mathbf{A}$ 是被假设为正定的，向量 $\mathbf{s}(0)$

定义为非零。因此可以得出 **A**-共轭的向量 $\mathbf{s}(0),\mathbf{s}(1),\cdots,\mathbf{s}(W-1)$ 不能是线性相关的；也就是说，它们必须是线性无关的。

对于一个给定的 **A**-共轭向量 $\mathbf{s}(0),\mathbf{s}(1),\cdots,\mathbf{s}(W-1)$ 的集合，相应的二次误差函数 $f(\mathbf{x})$ 的无约束最小化共轭方向方法定义为

$$\mathbf{x}(n+1)=\mathbf{x}(n)+\eta(n)\mathbf{s}(n)\qquad n=0,1,\cdots,W-1 \tag{4.119}$$

其中 $\mathbf{x}(0)$是任意的开始向量，$\eta(n)$是由

$$f(\mathbf{x}(n)+\eta(n)\mathbf{s}(n))=\min_{\eta} f(\mathbf{x}(n)+\eta\mathbf{s}(n)) \tag{4.120}$$

定义的标量（Fletcher，1987；Bertsekas，1995）。通过选择 η 对某个固定的 n 寻找使函数 $f(\mathbf{x}(n)+\eta\mathbf{s}(n))$最小的过程称为线搜索，这表示一维最小化问题。

根据式(4.118)，(4.119)和(4.120)，可以得到如下结果：

1. 由于 **A**-共轭的向量 $\mathbf{s}(0),\mathbf{s}(1),\cdots,\mathbf{s}(W-1)$ 线性无关，它们组成 **w** 的向量空间的一组基。

2. 更新公式(4.119)和式(4.120)的线最小化导出学习率参数相同的公式，即

$$\eta(n)=-\frac{\mathbf{s}^T(n)\mathbf{A}\mathbf{e}(n)}{\mathbf{s}^T(n)\mathbf{A}\mathbf{s}(n)},\qquad n=0,1,\cdots,W-1 \tag{4.121}$$

其中 $\epsilon(n)$是误差向量，定义为

$$\mathbf{e}(n)=\mathbf{x}(n)-\mathbf{x}^* \tag{4.122}$$

3. 从任意一个点 $\mathbf{x}(0)$出发，共轭方向方法确保能在最多迭代 W 次中找到二次函数 $f(\mathbf{x})=0$ 的最优解 $\mathbf{x}^*$。

共轭方向方法的主要性质如下（Fletcher，1987；Bertsekas，1995）：

在连续的迭代中，共轭方向方法在逐渐扩张的线性向量空间上最小化二次函数 $f(\mathbf{x})$，最终包含 $f(\mathbf{x})$的全局最小值。

特别地，对于每次迭代 n，迭代结果 $\mathbf{x}(n+1)$ 在通过某个任意点 $\mathbf{x}(0)$并且由 **A**-共轭的向量 $\mathbf{s}(0),\mathbf{s}(1),\cdots,\mathbf{s}(n)$ 扩展成的线性向量空间$\mathcal{D}_n$ 上使函数 $f(\mathbf{x})$最小化，表示为

$$\mathbf{x}(n+1)=\arg\min_{\mathbf{x}\in\mathcal{D}_n} f(\mathbf{x}) \tag{4.123}$$

其中空间$\mathcal{D}_n$ 定义为

$$\mathcal{D}_n=\left\{\mathbf{x}(n)\,|\,\mathbf{x}(n)=\mathbf{x}(0)+\sum_{j=0}^{n}\eta(j)\mathbf{s}(j)\right\} \tag{4.124}$$

为了使共轭方向方法起作用，要求具备一个可用的 **A**-共轭向量 $\mathbf{s}(0),\mathbf{s}(1),\cdots,\mathbf{s}(W-1)$ 集合。在这种方法的一种称为*共轭梯度方法*[12]的特殊形式中，随着方法的逐步进行，二次函数 $f(\mathbf{x})$的后继梯度向量的 *A*-共轭形式被作为后继方向向量而产生，因此以此来命名这种方法。这样，除了 $n=0$ 之外，方向向量的集合 $\{\mathbf{s}(n)\}$ 并不是预先指定的，相反它是在该方法的后继的步骤中串行决定的。

首先，定义残差作为最速下降方向：

$$\mathbf{r}(n)=\mathbf{b}-\mathbf{A}\mathbf{x}(n) \tag{4.125}$$

进而通过 $\mathbf{r}(n)$和 $\mathbf{s}(n-1)$的线性组合来继续，表示为：

$$\mathbf{s}(n)=\mathbf{r}(n)+\beta(n)\mathbf{s}(n-1),\quad n=1,2,\cdots,W-1 \tag{4.126}$$

其中 $\beta(n)$是需要确定的一个比例因子。利用方向向量 **A**-共轭的性质，方程的两边乘以 **A**，并将结果表达式和 $\mathbf{s}(n-1)$作内积，然后求解 $\beta(n)$的结果表达式，得到

$$\beta(n)=-\frac{\mathbf{s}^T(n-1)\mathbf{A}\mathbf{r}(n)}{\mathbf{s}^T(n-1)\mathbf{A}\mathbf{s}(n-1)} \tag{4.127}$$

通过式(4.126)和式(4.127)，我们发现这样得到的向量 $\mathbf{s}(0),\mathbf{s}(1),\cdots,\mathbf{s}(W-1)$ 确实是 **A**-共轭的。

根据递归公式(4.126)产生的方向向量依赖于系数 $\beta(n)$。由于 $\beta(n)$ 目前的表示形式，计算 $\beta(n)$ 的公式(4.127)需要矩阵 **A** 的知识。出于计算上的原因，希望不利用 **A** 的明显知识的情况下对 $\beta(n)$ 进行计算。这样的计算可以通过两个不同公式中的一个得到 (Fletcher，1987)：

1. Polak-Ribiére 公式，其中 $\beta(n)$ 定义为：

$$\beta(n)=\frac{\mathbf{r}^T(n)(\mathbf{r}(n)-\mathbf{r}(n-1))}{\mathbf{r}^T(n-1)\mathbf{r}(n-1)} \tag{4.128}$$

2. Fletcher-Reeves 公式，其中 $\beta(n)$ 定义为：

$$\beta(n)\frac{\mathbf{r}^T(n)\mathbf{r}(n)}{\mathbf{r}^T(n-1)\mathbf{r}(n-1)} \tag{4.129}$$

为了用共轭梯度方法处理属于多层感知器无监督训练的代价函数 $\mathscr{E}_{av}(\mathbf{w})$ 的无约束最优化问题，我们做两件事情：

- 用一个二次函数来逼近代价函数 $\mathscr{E}_{av}(\mathbf{w})$。也就是说，式(4.111)中三阶和更高阶项被忽略，这意味着我们正在逼近误差曲面上的一个局部最小值。在这个基础上，比较式(4.111)和式(4.116)，可以得到表 4.2 显示的联系。
- 用公式表示在共轭梯度算法中系数 $\beta(n)$ 和 $\eta(n)$ 的计算，使得仅仅要求梯度信息。

表 4.2　$f(\mathbf{x})$ 和 $\mathscr{E}_{av}(\mathbf{w})$ 之间的对应

二次函数 $f(\mathbf{x})$	代价函数 $\mathscr{E}_{av}(\mathbf{w})$
参数向量 $\mathbf{x}(n)$	突触权值向量 $\mathbf{w}(n)$
梯度向量 $\partial f(\mathbf{x})/\partial \mathbf{x}$	梯度向量 $\mathbf{g}=\partial \mathscr{E}_{av}/\partial \mathbf{w}$
矩阵 **A**	Hessian 矩阵 **H**

后面一点在多层感知器中特别重要，因为它避免了使用 Hessian 矩阵 $\mathbf{H}(n)$，该矩阵的估值会遭遇严重的计算困难。

当没有 Hessian 矩阵 $\mathbf{H}(n)$ 的明显知识时，为了计算决定搜索方向 $\mathbf{s}(n)$ 的系数 $\beta(n)$，可以利用式(4.128)的 Polak-Ribiére 公式或者式(4.129)中的 Fletcher-Reeves 公式。这两个公式都仅包含残差的使用。假定一个二次函数，在共轭梯度方法的线性形式中，Polak-Ribiére 公式和 Fletcher-Reeves 公式是等价的。另一方面，在非二次代价函数的情形下，它们不再等价。

对于非二次最优化问题，共轭梯度算法的 Polak-Ribiére 形式优先于该算法的 Fletcher-Reeves 式，针对这个问题我们在下面提供启发性的解释 (Bertsekas，1995)：由于代价函数 $\mathscr{E}_{av}(\mathbf{w})$ 中三阶与更高阶项的存在和线搜索中可能的不精确性，所产生的搜索方向的共轭性逐渐丧失。这使得生成的方向向量 $\mathbf{s}(n)$ 近似正交于残差 $\mathbf{r}(n)$，在这种意义上算法可能陷入“堵塞”。当这种现象出现的时候，我们有 $\mathbf{r}(n)=\mathbf{r}(n-1)$，在这种情况下标量 $\beta(n)$ 接近于零。相应地，方向向量 $\mathbf{s}(n)$ 近似于残差 $\mathbf{r}(n)$，从而打破堵塞。与此相反，当使用 Fletcher-Reeves 公式的时候，共轭梯度算法在相似的条件下继续堵塞。

然而，在极少数的情况下，Polak-Ribiére 方法可以无限循环下去而不收敛。值得庆幸的是，Polak-Ribiére 方法的收敛可以通过选择

$$\beta=\max\{\beta_{PR},0\} \tag{4.130}$$

得到保证，其中 β_{PR} 是由式(4.128)的 Polak-Ribière 公式定义的值 (Shewchuk，1994)。如果 $\beta_{PR}<0$，利用式(4.130)中定义的 β 的值等于重新开始共轭梯度算法。重新开始运算等于遗忘最后的搜索方向并且在最速下降方向上重新开始。

考虑下一个计算参数 $\eta(n)$ 的问题，它决定了共轭梯度算法的学习率。和计算 $\beta(n)$ 一样，计算 $\eta(n)$ 的首选方法是避免必须使用 Hessian 矩阵 $\mathbf{H}(n)$。回忆一下，基于式(4.120)的线最小化导出的 $\eta(n)$ 的公式和源于更新公式(4.119)得到的 $\eta(n)$ 计算公式相同。因此需要一个直线搜

索[12]，这样的目的是对 η 最小化函数 $\mathscr{E}_{av}(\mathbf{w}+\eta\mathbf{s})$。也就是说，给定向量 $\mathbf{w}$ 和 $\mathbf{s}$ 的固定值，现在的问题是改变 η 使得函数最小化。随着 η 的变化，自变量 $\mathbf{w}+\eta\mathbf{s}$ 在 $\mathbf{w}$ 的 W 维向量空间中画出一条直线，因此称为“直线搜索”。直线搜索算法是一个迭代过程，它为共轭梯度算法的每次迭代产生一个估计序列 $\{\eta(n)\}$。当找到令人满意的解时，直线搜索被停止。直线搜索必须在每个搜索方向上进行。

文献中提出了几种直线索搜算方法，并且选择一个好的算法是重要的，因为它对被嵌入其中的共轭梯度法的性能具有深远的影响。任何直线搜索算法都有两个阶段（Fletcher，1987）：

- 区间阶段，也就是搜索一段区间（bracket）（即包含一个最小值的非平凡间隔）；
- 截段阶段，在这个阶段中，阶层被截成段（即被分割），因此产生一系列长度越来越小的子区间。

现在我们给出一个直接处理这两个阶段的曲线拟合过程。

令 $\mathscr{E}_{av}(\eta)$ 表示多层感知器的代价函数，表示为 η 的函数。假设 $\mathscr{E}_{av}(\eta)$ 是严格单峰的（unimodal）（即它在当前点 $\mathbf{w}(n)$ 的附近只有单一的最小值）并且是二次连续可微的。我们沿直线开始搜索过程，直到求出满足条件：

$$\mathscr{E}_{av}(\eta_1)\geqslant\mathscr{E}_{av}(\eta_3)\geqslant\mathscr{E}_{av}(\eta_2),\quad \text{当 } \eta_1<\eta_2<\eta_3 \tag{4.131}$$

的三个点 η_1、η_2、η_3，如图 4.21 所示。由于 $\mathscr{E}_{av}(\eta)$ 是 η 的连续函数，式(4.131)描述的选择保证区间 $[\eta_1,\eta_3]$ 包含函数 $\mathscr{E}_{av}(\eta)$ 的一个最小值。假设函数 $\mathscr{E}_{av}(\eta)$ 充分光滑，可以认为这个函数在紧邻最小值的区间是抛物线形的。因此，可以使用反抛物线插值法（inverse parabolic interpolation）进行分段（Press 等，1988）。具体地讲，这个抛物线函数可以通过三个初始点 η_1、η_2、η_3 拟合，如图 4.22 所示，图中实线对应于 $\mathscr{E}_{av}(\eta)$，虚线表示分段过程的第一次迭代。令 η_4 表示通过三点 η_1、η_2、η_3 的抛物线的最小值点。在图 4.22 所示的例子中，我们有 $\mathscr{E}_{av}(\eta_4)<\mathscr{E}_{av}(\eta_2)$，$\mathscr{E}_{av}(\eta_4)<\mathscr{E}_{av}(\eta_1)$。点 η_3 由 η_4 代替得到新的区间 $[\eta_1,\eta_4]$。通过构造一条通过点 η_1、η_2、η_4 的抛物线重复这个过程。上述包括区间后再分段的过程重复多次，直到找到一个足够接近 $\mathscr{E}_{av}(\eta)$ 的最小值的点，此时直线搜索终止。

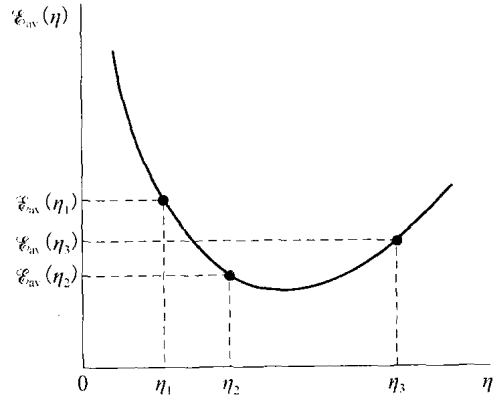


图 4.21　直线搜索示意图

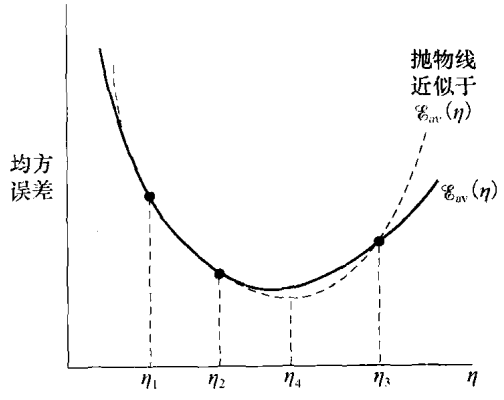


图 4.22　反抛物线插值

Brent 方法建立刚才所述的三点曲线拟合过程的一个高度精练的形式（Press 等，1988）。在计算的任何特殊阶段，Brent 方法保持 $\mathscr{E}_{av}(\eta)$ 函数 6 个点的轨迹，所有点可能不必互不相同。如前所述，抛物线插值试图通过这些点中的三个。为了使得这个插值法是可接受的，剩下的三点必须满足一定标准。最终结果是一个鲁棒直线搜索算法。

非线性共轭梯度算法小结

现在我们已经给出了正式描述用于多层感知器监督训练的共轭梯度算法的非线性（非二

次）形式所有需要的要素。表 4.3 给出了该算法的小结。

表 4.3　用于多层感知器有监督训练的非线性共轭梯度算法小结

初始化

除非权值向量 $\mathbf{w}$ 的先验知识是可用的，否则使用与反向传播算法相似的过程选择初始值 $\mathbf{w}(0)$。

计算

1. 对于 $\mathbf{w}(0)$，用反向传播算法计算梯度向量 $\mathbf{g}(0)$。
2. 置 $\mathbf{s}(0)=\mathbf{r}(0)=-\mathbf{g}(0)$。
3. 在时间步 n，用直线搜索寻找充分最小化 $\mathscr{E}_{av}(\eta)$ 的 $\eta(n)$，对于固定的 $\mathbf{w}$ 和 $\mathbf{s}$，代价函数 $\mathscr{E}_{av}$ 表示为 η 的函数。
4. 测试决定 $\mathbf{r}(n)$ 的欧几里得范数是否下降到一个特定的值之下，即为初始值 $\|\mathbf{r}(0)\|$ 的很小的一部分。
5. 更新权值向量：
$$\mathbf{w}(n+1)=\mathbf{w}(n)+\eta(n)\mathbf{s}(n)$$
6. 对于 $\mathbf{w}(n+1)$，用反向传播算法计算更新的梯度向量 $\mathbf{g}(n+1)$。
7. 置 $\mathbf{r}(n+1)=-\mathbf{g}(n+1)$。
8. 用 Polak-Ribière 方法计算 $\beta(n+1)$：
$$\beta(n+1)=\max\left\{\frac{\mathbf{r}^T(n+1)(\mathbf{r}(n+1)-\mathbf{r}(n))}{\mathbf{r}^T(n)\mathbf{r}(n)},0\right\}$$
9. 更新方向向量：
$$\mathbf{s}(n+1)=\mathbf{r}(n+1)+\beta(n+1)\mathbf{s}(n)$$
10. 置 $n=n+1$，转第 3 步。

停止准则。当下述条件满足时结束算法：
$$\|\mathbf{r}(n)\|\leqslant\varepsilon\|\mathbf{r}(0)\|$$
其中 ε 是一个指定的小的数。

拟牛顿法

重新开始讨论拟牛顿法，我们发现这基本上是用更新公式：

$$\mathbf{w}(n+1)=\mathbf{w}(n)+\eta(n)\mathbf{s}(n) \tag{4.132}$$

表示的梯度方法，其中方向向量 $\mathbf{s}(n)$ 用梯度向量 $\mathbf{g}(n)$ 定义为：

$$\mathbf{s}(n)=-\mathbf{S}(n)\mathbf{g}(n) \tag{4.133}$$

矩阵 $\mathbf{S}(n)$ 是在每次迭代中调整的正定矩阵。这样做是为了使得方向向量 $\mathbf{s}(n)$ 逼近牛顿方向，即

$$-(\partial^2\mathscr{E}_{av}/\partial\mathbf{w}^2)^{-1}(\partial\mathscr{E}_{av}/\partial\mathbf{w})$$

拟牛顿法使用误差曲面的二阶（曲率）信息，实际上不要求 Hessian 矩阵 $\mathbf{H}$ 的知识。这通过使用两次连续迭代 $\mathbf{w}(n)$、$\mathbf{w}(n+1)$ 与梯度向量 $\mathbf{g}(n)$、$\mathbf{g}(n+1)$ 来实现。令

$$\mathbf{q}(n)=\mathbf{g}(n+1)-\mathbf{g}(n) \tag{4.134}$$

和

$$\Delta\mathbf{w}(n)=\mathbf{w}(n+1)-\mathbf{w}(n) \tag{4.135}$$

这样可以通过逼近式：

$$\mathbf{q}(n)\simeq\left(\frac{\partial}{\partial\mathbf{w}}\mathbf{g}(n)\right)\Delta\mathbf{w}(n) \tag{4.136}$$

得到曲率信息。特别地，给定 W 个线性独立的权值增量 $\Delta\mathbf{w}(0),\Delta\mathbf{w}(1),\cdots,\Delta\mathbf{w}(W-1)$ 和各自的梯度增量 $\mathbf{q}(0),\mathbf{q}(1),\cdots,\mathbf{q}(W-1)$，可以逼近 Hessian 矩阵 $\mathbf{H}$ 如下：

$$\mathbf{H}\simeq[\mathbf{q}(0),\mathbf{q}(1),\cdots,\mathbf{q}(W-1)][\Delta\mathbf{w}(0),\Delta\mathbf{w}(1),\cdots,\Delta\mathbf{w}(W-1)]^{-1} \tag{4.137}$$

也可以逼近逆 Hessian 矩阵如下：

$$\mathbf{H}^{-1}\simeq[\Delta\mathbf{w}(0),\Delta\mathbf{w}(1),\cdots,\Delta\mathbf{w}(W-1)][\mathbf{q}(0),\mathbf{q}(1),\cdots,\mathbf{q}(W-1)]^{-1} \tag{4.138}$$

当代价函数 $\mathscr{E}_{av}(\mathbf{w})$ 为二次函数的时候，式(4.137)和式(4.138)是精确的。

在最常用的一类拟牛顿法中，矩阵 $\mathbf{S}(n+1)$由它先前的值 $\mathbf{S}(n)$，向量 $\Delta\mathbf{w}(n)$和 $\mathbf{q}(n)$使用如下的递归算式得到（Fletcher，1987；Bertsekas，1995）：

$$\mathbf{S}(n+1)=\mathbf{S}(n)+\frac{\Delta\mathbf{w}(n)\Delta\mathbf{w}^T(n)}{\mathbf{q}^T(n)\mathbf{q}(n)}-\frac{\mathbf{S}(n)\mathbf{q}(n)\mathbf{q}^T(n)\mathbf{S}(n)}{\mathbf{q}^T(n)\mathbf{S}(n)\mathbf{q}(n)}+\xi(n)[\mathbf{q}^T(n)\mathbf{S}(n)\mathbf{q}(n)][\mathbf{v}(n)\mathbf{v}^T(n)] \tag{4.139}$$

其中

$$\mathbf{v}(n)=\frac{\Delta\mathbf{w}(n)}{\Delta\mathbf{w}^T(n)\Delta\mathbf{w}(n)}-\frac{\mathbf{S}(n)\mathbf{q}(n)}{\mathbf{q}^T(n)\mathbf{S}(n)\mathbf{q}(n)} \tag{4.140}$$

并且

$$0\leqslant\xi(n)\leqslant 1,\quad \text{对于所有 } n \tag{4.141}$$

该算法由任意定义的正定矩阵 $\mathbf{S}(0)$进行初始化。拟牛顿法的特殊形式参数化为如何定义标量 $\xi(n)$，如下面的两点所示（Fletcher，1987）：

- 对于所有 n 满足 $\xi(n)=0$，我们得到 Davidon-Fletcher-Powell(DFP) 算法，它是历史上最初的拟牛顿法。
- 对于所有 n 满足 $\xi(n)=1$，我们得到 Broyden-Fletcher-Goldfarb-Shanno(BFGS) 算法，它在目前被认为是拟牛顿法的最好形式。

拟牛顿法和共轭梯度法的比较

我们通过在非二次最优化问题背景下对拟牛顿法和共轭梯度法的比较，来结束拟牛顿法的简要讨论（Bertsekas，1995）：

- 拟牛顿法和共轭梯度法都不使用 Hessian 矩阵。然而，拟牛顿法通过逼近逆 Hessian 矩阵来进行下一步计算。所以，当直线搜索是精确的并且充分逼近一个具有正定 Hessian 矩阵的局部最小值时，拟牛顿法趋于逼近牛顿法，因此得到的收敛速度比共轭梯度法可能的收敛速度更快。
- 拟牛顿法不如共轭梯度法那样对在最优化的直线搜索阶段的精度敏感。
- 除了方向向量 $\mathbf{S}(n)$计算相关的矩阵向量乘法之外，拟牛顿法还要求存储矩阵 $\mathbf{S}(n)$。最后结果是拟牛顿法的计算复杂度是 $O(W^2)$。相反，共轭梯度法的计算复杂度为 $O(W)$。这样，当维数 W(即权值向量 $\mathbf{w}$ 的个数）很大时，共轭梯度法比拟牛顿法在计算上具有更大的优越性。

正是因为最后这一点，实际上拟牛顿法限于小规模神经网络的设计。

Levenberg-Marquardt 方法

归功于 Levenberg(1994) 和 Marquardt(1963) 的 Levenberg-Marquardt 方法，是如下两种方法的折中：

- 牛顿法，在局部或者全局最小点附近快速收敛，但也可能发散；
- 梯度下降，通过对于步长参数的正确选择保证了收敛性，但收敛缓慢。

具体来说，考虑二阶函数 $F(\mathbf{w})$的最优化，且令 $\mathbf{g}$ 为其梯度向量，$\mathbf{H}$ 为其 Hessian 矩阵。根据 Levenberg-Marquardt 方法，作用于参数向量 $\mathbf{w}$ 的最优调整量 $\Delta\mathbf{w}$ 定义为：

$$\Delta\mathbf{w}=[\mathbf{H}+\lambda\mathbf{I}]^{-1}\mathbf{g} \tag{4.142}$$

其中 $\mathbf{I}$ 为和 $\mathbf{H}$ 具有相同维数的单位矩阵，λ 是*正则或负荷参数*，用来强制矩阵 $(\mathbf{H}+\lambda\mathbf{I})$ 为正定的，并且在计算过程中是完全充分条件的。还需要注意的是式(4.142)的调整量 $\Delta\mathbf{w}$ 是由式(4.115)定义的公式的小的修正。

有了这样的背景，考虑具有一个单一输出神经元的多层感知器。网络是通过最小化如下的代价函数来训练的：

$$\mathscr{E}_{av}(\mathbf{w}) = \frac{1}{2N}\sum_{i=1}^{N}[d(i) - F(\mathbf{x}(i);\mathbf{w})]^2 \tag{4.143}$$

其中 $\{\mathbf{x}(i), d(i)\}_{i=1}^{N}$ 是训练样本，$F(\mathbf{x}(i);\mathbf{w})$ 是网络实现的逼近函数；网络的突触权值按某种顺序排列形成权值向量 $\mathbf{w}$。代价函数 $\mathscr{E}_{av}(\mathbf{w})$ 的梯度和 Hessian 矩阵分别定义为：

$$\mathbf{g}(\mathbf{w}) = \frac{\partial \mathscr{E}_{av}(\mathbf{w})}{\partial \mathbf{w}} = -\frac{1}{N}\sum_{i=1}^{N}[d(i) - F(\mathbf{x}(i);\mathbf{w})]\frac{\partial F(\mathbf{x}(i);\mathbf{w})}{\partial \mathbf{w}} \tag{4.144}$$

和

$$\mathbf{H}(\mathbf{w}) = \frac{\partial^2 \mathscr{E}_{av}(\mathbf{w})}{\partial \mathbf{w}^2} = \frac{1}{N}\sum_{i=1}^{N}\left[\frac{\partial F(\mathbf{x}(i);\mathbf{w})}{\partial \mathbf{w}}\right]\left[\frac{\partial F(\mathbf{x}(i);\mathbf{w})}{\partial \mathbf{w}}\right]^T - \frac{1}{N}\sum_{i=1}^{N}[d(i) - F(\mathbf{x}(i);\mathbf{w})]\frac{\partial^2 F(\mathbf{x}(i);\mathbf{w})}{\partial \mathbf{w}^2} \tag{4.145}$$

因此，将式(4.144)和式(4.145)代入到式(4.142)，Levenberg-Marquardt 算法每一步迭代的期望调整量 $\Delta\mathbf{w}$ 就得到了计算。

然而，从实际的角度来看，式(4.145)的计算复杂度是需要考虑的，尤其是当权值向量 $\mathbf{w}$ 的维数高的情况下；这里的计算困难是由 Hessian 矩阵 $\mathbf{H}(\mathbf{w})$ 的复杂性引起的。为了减轻这一困难，推荐方法是忽略式(4.145)右边第二项，因此简单地用下式逼近 Hessian 矩阵：

$$\mathbf{H}(\mathbf{w}) \approx \frac{1}{N}\sum_{i=1}^{N}\left[\frac{\partial F(\mathbf{x}(i);\mathbf{w})}{\partial \mathbf{w}}\right]\left[\frac{\partial F(\mathbf{x}(i);\mathbf{w})}{\partial \mathbf{w}}\right]^T \tag{4.146}$$

这个逼近可以看成是偏导数 $\partial F(\mathbf{w},\mathbf{x}(i))/\partial \mathbf{w}$ 对其自身的外积在训练样本上的平均；相应地，这可以被称为 Hessian 矩阵的外积逼近。这一逼近的使用在 Levenberg-Marquardt 算法运行于局部或全局最小点附近时得到了证明。

显然，基于式(4.144)的梯度向量和式(4.146)的 Hessian 矩阵的 Levenberg-Marquardt 算法的逼近版本，是非常适用于非线性最小二乘估计问题最优化的一阶方法。而且，由于这些方程都包含了在训练样本上的平均的事实，算法属于批量方式。

正则参数 λ 在 Levenberg-Marquardt 算法工作过程中起决定性作用。如果设 λ 等于 0，则式(4.142)的公式简化为牛顿法。另一方面，如果给 λ 分配一个大的值使得 $\lambda\mathbf{I}$ 远大于 Hessian 矩阵 $\mathbf{H}$ 元素的值，Levenberg-Marquardt 算法从效果上作为梯度下降法起作用。根据这两个观察，在算法的每一步迭代中，分配给 λ 的值就需要恰好足够大到保持和矩阵 $(\mathbf{H}+\lambda\mathbf{I})$ 的正定形式。具体来说，对于 λ 的选择我们推荐如下的 Marquardt 方法 (Press 等，1988)：

1. 在迭代步 $n-1$ 计算 $\mathscr{E}_{av}(\mathbf{w})$。
2. 选择一个适度的 λ 值，比方说 $\lambda = 10^{-3}$。
3. 解方程(4.142)得到迭代步 n 的调整量 $\Delta\mathbf{w}$ 和评价 $\mathscr{E}_{av}(\mathbf{w}+\Delta\mathbf{w})$。
4. 如果 $\mathscr{E}_{av}(\mathbf{w}+\Delta\mathbf{w}) \geqslant \mathscr{E}_{av}(\mathbf{w})$，通过一个因子 10(或者任意其他大因子) 来增加 λ，转第 3 步。
5. 另一方面，如果 $\mathscr{E}_{av}(\mathbf{w}+\Delta\mathbf{w}) < \mathscr{E}_{av}(\mathbf{w})$，通过因子 10 降低 λ，更新试验解 $\mathbf{w} \rightarrow \mathbf{w}+\Delta\mathbf{w}$，转第 3 步。

由于明显的原因，终止迭代过程的规则是必需的。Press 等 (1998) 指出，通过小量增加改变 $\mathscr{E}_{av}(\mathbf{w})$ 的参数向量 $\mathbf{w}$ 的调整从来都不是统计有意义的。因此可以利用这一具深刻见解的评论来作为终止规则的基础。

作为最后的评论：为了在算法的每一步评估偏导数 $\partial F(\mathbf{x};\ \mathbf{w})/\partial \mathbf{w}$，可以利用 4.8 节描述的反向传播的方式。

在线学习的二阶随机梯度下降

到目前为止，本节集中于批量学习的二阶最优技术。从这里开始，我们将注意力转移到在

线学习的二阶随机梯度下降方法来。尽管这两类技术是根本不同的，但它们具有一个共同的目的：

代价函数的 Hessian 矩阵（曲率）所包含的二阶信息被用来提高监督学习算法的性能。

对于第 4.10 节所考虑的最优退火在线学习算法性能扩展的一个简单途径是将式(4.60)中的学习率参数 $\eta(n)$用 Hessian 矩阵 $\mathbf{H}$ 的逆的尺度来代替，如下所示：

$$\underbrace{\hat{\mathbf{w}}(n+1)}_{\text{更新估计}} = \underbrace{\hat{\mathbf{w}}(n)}_{\text{老的估计}} - \underbrace{\frac{1}{n}\mathbf{H}^{-1}}_{\substack{\text{Hessian矩阵}\mathbf{H}\\ \text{的逆的退火}}} \underbrace{\mathbf{g}(\mathbf{x}(n+1),\mathbf{d}(n+1);\hat{\mathbf{w}}(n))}_{\text{梯度向量}} \tag{4.147}$$

将 $\eta(n)$用新的项$\frac{1}{n}\mathbf{H}^{-1}$来代替是为了加速最优退火方式下在线算法的收敛速度。这里假设 Hessian 矩阵 $\mathbf{H}$ 是先验已知的，其逆 $\mathbf{H}^{-1}$因此可以预计算。

“没有免费的午餐”，加速收敛所付出的代价总结如下（Bottou，2007）：

1）在式(4.60)的随机梯度下降中，算法每步迭代的计算花费是 $O(W)$，这里 W 是被估计的权值向量 $\mathbf{w}$ 的维数，而相应地式(4.147)中二阶随机梯度下降算法每步迭代的计算代价是 $O(w^2)$。

2）对于由式(4.147)算法处理的每个训练样本 $(\mathbf{x},\mathbf{d})$，算法需要 $W\times 1$ 的梯度向量 $\mathbf{g}$ 和 $W\times W$ 的逆矩阵 $\mathbf{H}^{-1}$相乘，并需要存储乘积。

3）在通常情况下，当训练样本中存在某种形式的稀疏性时，自然的步骤是开发这种稀疏性以达到改善算法性能的目的。遗憾的是，Hessian 矩阵 $\mathbf{H}$ 是一个典型的全矩阵因此不是稀疏的，这就排除了开发训练样本稀疏性的可能。

为了克服这些局限性，我们可以求助于如下逼近过程中的一种：

1）*对角逼近*：(Becker and LeCun，1989)。在这一过程中，Hessian 矩阵仅有对角元素被保留，这意味着逆矩阵 $\mathbf{H}^{-1}$同样也是对角矩阵。由矩阵理论可知，矩阵乘积 $\mathbf{H}^{-1}\mathbf{g}$将由形式为 $h_{ii}^{-1}g_i$ 的项的和组成，其中 h_{ii}是 Hessian 矩阵 $\mathbf{H}$ 的第 i 个对角元素，g_i 是相应的梯度 $\mathbf{g}$ 的元素，$i=1,2,\cdots,W$。梯度向量 $\mathbf{g}$ 对权值为线性的，这就意味着逼近二阶在线学习算法的计算复杂度是 $O(W)$。

2）*低秩逼近*：(LeCun 等，1998)。根据定义，矩阵的秩等于矩阵的线性无关列的个数。给定一个 Hessian 矩阵 $\mathbf{H}$，*奇异值分解*（SVD）为 Hessian 矩阵 $\mathbf{H}$ 的低秩逼近提供了一个重要程序。令 $\mathbf{H}$ 的秩记为 $\mathbf{p}$，$\mathbf{H}$ 的秩 r 逼近被记为 $\mathbf{H}_r$，其中 $r<p$。在 Hessian 矩阵和其逼近之间的平方误差通过 Frobenius *范数*来定义：

$$e^2 = \text{tr}[(\mathbf{H}-\mathbf{H}_r)^T(\mathbf{H}-\mathbf{H}_r)] \tag{4.148}$$

其中 tr[·]表示方括号中的方阵的迹（即对角元素的和）。对矩阵 $\mathbf{H}$ 和 $\mathbf{H}_r$ 作 SVD，我们写为：

$$\mathbf{H} = \mathbf{V}\sum\mathbf{U}^T \tag{4.149}$$

和

$$\mathbf{H}_r = \mathbf{V}\sum\nolimits_r\mathbf{U}^T \tag{4.150}$$

其中正交矩阵 $\mathbf{U}$ 和 $\mathbf{V}$ 定义了相应的*通常右和左奇异向量*，矩形矩阵

$$\sum\nolimits_r = \text{diag}[\lambda_1,\lambda_2,\cdots,\lambda_r,0,\cdots,0] \tag{4.151}$$

定义了低秩逼近 $\mathbf{H}_r$ 的奇异值。新的方阵

$$\mathbf{H}_r = \mathbf{U}\sum\nolimits_r\mathbf{V}^T \tag{4.152}$$

提供了对 Hessian 矩阵 $\mathbf{H}$ 的*最小二乘秩 r 逼近*（Scharf，1991）。相应地，在式(4.147)的在线学习算法中利用新矩阵 $\mathbf{H}_r$ 来代替 Hessian 矩阵 $\mathbf{H}$ 将算法的计算复杂度降低到了 $O(W)$和 $O(W^2)$之间的

某个地方。

3）BFGS 逼近：（Schraudolph 等，2007）。正如本节前面所指出的那样，BFGS 被认为是拟牛顿法的最好形式。在 Schraudolph 等的 2007 年的论文中，BFGS 被修改为全记忆和有限记忆版本，使其对于梯度的随机逼近变得可用。这一修正算法为在线凸优化提供了一种快速、可扩缩的、随机拟牛顿过程。在 Yu 等（2008）中，BSGF 拟牛顿法和其有限记忆变形被扩展用来处理非光滑凸目标函数。

4.17　卷积网络

到目前为止，我们都在考虑多层感知器算法设计和相关的问题。本节重点讨论多层感知器本身的结构布局问题。特别地，我们描述一类特定的通称为*卷积网络*的多层感知器，它对于模式分类非常适合。这些网络的提出所隐含的思想受到了神经生物学的启发，可以回溯到 Hubel and Wiesel(1962，1977）的开创性研究，该研究是关于猫的视觉皮质上局部传感和方位选择神经元的。

一个*卷积网络*是为识别二维形状而特殊设计的一个多层感知器，这种二维形状对平移、比例缩放、倾斜或者其他形式的变形具有高度不变性。这个艰巨的任务是通过如下网络在监督方式下学会的，网络的结构包括如下形式的*约束*（LeCun and Bengio，2003）：

1. *特征提取*。每一个神经元从上一层的局部*接受域*得到突触输入，因而迫使它提取局部特征。一旦一个特征被提取出来，只要它相对于其他特征的位置被近似地保留下来，它的精确位置就变得没有那么重要了。

2. *特征映射*。网络的每一个计算层都是由多个特征映射组成的，每个特征映射都是平面形式的，平面中单独的神经元在约束下共享相同的突触权值集。这种结构约束的第二种形式具有如下的有益效果：

- *平移不变性*，强迫特征映射的执行使用具有小尺度核的卷积，再接着用一个 sigmoid 函数。
- *自由参数数量的缩减*，通过权值共享实现。

3. *子抽样*。每个卷积层跟着一个实现局部平均和子抽样的计算层，由此特征映射的分辨率降低。这种操作具有使特征映射的输出对平移和其他形式的变形的敏感度下降的作用。

注意，在一个卷积网络所有层中的所有权值都是通过训练来学习的。此外，网络自动地学习提取它自身的特征。

图 4.23 表明由一个输入层和四个隐藏层与一个输出层组成的卷积网络的体系结构布局。这个网络被设计用于实现图像处理（例如手写体的识别）。输入层由 28×28 个感知节点组成，接收已经近似处于中心位置和在大小上规整化的不同字符的图像。然后，计算流程在卷积和子抽样之间交替，如下所述：

1. 第一隐藏层进行卷积。它由四个特征映射组成，每个特征映射由 24×24 个神经元组成。每个神经元指定一个 5×5 的接受域；

2. 第二隐藏层实现子抽样和局部平均。它同样由四个特征映射组成，但其每个特征映射由 12×12 个神经元组成。每个神经元具有一个 2×2 的接受域，一个可训练系数，一个可训练偏置和一个 sigmoid 激活函数。可训练系数和偏置控制神经元的操作点；例如，如果系数很小，该神经元以拟线性方式操作。

3. 第三隐藏层进行第二次卷积。它由 12 个特征映射组成，每个特征映射由 8×8 个神经元组成。该隐藏层中的每个神经元可能具有和上一个隐藏层几个特征映射相连的突触连接。否则，它以第一个卷积层相似的方式操作。

4. 第四个隐藏层进行第二次子抽样和局部平均计算。它由 12 个特征映射组成，但每个特

征映射由 4×4 个神经元组成。否则它以第一次抽样相似的方式操作。

5. 输出层实现卷积的最后阶段，它由 26 个神经元组成，每个神经元指定为 26 个可能的字符中的一个，跟前面一样，每个神经元指定一个 4×4 的接受域。

相继的计算层在卷积和抽样之间的连续交替，我们得到一个“双尖塔”的效果。也就是在每个卷积或抽样层，随着空间分辨率下降，与相应的前一层相比特征映射的数量增加。卷积之后进行子抽样的思想是受到 Hubel 和 Wiesel(1962) 首先提出的“简单的”细胞后面跟着“复杂的”细胞[14]的概念启发而产生的。

图 4.23 所示的多层感知器包含近似 100 000 个突触连接，但只有大约 2 600 个自由参数。自由参数在数量上显著减少是通过权值共享实现的。机器学习的能力因而下降，这又提高了它的泛化能力。甚至更值得注意的事实是对自由参数的调整通过反向传播学习的随机形式来实现。

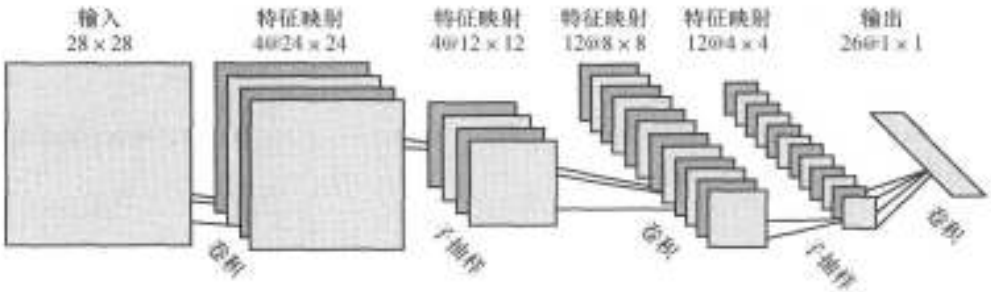


图 4.23 用于图像处理如手写体识别的卷积网络（经 MIT 出版社授权）

另一个显著的特点是使用权值共享使得以并行形式实现卷积网络成为可能。这是卷积网络对完全连接的多层感知器而言的另一个优点。

从图 4.23 的卷积网络中收获了以下两方面经验。首先，通过结合当前任务的先验知识约束其设计，一个可调整大小的多层感知器能够学习一个复杂的、高维的和非线性的映射。其次，突触权值和偏置水平可以周而复始地执行通过训练集的简单反向传播算法进行学习。

4.18 非线性滤波

以多层感知器为例的静态神经网络的原型应用是结构化模式识别。在所考虑的应用范围内，本章所讲述的素材集中于结构化模式识别。相反，时序模式识别或非线性滤波要求对随时间演化的模式进行处理，对特定时刻的响应不仅依赖于输入的当前值，还依赖于以前的值。简单说，时间是有序的量，构成了时序模式识别任务中学习过程的重要成分。

对于动态神经网络来说，它必须以一种或另一种形式给定短期记忆。完成这一修改的一个简单途径是利用时间延迟，时间延迟可以在网络内部的突触层或者外部地在网络的输入层上执行。确实，神经网络中时间延迟的使用是受神经生物学启发的，因为众所周知在大脑中信号延迟是无所不在的，且在神经生物信息处理中起着重要作用（Braitenberg，1967，1977，1986；Miller，1987）。时间可以通过如下的两种基本途径来嵌入神经网络的运行中：

- *隐式表示*。时间是通过其作用于信号处理的效果以一种隐含方式来表示的。例如，在神经网络的数字执行中，输入信号经过一致采样，和网络输入层相连的每个神经元的突触权值序列和不同的输入样本序列作卷积（convolved）。这样，输入信号的时间结构嵌入在网络的空间结构里。
- *显式表示*。在网络结构内时间由它自身的特定表示给出。例如，蝙蝠的回声定位系统

是通过发射短的频率调制（FM）信号，使得对于每个限制在 FM 扫描期间的很短的一个时间段的频道维持相同的强度等级。被一组听觉接收器编码的几个不同频率之间的多种比较是为了抽取目标物的准确的距离信息（Suga and Kanwal，1995）。当从目标的回声在经一段未知时延以后被接收时，一个具有匹配的延迟线的神经元（在听觉系统）进行响应，从而提供目标范围的估计值。

本节我们关心时间的隐式表达，这由通过外部方式对一个静态神经网络（如多层感知器）提供动态属性而得到。

图 4.24 显示了非线性滤波器的框图，它由两个子系统的层叠连接组成：短期记忆和静态神经网络（如多层感知器）。这一结构对于处理规则提供了明确的分割：静态网络对应于非线性，记忆对应于时间。具体来说，假设给定了具有大小为 m 的输入层的多层感知器。那么，在一个对应的途径下，记忆是一个单输入多输出（SIMO）的结构，提供对模拟神经网络的输入信号的 m 个不同延迟版本。

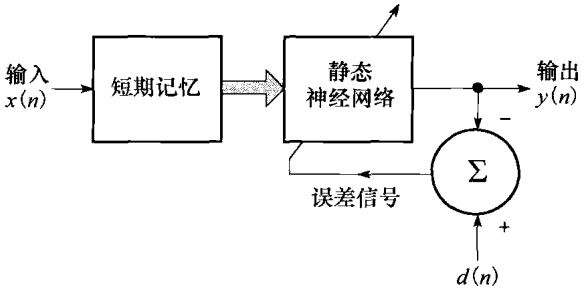


图 4.24　建立在静态神经网络上的非线性滤波

短期记忆结构

图 4.25 显示了离散时间记忆结构的框图，它由 p 个等同片断层叠连接。每一个片断由一个脉冲响应来描述，记为 $h(n)$，其中 n 记为离散时间。片断数 p 称为记忆的阶。相应地，由记忆提供的输出终端个数（即抽头（tap））为 $p+1$，这包含了从输入到输出的直接连接。因此，用 m 记静态神经网络输入层的大小，我们有

$$m = p + 1$$

记忆的每一个延迟片断的脉冲响应满足两个性质：

- 因果关系，这意味着对于 $n<0$ 有 $h(n)$为零。
- 归一性，这意味着 $\sum_{n=0}^{\infty} |h(n)| = 1$

在这个基础上，我们将 $h(n)$称为离散时间记忆的产生核。

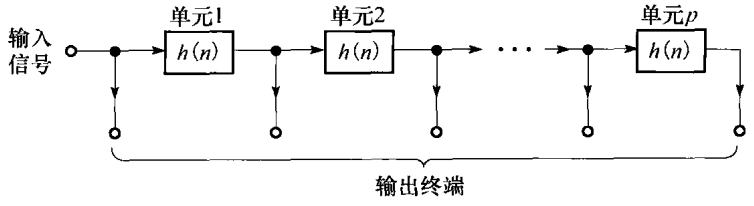


图 4.25　p 阶一般抽头延迟线记忆

可以用深度和分辨率来衡量记忆结构的属性（deVries and Principe，1992）。设记忆结构中总的脉冲响应为 $h_{\text{overall}}(n)$。具有 p 个记忆片断，因此 $h_{\text{overall}}(n)$定义为 $h(n)$的 p 个逐次卷积。相应地，记忆深度记为 D，定义为 $h_{\text{overall}}(n)$的第一时间矩，表示为

$$D = \sum_{n=0}^{\infty} n h_{\text{overall}}(n) \tag{4.153}$$

一个低深度 D 的记忆只能将信息内容保持较短的时间，而高深度的记忆则能保持较长时间。记忆分辨率记为 R，指的是每个单位时间内记忆结构中的抽头数目。一个高分辨率的记忆结构能将输入的序列信息保持在精确的层次上，而低分辨率的记忆结构只能保持在粗糙的层次

上。对于固定的记忆阶 p，记忆深度 D 和记忆分辨率 R 的乘积是一个常量并证明等于 p。

自然，选择不同的产生核 $h(n)$会产生不同的深度 D 和记忆分辨率 R，这可以用下面两个记忆结构来说明。

1. *抽头延迟线记忆*（tapped-delay-line memory），对它而言，产生核被简单定义为单位脉冲 $\delta(n)$，即

$$h(n) = \delta(n) = \begin{cases} 1, n = 0 \\ 0, n \neq 0 \end{cases} \tag{4.154}$$

对应地，总的脉冲响应是

$$h_{\text{overall}}(n) = \delta(n-p) = \begin{cases} 1, n = p \\ 0, n \neq p \end{cases} \tag{4.155}$$

将式(4.155)代入式(4.153)，产生记忆深度 $D=p$，这一点直观上是满足的。而且，每个时间单元内只有一个抽头，因此，分辨率 $R=1$，深度－分辨率积就等于 p。

2. Gamma 记忆，对于它产生核被定义为

$$h(n) = \mu(1-\mu)^{n-1}, n \geqslant 1 \tag{4.156}$$

其中 μ 是一个可调参数（deVries and Principe，1992）。为了 $h(n)$能够收敛（即为了短期记忆能够稳定），我们需要

$$0 < \mu < 2$$

相应地，Gamma 记忆的完整的脉冲响应为

$$h_{\text{overall}}(n) = \binom{n-1}{p-1} \mu^p (1-\mu)^{n-p}, n \geqslant p \tag{4.157}$$

其中$\binom{\cdot}{\cdot}$是一个二项式系数。对于变化的 p 的脉冲响应 $h_{\text{overall}}(n)$表达了 gamma 函数的被积函数的离散版本（deVries and Principe，1992）——因此命名为“gamma 记忆”。图 4.26 画出了对 μ 归一后脉冲响应 $h_{\text{overall}}(n)$对于变化的记忆阶的图，$\mu=0.7$。还要注意的是时间轴已经被参数 μ 所标度，这种标度具有将 $h_{\text{overall}}(n)$的峰值定位在 $n=p-1$ 的效果。

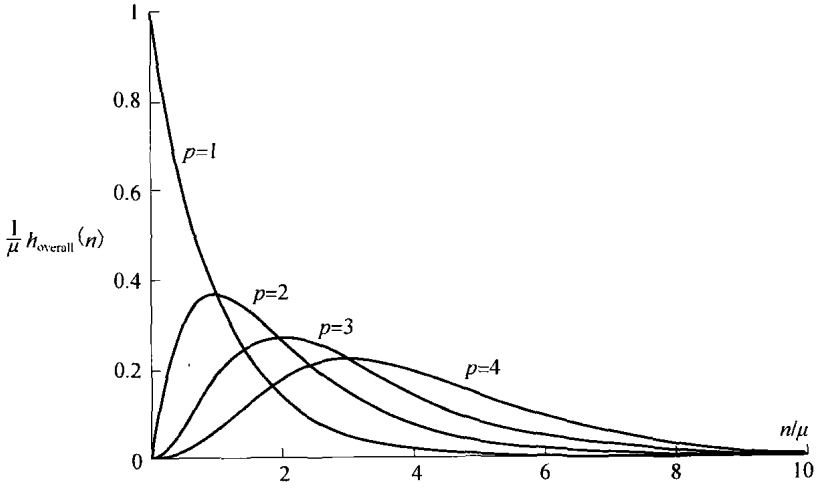


图 4.26　对 $p = 1,2,3,4$ 和 $\mu=0.7$ 的 gamma 记忆的脉冲响应族

已经证明 Gamma 记忆的深度为 p/μ，分辨率为 μ，再一次深度－分辨率的乘积为 p。相应地，通过选择小于单位 1 的 μ 值，Gamma 记忆的深度有所提高，但是牺牲了分辨率。对于特例 $\mu=1$，Gamma 记忆衰减为通常的抽头延迟线记忆，那里每个片断简单地由一个单位时间延

迟算子组成。

通用短视映射定理

图 4.24 中的非线性滤波器可以推广为图 4.27 所示的滤波器。这个一般的动态结构包含两个功能模块。标号为 $\{h_j\}_{j=1}^{L}$ 的模块表示时域的多重卷积，即，一个并行运行的线性滤波器组。h_j 是从一个较大的实值核集合中抽取出来的，每一个都代表一个线性滤波器的脉冲响应。块标号为$\mathcal{N}$的模块表示静态的（即无记忆的）非线性前馈网络，如多层感知器。图 4.27 中的结构是一个通用动态映射器（universal dynamic mapper）。在 Sandberg and Xu(1997a) 中证明对于任何平移不变的短视映射（myopic map），在适度的条件下利用图 4.27 描绘的结构能够以任意精度一致逼近。要求一个映射为短视的等价于“一致衰减记忆”；这里假设映射是因果的（causal），这意味着只有在 $n=0$ 时应用输入信号，才能在时刻 $n\geqslant 0$ 由映射产生输出信号。通过“平移不变”，我们是指如果 $y(n)$是由输入 $x(n)$产生的映射的输出，那么对于平移输入 $x(n-n_0)$产生的映射的输出就是 $y(n-n_0)$，这里时间位移 n_0 是一个整数。Sandberg and Xu(1997b) 中进一步证明了对单变量的、平移不变的、因果的和一致衰减的记忆映射，存在一个 Gamma 记忆和静态神经网络，它们的组合能够以任意精度一致逼近该映射。

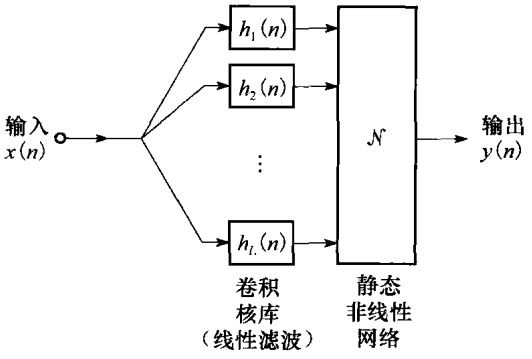


图 4.27　通用短视映射定理的一般结构

现在可以正式地将通用短视映射定理[15]描述如下（Sandberg and Xu，1997a，1997b）：

任何平移不变的短视动态映射可以由含有两个功能块的结构任意地一致逼近：一组线性滤波器馈给一个静态神经网络。

正如已经指出的那样，多层感知器可以作为静态网络的规则。值得注意的是当输入输出信号是固定变量数的函数时定理得到保持，例如在图像处理中。

定理的实际含义

这个定理具有深厚的实际含义：

1. 这个定理为 NETtalk 提供了证明。NETtalk 是将英语语音转化为音素的大规模并行分布式网络的第一个示范。音素（phoneme）是一个基本的语言单位（Sejnowski and Rosenberg，1987）。图 4.28 显示了一个 NETtalk 系统的示意图，它建立在一个多层感知器的基础上，输入层有 203 个感知（源）节点，隐藏层有 80 个神经元，输出层有 26 个神经元。所有神经元都使用 sigmoid(logistic) 型激活函数。这个网络的突触连接有 18 629 个，每个神经元包含有可变的阈值。阈值是偏置的负值。这个网络使用标准的反向传播算法进行训练。这个网络有 7 组输入层节点。每组对输入文本的 1 个字母进行编码。从而每次将 7 个字母组成的串呈现给输入层。训练过程的期望响应是和 7 个字母窗口中央的一个（即第 4 个）相联系的正确音素。另外 6 个字母（在中间字母两边各 3 个）对网络的每一个决策来说提供部分的上下文。通过一个字母接着一个字母的方式使文本通过窗口。在处理的每一步中，网络都计算一个音素，每学完一个单词后，网络的突触权值就根据计算出的发音与正确的发音的接近程度进行调整。NETtalk 的性能显示出和观察到的人类表现的相似之处，可总结为以下几点（Sejnowski and Rosenberg，1987）：

- 训练遵守有力的规律（power law）。

- 网络学习的单词越多，它的泛化性能和对新词正确发音的性能就越好。
- 当网络的突触连接被破坏时，网络性能的下降非常缓慢。
- 在网络遭到破坏以后，进行重新学习，学习的速度要比原始训练快得多。

NETtalk 出色地说明了学习的很多方面的微小细节，在开始的时候，在它的输入模式中具有大量“先天”的知识并且通过实践逐渐获得将英语语音转化为音素的能力。

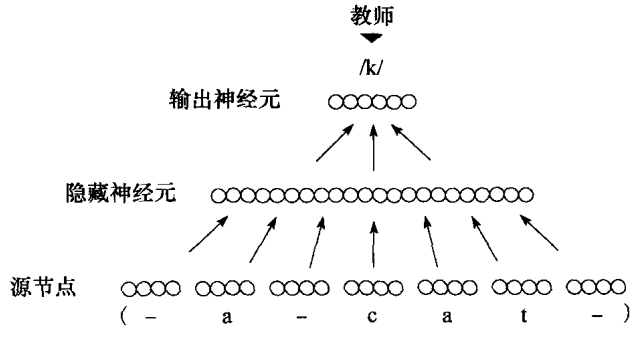


图 4.28　NETtalk 网络结构的示意图

2. 通用短视定理为更复杂的非线性系统模型的设计建立框架。在图 4.27 结构前端的多个卷积可以使用具有有限冲激响应（FIR）或者无限冲激响应（IIR）的线性滤波器来实现。更重要的是，图 4.27 的结构是*固有稳定的*（inherently stable），因此线性滤波器自身是稳定的。因此，在建立稳定动态系统时对于如何处理短期记忆和无记忆非线性性，我们对它们的作用有清晰的分工。

3. 给定稳定的时间序列 $x(1),x(2),\cdots,x(n)$，通过设 $y(n)=x(n+1)$，可以利用图 4.27 的通用短视映射结构来建造潜在的非线性物理规律的预测模型，该模型用于时间序列的生成，而不管规律是多么复杂。事实上，未来的样本 $x(n+1)$ 起着期望响应的作用。当用一个多层感知器作为图 4.27 的静态网络来实现这一应用时，为网络的输出单元提供线性神经元是明智的。这将保证在预测模型的动态范围上没有振幅的局限。

4.19　小规模和大规模学习问题

在本章和本书其他地方，我们已经多次提及小规模和大规模学习问题。然而，我们没有严格地详细说明这两类监督学习的意义。本节的目的是突出将两者区分开的统计和计算方面的论点。

结构风险最小化

监督学习的可行性依赖于下面的关键问题：

由 N 个独立同分布的样本

$$(\mathbf{x}_1,d_1),(\mathbf{x}_2,d_2),\cdots,(\mathbf{x}_N,d_N)$$

组成的训练样本是否包含了构造具有良好泛化性能的机器学习的足够信息？

这一基本问题的答案在于 Vapnik(1982，1998）所描述的*结构风险最小化*（structural risk minimization）方法。

为了讲述这一方法的意义，令产生训练样本的自然源或者环境表示为非线性回归模型

$$d=f(\mathbf{x})+\varepsilon \tag{4.158}$$

其中，和第 2 章中引入的术语一样，向量 $\mathbf{x}$ 是回归量，标量 d 是响应，ε 是解释（模型）误差。函数 f 是未知的，目标是估计它。为了实现这个估计，我们定义期望风险（即总体-平均代价函数）为：

$$J_{\text{actual}}(f) = \mathbb{E}_{\mathbf{x},d}\left[\frac{1}{2}(d - f(\mathbf{x}))^2\right] \tag{4.159}$$

其中期望是对于回归量一响应对（$\mathbf{x}$，d）联合完成的。在第 5 章，我们将证明条件均值估计

$$\hat{f}^* = \mathbb{E}[d|\mathbf{x}] \tag{4.160}$$

是代价函数 $J_{\text{actual}}(f)$的最小点。相应地，将式(4.159)定义的代价函数的最小值写为 $J_{\text{actual}}(\hat{f}^*)$；它可作为能达到的绝对最优（absolute optimum）。

决定条件均值估计 $\hat{f}^*$ 需要回归量 $\mathbf{x}$ 和响应 d 的潜在的联合概率分布知识。然而，我们发现这一知识是无法提供的。为了解决这一困难，我们转向机器学习来寻找可行的解。例如，假设选择单层多层感知器来做机器学习。令函数 $F(\mathbf{x};\mathbf{w})$记为神经网络的输入输出关系，神经网络的参数是权值向量 $\mathbf{w}$。然后通过设

$$f(\mathbf{x}) = F(\mathbf{x};\mathbf{w}) \tag{4.161}$$

来做第一个逼近（first approximation）。

相应地，将模型的代价函数公式化为：

$$J(\mathbf{w}) = \mathbb{E}_{\mathbf{x},d}\left[\frac{1}{2}(d - F(\mathbf{x};\mathbf{w}))^2\right] \tag{4.162}$$

其中，如前所述，期望是联合地在对（$\mathbf{x}$,d）上完成的。这第二个代价函数和属于原始源的代价函数 $J_{\text{actual}}(f)$本质上是不同的——因此对它们使用了不同的记号。将式(4.161)的等式应用于神经网络，我们从效果上限制了逼近函数 $F(\mathbf{x};\mathbf{w})$ 的选择。

令

$$\hat{\mathbf{w}}^* = \arg\min_{\mathbf{w}} J(\mathbf{w}) \tag{4.163}$$

为代价函数 $J(\mathbf{w})$的最小值。然而，实际上甚至即使我们能找到最小值$\hat{\mathbf{w}}^*$，有很大可能结果代价函数 $J(\hat{\mathbf{w}}^*)$将比最小化代价函数 $J_{\text{actual}}(\hat{f}^*)$更坏，我们可以写为：

$$J(\hat{\mathbf{w}}^*) > J_{\text{actual}}(\hat{f}^*) \tag{4.164}$$

遗憾的是，我们仍然面对如前所述的同样的实际问题，即不知道（$\mathbf{x}$,d）的内在联合概率分布。为了缓和这一困难，我们通过利用实验风险（即时间平均能量函数）来做第二个逼近(second approximation)

$$\mathscr{E}_{\text{av}}(N;\mathbf{w}) = \frac{1}{2N}\sum_{n=1}^{N}(d(n) - F(\mathbf{x}(n);\mathbf{w}))^2 \tag{4.165}$$

其最小点定义为

$$\hat{\mathbf{w}}_N = \arg\min_{\mathbf{w}} \mathscr{E}_{\text{av}}(N;\mathbf{w}) \tag{4.166}$$

显然，最小化代价函数 $J(\hat{\mathbf{w}}_N)$不小于 $J(\hat{\mathbf{w}}^*)$。事实上，有很大可能发现：

$$J(\hat{\mathbf{w}}_N) > J(\hat{\mathbf{w}}^*) > J_{\text{actual}}(\hat{f}^*) \tag{4.167}$$

有了已经做出的两个逼近，我们可以惊讶于为什么我们需要精确计算最小值$\hat{\mathbf{w}}_N$。在解决这一问题之前，让我们检查一下当示例的多层感知器的隐藏层大小变大时会发生什么情况。

回顾一下第 4.12 节，多层感知器是未知函数 $f(\mathbf{x})$的通用逼近器。从理论上，当隐藏层大小足够大时，参数函数 $F(\mathbf{x};\mathbf{w})$能以任意期望精度逼近未知函数 $f(\mathbf{x})$。这反过来意味着 $J(\hat{\mathbf{w}}^*)$变得接近于绝对最优 $J_{\text{actual}}(\hat{f}^*)$。然而，通过放大隐藏层大小，我们可能连累多层感知器的泛化能力。特别地，作为放大隐藏层的结构，误差（$J(\hat{\mathbf{w}}^*) - J_{\text{actual}}(\hat{f}^*)$）有可能增加。除非训练样本大小相应地增加。刚刚讨论的问题是 Vapnik 结构风险最小化的本质内容，它证明了“逼近一估计折中”。

为了详细说明这种折中，令过剩误差（$J(\hat{\mathbf{w}}_N) - J_{\text{actual}}(\hat{f}^*)$）分解为如下两项：

$$\underbrace{J(\hat{\mathbf{w}}_N)-J_{\text{actual}}(\hat{f}^*)}_{\text{过剩误差}}=\underbrace{J(\hat{\mathbf{w}}_N)-J(\hat{\mathbf{w}}^*)}_{\text{逼近误差}}+\underbrace{J(\hat{\mathbf{w}}^*)-J_{\text{actual}}(\hat{f}^*)}_{\text{估计误差}} \tag{4.168}$$

在这一经典的误差分解中，以下几点是值得注意的：

①逼近误差提供了一种性能损失的度量，该损失是使用了预设大小为 N 的训练样本而导致的。而且，由于 $\hat{\mathbf{w}}_N$ 依赖于训练样本，逼近误差就和网络训练的评估相关。

②估计误差提供了一种性能损失的度量，该损失是选择由逼近函数 $F(\mathbf{x},\mathbf{w})$ 刻画的模型导致的。而且，由于 $\hat{f}^*$ 是给定回归量 $\mathbf{x}$ 时响应 d 的条件估计，因此估计误差和网络测试的评估相关。

在 Vapnik 的理论框架中，逼近和估计误差是通过 VC 维数来公式化的，VC 维数通常记为 h。这一新的参数，是 Vapnik-Chervonenkis dimension 的缩写（Vapnik and Chervonenkis, 1971），是关于用机器学习实现的二值分类函数族的容量或者表达能力的测量[16]。对于单层多层感知器的例子，VC 维数是由隐藏层的大小决定的；隐藏层越大，VC 维数 h 也越大。

为了将 Vapnik 理论在实际背景下应用，考虑一族嵌套逼近网络函数

$$\mathcal{F}_k=\{F(\mathbf{x};\mathbf{w})(\mathbf{w}\in\mathcal{W}_k)\},\qquad k=1,2,\cdots,K \tag{4.169}$$

使得我们有

$$\mathcal{F}_1\subset\mathcal{F}_2\subset\cdots\subset\mathcal{F}_K$$

其中记号⊂意为“包含”。相应地，$\mathcal{F}_k$ 的各个子集的 VC 维数满足条件

$$h_1<h_2<\cdots<h_K$$

从效果上，$\mathcal{F}_k$ 的大小是机器容量的测量。从现在开始，我们利用式(4.169)的定义来代替 VC 维数。

图 4.29 是逼近和估计误差关于逼近网络函数族 $\mathcal{F}_k$ 的大小 K 的图。对于单层多层感知器的例子，隐藏层的最优大小是由逼近误差和估计误差假设具有共同值的点来决定的。在这一最优条件达到之前，学习问题是超定的（overdetermined），这意味着机器容量对于包含在训练样本中的细节数量而言太小。在最小点之外，学习问题称为欠定的（underdetermined），这意味着对于训练样本而言机器容量太大。

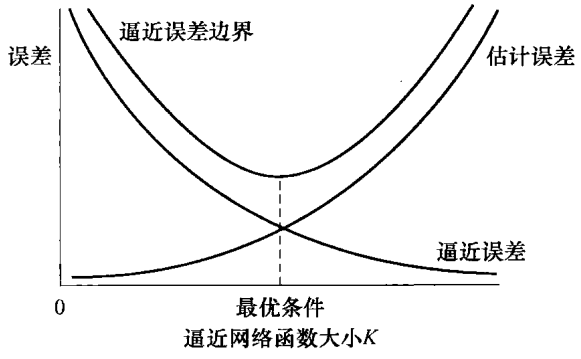


图 4.29　随大小 K 的变化逼近和估计误差的变化

计算考虑

神经网络模型（例如单层多层感知器）必须是可控变量，使得它能够被自由地调整以达到对从未出现过的数据的最好测试性能。另一个可控变量是用于训练的样本个数。为了增加监督训练过程的实际真实性，Bottou(2007) 通过考虑一个新的可控变量来介绍计算代价。这个新的可控变量就是*最优精确度*。

在实际中，计算最小值 $\hat{\mathbf{w}}_N$ 的任务可能会产生很大开销。而且，在满意的网络设计讨论的进程中，我们通常做多个逼近。然后，假设我们选定一个由权值向量 $\tilde{\mathbf{w}}_N$ 刻画的网络模型，它和 $\hat{\mathbf{w}}_N$ 不同；这样做，我们将给出第三个，也是最后一个逼近。例如，由于计算时间的限制，在线学习算法可以在收敛远未到达之前终止。在多数情形下，$\tilde{\mathbf{w}}_N$ 是满足下述条件的次优解：

$$\mathcal{E}_{\text{av}}(N;\tilde{\mathbf{w}}_N)\leqslant\mathcal{E}_{\text{av}}(N;\hat{\mathbf{w}}_N)+\rho \tag{4.170}$$

其中 ρ 组成了一个新的可控参数；它提供了对于计算精确度的测量。

受这一实例的启发，现在我们有了一个比结构风险最小化方法遇到的更复杂的问题。具体地说，现在必须调整三个变量：

- 网络模型（例如，多层感知器中隐藏神经元个数）。
- 训练样本个数。
- 最优精确度（例如，过早地终止对最小值$\hat{\mathbf{w}}_N$的计算并选定次优解$\tilde{\mathbf{w}}_N$）。

为了达到最好的测试性能，必须满足预算约束，这定义了能用的最大训练样本个数以及我们能提供的最大计算时间。在实际的背景下，我们因此面对相当复杂的折中。为了解决这一约束最优问题，折中将依赖于我们是否首先达到样本数量的限制或者计算时间的限制。这两个限制的折中是主动预算约束，依赖于监督学习过程是小规模的还是大规模的，如我们下面要讨论的那样。

定义

根据 Bottou(2007)，小规模和大规模问题可以分别定义如下：

定义Ⅰ. 小规模学习

一个监督学习问题称为小规模的，此时训练样本的大小（即样本的个数）是强加于学习过程的主动预算约束。

定义Ⅱ. 大规模学习

一个监督学习问题称为大规模的，此时计算时间是强加于学习过程的主动预算约束。

换句话说，主动预算约束（active budget constraint）将两个学习问题区别开。

作为说明小规模学习问题的一个例子，我们可以给出自适应平衡装置（adaptive equalizer）的设计，其目的是为了补偿不可避免的在信道传输过程中信息数据的失真。起源于随机梯度下降并在第 3 章中讨论过的 LMS 算法被广泛应用于解这一在线学习问题（Haykin，2002）。

作为说明大规模学习问题的一个例子，我们可以给出支票读取机的设计，其训练样本是由联合对组成的，每个样本描述一个特定的〈图像，数额〉对，其中“图像”是关于支票的而数额是关于支票上钱的数量的。这样的学习问题由于如下几点具有复杂的强结构（Bottou，2007）：

- 区域分割
- 文字分割
- 文字识别
- 句法解释

4.17 节介绍的包含可微单元的卷积网络，通过几个星期的随机梯度算法的训练，被广泛用于解这一挑战性学习问题（LeCun 等，1998）。事实上，这一新型网络已经从 1996 年开始在工业界广泛应用，处理数十亿支票。

小规模学习问题

只考虑小规模学习问题时，机器学习的设计者可以得到以下三个变量：

- 训练样本个数，N
- 逼近网络函数族 F 的容许大小 K
- 式(4.170)引入的计算误差 ρ

当主动预算约束是样本个数时，第一种学习问题的设计选择如下所述（Bottou，2007）：

- 通过使得 N 大到预算允许的最大来减少估计误差。
- 通过令计算误差 $\rho=0$ 来减少最优化误差，这意味着令 $\tilde{\mathbf{w}}_N=\hat{\mathbf{w}}_N$。
- 调整$\mathcal{F}$的大小到认为是合理的程度。

当 $\rho=0$ 时，如图 4.29 所示的包括逼近估计折中的结构风险最小化方法，对于处理小规模学习问题是足够的。

大规模学习问题

正如前面所指出的那样，大规模问题的主动预算约束是计算时间。在处理这第二类学习问

题时，我们面对更复杂的折中，因为现在必须对计算时间 T 负责。

在大规模学习问题中，过剩误差是由差 $(J(\tilde{\mathbf{w}}_N)-J_{\text{actual}}(\hat{f}^*))$ 定义的，它可以分解为如下的三项（Bottou，2007）：

$$\underbrace{J(\tilde{\mathbf{w}}_N)-J_{\text{actual}}(\hat{f}^*)}_{\text{过剩误差}}=\underbrace{J(\tilde{\mathbf{w}}_N)-J(\hat{\mathbf{w}}_N)}_{\text{最优化误差}}+\underbrace{J(\hat{\mathbf{w}}_N)-J(\hat{\mathbf{w}}^*)}_{\text{逼近误差}}+\underbrace{J(\hat{\mathbf{w}}^*)-J_{\text{actual}}(\hat{f}^*)}_{\text{估计误差}} \tag{4.171}$$

最后的两项组成了逼近误差和估计误差，对小规模和大规模学习问题都是通常存在的。正是式(4.171)的第一项将大规模学习问题和小规模学习问题区别开来。这一新的称为最优化误差的项显然和计算误差 ρ 相关。

图 4.29 中逼近误差边界的计算对于小规模问题来说是很好理解的（利用 VC 理论）。遗憾的是，当这一公式用于大规模学习问题时，包含在公式中对边界的约束是很不好理解的。在这些更困难的情形下，用收敛速率而不是边界对式(4.171)进行分析是更富有成效的。

要求对式(4.171)中的三项的和通过调整如下可提供变量来最小化：

- 样本个数，N。
- 逼近网络函数$\mathcal{F}_K$ 的容许大小 K。
- 计算误差 ρ，它不再是 0。

做这样的最小化分析是极为困难的，因为计算时间 T 实际上依赖于所有三个变量 N，$\mathcal{F}$和 ρ。为了解释这一依赖性，我们给误差 ρ 分配一个小的值来减少最优化误差。为了实现这一减少，遗憾的是，我们必须增加 N，$\mathcal{F}$或两者，它们中的任一个都将具有对逼近和估计误差的不良影响。

虽然如此，在某些情形下，可能计算当 ρ 下降且$\mathcal{F}$和 N 都上升时三个误差倾向于下降的指数。类似地，也可以计算当 ρ 下降且$\mathcal{F}$和 N 都上升时计算时间 T 上升的指数。将这些片断放到一起，就有了应付大规模学习问题折中的逼近解的元素。更重要的是，在最后的分析中，折中依赖于最优化算法的选择。

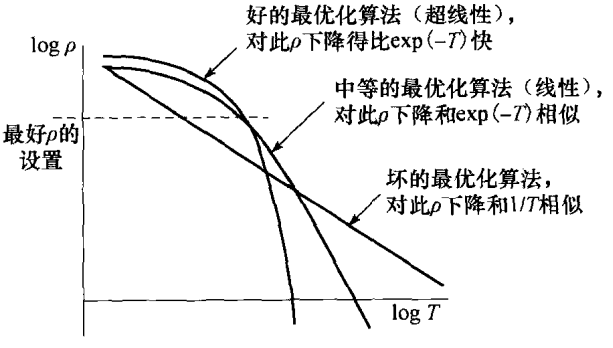


图 4.30　计算误差 ρ 和计算时间 T 的变化图，对三类最优化算法：坏的、中等的、好的（这个图的复制得到了 Dr. Leon Bottou 的同意）

图 4.30 给出了对大规模学习问题，采用不同最优化算法，$\log\rho$ 随着 $\log T$ 的变化曲线。这个图中给出了三类最优化算法（即坏的、中等的、好的）例子，相应地这些算法包含了随机梯度下降（即在线学习）、梯度下降（即批量学习）、二阶梯度下降（即 BFGS 类或其扩展的拟牛顿最优化算法）。表 4.4 总结了这三类最优化算法之间的不同特征。

现在我们可以总结从本节中给出的资料中得到的关于监督学习的消息如下：

小规模学习问题的研究已经有了良好的发展，但是大规模学习问题的研究还处在发展的早期阶段。

表 4.4　三种最优化算法统计特性的小结①

算法	每次迭代的代价	到达 ρ 的时间
1. 随机梯度下降（在线学习）	$O(m)$	$O\left(\frac{1}{\rho}\right)$
2. 梯度下降（批量学习）	$O(Nm)$	$O\left(\log\frac{1}{\rho}\right)$
3. 二阶梯度下降（在线学习）	$O(m(m+N))$	$O\left(\log\left(\log\frac{1}{\rho}\right)\right)$

注：m：输入向量 $\mathbf{x}$ 的维数
N：用于训练的样本的个数
ρ：计算误差
①这个表格是由 Bottou(2007) 编辑的。

4.20　小结和讨论

反向传播算法为多层感知器的训练建立了一个计算有效和有用的算法。这一算法的名字来源于这样的事实：其代价函数关于网络自由参数（突触权值和偏置）的偏导数（性能测试）是由误差信号（由输出神经元计算）通过网络一层一层反向传播来决定的。这样做，算法以一种最精致的方式解决了信用分配问题。算法的计算能力基于两个主要贡献：

- 局部方法，更新多层感知器的突触权值和偏置。
- 计算代价函数高效的算法，用于计算代价函数对这些自由参数的所有偏导数。

训练的随机和批量方法

对于训练数据的一个给定回合，反向传播算法以两种方式中的一种来操作：随机或者批量。在随机方式中，网络的所有神经元的突触权值都是在一个模式接着一个模式的逐次方式上调整的。因此，在计算中使用的误差曲面梯度向量的估算值在本质上是随机的——因此有了“随机反向传播”的名称。另一方面，在批量方式中，对所有突触权值和偏置的调整是在一个回合接一个回合的基础上进行的，这样在计算中使用梯度向量更精确的估计。无论它的缺点如何，反向传播学习的随机形式是神经网络设计中使用频率最高的，特别是在大规模问题上。为了得到最好的结果，需要小心地调整算法。

模式分类和非线性滤波

多层感知器设计中的特定细节问题自然依赖于有关具体的应用。然而，我们可以做出两种区分：

1. 在涉及非线性可分模式的模式分类中，网络中的所有神经元都是非线性的。这个非线性是通过使用 sigmoid 函数来获得的，该函数的两种通常用法是（a）logistic 函数，和（b）双曲正切函数。每个神经元负责在决策空间中产生它自己的超平面。通过一个监督学习过程，网络中由所有神经元形成的超平面的组合被反复调整，使其对来自不同类且未出现过的模式分类时，平均分类误差最小。对于模式分类来说，随机反向传播算法是实现训练最广泛使用的算法，特别是在大规模问题上（例如光学字符识别）。
2. 在非线性滤波中，多层感知器的输出的动态范围应该大到足以包含过程值；在这样的背景下，线性输出神经元的使用是最明智的选择。对学习算法，我们提供如下的观察事实：
   - 在线学习比批量学习慢得多。
   - 假设批量学习是期望的选择，标准反向传播算法比共轭梯度方法慢。

本章讨论的非线性滤波方法，集中于利用静态网络，以多层感知器为例；输入信号通过一个提供了时间的短期记忆结构（如抽头延迟线或者 gamma 滤波器）应用于多层感知器，而时间是滤波的重要一维。在第 15 章，我们将再次讨论非线性滤波器的设计，在该章中反馈作用于多层感知器，从而将之转化为循环神经网络。

小规模和大规模学习问题

一般来说，在机器学习问题的研究中出现三种误差：

1. *逼近误差*，这是在给定训练样本的固定大小 N 后，由训练神经网络或者机器学习所招致的误差。
2. *估计误差*，这是在机器的训练完成后，用以前没有出现过的数据测试其性能所招致的误差；从效果上而言，估计误差是泛化误差的另一个途径。
3. *最优化误差*，这是对于预先给定的计算时间 T 来说，训练机器的计算精确度所引起的。

在小规模学习问题中，我们发现主动预算约束是训练样本大小，其隐含意义在于最优化误差实际上通常是零。因此结构风险最小化的 Vapnik 理论对于处理小规模学习问题来说是足够的。另一方面，在大规模学习问题中，主动预算约束是可用的计算时间 T，此时最优化误差自身起着关键的作用。特别地，学习过程的计算精确度以及因此而来的最优化误差受到用于求解学习问题的最优化算法类型的巨大影响。

注释和参考文献

1. sigmoid 函数的图形是“s”形的；Menon 等（1996）对两类 sigmoid 函数进行了深入的研究：
   - 简单 sigmoid，定义为渐进有界的和完全单调的单变量奇函数。
   - 双曲 sigmoid，代表简单 sigmoid 的一个真子集和双曲正切函数的自然推广。
2. 对于 LMS 算法的特殊情形，已经证明使用动量常数 α 降低学习率参数 η 的稳定范围，并且如果 η 没有被适当调整，这样会导致不稳定。此外，错误调整也随 α 的增加而增长；更详细的论述参见 Roy and Shynk (1990)。
3. 如果向量 $\mathbf{w}^*$ 不比它邻近的点向量更差的话，向量 $\mathbf{w}^*$ 被称为输入输出函数 F 的一个局部最小值；也就是，存在一个 ε 使得
$$F(\mathbf{w}^*) \leqslant F(\mathbf{w}) \qquad \text{对所有满足 } \|\mathbf{w}-\mathbf{w}^*\| < \varepsilon \text{ 的 } \mathbf{w}$$
(Bertsekas, 1995)。如果 $\mathbf{w}^*$ 不比其他所有的向量都差，则称它为函数 F 的一个全局最小值；也就是，
$$F(\mathbf{w}^*) \leqslant F(\mathbf{w}) \qquad \text{对于所有 } \mathbf{w} \in \mathbb{R}^n$$
其中 n 是 $\mathbf{w}$ 的维数。
4. 对有效梯度估计应用反向传播的首次文献记载应归功于 Werbos(1974)。在第 4.8 节中给出的材料依照 Saarinen 等（1992）给出的处理方法；Werbos(1990) 对该题目给出更一般的讨论。
5. Battiti(1992) 回顾了计算 Hessian 矩阵的精确算法和近似算法，并有特别针对神经网络的参考文献。
6. Muller 等（1998）研究了将式(4.77)的退火在线学习算法应用于不稳定盲源分离问题，这说明了 Murata (1998) 的学习率自适应控制的广泛算法适用性。盲源分离问题在第 10 章中讨论。
7. 式(4.80)的公式遵循根据 Sompolinski 等（1995）最优退火在线学习算法的对应部分，用于处理学习率参数的自适应。这一算法的实际局限包括需要在每一步迭代计算 Hessian 矩阵，并且需要知道学习曲线的最小损失。
8. 通用逼近定理可以看作是 Weierstrass 定理（Weierstrass, 1885；Kline, 1972）的自然扩展。这个定理表明
   > 任何一个在实轴闭区间上的连续函数都可以表示成该区间上绝对一致收敛的多项式级数的极限。

   利用多层感知器来表示任意连续函数的优势，这一研究可能是 HechtNielsen(1987) 首先关注的。他引用了 Sprecher(1965) 的 Kolomogorov 叠加定理的改进版。然后 Gallant 和 White(1988) 证明，在隐藏层具有单调“余弦”挤压和在输出无挤压的单隐藏层多层感知器是被作为“Fourier 网络”的特殊情形嵌入的，它的输出产生给定函数的 Fourier 级数逼近。然而，在传统的多层感知器背景下，Cybenko 第一次严格证明了一个隐藏层足够一致逼近任何具有在单位超立方体中的支集的函数；这项工作作为 1988 伊利诺斯大学的技术报告发表，一年之后作为论文发表（Cybenko, 1988, 1989）。在 1989 年，另外两篇关于多层感知器通用逼近器的论文独立发表了，一篇由 Funahashi 完成，另外一篇由 Hornik 等（1990）完成。对后来关于逼近问题的贡献，参见 Light(1992b)。
9. 交叉验证的发展历史在 Stone(1974) 中有记载。交叉验证的思想至少在 20 世纪 30 年代就已广泛传播，但该项技术的改进是在 20 世纪 60 年代和 70 年代完成的。该领域的两篇重要论文是 Stone（1974）和 Geisser (1975)，他们独立并且几乎同时提出这项技术。这项技术被 Stone 命名为“交叉验证方法”，而 Geisser 则称为“预测样本复用方法”。
10. Hecht-Nielsen(1995) 描述了一种复制器神经网络，它是具有三个隐藏层和一个输出层的多层感知器的形式：
    - 在第一和第三隐藏层中的激活函数通过双曲正切函数定义：
$$\varphi^{(1)}(v) = \varphi^{(3)}(v) = \tanh(v)$$
      其中 v 是在这些层中神经元的诱导局部域。
    - 在第二隐藏层中的每个神经元的激活函数由

$$\varphi^{(2)}(v) = \frac{1}{2} + \frac{2}{2(N-1)} \sum_{j=1}^{N-1} \tanh\left(a\left(v - \frac{j}{N}\right)\right)$$

给出，其中 a 是一个增益参数，v 是该层中神经元的诱导局部域。函数 $\varphi^{(2)}(v)$ 描述一个光滑的具有 N 级的阶梯激活函数，因而本质上把相关神经元层的输出向量转化为 $K = N^n$ 级，其中 n 是中间隐藏层的神经元数目。

- 输出层中的神经元是线性的，它们的激活函数定义为

$$\varphi^{(4)}(v) = v$$

- 基于这种神经网络结构，Hecht-Nielsen 提出了一个定理，证明对随机输入数据向量的最佳数据压缩是可以得到的。

11. 共轭梯度方法的经典参考文献是 Hestenes and Stiefel(1952) 的著作。关于共轭梯度算法收敛行为的讨论，见 Luenberger(1984)and Bertsekas(1995)。关于共轭梯度算法的许多方面的指导性处理方法，见 Shewchuk (1994)。关于在神经网络领域中该算法的易读文献见 Johansson 等 (1990)。
12. 共轭梯度算法的传统形式要求使用直线搜索方法，它可能因为自身的尝试性和误差性而花费时间。Møller (1993) 描述共轭梯度算法的一个修改版本，称为比例共轭梯度算法，它避免使用直线搜索。从本质上来说，直线搜索由算法的一维空间的 Levenberg-Marquardt 形式代替。使用这种办法的动机是避开由非正定 Hessian 矩阵引起的困难 (Fletcher, 1987)。
13. 被称为$\mathcal{R}$的技术是由 Pearlmutter(1994) 而来，它提供了计算矩阵向量乘积的有效程序；因此，这一技术能够实际应用于计算式(4.138)中的逆 Hessian 矩阵 $\mathbf{H}^{-1}$。习题 4.6 中会用到$\mathcal{R}$技术。
14. Fukushima(1980, 1995) 在设计一个称为神经认知机的学习机时，引用了 Hubel 和 Wiesel 关于“简单”和“复杂”细胞的概念，这是该概念在神经网络文献中首次被引用。然而，这个学习机以自组织的形式运行，而图 4.23 描述的卷积网络使用标定的样本以监督的形式运行。
15. 对于通用短视映射定理的起源，参看 Sandberg(1991)。
16. 对于 VC 维数的细节和相关的实验误差的讨论，参看 Vapnik(1998) 关于统计学习理论的经典书籍。VC 维数也在 Schölkopf and Smola(2002) 以及 Herbrich(2002) 的书中做了讨论。值得一提的是：VC 维数和 Cover 分离能力有关，这将在第 5 章中讨论。

习题

反向传播学习

4.1 为了解决 XOR 问题，图 P4.1 表示一个包括单个隐藏神经元的神经网络；这个网络可以看作是在第 4.5 节中所考虑模型的替代模型。通过构建 (a) 决策区域和 (b) 网络的真值表，证明图 P4.1 表示的网络解决了 XOR 问题。

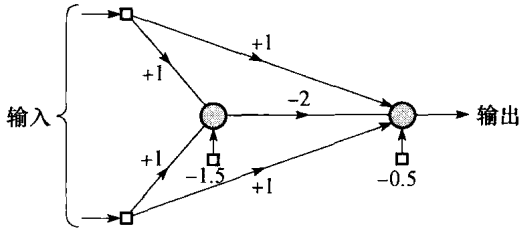


图　P4.1

4.2 使用反向传播算法为图 4.8 所示的神经网络计算一组突触权值和偏置的值以解决 XOR 问题。假设非线性使用一个 logistic 函数。

4.3 动量项 α 通常被指定为在 $0 \leqslant \alpha < 1$ 范围的正值。如果 α 是赋予在 $-1 < \alpha \leqslant 0$ 之间的一个负值，研究在这样的条件下使得式(4.43)关于时间 t 的行为差异。

4.4 考虑包括单个权值的网络的简单例子，它的代价函数是：

$$\mathcal{E}(w) = k_1(w - w_0)^2 + k_2$$

其中 w_0、k_1 和 k_2 是常数。用具有动量项 α 的反向传播算法最小化$\mathcal{E}(w)$。

探索包含的动量项常数 α 是怎样影响学习过程的。特别注意与 α 相对而言达到收敛所需的步数。

4.5 式(4.51)到式(4.53)定义图 4.14 中的多层感知器实现的逼近函数 $F(\mathbf{w},\mathbf{x})$ 的偏导数。根据如下的条件推导这些公式：

(a) 代价函数：

$$\mathscr{E}(n)=\frac{1}{2}[d-F(\mathbf{w},\mathbf{x})]^2$$

(b) 神经元 j 的输出：

$$y_j=\varphi\Big(\sum_{i=1}w_{ji}y_i\Big)$$

其中 w_{ji} 是从神经元 i 到神经元 j 的突触权值，y_i 是神经元 i 的输出。

(c) 非线性：

$$\varphi(v)=\frac{1}{1+\exp(-v)}$$

4.6 由 Pearlmutter(1994) 提出的$\mathscr{R}$技术，提供了计算矩阵向量乘积的快速计算程序。为了说明这一程序，考虑一个单一隐藏层的多层感知器；网络的前向传播公式定义为：

$$v_j=\sum_i w_{ji}x_i$$

$$z_j=\varphi(v_j)$$

$$y_k=\sum_j w_{kj}z_j$$

$\mathscr{R}[\cdot]$记为作用于括号内的量的一个算子，用于对手头的示例网络产生如下的结果：

$$\mathscr{R}[v_j]=\sum_i a_{ji}x_i,\qquad \mathscr{R}[w_{ji}]=a_{ji}$$

$$\mathscr{R}[v_j]=\varphi'(v_j)\,\mathscr{R}[v_j],\qquad \varphi'(v_j)=\frac{\partial}{\partial v_j}\varphi(v_j)$$

$$\mathscr{R}[y_k]=\sum_j w_{kj}\,\mathscr{R}[z_j]+\sum_i a_{ji}z_j,\qquad \mathscr{R}[w_{kj}]=a_{kj}$$

$\mathscr{R}$结果被视为新的变量。从效果上讲，算子$\mathscr{R}[\cdot]$遵循附加如下条件的微积分学的通常规则：

$$\mathscr{R}[\mathbf{w}_j]=\mathbf{a}_j$$

其中 $\mathbf{w}_j$ 是连接到节点 j 的权值向量，$\mathbf{a}_j$ 是由应用$\mathscr{R}$算子所得到的关联向量。

(a) 对反向传播算法应用$\mathscr{R}$技术，推导矩阵向量乘积 $\mathbf{Ha}$ 的元素的表达式，识别隐藏和输出神经元的新变量。对于这一应用，利用本习题开始所描述的多层感知器。

(b) 证明$\mathscr{R}$技术是计算快速的。

监督学习问题

4.7 在这一习题中，我们研究多层感知器完成的输出表达和决策规则。从理论上讲，对于 M 类分类问题，M 个不同类的结合形成了整个输入空间，我们共需要 M 个输出来表示所有可能的分类决策，如图 P4.7 所示。在这个图中，向量 $\mathbf{x}_j$ 记为由多层感知器分类的 m 维随机向量 $\mathbf{x}$ 的第 j 个原型 (prototype)(即，唯一样本)。$\mathbf{x}$ 能属于的 M 个可能类的第 k 个类记为$\mathscr{C}_k$。令 y_{kj} 为响应于原型 x_j 的网络第 k 个输出，如下所示：

$$y_{kj}=F_k(\mathbf{x}_j),\qquad k=1,2,\cdots,M$$

其中函数 $F_k(\cdot)$定义网络学习的从输入到第 k 个输出的映射。

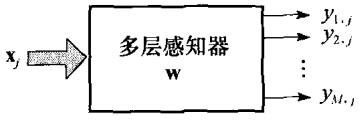


图 P4.7　习题 4.7 中模式分类器框图

为了表述的方便，令

$$\mathbf{y}_j=[y_{1j},y_{2j},\cdots,y_{Mj}]^T=[F_1(\mathbf{x}_j),F_2(\mathbf{x}_j),\cdots,F_M(\mathbf{x}_j)]^T=\mathbf{F}(\mathbf{x}_j)$$

其中 $F(\cdot)$是向量值函数。我们在这一问题中希望解决的基本问题是：

在多层感知器训练之后，对于分类网络的 M 个输出而言什么是最优决策规则？

为了解决这一问题，考虑使用对隐藏层神经元嵌入 logistic 函数的多层感知器并且在如下假设下运行：

- 训练样本的大小足够大使得能够对正确分类概率做合理的精确估计。
- 用于训练多层感知器的反向传播算法不陷入局部极小点。

具体来说，对多层感知器的 M 个输出提供后验类概率估计的性质进行数学讨论。

4.8 在这一问题中，我们回顾第 4.10 节中讨论过的学习率的自适应控制。感兴趣的问题是论证式(4.85)中

的学习率 $\eta(n)$的渐进行为，当迭代次数趋于无穷时不收敛于零。

(a) 令 $\bar{\mathbf{r}}(n)$记对于样本 $\{\mathbf{x},\mathbf{d}\}$ 的辅助向量 $\mathbf{r}(n)$的期望。证明如果估计$\hat{\mathbf{w}}(n)$是在最优估计 $\mathbf{w}^*$ 的很邻近，我们可以写为

$$\bar{\mathbf{r}}(n+1) \approx (1-\delta)\bar{\mathbf{r}}(n) + \delta\mathbf{K}^* (\hat{\mathbf{w}}(n) - \bar{\mathbf{w}}(n))$$

其中 $\bar{\mathbf{w}}(n)$是估计$\hat{\mathbf{w}}(n)$的均值，δ 是小的正参数。

(b) 在 Heskas and Kappen(1991) 中，证明了估计$\hat{\mathbf{w}}(n)$ 被一个高斯分布的随机变量逼近。因此，证明下面的渐进行为：

$$\lim_{n\to\infty} \hat{\mathbf{w}}(n) \neq \mathbf{0}$$

这一条件关于学习率参数 $\eta(n)$的渐近行为教给我们什么？

4.9 最小描述长度 (MDL) 准则的组成描述如下 (参看式(2.37))：

$$\text{MDL} = (\text{误差项}) + (\text{复杂项})$$

讨论权延迟方法应用于网络修剪是如何符合 MDL 形式的。

4.10 在网络修剪的最优脑损伤 (OBD) 算法中，根据 LeCun 等 (1990b)，Hessian 矩阵 $\mathbf{H}$ 由其对角版本逼近。利用这一逼近，推导作为最优脑外科 (OBS) 算法的特殊情形的 OBD 过程，这已经在 4.14 节中学习过了。

4.11 在 Jacobs (1988) 中，对在线反向传播学习的加速收敛，提出了以下启发：

(i) 代价函数的每一个可调整网络参数将具有其自身的学习率参数。

(ii) 每一个学习率参数将被允许从一次迭代到下一次迭代之间发生变化。

(iii) 当代价函数对于突触权值的导数和算法几次连续迭代的代数符号相同时，这一特定权值的学习率参数将被增加。

(iv) 当代价函数对于特定突触权值的代数符号和算法的几次连续迭代发生变化时，对该权值的学习率参数将被降低。

这四个启发满足反向传播算法的位置约束。

(a) 利用直觉讨论来验证这四个启发。

(b) 反向传播算法中权值更新的动量的包括可以看作是满足了启发 (iii) 和 (iv) 的机制。证明这一声明的有效性。

二阶最优化方法

4.12 在式(4.41)所述的权值修改中动量项的使用可以被认为是共轭梯度方法的近似 (Battiti, 1992)。讨论这种说法的正确性。

4.13 以式(4.127)中 $\beta(n)$的公式开始，推导 Hesteness-Stiefel 公式：

$$\beta(n) = \frac{\mathbf{r}^T(n)(\mathbf{r}(n) - \mathbf{r}(n-1))}{\mathbf{s}^T(n-1)\mathbf{r}(n-1)}$$

其中 $\mathbf{s}(n)$是方向向量，$\mathbf{r}(n)$是共轭梯度方法中的余项。利用这个结果，推导式(4.128)中的 Polak-Ribiére 公式和式(4.129)中的 Fletcher-Reeves 公式。

时序处理

4.14 图 P4.14 描述用高斯形式的时间窗口作为时序处理的方法，这是受到神经生物学考虑的启发 (Bodenhausen and Waibel, 1991)。与神经元 j 的突触 i 相联系的时间窗口，记为 $\theta(n, \tau_{ji}, \sigma_{ji})$，其中 τ_{ji} 和 σ_{ji} 分别表示时延和窗口的宽度，表示为

$$\theta(n,\tau_{ji},\sigma_{ji}) = \frac{1}{\sqrt{2\pi}\sigma_{ji}} \exp\left(-\frac{1}{2\sigma_{ji}^2}(n-\tau_{ji})^2\right), i = 1,2,\cdots,m_0$$

神经元 j 的输出模型为

$$y_j(n) = \varphi\left(\sum_{i=1}^{m_0} w_{ji} u_i(n)\right)$$

其中 $u_i(n)$是输入 $x_i(n)$和时间窗口 $\theta(n,\tau_{ji},\sigma_{ji})$的卷积。属于神经元 j 的突触 i 的权值 w_{ji} 和时延 τ_{ij} 都使用监督方式学习。

这个学习可以通过标准的反向传播算法来实现。试通过推导 w_{ji}，τ_{ji}，σ_{ji} 的更新公式来演示这个学习过程。

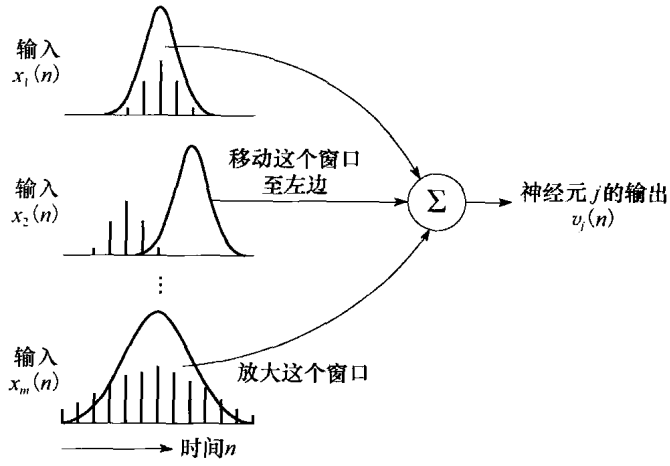


图 P4.14　习题 4.14 的图：附加于高斯窗口的指示是针对学习算法的

计算机实验

4.15　研究使用 sigmoid 非线性函数的反向传播学习方法获得一对一映射，描述如下：

1. $f(x)=\dfrac{1}{x}$，$1\leqslant x\leqslant 100$

2. $f(x)=\log_{10}x$，$1\leqslant x\leqslant 10$

3. $f(x)=\exp(-x)$，$1\leqslant x\leqslant 10$

4. $f(x)=\sin x$，$0\leqslant x\leqslant \dfrac{\pi}{2}$

对每个映射，完成如下工作：

(a) 建立两个数据集，一个用于网络训练，另一个用于测试。

(b) 假设具有单个隐藏层，利用训练数据集计算网络的突触权值。

(c) 通过使用测试数据求网络计算精度的值。

使用单个隐藏层，但隐藏神经元数目可变，研究网络性能是如何受隐藏层大小变化影响的。

4.16　重复 4.7 节对 MLP 分类器的计算机试验，其中两月之间的距离设为 $d=0$。根据习题 1.6 中关于感知器对于同样设置的相应试验来评价你的试验发现。

4.17　在这一试验中，考虑一个理论上已知其决策边界的模式分类试验。本试验的主要目的是看看如何就最优决策边界而言从试验上最优化多层感知器的设计。

具体来说，要求如何区分两个具有相互覆盖的二维高斯分布模式的等可能类，这两个类标示为$\mathscr{C}_1$和$\mathscr{C}_2$。这两个类的条件概率密度函数是：

$$\text{Class }\mathscr{C}_1 \qquad p_{\mathbf{x}|\mathscr{C}_1}(\mathbf{x}|\mathscr{C}_1)=\frac{1}{2\pi\sigma_1^2}\exp\left(-\frac{1}{2\sigma_1^2}\|\mathbf{x}-\boldsymbol{\mu}_1\|^2\right)$$

其中

$\boldsymbol{\mu}_1$ = 均值向量 $=[0,0]^T$

σ_1^2 = 方差 $=1$

$$\text{Class }\mathscr{C}_2 \qquad p_{\mathbf{x}|\mathscr{C}_2}(\mathbf{x}|\mathscr{C}_2)=\frac{1}{2\pi\sigma_2^2}\exp\left(-\frac{1}{2\sigma_2^2}\|\mathbf{x}-\boldsymbol{\mu}_2\|^2\right)$$

其中

$\boldsymbol{\mu}_2=[2,0]^T$

$\sigma_2^2=4$

(a) 最优贝叶斯决策边界是由似然比测试

$$\Lambda(\mathbf{x})\underset{\mathscr{C}_2}{\overset{\mathscr{C}_1}{\gtrless}}\lambda$$

定义的，其中

$$\Lambda(\mathbf{x}) = \frac{p_{\mathbf{x}|\mathscr{C}_1}(\mathbf{x}|\mathscr{C}_1)}{p_{\mathbf{x}|\mathscr{C}_2}(\mathbf{x}|\mathscr{C}_2)}$$

λ 是两类的先验概率决定的阈值。证明最优决策边界是一个圆，其圆心处于

$$\mathbf{x}_{\mathscr{C}} = \begin{bmatrix} -2/3 \\ 0 \end{bmatrix}$$

半径是 $r=2.34$。

(b) 假设利用单一隐藏层。要求通过试验决定隐藏神经元的最优个数。

- 从具有两个隐藏神经元的多层感知器开始，利用学习率 $\eta=0.1$ 和动量常数 $\alpha=0$ 的反向传播算法来训练网络，利用下面的方案来计算正确分类的概率：

训练样本大小	回合数
500	320
2 000	80
8 000	20

- 重复这一试验，这一次利用四个隐藏神经元，其他都与前面相同。比较这第二个试验的结果和前面的试验结果，然后根据你考虑的最优选择来选择两个还是四个隐藏神经元的网络结构。

(c) 对于 (b) 部分选择的“最优”网络，现在转向试验性地寻找学习率参数 η 和动量常数 α 的最优值。为了这样做，利用下面参数的组合来完成试验：

$$\eta \in [0.01, 0.1, 0.5]$$
$$\alpha \in [0.0, 0.1, 0.5]$$

从而，决定产生正确分类最好概率的 η 和 α 的值。

(d) 已经有了隐藏层最优大小以及 η 和 α 的最优集后，完成最后的试验来寻找最优决策边界和相应的最优分类概率。比较这样通过试验获得的最优性能和理论最优解，对你的结果做出评论。

4.18 在这个习题里我们用标准的反向传播算法来解决困难的非线性预测问题，比较它与 LMS 算法的性能。要考虑的时间序列由离散 Volterra 模型建立，其形式为

$$x(n) = \sum_i g_i v(n-i) + \sum_i \sum_j g_{ij} v(n-i)v(n-j) + \cdots$$

其中 $g_i, g_{ij}, \cdots$ 是 Volterra 系数。$v(n)$是独立的高斯分布白噪声序列的抽样。$x(n)$是 Volterra 模型的输出结果。第一个求和项是我们熟悉的滑动平均 (MA) 时间序列模型，剩余的求和项是更高阶的非线性的部分。一般而言，对 Volterra 系数的估计通常认为是困难的，主要是因为它们和数据的非线性关系。考虑一个简单的例子：

$$x(n) = v(n) + \beta v(n-1)v(n-2)$$

时间序列具有零均值，不相关，从而有一个白噪声的谱。然而，时间序列的样本并不是互相独立的，因而可以构造一个高阶预测器。模型输出的方差由

$$\sigma_x^2 = \sigma_v^2 + \beta^2 \sigma_v^4$$

给出，其中 σ_v^2 是白噪声的方差。

(a) 构造一个多层感知器，有 6 个输入节点，隐藏层含有 16 个神经元，只有一个输出神经元。使用抽头延时线记忆馈给网络的输入层。隐藏层神经元使用 sigmoid 激活函数，限制在区间 [0,1] 内，而输出神经元充当一个线性的组合器。网络使用标准反向传播算法进行训练，有关参数如下：

学习率参数	$\eta=0.001$
动量常数	$\alpha=0.6$
处理的样本总数	100 000
每个回合的样本数目	1 000
总的回合数目	2 500

白噪声方差 σ_v^2 为 1。因此，用 $\beta=0.5$，我们求出预测器的输出方差为 $\sigma_x^2=1.25$。

计算非线性预测器的学习曲线，将预测器输出 $x(n)$的方差绘制成训练样本的回合数的函数，一直画到 2 500 个回合。为了准备进行训练的每个回合，探讨下述两种方式：

（ⅰ）维持训练样本的时序，从一个回合到下一个回合与产生它的时序一样。

（ⅱ）训练样本的顺序从一个模式（状态）到另一个模式是随机产生的。

同时，对 1 000 个样本的验证集使用交叉验证（在第 4.13 节中描述），来监测预测器的学习行为。

(b) 重复试验，使用 LMS 算法对 6 个样本的输入执行线性预测。算法的学习率参数设置为 $\eta=10^{-5}$。

(c) 重复整个实验，用 $\beta=1$，$\sigma_v^2=2$；接着再重复，用 $\beta=2$，$\sigma_v^2=5$。

每个实验的结果应该揭示反向传播算法和 LMS 算法最初基本遵循相似的途径，然而反向传播算法继续改进，最终产生一个接近预定值 σ_x^2 的预测方差。

4.19 在本试验中，我们利用由反向传播算法训练的多层感知器来完成 Lorenz 吸引子的一步预测。这一吸引子的动力学系统由下面的三个方程来定义：

$$\frac{dx(t)}{dt}=-\sigma x(t)+\sigma y(t)$$

$$\frac{dy(t)}{dt}=-x(t)z(t)+rx(t)-y(t)$$

$$\frac{dz(t)}{dt}=x(t)y(t)-bz(t)$$

其中 σ，r，和 b 是无量纲的参数。这些参数的典型值是 $\sigma=10$，$b=8/3$ 和 $r=28$。

多层感知器的详细情况如下所示：

源节点个数：20

隐藏层神经元个数：200

输出神经元个数：1

数据集的特性如下所示：

训练样本：700 个数据点

测试样本：800 个数据点

用于训练的回合数：50

反向传播算法的参数如下所示：

学习率参数 η从 10^{-5}线性退火到 10^{-10}。

动量：$\alpha=0$

(a) 计算 MLP 的学习曲线，画出均方误差对用于训练的回合数的图。

(b) 计算 Lorenz 吸引子的一步预测；具体来说，画出时间的函数所获得的结果，比较预测结果和 Lorenz 吸引子的演化结果。

第5章
Neural Networks and Learning Machines, 3E

核方法和径向基函数网络

本章组织

在本章中，我们学习机器学习的另一种途径：基于聚类的核方法。在5.1节的引言之后，本章剩下的章节组织如下：

5.2节介绍关于模式可分的Cover定理。该定理是通过对XOR问题的回顾来描述的。

5.3节讨论利用径向基函数来求解插值问题。

5.4节讨论构造径向基函数（RBF）网络，也包括了对于RBF网络的实际考虑。

5.5节讨论K-均值算法，该算法提供一个用于聚类的简单但普及的算法，对于在非监督方式下训练隐藏层是很适合的。5.6节是在K-均值聚类算法之后描述最小二乘估计的递归执行，这是用于在监督方式下训练RBF网络的输出层。5.7节讲述设计RBF网络时对于这两阶段过程的实际考虑。这一过程在5.8节的计算机实验中具体说明，并和第4章中运用反向传播算法所做的同样的计算机试验的结果作了比较。

5.9节考察高斯隐藏单元的解释，5.10节考察统计学中核回归和RBF网络之间的关系。

最后是5.11节的小结和讨论。

5.1 引言

对神经网络的监督学习有多种不同的方法。第4章所描述的多层感知器的反向传播算法，可以看作是递归技术的应用，这种技术在统计学中通称为随机逼近。

在本章中，我们采用完全不同的途径。具体来说，通过包含如下两阶段的混合方式来解决非线性可分模式的分类问题：

- 第一阶段将一个给定的非线性可分模式的集合转换为新的集合，在一定的条件下，转换后的模式变为线性的可能性很高；关于这一转换的数学证明可以追溯到Cover（1965）的早期论文。
- 第二阶段通过最小二乘估计（第2章已讨论过）来解给定的分类问题。

我们首先通过插值问题的讨论来描述关于这一混合方式对模式分类问题的一种执行方式：使用径向基函数网络（radial-basis function network，RBF)[1]，该网络结构由三层组成：

- 输入层由一些源节点（感知单元）组成，它们将网络与外界环境连接起来。
- 第二层由隐藏单元组成，它的作用是从输入空间到隐藏（特征）空间之间进行非线性变换。在大多数情况下网络仅有的隐藏层具有较高的维数，这一层是利用混合学习过程的第一阶段在非监督方式下训练的。
- 输出层是线性的，它是为提供网络的响应而专门设计的，该响应提供给应用于输入层的激活模式。这一层是利用混合过程的第二阶段在监督方式下训练的。

从输入空间到隐藏空间的非线性变换以及隐藏空间的高维数满足了Cover定理仅有的两个条件。

RBF网络的多数理论建立在高斯函数之上，这一类中一个重要的成员是径向基函数。高斯函数可以看作是一个核——因此基于高斯函数的两阶段过程的设计可看成是核方法。

讲到核，在本章的后面部分，我们也要讨论统计学中的核回归和径向基函数网络之间的关系。

5.2 模式可分性的Cover定理

当用径向基函数神经网络来解决一个复杂的模式分类任务时，问题基本可通过以下方式解

决：首先用非线性方法将其变换到高维空间，然后在输出层进行分类。模式可分性的 Cover 定理，说明了这样做的潜在合理性，该定理可以定性地表述如下（Cover，1965）：

假设空间不是稠密分布的，将复杂的模式分类问题非线性地投射到高维空间将比投射到低维空间更可能是线性可分的。

从第 1 章到第 3 章对单层结构的研究中知道，一旦模式具有线性可分性，则分类问题相对而言就更容易解决。因此，我们通过研究模式的可分性可以深入了解 RBF 网络作为模式分类器是如何工作的。

考虑一族曲面，每一个曲面都自然地将输入空间分成两个区域。用$\mathscr{X}$代表 N 个模式（向量）$\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_N$ 的集合，其中每一个模式都分属于两个类$\mathscr{X}_1$ 和$\mathscr{X}_2$ 中的一类。如果在这一族曲面中存在一个曲面能够将分别属于$\mathscr{X}_1$ 和$\mathscr{X}_2$ 的这些点分成两部分，我们就称这些点的二分（二元划分）关于这族曲面是可分的。对于每一个模式 $\mathbf{x}\in\mathscr{X}$，定义一个由一组实值函数 $\{\varphi_i(\mathbf{x})\mid i=1,2,\cdots,m_1\}$ 组成的向量，表示如下：

$$\boldsymbol{\phi}(\mathbf{x})=[\varphi_1(\mathbf{x}),\varphi_2(\mathbf{x}),\cdots,\varphi_{m_1}(\mathbf{x})]^T \tag{5.1}$$

假设模式 $\mathbf{x}$ 是 m_0 维输入空间的一个向量，则向量$\boldsymbol{\phi}(\mathbf{x})$将 m_0 维输入空间的点映射到新的 m_1 维空间的相应的点上。我们将 $\varphi_i(\mathbf{x})$称为*隐藏函数*，因为它与前馈神经网络中的隐藏单元起着同样的作用。相应地，由隐藏函数集合 $\{\varphi_i(\mathbf{x})\}_{i=1}^{m_1}$所生成的空间被称为隐藏空间或者特征空间。

我们称一个关于$\mathscr{X}$的二分 $\{\mathscr{X}_1,\mathscr{X}_2\}$ 是$\boldsymbol{\phi}$可分的，如果存在一个 m_1 维的向量 $\mathbf{w}$ 使得我们得到如下公式（Cover，1965）：

$$\begin{aligned}\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x})>0,\quad &\mathbf{x}\in\mathscr{X}_1\\ \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x})<0,\quad &\mathbf{x}\in\mathscr{X}_2\end{aligned} \tag{5.2}$$

由方程

$$\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x})=0$$

定义的超平面描述$\boldsymbol{\phi}$空间（即特征空间）中的分离曲面。这个超平面的逆像，即

$$\mathbf{x}:\quad \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x})=0 \tag{5.3}$$

定义输入空间中的*分离曲面*（即决策边界）。

考虑一个利用 r 次模式向量坐标乘积的线性组合实现的一个自然类映射。与此种映射相对应的分离曲面被称为 r *阶有理簇*。一个 m_0 维空间的 r 阶有理簇可描述为输入向量 $\mathbf{x}$ 的坐标的一个 r 次齐次方程，表示为

$$\sum_{0\leqslant i_1\leqslant i_2\leqslant\cdots\leqslant i_r\leqslant m_0} a_{i_1i_2\cdots i_r}x_{i_1}x_{i_2}\cdots x_{i_r}=0 \tag{5.4}$$

其中 x_i 是输入向量 $\mathbf{x}$ 的第 i 个元素。为了用齐次形式来表达方程，将 x_0 的值置为单位值 1。$\mathbf{x}$ 中项 x_i 的 r 阶乘积，即 $x_{i_1}x_{i_2}\cdots x_{i_r}$，被称为*单项式*。对于一个 m_0 维的输入空间在式（5.4）中一共有

$$\frac{(m_0-r)!}{m_0!r!}$$

个单项式。式(5.4)所描述的分离曲面的类型的例子有*超平面*（一阶有理簇）、*二次曲面*（二阶有理簇）和*超球面*（带有某种线性限制系数的二次曲面）等。这些例子的说明见图 5.1，该图说明在二维输入空间中的五点的构形。通常情况下，线性可分性暗示着球面可分性，而球

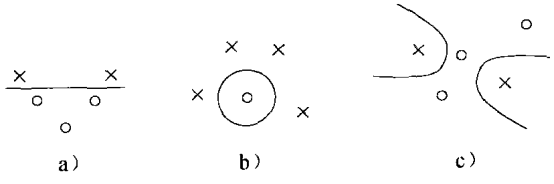


图 5.1　二维平面上 5 个点的不同集合的 φ—可分二分的 3 个例子：a）线性可分的二分；b）球形可分的二分；c）二次可分的二分

面可分性又暗示着二次可分性；反之则不一定成立。

在概率实验中，一个模式集合的可分性是一个随机事件，该随机事件依赖于选择的二分以及输入空间的模式分布。假设激活模式 $\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_N$ 是根据输入空间中的概率特性而独立选取的。同时假设所有的关于$\mathscr{X}=\{x_i\}_{i=1}^{N}$的二分都是等概率的。令 $P(N,m_1)$表示某一随机选取的二分是 φ 可分的概率，这里被选中的分离曲面的类具有 m_1 维的自由度。根据 Cover (1965)，可以将 $P(N,m_1)$表述为：

$$P(N,m_1)=\begin{cases}\left(\frac{1}{2}\right)^{N-1}\sum_{m=0}^{m_1-1}\binom{N-1}{m} & \text{对 } N>m_1-1\\ 1 & \text{对 } N\leqslant m_1-1\end{cases} \tag{5.5}$$

这里，包括 $N-1$ 和 m 的二项式系数对所有的整数 l 和 m 定义如下：

$$\binom{l}{m}=\frac{l!}{(l-m)!m!}$$

要说明式(5.5)的图形，最好通过令 $N=\lambda m_1$ 来归一化方程并对 m_1 的变化值画出概率 $P(\lambda m_1, m_1)$对 λ 的图。这个图揭示了两个有趣的性质 (Nilsson, 1965)：

- 在 $\lambda=2$ 附近宣称的阈值效应 (threshold effect)；
- 对于 m_1 的每个值 $P(2m_1, m_1)=1/2$。

式(5.5)隐含了 Cover 的可分性定理对于随机模式的本质[2]。它说明累计二项概率分布，相当于抛 $(N-1)$ 次硬币有 (m_1-1) 次或更少次头像向上的概率。

尽管在式(5.5)的推导中遇见的隐藏单元曲面是一个多项式的形式，因而与我们通常在径向基函数网络中用到的有所不同，但是该式的核心内容却具有普遍的适用性。具体来说，隐藏空间的维数 m_1 越高，则概率 $P(N,m_1)$就越趋向于 1。总之，关于模式可分性的 Cover 定理主要包含下面两个基本部分：

1. 由 $\varphi_i(\mathbf{x})$定义的隐藏函数的非线性构成，这里 $\mathbf{x}$ 是输入向量，且 $i=1,2,\cdots,m_1$。
2. 高维数的隐藏（特征）空间，这里的高维数是相对于输入空间而言的。维数由赋给 m_1 的值（即隐藏单元的个数）决定。

如前所述，通常将一个复杂的模式分类问题非线性地投射到高维数空间将会比投射到低维数空间更可能是线性可分的。但是需要强调的是，有时使用非线性映射（即第 1 点）就足够导致线性可分，而不必升高隐藏单元空间维数，如下面例子所说明的那样。

例 1　XOR 问题

为了说明模式的 φ 可分性思想的意义，考虑一个简单却又十分重要的 XOR 问题。在 XOR 问题中有四个二维输入空间上的点（模式）：(1,1),(0,1),(0,0) 和(1,0)，如图 5.2a 所示。要求建立一个模式分类器产生二值输出响应，其中点 (1,1) 或(0,0) 对应于输出 0，点 (1,0) 或(0,1) 对应于输出 1。因此在输入空间中依 Hamming 距离最近的点映射到在输出空间中最大分离的区域。一个序列的 Hamming 距离定义为二值序列中从符号 1 变为 0 的个数，反之亦然。因此，11 和 00 的 Hamming 距离是 0，01 和 10 的 Hamming 距离为 1。

定义一对高斯隐藏函数如下：

$$\varphi_1(\mathbf{x})=\exp(-\|\mathbf{x}-t_1\|^2),\qquad t_1=[1,1]^T$$
$$\varphi_2(\mathbf{x})=\exp(-\|\mathbf{x}-t_2\|^2),\qquad t_2=[0,0]^T$$

这样我们可以得到 4 个不同的输入模式作为输入时的结果，如表 5.1 所示。如图 5.2b 所示，输入模式被映射到 (φ_1,φ_2) 平面上。这里可以看到输入 (0,1),(1,0) 与剩下的两个输入 (1,1),(0,0) 是线性可分的。然后，我们将 $\varphi_1(\mathbf{x})$和 $\varphi_2(\mathbf{x})$作为一个线性分类器（如感知器）模型的输入，则 XOR 问题就迎刃而解了。 ■

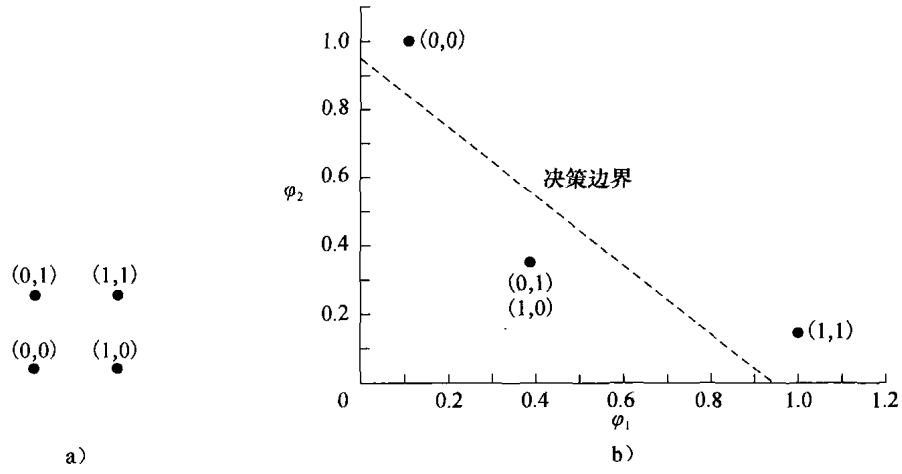


图 5.2 a) XOR 问题的 4 个模式；b) 决策图

表 5.1 用于例 1 的 XOR 问题的隐藏函数设置

输入模式 x	第一隐藏函数 $\varphi_1(\mathbf{x})$	第二隐藏函数 $\varphi_2(\mathbf{x})$	输入模式 x	第一隐藏函数 $\varphi_1(\mathbf{x})$	第二隐藏函数 $\varphi_2(\mathbf{x})$
(1,1)	1	0.135 3	(0,0)	0.135 3	1
(0,1)	0.367 8	0.367 8	(1,0)	0.367 8	0.367 8

在这个例子中隐藏空间的维数相对于输入空间并没有增加。也就是说，以高斯函数作为非线性的隐藏函数，足以将 XOR 问题转化为一个线性可分问题。

曲面的分离能力

式(5.5)对于在多维空间中随机指定输入模式线性可分的期望最大数目有重要意义。为了研究这个问题，如前所述将 $\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_N$ 视为一个随机模式（向量）序列。令 N 为一个随机变量，定义为该序列为 φ 可分时的最大整数，这里 φ 具有 m_1 的自由度。于是由式(5.5)可以导出当 $N=n$ 时的概率

$$\text{Prob}(N=n)=P(n,m_1)-P(n+1,m_1)=\left(\frac{1}{2}\right)^n\binom{n-1}{m_1-1},n=0,1,2,\cdots \tag{5.6}$$

为了解释上述结果，我们回想一下负二项分布的定义。该分布相当于在一组重复的 Bernoulli 实验中有 r 次成功、k 次失败的概率。在这种概率实验中，每一次实验只有两种结果，不是成功就是失败，并且成功和失败的概率在整组实验中都是相同的。令 p 代表成功的概率，q 代表失败的概率，$p+q=1$。负二项分布定义（Feller，1968）如下：

$$f(k;r,p)=p^rq^k\binom{r+k-1}{k}$$

在 $p=q=\frac{1}{2}$（即成功和失败具有相等的概率）且 $k+r=n$ 的特殊情况下，负二项分布将变为

$$f\left(k;n-k,\frac{1}{2}\right)=\left(\frac{1}{2}\right)^n\binom{n-1}{k},n=0,1,2,\cdots$$

根据上述定义，我们现在可以看出由式(5.6)所表示的结果恰好是负二项分布，只不过右移了 m_1 个单位且具有参数 m_1 和 1/2。这样，N 相当于在一组抛硬币的实验中出现第 m_1 次失败的“等待时间”。随机变量 N 的期望及其中位数分别为

$$\mathbb{E}[N]=2m_1 \tag{5.7}$$

和

$$\text{median}[N] = 2m_1 \tag{5.8}$$

因此，可以得到 Cover 定理的一个推论，用著名的渐近结果的形式可表述如下：

一组随机指定的输入模式（向量）的集合在 m_1 维空间中线性可分，它的元素数目的最大期望等于 $2m_1$。

该结果表明，$2m_1$ 是对一族具有 m_1 维自由度的决策曲面的分离能力的自然定义。在一定程度上，一个曲面的分离能力与第 4 章讨论的 VC 维数的概念有着紧密的联系。

5.3　插值问题

从关于模式可分性的 Cover 定理得到的重要思想是：在解决一个非线性可分的模式分类问题时，如果将输入空间映射到一个新的维数足够高的空间中去，将会有助于问题的解决。基本说来是用一个非线性映射将一个非线性可分的分类问题转变为一个高概率的线性可分问题。同样，我们可以用非线性映射将一个复杂的非线性滤波问题转化为一个较简单的线性滤波问题。

现在考虑一个由输入层、一个隐藏层和只有一个输出单元的输出层组成的前馈网络。我们选择只有一个输出单元的输出层的目的主要是为了简化说明而又不失一般性。设计这个网络实现从输入空间到隐藏空间的一个非线性映射，随后从隐藏空间到输出空间则是线性映射。令 m_0 为输入空间的维数。这样从总体上看这个网络就相当于一个从 m_0 维输入空间到一维输出空间的映射，可以写成如下形式：

$$s: \mathbb{R}^{m_0} \to \mathbb{R}^1 \tag{5.9}$$

我们可以将映射 s 视为一个超曲面（图）$\Gamma \subset \mathbb{R}^{m_0+1}$，就好像可以将一个最基本的映射 $s: \mathbb{R}^1 \to \mathbb{R}^1$，其中 $s(x)=x^2$，视为 $\mathbb{R}^2$ 空间中的一条抛物线一样。超曲面 Γ 作为输入的函数是输出空间的一个多维曲面。在实际情况下，曲面 Γ 是未知的，并且训练数据中通常带有噪声。学习过程中的训练阶段和泛化阶段可叙述如下（Broomhead and Lowe，1988）：

- 训练阶段由曲面 Γ 的拟合过程的最优化构成，它根据以输入-输出样本（模式）形式呈现给网络的已知数据进行。
- 泛化阶段的任务就是在数据点之间进行插值，插值是在真实曲面 Γ 的最佳逼近的拟合过程产生的约束曲面上进行的。

这样我们将引出具有悠久历史的高维空间多变量插值理论（Davis，1963）。从严格意义上说，插值问题可以叙述如下：

给定一个包含 N 个不同点的集合 $\{\mathbf{x}_i \in \mathbb{R}^{m_0} \mid i = 1,2,\cdots,N\}$ 和相应的 N 个实数的一个集合 $\{d_i \in \mathbb{R}^1 \mid i = 1,2,\cdots,N\}$，寻找一个函数 $F: \mathbb{R}^N \to \mathbb{R}^1$ 满足下述插值条件：

$$F(\mathbf{x}_i) = d_i, \qquad i = 1,2,\cdots,N \tag{5.10}$$

对于这里所述的严格插值来说，插值曲面（即函数 F）必须通过所有的训练数据点。

径向基函数（RBF）技术就是要选择一个函数 F 具有下列形式：

$$F(\mathbf{x}) = \sum_{i=1}^{N} w_i \varphi(\| \mathbf{x} - \mathbf{x}_i \|) \tag{5.11}$$

其中 $\{\varphi(\| \mathbf{x} - \mathbf{x}_i \|) \mid i = 1,2,\cdots,N\}$ 是 N 个任意（一般是线性）函数的集合，称为径向基函数；$\| \cdot \|$ 表示范数，通常是欧几里得范数（Powell，1988）。已知数据点 $\mathbf{x}_i \in \mathbb{R}^{m_0}$ $(i = 1,2,\cdots,N)$ 是径向基函数的中心。

将式(5.10)的插值条件代入式(5.11)中，可以得到一组关于未知系数（权值）的展开 $\{w_i\}$ 的线性方程：

$$\begin{bmatrix} \varphi_{11} & \varphi_{12} & \cdots & \varphi_{1N} \\ \varphi_{21} & \varphi_{22} & \cdots & \varphi_{2N} \\ \vdots & \vdots & \vdots & \vdots \\ \varphi_{N1} & \varphi_{N2} & \cdots & \varphi_{NN} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_N \end{bmatrix} \tag{5.12}$$

其中

$$\varphi_{ij} = \varphi(\| \mathbf{x}_i - \mathbf{x}_j \|), \quad i,j = 1,2,\cdots,N$$

令

$$\begin{aligned} \mathbf{d} &= [d_1, d_2, \cdots, d_N]^T \\ \mathbf{w} &= [w_1, w_2, \cdots, w_N]^T \end{aligned} \tag{5.13}$$

上式中的 $N \times 1$ 向量 $\mathbf{d}$ 和 $\mathbf{w}$ 分别表示期望响应向量和线性权值向量，其中 N 表示训练样本的长度。令 $\boldsymbol{\Phi}$ 表示元素为 φ_{ij} 的 $N \times N$ 阶的矩阵：

$$\boldsymbol{\Phi} = \{\varphi_{ij}\}_{i,j=1}^{N} \tag{5.14}$$

我们称该矩阵为插值矩阵。于是式(5.12)可以写成如下紧凑形式：

$$\boldsymbol{\Phi}\mathbf{w} = \mathbf{x} \tag{5.15}$$

假设 $\boldsymbol{\Phi}$ 为非奇异矩阵，因此存在逆矩阵 $\boldsymbol{\Phi}^{-1}$。这样就可以从式(5.15)中解出权值向量 $\mathbf{w}$，表示为

$$\mathbf{w} = \boldsymbol{\Phi}^{-1}\mathbf{x} \tag{5.16}$$

问题的关键是：怎么能保证插值矩阵 $\boldsymbol{\Phi}$ 是非奇异的？

可以证明，对于大量径向基函数来说，在某种条件下，上述问题的答案可以由下面的重要定理给出。

Micchelli 定理

Micchelli(1986) 证明了如下定理：

如果 $\{\mathbf{x}_i\}_{i=1}^{N}$ 是 $\mathbb{R}^{m_0}$ 中 N 个互不相同的点的集合，则 $N \times N$ 阶的插值矩阵 $\boldsymbol{\Phi}$(第 ij 个元素是 $\varphi_{ij} = \varphi(\| \mathbf{x}_i - \mathbf{x}_j \|)$ 是非奇异的。

有大量的径向基函数满足 Micchelli 定理，包括下面三个在径向基函数网络中有重要地位的函数：

1. 多二次 (multiquadrics) 函数：

$$\varphi(r) = (r^2 + c^2)^{1/2} \quad \text{对某些 } c > 0 \text{ 及 } r \in \mathbb{R} \tag{5.17}$$

2. 逆多二次 (inverse multiquadrics) 函数：

$$\varphi(r) = \frac{1}{(r^2 + c^2)^{1/2}} \quad \text{对某些 } c > 0 \text{ 及 } r \in \mathbb{R} \tag{5.18}$$

3. 高斯函数：

$$\varphi(r) = \exp\left(-\frac{r^2}{2\sigma^2}\right) \quad \text{对某些 } \sigma > 0 \text{ 及 } r \in \mathbb{R} \tag{5.19}$$

多二次函数和逆多二次函数都应归功于 Hardy(1971)。

为了使式(5.17)至式(5.19)所示的径向基函数是非奇异的，必须使所有的输入点 $\{\mathbf{x}_i\}_{i=1}^{N}$ 互不相同。这就是使插值矩阵 $\boldsymbol{\Phi}$ 非奇异的全部要求，与所给样本的长度 N 和向量 (点)$\mathbf{x}_i$ 的维数 m_0 无关。

式(5.18)的逆多二次函数和式(5.19)的高斯函数具有一个共同的性质：它们都是局部化的函数，因为当 $r \to \infty$ 时，$\varphi(r) \to 0$。以上面两个函数作为径向基函数所组成的插值矩阵 $\boldsymbol{\Phi}$ 都是正定的。与此相反，由式(5.17)所定义的多二次函数是非局部化函数，因为当 $r \to \infty$ 时，$\varphi(r)$ 是无界的；与其相对应的插值矩阵 $\boldsymbol{\Phi}$ 有 $N-1$ 个负的特征值，只有一个正的特征值，所以不是

正定的（Micchelli，1986）。但值得注意的是，在 Hardy 的多二次函数基础上建立的插值矩阵 $\boldsymbol{\Phi}$ 却是非奇异的，因此适合在 RBF 网络设计中应用。

更加值得注意的是无限增长的径向基函数，例如多二次函数，与其他产生正定插值矩阵的函数相比，能以更高的精度逼近一个光滑的输入-输出映射。Powell(1988) 讨论了这个令人惊奇的结果。

5.4　径向基函数网络

受式(5.10)到式(5.16)的启发，现在我们可以预想一个多层结构形式的径向基函数（RBF）网络，如图 5.3 所示；具体有三层：

1. *输入层*，由 m_0 个源节点组成，其中 m_0 是输入向量 $\mathbf{x}$ 的维数。

2. *隐藏层*，由和训练样本的大小 N 相同个数的计算单元组成，每个单元都从数学上用一个径向基函数来描述：

$$\varphi_j(\mathbf{x}) = \varphi(\| \mathbf{x} - \mathbf{x}_j \|), \qquad j = 1,2,\cdots,N$$

第 j 个输入数据点 $\mathbf{x}_j$ 定义了该径向基函数的中心，向量 $\mathbf{x}$ 是作用于输入层的信号（模式）。因此，与多层感知器不同，源节点和隐藏单元的连接是直接连接，没有权值。

3. *输出层*，在图 5.3 的 RBF 结构中由单一计算单元构成。很明显，除了一般情况下输出层的大小比隐藏层的大小要小得多之外，对于输出层的大小没有限制。

自此之后，我们重点关注高斯函数作为径向基函数的使用，在这样的情形下，图 5.3 中隐藏层的每个计算单元可以定义为：

$$\varphi_j(\mathbf{x}) = \varphi(\mathbf{x} - \mathbf{x}_j) = \exp\left(-\frac{1}{2\sigma_j^2} \| \mathbf{x} - \mathbf{x}_j \|^2\right), \qquad j = 1,2,\cdots,N \tag{5.20}$$

其中 σ_j 是第 j 个以 $\mathbf{x}_j$ 为中心的高斯函数的宽的测量。一般情况下，*高斯隐藏单元被分配给一个共用的宽* σ。在这一类情形下，将隐藏单元区分开的参数是中心 $\mathbf{x}_j$。在建立 RBF 网络时选择高斯函数作为径向基函数背后的基本原理是它具有多个所希望的性质，随着讨论的进行这些性质将变得很明显。

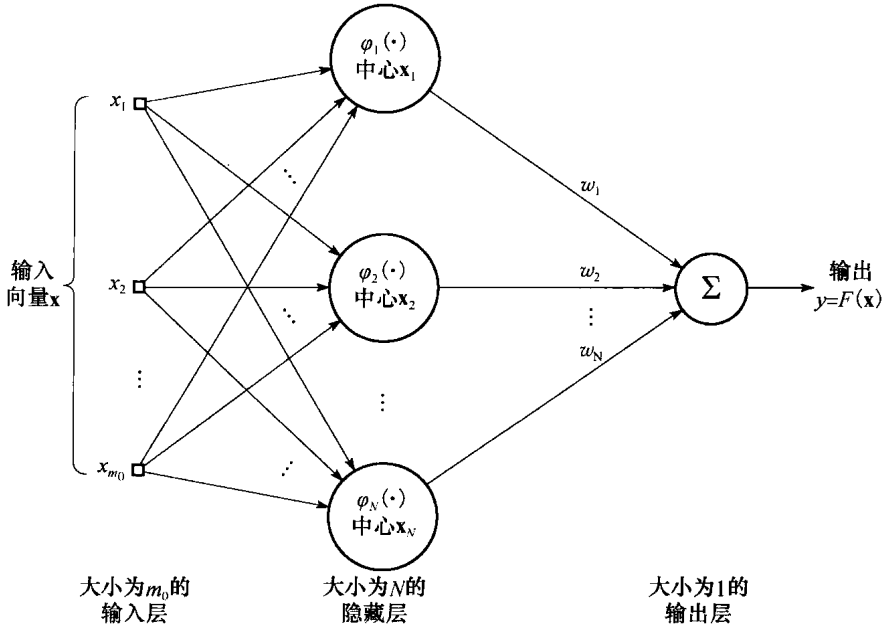


图 5.3　RBF 网络结构，基于插值理论

RBF 网络的实际修正

图 5.3 给出的通过插值理论的 RBF 网络形式非常整洁。然而在实际中，我们发现在模式识别或者非线性回归的背景下训练样本 $\{\mathbf{x}_i, d_i\}_{i=1}^{N}$ 通常是含噪声的。遗憾的是，基于噪声数据使用插值将导致引入歧途的结果——因此需要 RBF 网络的不同设计途径。

需要注意的另一个实际问题是：使隐藏层具有和输入样本个数相同的大小可能导致计算资源的浪费，尤其是处理大规模训练样本时。当 RBF 网络的隐藏层是由式(5.20)描述的方式所指定时，我们发现在训练样本中毗连数据点之间存在的相关性相应地移植到了隐藏层的毗连单元上。换种方式讲，当通过式(5.20)选择隐藏层神经元时，由于训练样本中可能存在的固有冗余，隐藏层神经元也具有冗余。在这种情况下，使得隐藏层的大小是训练样本大小的一部分因而是一个好的设计实践，如图 5.4 所示。注意到尽管图 5.3 和图 5.4 的 RBF 网络是确实不同的，但它们有一个共同的特征：与多层感知器的情况不同，RBF 网络的训练不包括误差信号的反向传播。

通过这两个 RBF 结构实现的逼近函数具有相同的数学形式：

$$F(\mathbf{x}) = \sum_{j=1}^{K} w_j \varphi(\mathbf{x}, \mathbf{x}_j) \tag{5.21}$$

其中输入向量 $\mathbf{x}$ 的（因此是输入层的）维数是 m_0，每个隐藏单元由径向基函数 $\varphi(\mathbf{x}, \mathbf{x}_j)$ 来刻画，其中 $j = 1, 2, \cdots, K$，K 比 N 要小。输出层假设由单一单元组成，由权值向量 $\mathbf{w}$ 来刻画，其维数也是 K。图 5.3 和图 5.4 的结构在两个方面有所不同：

1. 在图 5.3 中，隐藏层的维数是 N，这里 N 是训练集的大小，而图 5.4 中隐藏层维数 $K<N$。

2. 假设训练样本 $\{\mathbf{x}_i, d_i\}_{i=1}^{N}$ 是无噪声的，图 5.3 的隐藏层的设计可通过简单地利用输入向量 $\mathbf{x}_j$ 来定义径向基函数 $\varphi(\mathbf{x}, \mathbf{x}_j)(j = 1, 2, \cdots, N)$ 的中心。而为了设计图 5.4 中的隐藏层，我们需要讨论新的过程。

下一节将对于隐藏层使用高斯函数的情况，从实际的角度说明上述的第 2 点。

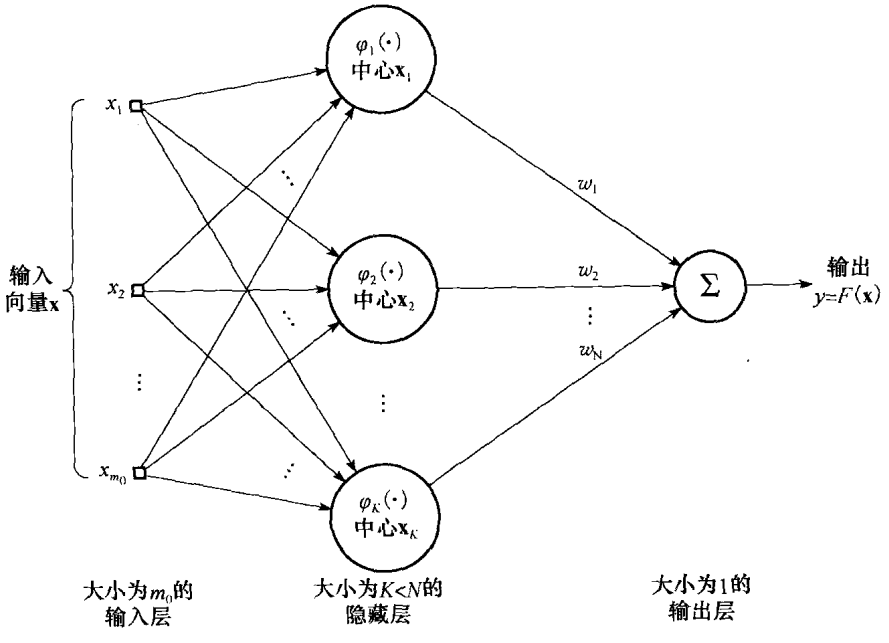


图 5.4　实际 RBF 网络的结构。注意这一网络从结构上和图 5.3 相似，然而两个网络是不同的，图 5.4 中隐藏层的大小小于图 5.3 中隐藏层的大小

5.5　K-均值聚类

在设计图 5.4 的 RBF 网络时，需要解决的一个关键问题是如何利用无标签数据来计算构成隐藏层的高斯单元的参数。换句话说，这一计算是在非监督方式下完成的。在本节中，我们描述关于这一问题的根植于聚类的一个解，其意义如下：

聚类是非监督学习的一种形式，它将一个观测集（即数据点）划分到自然组或者模式聚类。聚类的途径是测量分配给每个聚类的观测对之间的相似性以最小化一个指定的代价函数。

有很多聚类技术可供选择。我们选择重点关注所谓的 K-均值（K-means）的算法[3]，因为它简单易实现，同时有良好的性能，以上两个特征使得该算法高度普及。

令 $\{\mathbf{x}_i\}_{i=1}^{N}$ 表示一个用于划分到 K 个聚类的多维观测集，其中 K 小于观测数 N。令关系：

$$j = C(i), \qquad i = 1,2,\cdots,N \tag{5.22}$$

表示一个多对一映射器，称为编码器，它将第 i 个观测 $\mathbf{x}_i$ 根据某种仍然需要定义的规则分配到第 j 个聚类中。（细心的读者会奇怪为什么我们选择索引 j 来表示一个聚类，而合乎逻辑的选择应该是 k；这个选择的理由是符号 k 被用于表示将在本章后面讨论的核函数。）为了进行这样的编码，我们需要在向量 $\mathbf{x}_i$ 和 $\mathbf{x}_{i'}$ 对之间的相似性度量，记为 $d(\mathbf{x}_i,\mathbf{x}_{i'})$。当测度 $d(\mathbf{x}_i,\mathbf{x}_{i'})$ 足够小的时候，$\mathbf{x}_i$ 和 $\mathbf{x}_{i'}$ 被分配给相同的聚类；否则，它们被分配给不同的聚类。

为了最优化这个聚类过程，我们引入下面的代价函数（Hastie 等，2001）：

$$J(C) = \frac{1}{2}\sum_{j=1}^{K}\sum_{C(i)=j}\sum_{C(i')=j} d(\mathbf{x}_i,\mathbf{x}_{i'}) \tag{5.23}$$

对于预先指定的 K，要求找到使得代价函数 $J(C)$ 最小的编码器 $C(i)=j$。在讨论中，我们注意到编码器 C 是未知的——因此代价函数 J 依赖于 C。

在 K-均值聚类中，欧几里得范数的平方用于定义在观测 $\mathbf{x}_i$ 和 $\mathbf{x}_{i'}$ 之间的相似性度量，如下所示：

$$d(\mathbf{x}_i,\mathbf{x}_{i'}) = \|\mathbf{x}_i - \mathbf{x}_{i'}\|^2 \tag{5.24}$$

因此，将式(5.24)代入式(5.23)中，我们有

$$J(C) = \frac{1}{2}\sum_{j=1}^{K}\sum_{C(i)=j}\sum_{C(i')=j}\|\mathbf{x}_i - \mathbf{x}_{i'}\|^2 \tag{5.25}$$

现在给出如下两点：

1. 观测 $\mathbf{x}_i$ 和 $\mathbf{x}_{i'}$ 之间的欧几里得距离的平方是对称的，即

$$\|\mathbf{x}_i - \mathbf{x}_{i'}\|^2 = \|\mathbf{x}_{i'} - \mathbf{x}_i\|^2$$

2. 式(5.25)的内部和可以如下解释：对于给定的 $\mathbf{x}_{i'}$，编码器 C 将所有和 $\mathbf{x}_i$ 最近的观测 $\mathbf{x}_{i'}$ 分配给聚类 j。除了一个尺度因子，这样分配的观测 $\mathbf{x}_{i'}$ 的和是属于聚类 j 的均值向量估计；这里的尺度因子是 $1/N_j$，其中 N_j 是聚类 j 中数据点的个数。

由于这两点，可以将式(5.25)简化为：

$$J(C) = \sum_{j=1}^{K}\sum_{C(i)=j}\|\mathbf{x}_i - \hat{\boldsymbol{\mu}}_j\|^2 \tag{5.26}$$

这里 $\hat{\boldsymbol{\mu}}_j$ 记为属于聚类 j 的“估计”均值向量[4]。实际上，均值 $\hat{\boldsymbol{\mu}}_j$ 可以看成是聚类 j 的中心。受式(5.26)的启发，现在可以将聚类问题重新描述如下：

给定 N 个观测值的集合，通过以下方式寻找编码器 C：将这些观测值分配给 K 个聚类，使得在每个聚类中，给定的观测值与聚类均值的不相似性的平均度量最小。

当然，正是由于这一陈述的本质，这里所描述的聚类技术通常称为 K-均值算法。

对于式(5.26)中定义的代价函数 $J(C)$ 的解释，我们可以这样说，对于给定的编码器 C，除了尺度因子 $1/N_j$，这一等式的内部和是聚类 j 所属观测的方差的估计，如下所示：

$$\hat{\sigma}_j^2 = \sum_{C(i)=j} \| \mathbf{x}_i - \hat{\boldsymbol{\mu}}_j \|^2 \tag{5.27}$$

相应地，可以将代价函数 $J(C)$ 看作通过编码器 C 将所有 N 个观测分配给 K 个聚类的总聚类方差的测度。

当编码器 C 未知时，如何最小化代价函数 $J(C)$？为了解决这一关键问题，我们利用迭代下降算法，这一算法的每一次迭代包含两步最优化。第一步对于给定的编码器 C 利用最邻近规则来最小化式(5.26)的代价函数 $J(C)$(关于均值向量 $\hat{\boldsymbol{\mu}}_j$)。第二步对给定的均值向量 $\hat{\boldsymbol{\mu}}_j$ 最小化式(5.26)的内部和（关于编码器 C)。连续进行这样的两步迭代过程直到收敛为止。

因此，从数学上，K-均值算法分两步进行[5]：

第 1 步　对于给定的编码器 C，关于聚类均值 $\{\hat{\boldsymbol{\mu}}_j\}_{j=1}^K$ 最小化总聚类方差；即完成下面的最小化：

$$\min_{\{\hat{\boldsymbol{\mu}}_j\}_{j=1}^K} \sum_{j=1}^{K} \sum_{C(i)=j} \| \mathbf{x}_i - \hat{\boldsymbol{\mu}}_j \|^2 \qquad \text{对于给定的 } C \tag{5.28}$$

第 2 步　在第 1 步中已经计算了最优聚类均值 $\{\hat{\boldsymbol{\mu}}_j\}_{j=1}^K$，下一步最小化编码器：

$$C(i) = \arg \min_{1 \leqslant j \leqslant K} \| \mathbf{x}(i) - \hat{\boldsymbol{\mu}}_j \|^2 \tag{5.29}$$

从一些最初选择的编码器 C 开始，算法在这两步之间来回进行，直到在聚类分配上没有进一步的变化为止。

这两步中的每一步都被设计为按其自身的方式降低代价函数 $J(C)$；因此，算法的收敛性是可以保证的。然而，由于算法缺少全局最优准则，结果可能收敛于局部最小值，导致对聚类分配的次优解。无论如何，这一算法具有实际上的优势：

1. K-均值算法是计算有效的，其计算复杂度对于聚类数目而言是线性的。

2. 当聚类紧致分布在数据空间中时，它们由算法忠实再现。

最后，为了初始化 K-均值算法，建议采用以下步骤：对建议的大小 K，对于均值 $\{\hat{\boldsymbol{\mu}}_j\}_{j=1}^K$，随机选择不同的值来开始算法，然后选择使得式(5.26)中的双重和具有最小值的集合(Hastie 等，2001)。

K-均值算法适用于 Cover 定理框架

K-均值算法对输入信号 $\mathbf{x}$ 应用了非线性变换。我们这样说是因为其不相似测度（即这一算法的基础，欧几里得距离的平方 $\| \mathbf{x} - \mathbf{x}_j \|^2$）是对于给定的聚类中心 $\mathbf{x}_j$ 而言关于输入信号 $\mathbf{x}$ 的非线性函数。而且，由 K-均值算法揭示的每个聚类定义了隐藏层的一个特殊的计算单元，如果聚类数目 K 足够大，K-均值算法将满足 Cover 定理的其他要求，即隐藏层维数足够高。因此得出结论：根据这一定理，K-均值算法确实有足够的计算能力将非线性的可分离模式集合转化为可分离模式。

现在这一目的已经得到了满足，我们就可以考虑设计 RBF 网络的线性输出层了。

5.6　权向量的递归最小二乘估计

K-均值算法的计算是用递归方式来实现的。因此需要重做最小二乘法——在第 2 章中讨论过——对 RBF 网络输出层的权重向量的计算，也用递归的方式来实现。为了这一目的，我们将式(2.23)改写为以下形式

$$\mathbf{R}(n)\,\hat{\mathbf{w}}(n) = \mathbf{r}(n), \qquad n = 1,2,\cdots, \tag{5.30}$$

这里所有三个量都表达为离散时间 n 的函数。在书写这个统计学上称为*法方程*的时候，我们引入了三个项：

1. 隐藏单元输出的 $K\times K$ 相关函数，由下式定义：

$$\mathbf{R}(n)=\sum_{i=1}^{n}\boldsymbol{\phi}(\mathbf{x}_i)\,\boldsymbol{\phi}^T(\mathbf{x}_i) \tag{5.31}$$

其中

$$\boldsymbol{\phi}(\mathbf{x}_i)=[\varphi(\mathbf{x}_i,\mathbf{x}_1),\varphi(\mathbf{x}_i,\mathbf{x}_2),\cdots,\varphi(\mathbf{x}_i,\mathbf{x}_K)]^T \tag{5.32}$$

和

$$\varphi(\mathbf{x}_i,\mathbf{x}_j)=\exp\left(-\frac{1}{2\sigma_j^2}\|\mathbf{x}_i-\mathbf{x}_j\|^2\right),j=1,2,\cdots,K \tag{5.33}$$

2. RBF 网络输出的期望响应和隐藏单元输出之间的 $K\times 1$ *互相关向量*，定义为：

$$\mathbf{r}(n)=\sum_{i=1}^{n}\boldsymbol{\phi}(\mathbf{x}_i)d(i) \tag{5.34}$$

3. *未知权值向量*$\hat{\mathbf{w}}(n)$，在最小二乘下最优化。

要求对权值向量 $\mathbf{w}(n)$求解式(5.30)的法方程。当然，可以首先计算相关矩阵 $\mathbf{R}(n)$的逆矩阵，然后将求得的逆矩阵 $\mathbf{R}^{-1}(n)$和互相关向量 $\mathbf{r}(n)$相乘，这就是最小二乘法所做的。然而，当隐藏层大小 K 很大时，通常情形下对于 $n=K$ 计算逆矩阵 $\mathbf{R}^{-1}(n)$是一个吃力的任务。所计划的运用最小二乘法的递归执行将应对这一计算困难。其结果算法称为递归最小二乘（RLS）算法[6]，关于这一算法的推导将在下面讨论。

RLS 算法

通过重新组织式(5.34)的互相关向量 $\mathbf{r}(n)$来开始推导 RLS 算法，如下所示：

$$\begin{aligned}\mathbf{r}(n)&=\sum_{i=1}^{n-1}\boldsymbol{\phi}(\mathbf{x}_i)d(i)+\boldsymbol{\phi}(\mathbf{x}_n)d(n)=\mathbf{r}(n-1)+\boldsymbol{\phi}(\mathbf{x}_n)d(n)\\&=\mathbf{R}(n-1)\,\hat{\mathbf{w}}(n-1)+\boldsymbol{\phi}(\mathbf{x}_n)d(n)\end{aligned} \tag{5.35}$$

其中，在第一行将相对于 $i=n$ 的项从式(5.34)的和中独立出来，最后一行利用了式(5.30)，用 $n-1$ 来代替 n。然后，在式(5.35)的右边有目的地加上项$\boldsymbol{\phi}(n)\boldsymbol{\phi}^T(n)\hat{\mathbf{w}}(n-1)$并在等式的另一部分减去相同项，使得方程本身没有改变；因此可以写出（在提取公共因子后）：

$$\mathbf{r}(n)=[\mathbf{R}(n-1)+\boldsymbol{\phi}(n)\,\boldsymbol{\phi}^T(n)]\,\hat{\mathbf{w}}(n-1)+\boldsymbol{\phi}(n)[d(n)-\boldsymbol{\phi}^T(n)\,\hat{\mathbf{w}}(n-1)] \tag{5.36}$$

在式(5.36)右边第一个方括号中的表达式被认为是相关函数：

$$\mathbf{R}(n)=\mathbf{R}(n-1)+\boldsymbol{\phi}(n)\,\boldsymbol{\phi}^T(n) \tag{5.37}$$

在式(5.36)右边第二个方括号中的表达式中，引入了一个新的项：

$$\alpha(n)=d(n)-\boldsymbol{\phi}^T(n)\mathbf{w}(n-1)=d(n)-\mathbf{w}^T(n-1)\,\boldsymbol{\phi}(n) \tag{5.38}$$

这一新的项称为*先验估计误差*，这里使用“先验”是为了强调估计误差 $\alpha(n)$是基于权值向量$\hat{\mathbf{w}}(n-1)$的老的估计的（即在权值估计被更新“之前”）。$\alpha(n)$也称为“革新”，因为嵌入$\boldsymbol{\phi}(n)$中的输入向量 $\mathbf{x}(n)$和其相应的期望响应 $d(n)$表示第 n 步时间估计时作用于算法的“新”信息。

回到式(5.36)，利用式(5.37)和式(5.38)将问题简化为：

$$\mathbf{r}(n)=\mathbf{R}(n)\,\hat{\mathbf{w}}(n-1)+\boldsymbol{\phi}(n)\alpha(n) \tag{5.39}$$

相应地，将这一方程应用到式(5.30)中得到：

$$\mathbf{R}(n)\,\hat{\mathbf{w}}(n)=\mathbf{R}(n)\,\hat{\mathbf{w}}(n-1)+\boldsymbol{\phi}(n)\alpha(n) \tag{5.40}$$

这可以表达为更新权值的期望形式，如下所示：

$$\hat{\mathbf{w}}(n)=\hat{\mathbf{w}}(n-1)+\mathbf{R}^{-1}(n)\,\boldsymbol{\phi}(n)\alpha(n) \tag{5.41}$$

这里在式(5.40)的两边同时乘以逆矩阵 $\mathbf{R}^{-1}(n)$。然而，为了以计算有效方式来实现这一更新，

我们需要相应的通过给定其过去值 $\mathbf{R}^{-1}(n-1)$来计算逆矩阵 $\mathbf{R}^{-1}(n)$的公式。这一问题将在下面讨论。

计算 $\mathbf{R}^{-1}(n)$的递归公式

回到式(5.37)，可以看到确实有一个公式递归地更新相关矩阵 $\mathbf{R}(n)$。我们关注这一递归式，通过利用矩阵逆引理得到逆矩阵 $\mathbf{R}^{-1}(n)$的递归形式，而矩阵逆引理已经在 4.14 节讨论过了。

作为扼要概述，考虑矩阵：

$$\mathbf{A} = \mathbf{B}^{-1} + \mathbf{C}\mathbf{D}\mathbf{C}^T \tag{5.42}$$

这里假设矩阵 $\mathbf{B}$ 是非奇异的且矩阵 $\mathbf{B}^{-1}$因而存在。矩阵 $\mathbf{A}$ 和 $\mathbf{B}$ 具有相同维数，矩阵 $\mathbf{D}$ 是另一个具有不同维数的非奇异矩阵，矩阵 $\mathbf{C}$ 是具有合适维数的矩形矩阵。根据矩阵逆引理，得到：

$$\mathbf{A}^{-1} = \mathbf{B} - \mathbf{B}\mathbf{C}(\mathbf{D} + \mathbf{C}^T\mathbf{B}\mathbf{C})^{-1}\mathbf{C}^T\mathbf{B} \tag{5.43}$$

对于这个问题，使用式(5.37)来做如下标示：

$$\mathbf{A} = \mathbf{R}(n)$$
$$\mathbf{B}^{-1} = \mathbf{R}(n-1)$$
$$\mathbf{C} = \boldsymbol{\phi}(n)$$
$$\mathbf{D} = 1$$

相应地，式(5.43)作用于这一矩阵特殊集就产生：

$$\mathbf{R}^{-1}(n) = \mathbf{R}^{-1}(n-1) - \frac{\mathbf{R}^{-1}(n-1)\,\boldsymbol{\phi}(n)\,\boldsymbol{\phi}^T(n)\mathbf{R}^{-1}(n-1)}{1 + \boldsymbol{\phi}(n)\mathbf{R}^{-1}(n-1)\,\boldsymbol{\phi}(n)} \tag{5.44}$$

这里，在方程右端第二项，我们利用了相关矩阵的对称性；即

$$\mathbf{R}^T(n-1) = \mathbf{R}(n-1)$$

为了简化 RLS 算法的公式，我们现在介绍两个新的定义：

1. $\mathbf{R}^{-1}(n) = \mathbf{P}(n)$

因此，将式(5.44)重写为：

$$\mathbf{P}(n) = \mathbf{P}(n-1) - \frac{\mathbf{P}(n-1)\,\boldsymbol{\phi}(n)\,\boldsymbol{\phi}^T(n)\mathbf{P}(n-1)}{1 + \boldsymbol{\phi}^T(n)\mathbf{P}(n-1)\,\boldsymbol{\phi}(n)} \tag{5.45}$$

这里方程右边的分母是二次形式因而是一个标量。

为了说明 $\mathbf{P}(n)$，考虑线性回归模型：

$$d(n) = \mathbf{w}^T\boldsymbol{\phi}(n) + \varepsilon(n)$$

作为期望响应 $d(n)$的一般模型，$\boldsymbol{\phi}(n)$是回归量。假设附加噪声项 $\varepsilon(n)$为白噪，具有 0 均值和 σ_ε^2 的方差。然后，将未知权值向量 $\mathbf{w}$ 看成模型的状态且$\hat{\mathbf{w}}(n)$是由 RLS 算法产生的估计，定义状态误差协方差矩阵如下：

$$\mathbb{E}[(\mathbf{w} - \hat{\mathbf{w}}(n))(\mathbf{w} - \hat{\mathbf{w}}(n))^T] = \sigma_\varepsilon^2\mathbf{P}(n) \tag{5.46}$$

对于这一结果的证明在习题 5.5 中给出。

2. $\mathbf{g}(n) = \mathbf{R}^{-1}(n)\,\boldsymbol{\phi}(n) = \mathbf{P}(n)\,\boldsymbol{\phi}(n)$ (5.47)

新的项 $\mathbf{g}(n)$称为 RLS 算法的增益向量 (gain vector)，因为，根据式(5.41)可以将先验估计误差 $\alpha(n)$和 $\mathbf{g}(n)$的乘积看成是将老的估计$\hat{\mathbf{w}}(n-1)$更新到新值$\hat{\mathbf{w}}(n)$的校正，如下所示：

$$\hat{\mathbf{w}}(n) = \hat{\mathbf{w}}(n-1) + \mathbf{g}(n)\alpha(n) \tag{5.48}$$

RLS 算法小结[7]

有了式 (5.45) 、式(5.47) 、式(5.38) 和式(5.48) ，根据这样的顺序，现在可以给出如下 RLS 算法的小结：

给定训练样本 $\{\boldsymbol{\phi}(i), d(i)\}_{i=1}^N$，对 $n = 1,2,\cdots,N$ 进行如下计算：

$$\mathbf{P}(n) = \mathbf{P}(n-1) - \frac{\mathbf{P}(n-1)\,\boldsymbol{\phi}(n)\,\boldsymbol{\phi}^T(n)\mathbf{P}(n-1)}{1+\boldsymbol{\phi}^T(n)\mathbf{P}(n-1)\,\boldsymbol{\phi}(n)}$$

$$\mathbf{g}(n) = \mathbf{P}(n)\,\boldsymbol{\phi}(n)$$

$$\alpha(n) = d(n) - \hat{\mathbf{w}}^T(n-1)\,\boldsymbol{\phi}(n)$$

$$\hat{\mathbf{w}}(n) = \hat{\mathbf{w}}(n-1) + \mathbf{g}(n)\alpha(n)$$

为了初始化这一算法，令

$$\hat{\mathbf{w}}(0) = \mathbf{0}$$

且

$$\mathbf{P}(0) = \lambda^{-1}\mathbf{I}, \qquad \lambda \text{ 是小的正常数}$$

注意算法初始化中使用的 λ 提供了代价函数中正则化参数的规则：

$$\mathscr{E}_{av}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}(d(i) - \mathbf{w}^T\boldsymbol{\phi}(i))^2 + \frac{1}{2}\lambda\|\mathbf{w}\|^2$$

这里 λ 选择相对较小的数，这是一个典型的情况，然后，我们非直接地确认训练样本 $\{\mathbf{x}(i), d(i)\}_{i=1}^N$ 的质量。

5.7　RBF 网络的混合学习过程

根据 5.5 节介绍的 K-均值聚类算法和 5.6 节推导的递归最小二乘（RLS）算法，现在我们可以给出图 5.4 的 RBF 网络的混合学习过程[8]。首先将 K-均值算法用于训练隐藏层；然后利用 RLS 算法来训练输出层。此后，我们将这一混合学习过程称为“K-均值，RLS”算法，目的是用下面的过程来训练 RBF 网络。

输入层　输入层的大小是由输入向量 $\mathbf{x}$ 的维数决定的，记为 m_0。

隐藏层

1. 隐藏层的大小 m_1 是由计划的聚类数 K 决定的。事实上，参数 K 可以看成是在设计者控制下的自由度。因此，参数 K 是模型选择问题的关键并因而不仅控制着性能而且控制着网络的计算复杂度。

2. 聚类均值 $\hat{\boldsymbol{\mu}}_j$，由工作于输入向量作为无标志样本 $\{\mathbf{x}_i\}_{i=1}^N$ 之上的 K-均值算法来计算，决定了分配给隐藏单元 $j = 1,2,\cdots,K$ 的高斯函数 $\varphi(\cdot,\mathbf{x}_j)$ 的中心 $\mathbf{x}_j$。

3. 为了简化设计，记为 σ 的相同的宽度被分配给所有的高斯函数，和 K-均值算法揭示的中心的散布相一致，如下所示：

$$\sigma = \frac{d_{\max}}{\sqrt{2K}} \tag{5.49}$$

这里 K 是中心个数，$d_{\max}$ 是它们之间的最大距离（Lowe，1989）。这一公式保证了各个高斯单元不是太尖峰也不是太平坦；这两种极端情况在实际中都可以得到避免。

输出层　一旦隐藏层的训练完成后，就可以开始输出层的训练了。令 $K\times 1$ 向量

$$\boldsymbol{\phi}(\mathbf{x}_i) = \begin{bmatrix} \varphi(\mathbf{x}_i,\mathbf{x}_1) \\ \varphi(\mathbf{x}_i,\mathbf{x}_2) \\ \vdots \\ \varphi(\mathbf{x}_i,\mathbf{x}_K) \end{bmatrix}$$

定义为隐藏层 K 个单元的输出。这个向量是响应于刺激 $x_i, i = 1,2\cdots,N$ 而产生的。因此，目前所考虑过的输出层的监督训练中，训练样本可定义为 $\{\boldsymbol{\phi}(\mathbf{x}_i), d_i\}_{i=1}^N$，其中 d_i 是对输入 $\mathbf{x}_i$ 的 RBF 网络的总输出的期望响应。这一训练是通过 RLS 算法来进行的。一旦网络训练完成，就可以开始使用没有出现过的数据来测试整个网络。

“K-均值，RLS”算法的一个有吸引力的特征是它的计算高效性，这是由于 K-均值和 RLS 算法都在其各自的方式上是计算高效的这一事实。这一算法唯一可疑的特征是缺少将隐藏层的训练和输出层的训练结合起来的总的最优准则，从而在统计意义上保证整个系统的最优性。

5.8　计算机实验：模式分类

本节我们用一个计算机实验来评价用于训练 RBF 网络的“K-均值，RLS”算法的模式分类性能。这一实验所使用的数据是通过对图 1.8 的双月结构随机采样而获得的。这一实验的具体目的是比较其性能：通过这一途径训练的网络的性能和通过利用反向传播算法训练的多层感知器（MLP）的性能。而 MLP 的性能已经在第 4.7 节完成的实验中集中讨论了。

RBF 网络的隐藏层选择包含 20 个高斯单元，因此这和第 4.7 节中 MLP 的隐藏层采用了相同的大小。为了训练 RBF 网络，使用了 1 000 个数据点；对于测试，使用了 2 000 个数据点。与 MLP 实验的方式相似，对两个不同的双月图设置，$d=-5$ 和 $d=-6$，进行 RBF 实验，后者是这两者中更难的一个。

(a) 垂直分隔：$d=-5$

对于这个在两月之间的垂直分隔，$K=20$ 被分配给聚类数（即隐藏单元个数）。通过应用 K-均值算法作用于训练样本的无标志部分，聚类的中心以及因此而来的隐藏层中高斯单元的中心就被决定了。由于中心的散布是已知的，则利用式(5.49)的公式来计算分配给高斯单元的共同宽 $\sigma=2.6$。RBF 网络隐藏层的设计就完成了。最后，RLS 算法被用于训练输出层，从而计算决策边界，为测试阶段准备好了途径。

实验第一部分的结果在图 5.5 中给出。图 5.5a 给出了 RLS 算法的学习曲线，图 5.5b 给出了 RBF 网络所学习的决策边界。如图 5.5a 所示，在两个回合的训练之后，输出层的设计就完成了。图 5.5b 确认了 RBF 网络几乎能够完美地将两个月亮形状的模式分离开。

(b) 垂直分隔：$d=-6$

然后重复 RBF 网络关于模式分类的实验，这一次对于图 1.8 的双月结构给出了更加困难的设置。这次，共同宽 $\sigma=2.4$ 被分配给 20 个高斯单元，该分配再次根据式(5.49)给出。

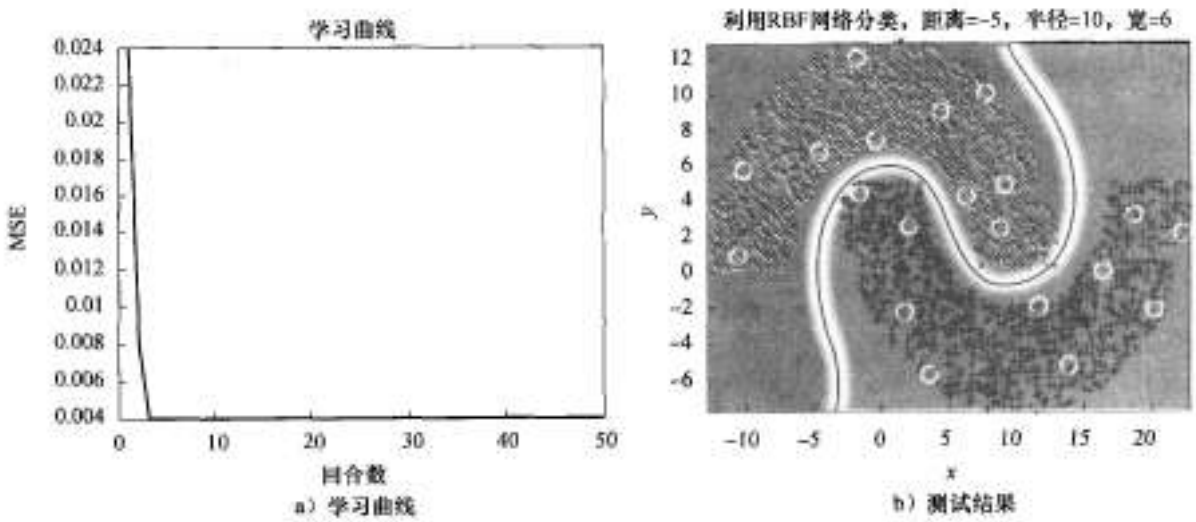


图 5.5　当 $d=-5$ 时用 K-均值和 RLS 算法训练的 RBF 网络。图 a 中的 MSE 表示均方误差

图 5.6 中给出实验第二部分的结果，图 5.6a 给出了 RLS 算法的学习曲线，图 5.6b 给出了在“K-均值，RLS”算法训练结果下 RBF 网络学习的决策边界。在 2 000 个测试数据点中总

共报告有 10 个分类错误，产生的识别误差率是 0.5%。

对于实验的（a）部分和（b）部分，在 RBF 网络的单一输出处的分类阈值被设为零。

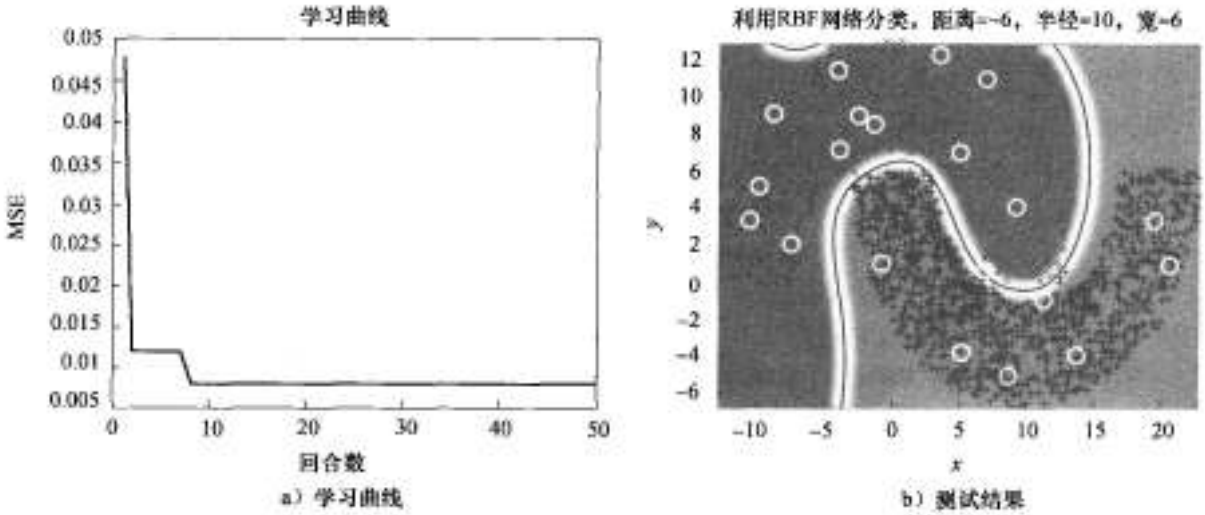


图 5.6　当 $d=-6$ 时用 K-均值和 RLS 算法训练的 RBF 网络。图 5.6a 中的 MSE 表示均方误差

比较 MLP 和 RBF 结果

将本节中 RBF 网络完成的实验（a）和（b）的结果和 4.7 节中 MLP 完成的对应的实验结果相比较，我们得出如下结论：

1. 用“K-均值，RLS”算法训练的 RBF 网络胜过用反向传播算法训练的 MLP。具体来说，当双月结构中 $d=-5$ 时 MLP 不能达到完美分类，而 RBF 网络报告了近乎完美的分类。对于困难的设置 $d=-6$，RBF 网络产生的误识率是 0.5%，比 MLP 算法对于容易的设置 $d=-5$ 时所得到的误识率 0.15%要稍微差一些。当然，MLP 的设计可以得到改进。然而，我们同样可以说 RBF 网络也可以得到改进。

2. RBF 网络的训练过程明显比 MLP 的训练过程快。

5.9　高斯隐藏单元的解释

感受野思想

在神经生物学中，*感受野*（receptive field）定义为“感觉场的区域，其中充分的感觉刺激将引起响应”（Churchland and Sejnowski，1992）。一个有趣的现象是，在视皮层的更高区域中细胞的感受野倾向于远大于视觉系统的早期阶段的细胞。

根据感受野这一神经生物学上的定义，我们可以想象每个神经网络的隐藏单元都具有一个其自身的感受野。实际上，我们可以继续做如下对应的陈述：

神经网络中计算单元（如隐藏单元）的感受野，通常是指感觉场（例如，源节点的输入层）的区域，其中充分的感觉刺激（如模式）将引起响应。

这一定义可以很好地等价应用于多层感知器和 RBF 网络。然而，关于感受野的数学描述在 RBF 网络下比多层感知器下更容易决定。

令 $\varphi(\mathbf{x},\mathbf{x}_i)$ 定义计算单元关于输入向量 $\mathbf{x}$ 的函数依赖，这一单元是以 $\mathbf{x}_i$ 为中心的。根据 Xu 等（1994），计算单元的感受野定义为

$$\psi(\mathbf{x}) = \varphi(\mathbf{x},\mathbf{x}_i) - a \tag{5.50}$$

这里 a 是某个正常数。用文字表达，这一方程说明，函数 $\varphi(\mathbf{x},\mathbf{x}_i)$ 的感受野是输入向量 $\mathbf{x}$ 的范围的特别子集。对于 $\mathbf{x}$，函数 $\varphi(\mathbf{x},\mathbf{x}_i)$ 有充分大的值，所有这些值都等于或者大于预先指定的 a 级别。

例 2　高斯隐藏单元的感受野

考虑由下式定义的高斯计算单元

$$\varphi(\mathbf{x},\mathbf{x}_i)=\exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x}-\mathbf{x}_i\|^2\right)$$

根据式(5.50)，这一单元的感受野是

$$\psi(\mathbf{x})=\exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x}-\mathbf{x}_i\|^2\right)-a$$

这里 $a<1$。$\psi(\mathbf{x})$的最小允许值是零，对此方程产生

$$\|\mathbf{x}-\mathbf{x}_i\|=\sigma\sqrt{2\log\left(\frac{1}{a}\right)}$$

因此高斯函数 $\varphi(\mathbf{x},\mathbf{x}_i)$ 的感受野由多维曲面定义，在以点 $\mathbf{x}_i$ 为中心的周围以一种类似球体的方式对称。感受野的类似球体对称性质是从高斯函数自身自然继承的。

图 5.7 给出了这一曲面的两个具体例子：

1. 一维感受野 $\psi(x)$，对此输入 x 的范围被限制于闭区间 $[(x_i-\sigma\sqrt{2\log(1/a)}),(x_i+\sigma\sqrt{2\log(1/a)})]$中，如图 5.7a 所示。

2. 二维感受野 $\psi(\mathbf{x})$，对此输入 $\mathbf{x}$ 的范围被限制在中心为 $\mathbf{x}_i=[x_{i,1},x_{i,2}]^T$ 的圆盘上，半径是 $\sigma\sqrt{2\log(1/a)}$，如图 5.7b 所示。 ■

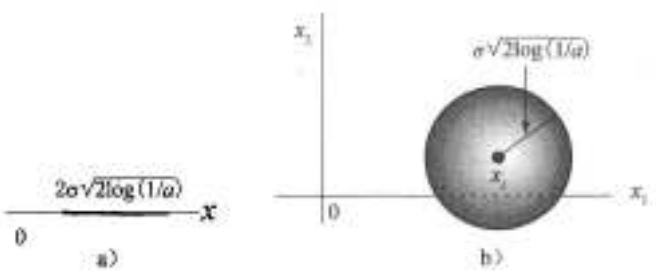


图 5.7　对两个具体情形关于感受野概念的图示：a) 一维；b) 二维

高斯函数作为核的解释

高斯函数 $\varphi(\mathbf{x},\mathbf{x}_i)$的另一个重要方面是它可以被解释为一个核，这是在统计学文献中广泛使用的术语；在机器学习文献中的使用也在渐渐普及。

考虑一个依赖于输入向量 $\mathbf{x}$ 的函数，其中心处在欧几里得空间的原点。记为 $k(\mathbf{x})$的核函数形式的基础是，这一函数具有和随机变量的概率密度函数相似的性质：

性质 1　*核 $k(\mathbf{x})$是关于 $\mathbf{x}$ 连续、有界的实函数，关于原点对称，在原点处获得最大值。*

性质 2　*在核 $k(\mathbf{x})$的曲面下的总体积是 1；即对于一个 m 维的向量 $\mathbf{x}$，我们有*

$$\int_{\mathbb{R}^m}k(\mathbf{x})\mathrm{d}\mathbf{x}=1$$

除了一个尺度因子，对于中心 $\mathbf{x}_i$ 处于原点时高斯函数 $\varphi(\mathbf{x},\mathbf{x}_i)$满足这两个性质。对于非零的 $\mathbf{x}_i$ 性质 1 和 2 仍然保持，因为 $\mathbf{x}_i$ 代替了原点。

正由于高斯函数可以解释为核，因此本章大标题采用了术语“核方法”。

5.10　核回归及其与 RBF 网络的关系

在第 5.3 节中介绍的 RBF 网络理论是建立在插值概念之上的。在本节中，我们采取另一个观点——*核回归*，这是建立在密度估计概念之上的。

具体来说，考虑由下式定义的非线性回归模型：

$$y_i = f(\mathbf{x}_i) + \varepsilon(i), \quad i = 1,2,\cdots,N \tag{5.51}$$

其中 $\varepsilon(i)$是附加的白噪项，其均值为 0 方差为 σ_ε^2。为了避免混淆，使用符号 y_i(代替了前面所用的 d_i) 来标记模型的输出。作为未知回归函数 $f(\mathbf{x})$的合理估计，我们可以取点 $\mathbf{x}$ 附近可观测量（即模型输出 y 的值）的均值。然而，要使这一途径成功，局部平均将被限制在点 $\mathbf{x}$ 周围的小邻域（即感受野）中的观测值里，因为通常对应于远离 $\mathbf{x}$ 的点的观测将具有不同的均值。更精确地，我们发现未知函数 $f(\mathbf{x})$等价于给定回归量 $\mathbf{x}$ 时观测 y 的条件均值，如下所示：

$$f(\mathbf{x}) = \mathbb{E}[y|\mathbf{x}] = \int_{-\infty}^{\infty} y p_{Y|\mathbf{X}}(y|\mathbf{x})\mathrm{d}y \tag{5.52}$$

这里 $p_{Y|\mathbf{X}}(y|\mathbf{x})$是条件概率密度函数（pdf)，该函数是在随机向量 $\mathbf{X}$ 赋值为 x 的条件下，关于随机变量 Y 函数[9]。由概率理论，我们有

$$p_{Y|\mathbf{X}}(y|\mathbf{x}) = \frac{p_{\mathbf{X}|Y}(\mathbf{x}|y)}{p_{\mathbf{X}}(\mathbf{x})} \tag{5.53}$$

这里是 $p_{\mathbf{X}}(\mathbf{x})$是 $\mathbf{X}$ 的 pdf，$p_{x,y}(\mathbf{x},y)$是 $\mathbf{X}$ 和 Y 的联合 pdf。因此，将式(5.53)代入到式(5.52)，获得下面的回归函数公式：

$$f(\mathbf{x}) = \frac{\int_{-\infty}^{\infty} y p_{\mathbf{X},Y}(\mathbf{x},y)\mathrm{d}y}{p_{\mathbf{X}}(\mathbf{x})} \tag{5.54}$$

我们的特别兴趣在于 $p_{\mathbf{X},Y}(\mathbf{x},y)$的联合概率密度函数未知而我们所能用的仅是训练样本 $\{(\mathbf{x}_i,y_i)\}_{i=1}^N$这一情况。为了估计 $p_{\mathbf{X},Y}(\mathbf{x},y)$以及 $p_{\mathbf{X}}(\mathbf{x})$，可以利用熟知的 Parzen-Rosenblatt 密度估计（Rosenblatt，1956，1970；Parzen，1962）这一非参数估计。这一估计形成的基础在于核 $k(\mathbf{x})$的可用性。假设观测 $\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_N$ 是统计独立同分布的（iid)，我们可以定义 $f_{\mathbf{x}}(\mathbf{x})$的 Parzen-Rosenblatt 密度估计如下：

$$\hat{p}_{\mathbf{x}}(\mathbf{x}) = \frac{1}{Nh^{m_0}}\sum_{i=1}^{N} k\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right), \quad \mathbf{x} \in \mathbb{R}^{m_0} \tag{5.55}$$

这里平滑参数 h 是正数，称为带宽，或简单称为宽；h 控制着核的大小。Parzen-Rosenblatt 密度估计的重要性质是其为一致估计[10]（即渐进无偏)，意味着如果 $h=h(N)$被选择为关于 N 的函数使得

$$\lim_{N\to\infty} h(N) = 0$$

则

$$\lim_{N\to\infty} \mathbb{E}[\hat{p}_{\mathbf{x}}(\mathbf{x})] = p_{\mathbf{x}}(\mathbf{x})$$

为了保持后一方程，x 必须是$\hat{p}_{\mathbf{x}}(\mathbf{x})$的连续点。

与式(5.55)相似的方式，可以公式化联合概率密度函数 $p_{\mathbf{X},Y}(\mathbf{x},y)$的 Parzen-Rosenblatt 密度估计：

$$\hat{p}_{\mathbf{X},Y}(\mathbf{x},y) = \frac{1}{Nh^{m_0+1}}\sum_{i=1}^{N} k\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right)k\left(\frac{y-y_i}{h}\right) \quad \text{当 } \mathbf{x} \in \mathbb{R}^{m_0} \text{ 且 } y \in \mathbb{R} \tag{5.56}$$

关于 y 对$\hat{p}_{\mathbf{X},Y}(\mathbf{x},y)$积分，得到式(5.55)的$\hat{p}_{\mathbf{x}}(\mathbf{x})$。

而且，

$$\int_{-\infty}^{\infty} y\,\hat{p}_{\mathbf{X},Y}(\mathbf{x},y)\mathrm{d}y = \frac{1}{Nh^{m_0+1}}\sum_{i=1}^{N} k\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right)\int_{-\infty}^{\infty} yk\left(\frac{y-y_i}{h}\right)\mathrm{d}y$$

通过设置 $\xi=(y-y_i)/h$ 来改变积分变量并运用核 $k(\cdot)$的性质 2，得到如下结果：

$$\int_{-\infty}^{\infty} y\,\hat{p}_{\mathbf{X},Y}(\mathbf{x},y)\mathrm{d}y = \frac{1}{Nh^{m_0}}\sum_{i=1}^{N} y_i k\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right) \tag{5.57}$$

因此，利用式(5.57)和式(5.55)作为式(5.54)相应的分子和分母的量的估计，得到如下在消除共同项之后关于回归函数 $f(\mathbf{x})$的估计：

$$F(\mathbf{x}) = \hat{f}(\mathbf{x}) = \frac{\sum_{i=1}^{N} y_i k\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right)}{\sum_{j=1}^{N} k\left(\frac{\mathbf{x}-\mathbf{x}_j}{h}\right)} \tag{5.58}$$

这里，为了表达清晰，在分母部分，用 j 代替 i 作为和的索引。

有两个观点来考察式(5.58)的逼近函数 $F(\mathbf{x})$：

1. Nadaraya-Watson 回归估计。第一个观点，定义归一权函数

$$W_{N,i}(\mathbf{x}) = \frac{k\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right)}{\sum_{j=1}^{N} k\left(\frac{\mathbf{x}-\mathbf{x}_j}{h}\right)} \qquad i = 1,2,\cdots,N \tag{5.59}$$

有

$$\sum_{i=1}^{N} W_{N,i}(\mathbf{x}) = 1, \quad \text{对于所有的 } \mathbf{x} \tag{5.60}$$

然后可以重写式(5.58)的核回归估计，简化形式为：

$$F(\mathbf{x}) = \sum_{i=1}^{N} W_{N,i}(\mathbf{x}) y_i \tag{5.61}$$

它将 $F(\mathbf{x})$描述为观察值$\{y_i\}_{i=1}^{N}$的加权均值。式(5.61)给出的加权函数 $W_N(\mathbf{x}, i)$形式是由 Nadaraya(1964) 和 Watson(1964) 提出的，所以式(5.61)所示的逼近函数通常称为 Nadaraya-Watson 回归估计器 (NWRE)[11]。

2. 归一化的 RBF 网络。第二个观点，假设核 $k(\mathbf{x})$ 是球对称的，这样我们就可以令

$$k\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right) = k\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|}{h}\right), \quad \text{对于所有的 } i \tag{5.62}$$

这里 $\|\cdot\|$ 表示包含向量的欧几里得范数 (Krzyzak 等，1996)。相应地，定义归一化径向基函数为：

$$\psi_N(\mathbf{x},\mathbf{x}_i) = \frac{k\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|}{h}\right)}{\sum_{j=1}^{N} k\left(\frac{\|\mathbf{x}-\mathbf{x}_j\|}{h}\right)} \qquad i = 1,2,\cdots,N \tag{5.63}$$

其中，对所有的 $\mathbf{x}$ 有

$$\sum_{i=1}^{N} \psi_N(\mathbf{x},\mathbf{x}_i) = 1, \quad \text{对于所有的 } \mathbf{x} \tag{5.64}$$

$\psi_N(\mathbf{x},\mathbf{x}_i)$中的下标 N 表示使用归一化 (normalization)。

对于第二个观点所考虑的回归问题，我们可以看出应用于基函数 $\psi_N(\mathbf{x},\mathbf{x}_i)$的“线性权值” w_i，就是回归模型中对应于 $\mathbf{x}_i$ 的观察值 y_i。因此令

$$y_i = w_i, \quad i = 1,2,\cdots,N$$

重新将式(5.58)所示的逼近函数写成一般形式：

$$F(\mathbf{x}) = \sum_{i=1}^{N} w_i \psi_N(\mathbf{x}, \mathbf{x}_i) \tag{5.65}$$

式(5.65)表示的是一个归一化 RBF 网络的输入-输出映射（Moody and Darken，1989；Xu 等，1994）。注意：

$$0 \leqslant \psi_N(\mathbf{x}, \mathbf{x}_i) \leqslant 1, \quad \text{对所有的 } \mathbf{x} \text{ 和 } \mathbf{x}_i \tag{5.66}$$

因此，$\psi_N(\mathbf{x}, \mathbf{x}_i)$可以解释为以 $\mathbf{x}_i$ 为条件的由输入向量 $\mathbf{x}$ 描述的事件的概率。

式(5.63)的归一化径向基函数 $\psi_N(\mathbf{x}, \mathbf{x}_i)$与一般径向基函数的不同之处在于 $\psi_N(\mathbf{x}, \mathbf{x}_i)$有一个组成归一化因子的分母。归一化因子是关于输入向量 $\mathbf{x}$ 的固有概率密度函数。因此，对所有的 $\mathbf{x}$ 基函数 $\psi_N(\mathbf{x}, \mathbf{x}_i)$的 $i = 1, 2, \cdots, N$ 项之和等于 1，即式(5.64)。

多元高斯分布

一般说来可以选择各种各样的核函数。但是，理论和实际的考虑限制了我们的选择。一个广泛使用的核函数是多元高斯分布

$$k(\mathbf{x}) = \frac{1}{(2\pi)^{m_0/2}} \exp\left(-\frac{\|\mathbf{x}\|^2}{2}\right) \tag{5.67}$$

其中，m_0 是输入向量 $\mathbf{x}$ 的维数。很明显，式(5.67)所示的核 $k(\mathbf{x})$具有球对称性。假设使用相同的带宽 σ，σ 与平滑参数 h 对每一个高斯分布的作用相同，且以数据点 $\mathbf{x}_i$ 作为核函数的中心，可写成

$$k\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right) = \frac{1}{(2\pi\sigma^2)^{m_0/2}} \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{2\sigma^2}\right), i = 1, 2, \cdots, N \tag{5.68}$$

因此，使用式(5.68)，Nadaraya-Watson 回归估计可以写成

$$F(\mathbf{x}) = \frac{\sum_{i=1}^{N} y_i \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{2\sigma^2}\right)}{\sum_{j=1}^{N} \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_j\|^2}{2\sigma^2}\right)} \tag{5.69}$$

其中分母项表示 Parzen-Rosenblatt 密度估计器，由 N 个以数据点 $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N$ 为中心的多元高斯分布之和构成（Specht，1991）。

相应地，将式(5.68)代入式(5.63)和式(5.65)，可以得到归一化 RBF 网络的输入-输出映射函数的如下形式：

$$F(\mathbf{x}) = \frac{\sum_{i=1}^{N} w_i \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{2\sigma^2}\right)}{\sum_{j=1}^{N} \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_j\|^2}{2\sigma^2}\right)} \tag{5.70}$$

在式(5.69)和式(5.70)中，归一化径向基函数的中心与输入数据点 $\{\mathbf{x}_i\}_{i=1}^{N}$ 一致。与一般径向基函数相同，可以使用较小数量的归一化径向基函数，它们的中心被看作可以根据某种启发式方法选择的自由参数（Moody and Darken，1989），或者按第 7 章讨论的原则方式来确定。

5.11　小结和讨论

在本章中，我们集中讨论作为多层感知器替代的径向基函数（RBF）网络。和第 4 章讨论的多层感知器相似，RBF 网络就其本身来说是一个通用逼近器（Sandberg and Xu，1997a，1997b）。它们基本结构上的不同可总结如下：

在多层感知器中，函数逼近是通过加权和的嵌套集来定义的，而 RBF 网络中逼近是由单一加权和定义的。

设计考虑

RBF 网络的设计遵循插值理论，这从数学上讲是精致的。然而，从实际的观点，该设计方法有两个缺点。第一，训练样本可能是含噪声的，通过 RBF 网络可能产生误入歧途的结果。第二，当训练样本的大小很大时，RBF 网络的隐藏层采用和训练样本相同大小的话，将会浪费计算资源。

设计 RBF 网络的更实际的方法是遵循 5.5 节到 5.7 节描述的混合学习过程。基本上，这一过程按如下两阶段操作：

- 第一阶段应用 K-均值聚类算法按非监督方式来训练隐藏层。典型地，聚类个数，也就是隐藏层的计算单元个数，明显小于训练样本的大小。
- 第二阶段应用递归最小二乘法来计算线性输出层的权值向量。

这两阶段设计过程具有两个期望的特征：计算简单性和加速收敛性。

试验结果

5.8 节在双月“玩具”问题上的计算机试验结果揭示了混合“K-均值，RLS”分类器能够达到令人印象深刻的性能。将试验的结果和下一章将要讨论的支持向量机（SVM）的同样试验做比较的时候，我们发现这两个分类器执行得非常相似。然而，“K-均值，RLS”分类器比 SVM 收敛速度更快，需要更少的计算。

值得注意的是 Rifkin（2002）在他的博士论文中，很细致地比较了 RLS 和 SVM 对于线性可分模式的分类，使用了玩具样本的选集。下面是他实验结果的小结：

- RLS 和 SVM 分类器表现出近乎相同的性能。
- 它们都对训练样本中异常的出现敏感。

Rifkin(2002) 也完成了图像分类的试验，使用了两个不同的数据集合：

- U. S. 邮政服务（USPS）手写字数据集，包括 7291 个训练样本和 2007 个测试样本。训练集包含 6639 个负样本和 652 个正样本，而测试集包含 1807 个负样本和 200 个正样本。
- MIT 识别集，被称为 faces。训练集包含 2429 个人脸和 4548 个非人脸，测试集包含 572 个人脸和 23 573 个非人脸。

对于 USPS 数据集，报告指出非线性 RLS 分类器和 SVM 相比在接收机工作特性（ROC）曲线的全部范围上完成得一样好或者更好。ROC 曲线画出了当使用单一网络输出时在变化的决策阈值上真实位置率（true-positive rate）和错误位置率（false-positive rate）的图；术语“率”是衡量分类概率的另一个途径。在 faces 上完成的测试产生了混合结果：对于一个设计参数集合，SVM 本质上比非线性 RLS 分类器完成得好。对于另一个设计参数集合，性能相近。我们要指出的是 Rifkin(2002) 中设计非线性 RLS 分类器的隐藏层的策略和本章中考虑的 K-均值聚类算法有很大不同。

一个重要的信息是，对于本书中的双月“玩具”实验以及 Rifkin(2002) 报告的更多方面的试验来说，包含两个方面：

1. RLS 算法在信号处理和控制理论的文献中进行了彻底研究（Haykin，2002；Goodwin and Sin，1984）。遗憾的是，在机器学习的文献中，除了 Rifkin(2002) 的博士论文以及少量其他文献之外基本上被完全忽视了。

2. 需要利用实践数据集来进行更广泛的实验，以便对于基于 RLS 算法（用于设计其输出层）的 RBF 网络和 SVM 之间的相互比较作出更精确的结论，不仅仅从性能的角度，也从收敛速率和计算复杂度的角度。

核回归

本章研究的另一个重要的课题是核回归，这建立在密度估计的概念之上。特别地，我们集中讨论了被熟知为 Parzen-Rosenblatt 密度估计器的非参数估计器，其形成依赖于核的可利用性。这一研究让我们通过两种观点来考察定义为非线性回归模型的逼近函数：Nadaraya-Watson 回归估计器和归一化 RBF 网络。对于这两者，多变量高斯分布提供了对于核的一种好的选择。

注释和参考文献

1. 径向基函数是在解决实多变量插值问题时首次提出的。这方面的早期工作在 Powell(1985) 中有所论述。现在径向基函数是数值分析研究中的一个主要方向。

   Broomhead and Lowe(1988) 首先将径向基函数应用于神经网络设计。Poggio and Girosi(1990a) 在径向基函数网络的理论与设计中也做出了重大贡献。后一篇论文强调将正则化理论应用于这类神经网络，以提高对新数据的泛化能力；正则化理论将在第 10 章详细讨论。
2. Cover 定理的证明遵循如下两个基本考虑 (Cover，1965)：
   - Schlafli 定理，或函数计数定理：对 m_1 维欧几里得空间上的 N 个处于一般位置的向量进行二分，可得到的齐次线性可分的二分方式的数目等于：
     $$C(N,m_1)=2\sum_{m=0}^{m_1-1}\binom{N-1}{m}$$
     如果每一个含有 m_1 个或小于 m_1 个的向量子集都是线性独立的，就说 m_1 维欧几里得空间上的集合 $\mathcal{H}=\{\mathbf{x}_i\}_{i=1}^{N}$ 处于一般位置。
   - $\mathcal{H}$的联合概率分布的反射不变性：一个随机二分是可分的概率（在$\mathcal{H}$的条件下）等于$\mathcal{H}$的一个特定二分（所有的 N 个向量都属于一类）的非条件概率。

   Cameron(1960)、Joseph(1960) 和 Winder(1961) 以不同的形式独立证明了函数计数定理，并应用于特定的感知器配置（即线性阈值单元）。在 Cover(1968) 中这个定理还被用于根据所有可调参数的总数估计感知器网络的能力，它的下界是 $N/(1+\log_2/N)$，其中 N 是输入模式的数量。
3. 通常聚类在多本书中都有讨论，包括 Theodoridis and Koutroumbas(2003)、Duda 等 (2001) 和 Fukunaga (1990)。

   K-均值算法是从 MacQueen(1967) 之后采取了这个名字，他在统计聚类过程中研究了 K-均值算法，包括算法的收敛性。这一思想在 Foregey(1962) 中在讨论聚类时进行了描述。

   Ding and He(2004) 介绍了在聚类的 K-均值算法和数据削减的主分量分析之间一个非常有趣的关系。特别证明了主分量表达了 K-均值聚类中聚类成员指标的连续的（随意的）解。在某种方式下，这两种观点是一致的，即数据的聚类也是某种形式的数据削减，当然这两者都是在非监督方式下完成的。主分量分析将在第 8 章介绍。

   在通信文献中处理矢量量化时，称 K-均值算法为广义 Lloyd 算法，这是 Bell 试验室 1957 年一篇未发表的报告中出现的 Lloyd 原始算法的广义化版本。后来，在 1982 年，Lloyd 的报告被正式出版。
4. Fisher 线性判别　式(5.26)定义的代价函数不是别的正是称为类内协方差（分散）矩阵的迹 (Theodoridis and Koutroumbas，2003)。

   要理解这句话的意思，考虑由如下的内积定义的变量 y：
   $$y=\mathbf{w}^T\mathbf{x} \tag{A}$$
   向量 $\mathbf{x}$ 是从两个族群$\mathscr{C}_1$ 和$\mathscr{C}_2$ 中的其一取出的，这两个族群通过均值 $\boldsymbol{\mu}_1$ 和 $\boldsymbol{\mu}_2$ 将彼此区别开来，$\mathbf{w}$ 是可调整参数。这两类之间 Fisher 判别准则由下式定义：
   $$J(\mathbf{w})=\frac{\mathbf{w}^T\mathbf{C}_b\mathbf{w}}{\mathbf{w}^T\mathbf{C}_t\mathbf{w}} \tag{B}$$
   其中 $\mathbf{C}_b$ 是类间协方差矩阵，定义为
   $$\mathbf{C}_b=(\boldsymbol{\mu}_2-\boldsymbol{\mu}_1)(\boldsymbol{\mu}_2-\boldsymbol{\mu}_1)^T \tag{C}$$
   $\mathbf{C}_t$ 是总类内协方差矩阵，定义为

$$\mathbf{C}_t = \sum_{n\in\mathcal{C}_1}(\mathbf{x}_n - \boldsymbol{\mu}_1)(\mathbf{x}_n - \boldsymbol{\mu}_1)^T + \sum_{n\in\mathcal{C}_2}(\mathbf{x}_n - \boldsymbol{\mu}_2)(\mathbf{x}_n - \boldsymbol{\mu}_2)^T \tag{D}$$

类内协方差矩阵 $\mathbf{C}_t$ 是正比于训练样本的样本协方差矩阵的。它是对称非负定的且如果训练样本的大小足够大它通常是非奇异的。类间协方差矩阵 $\mathbf{C}_b$ 也是对称非负定的但奇异。一个特别感兴趣的性质是矩阵乘积 $\mathbf{C}_b\mathbf{w}$ 总是差分均值向量 $\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2$ 的方向。这一性质直接由 $\mathbf{C}_b$ 的定义得到。

定义 $J(\mathbf{w})$的表达式被熟知为广义 Rayleigh 商。最大化 $J(\mathbf{w})$的 $\mathbf{w}$ 必须满足 i 条件

$$\mathbf{C}_b\mathbf{w} = \lambda\mathbf{C}_t\mathbf{w} \tag{E}$$

其中 λ 是尺度因子。方程（E）是一个广义特征值问题。认识到，在我们的情形下，矩阵乘积 $\mathbf{C}_b\mathbf{w}$ 总是差分均值向量 $\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2$ 的方向，我们发现式（E）的解为

$$\mathbf{w} = \mathbf{C}_t^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) \tag{F}$$

这称为 Fisher 线性判别（Duda 等，2001）。

考虑到式（D）中类内协方差矩阵 $\mathbf{C}_t$ 的迹，我们确实发现式(5.26)的代价函数是这一协方差矩阵的迹，正如我们已经说明的那样。

5. 从哲学的角度讲，K-均值算法的两步最优化过程相似于 EM 算法的两步最优化，这里第一步是期望的某一种，记为“E”，第二步是最大化的某一种，记为“M”。EM 算法是从最大似然计算的基础上发展起来的；将在第 11 章中讲述。
6. 在文献中，缩写字母“RLS”被用于简称第 2 章中讨论过的正则最小二乘法和本章中讨论过的递归最小二乘法。在相关讨论中，我们通常能够分辨这个缩写字指的两个算法中的哪一个。
7. 对于在 5.6 节中总结的 RLS 算法的基本内容的经典内容，在 Diniz(2002) 和 Haykin(2002) 的书中有讲述。
8. RBF 网络的混合学习过程已经在多个不同的文献中讲述过，对于这两个阶段利用不同的算法；参看 Moody and Darken(1989) 和 Lippman(1989b)。
9. 式(5.52)的条件均值估计器也是最小均方估计器；这一说法的证明在第 14 章的注释 7 中在贝叶斯估计理论下给出。
10. 关于 Parzen-Rosenblatt 密度估计器的渐近无偏性的证明，参看 Parzen(1962) 和 Cacoullos(1966)。
11. Nadaraya-Watson 回归估计器在统计学文献中已是一个广泛研究的主题。从更广的意义上说，非参数泛函估计在统计学中占有中心地位；参看 Hardle(1990) 和 Roussas(1991) 的论文集。

习题

Cover 定理

5.1 如 5.2 节所建议的，学习式(5.5)的最好方式是通过设 $N=\lambda m_1$ 来归一化。利用这一归一化，对 $N=1$，5，15 和 25，绘出 $P(\lambda m_1, m_1)$对于 λ 的图形，从而验证这一节中讲述的式(5.5)的两个特性。

5.2 确认在 5.2 节开始时指出的 Cover 定理的优缺点。

5.3 在图 5.1b 中给出的例子画出了一个球形可分的二分。假设分离曲面之外的四个点位于一个圆上，而在分离曲面内部仅有的一个数据点位于分离曲面的中心。调查这些数据点样本是如何非线性变化的，使用

(a) 多二次函数

$$\varphi(x) = (x^2 + 1)^{1/2}$$

(b) 逆多二次函数

$$\varphi(x) = \frac{1}{(x^2 + 1)^{1/2}}$$

K-均值聚类

5.4 考虑下面对于定义在式(5.26)的代价函数的修正：

$$J(\boldsymbol{\mu}_j) = \sum_{j=1}^{K}\sum_{i=1}^{N} w_{ij} \| \mathbf{x}_i - \boldsymbol{\mu}_j \|^2$$

在这一函数中，权因子 w_{ij} 定义如下：

$$w_{ij} = \begin{cases} 1 & \text{如果数据 } \mathbf{x}_i \text{ 位于聚类 } j \\ 0 & \text{否则} \end{cases}$$

证明代价函数的最小解是：

$$\hat{\boldsymbol{\mu}}_j = \frac{\sum_{i=1}^{N} w_{ij}\mathbf{x}_i}{\sum_{i=1}^{N} w_{ij}}, \qquad j = 1,2,\cdots,K$$

如何解释这个公式中分子和分母的表达式呢？对比两个答案的结论和我们在聚类的部分已经学习过的结论。

递归最小二乘算法

5.5 在这一习题中，我们采用矩阵 **P** 的统计解释，**P** 定义为相关矩阵 **R** 的逆矩阵。

(a) 利用线性回归模型

$$d_i = \mathbf{w}^T \boldsymbol{\phi}_i + \varepsilon_i, \qquad i = 1,2\cdots,N$$

证明 **w** 的最小二乘最优估计可表示为

$$\hat{\mathbf{w}} = \mathbf{w} + (\boldsymbol{\Phi}^T\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^T\boldsymbol{\varepsilon}$$

其中

$$\boldsymbol{\Phi} = \begin{bmatrix} \boldsymbol{\phi}_1^T \\ \boldsymbol{\phi}_2^T \\ \vdots \\ \boldsymbol{\phi}_N^T \end{bmatrix}$$

和

$$\boldsymbol{\varepsilon} = [\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_N]^T$$

假设误差 ε_i 是方差为 σ^2 的白噪过程的样本。

(b) 从而，证明协方差矩阵

$$\mathbb{E}[(\mathbf{w} - \hat{\mathbf{w}})(\mathbf{w} - \hat{\mathbf{w}})^T] = \sigma^2\mathbf{R}^{-1} = \sigma^2\mathbf{P}$$

其中

$$\mathbf{R} = \sum_{i=1}^{N} \boldsymbol{\phi}_i \boldsymbol{\phi}_i^T$$

5.6 从如下的正则代价函数开始：

$$\mathscr{E}_{av}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}(d(i) - \mathbf{w}^T\boldsymbol{\phi}(i))^2 + \frac{1}{2}\lambda \| \mathbf{w} \|^2$$

进行如下工作：

(a) 证明正则项 $\frac{1}{2}\lambda \| \mathbf{w} \|^2$ 的附加无论如何对于 RLS 算法的构成没有影响，如 5.6 节所总结的那样。

(b) 引入正则项的仅有的效果是将输入数据的相关矩阵表达式修改为如下形式：

$$\mathbf{R}(n) = \sum_{i=1}^{n} \boldsymbol{\phi}(i)\boldsymbol{\phi}^T(i) + \lambda\mathbf{I}$$

其中 **I** 是单位矩阵。证明这一相关矩阵 $\mathbf{R}(n)$ 的新的表达式，并证明通过引入正则化所获得的实际效益。

5.7 自适应滤波的最小均方（LMS）算法已经在第 3 章讨论了。比较递归最小二乘（RLS）算法和 LMS 算法的优缺点。

RBF 网络的监督训练

5.8 基于高斯的 RBF 网络的输入-输出关系定义如下：

$$y(i) = \sum_{j=1}^{K} w_j(n)\exp\left(-\frac{1}{2\sigma^2(n)} \| \mathbf{x}(i) - \boldsymbol{\mu}_j(n) \|^2\right), \qquad i = 1,2,\cdots,n$$

其中 $\boldsymbol{\mu}_j(n)$ 是第 j 个高斯单元的中心点，宽 $\sigma(n)$ 对所有的 K 个单元是共同的，$w_j(n)$ 是分配给第 j 个输出单元的线性权值；所有这些参数都在时间 n 处测量。用于训练网络的代价函数定义为：

$$\mathscr{E} = \frac{1}{2}\sum_{i=1}^{n} e^2(i)$$

其中

$$e(i) = d(i) - y(i)$$

代价函数 $\mathscr{E}$ 是输出层中线性权值的凸函数，但对于高斯单元的中心和宽非凸。

(a) 计算代价函数对每一个网络参数 $w_j(n)$，$\boldsymbol{\mu}_j(n)$ 和 $\sigma(n)$ 对所有 i 的偏导数。

(b) 利用 (a) 所获得的梯度来对所有的网络参数表达更新公式，对网络的可调整参数相应地假设为 η_w，η_μ，η_σ。

(c) 梯度向量 $\partial \mathscr{C}/\mu_j(n)$对于输入数据有一个类似于聚类的效果，证明这一说法。

核估计

5.9 假设给你“无噪声”的训练样本$\{f(x_i)\}_{i=1}^N$，要求设计一个神经网络对被噪声破坏的数据样本（因而不包含在训练集中）泛化。令 $F(\mathbf{x})$标记为被这样的神经网络实现的逼近函数，这一函数被选择为使得期望平方误差

$$J(F)=\frac{1}{2}\sum_{i=1}^{N}\int_{\mathbb{R}^{m_0}}[f(\mathbf{x}_i)-F(\mathbf{x}_i,\boldsymbol{\xi})]^2 f_{\xi}(\boldsymbol{\xi})d\boldsymbol{\xi}$$

最小，其中 $f_{\xi}(\xi)$是输入空间$\mathbb{R}^{m_0}$中噪声分布的概率密度函数。证明这个最小二乘问题的解由下式给出(Webb, 1994)：

$$F(\mathbf{x})=\frac{\sum_{i=1}^{N}f(\mathbf{x}_i)f_{\xi}(\mathbf{x}-\mathbf{x}_i)}{\sum_{i=1}^{N}f_{\xi}(\mathbf{x}-\mathbf{x}_i)}$$

比较这个估计器和 Nadaraya-Watson 估计器。

计算机实验

5.10 这个计算机实验的目的是调查由 K-均值算法完成的聚类过程。为了对该实验提供深入洞察，我们将聚类个数固定为 $K=6$，但变化图 1.8 所示的两月之间的垂直分隔距离。具体来说，要求利用从图 1.8 的双月中两个区域随机取得的 1 000 个数据点作为无标志训练样本进行如下工作：

(a) 对于 8 个一致隔开的垂直分隔序列通过实验决定均值$\hat{\boldsymbol{\mu}}_j$ 和方差 $\hat{\sigma}_j^2, j=1,2,\cdots,6$，从 $d=1$ 开始每次减少 1 直到达到最后的分隔距离 $d=-6$。

(b) 根据 (a) 所获得的结果，对于聚类 j 的均值$\hat{\boldsymbol{\mu}}_j$ 是如何被减小的分隔距离 d 所影响的作一个评论，$j=1,2,3$。

(c) 对 $j=1,2,\cdots,6$ 画出方差$\hat{\sigma}^2$ 和分隔距离 d 的关系图。

(d) 将实验公式(5.49)计算出的 σ^2 和 (c) 中的图显示的趋势做比较。

5.11 第二个实验的目的是比较两个混合学习算法的分类性能：在第 5.8 节调查过的“K-均值，RLS”算法和这一习题中调查的“K-均值，LMS”算法。

如第 5.8 节所述，假设如下的规格：

隐藏高斯单元数：20

训练样本数：1 000 个数据点

测试样本数：2 000 个数据点

令 LMS 算法的学习率参数从 0.6 线性地退火到 0.01。

(a) 对于图 1.8 的两个月之间的垂直分隔设为 $d=-5$ 时，用“K-均值，LMS”算法构造决策边界。

(b) 当 $d=-6$ 时重复这一实验。

(c) 比较用“K-均值，LMS”算法和 5.8 节学习过的“K-均值，RLS”算法获得的分类结果。

(d) 比较一般性的“K-均值，LMS”算法和“K-均值，RLS”算法之间的复杂性。

第6章
Neural Networks and Learning Machines, 3E

支持向量机

本章组织

本章学习支持向量机，此算法也许是所有使用核学习方法中最好的机器学习算法。首先是6.1节的引言，接下来的内容组织如下：

6.2节主要讨论在模式线性可分的情况下如何构造一个优化的超平面，在6.3节，考虑更加复杂情况下的模式分类，即线性不可分的情况下如何构造一个优化的超平面。

6.4节中将引入内积核的思想，由此建立将支持向量机作为一种核方法的学习算法框架，同时，我们还引入广泛使用的思想——核技巧。6.5节总结支持向量机设计的主要思想，6.6节重新考虑XOR问题。6.7节将对一个具体的模式分类问题进行计算机实验。

6.8节中引入ε-不敏感损失函数，从而用于解决6.9节出现的回归问题。

6.10节主要介绍表达定理，它使人洞悉在Mercer核的环境下近似函数的生成。

最后，6.11节对本章进行总结和讨论。

6.1 引言

在第4章，我们学习了由反向传播算法训练的多层感知器，该算法好的特点是其简单性，但是算法收敛速度慢且缺少最优化性。在第5章，我们研究了另一类前馈网络，即径向基函数网络，其主要思想来自于插值理论，然后描述了次最优的两阶段设计过程。在这一章，我们将讨论另一种通用的前馈网络的类型，称为支持向量机（support vector machines，SVMs）[1]。

从本质上来说，支持向量机是具有很多优秀性能的两类机器学习方法。要解释它是如何工作的，从模式分类中可分离模式的情况开始可能是最容易的。在此背景下，支持向量机的主要思想可以总结如下：

给定训练样本，支持向量机建立一个超平面作为决策曲面，使得正例和反例之间的隔离边缘被最大化。

在处理更加复杂的线性不可分的模式时，我们原则性地对算法的基本思想进行扩展。

在支持向量$\mathbf{x}_i$和从输入空间提取的向量$\mathbf{x}$之间的内积核这一概念是构造支持向量机学习算法的关键。最重要的是，支持向量是由算法从训练数据中抽取的小的子集构成。事实上，支持向量机被称为核方法是由于其构造过程中这一关键的性质。但是不同于第5章中描述的次优化核方法，对于支持向量机的设计来说核方法本质上是最优的，而最优性是根植于凸最优。但是支持向量机这些令人满意的特点是通过增加计算复杂度得到的。

与第4章和第5章讨论的过程一样，支持向量机可以用来解模式识别和非线性回归问题，但是对于解复杂的模式分类问题而言支持向量机具有尤为重要的影响。

6.2 线性可分模式的最优超平面

考虑训练样本$\{(x_i,d_i)\}_{i=1}^N$，其中$\mathbf{x}_i$是输入模式的第i个样例，d_i是对应的期望响应（目标输出）。首先假设由子集$d_i=+1$代表的模式（类）和$d_i=-1$代表的模式是“线性可分的”。用于分离的超平面形式的决策曲面方程是：

$$\mathbf{w}^T\mathbf{x}+b=0 \tag{6.1}$$

其中$\mathbf{x}$是输入向量，$\mathbf{w}$是可调的权值向量，b是偏置。因此可以写成：

$$\begin{aligned}\mathbf{w}^T\mathbf{x}_i+b\geqslant 0 \quad & \text{当 } d_i=+1\\ \mathbf{w}^T\mathbf{x}_i+b<0 \quad & \text{当 } d_i=-1\end{aligned} \tag{6.2}$$

在这里做了模式线性可分的假设，以便在相当简单的环境里解释支持向量机背后的基本思想；在第 6.3 节将放宽这个假设。

对于一个给定的权值向量 $\mathbf{w}$ 和偏置 b，由式(6.1)定义的超平面和最近的数据点之间的间隔被称为分离边缘，用 ρ 表示。支持向量机的目标是找到一个特殊的超平面，这个超平面的分离边缘 ρ 最大。在这种条件下，决策曲面称为最优超平面（optimal hyperplane）。图 6.1 描述二维输入空间中最优超平面的几何结构。

设 $\mathbf{w}_o$ 和 b_o 分别表示权值向量和偏置的最优值。相应地，在输入空间里表示多维线性决策面的最优超平面形式如下：

$$\mathbf{w}_o^T\mathbf{x}+b_o=0 \tag{6.3}$$

它是式(6.1)的改写。判别函数

$$g(\mathbf{x})=\mathbf{w}_o^T\mathbf{x}+b_o \tag{6.4}$$

给出从 $\mathbf{x}$ 到最优超平面的距离的一种代数度量（Duda and Hart，1973）。看出这一点的最简单方法或许是将 $\mathbf{x}$ 表达为

$$\mathbf{x}=\mathbf{x}_p+r\frac{\mathbf{w}_o}{\|\mathbf{w}_o\|}$$

其中，$\mathbf{x}_p$ 是 $\mathbf{x}$ 在最优超平面上的正轴投影，r 是期望的代数距离；如果 $\mathbf{x}$ 在最优超平面的正面，r 是正值；相反如果 $\mathbf{x}$ 在最优超平面的负面，r 是负值。因为由定义知 $g(\mathbf{x}_p)=0$，由此推出：

$$g(\mathbf{x})=\mathbf{w}_o^T\mathbf{x}+b_o=r\|\mathbf{w}_o\|$$

或者等价于：

$$r=\frac{g(\mathbf{x})}{\|\mathbf{w}_o\|} \tag{6.5}$$

尤其，从原点（即 $\mathbf{x}=0$）到最优超平面的距离由 $b_o/\|\mathbf{w}_o\|$ 给定。如果 $b_o>0$，原点在最优超平面的正面；如果 $b_o<0$，原点在负面；如果 $b_o=0$，最优超平面通过原点。这些代数的几何解释在图 6.2 中给出。

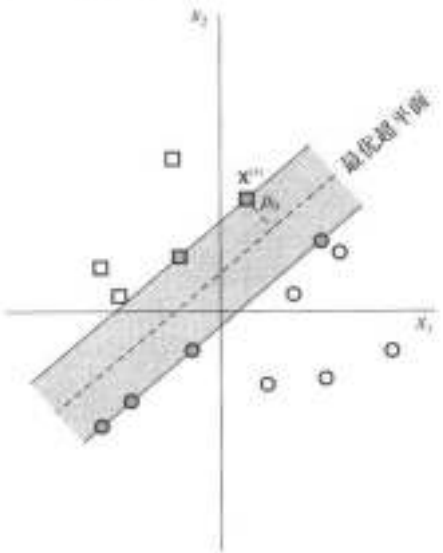


图 6.1　线性可分模式最优超平面的思想示意图；灰色阴影表示的点是支持向量

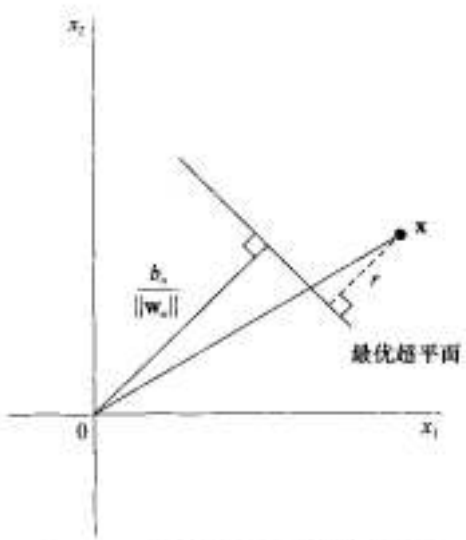


图 6.2　二维情况下点到最优超平面的代数距离的几何解释

现在的问题是对于给定的数据集 $\mathcal{T}=\{(x_i,d_i)\}$，找到最优超平面的参数 $\mathbf{w}_o$ 和 b_o。根据图 6.2 描绘的结果，可以看出一对 $(\mathbf{w}_o,b_o)$ 一定满足条件：

$$\begin{aligned}\mathbf{w}_o^T\mathbf{x}_i+b_o&\geqslant 1 \quad \text{当 } d_i=+1\\ \mathbf{w}_o^T\mathbf{x}_i+b_o&\leqslant -1 \quad \text{当 } d_i=-1\end{aligned} \tag{6.6}$$

注意如果式(6.2)成立，即模式是线性可分的，总可以重新调整 $\mathbf{w}_o$ 和 b_o 的值使得式(6.6)成立；这种重新调整并不改变式(6.3)。

满足式(6.6)第一行或第二行等号情况的特殊数据点 $(\mathbf{x}_i,d_i)$ 称为**支持向量**，“支持向量机”因此得名。其他的训练样本点完全不重要。由于支持向量的特点，这些向量在这类机器学习的运行中起着主导作用。用概念性的术语来说，支持向量是最靠近决策面的数据点，这样的数据点是最难分类的。因此，它们和决策面的最优位置直接相关。

考虑一个支持向量 $\mathbf{x}^{(s)}$ 对应于 $d^{(s)}=+1$。然后根据定义，得出：

$$g(\mathbf{x}^{(s)})=\mathbf{w}_o^T\mathbf{x}^{(s)}+b_o=\mp 1, \quad \text{当 } d^{(s)}=\mp 1 \tag{6.7}$$

从式(6.5)知从支持向量 $\mathbf{x}^{(s)}$ 到最优超平面的代数距离是

$$r=\frac{g(\mathbf{x}^{(s)})}{\|\mathbf{w}_o\|}=\begin{cases}\dfrac{1}{\|\mathbf{w}_o\|}, & \text{当 } d^{(s)}=+1\\ -\dfrac{1}{\|\mathbf{w}_o\|}, & \text{当 } d^{(s)}=-1\end{cases} \tag{6.8}$$

其中加号表示 $\mathbf{x}^{(s)}$ 在最优超平面的正面，而减号表示 $\mathbf{x}^{(s)}$ 在最优超平面的负面。让 ρ 表示在两个类之间的分离边缘的最优值，其中这两个类构成训练集合。因此从式(6.8)得到

$$\rho=2r=\frac{2}{\|\mathbf{w}_o\|} \tag{6.9}$$

式(6.9)说明：

最大化两个类之间的分离边缘等价于最小化权值向量 w 的欧几里得范数。

总之，由式(6.3)定义的最优超平面是唯一的，意味着最优权值向量 $\mathbf{w}_o$ 提供正反例之间最大可能的分离。这个优化条件是通过最小化权值向量 $\mathbf{w}$ 的欧几里得范数获得的。

寻找最优超平面的二次最优化

支持向量机灵活地根植于*凸优化理论*[2]—因此机器具有良好的最优化性。基本上分以下四个步骤来进行：

1. 寻找最优超平面的问题，以这样一个陈述为开始：即在原始权重空间的带约束的优化问题。
2. 对于上述约束问题建立拉格朗日函数。
3. 推导出机器最优化条件。
4. 问题的最后阶段是在对偶空间解决带拉格朗日乘子的优化问题。

要继续讲解，我们首先注意到训练样本

$$\mathcal{T}=\{\mathbf{x}_i,d_i\}_{i=1}^N$$

是嵌入式(6.6)的两行的。把两个等式合并到一个等式是有益的：

$$d_i(\mathbf{w}^T\mathbf{x}_i+b)\geqslant 1 \quad \text{当 } i=1,2,\cdots,N \tag{6.10}$$

手握这样的约束形式，现在我们准备将约束最优问题正式地陈述如下：

给定训练样本 $\{(x_i,d_i)\}_{i=1}^N$，找到权值向量 $\mathbf{w}$ 和偏置 b 的最优值使得它们满足下面的约束条件：

$$d_i(\mathbf{w}^T\mathbf{x}_i+b)\geqslant 1 \quad \text{当 } i=1,2,\cdots,N$$

并且权值向量 $\mathbf{w}$ 最小化代价函数

$$\Phi(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

这里包含比例因子 1/2 是为了讲解方便。这个约束优化问题称为原问题（primal problem）。它的基本特点如下：

- 代价函数 $\Phi(\mathbf{w})$ 是 $\mathbf{w}$ 的凸函数。
- 约束条件关于 $\mathbf{w}$ 是线性的。

相应地，可以使用拉格朗日乘子方法解决约束最优问题（Bertsekas，1995）。

首先，建立拉格朗日函数

$$J(\mathbf{w},b,\alpha) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N}\alpha_i[d_i(\mathbf{w}^T\mathbf{x}_i + b) - 1] \tag{6.11}$$

其中辅助非负变量 α_i 称作拉格朗日乘子。约束最优问题的解由拉格朗日函数 $J(\mathbf{w},b,\alpha)$ 的鞍点决定。拉格朗日函数的鞍点具有实根但是符号相反；这样的奇异点一定是不稳定的。鞍点关于 $\mathbf{w}$ 和 b 必定最小化；同时关于 α 必定最大化。$J(\mathbf{w},b,\alpha)$ 对 $\mathbf{w}$ 和 b 求微分并设置为 0，我们得到下面两个最优化条件：

$$\text{条件 1：}\frac{\partial J(\mathbf{w},b,\alpha)}{\partial \mathbf{w}} = \mathbf{0}$$

$$\text{条件 2：}\frac{\partial J(\mathbf{w},b,\alpha)}{\partial b} = 0$$

应用最优化条件 1 到式(6.11)的拉格朗日函数，得到（在重新安排项之后）

$$\mathbf{w} = \sum_{i=1}^{N}\alpha_i d_i \mathbf{x}_i \tag{6.12}$$

应用最优条件 2 到式(6.11)的拉格朗日函数，得到

$$\sum_{i=1}^{N}\alpha_i d_i = 0 \tag{6.13}$$

解向量 $\mathbf{w}$ 定义为 N 个训练样本的展开。但是注意，尽管拉格朗日函数的凸性的解是唯一的，但并不能认为拉格朗日系数 α_i 也是唯一的。

同样需要十分注意的是，所有以不等式满足约束条件的式子，相应的乘子 α_i 必须为 0。换句话说，只有确切满足

$$\alpha_i[d_i(\mathbf{w}^T\mathbf{x}_i + b) - 1] = 0 \tag{6.14}$$

的乘子可以假定非零。这个性质是 Karush-Kuhn-Tucker[3] 条件的陈述（Fletcher，1987；Bertsekas，1995）。

就像前面提到的，原问题是处理凸代价函数和线性约束的。给定这样一个约束最优化问题，可能构造另一个问题，称为对偶问题（dual problem）。第二个问题与原问题有同样的最优值，但是由拉格朗日乘子提供最优解。特别地，可以陈述对偶定理如下（Bertsekas，1995）：

(a) 如果原问题有最优解，对偶问题也有最优解，并且相应的最优值是相同的。

(b) 为了使得 $\mathbf{w}_o$ 为原问题的一个最优解和 α_0 为对偶问题的一个最优解的充分必要条件是 $\mathbf{w}_o$ 对原问题是可行的，并且

$$\Phi(\mathbf{w}_o) = J(\mathbf{w}_o,b_o,\alpha_0) = \min_{\mathbf{w}} J(\mathbf{w},b,\alpha)$$

为了说明对偶问题是原问题的前提，首先逐项展开式(6.11)如下：

$$J(\mathbf{w},b,\alpha) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N}\alpha_i d_i \mathbf{w}^T\mathbf{x}_i - b\sum_{i=1}^{N}\alpha_i d_i + \sum_{i=1}^{N}\alpha_i \tag{6.15}$$

根据式(6.13)最优条件的性质，式(6.15)右端第三项为零。而且从式(6.12)有

$$\mathbf{w}^T\mathbf{w} = \sum_{i=1}^{N}\alpha_i d_i \mathbf{w}^T\mathbf{x}_i = \sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j d_i d_j \mathbf{x}_i^T\mathbf{x}_j$$

相应地，设置目标函数 $J(\mathbf{w},b,\alpha) = Q(\alpha)$，可以改写式(6.15)为

$$Q(\alpha) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j d_i d_j \mathbf{x}_i^T\mathbf{x}_j \tag{6.16}$$

其中 α_i 是非负的。注意，从 $J(\mathbf{w},b,\alpha)$ 转向 $Q(\alpha)$，其中反映出将原问题转化为对偶问题。

现在可以陈述对偶问题如下：

给定训练样本 $\mathcal{T} = \{(x_i,d_i)\}_{i=1}^N$，寻找最大化如下目标函数的拉格朗日乘子 $\{\alpha_i\}_{i=1}^N$：

$$Q(\alpha) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j d_i d_j \mathbf{x}_i^T\mathbf{x}_j$$

满足约束条件

$$(1)\ \sum_{i=1}^{N}\alpha_i d_i = 0$$

$$(2)\ \alpha_i \geqslant 0 \qquad \text{当 } i = 1,2,\cdots,N \text{ 时}$$

不同于式(6.11)中基于拉格朗日函数的原问题，式(6.16)中定义的对偶问题完全是根据训练数据来表达的。而且，函数 $Q(\alpha)$的最大化仅依赖于输入模式点积的集合：

$$\{\mathbf{x}_i^T\mathbf{x}_j\}_{i,j=1}^N$$

一般地，支持向量是训练样本的子集，这意味着解是稀疏的[4]。也就是说对于所有的支持向量，对偶问题的约束（2）以不等式的形式满足，它们的 α 非零；而对于训练样本中的其他点，约束条件以等式条件满足，它们的 α 为零。相应地，确定用 $\alpha_{o,i}$表示的最优拉格朗日乘子后，可以用式(6.12)计算最优权值向量 $\mathbf{w}_o$，并写成

$$\mathbf{w}_o = \sum_{i=1}^{N_s}\alpha_{o,i} d_i \mathbf{x}_i \tag{6.17}$$

其中 N_s 是支持向量的个数（即拉格朗日乘子 $\alpha_{o,i}$非零的个数），要计算偏置 b_o，可以使用获得的 $\mathbf{w}_o$，并对一个正的支持向量应用式(6.7)，这样有

$$\begin{aligned} b_o &= 1 - \mathbf{w}_o^T\mathbf{x}^{(s)} \qquad \text{当 } d^{(s)} = 1 \text{ 时} \\ &= 1 - \sum_{i=1}^{N_s}\alpha_{o,i} d_i \mathbf{x}_i^T\mathbf{x}^{(s)} \end{aligned} \tag{6.18}$$

回忆所有的支持向量 $\mathbf{x}^{(s)}$，相应于训练样本中拉格朗日乘子 $\alpha_{o,i}$不为零的点 $(\mathbf{x}_i,d_i)$。从数值（实际）角度来看，对于支持向量使用平均式(6.18)也许更好，即对所有非零的拉格朗日乘子平均。

最优超平面的统计特性

在支持向量机中，通过约束权值向量 $\mathbf{w}$ 的欧几里得范数对分离超平面集合施加一个结构。特别地，我们可以将定理表述如下（Vapnik，1995，1998）：

令 D 表示包括所有输入向量 $x_1,x_2\cdots x_N$ 的最小球的直径。由方程定义的最优超平面

$$\mathbf{w}_o^T\mathbf{x} + b_o = 0$$

有一个 VC 维数 h 的上界

$$h \leqslant \min\left\{\left\lceil\frac{D^2}{\rho^2}\right\rceil, m_0\right\} + 1 \tag{6.19}$$

其中顶符号$\lceil\cdot\rceil$表示大于等于所包含的数值的最小整数，ρ 是等于 $2/\|\mathbf{w}_o\|$的分离边缘，m_0 是输入

空间的维数。

正如第 4 章提到的，Vapnik-Chervonenkis 维，简称 VC 维，提供了一种空间函数复杂度的度量。这个定理告诉我们，可以尝试通过正确选择分离边缘 ρ，控制最优超平面的 VC 维数（即复杂性），它与输入空间的维数 m_0 无关。

假定，有一个通过分离超平面描述的嵌套结构如下：

$$S_k = \{\mathbf{w}^T\mathbf{x} + b_1 \mid \|\mathbf{w}\|^2 \leqslant c_k\}, \qquad k = 1,2,\cdots \tag{6.20}$$

由 VC 维数 h 在式(6.19)定义的上界，在式(6.20)中描述的嵌套结构可以通过分离边缘改写为等价形式

$$S_k = \left\{\left\lceil \frac{r^2}{\rho^2} \right\rceil + 1 ; \rho^2 \geqslant a_k\right\}, \qquad k = 1,2,\cdots \tag{6.21}$$

其中 a_k 和 c_k 都是常数。

式(6.20)说明了最优超平面是使正反比例之间的隔离边缘达到最大可能的平面。等价地，式(6.21)说明了通过最小化权值向量 $\mathbf{w}$ 的欧几里得范数建立最优超平面。一定意义上，上述方程更加肯定了我们对式(6.9)所做出的结论。

6.3　不可分模式的最优超平面

到目前为止重点关注线性可分模式的情况。本节我们考虑更难的不可分模式的情况。给定这样一组训练数据，肯定不能建立一个不具有分类误差的分离超平面。然而，我们希望找到一个最优超平面，使之对整个训练集合平均的分类误差的概率达到最小。

在类之间的分离边缘称为是软的，如果数据点 (x_i, d_i) 不满足下面的条件（见式(6.10)）：

$$d_i(\mathbf{w}^T\mathbf{x}_i + b) \geqslant +1, \qquad i = 1,2,\cdots,N$$

这种违反条件以下面两种方式之一出现：

- 数据点 $(\mathbf{x}_i, d_i)$ 落在分离区域之内，但在决策面正确的一侧，如图 6.3a 所示。
- 数据点 $(\mathbf{x}_i, d_i)$ 落在决策面错误的一侧，如图 6.3b 所示。

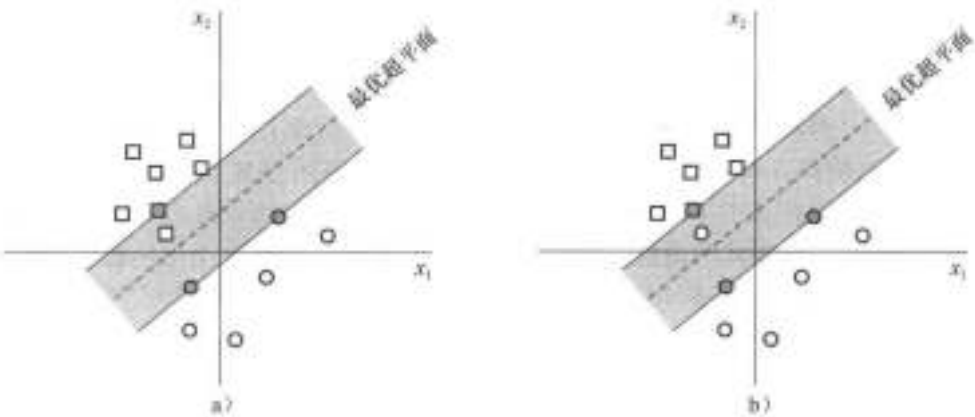


图 6.3　软分离边缘平面：a）数据点 $\mathbf{x}_i$（属于类别 1，用小方块表示）落入了分离区域内，但是在决策平面的正确一面；b）数据点 $\mathbf{x}_i$（属于类别 1，用小圆圈表示）落入决策平面的错误一面

注意，在情况 1 我们有正确的分类，但在情况 2 分类是错误的。

为了处理不可分离数据点，我们引入一组新的非负标量变量 $\{\xi_i\}_{i=1}^N$ 到分离超平面（即决策面）的定义中，表示为

$$d_i(\mathbf{w}^T\mathbf{x}_i + b) \geqslant 1 - \xi_i, \qquad i = 1,2,\cdots,N \tag{6.22}$$

这里 ξ_i 称为松弛变量（slack variable），它们度量一个数据点对模式可分的理想条件的偏离程

度。当 $0<\xi_i\leqslant 1$ 时，数据点落入分离区域的内部，但是在决策面的正确一侧，如图 6.3a 所示。当 $\xi_i>1$ 时，数据点落到分离超平面的错误一侧，如图 6.3b 所示。注意到支持向量是那些精确满足式(6.22)的特殊数据点，即使 $\xi_i>0$。此外满足 $\xi_i=0$ 的点也是支持向量。注意，如果一个对应的样本 $\xi_i>0$ 被遗弃在训练集外，决策面就要改变。因此，支持向量的定义对线性可分和不可分的情况都是相同的。

我们的目标是找到分离超平面使其在训练集上的平均错误分类的误差最小。我们可以通过最小化关于权值向量 $\mathbf{w}$ 的泛函达到此目的

$$\Phi(\boldsymbol{\xi})=\sum_{i=1}^{N} I(\xi_i-1)$$

泛函满足式(6.22)的约束条件和对 $\|\mathbf{w}\|^2$ 的限制。函数 $I(\xi)$是一个指标函数，定义为

$$I(\xi)=\begin{cases}0, & \xi\leqslant 0\\ 1, & \xi>0\end{cases}$$

不幸的是，$\Phi(\xi)$对 $\mathbf{w}$ 的最小化是非凸的最优化问题，它是 NP-完全的[5]。

为了使最优化问题数学上易解，为了逼近泛函 $\Phi(\xi)$重写函数：

$$\Phi(\boldsymbol{\xi})=\sum_{i=1}^{N}\xi_i$$

而且，通过形成泛函对权值向量 w 的最小化公式简化计算，得出

$$\Phi(\mathbf{w},\boldsymbol{\xi})=\frac{1}{2}\mathbf{w}^T\mathbf{w}+C\sum_{i=1}^{N}\xi_i \tag{6.23}$$

像前面一样，最小化式(6.23)中第 1 项与最小化支持向量机的 VC 维数有关。至于第 2 项 $\sum_i\xi_i$，它是测试错误数目的一个上界。

参数 C 控制机器的复杂性和不可分离点数之间的平衡；它也可以被看作是通常被称为“正则化”参数的倒数[6]。当参数 C 选得比较大的时候，暗示着支持向量机的设计对训练样本$\mathcal{T}$的质量具有高度的信心。相反，当参数 C 选得比较小的时候，认为训练样本$\mathcal{T}$中存在噪声，因此将对其不太强调。

在任何情况下，参数 C 由用户指定。也可通过使用训练（验证）集由实验决定，这属于粗略的重采样形式；在第 7 章讨论使用交叉验证来优化选择正则参数（即，$1/C$）。

在任何情况下，都对泛函 $\Phi(\mathbf{w},\boldsymbol{\xi})$ 关于 w 和$\{\xi_i\}_{i=1}^{N}$求最优化，要求满足式(6.22)描述的约束条件和 $\xi_i\geqslant 0$。这样做，$\mathbf{w}$ 的范数平方被认为是一个关于不可分离点的联合最小化中一个数量项，而不是作为强加在关于不可分离点数量的最小化上的一个约束条件。

对刚刚陈述的不可分模式的最优化问题而言，线性可分模式的最优化问题可作为它的一种特殊情况。具体地讲，在式(6.22)和式(6.23)中对所有的 i 置 $\xi_i=0$，就把它们化简为相应的线性可分情形。

我们现在对不可分离的情况的原问题正式地陈述如下：

给定训练样本 $\{(x_i,d_i)\}_{i=1}^{N}$，寻找权值向量 $\mathbf{w}$ 和偏置 b 的最优值，使得它们满足约束条件

$$d_i(\mathbf{w}^T\mathbf{x}_i+b)\geqslant 1-\xi_i,\quad \text{当 } i=1,2,\cdots,N \tag{6.24}$$

$$\xi_i\geqslant 0,\quad \text{对所有 } i \tag{6.25}$$

并且使得权值向量 $\mathbf{w}$ 和松弛变量 ξ_i 最小化代价函数

$$\Phi(\mathbf{w},\boldsymbol{\xi})=\frac{1}{2}\mathbf{w}^T\mathbf{w}+C\sum_{i=1}^{N}\xi_i \tag{6.26}$$

其中，C 是用户选定的正参数。

使用拉格朗日乘子方法，以一种与6.2节所描述的相似方式来处理，可以得到不可分离模式的对偶问题的表示如下（参看习题6.3）：

给定训练样本 $\{(x_i,d_i)\}_{i=1}^N$，寻找拉格朗日乘子 $\{\alpha_i\}_{i=1}^N$ 来最大化目标函数

$$Q(\alpha)=\sum_{i=1}^{N}\alpha_i-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j d_i d_j \mathbf{x}_i^T\mathbf{x}_j \tag{6.27}$$

并满足约束条件

$$(1)\sum_{i=1}^{N}\alpha_i d_i=0$$

$$(2)0\leqslant\alpha_i\leqslant C \qquad \text{当 } i=1,2,\cdots,N$$

其中，C 是使用者选定的正参数。

注意，松弛变量 ξ_i 及其拉格朗日乘子都不出现在对偶问题中。除了少量但很重要的差别外，不可分模式的对偶问题与线性可分模式的简单情况相似。在这两种情况下，要最大化的目标函数 $Q(\alpha)$ 是相同的。不可分离情况与可分离情况的不同在于，限制条件 $\alpha_i\geqslant 0$ 被替换为条件更强的 $0\leqslant\alpha_i\leqslant C$。除了这个变化，不可分离情况的约束最优化问题和权值向量 $\mathbf{w}$ 和偏置 b 的最优值计算过程与线性可分情况一样。还要注意支持向量和以前的定义相同。

无界的支持向量

对于一个规定的参数 C，满足 $0<\alpha_i<C$ 的点 (x_i,d_i) 称为无界或者自由支持向量。当 $\alpha_i=C$ 时，我们发现

$$d_i F(\mathbf{x}_i)\leqslant 1, \qquad \alpha_i=C$$

其中 $F(\mathbf{x}_i)$ 是 $\mathbf{x}_i$ 通过支持向量机实现的近似函数。另一方面，当 $\alpha_i=0$ 时，我们发现

$$d_i F(\mathbf{x}_i)\geqslant 1, \qquad \alpha_i=0$$

就上述两个方面而言，对于无界的支持向量有

$$d_i F(\mathbf{x}_i)=1$$

不幸的是逆命题是不成立的，即对于特定的点 $(\mathbf{x}_i,d_i)$ 有 $dF(\mathbf{x}_i)=1$ 成立，这个条件不能必然地说明相应的拉格朗日乘子 α_i 的情况。

因此，通过支持向量机来解决模式分类问题的时候存在明显退化的可能性（即，弱化的最优化条件）。由此，我们说一个点 $(\mathbf{x}_i,d_i)$ 精确满足隔离边缘要求是指对相应的乘子 α_i 可能的值没有限制。

Rifkin(2002) 讨论了就计算而言，无界支持向量的个数是对支持向量机进行训练的难度的主要原因。

用于模式识别的支持向量机的潜在思想

有了关于对不可分离模式如何找到最优超平面的知识后，我们现在建立用于模式识别任务的支持向量机。

从根本上说，支持向量机的关键在于如图6.4中说明和总结的两个数学运算：

1. 输入向量到高维特征空间的非线性映射，对输入和输出特征空间都是隐藏的。
2. 构造一个最优超平面用于分离在第1步中发现的特征。

两个操作的基本理由在下面解释。

作为最后重要的注释，支持向量的个数决定了图6.4隐藏空间特征的个数。所以，支持向量理论提供了有关决定特征空间特征优化个数的分析方法，从而保证了对于分类任务的最优性。

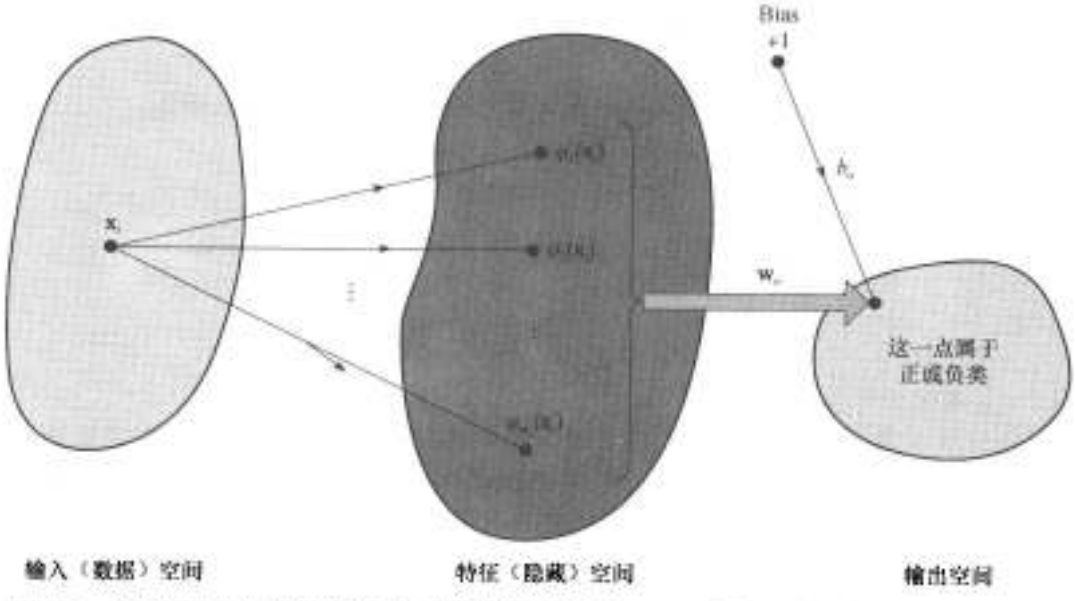


图 6.4 说明支持向量机用来处理模式分类的两个映射：（Ⅰ）输入空间到特征空间的非线性映射；（Ⅱ）特征空间到输出空间的线性映射

6.4 使用核方法的支持向量机

内积核

令 x 表示从输入空间中取出的向量，假定维数为 m_0。令 $\{\varphi_j(\mathbf{x})\}_{j=1}^{m_1}$ 表示一系列非线性函数的集合，从维数 m_0 的输入空间转换成无限维输出空间。给定这样的变换，我们可以定义一个与方程一致的充当决策面的超平面

$$\sum_{j=1}^{\infty} w_j \varphi_j(\mathbf{x}) = 0 \tag{6.28}$$

其中 $\{w_j\}_{j=1}^{\infty}$ 表示把特征空间转换成输出空间的无限大的权值集合。在输出空间中，由决策平面决定输入空间中的点 $\mathbf{x}$ 属于两个可能类之一：正例或者反例。为了表示方便，我们将式(6.28)中的偏置设为0。使用矩阵的观点，重写等式为如下的紧凑形式

$$\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) = 0 \tag{6.29}$$

其中 $\boldsymbol{\phi}(\mathbf{x})$ 是特征向量，$\mathbf{w}$ 是相应的权重向量。

正如 6.3 节所述，我们试图寻找在特征空间中“转化后模式的线性可分性”。带着这个目标，可以将式(6.17)的形式用权重向量改写成下列形式：

$$\mathbf{w} = \sum_{i=1}^{N_s} \alpha_i d_i \boldsymbol{\phi}(\mathbf{x}_i) \tag{6.30}$$

其中特征向量表示为：

$$\boldsymbol{\phi}(\mathbf{x}_i) = [\varphi_1(\mathbf{x}_i), \varphi_2(\mathbf{x}_i), \cdots]^T \tag{6.31}$$

N_s 是支持向量的个数。所以，把式(6.29)代入式(6.30)中，将输出空间中的决策面表示为：

$$\sum_{i=1}^{N_s} \alpha_i d_i \boldsymbol{\phi}^T(\mathbf{x}_i) \boldsymbol{\phi}(\mathbf{x}) = 0 \tag{6.32}$$

我们立刻注意到式(6.32)中的标量项 $\boldsymbol{\phi}^T(\mathbf{x}_i)\boldsymbol{\phi}(\mathbf{x})$ 代表一个内积。相应地，将这个内积写成标量

$$k(\mathbf{x},\mathbf{x}_i)=\boldsymbol{\phi}^T(\mathbf{x}_i)\,\boldsymbol{\phi}(\mathbf{x})=\sum_{j=1}^{\infty}\varphi_j(\mathbf{x}_i)\varphi_j(\mathbf{x}),\qquad i=1,2,\cdots,N_s \tag{6.33}$$

相应地，可以将输出空间的决策超平面（超平面）写成

$$\sum_{i=1}^{N_s}\alpha_i d_i k(\mathbf{x},\mathbf{x}_i)=0 \tag{6.34}$$

函数 $k(\mathbf{x},\mathbf{x}_i)$被称为内积核[7]，或者简称核，正式定义如下（Shawe-Taylor and Cristianini，2004）：

核函数 $k(\mathbf{x},\mathbf{x}_i)$是这样一种函数，计算嵌入 Φ 输入空间的两个数据点在特征空间中像的内积。

根据第 5 章引入的核的定义，我们可以说明核 $k(\mathbf{x},\mathbf{x}_i)$是具有两个基本特点[8] 的函数

特点 1　*内积核是自变量的对称函数，表示为*

$$k(\mathbf{x},\mathbf{x}_i)=k(\mathbf{x}_i,\mathbf{x}),\quad \text{对所有的 } \mathbf{x}_i,$$

当 $\mathbf{x}=\mathbf{x}_i$ 时达到最大值。

注意，最大值不一定出现；例如 $k(\mathbf{x},\mathbf{x}_i)=\mathbf{x}^T\mathbf{x}_i$，作为核没有最大值。

特点 2　*在一个平面上的核函数 $k(\mathbf{x},\mathbf{x}_i)$的总和是一个常数。*

如果可以使核 $k(\mathbf{x},\mathbf{x}_i)$通过合适的规划使得在特点 2 下的常数变成单位数，它将会具有类似于一个随机变量的概率密度函数的性质。

核技巧

检查式(6.34)，我们可以得出两点重要的观察：

1. 就模式分类的输出空间而言，具体指定核函数 $k(\mathbf{x},\mathbf{x}_i)$是充分的。换句话说，无需显式计算出权重向量 $\mathbf{w}_o$；这也是把式(6.33)的应用称为核技巧的原因。

2. 即使假设特征空间是无限维的，但式(6.34)也定义了包括有限项的最优超平面，项的数目与分类器中训练模式的个数相等。

就观察 1 而言支持向量机也被称为核机器。对于模式分类，机器是由一个 N 维向量参数化的，其中第 i 个参数是 $\alpha_i d_i, i=1,2,\cdots,N$。

我们可以将核函数 $k(\mathbf{x}_i,\ \mathbf{x}_j)$看成一个 $N\times N$ 对称矩阵的 ij_i 个元素矩阵

$$\mathbf{K}=\{k(\mathbf{x}_i,\mathbf{x}_j)\}_{i,j=1}^{N} \tag{6.35}$$

$\mathbf{K}$ 是一个非负定矩阵，被称为核矩阵；通常也简称为 Gram 矩阵。它的非负性或者半正定性是指对于任何与矩阵 $\mathbf{K}$ 可以相容的实向量 $\mathbf{a}$ 满足以下条件：

$$\mathbf{a}^T\mathbf{K}\mathbf{a}\geqslant 0$$

Mercer 定理

式(6.33)对于对称核函数 $k(\mathbf{x},\mathbf{x}_i)$的展开是在泛函分析中出现的 Mercer 定理的一种特殊情形。这个定理可以正式表述如下（Mercer，1908；Courant and Hilbert，1970）：

$k(\mathbf{x},\mathbf{x}')$表示一个连续的对称核，其中 x 定义在闭区间 $\mathbf{a}\leqslant\mathbf{x}\leqslant\mathbf{b}$ 上，$\mathbf{x}'$和 $\mathbf{x}$ 类似。核 $k(\mathbf{x},\mathbf{x}')$可以被展开为级数

$$k(\mathbf{x},\mathbf{x}')=\sum_{i=1}^{\infty}\lambda_i\varphi_i(\mathbf{x})\varphi_i(\mathbf{x}') \tag{6.36}$$

其中所有的 λ_i 均是正的。为了保证这个展开式是合理的并且为绝对一致收敛的，充要条件是

$$\int_b^a\int_b^a k(\mathbf{x},\mathbf{x}')\psi(\mathbf{x})\psi(\mathbf{x}')\mathrm{d}\mathbf{x}\mathrm{d}\mathbf{x}' \geqslant 0 \tag{6.37}$$

对于所有的 $\psi(\cdot)$ 成立，这样就有

$$\int_b^a \psi^2(\mathbf{x})\mathrm{d}\mathbf{x} < \infty \tag{6.38}$$

成立，其中 **a** 和 **b** 是实整数。

函数 $\varphi_i(\mathbf{x})$称为展开的特征函数，λ_i 称为特征值。所有的特征值均为正数，这个事实意味着核 $k(\mathbf{x},\mathbf{x}')$是正定的。反之，这个特点意味着对于权重向量 **w** 我们可以有效地解决复杂的问题，这将在以后讨论。

但是注意，Mercer 定理只是告诉我们对于有的空间是否存在一个候选的核是积核，因此是否能被支持向量机采用。但是它没有告诉我们如何去构造函数 $\varphi_i(\mathbf{x})$；需要我们自己去构造。不过，Mercer 定理是重要的，原因在于对于可用核的数量进行了限制。注意到式(6.33)是 Mercer 定理的特殊形式，因为所有的特征值都已经归一到单位范围内。这也就是为什么内积核被称为 Mercer 核的原因。

6.5　支持向量机的设计

式(6.33)的内积核 $k(\mathbf{x},\mathbf{x}_i)$的展开式允许我们建立一个决策面，在输入空间中它是非线性的，但它在特征空间的像是线性的。有了这个展开式，我们现在对支持向量机受约束的最优化的对偶形式陈述如下：

给定训练样本$\{(\mathbf{x}_i, d_i)\}_{i=1}^N$，寻找拉格朗日乘子$\{\alpha_i\}_{i=1}^N$以最大化目标函数

$$Q(\alpha) = \sum_{i=1}^N \alpha_i - \frac{1}{2}\sum_{i=1}^N\sum_{j=1}^N \alpha_i\alpha_j d_i d_j k(\mathbf{x}_i,\mathbf{x}_j) \tag{6.39}$$

并满足约束条件

$$(1)\sum_{i=1}^N \alpha_i d_i = 0$$

$$(2)0 \leqslant \alpha_i \leqslant C \qquad \text{当 } i = 1,2,\cdots,N$$

其中，C 是用户选定的正参数。

约束条件 (1) 由拉格朗日函数 $Q(\alpha)$对应的偏置 b 的最优化产生，是式(6.13)的重写。这里陈述的对偶问题与在第 6.3 节中考虑的不可分模式情况的形式相同，除了内积 $\mathbf{x}_i^T\mathbf{x}_j$ 被内积核 $k(\mathbf{x},\mathbf{x}_j)$所代替。

支持向量机的例子

核 $k(\mathbf{x},\mathbf{x}_i)$的要求是满足 Mercer 定理。只要满足这个要求，怎样选择它是有一定自由度的。表 6.1 总结了支持向量机的三个普遍类型的内积核函数：多项式学习机器、径向基函数网络和两层感知器。下面几点是值得注意的：

1. 用于支持向量机的多项式核和径向基函数核通常满足 Mercer 定理。相反，用于支持向量机的两层感知器的类型，其内积核受到某种限制，如表 6.1 最后一行所示。后面的条目证实如下的事实：判定一个给定的核是否符合 Mercer 定理确实是一件困难的事情。

2. 对所有三种机器类型，特征空间维数由从训练数据抽取的支持向量的个数决定，这些训练数据是通过解决受约束最优化问题来获得的。

3. 支持向量机的基本理论避免启发式的需要，它们常被用在传统的径向基函数网络和多

层感知器的设计上面。

4. 在径向基函数类型的支持向量机中，径向基函数的数量和它们的中心分别由支持向量的个数和支持向量的值自动决定。

表 6.1　Mercer 核总结

支持向量种类	Mercer 核 $k(\mathbf{x},\mathbf{x}_i), i=1,2,\cdots,N$	评论
多项式学习机器	$(\mathbf{x}^T\mathbf{x}_i+1)^p$	用户事先指定指数 p
径向基函数网络	$\exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x}-\mathbf{x}_i\|^2\right)$	和所有核一样，由用户实现指定宽度 σ^2
两层感知器	$\tanh(\beta_0\mathbf{x}^T\mathbf{x}_i+\beta_1)$	对于某些 β_0 和 β_1 满足 Mercer 定理

图 6.5 显示一个支持向量机的体系结构，其中 m_1 是隐藏层的大小（如特征空间）。

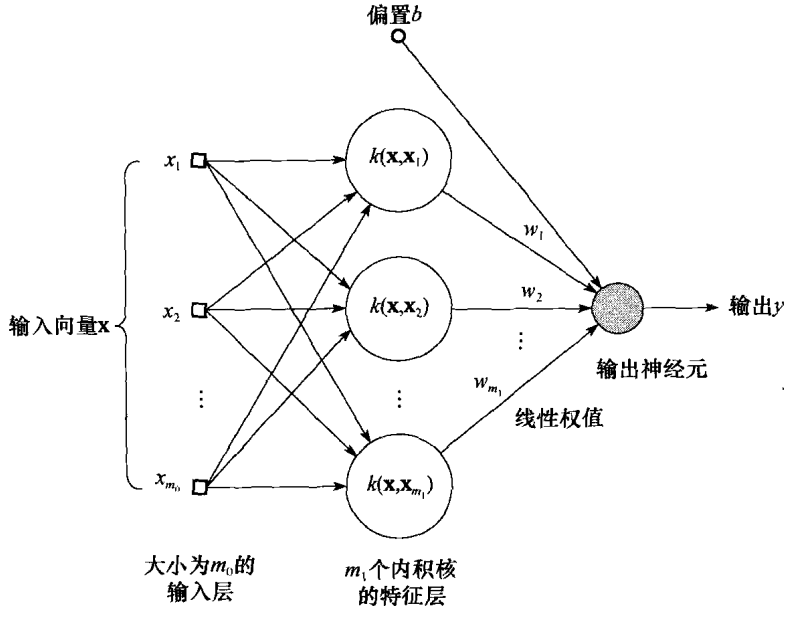


图 6.5　使用径向基函数网络的支持向量机的结构

不管支持向量机是如何实现的，基本上它与传统的设计多层感知器的方法不同。在传统的方法中，模型复杂性由保持特征（即隐藏神经元）的数量最小所控制。另一方面，支持向量机提供一个机器学习设计的解决方案，其模型复杂性的控制独立于维数，总结如下（Vapnik，1995，1998）：

- 概念问题　有意使特征（隐藏）空间的维数足够大，使得可以在这个空间建立超平面形式的决策面。为了一个好的泛化性能，模型的复杂性通过对所建立的超平面添加一些特定的约束条件来控制，这导致训练数据中的一小部分被抽出来作为支持向量。
- 计算问题　通过使用核技巧可以避免计算径向基函数网络输出层中的权重向量和偏置。

6.6　XOR 问题

要说明支持向量机设计过程，我们再次讨论在第 4 章和第 5 章讨论过的 XOR（异或）问题。表 6.2 给出了 4 个可能状态的输入向量和期望的响应。

表 6.2　XOR 问题

输入向量 $\mathbf{x}$	期望的响应 d
$(-1, -1)$	-1
$(-1, +1)$	$+1$
$(+1, -1)$	$+1$
$(+1, +1)$	-1

为了继续讨论，我们定义如下核（Cherkassky and Mulier，1998）：

$$k(\mathbf{x},\mathbf{x}_i) = (1+\mathbf{x}^T\mathbf{x}_i)^2 \tag{6.40}$$

$\mathbf{x} = [x_1, x_2]^T$ 和 $\mathbf{x}_i = [x_{i1}, x_{i2}]^T$，内积核 $k(\mathbf{x},\mathbf{x}_i)$可应用不同次数的单项式表示如下：

$$k(\mathbf{x},\mathbf{x}_i) = 1 + x_1^2 x_{i1}^2 + 2x_1 x_2 x_{i1} x_{i2} + x_2^2 x_{i2}^2 + 2x_1 x_{i1} + 2x_2 x_{i2} \tag{6.41}$$

输入向量 $\mathbf{x}$ 在特征空间中诱导的像可推断为

$$\boldsymbol{\phi}(\mathbf{x}) = [1, x_1^2, \sqrt{2}x_1 x_2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2]^T$$

类似地

$$\boldsymbol{\phi}(\mathbf{x}_i) = [1, x_{i1}^2, \sqrt{2}x_{i1}x_{i2}, x_{i2}^2, \sqrt{2}x_{i1}, \sqrt{2}x_{i2}]^T \qquad i = 1,2,3,4 \tag{6.42}$$

使用式(6.35)中的定义，得到 Gram 矩阵

$$\mathbf{K} = \begin{bmatrix} 9 & 1 & 1 & 1 \\ 1 & 9 & 1 & 1 \\ 1 & 1 & 9 & 1 \\ 1 & 1 & 1 & 9 \end{bmatrix}$$

因此目标函数的对偶形式为（参见式(6.39)）：

$$\begin{aligned} Q(\alpha) = {} & \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 - \frac{1}{2}(9\alpha_1^2 - 2\alpha_1\alpha_2 - 2\alpha_1\alpha_3 + 2\alpha_1\alpha_4 \\ & + 9\alpha_2^2 + 2\alpha_2\alpha_3 - 2\alpha_2\alpha_4 + 9\alpha_3^2 - 2\alpha_3\alpha_4 + 9\alpha_4^2) \end{aligned} \tag{6.43}$$

对拉格朗日乘子优化 $Q(\alpha)$产生下列联立方程组：

$$\begin{aligned} 9\alpha_1 - \alpha_2 - \alpha_3 + \alpha_4 &= 1 \\ -\alpha_1 + 9\alpha_2 + \alpha_3 - \alpha_4 &= 1 \\ -\alpha_1 + \alpha_2 + 9\alpha_3 - \alpha_4 &= 1 \\ \alpha_1 - \alpha_2 - \alpha_3 + 9\alpha_4 &= 1 \end{aligned}$$

因此，拉格朗日乘子的最优值为：

$$\alpha_{o,1} = \alpha_{o,2} = \alpha_{o,3} = \alpha_{o,4} = \frac{1}{8}$$

这个结果说明，本例中所有 4 个输入向量$\{\mathbf{x}_i\}_{i=1}^4$都是支持向量。$Q(\alpha)$的最优值是

$$Q_o(\alpha) = \frac{1}{4}$$

相应地，可写出

$$\frac{1}{2}\|\mathbf{w}_o\|^2 = \frac{1}{4}$$

或者

$$\|\mathbf{w}_o\| = \frac{1}{\sqrt{2}}$$

从式(6.30)中，可以找到优化权重向量

$$\mathbf{w}_o = \frac{1}{8}[-\varphi(\mathbf{x}_1)+\varphi(\mathbf{x}_2)+\varphi(\mathbf{x}_3)-\varphi(\mathbf{x}_4)]$$

$$= \frac{1}{8}\left[-\begin{bmatrix}1\\1\\\sqrt{2}\\1\\-\sqrt{2}\\-\sqrt{2}\end{bmatrix}+\begin{bmatrix}1\\1\\-\sqrt{2}\\1\\-\sqrt{2}\\\sqrt{2}\end{bmatrix}+\begin{bmatrix}1\\1\\-\sqrt{2}\\1\\\sqrt{2}\\-\sqrt{2}\end{bmatrix}-\begin{bmatrix}1\\1\\\sqrt{2}\\1\\\sqrt{2}\\\sqrt{2}\end{bmatrix}\right]=\begin{bmatrix}0\\0\\-1/\sqrt{2}\\0\\0\\0\end{bmatrix}$$

$\mathbf{w}_0$ 的第一个分量表示偏置 b 为 0。

最优超平面定义为

$$\mathbf{w}_o^T\,\boldsymbol{\phi}(\mathbf{x}) = 0$$

扩展内积 $\mathbf{w}_0{}^T\,\boldsymbol{\phi}(\mathbf{x})$ 产生

$$\left[0,0,\frac{-1}{\sqrt{2}},0,0,0\right]\begin{bmatrix}1\\x_1^2\\\sqrt{2}x_1x_2\\x_2^2\\\sqrt{2}x_1\\\sqrt{2}x_2\end{bmatrix}=0$$

这归结为

$$-x_1x_2 = 0$$

关于 XOR 问题的多项式形式的支持向量机参见图 6.6a。对 $x_1=x_2=-1$ 和 $x_1=x_2=+1$，输出 $y=-1$；对 $x_1=-1$，$x_2=+1$ 以及 $x_1=+1$，$x_2=-1$，输出 $y=+1$。因此如图 6.6b 所示，XOR 问题获得解。

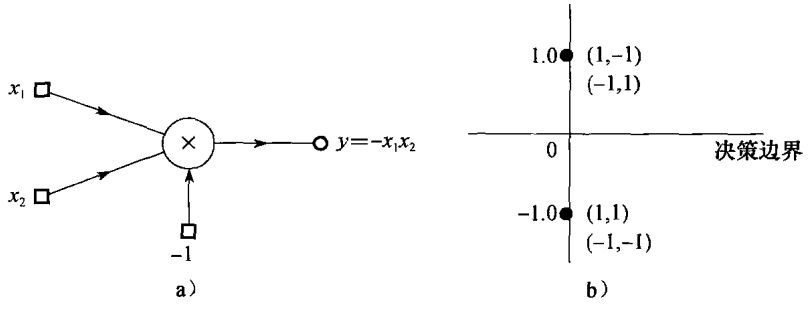


图 6.6　a）多项式机器学习用来解决 XOR 问题；b）XOR 问题的四个点推导出的特征空间的像

6.7　计算机实验：模式分类

本节通过图 1.8 中的双月问题来讨论模式分类问题。这次，我们使用单隐层的非线性支持向量机。实验在垂直可分的两种不同的环境下进行，分别为 $d=-6.0$ 和 $d=-6.5$。设两个实验参数 C 为无穷。训练样本包括 300 个样本点，测试样本是 2 000 个数据。同样采用 1.5 节中数据预处理的方式。

实验第一部分我们采用距离 $d=-6.0$ 的方案为的是提供例证方式，这种方式将提供 SVM 和第 5 章中的用来训练 RBF 网络的“K-均值，RLS”算法的对比，该算法的训练误差很小。

图6.7展示了$d=-6.0$时支持向量机的计算结果。图6.7a显示当$d=-6.0$时的结果，显示相应的支持向量和决策边界。从图6.7b中我们可以看到，对于未见过的数据的分类误差为零。

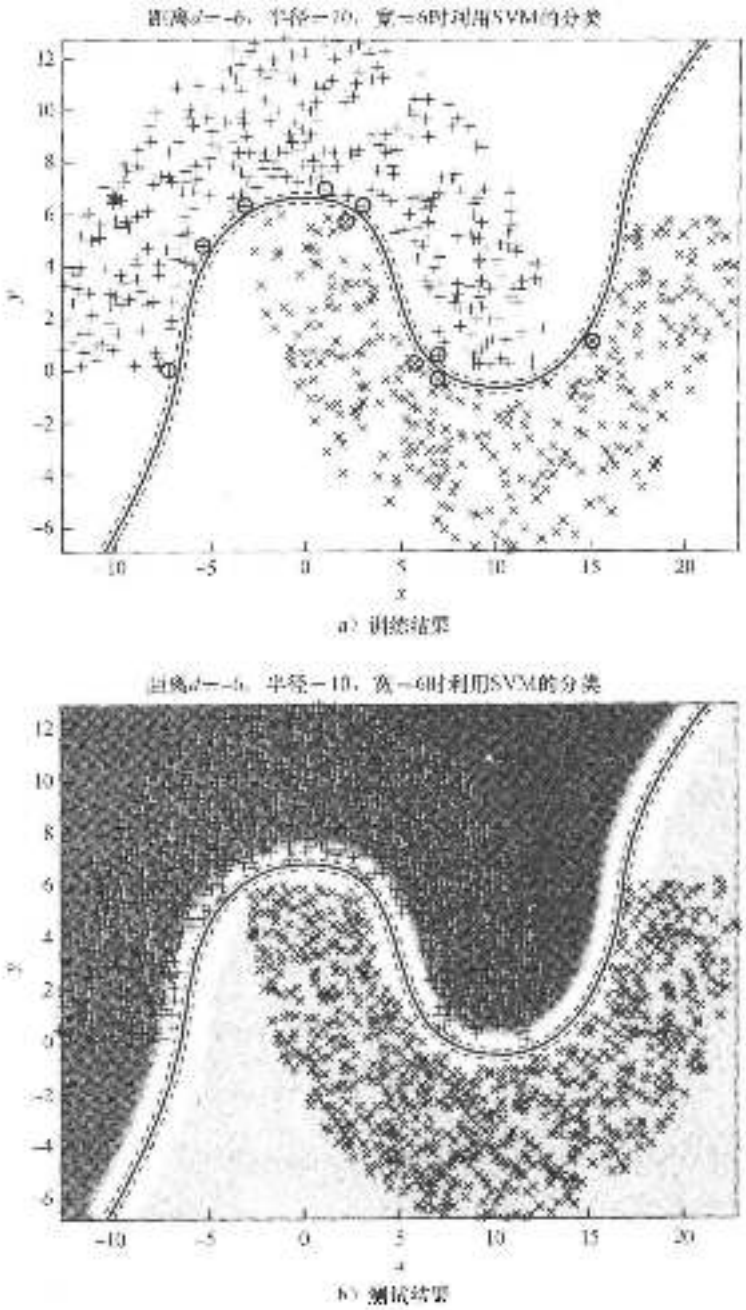


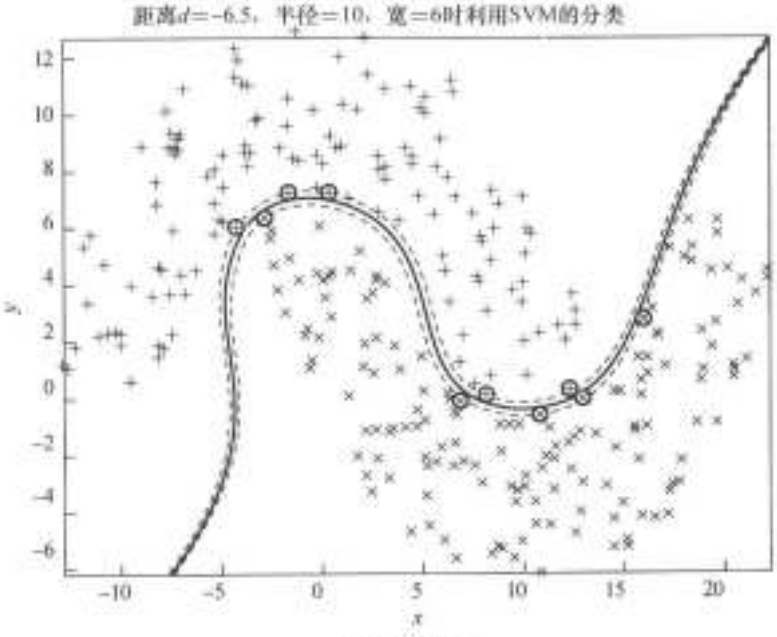
图6.7 当距离$d=-6$时SVM应用于图1.8双月的结果

图6.8显示了实验的第二部分，在复杂的情况下使用了SVM，其中两个月亮之间的垂直距离为$d=-6.5$。同样，图6.8a显示了相应的支持向量和决策边界，图6.8b显示了相应的测试结果。这次，在2000个测试数据中有11个分错，所以误分率为0.55%。

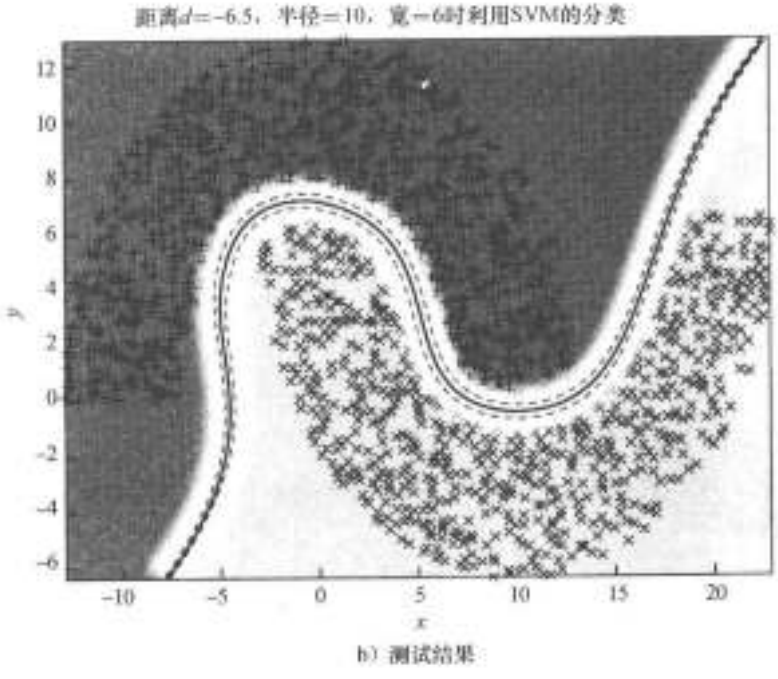
如前所说，试验的两个部分都采用了 $C=\infty$，在这种环境下，需要考虑如下两方面：

1. $d=-6.0$ 时，两个月亮是非常好的非线性可分情形；如图 6.7b 所示，测试数据上没有误差恰好证明了这点。

2. $d=-6.5$ 时，图 1.8 中的两个月亮轻微地重合。相应地，不再是可分的，图 6.8b 中证明了测试数据误差很小。在实验的第二部分，没有寻找优化的 C 来使训练误差变小；这个问题将在习题 6.24 中解决。



a) 训练结果



b) 测试结果

图 6.8 当距离 $d=-6.5$ 时 SVM 应用于图 1.8 双月的结果

6.8 回归：鲁棒性考虑

本章到目前为止，我们重点关注利用支持向量机求解模式识别任务。现在，我们将考虑利用支持向量机求解非线性回归问题。为了准备这个讨论，我们首先讨论适合这类学习任务的最优化准则问题，以鲁棒性作为主要目标。有了这样的目标，我们需要一个模型，该模型对模型参数中小的变化不敏感，这在后面解决。

ε-不敏感损失函数

以鲁棒性作为设计目标，对于任何鲁棒性的定量测量必须考虑到由于微小噪声模型的一个ε-偏差而可能产生最大性能退化。根据这种观点，一种*最优鲁棒估计过程*是最小化最大的性能恶化，因而是一种*最小最大过程*[9]（Huber，1981）。当加性噪声的概率密度函数关于原点对称时，求解非线性回归问题的最小最大过程利用绝对误差作为被最小化的量（Huber，1964），也就是说，损失函数具有以下形式：

$$L(d,y)=|d-y| \tag{6.44}$$

其中 d 是期望响应，$y=\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x})$ 是相应的估计量输出。

为了构造支持向量机逼近期望的响应 d，我们利用式(6.44)的损失函数的扩展，它由Vapnik(1995，1998)最早提出，描述为

$$L_\varepsilon(d,y)=\begin{cases}|d-y|-\varepsilon, & \text{当 } |d-y|\geqslant\varepsilon\\ 0, & \text{否则}\end{cases} \tag{6.45}$$

ε 是指定的参数，损失函数 $L_\varepsilon(d,y)$ 称为ε-*不敏感损失函数*（ε-insensitive loss function）。如果估计器输出 y 和期望输出 d 的偏差的绝对值小于ε，则它等于零，否则它等于偏差绝对值减去ε。式(6.44)的损失函数是ε-不敏感损失函数在 $\varepsilon=0$ 时的特殊情形，图6.9说明 $L_\varepsilon(d,y)$ 和误差（$d-y$）的依赖关系。

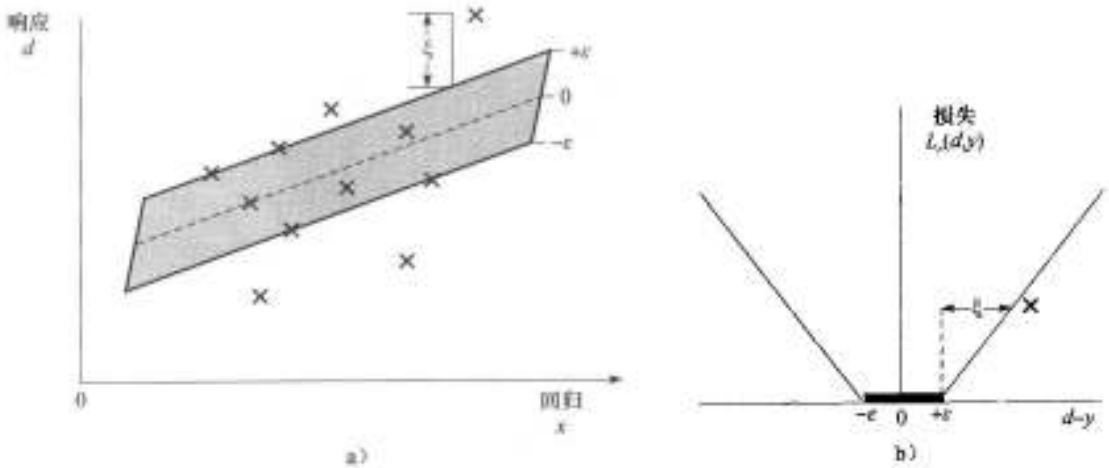


图6.9 线性回归：a）图解半径为ε的ε-不敏感区域，使适应于用×表示的数据点；b）相应的ε-不敏感函数的图

以式(6.45)中的ε-不敏感损失函数作为鲁棒性的基础，我们在后面讨论应用支持向量机理论来解决线性回归问题。

6.9 线性回归问题的最优化解

考虑线性回归模型，标量 d 对向量 $\mathbf{x}$ 的依赖可描述为

$$d = \mathbf{w}^T\mathbf{x} + b \tag{6.46}$$

其中参数向量 $\mathbf{w}$ 和偏置 b 都是未知的。问题是给定训练样本$\mathcal{T}=\{(\mathbf{x}_i,d_i)\}_{i=1}^N$来估计这两个参数，其中数据来自于独立同分布（iid）。

给定训练样本$\mathcal{T}$，考虑风险函数

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}|y_i - d_i|_\varepsilon \tag{6.47}$$

其中加和是 ε-不敏感训练误差的总和，C 是一个训练误差和惩罚项 $\|\mathbf{w}\|^2$ 之间的权衡。y_i 是输入样本 $\mathbf{x}_i$ 对应的估计输出。为了达到要求我们做如下处理。

最小化式(6.47)中的风险函数，约束如下：

$$d_i - y_i \leqslant \varepsilon + \xi_i \tag{6.48}$$

$$y_i - d_i \leqslant \varepsilon + \xi_i' \tag{6.49}$$

$$\xi_i \geqslant 0 \tag{6.50}$$

$$\xi_i' \geqslant 0 \tag{6.51}$$

其中，$i = 1,2,\cdots,N$。ξ_i 和 ξ_i'是两个非负松弛向量，用来描述式(6.45)中 ε-敏感损失函数。

为了解这个优化问题中的拉格朗日乘子 α_i 和 α_i'，可以使用 6.2 节中处理线性可分模式的方法。首先，建造一个拉格朗日函数（包括约束条件），我们将继续相应的对偶变量集。具体地，首先写出函数

$$\begin{aligned} J(\mathbf{w},\xi,\xi_i',\alpha,\alpha',\gamma,\gamma') = {} & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}(\xi_i + \xi_i') - \sum_{i=1}^{N}(\gamma_i\xi_i + \gamma_i'\xi_i') \\ & - \sum_{i=1}^{N}\alpha_i(\mathbf{w}^T\mathbf{x}_i + b - d_i + \varepsilon + \xi_i) \\ & - \sum_{i=1}^{N}\alpha_i'(d_i - \mathbf{w}^T\mathbf{x}_i - b + \varepsilon + \xi_i') \end{aligned} \tag{6.52}$$

如先前一样，其中 α_i 和 α_i'是拉格朗日乘子。在式(6.52)中引入新的乘子 γ_i 和 γ_i'，用来保证对于乘子 α_i 和 α_i'假设变量的形式的最优性约束。最小化式(6.52)关于回归模型中参数 $\mathbf{w}$ 和 b 的拉格朗日函数的要求，正如对松弛变量 ξ 和 ξ'一样。

如先前优化过程一样，对参数求导并且令其为 0，分别获得如下等式：

$$\hat{\mathbf{w}} = \sum_{i=1}^{N}(\alpha_i - \alpha_i')\mathbf{x} \tag{6.53}$$

$$\sum_{i=1}^{N}(\alpha_i - \alpha_i') = 0 \tag{6.54}$$

$$\alpha_i - \gamma_i = C, \qquad i = 1,2,\cdots,N \tag{6.55}$$

$$\alpha_i' - \gamma_i' = C, \qquad i = 1,2,\cdots,N \tag{6.56}$$

对于计算出来的乘子 α_i 和 α_i'而言，式(6.53)中的支持向量展开定义了期望的参数估计$\hat{\mathbf{w}}$。为了找到相应的偏置（用$\hat{b}$表示），我们采用 Karush-Kuhn-Tuner 条件。从第 6.2 节的讨论中，可以推断出为了满足这些条件，对于所有不满足作为等式的约束，相应的对偶变量必须变为 0。对于目前的问题，我们有两组约束：

- 第一组如式(6.48)和式(6.49)不等式所描述，分别对应于对偶变量 α_i 和 α_i'。
- 第二组如式(6.50)和式(6.51)不等式所描述，分别对应于对偶变量 γ_i 和 γ_i'，从式(6.55)和式(6.56)，可以发现 $\gamma_i = \alpha_i - C$ 和 $\gamma_i' = \alpha_i - C$。

相应地，我们根据对应的对偶变量使用 Karush-Kuhn-Tuner 条件到这四个约束条件，分别得到

$$\alpha_i(\varepsilon+\xi_i+d_i-y_i)=0 \tag{6.57}$$

$$\alpha_i'(\varepsilon+\xi_i'-d_i+y_i)=0 \tag{6.58}$$

$$(\alpha_i-C)\xi_i=0 \tag{6.59}$$

$$(\alpha_i'-C)\xi_i'=0 \tag{6.60}$$

通过查看上述条件，我们得出三点重要结论：

1. 式(6.59)和式(6.60)说明当 $\alpha_i=0$ 和 $\alpha_i'=C$ 的样本 (x_i,d_i) 位于 $\xi_i>0$ 和 $\xi_i'>0$ 时；这些松弛变量相对应的点在 ε-不敏感区域之外，该区域中心就是回归函数 $f(\mathbf{x})=\mathbf{w}^T\mathbf{x}+b$（如图 6.10a 所示）。

2. 将式(6.57)乘上 α_i' 和式(6.58)乘上 α_i，然后相加相应的结果，得到

$$\alpha_i\alpha_i'(2\varepsilon+\xi_i+\xi_i')=0$$

所以，当任意 $\varepsilon>0$，以及 $\xi_i>0$ 和 $\xi_i'>0$ 时，我们有如下条件

$$\alpha_i\alpha_i'=0$$

从中可以看到两个乘子 α_i 和 α_i' 不可能同时为非零。

3. 从式(6.59)和式(6.60)，我们分别观察到

$$\xi_i=0,\quad \text{当 } 0<\alpha_i<C$$

$$\xi_i'=0,\quad \text{当 } 0<\alpha_i'<C$$

在这种情况下，由式(6.57)和式(6.58)可以看到

$$\varepsilon-d_i+y_i=0,\qquad \text{当 } 0<\alpha_i<C \tag{6.61}$$

$$\varepsilon+d_i-y_i=0,\qquad \text{当 } 0<\alpha_i'<C \tag{6.62}$$

通过式(6.61)和式(6.62)，我们可以计算偏置的估计 $\hat{b}$。首先，我们重新改写回归函数优化输出如下：

$$y=\hat{\mathbf{w}}^T\mathbf{x}+\hat{b}$$

对于输入向量 x_i，有

$$y=\hat{\mathbf{w}}^T\mathbf{x}_i+\hat{b} \tag{6.63}$$

把式(6.63)代入式(6.61)和式(6.62)中得到：

$$\hat{b}=d_i-\hat{\mathbf{w}}^T\mathbf{x}_i-\varepsilon,\quad \text{当 } 0<\alpha_i<C \tag{6.64}$$

和

$$\hat{b}=d_i-\hat{\mathbf{w}}^T\mathbf{x}_i+\varepsilon,\quad \text{当 } 0<\alpha_i'<C \tag{6.65}$$

所以，给定 ε 和 d_i 并且从式(6.53)中知道 $\hat{\mathbf{w}}$，我们可以计算出偏置的估计 $\hat{b}$。

对于 $\hat{b}$ 的计算，理论上可以使用任何属于 $(0,C)$ 之间的乘子。但是，在实际计算中，用所有属于这个域的乘子计算出的平均值更好。

支持向量展开的稀疏性

从式(6.57)和式(6.58)，我们可以看到所有在 ε-不敏感的区域里面，有

$$|d_i-y_i|\geqslant\varepsilon$$

在这种情况下，两个式子括号中的因子都是非零的，因此，为了使式(6.57)和式(6.58)都成立(即满足 KKT 条件)，我们没有必要使用所有的 $\mathbf{x}_i$ 来计算 $\hat{\mathbf{w}}$。换句话说，式(6.53)的支持向量展开是*稀疏的*。

拉格朗日乘子 α_i 和 α_i' 非零的样本定义了支持向量。就式(6.53)而言，在 ε-不敏感的区域里面的点对最后的解没有贡献，这点在几何上似乎是合理的。这些特殊的点不包含对最后的解有用的信息（Schölkopf and Smola，2002）。

6.10　表示定理和相关问题

我们通过建立表示定理来完成核机器（包括支持向量机）的讨论，不管这些核机器是线性的还是非线性的。表示定理为我们更好理解这类重要的学习机器提供了很多帮助。为了证明这个定理，我们首先来描述什么是 Hilbert 空间，然后介绍什么是再生核 Hilbert 空间。

Hilbert 空间[10]

令$\{\mathbf{x}_k\}_{k=1}^{\infty}$是内积空间$\mathscr{F}$中的一组标准正交基，同样假定其是无限维的。注意，两个向量 $\mathbf{x}_j$ 和 $\mathbf{x}_k$ 满足标准正交性是指满足如下双重条件：

$$\mathbf{x}_j^T\mathbf{x}_k = \begin{cases} 1, & \text{当 } j = k \\ 0, & \text{其他} \end{cases} \tag{6.66}$$

第一部分是关于规范性，第二部分是关于正交性。这样的空间称为 pre-Hilbert 空间。赋范空间，每个向量都有有限的欧几里得范数（长度），是 pre-Hilbert 空间的特例。

令$\mathscr{H}$为最大且最广泛的向量空间，并具有无限基$\{\mathbf{x}_k\}_{k=1}^{\infty}$。在空间$\mathscr{F}$中的向量不一定具有如下形式：

$$\mathbf{x} = \sum_{k=1}^{\infty} a_k\mathbf{x}_k \tag{6.67}$$

被称为由$\{x_k\}_{k=1}^{\infty}$所张成的，a_k 是系数。定义新的向量

$$\mathbf{y}_n = \sum_{k=1}^{n} a_k\mathbf{x}_k \tag{6.68}$$

可以以相似的方式定义另一个向量 $\mathbf{y}_m$。当 $n>m$ 时，我们计算两者之间欧几里得距离的平方

$$\|\mathbf{y}_n - \mathbf{y}_m\|^2 = \|\sum_{k=1}^{n} a_k\mathbf{x}_k - \sum_{k=1}^{m} a_k\mathbf{x}_k\|^2 = \|\sum_{k=m+1}^{n} a_k\mathbf{x}_k\|^2 = \sum_{k=m+1}^{n} a_k^2 \tag{6.69}$$

其中，在最后一行我们调用了式(6.66)的双重条件。

鉴于式(6.69)，可以推导出以下公式：

$$1.\ \sum_{k=m+1}^{n} a_k^2 \to 0, \text{当 } n,m \to \infty \text{ 时}$$

$$2.\ \sum_{k=1}^{m} a_k^2 < \infty$$

另外，对于给定的正数 ε，我们可以找到一个足够大的整数 m 来满足

$$\sum_{m+1}^{\infty} a_k^2 < \varepsilon$$

因为

$$\sum_{k=1}^{\infty} a_k^2 = \sum_{k=1}^{m} a_k^2 + \sum_{k=m+1}^{\infty} a_k^2$$

因此

$$\sum_{k=1}^{\infty} a_k^2 < \infty \tag{6.70}$$

在赋范空间中，当 $\mathbf{y}_m$ 和 $\mathbf{y}_n$ 之间的距离满足

$$\|\mathbf{y}_n - \mathbf{y}_m\| < \varepsilon, \text{对任意 } \varepsilon > 0 \text{ 且所有的 } m,n > M,$$

时一列向量$\{\mathbf{y}_k\}_{k=1}^{n}$就是一个收敛序列；这样的序列被称为 Cauchy 序列。注意到所有的收敛序列都是 Cauchy 序列，但不是所有 Cauchy 序列都收敛。

所以，向量 $\mathbf{x}$ 可以由基$\{\mathbf{x}_k\}_{k=1}^{\infty}$所张成，当且仅当 $\mathbf{x}$ 是这组基的线性组合而且其长度的平方

是系数$\{a_k\}_{k=1}^{\infty}$的平方和。相反地，系数$\{a_k\}_{k=1}^{\infty}$的平方和说明了当 n 和 m 都接近无穷的时候 $\|\mathbf{y}_n-\mathbf{y}_m\|^2$趋向 0，也说明了收敛序列$\{\mathbf{y}_n\}_{n=1}^{\infty}$是一个 Cauchy 序列。

根据以上讨论，显然空间$\mathcal{H}$比内积空间$\mathcal{F}$更“完备”，我们可做如下重要总结：

一个内积空间$\mathcal{H}$是完备的，如果该空间$\mathcal{H}$中的所有 Cauchy 序列收敛到空间$\mathcal{H}$中的一个极限；一个完备的内积空间被称为 Hilbert 空间。

事实上，就上述总结而言，内积空间$\mathcal{F}$通常被称为 pre-Hilbert 空间。

再生核 Hilbert 空间[11]

考虑一个 Mercer 核 $k(\mathbf{x},\cdot)$，其中向量 $x\in\mathcal{H}$，$\mathcal{F}$是关于 $\mathbf{x}$ 所有实值函数的向量空间，这些函数是由核 $k(\mathbf{x},\cdot)$所产生的。假定 $f(\cdot)$和 $g(\cdot)$是由空间$\mathcal{F}$中抽取出的两个函数，分别表示为

$$f(\cdot)=\sum_{i=1}^{l}a_i k(\mathbf{x}_i,\cdot) \tag{6.71}$$

和

$$g(\cdot)=\sum_{j=1}^{n}b_j k(\tilde{\mathbf{x}}_j,\cdot) \tag{6.72}$$

其中 a_i 和 b_i 是对于 $\mathbf{x}_i$ 和 $\tilde{\mathbf{x}}_j\in\mathcal{X}$的展开系数，对于所有的 i 和 j。

给定函数 $f(\cdot)$和 $g(\cdot)$，我们引入双线性形式

$$\begin{aligned}\langle f,g\rangle&=\sum_{i=1}^{l}\sum_{j=1}^{n}a_i k(\mathbf{x}_i,\tilde{\mathbf{x}}_j)b_j\\&=\mathbf{a}^T\mathbf{K}\mathbf{b}\end{aligned} \tag{6.73}$$

其中 $\mathbf{K}$ 是一个 Gram 矩阵，或者核矩阵，在式子的第一行使用关系

$$k(\mathbf{x}_i,\cdot)k(\mathbf{x}_j,\cdot)=k(\mathbf{x}_i,\mathbf{x}_j) \tag{6.74}$$

然后式(6.73)可以重写为简单形式

$$\langle f,g\rangle=\sum_{i=1}^{l}a_i\sum_{j=1}^{n}b_j k(\mathbf{x}_i,\tilde{\mathbf{x}}_j)=\sum_{i=1}^{l}a_i\underbrace{\sum_{j=1}^{n}b_j k(\tilde{\mathbf{x}}_j,\mathbf{x}_i)}_{g(\mathbf{x}_i)}=\sum_{i=1}^{l}a_i g(\mathbf{x}_i) \tag{6.75}$$

其中，第二行，使用 Mercer 核的对称性。简化成：

$$\langle f,g\rangle=\sum_{j=1}^{n}b_j f(\tilde{\mathbf{x}}_j) \tag{6.76}$$

式(6.73)的双线性的定义是独立于函数 $f(\cdot)$和 $g(\cdot)$的表示。我们这样说是因为式(6.75)中的和式 $\sum_{i=1}^{l}a_i g(\mathbf{x}_i)$ 不随下标数 n、系数向量 $\mathbf{b}$ 和 n 维向量 $\tilde{x}_j$ 的改变而改变。同样，式(6.76)中的和式 $\sum_{j=1}^{n}b_j f(\tilde{\mathbf{x}}_j)$ 也具有这样的性质。

另外，从式(6.73)中，可以推导出下列三个性质：

性质 1　对称性　对于空间$\mathcal{F}$中的所有函数 f 和 g 来说 $\langle f,g\rangle$ 是对称的，即

$$\langle f,g\rangle=\langle g,f\rangle \tag{6.77}$$

性质 2　缩放性和可分配性　常数对 c 与 d 与和空间$\mathcal{F}$中任何函数 f，g 和 h 的任意集合有

$$\langle(cf+dg),h\rangle=c\langle f,h\rangle+d\langle g,h\rangle \tag{6.78}$$

性质 3　范数平方　对空间$\mathcal{F}$中的任何实值函数 f，我们把式(6.73)改成 f 对自己作用，有如下的平方范数或者二次度量：

$$\| f \|^2 = \langle f, f \rangle = \mathbf{a}^T \mathbf{K} \mathbf{a}$$

因为 Gram 矩阵具有非负性，所以范数平方

$$\| f \|^2 \geqslant 0 \tag{6.79}$$

借助这样的事实，即对空间$\mathcal{F}$中的任何实值函数 f 和 g，双线性项 $\langle f, g \rangle$ 满足对称性与缩放性和可分配性，并且范数 $\| f \|^2 = \langle f, f \rangle$ 满足非负性，我们可以正式地提出式(6.73)中的 $\langle f, g \rangle$ 实际上是一个内积；而且这个内积满足条件当且仅当 $f=0$ 时 $\langle f, g \rangle = 0$。换句话说，包括函数 f 和 g 的空间$\mathcal{F}$是一个内积空间。

由式(6.75)可以直接得到附加的一个性质。具体地，令

$$g(\cdot) = k(\mathbf{x}, \cdot)$$

有

$$\langle f, k(\mathbf{x}, \cdot) \rangle = \sum_{i=1}^{l} a_i k(\mathbf{x}, \mathbf{x}_i) = \sum_{i=1}^{l} a_i k(\mathbf{x}_i, \mathbf{x}) = f(\mathbf{x}), \quad k(\mathbf{x}, \mathbf{x}_i) = k(\mathbf{x}_i, \mathbf{x}) \tag{6.80}$$

显然，Mercer 核 $k(\mathbf{x}, \cdot)$ 的这个特征被称为*再生性*。

表示两个向量 $\mathbf{x}, \mathbf{x}_i \in \mathcal{X}$ 的函数的核 $k(\mathbf{x}, \mathbf{x}_i)$ 被称为向量空间$\mathcal{F}$的再生核，当满足以下两个条件（Aronszajn，1950）时：

1. 对于任何 $\mathbf{x}_i \in \mathcal{X}$，有关向量 x 的函数 $k(\mathbf{x}, \mathbf{x}_i)$属于$\mathcal{F}$。
2. 它满足再生性。

而 Mercer 核确实满足以上两个条件，因此赋予了其“再生核”的名称。如果在其中定义了再生核空间的内积（向量）空间$\mathcal{F}$也是*完备的*，我们就可以更进一步讨论一种“再生核 Hilbert 空间”。

为了证明完备性，考虑一个固定输入向量 x 和一对 Cauchy 序列 $\{f_n(\mathbf{x})\}_{n=1}^{\infty}$ 和 $\{f_m(\mathbf{x})\}_{m=1}^{\infty}$，其中 $n > m$，然后对式 $f_n(\mathbf{x})$ 和 $f_m(\mathbf{x})$ 应用式(6.80)的再生核性质，可以写出

$$f_n(\mathbf{x}) - f_m(\mathbf{x}) = \langle f_n(\cdot) - f_m(\cdot) \rangle k(\mathbf{x}, \cdot)$$

其中右边是一个内积。然后使用 Cauchy-Schwarz 不等式[12]，我们有

$$(f_n(\mathbf{x}) - f_m(\mathbf{x}))^2 \leqslant \langle f_n(\cdot) - f_m(\cdot) \rangle^2 \underbrace{k(\mathbf{x}, \cdot) k(\mathbf{x}, \cdot)}_{k(\mathbf{x}, \mathbf{x})} \tag{6.81}$$

因此，$f_n(\mathbf{x})$是有界的 Cauchy 序列，收敛到空间$\mathcal{F}$中的某个实值函数 f。最后，定义函数

$$y(\mathbf{x}) = \lim_{n \to \infty} f_n(\mathbf{x})$$

通过这样的收敛 Cauchy 序列来完备空间$\mathcal{F}$，就获得了 Hilbert 空间$\mathcal{H}$。我们已经说明了每个 Mercer 核 $k(\mathbf{x}, \cdot)$定义了一个 Hilbert 空间$\mathcal{H}$，其中通过 $f(\mathbf{x})$和 $k(\mathbf{x}, \cdot)$的内积再生了函数 $f(\mathbf{x})$的值。这样定义的 Hilbert 空间被称为*再生核* Hilbert 空间，以后我们使用首字母缩写 RKHS。

在下面我们用一个重要定理来说明 RKHS 强大的分析能力。

表示定理的规范表述[13]

我们由 Mercer 核 $k(\mathbf{x}, \cdot)$导出一个 RKHS，记为$\mathcal{H}$。给定任意实值函数 $f(\mathbf{x}) \in \mathcal{H}$，可将其分解为两部分的和，这两部分都自然地位于空间$\mathcal{H}$中：

- 第一部分是核函数 $k(\mathbf{x}_1, \cdot), k(\mathbf{x}_2, \cdot), \cdots, k(\mathbf{x}_l, \cdot)$ 的展开形式；用 $f_{\parallel}(\mathbf{x})$表示这个部分，使用式(6.71)来表示该部分

$$f_{\parallel}(\cdot) = \sum_{i=1}^{l} a_i k(\mathbf{x}_i, \cdot)$$

- 第二部分是正交于核函数的；用 $f_{\perp}(\mathbf{x})$表示这个部分。

因此可以表示函数 $f(\cdot)$为

$$f(\cdot)=f_{\parallel}(\cdot)+f_{\perp}(\cdot)=\sum_{i=1}^{l}a_i k(\mathbf{x}_i,\cdot)+f_{\perp}(\cdot) \tag{6.82}$$

对式(6.82)使用式(6.78)的可分配特征，我们有

$$\begin{aligned}f(\mathbf{x}_j)&=\langle f(\cdot),k(\mathbf{x}_j,\cdot)\rangle_{\mathscr{H}}\\&=\Big\langle\sum_{i=1}^{l}a_i k(\mathbf{x}_i,\cdot),k(\mathbf{x}_j,\cdot)\Big\rangle_{\mathscr{H}}\langle k(\mathbf{x}_j,\cdot),f_{\perp}\rangle_{\mathscr{H}}\end{aligned}$$

由于 $f_{\perp}(\mathbf{x})$垂直于核函数的张量，所以第二项为 0，等式因此变为

$$f(\mathbf{x}_j)=\Big\langle\sum_{i=1}^{l}a_i k(\mathbf{x}_i,\cdot),k(\mathbf{x}_j,\cdot)\Big\rangle_{\mathscr{H}}=\sum_{i=1}^{l}a_i k(\mathbf{x}_i,\mathbf{x}_j) \tag{6.83}$$

等式(6.83)是表示定理的数学表达：

任何 RKHS 中定义的函数可以由一系列 Mercer 核函数的线性组合来表示。

然而，还需要介绍更多内容。

表示定理的泛化能力

表示定理的重要特点是：式(6.83)给定的展开式使如下的正则风险（价值函数）最小

$$\mathscr{C}(f)=\frac{1}{2N}\sum_{n=1}^{N}(d(n)-f(\mathbf{x}(n)^2+\Omega(\|f\|_{\mathscr{H}}) \tag{6.84}$$

其中 $\{\mathbf{x}(n),d(n)\}_{n=1}^{N}$ 是训练样本，f 是未知函数，$\Omega(\|f\|_{\mathscr{H}})$ 是正则函数 (Scholkpf and Smola，2002)。要使定理成立，正则函数必须是参数的单调增函数；这个条件简称为单调性条件。

式(6.84)右边的第一项是标准误差，是 f 的二次函数。所以，通过使用固定的 $a_i\in\mathbb{R}$，式(6.83)的展开形式使这项最小。

为了证明展开式也使风险函数$\mathscr{C}(f)$的正则部分达到最小，我们分以下三步处理：

1. 让 $f_{\perp}$代表与核函数 $\{k(\mathbf{x}_i,\cdot)\}_{i=1}^{l}$ 的张量正交的部分。所以，根据式(6.82)，每个函数可以用训练样本上的核展开并合并 $f_{\perp}$来表示，有

$$\Omega(\|f\|_{\mathscr{H}})=\Omega\Big(\Big\|\sum_{i=1}^{l}a_i k(\mathbf{x}_i,\cdot)+f_{\perp}(\cdot)\Big\|_{\mathscr{H}}\Big) \tag{6.85}$$

为了数学上的方便，我们使用新的函数

$$\widetilde{\Omega}(\|f\|_{\mathscr{H}}^2)=\Omega(\|f\|_{\mathscr{H}}) \tag{6.86}$$

而不是使用原正则函数 $\Omega(\|f\|_{\mathscr{H}})$。这是允许的，因为二次函数在 $[0,\infty)$ 区间上是严格单调的。所以，当且仅当 $\widetilde{\Omega}(\|f\|_{\mathscr{H}}^2)$ 满足单调性条件时 $\Omega(\|f\|_{\mathscr{H}})$ 在 $[0,\infty)$ 上是严格单调的。对于所有的 $f_{\perp}$，我们可以写成

$$\widetilde{\Omega}(\|f\|_{\mathscr{H}}^2)=\widetilde{\Omega}\Big(\Big\|\sum_{i=1}^{l}a_i k(\mathbf{x}_i,\cdot)+f_{\perp}(\cdot)\Big\|_{\mathscr{H}}^2\Big) \tag{6.87}$$

2. 对式(6.87)右边的参数 $\widetilde{\Omega}$ 使用 Pythagorean 分解，可以写成

$$\widetilde{\Omega}(\|f\|_{\mathscr{H}}^2)=\widetilde{\Omega}\Big(\Big\|\sum_{i=1}^{l}a_i k(\mathbf{x}_i,\cdot)\Big\|_{\mathscr{H}}^2+\|f_{\perp}\|_{\mathscr{H}}^2\Big)\geqslant\widetilde{\Omega}\Big(\Big\|\sum_{i=1}^{l}a_i k(\mathbf{x}_i,\cdot)\Big\|_{\mathscr{H}}^2\Big)$$

对优化条件，令 $f_{\perp}=0$ 生成以下等式

$$\widetilde{\Omega}(\|f\|_{\mathscr{H}}^2)=\widetilde{\Omega}\Big(\Big\|\sum_{i=1}^{l}a_i k(\mathbf{x}_i,\cdot)\Big\|_{\mathscr{H}}^2\Big) \tag{6.88}$$

3. 最后，就式(6.86)所引入的定义，我们得到期望的结果

$$\Omega(\|f\|_{\mathscr{H}})=\Omega\Big(\Big\|\sum_{i=1}^{l}a_i k(\mathbf{x}_i,\cdot)\Big\|_{\mathscr{H}}\Big) \tag{6.89}$$

于是以下事实成立：只要单调性条件满足，对于固定的 $a_i \in \mathbb{R}$，表示定理使正则函数 $\Omega(\|f\|_{\mathcal{H}})$ 最小。

在分解标准误差和正则项时把它们作为一个整体，它们两项之间会有个均衡。在任何情况下，对于某些固定的 $a_i \in \mathbb{R}$，式(6.83)所描述的表示定理将会使式(6.84)的正则函数达到最小，从而确定了表示定理良好的泛化能力（Scholkpf and Smola，2002）。

在第7章正则化理论中我们将使用这个重要的定理。

6.11 小结和讨论

支持向量机是为了设计仅有一个非线性单元隐藏层的前馈网络而设计的简洁而高度原则性强的学习方法。它由植根于VC维理论的结构风险最小化原则导出，这一点使得它的推导更加深刻，结构风险最小化在第4章讨论过。顾名思义，机器的设计随抽取训练数据的子集作为支持向量而定，因而代表数据的一个稳定特征。支持向量机包括多项式学习机器、径向基函数网络和两层感知器作为其特殊情形。因此，虽然这些方法提供训练数据的内在统计规则的不同的表示，但是它们都源于支持向量机这一共同基础。

支持向量机的另一个突出的特点就是批量学习的核方法[14]。

计算考虑

支持向量机的行为随着训练样本的数目增加而渐进地线性增长。存在这样的事实，用来解决模式识别和回归问题的计算代价都包括一个二次项和三次项。具体地，当 C 很小时，计算代价以 N^2 增加，当 C 很大时，计算代价以 N^3 增长（Bottou and Lin，2007）。

为了缓解这个问题，许多商业优化库被用于解决二次规划问题。但是这些库的用处比较有限。解决二次规划问题的内存需求也是随着样本数目二次增长。在现实生活中的应用通常包括上千个点，因此二次规划问题的解不能通过直接商业优化库来获得。即：通常，SVM问题的解很稀疏，这导致问题更加复杂，因为机器输出层的权重向量只包括相对于训练样本数目来说极少的非零元素。相应地，直接用来解支持向量机中二次规划问题的尝试对于大型问题来说行不通。为了克服这个困难，学术界提出了好几种新方法，总结如下[15]：

1. Osuma等（1997）发明了一种新的分解算法通过解决一系列更小子问题来得到优化。特别地，分解算法利用支持向量系数的特点，即在 $a_i = 0$ 或 $a_i = C$ 定义的两个边界上是活跃的。据称分解算法能解决大约100 000个数据点的问题，表现令人满意。

2. Platt(1999) 扩展了Osuna的方法，引入了一个称为*序列最小优化的算法*（SMO），将大的二次规划问题分解成一系列很小的二次规划子问题，从而不用二次规划库。SMO的计算时间主要由核计算决定，所以使用核优化能加快速度。

3. Joachims(1999) 提出几种他自己的新方法。具体地，一个大的SVM问题分解成一系列小的问题，相比于Osuna方法原则性更强。另外一种重要的新方法就是收缩的观点：如果一个点在一段时间内不是无界的支持向量，之后它以极高概率不会变成支持向量，这个点以后不用考虑，从而节省计算时间。

4. Rifkin(2002)发明了一种新的计算过程称为SVMFu算法，可以认为是上述三种方法的结合。具体地，利用三种算法的优点结合其他的特点。据称本方法可以通过解一系列小的子问题来解决大规模问题，而这些子问题足够小，它们相应的Hessian矩阵能放入内存。

5. Drineas and Mahoney(2005) 提出一种算法，算法计算一个对 $N \times N$ 阶Gram矩阵容易判断的低阶近似。通过这种方式使计算的速度更快。新算法和Nystrom算法的关系可以从积分方程理论谈起。

6. Hush等（2006）年提出多项式时间来求支持向量机问题中出现的一类二次规划问题的

近似解并能保证精度。算法分两步：第一步先产生对偶二次规划问题的近似解；第二步将这个对偶问题的解映射到原问题的解。

维数灾难

在多层网络中，支持向量机固有的复杂度作为一个逼近函数是随着 m_0 指数增长的，其中 m_0 是输入空间的维数。另外，复杂度随着 s 而降低，其中 s 是平滑指数，它是对逼近函数约束的数目的测量。从而，逼近函数的平滑指数是制止维数灾难的矫正措施。我们认为支持向量机为高维函数提供了一个很好的近似，只要相应的函数是平滑的。

结论

支持向量机是最为广泛使用的核学习算法。事实上，我们可以说在机器学习领域，支持向量机由于其优良的泛化能力，易于使用和严密的理论基础等优点代表了最新的算法。还有，在实际应用的环境下，存在对解决模式分类问题和回归问题的鲁棒性。

然而，支持向量机的主要缺陷是，随着训练样本的数目增加，计算和存储要求也快速增加。这些严重的要求使得处理大规模问题时超越了支持向量机的能力。实际的主要缺陷是二次规划问题，而它是 SVM 优化理论中的一部分。为了缓解问题的难度，许多方法的提出加快了 SVM 解的速度，例如许多上述提到的并行实现的技术和分解计算算法（Durdanovic et al.，2007；Yom-Tov，2007）。

注释和参考文献

1. Vapnik 首先提出支持向量机；Boser，Guyon and Vapnik 于 1992 年给出它的第一个描述。而关于它的最综合且详细的描述是出现于 Vapnik 在 1998 创作的题为“Statistical Learning Theory”一书中，该书已经成为该领域的一个经典。

   Cucker and Smale(2001) 的标题为“On the Mathematical Foundations of Learning”的文章中为监督学习理论提供了严格的数学处理技术，重点放在近似学习和归纳推理的重要性上。

   Schölkopf and Smola(2002)，Herbrich(2002)，and Shawe-Taylor and Cristianini(2004) 的书中都有对核机器包括支持向量的综合论述。
2. 凸优化是一种特殊的优化技术，包括最小二乘法问题和线性规划问题，理论基础已经完善。而且可以转化到凸优化的问题已经不止是最小二乘法问题和线性规划问题。将问题转化到凸优化问题可以获得如下优点：
   - 解是可靠且有效的。
   - 理论优点，以形成对偶问题为例，相比于原问题转化的解，计算上更加有效且概念上更清晰。

   有关凸分析和优化的详细的论述，可以查看 Byod and Vandenbergh(2004)and Bertsekas et al.(2003) 的书。
3. 对偶性适用可导目标函数且带约束的任何优化问题，原问题和对偶问题都要满足 Karush-Kuhn-Tucker (KKT) 条件，这个条件以 Karush(1939) 和 Kuhn 与 Tucker(1951) 的名字命名的。Kuhn(1976) 的文章给出解决不等式约束问题的历史性的报告，其中凸优化起到主要作用。
4. Girosi(1998)and Vapnik(1998) 首先讨论了稀疏近似和支持向量展开的关系。

   Steinwart(2003) 对于在通过支持向量机解决模式识别问题中出现的稀疏性给出了详细的讨论；特别地，这篇文章给出了支持向量个数的下限。沿着这条思路许多对于理解支持向量极重要的结果得到证明。这篇文章给出三个允许的损失函数：

   ⅰ. 铰链损失函数 $L(d,y)=\max(0,1-dy)$；

   ⅱ. 二次铰链损失函数 $L(d,y)=[\max(0,1-dy)]^2$；

   ⅲ. 最小二乘损失函数 $L(d,y)=(1-dy)^2$。

   相应的 SVM 分别表示为 L_1，L_2 和 LS。变量 d 和 y 分别表示相应的期望输出和给定输入相应计算出的响应。

   通过使用最小二乘误差的支持向量机在 Suykens 的书“Least-Squares Support Vector Machines”中有详尽的阐述。
5. 为了研究计算复杂度，我们鉴别两种类别的算法：

- *多项式时间算法*，需要问题规模的多项式时间来计算。例如，快速 Fourier 变换（FFT），用来做谱分析，就是一个多项式时间算法，运算时间是 nlogn，其中 n 是问题规模。
- *指数时间算法*，需要问题规模的指数时间来计算。例如，一个指数算法需要 2^n 的时间来计算，其中 n 是问题规模。

基本上，我们认为多项式时间算法和指数时间算法都是有效的算法。

在实际生活中很多问题没有有效的算法。其中的许多问题，但不是所有，似乎是不可解的，通常被归为一类称为 NP-完全（NP-complete）问题。术语 NP 是“nondeterministic polynomial”的缩写。

对于 NP-完全的讨论，请参考 Cook(1971)，Garey and Johnson(1979)和 Cormen et al.(1990)。

6. 在最小二乘法问题中，C 的倒数起到了正则参数的作用。我们在描述支持向量机中使用 C 基本上是为了和这种核机器学习早期的发展一致。
7. Aizerman 等（1964a，1964b）在设计该方法的潜在功能时首先提出关于内积核的思想，代表了径向基函数的先驱。同时，Vapnik and Chervonenkis(1964) 发展了最优超平面的思想。将两种强大的思想结合起来形成支持向量机首先出现于 Boser 等（1992）。
8. 除 6.4 节讨论的性质 1 和性质 2 之外有关核性质的讨论，可以参见 Schölkopf and Smola(2002)，Herbirch (2002)和 Cristianini(2004)。
9. 要描述最小最大化定理，考虑函数 $f(x,z)$，其中 $x\in\mathcal{X}$，$z\in\mathcal{Z}$，定理要求

$$\min_{z\in\mathcal{Z}} \sup f(x,z)$$
$$\text{s. t. } x\in\mathcal{X}$$

或者相应地

$$\max_{x\in\mathcal{X}} \inf f(x,z)$$
$$\text{s. t. } z\in\mathcal{Z}$$

例如在最糟糕的设计情况下，应用最小最大化定理具有很重要的工程应用。有关该定理的讨论，参见 Bertsekas 等（2003）。

Huber 最小最大化定理是基于邻域的，而不是全局的，由于它们排除非对称分布。然而这个定理成功地处理了传统统计中的许多问题，特别是回归问题。

10. 有关 Hilbert 空间的讨论参见 Dorny(1975) 和 Debnath and Mikusinski(1990)。
11. 再生核 Hilbert 空间（RKHS）首先出现在 Aronszajn(1950) 中，该文章是一个经典。同样可以参见 Shawe-Taylor and Cristianini(2004)、Schölkopf and Smola(2002) 和 Herbirch(2002)。
12. 令 x 和 y 是内积空间$\mathcal{H}$中任意两个，根据 Cauchy-Schwarz 不等式，我们有

$$\langle \mathbf{x},\mathbf{y}\rangle \leqslant \|\mathbf{x}\|^2 \cdot \|\mathbf{y}\|^2$$

证明是简单的。不等式说明了内积的平方不大于两个向量长度平方的乘积。而式(6.81)中的不等式是为了更方便在再生核 Hilbert 空间考虑问题。

13. 就历史背景而言，Kimeldorf and Wahba(1971) 为了解决基于最小二乘函数的实际统计估计问题而描述了著名的表示定理，同样也可以参考 Wahba(1990)。而该定理关于正则风险函数的泛化能力首先是由 Schölkopf and Smola(2002) 解决的。
14. 相对于支持向量机这种批量学习，核 LMS 算法（Liu 等，2008）则是一种在线学习算法。这种新的算法思想的来源包括第 3 章讨论的最小二乘算法和本章讨论的再生 Hilbert 空间，并把这些思想复合地集成到一起。特别地，核技巧被用于允许基于迭代的学习。
15. 关于二次规划优化的综述参见 Bottou and Lin(2007)。

习题

最优分离超平面

6.1 考虑用于线性可分模式的超平面，它由如下方程定义

$$\mathbf{w}^T\mathbf{x}+b=0$$

其中 $\mathbf{w}$ 表示权值向量，b 为偏置，$\mathbf{x}$ 为输入向量。如果输入模式集 $\{\mathbf{x}_i\}_{i=1}^N$ 满足附加的条件

$$\min_{i=1,2,\cdots,N} |\mathbf{w}^T\mathbf{x}_i+b| = 1$$

则称超平面对应于标准对 (w,b)。证明标准对的这个要求导致两类分离边界之间的距离为 $2/\|\mathbf{w}\|$。

6.2 在不可分类模式的背景下判断下列陈述：错分类意味着模式的不可分性，但相反则未必真。

6.3 以不可分模式的分离超平面的最优化作为原问题的开始，构造如 6.3 节描述的对偶问题的公式。

6.4 在本题中，利用在第 4 章讨论的“留一法”估计不可分模式的最优超平面产生的期望测试误差。通过删除训练样本中任意一个模式并且根据剩下的模式构造一个解，讨论使用这种方法可以引发的各种可能性。

6.5 数据空间中最优超平面的位置由被选为支持向量的数据点决定。如果数据有噪声，人们的第一反应也许是质疑分离边界对噪声的鲁棒性。但对最优超平面的详细研究揭示分离边界对噪声实际上是鲁棒的。讨论这种鲁棒性的根据。

6.6 内积核 $k(x_i, \mathbf{x}_j)$ 是在大小为 N 的训练样本集$\mathcal{T}$上计算的，它产生 $N\times N$ 矩阵

$$\mathbf{K} = \{k_{ij}\}_{i,j=1}^{N}$$

其中 $k_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$。由于它的所有元素的值为正，因此矩阵 $\mathbf{K}$ 是正的。利用相似变换

$$\mathbf{K} = \mathbf{Q}\Lambda\mathbf{Q}^T$$

其中 Λ 为特征对角矩阵，而 $\mathbf{Q}$ 为相应特征向量构成的矩阵，通过 $\mathbf{K}$ 的特征值和特征向量构造内积核 $k(\mathbf{x}_i, \mathbf{x}_j)$的表达式。你可以从这个表达式得出什么结论？

6.7 (a) 证明表 6.1 中的三种 Mercer 核满足酉不变性，即

$$k(\mathbf{x}, \mathbf{x}_i) = k(\mathbf{Q}\mathbf{x}, \mathbf{Q}\mathbf{x}_i)$$

其中 $\mathbf{Q}$ 为酉矩阵，定义为

$$\mathbf{Q}^{-1} = \mathbf{Q}^T$$

(b) 一般说来，这个性质是不是都成立？

6.8 (a) 说明 Mercer 核的正定性。

(b) 考虑 Mercer 核 $k(\mathbf{x}_i, \mathbf{x}_j)$。这样的核满足 Cauchy-Schwarz 不等式。

$$k(\mathbf{x}_i, \mathbf{x}_j)k(\mathbf{x}_j, \mathbf{x}_i) \leqslant k(\mathbf{x}_i, \mathbf{x}_i)k(\mathbf{x}_j, \mathbf{x}_j)$$

通过考虑 2×2 的 Gram 矩阵 K 的决定因子来证明 Mercer 核的这个特点。

6.9 考虑高斯核

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right), \quad i,j = 1,2,\cdots,N$$

其中 $\mathbf{x}_i$ 和 $\mathbf{x}_j$ 没有相同的。说明 Gram 矩阵：

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & k(\mathbf{x}_1, \mathbf{x}_2) & \cdots & k(\mathbf{x}_N, \mathbf{x}_1) \\ k(\mathbf{x}_2, \mathbf{x}_1) & k(\mathbf{x}_2, \mathbf{x}_2) & \cdots & k(\mathbf{x}_N, \mathbf{x}_2) \\ \vdots & \vdots & & \vdots \\ k(\mathbf{x}_N, \mathbf{x}_1) & k(\mathbf{x}_N, \mathbf{x}_2) & \cdots & k(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix}$$

具有满秩-从代数的角度讲，矩阵 K 的任何两列都是线性独立的。

6.10 Mahalanbis 核定义为

$$k(\mathbf{x}, \mathbf{x}_i) = \exp(-(\mathbf{x} - \mathbf{x}_i)^T \sum^{-1} (\mathbf{x} - \mathbf{x}_i))$$

其中 $x \in \mathcal{H}$是 M 维输入向量，$i = 1,2,\cdots,N$。$M\times M$ 阶矩阵

$$\sum = \mathrm{diag}(\sigma_1^2, \sigma_2^2, \cdots, \sigma_M^2)$$

其中 $\sigma_1, \sigma_2, \cdots, \sigma_M$ 都是正的。这个核区别于高斯核的显著性质就是，输入空间$\mathcal{H}$的每个坐标都带有一个平滑参数（即特殊的 σ）。

为了说明这个性质，考虑函数

$$F(\mathbf{x}) = \sum_{i=1}^{N} a_i \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{2\sigma_i^2}\right)$$

可以视之为一个密度估计（Herbrich，2002）。对于所有 i 给定 $a_i = 1$ 和 $\sigma_i = \sigma$，$M = 2$，$N = 20$，画出函数 $F(\mathbf{x})$对值坐标 x_1 和 x_2 的图形。

(i) $\sigma = 0.5$

(ii) $\sigma = 0.7$

(iii) $\sigma = 1.0$

(iv) $\sigma = 2.0$

并评述你的结果。

6.11 泛×泛内积空间中联合密度函数 $p_{X_1,X_2}(x_1,x_2)$被称为 *P*-矩阵，只要满足非负性（即半正定性）(Shawe-Taylor and Cristianini, 2004)。

考虑两个随机变量的集合 $\mathbf{X}=\{X_1,X_2\}$。证明下面陈述的正确性：所有 *P*-矩阵都是联合分布，但不是所有联合分布都是 *P*-矩阵。

模式分类

6.12 边界在支持向量机的设计中起了很重要的作用。鉴别其在解模式分类问题中的重要作用。

6.13 使用式(6.17)，说明线性可分的模式中的边界可以用拉格朗日乘子表示

$$\rho=\frac{1}{\left(\sum_{i=1}^{N_S}a_i\right)^{1/2}}$$

其中 N_s 是支持向量个数。

6.14 考虑带正反例的线性可分的训练样本 $\{(\mathbf{x}_i,d_i)\}_{i=1}^{N}$。证明下面的说法：

支持向量包括用来分别正反例的所有信息。

6.15 图 P6.15 说明了包括正反例的非线性可分的数据集合。具体地讲，正反例之间的决策边界是椭圆形。找到一种映射使得样本在特征空间中线性可分。

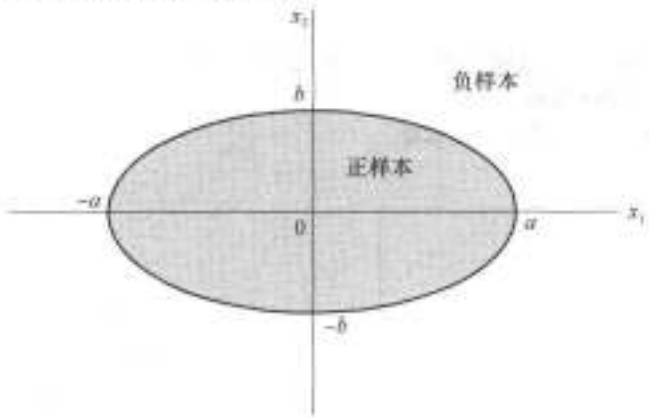


图 P6.15

6.16 用于求解 XOR 问题的多项式学习机使用的内积核定义为

$$k(\mathbf{x},\mathbf{x}_i)=(1+\mathbf{x}^T\mathbf{x}_i)^p$$

求解 XOR 问题的指数 p 的最小值是多少？假定 p 为正整数。使用比最小值大的 p 值会出现什么结果？

6.17 图 P6.17 表示三维模式 $\mathbf{x}$ 上运算的 XOR 函数，描述为

$$\mathrm{XOR}(x_1,x_2,x_3)=x_1\oplus x_2\oplus x_3$$

其中符号⊕表示异或布尔函数运算符。设计一个多项式学习机，分离由这个运算符输出所表示的两类点。

稀疏性

6.18 证明下面的说法：

支持向量机的解是稀疏的，但与之相关的 Gram 矩阵很少是稀疏的。

6.19 支持向量机解的二次规划例程提供了把训练数据分成三类的基础。定义这三个类，并且用一个二维的图来说明是如何完成这种分解的。

度量

6.20 许多不同的方法可用来快速获得支持向量机的解，所以对于不同的算法性能之间的比较很重要。建立一套度量体系用来处理这样的实际问题。

再生核空间

6.21 令 $k(\mathbf{x}_i,\cdot)$和 $k(\mathbf{x}_j,\cdot)$记为一对核，其中 $i,j=1,2,\cdots,N$，向量 $\mathbf{x}_i$ 和 $\mathbf{x}_j$ 有相同的维数，证明：

$$k(\mathbf{x}_i,\cdot)k(\mathbf{x}_j,\cdot)=k(\mathbf{x}_i,\mathbf{x}_j)$$

其中等式左边是内积核。

6.22 式(6.77)、式(6.78)和式(6.79)描述了式(6.75)内积 $\langle f,g\rangle$ 最重要的三个性质。证明这三个等式描述的性质。

6.23 证明下面的说法：

如果存在一个再生核 $k(\mathbf{x},\mathbf{x}')$，那么该核是唯一的。

计算机实验

6.24 考虑在图 1.8 中重合不可分的情况。

(a) 重复图 6.7 中的第二部分实验，两个月亮之间的垂直可分界为 $d=-6.5$。通过实验决定 C 值使得识别误差达到最小。

(b) 通过设定降低两个月亮之间的垂直距离 $d=-6.75$，识别误差比 $d=-6.5$ 时更高。通过实验决定参数 C 使得训练误差最小。

评价你的结果。

6.25 在至今的监督学习算法中，支持向量机以其强大的能力而著名。在这个问题上，支持向量机受到了图 P6.25 "紧握拳头" 形状分类问题的挑战。图中三个同心圆的半径分别为 $d_1=0.2$，$d_2=0.5$ 和 $d_3=0.8$

(a) 产生 100 回合，每个回合随机选择 200 个训练样本，对于图 P6.25 中的两个区域各产生相同的测试数据。

(b) 设 $C=500$，训练一个支持向量机。据此，构造此机器计算出的决策边界。

(c) 测试网络并且确定分类的误差率。

(d) 对 $C=100$ 和 $C=2\,500$ 重复以上试验。

评价你的结果。

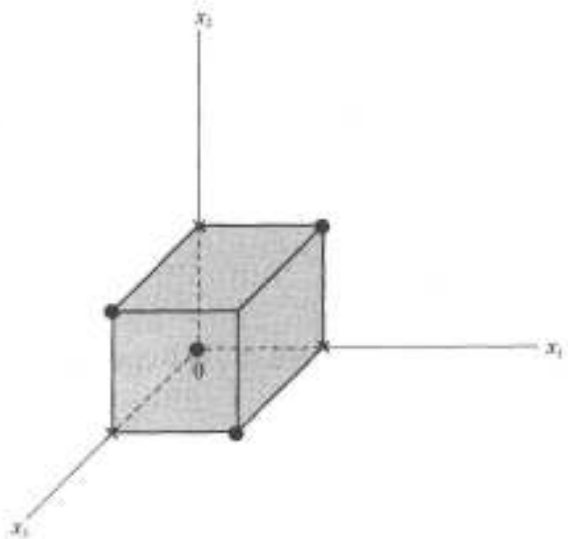


图 P6.17

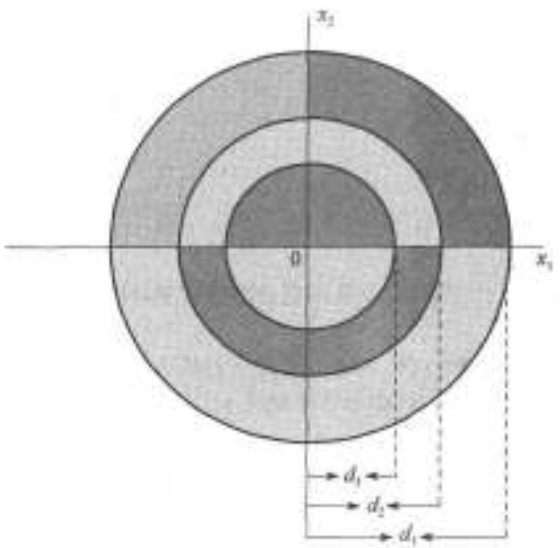


图 P6.25

正则化理论

本章组织

本章关注正则化理论的多个方面，它是所有神经网络和机器学习算法的核心。在7.1节介绍基础知识之后，我们按以下顺序组织本章：

7.2节给出病态求逆问题。

7.3节给出Tikhonov正则化理论，它提供了半监督学习算法的正则化的数学基础。本章此部分也包括7.4节，关注隐藏层与训练样本数量相同的正则化网络。7.5节讨论一类广义径向基函数网络，其隐藏层是具有正则化网络特征的子集。正则化最小二乘估计在7.6节中被重新提到，作为广义径向基函数类的特例。接下来，在7.7节中，我们说明由正则化最小二乘估计推出的深刻观点，即在不使用Tikhonov正则化理论的情况下，如何被用于其他估计算子的正则化。

7.8节描述基于交叉验证来估计正则化参数的一个过程。

本章的最后一部分开始于7.9节中对半监督学习的讨论。然后，关于流形正则化的基本观点在7.10节到7.12节中讨论。7.13节介绍光谱图理论。7.14节讨论在流形正则化理论下的广义表示定理。7.15节研究（使用带类标样本和无类标样本的）光谱图理论的正则化最小二乘估计，其作为广义正则化理论的一个应用实例。在7.16节中，我们采用最小二乘估计给出一个半监督学习的计算机实验。

7.17节给出本章的小结和讨论。

7.1 引言

在本书前几章所讨论的监督学习算法中，我们发现尽管过程不同，但它们都有一个共同点：

通过样本训练一个网络，对于给定的输入模式给出输出模式，等价于构造一个超平面（即多维映射），用输入模式定义输出模式。

从样本中学习是一个可逆的问题，因为其公式是建立在由相关直接问题的实例中获得的知识之上；后一类问题包含潜在的未知物理定律。但是，在现实情况下，我们通常发现训练样本会受到极大的局限：

训练样本所包含的信息内容通常不能够充分地由自身唯一地重构出未知的输入输出映射。因此就产生了机器学习的过拟合的可能性。

为了克服这个严重的问题，我们可以使用正则化方法，其目的是通过最小化如下的代价函数的方法把超平面重构问题的求解限制在压缩子集中：

$$\text{（正则化代价函数）}=\text{（经验代价函数）}+\text{（正则化参数）}\times\text{（正则化项）}$$

给定一个训练样本，假设经验风险或标准代价函数可以由误差平方和定义。附加的正则化算子是用来平滑超平面重构问题的解。因此，通过选择一个适当的正则化参数（在设计者控制下），正则化代价函数提供了在训练样本的精度（包含在均方误差中）和解的光滑程度之间的折中。

本章学习两个基本的重要问题：

1. 经典正则化理论，它建立在我们刚刚描述的正则化代价函数上。这个由Tikhonov

(1963)给出的优美理论，为前面章节中讨论的正则化算子提供了统一的数学基础。另外，它提出了新的思想。

2. 广义正则化理论，它通过引入第三个项，扩展了 Tikhonov 的经典正则化理论公式。这个新项叫做流形正则化算子，由 Belkin 等(2006) 提出，研究用于产生无类标样本（即没有预期响应的样本）的输入空间的边缘概率分布。此广义正则化理论对依赖于结合使用带类标样本和无类标样本的半监督学习提供了数学基础。

7.2　良态问题的 Hadamard 条件

良态（well posed）这个词是由 Hadamard(1902)最初提出的，并且在应用数学中一直沿用至今。要解释这个术语，假定我们有一个定义域$\mathscr{X}$和一个值域$\mathscr{Y}$，其中通过一个固定但未知的映射 f 关联。如果以下三个 Hadamard 条件成立，那么重构映射 f 的问题就被称为是良态的(Tikhonov and Arsenin，1977；Morozov，1993；Kirsch，1996)：

1. 存在性　对于每个输入向量 $\mathbf{x}\in\mathscr{X}$，存在一个输出 $y=f(\mathbf{x})$，其中 $y\in\mathscr{Y}$。

2. 唯一性　对于任意输入向量对 $\mathbf{x},t\in\mathscr{X}$，有 $f(\mathbf{x})=f(\mathbf{t})$当且仅当 $\mathbf{x}=\mathbf{t}$。

3. 连续性　映射 f 是连续的；即对于任意的 $\varepsilon>0$，存在 $\delta=\delta(\varepsilon)$使得条件 $\rho_x(\mathbf{x},\mathbf{t})<\delta$ 蕴含 $\rho_y(f(\mathbf{x}),f(\mathbf{t}))<\varepsilon$。其中 $\rho(\cdot,\cdot)$表示两个变量各自空间之间的距离。此准则如图 7.1 所示。连续性同样也称为稳定性。

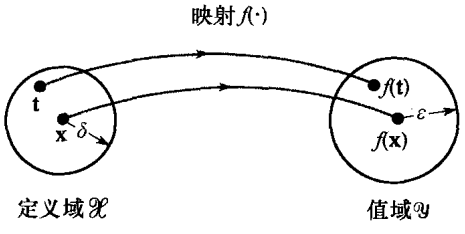


图 7.1　从输入域$\mathscr{X}$到输出域$\mathscr{Y}$的映射举例

如果这些条件中的任何一个都不满足，就称此问题为病态的（ill posed）。基本上说，病态问题意味着大的数据集可能只包含关于预期解的一小部分信息。

在监督学习的环境下，Hadamard 条件可能由于以下原因被破坏。一，存在性准则可能会因为对于每个输入不一定存在唯一的输出而被破坏。二，训练样本中可能没有许多我们所需要的用于构造一个唯一的输入输出映射的信息；因此，唯一性准则可能被破坏。三，在实际训练数据中噪声或不准确数据是不可避免的，这增加了重构过程的不确定性。特别地，如果输入数据中的噪声级别很高，神经网络或机器学习会对定义域$\mathscr{X}$中的特定输入 $\mathbf{x}$ 产生一个在值域$\mathscr{Y}$之外的输出；换而言之，连续性准则可能会被破坏。如果一个学习问题不具有连续性，则所计算的输入输出映射与学习问题的准确解无关。没有什么办法可以解决这些困难，除非我们可以获得一些关于输入输出映射的先验信息。在这个背景下，我们可以用 Lanczos 关于线性微分算子(Lanczos，1964) 的一句论断提醒我们自己：

任何数学技巧都不能补救信息的缺失。

7.3　Tikhonov 正则化理论

1963 年 Tikhonov 提出了一种新方法用以解决病态问题，该方法就是正则化。在曲面重建的问题上，正则化的基本思想就是通过某些含有解的先验知识的非负的辅助泛函来使解稳定。先验知识的一般形式涉及假设输入输出映射函数（即重建问题的解）是光滑的，即

对于一个光滑的输入输出映射，相似的输入对应着相似的输出。

具体来说，我们将用于逼近的输入输出数据（即训练样本）集合描述如下：

$$\begin{aligned}&\text{输入信号}\quad \mathbf{x}_i\in\mathbb{R}^m,\quad i=1,2,\cdots,N\\&\text{期望响应}\quad d_i\in\mathbb{R},\quad i=1,2,\cdots,N\end{aligned}\tag{7.1}$$

注意这里假定输出是一维的。这种假设并不会限制这里讨论的正则化理论的一般性应用。用$F(\mathbf{x})$表示逼近函数，这里为了方便表达，我们在变量中省掉了神经网络的权值向量$\mathbf{w}$。从根本上说，Tikhonov的正则化理论包含两项：

1. *误差函数*，该项用$\mathscr{E}_s(F)$表示，以逼近函数$F(\mathbf{x}_i)$和训练样本$\{\mathbf{x}_i, d_i\}_{i=1}^N$的形式定义。例如，对于最小二乘估计，我们有如下的标准代价（损失）函数：

$$\mathscr{E}_s(F) = \frac{1}{2}\sum_{i=1}^{N}(d_i - F(\mathbf{x}_i))^2 \tag{7.2}$$

其中$\mathscr{E}_s$中的下标s表示“标准化”。对于另外一个不同的例子，即支持向量机，我们有*边缘损失函数*：

$$\mathscr{E}_s(F) = \frac{1}{N}\sum_{i=1}^{N}\max(0, 1 - d_i F(\mathbf{x}_i)), \quad d_i \in \{-1, +1\}$$

我们当然可以把所有的例子包含在一个简单的公式中，但这两个基本损失函数的含义是完全不同的，它们的理论研究也早晚会被不同对待。为了能够清楚地阐述，我们将关注式(7.2)中的误差函数。

2. *正则化项*，用$\mathscr{E}_c(F)$表示，依赖于逼近函数$F(\mathbf{x}_i)$的“几何”性质。具体定义为

$$\mathscr{E}_c(F) = \frac{1}{2}\|\mathbf{D}F\|^2 \tag{7.3}$$

其中，$\mathscr{E}_c$中的下标c代表复杂度，$\mathbf{D}$是线性微分算子。关于解（即输入输出映射函数$F(\mathbf{x})$）的形式的先验知识就包含在算子$\mathbf{D}$中，这就自然使得$\mathbf{D}$的选取与所解的问题有关。我们也称$\mathbf{D}$为*稳定因子*（stabilizer），因为它使正则化问题的解稳定，使解光滑从而满足连续性的要求。但是，光滑性意味着连续性，而相反未必为真。用于处理式(7.3)所描述情况的解析方法是建立在第6章所讨论的Hilbert空间的概念之上的。在这样的多维（严格说来是无限多维）空间中，一个连续函数由一个向量来表示。通过使用几何图像，我们就可以在线性微分算子和矩阵之间建立深刻的联系。由此对线性系统的分析就可以转变为对线性微分方程的分析（Lanczos，1964）。于是，式(7.3)中的符号$\|\cdot\|$表示定义在$\mathbf{D}F(\mathbf{x})$所属的Hilbert空间上的范数。把线性微分算子$\mathbf{D}$看成一个从F所属的函数空间到Hilbert空间的映射，我们很自然地在式(7.3)中使用L_2范数。

训练样本$\mathcal{T}=\{\mathbf{x}_i, d_i\}_{i=1}^N$，由一个物理过程产生，用如下的回归模型表示：

$$d_i = f(\mathbf{x}_i) + \varepsilon_i, \quad i = 1, 2, \cdots, N$$

其中$\mathbf{x}_i$是回归量，d_i是响应，ε_i是解释误差。严格地说，我们需要函数$f(\mathbf{x})$是有Dirac delta分布形式的带有再生核的*再生核Hibert核空间*（RKHS）（Tapia and Thompson，1978）；此要求的必要性将在后面的讨论中给出。RKHS的概念已在第6章中讨论过。

令$\mathscr{E}_s(F)$表示标准代价（损失）函数，$\Omega(F)$表示正则化函数。则假定在正则化理论中，用于最小化的最小二乘损失量为：

$$\mathscr{E}(F) = \mathscr{E}_s(F) + \Omega(F) = \frac{1}{2}\sum_{i=1}^{N}[d_i - F(\mathbf{x}_i)]^2 + \frac{1}{2}\lambda\|\mathbf{D}F\|^2 \tag{7.4}$$

其中λ是一个称为*正则化参数*的正实数，$\mathscr{E}(F)$叫做Tikhonov*泛函*。泛函（定义在一些适当的函数空间中）把函数映射为实数。Tikhonov泛函$\mathscr{E}(F)$的最小点（即正则化问题的解）用$F_\lambda(\mathbf{x})$表示。值得注意的是，式(7.4)可以看作一个有约束的最优化问题：在施加在$\Omega(F)$上的约束条件下最小化$\mathscr{E}_s(F)$。为了实现此目的，我们强调一个在逼近函数F的复杂度上的显式约束。

另外，我们可以把正则化参数λ看作在由给定训练样本确定的解$F_\lambda(\mathbf{x})$的充分条件的指示器。特别地，在$\lambda \to 0$极限条件下，此问题是无约束的，因为$F_\lambda(\mathbf{x})$的解完全由样本确定。在

另一个 $\lambda \to \infty$ 的极限条件下，由微分算子 $\mathbf{D}$ 施加的先验光滑约束对求解 $F_\lambda(\mathbf{x})$ 是充分的。换句话说，样本是不可靠的。在实际应用中，正则化参数 λ 被赋予一个在这两种极限条件之间的值，所以训练样本和先验知识都可以对求解 $F_\lambda(\mathbf{x})$ 起到作用。因此，正则项 $\mathscr{C}_c(F) = \frac{1}{2}\|\mathbf{D}F\|^2$ 代表一个复杂度罚函数模型，其对最终解的影响由正则化参数 λ 控制。

另外，我们可以把正则化过程看作对第 2 章中所讨论的有偏方差问题的解决。特别地，正则化参数的最优选择可用来通过加入正确的先验信息，以在模型偏置和模型方差中平衡来实现。此方法可以解决一些学习问题。

Tikhonov 正则化应用

对正则化理论的讨论至此，我们一直强调如使用式(7.1)中 $d_i \in \mathbb{R}$ 的回归问题。然而，我们必须认识到 Tikhonov 正则化理论同样可以应用于以下两个其他领域：

1. *分类*。此问题可以简单地通过诸如把二值类标当作标准最小二乘回归中的实值来解决。在另外的例子中，我们可以使用经验风险（即代价）函数，比如说更适合模式分类问题的关键损失。第 6 章中讨论的支持向量机就是如此。

2. *结构预测*。在一些最近的工作中，已将 Tikhonov 正则化理论用于结构预测，比如，输出空间可以是一个序列、一棵树或其他一些结构的输出空间（Bakir 等，2007）。

这里我们希望强调的是，正则化理论在几乎所有的需要从有限数量的训练样本中学习的应用中都处于核心地位。

Tikhonov 泛函的 Fréchet 微分

正则化原理可以表述如下：

求使 Tikhonov 泛函 $\mathscr{C}(F)$ 最小的逼近函数 $F_\lambda(\mathbf{x})$，Tikhonov 泛函由

$$\mathscr{C}(F) = \mathscr{C}_s(F) + \lambda \mathscr{C}_c(F)$$

定义，其中 $\mathscr{C}_s(F)$ 是标准误差项，$\mathscr{C}_c(F)$ 是正则化项，而 λ 是正则化参数。

为进行代价泛函 $\mathscr{C}(F)$ 的最小化，我们首先需要求 $\mathscr{C}(F)$ 微分的规则。可以用 Fréchet 微分来处理这件事。在初等微积分中，曲线上某点的切线是在该点邻域上的曲线的最佳逼近直线。同理，一个泛函的 Fréchet 微分可以解释为一个最佳局部线性逼近。这样泛函 $\mathscr{C}(F)$ 的 Fréchet 微分可正式定义如下：

$$\mathrm{d}\mathscr{C}(F,h) = \left[\frac{\mathrm{d}}{\mathrm{d}\beta}\mathscr{C}(F+\beta h)\right]_{\beta=0} \tag{7.5}$$

其中 $h(\mathbf{x})$ 是一个固定的关于向量 $\mathbf{x}$ 的函数（Dorny，1975；Debnath and Mikusiński，1990；de Figueiredo and Chen，1993）。在式(7.5)中应用通常的微分法则。对于所有的 $h \in \mathscr{H}$，函数 $F(\mathbf{x})$ 为泛函 $\mathscr{C}(F)$ 的一个相对极值的必要条件是，泛函 $\mathscr{C}(F)$ 的 Fréchet 微分 $d\mathscr{C}(F,h)$ 在 $F(\mathbf{x})$ 处均为零，表示为

$$\mathrm{d}\mathscr{C}(F,h) = \mathrm{d}\mathscr{C}_s(F,h) + \lambda \mathrm{d}\mathscr{C}_c(F,h) = 0 \tag{7.6}$$

其中 $\mathrm{d}\mathscr{C}_s(F,h)$ 和 $\mathrm{d}\mathscr{C}_c(F,h)$ 分别是泛函 $\mathscr{C}_s(F)$ 和 $\mathscr{C}_c(F)$ 的 Fréchet 微分。为了简化表示，在式(7.5)中用 h 来代替 $h(\mathbf{x})$。

计算式(7.2)中标准误差项 $\mathscr{C}_s(F,h)$ 的 Fréchet 微分如下：

$$\begin{aligned}\mathrm{d}\mathscr{C}_s(F,h) &= \left[\frac{\mathrm{d}}{\mathrm{d}\beta}\mathscr{C}_s(F+\beta h)\right]_{\beta=0} = \left[\frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}\beta}\sum_{i=1}^{N}[d_i - F(\mathbf{x}_i) - \beta h(\mathbf{x}_i)]^2\right]_{\beta=0} \\ &= -\sum_{i=1}^{N}[d_i - F(\mathbf{x}_i) - \beta h(\mathbf{x}_i)]h(\mathbf{x}_i)\big|_{\beta=0} = -\sum_{i=1}^{N}[d_i - F(\mathbf{x}_i)]h(\mathbf{x}_i)\end{aligned} \tag{7.7}$$

Riesz 表示理论

为了继续处理 Hilbert 空间中的 Fréchet 微分问题，我们发现引入 Riesz 表示定理是有益的，陈述如下（Debnath and Mikusiński，1990）：

令 f 为 Hilbert 空间$\mathscr{H}$上的一个有界线性泛函。存在一个 $h_0 \in \mathscr{H}$，使得

$$f(h) = \langle h, h_0 \rangle_{\mathscr{H}}, \quad \text{对所有 } h \in \mathscr{H}$$

且

$$\| f \| = \| h_0 \|_{\mathscr{H}}$$

其中 h_0 和 f 在它们各自空间上都存在范数。

这里所用的符号 $\langle \cdot, \cdot \rangle_{\mathscr{H}}$表示$\mathscr{H}$空间上两个函数的内积（标量）。因此，根据 Riesz 表示定理，可以重写式(7.7)中的 Fréchet 微分 $\mathrm{d}\mathscr{E}_s(F, h)$如下：

$$\mathrm{d}\mathscr{E}_s(F,h) = -\langle h, \sum_{i=1}^{N} (d_i - F)\delta_{\mathbf{x}_i} \rangle_{\mathscr{H}} \tag{7.8}$$

其中 $\delta_{\mathbf{x}_i}$ 表示以 $\mathbf{x}_i$ 为中心的 $\mathbf{x}$ 的 Dirac delta 分布；即

$$\delta_{\mathbf{x}_i}(\mathbf{x}) = \delta(\mathbf{x} - \mathbf{x}_i) \tag{7.9}$$

下面计算式(7.3)的正则化项$\mathscr{E}_c(F)$的 Fréchet 微分。用与上面同样的方法可以得出（假设 $\mathbf{D}F \in L_2(\mathbb{R}^{m_0})$）：

$$\begin{aligned} \mathrm{d}\mathscr{E}_c(F,h) &= \frac{\mathrm{d}}{\mathrm{d}\beta}\mathscr{E}_c(F+\beta h)|_{\beta=0} = \frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}\beta}\int_{\mathbb{R}^{m_0}} (\mathbf{D}[F+\beta h])^2 \mathrm{d}\mathbf{x}|_{\beta=0} \\ &= \int_{\mathbb{R}^m} \mathbf{D}[F+\beta h]\mathbf{D}h \, \mathrm{d}\mathbf{x}|_{\beta=0} = \int_{\mathbb{R}^m} \mathbf{D}F\mathbf{D}h \, \mathrm{d}\mathbf{x} = \langle \mathbf{D}h, \mathbf{D}F \rangle_{\mathscr{H}} \end{aligned} \tag{7.10}$$

其中 $\langle \mathbf{D}h, \mathbf{D}F \rangle_{\mathscr{H}}$是函数 $\mathbf{D}h(\mathbf{x})$和 $\mathbf{D}F(\mathbf{x})$的内积，函数 $\mathbf{D}h(\mathbf{x})$和 $\mathbf{D}F(\mathbf{x})$分别代表了微分算子 $\mathbf{D}$ 作用在 $h(\mathbf{x})$和 $F(\mathbf{x})$上的结果。

Euler-拉格朗日方程

给定一个线性微分算子 $\mathbf{D}$，我们可以唯一确定它的伴随算子（adjoint operator）$\tilde{\mathbf{D}}$，使得对任一对足够可微且满足恰当的边界条件的函数 $u(\mathbf{x})$和 $v(\mathbf{x})$ 有（Lanczos，1964）：

$$\int_{\mathbb{R}^m} u(\mathbf{x})\mathbf{D}v(\mathbf{x})\mathrm{d}\mathbf{x} = \int_{\mathbb{R}^m} v(\mathbf{x})\tilde{\mathbf{D}}u(\mathbf{x})\mathrm{d}\mathbf{x} \tag{7.11}$$

等式(7.11)叫做 Green 恒等式，它为通过给定微分算子 $\mathbf{D}$ 来确定其伴随算子$\tilde{\mathbf{D}}$提供一个数学基础。将 $\mathbf{D}$ 看作一个矩阵，则其伴随算子$\tilde{\mathbf{D}}$的作用类似于一个转置矩阵的作用。

比较式(7.11)的左边和式(7.10)的第四行，我们可得出如下恒等式：

$$u(\mathbf{x}) = \mathbf{D}F(\mathbf{x})$$

$$\mathbf{D}v(\mathbf{x}) = \mathbf{D}h(\mathbf{x})$$

根据 Green 恒等式，可将式(7.10)重写为如下等价形式：

$$\mathrm{d}\mathscr{E}_c(F,h) = \int_{\mathbb{R}^{m_0}} h(\mathbf{x})\tilde{\mathbf{D}}\mathbf{D}F(\mathbf{x})\mathrm{d}\mathbf{x} = \langle h, \tilde{\mathbf{D}}\mathbf{D}F \rangle_{\mathscr{H}} \tag{7.12}$$

其中$\tilde{\mathbf{D}}$是 $\mathbf{D}$ 的伴随算子。

将式(7.8)和式(7.12)代入极值条件(7.6)中，可以重新得到 Fréchet 微分 $\mathrm{d}\mathscr{E}(F,h)$如下：

$$\mathrm{d}\mathscr{E}(F,h) = \left\langle h, \left[\tilde{\mathbf{D}}\mathbf{D}F - \frac{1}{\lambda}\sum_{i=1}^{N}(d_i - F)\delta_{\mathbf{x}_i} \right] \right\rangle_{\mathscr{H}} \tag{7.13}$$

因为正则化参数 λ 通常取开区间（0，∞）上的某个值，所以当且仅当下列条件在广义函

数 $F=F_\lambda$ 下满足时，对于空间$\mathcal{H}$中的所有函数 $h(\mathbf{x})$，Fréchet 微分 $\mathrm{d}\mathscr{E}(F,h)$才为零：

$$\tilde{\mathbf{D}}\mathbf{D}F_\lambda - \frac{1}{\lambda}\sum_{i=1}^{N}(d_i - F_\lambda)\delta_{\mathbf{x}_i} = 0$$

或者等价于：

$$\tilde{\mathbf{D}}\mathbf{D}F_\lambda(\mathbf{x}) = \frac{1}{\lambda}\sum_{i=1}^{N}[d_i - F_\lambda(\mathbf{x}_i)]\delta(\mathbf{x}-\mathbf{x}_i) \tag{7.14}$$

式(7.14)是 Tikhonov 泛函$\mathscr{E}(F)$的 Euler-拉格朗日方程，它定义了 Tikhonov 泛函$\mathscr{E}(F)$在 $F_\lambda(\mathbf{x})$处有极值的必要条件（Debnath and Mikusiński，1990）。

Green 函数

式(7.14)表示逼近函数 F_λ 的偏微分方程。该方程的解是由方程右边的积分变换组成的。我们现在先简单地介绍 Green 函数，然后再继续求解式(7.14)。

令 $G(\mathbf{x},\boldsymbol{\xi})$ 表示向量 $\mathbf{x}$ 和 $\boldsymbol{\xi}$ 的一个函数，两个向量的地位相同，但它们的目的不同：向量 $\mathbf{x}$ 作为参数，而向量 $\boldsymbol{\xi}$ 则作为自变量。对于给定的线性微分算子 $\mathbf{L}$，我们规定函数 $G(\mathbf{x},\boldsymbol{\xi})$满足如下条件（Courant and Hilbert，1970）：

1. 当固定的 $\boldsymbol{\xi}$,$G(\mathbf{x},\boldsymbol{\xi})$ 是 x 的函数，且满足规定的边界条件。

2. 除了在点 $\mathbf{x}=\boldsymbol{\xi}$ 外，$G(\mathbf{x},\boldsymbol{\xi})$ 对于 $\mathbf{x}$ 的导数是连续的。导数的次数由线性算子 L 的阶数决定。

3. 将 $G(\mathbf{x},\boldsymbol{\xi})$ 看作 x 的函数，除了在点 $\mathbf{x}=\boldsymbol{\xi}$ 奇异外，它满足偏微分方程

$$\mathbf{L}G(\mathbf{x},\boldsymbol{\xi}) = 0 \tag{7.15}$$

也即函数 $G(\mathbf{x},\boldsymbol{\xi})$满足（在广义函数的意义下）

$$\mathbf{L}G(\mathbf{x},\boldsymbol{\xi}) = \delta(\mathbf{x}-\boldsymbol{\xi}) \tag{7.16}$$

其中，如前定义 $\delta(\mathbf{x}-\boldsymbol{\xi})$是位于点 $\mathbf{x}=\boldsymbol{\xi}$ 的 Dirac delta 函数。

上述的函数 $G(\mathbf{x},\boldsymbol{\xi})$ 叫做微分算子 $\mathbf{L}$ 的 Green 函数（Courant and Hilbert，1970）。Green 函数对于线性微分算子的作用类似于一个矩阵的逆矩阵对该矩阵方程的作用。

令 $\varphi(\mathbf{x})$表示一个关于 $\mathbf{x}\in\mathbb{R}^{m_0}$ 的连续或者分段连续的函数。那么函数

$$F(\mathbf{x}) = \int_{\mathbb{R}^{m_0}} G(\mathbf{x},\boldsymbol{\xi})\varphi(\boldsymbol{\xi})\mathrm{d}\boldsymbol{\xi} \tag{7.17}$$

就是微分方程

$$\mathbf{L}F(\mathbf{x}) = \varphi(\mathbf{x}) \tag{7.18}$$

的解，其中 $G(\mathbf{x},\boldsymbol{\xi})$ 是线性微分算子 $\mathbf{L}$ 的 Green 函数。

为了证明 $F(\mathbf{x})$为式(7.18)的解，我们将微分算子 $\mathbf{L}$ 应用于式(7.17)的两端，可得

$$\mathbf{L}F(\mathbf{x}) = \mathbf{L}\int_{\mathbb{R}^{m_0}} G(\mathbf{x},\boldsymbol{\xi})\varphi(\boldsymbol{\xi})\mathrm{d}(\boldsymbol{\xi}) = \int_{\mathbb{R}^{m_0}} \mathbf{L}G(\mathbf{x},\boldsymbol{\xi})\varphi(\boldsymbol{\xi})\mathrm{d}\boldsymbol{\xi} \tag{7.19}$$

微分算子 $\mathbf{L}$ 将 $\boldsymbol{\xi}$ 视为常量，它作用于 $G(\mathbf{x},\boldsymbol{\xi})$ 时仅将其视为 $\mathbf{x}$ 的函数。将式(7.16)代入式(7.19)，有

$$\mathbf{L}F(\mathbf{x}) = \int_{\mathbb{R}^{m_0}} \delta(\mathbf{x}-\boldsymbol{\xi})\varphi(\boldsymbol{\xi})\mathrm{d}\boldsymbol{\xi}$$

最后，利用 Dirac Delta 函数的筛选性质，可得

$$\int_{\mathbb{R}^{m_0}} \varphi(\boldsymbol{\xi})\delta(\mathbf{x}-\boldsymbol{\xi})\mathrm{d}(\boldsymbol{\xi}) = \varphi(\mathbf{x})$$

这样就得到了如式(7.18)所描述的 $\mathbf{L}F(\mathbf{x})=\varphi(\mathbf{x})$。

正则化问题的解

回到当前的问题，下面我们来解 Euler-拉格朗日微分方程，即式(7.14)。令

$$\mathbf{L} = \tilde{\mathbf{D}}\mathbf{D} \tag{7.20}$$

和

$$\varphi(\boldsymbol{\xi}) = \frac{1}{\lambda}\sum_{i=1}^{N}[d_i - F(\mathbf{x}_i)]\delta(\boldsymbol{\xi} - \mathbf{x}_i) \tag{7.21}$$

那么根据式(7.17)，有

$$\begin{aligned} F_\lambda(\mathbf{x}) &= \int_{\mathbb{R}^{m_0}} G(\mathbf{x},\boldsymbol{\xi})\left\{\frac{1}{\lambda}\sum_{i=1}^{N}[d_i - F(\mathbf{x}_i)]\delta(\boldsymbol{\xi} - \mathbf{x}_i)\right\}\mathrm{d}\boldsymbol{\xi} \\ &= \frac{1}{\lambda}\sum_{i=1}^{N}(d_i - F(\mathbf{x}_i))\int_{\mathbb{R}^{m_0}} G(\mathbf{x},\boldsymbol{\xi})\delta(\boldsymbol{\xi} - \mathbf{x}_i)\mathrm{d}\boldsymbol{\xi} \end{aligned}$$

上式第二行交换了积分与求和的次序。最后，利用 Dirac Delta 函数的筛选性质，可以得到 Euler-拉格朗日微分方程（7.14）的解如下：

$$F_\lambda(\mathbf{x}) = \frac{1}{\lambda}\sum_{i=1}^{N}[d_i - F(\mathbf{x}_i)]G(\mathbf{x},\mathbf{x}_i) \tag{7.22}$$

式(7.22)说明正则化问题的最小化解 $\mathbf{F}_\lambda(\mathbf{x})$ 是 N 个 Green 函数的线性叠加。$\mathbf{x}_i$ 代表扩展中心，权值 $[d_i - F(\mathbf{x}_i)]/\lambda$ 代表展开系数。换句话说，正则化问题的解在光滑函数的空间的一个 N 维子空间上，以 $\mathbf{x}_i, i = 1,2,\cdots,N$ 为中心的一组 Green 函数 $\{G(\mathbf{x},\mathbf{x}_i)\}$ 组成了该子空间的基(Poggio and Girosi，1990a)。注意式(7.22)中，展开系数具有如下性质：

- 与系统的估计误差（定义为应有输出 d_i 和相应的网络实际计算输出 $F(\mathbf{x}_i)$之差）呈线性关系。
- 与正则化参数 λ 成反比。

确定展开系数

下面将要解决的问题是如何确定式(7.22)中的展开系数。令

$$w_i = \frac{1}{\lambda}[d_i - F(\mathbf{x}_i)], \quad i = 1,2,\cdots,N \tag{7.23}$$

则正则化问题的最小化的解式(7.22)可以改写成如下形式：

$$F_\lambda(\mathbf{x}) = \sum_{i=1}^{N} w_i G(\mathbf{x},\mathbf{x}_i) \tag{7.24}$$

分别在 $\mathbf{x}_j (j=1, 2, \cdots, N)$ 上计算式(7.24)的值，可得

$$F_\lambda(\mathbf{x}_j) = \sum_{i=1}^{N} w_i G(\mathbf{x}_j,\mathbf{x}_i), \quad j = 1,2,\cdots,N \tag{7.25}$$

现在我们引入如下定义：

$$\mathbf{F}_\lambda = [F_\lambda(\mathbf{x}_1), F_\lambda(\mathbf{x}_2), \cdots, F_\lambda(\mathbf{x}_N)]^T \tag{7.26}$$

$$\mathbf{d} = [d_1, d_2, \cdots, d_N]^T \tag{7.27}$$

$$\mathbf{G} = \begin{bmatrix} G(\mathbf{x}_1,\mathbf{x}_1) & G(\mathbf{x}_1,\mathbf{x}_2) & \cdots & G(\mathbf{x}_1,\mathbf{x}_N) \\ G(\mathbf{x}_2,\mathbf{x}_1) & G(\mathbf{x}_2,\mathbf{x}_2) & \cdots & G(x_2,\mathbf{x}_N) \\ \vdots & \vdots & & \vdots \\ G(\mathbf{x}_N,\mathbf{x}_1) & G(\mathbf{x}_N,\mathbf{x}_2) & \cdots & G(\mathbf{x}_N,\mathbf{x}_N) \end{bmatrix} \tag{7.28}$$

$$\mathbf{w} = [w_1, w_2, \cdots, w_N]^T \tag{7.29}$$

然后式(7.23)和式(7.25)可分别写成矩阵形式：

$$\mathbf{w} = \frac{1}{\lambda}(\mathbf{d} - \mathbf{F}_\lambda) \tag{7.30}$$

和

$$\mathbf{F}_\lambda = \mathbf{G}\mathbf{w} \tag{7.31}$$

消去式(7.30)和式(7.31)中的 $\mathbf{F}_\lambda$，重新调整项可得

$$(\mathbf{G}+\lambda\mathbf{I})\mathbf{w} = \mathbf{d} \tag{7.32}$$

其中 $\mathbf{I}$ 是一个 $N\times N$ 阶的单位矩阵。矩阵 $\mathbf{G}$ 称为 Green 矩阵。

式(7.20)所定义的线性微分算子 $\mathbf{L}$ 是自伴的，它的伴随算子等于它自身。因此，与其相关的 Green 函数 $G(\mathbf{x},\mathbf{x}_i)$ 是对称函数，即对所有的 i，j 都有

$$G(\mathbf{x}_i,\mathbf{x}_j) = G(\mathbf{x}_j,\mathbf{x}_i) \tag{7.33}$$

式(7.33)表明 Green 函数 $G(\mathbf{x},\boldsymbol{\xi})$ 的两个自变量 $\mathbf{x}$ 和 $\boldsymbol{\xi}$ 的位置是可以互换的而不影响它的值。等价地，式(7.28)所定义的 Green 矩阵 $\mathbf{G}$ 是对称矩阵，即

$$\mathbf{G}^T = \mathbf{G} \tag{7.34}$$

现在我们回顾一下插值定理，第5章利用插值矩阵 $\boldsymbol{\Phi}$ 对定理进行描述。我们首先注意到 Green 矩阵 $\mathbf{G}$ 在正则化理论中所起的作用与插值矩阵 $\boldsymbol{\Phi}$ 在 RBF 插值理论中所起的作用相同。它们都是 $N\times N$ 阶的对称阵。因此，我们可以说，对于某类 Green 函数，只要所提供的数据点 $\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_N$ 是互不相同的，则 Green 矩阵就是正定的。满足 Micchelli 定理的 Green 函数包括逆多二次函数和高斯函数，但是没有多二次函数。实际上，我们总是将 λ 选得足够大，使得 $\mathbf{G}+\lambda\mathbf{I}$ 是正定的，从而是可逆的。这样式(7.32)所表示的线性方程组就具有唯一解（Poggio and Girosi，1990a）：

$$\mathbf{w} = (\mathbf{G}+\lambda\mathbf{I})^{-1}\mathbf{d} \tag{7.35}$$

因此，只要选定了微分算子 $\mathbf{D}$，从而确定了相应的 Green 函数 $G(\mathbf{x}_j,\mathbf{x}_i)$，$i=1,2,\cdots,N$，我们就可以通过计算式(7.35)得到与某一特定期望输出向量 $\mathbf{d}$ 以及合适的正则化参数值 λ 相对应的权值向量 $\mathbf{w}$。

总之，我们可以说正则化问题的解可以由以下展开式给出：

$$F_\lambda(\mathbf{x}) = \sum_{i=1}^{N} w_i G(\mathbf{x},\mathbf{x}_i) \tag{7.36}$$

相应地，我们可以作出如下三条论断：

1. 最小化式(7.4)中的正则化代价函数 $\xi(F)$ 的逼近函数 $F_\lambda(\mathbf{x})$，由一系列 Green 函数的线性加权组合而成，其中每一个 Green 函数都仅依赖于一个稳定因子 $\mathbf{D}$。
2. 在展开式中所用到的 Green 函数的个数与训练过程中所用的样本数据点的个数相同。
3. 展开式中相应的 N 个权值由式(7.23)中的训练样本 $\{x_i,d_i\}_{i=1}^N$ 和正则化参数的形式定义。

如果所选的稳定因子 $\mathbf{D}$ 具有平移不变性，则以 x_i 为中心的 Green 函数 $G(\mathbf{x},\mathbf{x}_i)$ 只取决于自变量 x 和 x_i 之差；即

$$G(\mathbf{x},\mathbf{x}_i) = G(\mathbf{x}-\mathbf{x}_i) \tag{7.37}$$

如果稳定因子 $\mathbf{D}$ 是平移不变和旋转不变的，则 Green 函数 $G(\mathbf{x},\mathbf{x}_i)$ 只取决于向量 $\mathbf{x}-\mathbf{x}_i$ 的欧几里得范数，表示为

$$G(\mathbf{x},\mathbf{x}_i) = G(\|\mathbf{x}-\mathbf{x}_i\|) \tag{7.38}$$

在这些条件下，Green 函数一定是径向基函数。此时，式(7.36)的正则化问题的解可表示为如下形式：

$$F_\lambda(\mathbf{x}) = \sum_{i=1}^{N} w_i G(\|\mathbf{x}-\mathbf{x}_i\|) \tag{7.39}$$

式(7.39)所描述的解构造一个依赖于已知数据点的欧几里得距离度量的线性函数空间。

式(7.39)所描述的解叫做*严格插值解*，因为所有 N 个已知训练数据点都被用于生成插值函数 $F(\mathbf{x})$。但是，值得注意的是式(7.39)与式(5.11)所表示的解有根本不同。式(7.39)的解被式(7.35)给出的权重向量 w 的定义所正则化。只有当我们将正则化参数 λ 设为 0 时，这两个解才是一样的。

多元高斯函数

Green 函数 $G(\mathbf{x},\mathbf{x}_i)$ 的相应的线性微分算子 $\mathbf{D}$ 是平移不变和旋转不变的并且它满足式(7.38)的条件，此时 Green 函数具有重要实际意义。这类 Green 函数的一个例子是多元高斯函数，定义为

$$G(\mathbf{x},\mathbf{x}_i) = \exp\left(-\frac{1}{2\sigma_i^2}\| \mathbf{x}-\mathbf{x}_i \|^2\right) \tag{7.40}$$

其中 $\mathbf{x}_i$ 表示函数的中心，而 σ_i 则表示它的宽度。与式(7.40)所示 Green 函数相对应的自伴随算子 $\mathbf{L}=\tilde{\mathbf{D}}\mathbf{D}$ 由下式给出：

$$L = \sum_{n=0}^{\infty}(-1)^n \alpha_n \nabla^{2n} \tag{7.41}$$

其中

$$\alpha_n = \frac{\sigma_i^{2n}}{n!2^n} \tag{7.42}$$

而 ∇^{2n} 是 m_0 维多重拉普拉斯算子

$$\nabla^2 = \frac{\partial^2}{\partial x_1^2}+\frac{\partial^2}{\partial x_2^2}+\cdots+\frac{\partial^2}{\partial x_{m_0}^2} \tag{7.43}$$

因为式(7.41)中 L 的项数允许到无穷大，所以从标准意义上说 $\mathbf{L}$ 并不是一个微分算子。因此，我们将式(7.41)中的 $\mathbf{L}$ 称为*伪微分算子*。

由于定义 $\mathbf{L}=\tilde{\mathbf{D}}\mathbf{D}$，由式(7.41)可以推导出算子 $\mathbf{D}$ 和 $\tilde{\mathbf{D}}$：

$$\mathbf{D} = \sum \alpha_n^{1/2}\left(\frac{\partial}{\partial x_1}+\frac{\partial}{\partial x_2}+\cdots+\frac{\partial}{\partial x_{m_0}}\right)^n = \sum_{a+b+\cdots+k=n} \alpha_n^{1/2}\frac{\partial^n}{\partial x_1^a \partial x_2^b \cdots \partial x_{m_0}^k} \tag{7.44}$$

和

$$\tilde{\mathbf{D}} = \sum_n (-1)^n \alpha_n^{1/2}\left(\frac{\partial}{\partial x_1}+\frac{\partial}{\partial x_2}+\cdots+\frac{\partial}{\partial x_{m_0}}\right)^n = \sum_{a+b+\cdots+k=n} (-1)^n \alpha_n^{1/2}\frac{\partial^n}{\partial x_1^a \partial x_2^b \cdots \partial x_{m_0}^k} \tag{7.45}$$

因此通过使用所有可能偏导数在内的稳定因子，可以得到式(7.39)形式的正则解。

式(7.40)至(7.42)代入式(7.16)且令 $\boldsymbol{\xi}$ 为 $\mathbf{x}_i$，则有

$$\sum_{n=0}^{\infty}(-1)^n \frac{\sigma_i^{2n}}{n!2^n}\nabla^{2n}\exp\left(-\frac{1}{2\sigma_i^2}\| \mathbf{x}-\mathbf{x}_i \|^2\right) = \delta(\mathbf{x}-\mathbf{x}_i) \tag{7.46}$$

利用式(7.40)定义的 Green 函数 $G(\mathbf{x},\mathbf{x}_i)$ 的特殊形式，可以将式(7.36)给出的正则化解写成多元高斯函数的线性叠加形式，如下所示：

$$F_\lambda(\mathbf{x}) = \sum_{i=1}^{N} w_i \exp\left(-\frac{1}{2\sigma_i^2}\| \mathbf{x}-\mathbf{x}_i \|^2\right) \tag{7.47}$$

其中线性权值 w_i 由式(7.23)定义。

在式(7.47)中，定义逼近函数 $F_\lambda(\mathbf{x})$ 的各高斯项的方差是不同的。为简化起见，通常认为在 $F(\mathbf{x})$ 中对所有的 i 都有 $\sigma_i=\sigma$。尽管这样设计的 RBF 网络是受到一定限制的一种，但其仍不失为一个通用逼近器（Park and Sandberg,1991）。

7.4　正则化网络

式(7.36)给出的正则化逼近函数 $F_\lambda(\mathbf{x})$ 关于中心在 $\mathbf{x}_i$ 的 Green 函数 $G(\mathbf{x},\mathbf{x}_i)$ 的展开形式，

体现了如图 7.2 所示网络结构的一个实现方法。基于明显的原因，这种网络结构被称为正则化网络（Poggio and Girosi，1990a）。该网络包括三层。第一层是由输入节点组成的，输入节点数目等于输入向量 x 的维数 m_0（即问题的独立变量数）。第二层是隐藏层，它是由直接与所有输入节点相连的非线性单元组成的。一个隐藏单元对应一个数据点 $\mathbf{x}_i, i = 1,2,\cdots,N$，其中 N 表示训练样本的长度。每个隐藏单元的激活函数由 Green 函数定义。因此第 i 个隐藏单元的输出是 $G(\mathbf{x},\mathbf{x}_i)$。输出层仅包含一个线性单元，它与所有隐藏单元相连。这里所谓的“线性”指的是网络的输出是隐藏单元输出的线性加权和。输出层的权值就是未知的展开系数，如式(7.23)所示，它是由 Green 函数 $G(\mathbf{x},\mathbf{x}_i)$和正则化参数 λ 决定。图 7.2 描绘一个单输出的正则化网络的结构图。显然，我们可以将其推广为包括任意期望输出数目的正则化网络。

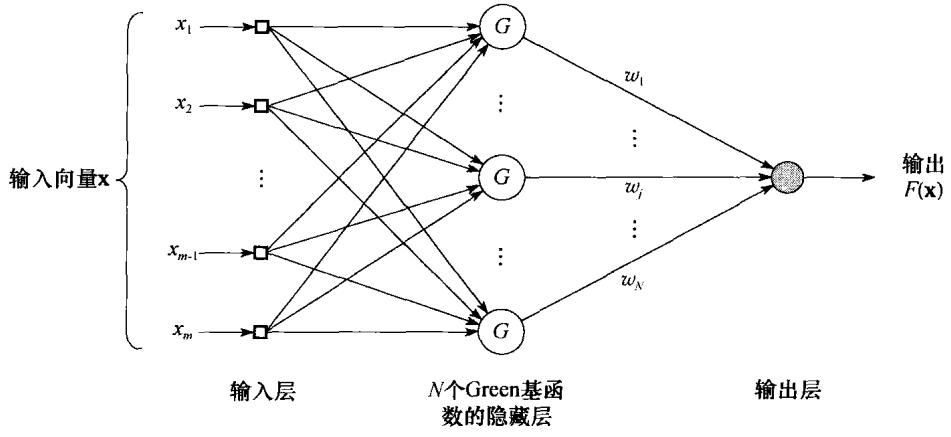


图 7.2　正则化网络

图 7.2 所示的正则化网络假设 Green 函数 $G(\mathbf{x},\mathbf{x}_i)$对所有的 i 都是正定的。假设上述条件成立，例如，Green 函数 $G(\mathbf{x},\mathbf{x}_i)$具有式(7.40)所示的高斯形式，则由该网络所得到的解在泛函$\mathscr{E}(F)$最小化的意义下将是一个“最佳”的内插解。而且，由逼近理论的观点，正则化网络具有如下三个如图 7.2 所示的径向基函数网络希望的性质（Poggio and Girosi，1990a）：

（ⅰ）正则化网络是一个通用逼近器，只要有足够多的隐藏单元，它可以以任意精度逼近定义在 R^{m_0} 的紧子集上的任何多元连续函数。

（ⅱ）由于正则化理论导出的逼近格式的未知系数是线性的，这样该网络具有最佳逼近性能。这说明给定一个未知的非线性函数 f，总可选择一组系数使得它对 f 的逼近优于所有其他可能选择。由正则化网络求得的解是最佳的。

7.5　广义径向基函数网络

由于输入向量 $\mathbf{x}_i$ 与 Green 函数 $G(\mathbf{x},\mathbf{x}_i)(i = 1,2,\cdots,N)$ 之间的一一对应的关系，有时候如果 N 太大了，实现它的计算量将大得惊人。特别是在计算网络的线性权值（即式(7.36)中的展开系数）时，要求计算一个 $N \times N$ 阶矩阵的逆，其计算量按 N 的多项式增长（大约为 N^3）。另外矩阵越大，其病态的可能性越高；一个矩阵的条件数被定义为该矩阵的最大特征值与其最小特征值的比值。要克服这些计算上的困难，我们通常要降低神经网络的复杂度，或者加大正则化参数值。

如图 7.3 所描绘的降低复杂度的 RBF 网络，在一个较低维数的空间中求一个次优解，以此来逼近式(7.36)所给出的正则化解。这可以通过变分问题中通称 Galerkin 方法的标准技术实现。根据这个技术，近似解 $F^*(\mathbf{x})$将在一个有限基上进行扩展，表示为

$$F^*(\mathbf{x}) = \sum_{i=1}^{m_1} w_i \varphi(\mathbf{x}, \mathbf{t}_i) \tag{7.48}$$

其中 $\{\varphi(\mathbf{x}, \mathbf{t}_i) \mid i = 1, 2, \cdots, m_1\}$ 是一组新的基函数，不失一般性我们假设它们线性独立（Poggio and Girosi，1990a）。典型情况下这组新的基函数的个数小于输入数据点的个数（即 $m_1 \leqslant N$），并且 w_i 组成一组新的权值集合。根据径向基函数，设

$$\varphi(\mathbf{x}, \mathbf{t}_i) = G(\| \mathbf{x} - \mathbf{t}_i \|), \qquad i = 1, 2, \cdots, m_1 \tag{7.49}$$

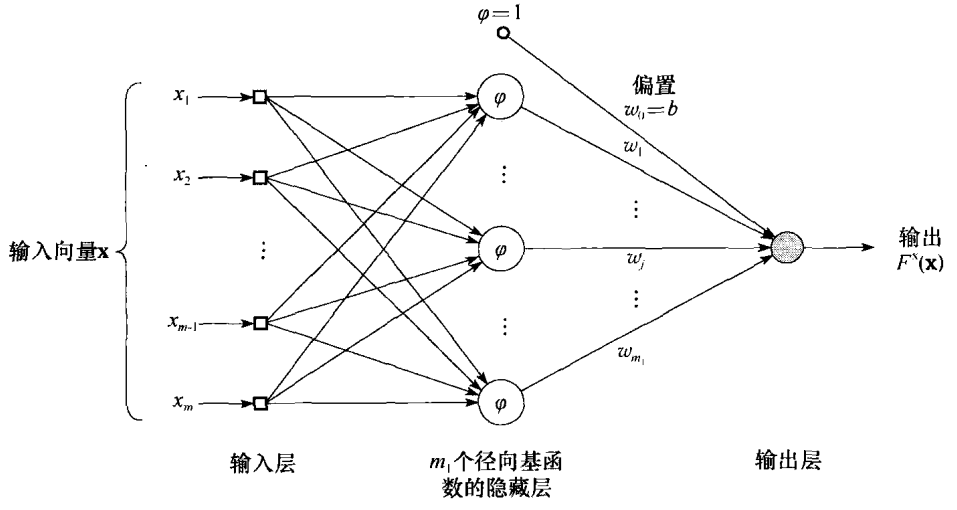


图 7.3　降低复杂度的径向基函数网络

基函数的这个特定选择是唯一的选择，只有当 $m_1 = N$，且

$$\mathbf{t}_i = \mathbf{x}_i, \qquad i = 1, 2, \cdots, N$$

时，其解与式(7.39)的正确解一致。因此将式(7.49)代入式(7.48)中，重新定义 $F^*(\mathbf{x})$ 为

$$F^*(\mathbf{x}) = \sum_{i=1}^{m_1} w_i G(\mathbf{x}, \mathbf{t}_i) = \sum_{i=1}^{m_1} w_i G(\| \mathbf{x} - \mathbf{t}_i \|) \tag{7.50}$$

对于给定的逼近函数 $F^*(\mathbf{x})$ 的（7.50）的展开形式，我们将要解决的问题是确定一组新的权值 $\{w_i\}_{i=1}^{m_1}$，使新的代价泛函 $\mathscr{E}(F^*)$ 最小化，新代价泛函由下式定义：

$$\mathscr{E}(F^*) = \sum_{i=1}^{N} \left(d_i - \sum_{j=1}^{m_1} w_j G(\| \mathbf{x}_i - \mathbf{t}_j \|) \right)^2 + \lambda \| \mathbf{D}F^* \|^2 \tag{7.51}$$

式(7.51)右边第一项可以写成欧几里得范数平方 $\| \mathbf{d} - \mathbf{G}\mathbf{w} \|^2$，其中

$$\mathbf{d} = [d_1, d_2, \cdots, d_N]^T \tag{7.52}$$

$$\mathbf{G} = \begin{bmatrix} G(\mathbf{x}_1, \mathbf{t}_1) & G(\mathbf{x}_1, \mathbf{t}_2) & \cdots & G(\mathbf{x}_1, \mathbf{t}_{m_1}) \\ G(\mathbf{x}_2, \mathbf{t}_1) & G(\mathbf{x}_2, \mathbf{t}_2) & \cdots & G(\mathbf{x}_2, \mathbf{t}_{m_1}) \\ \vdots & \vdots & & \vdots \\ G(\mathbf{x}_N, \mathbf{t}_1) & G(\mathbf{x}_N, \mathbf{t}_2) & \cdots & G(\mathbf{x}_N, \mathbf{t}_{m_1}) \end{bmatrix} \tag{7.53}$$

$$\mathbf{w} = [w_1, w_2, \cdots, w_{m_1}]^T \tag{7.54}$$

与前面一样，期望响应向量 $\mathbf{d}$ 是 N 维的。但是，Green 函数的矩阵 $\mathbf{G}$ 和权值向量 $\mathbf{w}$ 却有不同维数；矩阵 $\mathbf{G}$ 现在是 $N \times m_1$ 阶的，所以不再是对称的，而向量 $\mathbf{w}$ 是 $m_1 \times 1$ 的。由式(7.50)我们注意到，近似函数 F^* 是由稳定因子 $\mathbf{D}$ 决定的 Green 函数的线性组合。因此，可以将式(7.51)右边第二项写成：

$$\| \mathbf{D}F^* \|^2 = \langle \mathbf{D}F^*, \mathbf{D}F^* \rangle_{\mathscr{H}} = \left[\sum_{i=1}^{m_1} w_i G(\mathbf{x}, \mathbf{t}_i), \tilde{\mathbf{D}}\mathbf{D} \sum_{i=1}^{m_1} w_i G(\mathbf{x}; \mathbf{t}_i) \right]_{\mathscr{H}}$$

$$=\left[\sum_{i=1}^{m_1} w_i G(\mathbf{x},\mathbf{t}_i), \sum_{i=1}^{m_1} w_i \delta_{\mathbf{t}_i}\right]_{\mathcal{H}} = \sum_{j=1}^{m_1}\sum_{i=1}^{m_1} w_j w_i G(\mathbf{t}_j,\mathbf{t}_i) = \mathbf{w}^T\mathbf{G}_0\mathbf{w} \tag{7.55}$$

其中第二个和第三个相等项分别利用伴随算子的定义和式(7.16)。矩阵 $\mathbf{G}_0$ 是一个 $m_1 \times m_1$ 阶的对称阵，定义为

$$\mathbf{G}_0 = \begin{bmatrix} G(\mathbf{t}_1,\mathbf{t}_1) & G(\mathbf{t}_1,\mathbf{t}_2) & \cdots & G(\mathbf{t}_1,\mathbf{t}_{m_1}) \\ G(\mathbf{t}_2,\mathbf{t}_1) & G(\mathbf{t}_2,\mathbf{t}_2) & \cdots & G(\mathbf{t}_2,\mathbf{t}_{m_1}) \\ \vdots & \vdots & & \vdots \\ G(\mathbf{t}_{m_1},\mathbf{t}_1) & G(\mathbf{t}_{m_1},\mathbf{t}_2) & \cdots & G(\mathbf{t}_{m_1},\mathbf{t}_{m_1}) \end{bmatrix} \tag{7.56}$$

以权值向量 $\mathbf{w}$ 为变量求式(7.51)的最小值，可以得到以下结果（参看习题 7.4）：

$$(\mathbf{G}^T\mathbf{G} + \lambda\mathbf{G}_0)\,\hat{\mathbf{w}} = \mathbf{G}^T\mathbf{d}$$

在等式中解出权值向量$\hat{\mathbf{w}}$，得到：

$$\hat{\mathbf{w}} = (\mathbf{G}^T\mathbf{G} + \lambda\mathbf{G}_0)^{-1}\mathbf{G}^T\mathbf{d} \tag{7.57}$$

当正则化参数 λ 趋近零时，权值向量$\hat{\mathbf{w}}$趋于一个超定的最小二乘数据-拟合问题（因为 $m_1 < N$）的伪逆（最小范数）解，表示为：

$$\mathbf{w} = \mathbf{G}^+\mathbf{d}, \qquad \lambda = 0 \tag{7.58}$$

其中 G^+ 是矩阵 G 的伪逆（colub and Van Loan，1996）；即

$$\mathbf{G}^+ = (\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T \tag{7.59}$$

加权范数

式(7.50)中的范数通常指的是欧几里得范数。然而，当输入向量 $\mathbf{x}$ 的分量属于不同的类时，将其视为一般的*加权范数*会更合理，加权范数的平方形式由

$$\|\mathbf{x}\|_C^2 = (\mathbf{C}\mathbf{x})^T(\mathbf{C}\mathbf{x}) = \mathbf{x}^T\mathbf{C}^T\mathbf{C}\mathbf{x} \tag{7.60}$$

定义，其中 $\mathbf{C}$ 是一个 $m_0 \times m_0$ *加权矩阵*，m_0 是输入向量 $\mathbf{x}$ 的维数。

利用加权范数的定义，我们可以将式(7.50)中正则化问题的近似解写成如下更一般的形式(Lowe，1989；Poggio and Girosi，1990a)：

$$F^*(\mathbf{x}) = \sum_{i=1}^{m_1} w_i G(\|\mathbf{x} - \mathbf{t}_i\|_C) \tag{7.61}$$

引入加权范数可以用两种方式解释。我们可以简单地将其视为对原始输入空间做一个仿射变换。原则上这种变换并不会降低原来不加权的结果，因为原来不加权的范数实际上对应于一个单位矩阵的加权范数。另一方面，加权范数可以看作直接从式(7.44)定义的 m_1 维 Laplace 伪微分算子 $\mathbf{D}$ 的少许推广。使用加权范数的合理性在高斯径向基函数背景下可以解释如下。一个以 $\mathbf{t}_i$ 为中心和具有范数加权矩阵 $\mathbf{C}$ 的高斯径向基函数 $G(\|\mathbf{x}-\mathbf{t}_i\|_C)$可写成

$$G(\|\mathbf{x}-\mathbf{t}_i\|_C = \exp[-(\mathbf{x}-\mathbf{t}_i)^T\mathbf{C}^T\mathbf{C}(\mathbf{x}-\mathbf{t}_i)] = \exp\left[-\frac{1}{2}(\mathbf{x}-\mathbf{t}_i)^T\Sigma^{-1}(\mathbf{x}-\mathbf{t}_i)\right] \tag{7.62}$$

其中逆矩阵Σ^{-1}定义为

$$\frac{1}{2}\Sigma^{-1} = \mathbf{C}^T\mathbf{C}$$

式(7.62)中的广义多维高斯分布有一个指数等于 Mahalanobis 距离，见引言章节中的式(27)。因此，由式(7.62)所定义的核称为 Mahalanobis 核。这个核已在第 6 章的习题 6.10 中有所讨论。

式(7.51)中逼近问题的解为具有如图 7.3 结构的广义径向基函数网络提供了一个框架。在这种网络中，输出单元上有一个偏置（即独立于数据的变量）。要做到这一点可以简单地将输出层的一个线性权值置为偏置值，同时将与该权值相对应的径向基函数视为一个等于 +1 的

常量。

从结构上看，图 7.3 所示的广义 RBF 网络与图 7.2 所示的正则化 RBF 网络相似。但它们在以下两个重要的方面有所不同：

1. 图 7.3 所示的广义 RBF 网络隐藏层的节点数为 m_1，通常 m_1 总是小于用于训练的样本数 N。另一方面，图 7.2 所示的正则化 RBF 网络的隐藏单元数恰为 N。

2. 在图 7.3 的广义 RBF 网络中，与输出层相连的线性权值向量，以及与隐藏层相连的径向基函数的中心和范数加权矩阵，均为待学习的未知参数。而图 7.2 的正则化 RBF 网络隐藏层的激活函数是已知的，它定义为一组以训练样本点为中心的 Green 函数；输出层的权值向量是网络的唯一未知参数。

7.6　再论正则化最小二乘估计

我们一开始在第 2 章中学习了最小二乘估计。然后在第 5 章中使用它计算一个次最优径向基函数网络的输出层。在本节，我们再次讨论这个相对简单但很有效的估计方法。这里，我们注意两点：第一，我们要指出式(7.57)的公式包括正则化最小二乘估计，且后者是前者的一个特例。第二，我们要指出，与其他核方法一样，正则化最小二乘估计受到表示理论的控制。

把最小二乘估计看作式(7.57)的一个特例

对于给定的训练样本 $\{\mathbf{x}_i, d_i\}_{i=1}^N$，最小二乘估计的正则化代价函数由下式定义（见第 2 章）：

$$\mathscr{E}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}(d_i - \mathbf{w}^T\mathbf{x}_i)^2 + \frac{1}{2}\lambda \| \mathbf{w} \|^2 \tag{7.63}$$

其中权值向量 $\mathbf{w}$ 通过训练步长确定，是一个正则化参数。逼近此式和式(7.4)中的代价函数，我们可以发现正则化项以 $\mathbf{w}$ 的形式简单地定义：

$$\| \mathbf{D}F \|^2 = \| \mathbf{w} \|^2 = \mathbf{w}^T\mathbf{w}$$

根据上式，我们可以立即设立式(7.57)中的对称矩阵 G_0 为单位阵。相应地，式(7.57)之前的项缩减为：

$$(\mathbf{G}^T\mathbf{G} + \lambda\mathbf{I})\,\hat{\mathbf{w}} = \mathbf{G}^T\mathbf{d}$$

接下来，注意到因为最小二乘估计是线性的，且缺失隐藏层，我们可以把式(7.53)中的剩余矩阵 $\mathbf{G}$ 的转置表示为：

$$\mathbf{G}^T = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N] \tag{7.64}$$

然后，对 $\mathbf{G}^T$ 使用此表达式，对关于权值向量$\hat{\mathbf{w}}$的式(7.57)中的正则化解的预期响应 d 使用式(7.52)中的表达式，我们得到（经过一些代数操作）：

$$\hat{\mathbf{w}} = (R_{xx} + \lambda\mathbf{I})^{-1}\mathbf{r}_{dx} \tag{7.65}$$

其中

$$\mathbf{R}_{xx} = \sum_{i=1}^{N}\sum_{j=1}^{N}\mathbf{x}_i\mathbf{x}_j^T$$

且

$$\mathbf{r}_{dx} = \sum_{i=1}^{N}\mathbf{x}_i d_i$$

式(7.65)是在式(2.29)中定义的用于最大后验（MAP）估计的公式的重复。如前所述，此式同样可以用于正则化最小二乘估计。

对相关矩阵 $\mathbf{R}_{xx}$ 和互相关向量 $\mathbf{r}_{dx}$ 使用此表达式，我们以训练样本 $\{\mathbf{x}_i, d_i\}_{i=1}^N$ 的形式重申式(7.65)中的公式：

$$\hat{\mathbf{w}} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{d} \tag{7.66}$$

其中 $\mathbf{X}$ 是输入数据矩阵：

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1M} \\ x_{21} & x_{22} & \cdots & x_{2M} \\ \vdots & \vdots & & \vdots \\ x_{N1} & x_{N2} & \cdots & x_{NM} \end{bmatrix} \tag{7.67}$$

其中下标 N 是训练样本的个数，下标 M 是权值向量$\hat{\mathbf{w}}$的维数。向量 $\mathbf{d}$ 是预期响应向量，由式(7.52)定义；在此，为了方便起见，我们重新写作：

$$\mathbf{d} = [d_1, d_2, \cdots, d_N]^T$$

把最小二乘估计看作表示定理的形式

接下来，把最小二乘估计看成一个“核机器”，我们把它的核表示成内积的形式：

$$k(\mathbf{x}, \mathbf{x}_i) = \langle \mathbf{x}, \mathbf{x}_i \rangle = \mathbf{x}^T\mathbf{x}_i, \qquad i = 1, 2, \cdots, N \tag{7.68}$$

下面引入第 6 章讨论的表示定理，可以通过如下的正则化最小二乘估计表示逼近函数：

$$F_\lambda(\mathbf{x}) = \sum_{i=1}^{N} a_i k(\mathbf{x}, \mathbf{x}_i) \tag{7.69}$$

其中表示系数 $\{a_i\}_{i=1}^N$由训练样本 $\{\mathbf{x}_i, d_i\}_{i=1}^N$所唯一地确定；问题是如何确定？

要解决这个问题，首先使用如下等式：

$$\mathbf{X}^T(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I}_N)^{-1}\mathbf{d} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_M)^{-1}\mathbf{X}^T\mathbf{d} \tag{7.70}$$

其中 $\mathbf{X}$ 是一个 $N \times M$ 的矩阵，$\mathbf{d}$ 是一个 $N \times 1$ 的预期响应向量，它是正则化参数，$\mathbf{I}_N$ 和 $\mathbf{I}_M$ 分别是N 维和M 维的单位矩阵。其中 M 是权向量 $\mathbf{w}$ 的维数。对于式(7.70)中矩阵等式的证明可见习题 7.11。此等式的右端被认为是最优化权值向量$\hat{\mathbf{w}}$的公式；见式(7.66)。使用式(7.70)中的等式，我们可以通过如下的正则化最小二乘估计，以权值向量和输入向量 $\mathbf{x}$ 的形式来表示逼近函数：

$$F_\lambda(\mathbf{x}) = \mathbf{x}^T\hat{\mathbf{w}} = \mathbf{x}^T\mathbf{X}^T(\mathbf{X}\mathbf{X}^T + \lambda\mathbf{I}_N)^{-1}\mathbf{d} \tag{7.71}$$

此式可由内积的形式表示：

$$F_\lambda(\mathbf{x}) = \mathbf{k}^T(\mathbf{x})\mathbf{a} = \mathbf{a}^T\mathbf{k}(\mathbf{x}) \tag{7.72}$$

此式是式(7.69)的表示理论的矩阵形式。由此得出：

1. 核的行向量以输入向量 $\mathbf{x}$ 和数据矩阵 $\mathbf{X}$ 的形式定义，如下所示：

$$\mathbf{k}^T(\mathbf{x}) = [k(\mathbf{x}, \mathbf{x}_1), k(\mathbf{x}, \mathbf{x}_2), \cdots, k(\mathbf{x}, \mathbf{x}_N)] = \mathbf{x}^T\mathbf{X}^T = (\mathbf{X}\mathbf{x})^T \tag{7.73}$$

此向量是一个 1 乘以 N 的行向量。

2. 表示系数向量 $\mathbf{a}$ 由估计算子中的 $N \times N$ 的核矩阵或 Gram 矩阵 $\mathbf{K}$、正则化参数和预期响应向量 $\mathbf{d}$ 的形式定义，如下：

$$\mathbf{a} = [a_1, a_2, \cdots, a_N]^T = (\mathbf{K} + \lambda\mathbf{I}_N)^{-1}\mathbf{d} \tag{7.74}$$

其中

$$\mathbf{K} = \mathbf{X}\mathbf{X}^T = \begin{bmatrix} \mathbf{x}_1^T\mathbf{x}_1 & \mathbf{x}_1^T\mathbf{x}_2 & \cdots & \mathbf{x}_1^T\mathbf{x}_N \\ \mathbf{x}_2^T\mathbf{x}_1 & \mathbf{x}_2^T\mathbf{x}_2 & \cdots & \mathbf{x}_2^T\mathbf{x}_1 \\ \vdots & \vdots & & \vdots \\ \mathbf{x}_N^T\mathbf{x}_1 & \mathbf{x}_N^T\mathbf{x}_2 & \cdots & \mathbf{x}_N^T\mathbf{x}_N \end{bmatrix} \tag{7.75}$$

描述正则最小二乘估计的两个等价方式

由本章中所谈论的内容，我们发现，事实上有两种描述由正则化最小二乘估计实现的逼近函数 $F_\lambda(\mathbf{x})$：

1. 式(7.71)中的公式，由给定输入向量 $\mathbf{x}$ 的权值向量$\hat{\mathbf{w}}$所定义。基本上说，这个公式可以追溯到第 2 章中讨论过的用于最小二乘估计的规范等式。

2. 式(7.72)中的公式，由估计算子的核的形式定义。第二个公式来自于第 6 章中的表示理论。这个公式的重要实质在于其不需要计算 RLS 算法中的权值向量。这也是第 6 章中所讨论的核方法的本质。

对正则化最小二乘估计的第一个观点，以规范等式的形式给出，在统计学中是常见的。然而，以表示理论（在核学习中常见）给出的第二个等式是新的。

7.7　对正则化的附加要点

基于高斯的径向基函数网络的一个属性就是其本身是 Tikhonov 正则化理论的严格应用。这在 7.4 节和 7.5 节中已证明。而如 7.6 节所示，同样的表示可适用于最小二乘估计。

本节的目标是把最小二乘估计中所学的知识，延伸到使用 Tikhonov 正则化理论较为困难的情况。

回归

式(7.63)可以重写成如下形式：

$$\underbrace{\mathscr{C}(\mathbf{w})}_{\substack{\text{正则化}\\\text{代价函数}}} = \underbrace{\frac{1}{2}\sum_{i=1}^{N}(d_i-\mathbf{w}^T\mathbf{x}_i)^2}_{\text{经验风险}} + \underbrace{\frac{1}{2}\lambda\|\mathbf{w}\|^2}_{\text{正则化项}} \tag{7.76}$$

从回归的角度上看，项$\frac{1}{2}\|\mathbf{w}\|^2$有一个特定的直观的作用。从几何上说，最小化代价函数$\mathscr{C}(\mathbf{w})$过程中，包含正则化项$\frac{1}{2}\|\mathbf{w}\|^2$有利于找到带有好的逼近属性的平坦的函数。事实上，这也是 4.14 节中所提到的目标，我们提出最小化代价函数：

$$\underbrace{\mathscr{C}(\mathbf{w})}_{\substack{\text{正则化}\\\text{代价函数}}} = \underbrace{\frac{1}{2}\sum_{i=1}^{N}(d_i-F(\mathbf{x}_i,\mathbf{w}))^2}_{\text{经验风险}} + \underbrace{\frac{1}{2}\lambda\|\mathbf{w}\|^2}_{\text{正则化项}}$$

此最小化代价函数可作为一个用于函数逼近的多层感知器的正则化的可行方法。此方法的缺点是在数学上很难把 Tikhonov 正则化理论应用于多层感知器。不像径向基函数网络，多层感知器的可调整的突触权值在隐藏层和输出层中分布。从实用的角度上看，使用正则化项$\frac{1}{2}\|w\|^2$是一个理想的选择。

最大似然估计

从第 2 章处理的最小二乘方法和贝叶斯估计中，我们发现最大化后验参数估计的目标函数，作用于高斯环境，可以由如下公式来表示（见式(2.22)和式(2.28)）：

$$\underbrace{L(\mathbf{w})}_{\substack{\text{对数}\\\text{较晚的}}} = \underbrace{-\frac{1}{2}\sum_{i=1}^{N}(d_i-\mathbf{w}^T\mathbf{x}_i)^2}_{\text{对数似然比较}} \underbrace{-\frac{1}{2}\lambda\|\mathbf{w}\|^2}_{\text{对数优先的}} \tag{7.77}$$

因此，我们可以把项$\frac{1}{2}\|w\|^2$看作一个关于极大后验参数估计中潜在结构的先验信息。式(7.76)和式(7.77)这两个等式，分别定义了$\mathscr{C}(\mathbf{w})$和$L(\mathbf{w})$，有相同的数学结构，除了对数后验概率$L(\mathbf{w})$是正则化代价函数$\mathscr{C}(\mathbf{w})$的负数。因此，正则化项和先验信息项在最小二乘估计，

或高斯环境下的最大似然估计中起到相同的作用。

在最小二乘估计中推广此观察，我们可以推导出以正则化极大似然估计作为目标函数的估计，其表示式如下：

$$\underbrace{L(\mathbf{w})}_{\substack{\text{正则化}\\\text{对数似}\\\text{然概率}}} = \underbrace{\log l(\mathbf{w})}_{\substack{\text{对数似}\\\text{然概率}}} - \underbrace{\frac{1}{2}\lambda \| \mathbf{w} \|^2}_{\substack{\text{正则化项}\\\text{(惩罚项)}}}$$

其中，$\mathbf{w}$ 是待优化的参数向量。基本上说，当很难获得一个用于估计未知参数向量 $\mathbf{w}$ 的最大似然估计算法的先验知识时，从似然概率函数 $l(\mathbf{w})$ 中减去惩罚项 $\frac{\lambda}{2}\| \mathbf{w} \|^2$ 可能会对稳定最大似然估计过程提供一个理想的选择。

7.8　正则化参数估计

正则化参数 λ 在径向基函数网络，最小二乘估计和支持向量机的正则化理论中起着核心的作用。为了更好利用这个理论，我们需要一个估计 λ 的相当于原理性的方法。

要形成我们的思想，先考虑一个非线性回归问题，它由一个模型描述，其中与第 i 时间步的输入向量 $\mathbf{x}_i$ 相对应的可观测输出 y_i 定义为：

$$d_i = f(\mathbf{x}_i) + \varepsilon_i, \qquad i = 1,2,\cdots,N \tag{7.78}$$

此处 $f(\mathbf{x}_i)$ 是一条“光滑曲线”，ε_i 是一个均值为零和方差如下的白噪声过程的采样。即

$$\mathbb{E}[\varepsilon_i \varepsilon_k] = \begin{cases} \sigma^2, & \text{当 } k = i \\ 0, & \text{否则} \end{cases} \tag{7.79}$$

问题是在给定一组训练样本 $\{\mathbf{x}_i, y_i\}_{i=1}^N$ 的条件下，重建该模型的固有函数 $f(\mathbf{x}_i)$。

令 $F_\lambda(\mathbf{x})$ 为 $f(\mathbf{x})$ 相对于某个正则化参数 λ 的正则化估计。即 $F_\lambda(\mathbf{x})$ 为使表示非线性回归问题的 Tikhonov 泛函（见式(7.4)）达到最小的最小化函数：

$$\mathscr{E}(F) = \frac{1}{2}\sum_{i=1}^{N}[d_i - F(\mathbf{x}_i)]^2 + \frac{\lambda}{2}\| \mathbf{D}F(\mathbf{x}) \|^2$$

选择一个合适的 λ 值并不是一件简单的事情，它需要在下面两种矛盾的情况之间加以权衡：

- 由 $\| \mathbf{D}F(\mathbf{x}) \|^2$ 项来度量解的粗糙度。
- 由 $\sum_{i=1}^{N}[d_i - F(\mathbf{x}_i)]^2$ 项来度量数据的失真度。

这一节的主题是讨论如何选择好的正则化参数 λ。

均方误差

令 $R(\lambda)$ 表示模型的回归函数 $f(\mathbf{x})$ 和表示在正则化参数 λ 某一值下的解的逼近函数 $F_\lambda(\mathbf{x})$ 之间在整个给定集合上的均方误差。即：

$$R(\lambda) = \frac{1}{N}\sum_{i=1}^{N}[f(\mathbf{x}_i) - F_\lambda(\mathbf{x}_i)]^2 \tag{7.80}$$

所谓最佳 λ 指的是使 $R(\lambda)$ 取最小的 λ 值。

将 $F_\lambda(\mathbf{x}_k)$ 表示为给定的一组可观察值的线性组合：

$$F_\lambda(\mathbf{x}_k) = \sum_{i=1}^{N} a_{ki}(\lambda) d_i \tag{7.81}$$

用等价的矩阵形式写成：

$$\mathbf{F}_\lambda = \mathbf{A}(\lambda)\mathbf{d} \tag{7.82}$$

其中 d 是预期响应向量（即回归模型中的响应向量），

$$\mathbf{F}_\lambda = [F_\lambda(\mathbf{x}_1), F_\lambda(\mathbf{x}_2), \cdots, F_\lambda(\mathbf{x}_N)]^T$$

且

$$\mathbf{A}(\lambda) = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NN} \end{bmatrix} \tag{7.83}$$

其中 $N \times N$ 矩阵 $\mathbf{A}(\lambda)$称为*影响矩阵*。

用上述的矩阵符号，我们可将式(7.80)重新写成：

$$R(\lambda) = \frac{1}{N} \| \mathbf{f} - \mathbf{F}_\lambda \|^2 = \frac{1}{N} \| \mathbf{f} - \mathbf{A}(\lambda)\mathbf{d} \|^2 \tag{7.84}$$

其中 $N \times 1$ 的向量 $\mathbf{f}$ 为：

$$\mathbf{f} = [f(\mathbf{x}_1), f(\mathbf{x}_2), \cdots, f(\mathbf{x}_N)]^T$$

我们可以进一步将式(7.78)也写成矩阵形式：

$$\mathbf{d} = \mathbf{f} + \boldsymbol{\varepsilon} \tag{7.85}$$

其中：

$$\boldsymbol{\varepsilon} = [\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_N]^T$$

因此，将式(7.85)代入式(7.84)中并展开，可得

$$\begin{aligned} R(\lambda) &= \frac{1}{N} \| (\mathbf{I} - \mathbf{A}(\lambda))\mathbf{f} - \mathbf{A}(\lambda)\boldsymbol{\varepsilon} \|^2 \\ &= \frac{1}{N} \| (\mathbf{I} - \mathbf{A}(\lambda))\mathbf{f} \|^2 - \frac{2}{N}\boldsymbol{\varepsilon}^T\mathbf{A}^T(\lambda)(\mathbf{I} - \mathbf{A}(\lambda))\mathbf{f} + \frac{1}{N} \| \mathbf{A}(\lambda)\boldsymbol{\varepsilon} \|^2 \end{aligned} \tag{7.86}$$

其中 $\mathbf{I}$ 是一个 $N \times N$ 的单位矩阵。求 $R(\lambda)$的期望值，需要注意下述几点：

1. 式(7.86)的右边第一项是一个常数，因此它不受期望算子的影响。
2. 由式(7.86)可知，第二项的期望为零。
3. 标量 $\| \mathbf{A}(\lambda)\boldsymbol{\varepsilon} \|^2$ 的期望为：

$$\begin{aligned} \mathbb{E}[\| \mathbf{A}(\lambda)\boldsymbol{\varepsilon} \|^2] &= \mathbb{E}[\boldsymbol{\varepsilon}^T\mathbf{A}^T(\lambda)\mathbf{A}(\lambda)\boldsymbol{\varepsilon}] \\ &= \mathrm{tr}(\mathbb{E}[\boldsymbol{\varepsilon}^T\mathbf{A}^T(\lambda)\mathbf{A}(\lambda)\boldsymbol{\varepsilon}]) = \mathbb{E}[\mathrm{tr}(\boldsymbol{\varepsilon}^T\mathbf{A}^T(\lambda)\mathbf{A}(\lambda)\boldsymbol{\varepsilon})] \end{aligned} \tag{7.87}$$

其中我们首先用到了标量的迹等于标量本身的性质，然后交换了期望运算和求迹运算的次序。

4. 接下来我们利用矩阵代数中的如下规则：给定两个具有相容维数的矩阵 $\mathbf{B}$ 和 $\mathbf{C}$，$\mathbf{BC}$ 的迹等于 $\mathbf{CB}$ 的迹。令 $\mathbf{B}=\boldsymbol{\varepsilon}^T$，$\mathbf{C}=\mathbf{A}^T(\lambda)\mathbf{A}(\lambda)\boldsymbol{\varepsilon}$，则式(7.87)可以写成等价形式：

$$\mathbb{E}[\| \mathbf{A}(\lambda)\mathbf{f} \|^2] = \mathbb{E}\{\mathrm{tr}[\mathbf{A}^T(\lambda)\mathbf{A}(\lambda)\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}^T] = \sigma^2\mathrm{tr}[\mathbf{A}^T(\lambda)\mathbf{A}(\lambda)] \tag{7.88}$$

上式中的最后一行根据式(7.79)可得。最后注意到 $\mathbf{A}^T(\lambda)\mathbf{A}(\lambda)$的迹等于 $\mathbf{A}^2(\lambda)$的迹，则

$$\mathbb{E}[\| \mathbf{A}(\lambda)\mathbf{f} \|^2] = \sigma^2\mathrm{tr}[\mathbf{A}^2(\lambda)] \tag{7.89}$$

将这三项结果结合起来，$R(\lambda)$期望值可表示为：

$$\mathbb{E}[R(\lambda)] = \frac{1}{N} \| \mathbf{I} - \mathbf{A}(\lambda)\mathbf{f} \|^2 + \frac{\sigma^2}{N}\mathrm{tr}[\mathbf{A}^2(\lambda)] \tag{7.90}$$

但是，一个给定数据集的均方误差 $R(\lambda)$在实际中并不好用，因为式(7.90)中需要回归函数 $f(\mathbf{x})$的知识，它是有待重建的函数。我们引入如下定义作为 $R(\lambda)$的估计（Craven and Wahba，1979）：

$$\hat{R}(\lambda) = \frac{1}{N} \| (\mathbf{I} - \mathbf{A}(\lambda))\mathbf{d} \|^2 + \frac{\sigma^2}{N}\mathrm{tr}[\mathbf{A}^2(\lambda)] - \frac{\sigma^2}{N}\mathrm{tr}[(\mathbf{I} - \mathbf{A}(\lambda))^2] \tag{7.91}$$

它是无偏估计，因此（按照导出式(7.90)所述的相似过程）我们可证明：

$$\mathbb{E}[\hat{R}(\lambda)] = \mathbb{E}[R(\lambda)] \tag{7.92}$$

所以，使估计$\hat{R}(\lambda)$最小的λ值可以作为正则化参数λ的一个好的选择。

广义交叉验证

使用估计$\hat{R}(\lambda)$的一个缺陷是它要求知道噪声的方差σ^2。在实际情况中，σ^2通常是未知的。要处理这种情况，下面我们将介绍广义交叉验证，它最早是由 Craven and Wahba(1979) 提出的。

我们从修改通常的交叉验证的留一形式（在第 4 章描述）开始来处理这个问题。具体地说，令$F_{\lambda}^{[k]}(\mathbf{x})$为使泛函最小化的函数：

$$\mathscr{E}_{\text{modified}}(F) = \frac{1}{2}\sum_{\substack{i=1\\ i\neq k}}^{N}[d_i - F_{\lambda}(\mathbf{x}_i)]^2 + \frac{\lambda}{2}\|\mathbf{D}F(\mathbf{x})\|^2 \tag{7.93}$$

其中标准误差项中省略了第k项$[d_k - F_{\lambda}(\mathbf{x}_k)]$。通过留出这一项，我们将用$F_{\lambda}^{[k]}(\mathbf{x})$预报缺损数据点$d_k$的能力来衡量参数$\lambda$的好坏。因此，引入性能度量：

$$V_0(\lambda) = \frac{1}{N}\sum_{k=1}^{N}[d_k - F_{\lambda}^{[k]}(\mathbf{x}_k)]^2 \tag{7.94}$$

$V_0(\lambda)$仅依赖于数据点本身。这样λ的普通交叉验证估计即为使$V_0(\lambda)$最小化的函数（Wahba，1990）。

$F_{\lambda}^{[k]}(\mathbf{x}_k)$的一个有用的性质是如果用预测$F_{\lambda}^{[k]}(\mathbf{x}_k)$来代替数据点$d_k$的值，使用数据点$d_1, d_2, \cdots, d_{k-1}, d_k, d_{k+1}, \cdots, d_N$使式(7.4)的原始 Tikhonov 泛函$\mathscr{E}(F)$最小，则$F_{\lambda}^{[k]}(\mathbf{x}_k)$就是所求的解。对于每一个输入向量$\mathbf{x}$，该性质使$\mathscr{E}(F)$的最小化函数$F_{\lambda}(\mathbf{x})$线性依赖于$d_k$，这使我们有：

$$F_{\lambda}^{[k]}(\mathbf{x}_k) = F_{\lambda}(\mathbf{x}_k) + (F_{\lambda}^{[k]}(\mathbf{x}_k) - d_k)\frac{\partial F_{\lambda}(\mathbf{x}_k)}{\partial d_k} \tag{7.95}$$

由式(7.81)所定义的影响矩阵$\mathbf{A}(\lambda)$的分量，我们很容易看出：

$$\frac{\partial F_{\lambda}(\mathbf{x}_k)}{\partial d_k} = a_{kk}(\lambda) \tag{7.96}$$

其中$a_{kk}(\lambda)$是影响矩阵$\mathbf{A}(\lambda)$对角线上的第k个元素。将式(7.96)代入式(7.95)中并解$F_{\lambda}^{[k]}(\mathbf{x}_k)$的方程，可得

$$F_{\lambda}^{[k]}(\mathbf{x}_k) = \frac{F_{\lambda}(\mathbf{x}_k) - a_{kk}(\lambda)d_k}{1 - a_{kk}(\lambda)} = \frac{F_{\lambda}(\mathbf{x}_k) - d_k}{1 - a_{kk}(\lambda)} + d_k \tag{7.97}$$

将式(7.97)代入式(7.94)中，我们就可重新定义：

$$V_0(\lambda) = \frac{1}{N}\sum_{k=1}^{N}\left(\frac{d_k - F_{\lambda}(\mathbf{x}_k)}{1 - a_{kk}(\lambda)}\right)^2 \tag{7.98}$$

但是，对于不同的k，$a_{kk}(\lambda)$的值是不同的，这说明不同的数据点在$V_0(\lambda)$中具有不同的作用。为了避免通常的交叉验证的这一特性，Craven and Wahba(1979) 通过坐标旋转引入了广义交叉验证（generalized cross-validation，GCV）。特别地，式(7.98)中的$V_0(\lambda)$改变为：

$$V(\lambda) = \frac{1}{N}\sum_{k=1}^{N}\omega_k\left(\frac{d_k - F_{\lambda}(\mathbf{x}_k)}{1 - a_{kk}(\lambda)}\right)^2 \tag{7.99}$$

其中，权系数ω_k由下式所定义：

$$\omega_k = \left(\frac{1 - a_{kk}(\lambda)}{\frac{1}{N}\text{tr}[\mathbf{I} - \mathbf{A}(\lambda)]}\right)^2 \tag{7.100}$$

这样广义交叉验证函数$V(\lambda)$就变为：

$$V(\lambda)=\frac{\frac{1}{N}\sum_{k=1}^{N}(d_k-F_\lambda(\mathbf{x}_k))^2}{\left(\frac{1}{N}\mathrm{tr}[\mathbf{I}-\mathbf{A}(\lambda)]\right)^2} \tag{7.101}$$

最后，将式(7.81)代入式(7.101)，可得：

$$V(\lambda)=\frac{\frac{1}{N}\|(\mathbf{I}-\mathbf{A}(\lambda))\mathbf{d}\|^2}{\left(\frac{1}{N}\mathrm{tr}[\mathbf{I}-\mathbf{A}(\lambda)]\right)^2} \tag{7.102}$$

上式在计算上仅依赖于和数据有关的量。

广义交叉验证函数 V(λ)的最优性

广义交叉验证的期望无效度可定义为：

$$I^*=\frac{\mathbb{E}[R(\lambda)]}{\min\limits_{\lambda}\mathbb{E}[R(\lambda)]} \tag{7.103}$$

其中 $R(\lambda)$是由式(7.80)定义的数据集的均方误差。自然，I^*的渐近值满足条件：

$$\lim_{N\to\infty} I^*=1 \tag{7.104}$$

换句话说，对于一个很大的 N，使 $V(\lambda)$最小的 λ，同时也使 $R(\lambda)$接近最小的可能值，这使得 $V(\lambda)$成为一个很好的估计 λ 的工具。

评论小结

一般的想法是选择一个使在整个数据集上的均方误差 $R(\lambda)$最小化的 λ 值。但是这一想法不能直接实现，因为 $R(\lambda)$中包含有未知的回归函数 $f(\mathbf{x})$。因此，在实际中我们就要分两种可能性来处理：

- 如果噪声方差 σ^2 已知，就选择使式(7.91)的估计$\hat{R}(\lambda)$最小化的 λ 作为最佳值，这里的最佳是指它也使 $R(\lambda)$最小化。
- 如果 σ^2 未知，我们可以选择使得式(7.102)的广义交叉验证函数 $V(\lambda)$最小化的 λ 作为好的选择，当 $N\to\infty$时，这个 λ 可以使期望均方误差逼近其最小可能值。

值得注意的是，使用广义交叉验证方法估计 λ 所依赖的理论是渐近的。只有当所得的数据集大到能使信号和噪声相分离的程度，这种方法才能得到令人满意的结果。

在实际使用中，广义交叉验证方法对于非齐次方差和非高斯噪声情况，表现出很强的鲁棒性（Wahba，1990）。但是如果噪声过程是高度相关的，这种方法往往得不到满意的正则化参数 λ 的估计。

需要说明的是广义交叉验证函数的计算问题。对于一个给定的正则化参数的试验值 λ，求式(7.102)中分母 $[\mathrm{tr}[\mathbf{I}-\mathbf{A}(\lambda)]/N]^2$ 将是计算 $V(\lambda)$中计算量最大的部分。在 Wahba 等(1995)中描述的“随机化迹方法”可以用于计算 $\mathrm{tr}[\mathbf{A}(\lambda)]$；这种方法可用于超大规模的系统。

本节关注交叉验证，其用于监督学习中估计正则化参数。当我们在 7.12 节中讨论半监督学习时，会发现需要面对两个不同的正则化参数。这对此处的交叉验证理论产生一个有趣的扩展，使其适用于半监督学习。

7.9　半监督学习

在这本书中，从第 1 章的感知器开始，到这个话题为止，我们一直在关注监督学习。即根据给定的训练样本$\{\mathbf{x}_i, d_i\}_{i=1}^N$，学习一个输入输出映射关系。我们称这样的数据集叫带标记的，即对于每个 $\mathbf{i}$ 来说，输入向量 $\mathbf{x}_i$ 都配对了一个预期的响应或可称之为类标 d_i。从实用的角

度上看，对于以监督的方式训练一个网络，对样本手动标记类标不但是一个耗费大量时间和成本的工作，而且这个过程极其容易出错。相反，收集无类标样本（即不带有预期响应的样本）是相对低成本的，并且通常可以容易地获得大量的这类样本。根据这些现实，我们如何利用可得到的带类标以及不带类标的样本来训练网络呢？这个具有挑战性的问题的答案就是使用半监督学习。

在这个新的学习方法中，输入数据集$\{\mathbf{x}_i\}_{i=1}^{N}$被分成两部分：

1. 一个样本子集，记为$\{\mathbf{x}_i\}_{i=1}^{l}$，每个样本的类标 $\{d_i\}_{i=1}^{l}$是提供的。

2. 另一个子集记为$\{\mathbf{x}_i\}_{i=l+1}^{N}$，其中每个样本的类标是未知的。

基于此，我们可以把半监督学习看成一种在监督学习和非监督学习之间的新的学习形式。它比监督学习要困难一些，但又比非监督学习要容易一些。

作为一个具有许多潜在应用的课题，半监督学习使用广泛的学习算法。在本章，我们关注基于流形正则化的核方法。“流形”是指一个 k 维的拓扑空间嵌入到一个维数大于 k 的 n 维的欧几里得空间。如果描述流形的函数是可偏微分的，我们称这个流形是可微流形。因此我们可以把一个流形的概念看成$\mathbb{R}^3$空间中一个面的概念的泛化。同理，可以把可微分流形看成$\mathbb{R}^3$空间中可微面的泛化。

对于关注基于流形正则化的核方法有以下三点原因：

1. 对于半监督学习来说，核方法对本章所讨论的正则化理论很适合。

2. 流形正则化提供了对于构造一个用于半监督学习的依赖数据的、无参数的核的有力的方法。

3. 使用流形正则化使一些分类任务产生较好的结果。

简单地说，基于核方法的流形正则化具有对半监督学习理论产生深远影响的潜能。

7.10　流形正则化：初步的考虑

图 7.4 描述了一个半监督学习过程的模型。在图中和本章余下部分，为了简化表示，我们用“分布”指代“概率密度函数”。为了继续下面的讨论，图 7.4 中的模型简化为如下数学的形式：

1. 输入空间用$\mathscr{X}$来表示，并假定是静态的；它提供两个输入数据集，一个记为$\{\mathbf{x}_i\}_{i=1}^{l}$，另一个记为$\{\mathbf{x}_i\}_{i=l+1}^{N}$，两者都服从一个固定的分布$p_x(\mathbf{x})$。我们假定这也属于一个稳定的过程。

2. 对于$\{\mathbf{x}_i\}_{i=1}^{l}$集中的每一个输入向量 $\mathbf{x}$，“教师”提供类标 d_i。类标来自于输入空间$\mathscr{X}$，并同条件分布 $p_{D|\mathbf{x}}(d \mid \mathbf{x})$一致，是固定但未知的。

3. 此机器学习对于两个数据集产生一个输出：

- $\{\mathbf{x}_i, d_i\}_{i=1}^{l}$来自输入空间，并由教师给出类标的带类标数据，服从联合分布

$$p_{\mathbf{x},D}(\mathbf{x},d) = p_{D|\mathbf{x}}(d|\mathbf{x})p_{\mathbf{x}}(\mathbf{x}) \quad (7.105)$$

根据这个定义，$p_{\mathbf{x}}(\mathbf{x})$是边缘分布，通过对联合分布 $p_{\mathbf{x},D}(\mathbf{x},d)$在预期响应 d 上积分得到。

- 无类标数据 $\{\mathbf{x}_i\}_{i=l+1}^{N}$，由输入数据空间$\mathscr{X}$中直接得到，服从分布 $p_{\mathbf{x}}(\mathbf{x})$。

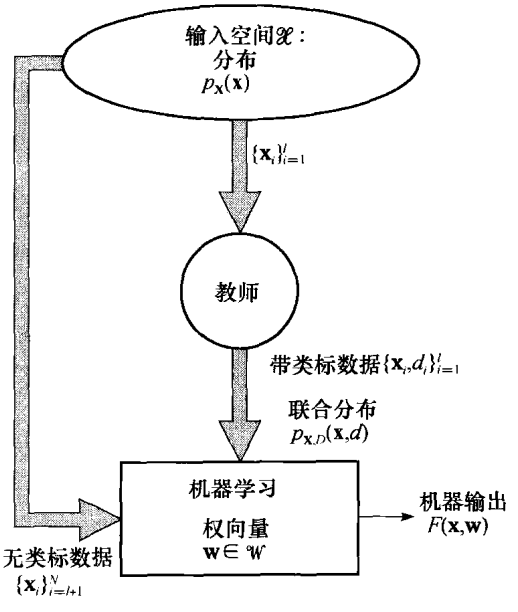


图 7.4　半监督学习过程模型

因此，不同于监督学习，半监督学习中的样本组成如下所示：

$$\text{（训练样本）} = (\underbrace{\{\mathbf{x}_i, d_i\}_{i=1}^{l}}_{\text{带类标的}}; \underbrace{\{\mathbf{x}_i\}_{i=l+1}^{N}}_{\text{无类标的}}) \tag{7.106}$$

在模式识别或回归相关问题中，由于用流形正则化在改进的函数学习中有所不同，因此假定在分布 $p_{\mathbf{x}}(\mathbf{x})$ 和条件分布 $p_{\mathbf{x}|D}(\mathbf{x}|d)$ 之间存在一个等价关系。基于如下两个重要的假定（Chapelle 等，2006），我们可以构造这两个分布之间可能的关联：

1. 流形假设，如下所述：

输入空间$\mathcal{X}$下的边缘分布 $p_{\mathbf{x}}(\mathbf{x})$ 由低维数的流形提供。

这第一个假设的含义是指条件概率函数 $p_{\mathbf{x}|D}(\mathbf{x}|d)$ 相应于流形下的结构而缓慢地变化（作为 x 的函数）。

这里我们提出一个问题：如何使用此流形假设？要回答这个问题，我们要注意如第 4 章一些篇幅中讨论的维数灾难问题。简单地说，随着输入空间维数的增加，一个学习任务对于样本数量的需求是指数增长的。如果，已知数据是在一个低维数的流形上的，我们可以通过在相应的低维数空间上实施学习，以避免维数灾问题。

流形假设对于某些物理过程是恰当的。比如说，考虑语音生成过程，这可以看成是在一发声源激发一个发声系统滤波器时，一种滤波的形式。发声系统由一系列非一致的交叉区域组成，由声门开始，到嘴唇结束。当声音随局部发声系统传递，声音信号的频谱由发声系统的频率选择性形成；这个效果与从管风琴中观察到的共鸣现象有些相似。这里需要注意的要点是语音信号空间是一个低维的流形，变化的参数是发声系统的长度和宽度。

2. 聚类假设，如下所述：

随着应用于函数学习的样本形成，边缘分布 $p_{\mathbf{x}}(\mathbf{x})$ 由如下方式定义：如果特定的样本点位于相同的聚类中，那么它们很有可能是同一类的。

这第二个假设具有合理性。这是因为它对于一个模式分类问题中的各个类是可行的。特别地，如果两个样本输入两个不同的类中，我们观察到它们位于同一个聚类的可能性是相对比较低的。

7.11 可微流形

我们用如下直觉上的概念来开始可微流形的讨论：

流形是一个抽象的数学空间。它其中每一个点都有一个局部的邻接点，这与欧几里得空间相似，但从全局的角度来说，此空间中的点之间具有内在的结构，这比欧几里得空间要复杂。

因此，我们可以把流形想象成一个嵌入欧几里得空间的平面的抽象。

在描述流形时，维数的概念十分重要。广义上说，如果一个点的局部邻居在流形上是 n 维欧几里得空间的，我们可以说这是一个 n 维流形，或 n-流形。

流形与欧几里得空间的局部相似度被假定足够接近，以便将微积分中的常用规则用于流形，使得流形学习更为简单。扩展这个论断，用 $\mathbb{R}$ 表示实数集，用 $\mathbb{R}^n$ 表示它们之间的 Cartesian 点集。在流形学习中，$\mathbb{R}^n$ 有以下含义：有时 $\mathbb{R}^n$ 只是表示一个拓扑空间；有时 $\mathbb{R}^n$ 用来表示一个 n 维向量空间，其上的操作是连续的，且与拓扑相关；有时 $\mathbb{R}^n$ 简单地等同于一个欧几里得空间。

概括地说，拓扑空间是一个几何物体。为了更准确地定义，我们必须引入集合论：

用 X 表示任何一个集合，用$\mathcal{T}$表示 X 子集组成的子集簇。则$\mathcal{T}$是一个拓扑，如果如下三点

成立：

1）集 X 和空集（即不包含任何元素的集合）都属于$\mathcal{T}$。

2）$\mathcal{T}$中有限个元素的并仍属于$\mathcal{T}$。

3）$\mathcal{T}$中任意多个元素的交集仍是$\mathcal{T}$中的元素。

如果$\mathcal{T}$是如上定义的拓扑，则集合 X（如上定义中的）连同$\mathcal{T}$组成了一个拓扑空间。

$\mathcal{T}$中的元素叫做 X 的开集。这个定义的本质是指它可以使我们定义“连续”映射：一个拓扑空间之间的映射（或称函数）f：$X\to Y$ 被称作连续的，如果 Y 中任何开集 A 的原象 $f^{-1}(A)$ 本身是 X 中的开集。原象 $f^{-1}(A)$指 X 中通过 f 映射到 Y 中的 A 的点 x 的集合。

出于对可微性这一问题的特别的考虑，令 $\mathbb{R}^n$ 中一子集 X 为开集。开集定义为一个其中任意点到它的边之间的距离都大于 0 的集合。让 $\mathbf{x}\in X$，记向量 $\mathbf{x}$ 第 i 个分量为 x_i，$f(\mathbf{x})$为从 $\mathbf{X}$ 到 $\mathbb{R}$ 的映射。我们可以作出如下的论断：

对于一个非负的整数 k，如果所有的偏微分 $\partial^\alpha f/\partial x_i^\alpha$ 在 X 上存在并且连续（$1\leqslant i\leqslant n$，且 $0\leqslant\alpha\leqslant k$），则函数 $f(\mathbf{x})$是可微的，称为开集 X 上的 C^k 类，或概括地说 f 属于 C^k 的。

基于此论断，我们可以说函数 f 属于 C^∞（即无穷可微故光滑的），如果对于任意 $k\geqslant 0$，f 都属于 C^k。

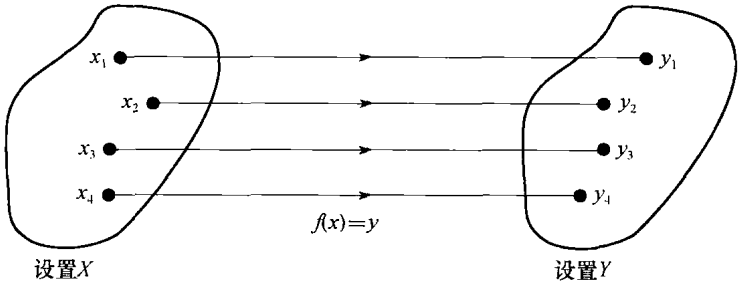


图 7.5　一个双射 f：$X\to Y$

我们仍没有为正式定义微分流行做好完全的准备。因此，我们需要引入一些其他概念，描述如下：

1. 同胚

考虑一个在集合 X 和 Y 之间的映射 f：$X\to Y$。如果 f 具有如图 7.5 所示的属性：对于 Y 中的每个 y，X 中都存在唯一的 x，使得 $f(x)=y$，则 f 称为双射。

X 和 Y 两拓扑空间之间的双射 f：$X\to Y$ 叫做同构映射，如果 f 和其逆映射 f^{-1}都是连续的。当这样的 f 存在时，我们称 X 和 Y 之间是互相同胚的。

从物理意义上看，我们把同胚看成一个拓扑空间的连续延伸和弯曲，使原空间被改变成了一种新的形态。比如说，一个咖啡杯和一个油炸圈饼之间是同构的，因为咖啡杯可以被连续地变形为一个油炸圈饼，反之亦然。另一方面，一个油炸圈饼绝不可能变形为一球，无论对其如何连续的延伸或弯曲。

直觉上，我们可以说同胚映射把一个拓扑空间中的距离接近的点映射到另一个拓扑空间中，使它们之间的距离仍然很接近。

2. 微分同胚

要定义这个概念，我们要求 X 和 Y 是 $\mathbb{R}^n$ 中的开集。我们说 f：$X\to Y$ 是微分同胚的，如果满足以下两个条件：

1）f 是同胚的。

2）f 和 f^{-1} 都是连续可微的。

这里，X 和 Y 被称为互相微分同胚。如果 f 和 f^{-1} 都是 k 次连续可微的，则称 f 为 C^k -可微同胚的。

3. 图表和图集

在学习世界地理时，我们发现使用图集和图表代替把世界表示为一个整体的办法是很方便的。对于世界的一个完整的图片，我们用图集，即一族可以覆盖世界不同部分的地图。

这种对世界地理非数学方式的视角导致我们在直觉上得到构造拓扑流形$\mathcal{M}$的过程：

1）选出一族重叠的简单空间，可以覆盖住整个拓扑空间$\mathcal{M}$。

2）每个简单空间都同 $\mathbb{R}^n$ 中的一个开集同胚。每个这样的同胚叫做一个图表。

3）这些图表被拼接成光滑的方式。

每个图表都由一个三元组（X，Y，f）组成，其中 X 是$\mathcal{M}$中的开集，Y 是 $\mathbb{R}^n$ 中的开集，f：$X \to Y$ 是一个同胚映射。

显然，一族覆盖住整个$\mathcal{M}$的重叠的图表叫做一个图集。很显然通过这个过程不存在构造流形的唯一方法。

从数学意义上说，我们可以看如下的关于图表和图集的定义：

1）用（X_i，f_i）表示第 i 个图表，则图集是所有这些图表的交。

2）图集中的任意两个图表（X_i，f_i）和（X_j，f_j），基于如图 7.6 中的意义，必须是相容的：

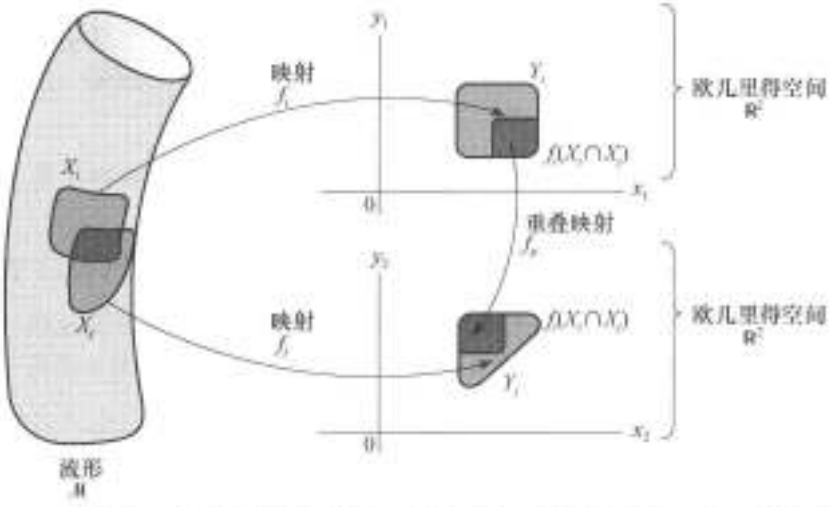


图 7.6　一个图集和组成它的图表之间的关系示例。[此图引用 Abraham 等(1988)]

- 对于两个图表的公共部分，如图 7.6 的阴影部分所示，必须是开的。
- 阴影的重叠部分，记为 f_{ji}，必须是 C^k 微分同胚的。

注意，f_{ji}是一个从像集 $f_i(X_i \cap X_j)$到像集 $f_j(X_i \cap X_j)$的映射，其中符号 $\cap$ 表示两个集合的乘积或交。通过要求每个 f_{ji} 都是 C^k-可微同胚的，我们可以确定$\mathcal{M}$上 C^k-可微函数的意义。

可微流形

最后，我们给出可微流形的定义：

一个由三元组 (X_i, Y_i, f_i)，$i = 1, \cdots, I$ 表示的 n 维的 C^k-可微流形$\mathcal{M}$是一个拓扑集，其中每个 Y_i 是一 $\mathbb{R}^n$ 上的开集，使得所有重叠的映射 f_{ji} 都是 C^k-可微的。

这个定义是指对于每个 n 维数据点 $\mathbf{x}\in\mathcal{M}$，都有一个表（X，Y，f），其中，$\mathbf{x}\in X$ 且 f 把开集 X 映射到 $\mathbb{R}^n$ 的开子集 Y 上。

为什么对流形学习感兴趣

为了评价流形在学习理论中的重要性，假定我们有一系列无类标样本，记为 $\mathbf{x}_1,\mathbf{x}_2,\cdots$，其中每个样本的维数都是 n。这些样本可以表示成 n 维欧几里得空间中的数据点集。大多数无监督学习算法只是在由样本 $\mathbf{x}_1,\mathbf{x}_2,\cdots$，标出的外围空间上执行。假定我们可以构造一个维数低于 n 的流形，使得真正的数据位于或在那个流形周围。这样，就很有可能通过了解流形以及其外围空间的几何属性，设计出一个更有效的半监督学习算法。这里描述的此思想不仅仅是数据表示的另一种方法，它通过采样数据点，提供了一种流形上的逼近问题的学习算法的新体制（Belkin，2003）。但是，为了使这些新方法成为现实，我们必须知道用来描述输入空间的内在几何结构的流形的特征。遗憾的是，这些知识在实际应用中很难获得。为了解决这个难题，如下面两节讨论的那样，我们尝试构造流形的模型。

7.12　广义正则化理论

在第 7.3 节中讨论的 Tikhonov 经典正则化理论，使用了一个反映类标样本所在的外围空间的简单罚函数。在本节，我们对此理论推广，使用另一个反映无类标样本所在的输入空间内在几何结构的罚函数。实际上，这个新理论，即广义正则化理论，使用了基于类标样本和无类标样本的半监督函数学习的思想。另外，它包括了在特殊情形下仅基于无类标样本的半监督函数学习。

成对出现的带类标样本记为（$\mathbf{x},d$），根据式(7.105)所定义的联合分布函数 $p_{\mathbf{x},D}(\mathbf{x},d)$ 所产生。无类标样本，$\mathbf{x}\in X$，由边缘分布函数 $p_{\mathbf{x}}(\mathbf{x})$ 所产生。此广义正则化理论潜在的前提是这两个分布之间存在一个等价关系。否则，边缘分布的知识不可能被实际使用。因此，我们作出如下的假定：

如果两个输入样本点 $\mathbf{x}_i,\mathbf{x}_j\in\mathbf{X}$ 在边缘分布函数 $p_{\mathbf{x}}(\mathbf{x})$ 的内在几何结构中是接近的，那么对于在点 $\mathbf{x}=\mathbf{x}_i$ 和 $\mathbf{x}=\mathbf{x}_j$ 的条件分布函数 $p_{\mathbf{x}|D}(\mathbf{x}|d)$ 是相似的。

为了把这个假定改成更为实际的方式，使得能得到实用的办法，我们如下表述：

如果两个数据点 $\mathbf{x}_i$ 和 $\mathbf{x}_j$ 在输入空间中很接近，半监督函数学习的目标是找到一个记为 $F(\mathbf{x})$ 的映射，使得能把相应的输出点 $F(\mathbf{x}_i)$，$F(\mathbf{x}_j)$ 映射到位于同一条实线上且距离很近的可能性较大。

要达到这个目标，我们需要在经典正则化理论中所考虑的罚项外，引入一个新的罚项。

具体地说，我们推广半监督学习的正则化代价函数，引入一个新的罚项，如下所示：

$$\mathscr{E}_\lambda(F)=\frac{1}{2}\sum_{i=1}^{l}(d_i-F(\mathbf{x}_i))^2+\frac{1}{2}\lambda_A\|F\|_K^2+\frac{1}{2}\lambda_I\|F\|_I^2 \tag{7.107}$$

其中两个罚项如下：

1. 由外围正则化参数 λ_A 控制的罚项 $\|F\|_K^2$，反映了外围空间中逼近函数 F 的复杂度。特别地，这个罚项以特征空间（即下标 K）复制核 Hibert 空间（RKHS）表示形式给出。

2. 由内在正则化参数 λ_I 控制的罚项 $\|F\|_I^2$，反映了输入空间（即下标 I）内在几何结构。

$\mathscr{E}_\lambda(F)$ 中的下标 λ 代表两个正则化参数 λ_A 和 λ_I。注意式(7.107)右端第一项，我们使用 l 表示带类标样本的数量。

因为没有内在罚项 $\|F\|_I^2$，RKHS 上的代价函数 $\mathscr{E}_\lambda(F)$ 的最小点由如下的经典表示理论定义：

$$F_\lambda^*(\mathbf{x})=\sum_{i=1}^{l}a_i k(\mathbf{x},\mathbf{x}_i),\quad \text{当 } \lambda_I=0 \tag{7.108}$$

根据此，这个问题可以规约到一个在由系数 $\{a_i\}_{i=1}^{l}$ 所定义的有穷维空间上的优化。我们

可以推广此理论以同样包含内在罚项 $\|F\|_I^2$。

为了此目标，我们提出用一个图来对输入空间的内在几何结构建模的办法。而如下面将讨论的那样，用于构造此图的无类标样本是足够多的。

7.13　光谱图理论

考虑这个训练样本：

$$X = \{\mathbf{x}_i\}_{i=1}^N$$

其中包含 N 个输入数据，既有带类标的，也有无类标的。根据这个训练样本，通过构造一个包含 N 个结点的带权无向图来处理。其中每个结点表示一个输入样本点，图中的一系列边连接相邻结点。任意两个结点 i 和 j 之间是有连接的，如果相应两数据点 $\mathbf{x}_i$ 和 $\mathbf{x}_j$ 之间的欧几里得距离足够小，对于一些指定的 ε，可满足如下条件：

$$\|\mathbf{x}_i - \mathbf{x}_j\| < \varepsilon \tag{7.109}$$

这个邻接准则有如下双重吸引人的特点：几何直观性和自然的对称性。然而，必须记住的是，因为这个图很有可能有多重的连通分量，对常量选择一个合适的值是比较困难的。

用 w_{ij} 表示连接结点 i 和 j 的无向边的权值。图中所有的权值通常都用实数表示，对于这些权值的选择需要满足以下三个条件：

1. 对称性，即指对于所有 (i, j)，$w_{ij} = w_{ji}$ 成立；
2. 连通性，即指如果相应的结点 i 和 j 是连接的，则权值 w_{ij} 非零，否则权值 w_{ij} 为零；
3. 非负性，指对于所有 (i, j)，$w_{ij} \geqslant 0$。

因此，$N \times N$ 的权值矩阵：

$$\mathbf{W} = \{w_{ij}\}$$

是一个对称非负定矩阵，其所有的元素都非负。矩阵 $\mathbf{W}$ 的行和列指代图中的结点，但它们的顺序并不重要。此后，我们指由权值矩阵 $\mathbf{W}$ 表示的无向图为 G。

用 $\mathbf{T}$ 表示一个 $N \times N$ 的对角矩阵，其中它的对角线上元素都如下定义：

$$t_{ii} = \sum_{j=1}^{N} w_{ij} \tag{7.110}$$

这叫做结点 i 的度。换句话说，结点 i 的度等于权值矩阵 $\mathbf{W}$ 所有第 i 行中元素的和。t_{ii} 度越大，结点 i 就越重要。在很少的情况下，t_{ii} 的值会为零，则结点 i 称为孤立的。

在权值矩阵 $\mathbf{W}$ 和对角矩阵 $\mathbf{T}$ 中，我们现在定义图 G 的拉普拉斯算子如下：

$$\mathbf{L} = \mathbf{T} - \mathbf{W} \tag{7.111}$$

如果我们假定不存在环，即对所有的 i，$w_{ii} = 0$，则对于拉普拉斯矩阵 $\mathbf{L}$ 的第 i 行 j 列中的元素，我们有：

$$l_{ij} = \begin{cases} t_{ii}, & \text{对于 } j = i \\ -w_{ij}, & \text{对于邻接点 } i \text{ 和 } j \\ 0, & \text{否则} \end{cases} \tag{7.112}$$

因此我们得知拉普拉斯矩阵 $\mathbf{L}$ 是对称矩阵。

如下所述，图拉普拉斯是构造一个合适的光滑函数而处理内罚项 $\|F\|_I^2$ 的关键所在。

因为拉普拉斯矩阵 $\mathbf{L}$ 是对称矩阵，它的特征值是实的。有关特征分解的话题，包括计算一个对称矩阵的特征值，在第 8 章将详细讨论。在此，我们发现用对称矩阵的 Rayleigh 系数去求拉普拉斯矩阵 $\mathbf{L}$ 的特征值的变化特征很适合。因此，用 $\mathbf{f}$ 表示一个人造的关于输入向量 $\mathbf{x}$ 的向量值函数。其中 $\mathbf{x}$ 是关于图 G 中的每一个结点赋一个实数值。然后可以用如下的比值来定义

拉普拉斯算子 **L** 的 Rayleigh 商：

$$\lambda_{\text{Rayleigh}} = \frac{\mathbf{f}^T\mathbf{L}\mathbf{f}}{\mathbf{f}^T\mathbf{f}} \tag{7.113}$$

此 Rayleigh 商体现了两个内积的比：

1. 函数 **f** 和矩阵 **Lf** 的内积，其中拉普拉斯矩阵 **L** 作为作用在函数 **f** 上的一个算子。

2. 函数 **f** 同它本身的内积，即 **f** 的欧几里得范数的平方。

应注意到根据式(7.113)，式中的拉普拉斯矩阵 **L** 是一个非负定的矩阵。

L 是一个 $N\times N$ 的矩阵，所以它应该有 N 个实的特征值。对它的特征值按顺序排列如下：

$$\lambda_0 \leqslant \lambda_1 \leqslant \cdots \leqslant \lambda_{N-1}$$

这些特征值就叫做拉普拉斯矩阵 **L** 的*特征光谱*，或关联矩阵 G 的特征光谱。不难看出，最小特征值 λ_0 是 0，且相应的特征向量是 **1**，即其所有的 N 个分量都是 1。第二小的特征向量 λ_1 对于光谱图理论起到了重要的作用。

且不说 λ_1 的重要性和拉普拉斯矩阵 **L** 的其他特征值，本章主要关注的是为处理内罚项 $\|F\|_I^2$ 找到一个合适度量。我们看式(7.113)，寻找的度量就是 Rayleigh 商的分子（二次项 $\mathbf{f}^T\mathbf{L}\mathbf{f}$）。相应地，我们引入光滑函数

$$S_G(F) = \mathbf{f}^T\mathbf{L}\mathbf{F} \tag{7.114}$$

这不仅合理，而且直觉上满足要求。向量值函数 **f** 就训练样本 X 定义如下：

$$\mathbf{f} = [F(\mathbf{x}_1), F(\mathbf{x}_2), \cdots, F(\mathbf{x}_N)]^T \tag{7.115}$$

因此，在式(7.114)中使用式(7.112)和式(7.115)，我们可以同样通过如下所示的和式来表达光滑函数：

$$S_G(F) = \sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}(F(\mathbf{x}_i) - F(\mathbf{x}_j))^2 \tag{7.116}$$

其中 w_{ij} 是连接结点 i 和 j 的边的权值。

为了完成对光滑函数 $S_G(f)$ 的描述，我们需要一个对图 G 的边权值估值的公式。根据核方法的精髓，我们用核函数来定义连接结点 i 和 j 的边的权值 w_{ij}；即

$$w_{ij} = k(\mathbf{x}_i, \mathbf{x}_j) \tag{7.117}$$

这个定义对权值 w_{ij} 满足对称性，连通性和非负性的条件。一个这样的核的例子是高斯函数：

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right) \tag{7.118}$$

其中 σ^2 是设计者控制的参数，假定对所有的 i 都相同，即所有的核都在光谱图中。

在此，对半监督学习的内容中的要点总结如下：

通过联合式(7.117)和式(7.118)，对光谱图理论的应用，使得关于半监督学习的机器学习称为核机器。此核机器的隐藏层通过产生无类标样本的输入空间的内在几何结构确定。

7.14　广义表示定理

通过已得到的式(7.114)中的光滑函数，我们可以把式(7.107)中的代价函数重写成预期的形式：

$$\mathscr{E}_A(F) = \frac{1}{2}\sum_{i=1}^{l}(d_i - F(\mathbf{x}_i))^2 + \frac{1}{2}\lambda_A\|F\|_K^2 + \frac{1}{2}\lambda_I\mathbf{f}^T\mathbf{L}\mathbf{f} \tag{7.119}$$

其中对一个再生核 Hilbert 空间进行优化（即 F 在 RKHS 中）。对代价函数 $\mathscr{E}_A(F)$ 的优化产生一个扩张形式：

$$F_A^*(\mathbf{x}) = \sum_{i=1}^{N} a_i^* k(\mathbf{x}, \mathbf{x}_i) \tag{7.120}$$

其中既包括带类标样本又包括无类标样本（Belkin 等，2006）。因此，这个扩张可以看作经典表示定理在半监督中的泛化。

为了证明此定理，我们首先需认识到任何再生核 Hilbert 空间中的函数 $F(\mathbf{x})$可以被分解成两个分量的和：一个分量 $F_{\parallel}(\mathbf{x})$，包含在核函数 $k(\cdot,x_1)$，$k(\cdot,x_2),\cdots,k(\cdot,x_N)$ 张成的空间中；另一个分量 $F_{\perp}(\mathbf{x})$，包含在它的正交补空间中。即可以写成：

$$F(\mathbf{x}) = F_{\parallel}(\mathbf{x}) + F_{\perp}(\mathbf{x}) = \sum_{i=1}^{N} a_i k(\mathbf{x},\mathbf{x}_i) + F_{\perp}(\mathbf{x}) \tag{7.121}$$

其中 a_i 是实系数。通过引入第 6 章中讨论的再生属性，我们发现当 $1\leqslant j\leqslant N$ 时，对函数 $F(\mathbf{x})$ 在任意数据点 x_j 的估值，是同正交分量独立的，即如下：

$$\begin{aligned} F(\mathbf{x}_j) &= \langle F, k(\cdot,\mathbf{x}_j)\rangle = \langle \sum_{i=1}^{N} a_i k(\cdot,\mathbf{x}_i), k(\cdot,\mathbf{x}_j)\rangle + \langle F_{\perp}, k(\cdot,\mathbf{x}_j)\rangle \\ &= \sum_{i=1}^{N} a_i \langle k(\cdot,\mathbf{x}_i), k(\cdot,\mathbf{x}_j)\rangle + \langle F_{\perp}, k(\cdot,\mathbf{x}_j)\rangle \end{aligned} \tag{7.122}$$

现在注意两点：

1. 在式(7.122)的第一项中，我们有

$$\langle k(\cdot,\mathbf{x}_i), k(\cdot,\mathbf{x}_j)\rangle = k(\mathbf{x}_i,\mathbf{x}_j)$$

2. 在第二项中 $\langle F_{\perp}, k(\cdot,\mathbf{x}_j)\rangle$，为零。

因此可以得到：

$$F(\mathbf{x}_j) = \sum_{i=1}^{N} a_i k(\mathbf{x}_i,\mathbf{x}_j) \tag{7.123}$$

此式显示包含正则化代价函数和最小化式(7.119)的内在范数的经验项仅依赖于系数 $\{a_i\}_{i=1}^{N}$和核函数的 Gram 矩阵。

下面，我们注意到对所有的 $F_{\perp}$，这个正交分量仅趋于增加再生核 Hilbert 空间中函数的范数。换句话说，

$$\| F \|_K^2 = \left\| \sum_{i=1}^{N} a_i k(\cdot,\mathbf{x}_i) \right\|_K^2 + \| F_{\perp} \|_K^2 \geqslant \left\| \sum_{i=1}^{N} a_i k(\cdot,\mathbf{x}_i) \right\|_K^2$$

其中最小下标 K 指再生核 Hilbert 空间。

因此，为了使对代价函数$\mathscr{E}(F)$最小化得以实现，我们必须有 $F_{\perp}=0$，这就证明了式(7.120)中的广义表示定理。在此式中使用了最优化设置的表示。

此广义表示理论的简单形式把一个外在-内在正则化框架转换成为一个相应的由有穷维系数 $\{a_i^*\}_{i=1}^{N}$ 空间所规定的优化问题成为了可能。其中 N 是所有带样本和无类标样本的数量的总和（Belkin 等，2006）。这样做，我们可以为了解决 7.15 节所示的困难的半监督学习问题而引入核方法。

7.15 拉普拉斯正则化最小二乘算法

在 7.12 节中，我们介绍了光滑函数的概念，其公式体现了光谱图理论下的拉普拉斯算子。特别地，定义的光滑函数的公式是核的，正如式(7.116)和式(7.118)所示，其使得函数非线性地依赖于输入向量 $\mathbf{x}$。下面我们将广义化该表示定理，使得该函数适应于带类标样本和无类标样本。利用这些我们可处理的工具，现在我们可以设定拉普拉斯正则化最小二乘算法的公式（Belkin 等，2006；Sindhwani 等，2006）。新算法的实用性体现在以下两点：

1. 对该算法的训练既使用带类标样本，又使用无类标样本，因此，可把算法的实用性提升到比那些现有的单独的监督训练算法更广的范围。

2. 通过核方法，算法可以对非线性可分离的模式进行识别，因此，拓展了最小二乘估计的应用。

基本上说，LapRLS 算法来源于最小化式(7.119)中关于函数 $F(\mathbf{x})$ 的代价函数。(对带类标样本和无类标样本) 使用表示定理，我们有

$$F(\mathbf{x}) = \sum_{i=1}^{N} a_i k(\mathbf{x}, \mathbf{x}_i)$$

在式(7.119)中使用矩阵符号，得到

$$\mathscr{E}_A(\mathbf{a}) = \frac{1}{2}(\mathbf{d} - \mathbf{JKa})^T(\mathbf{d} - \mathbf{JKa}) + \frac{1}{2}\lambda_A \mathbf{a}^T\mathbf{Ka} + \frac{1}{2}\lambda_I \mathbf{a}^T\mathbf{KLKa} \tag{7.124}$$

此处我们引入以下符号：

$\mathbf{d} = l$ 乘以 1 的预期相应向量
$= [d_1, d_2, \cdots, d_l]^T$
$\mathbf{a} = N$ 乘以 1 的扩张系数向量
$= [a_1, a_2, \cdots, a_N]^T$
$\mathbf{J} = N$ 乘以 N 的对角矩阵，其对角线上只有 1 个是单位项
$= \text{diag}[1, 1, \cdots, 1, 0, 0, \cdots, 0]$

此 $L \times L$ 的矩阵 K 是 Gram 矩阵，$\mathbf{L}$ 是拉普拉斯图矩阵。注意到式(7.124)右边的表达式是一个未知向量 $\mathbf{a}$ 的二次函数，因此代价函数可记为 $\mathscr{E}_A(\mathbf{a})$。对此等式关于向量 $\mathbf{a}$ 微分，合并并简化项，然后求解最小点值 $\mathbf{a}^*$，得到

$$\mathbf{a}^* = (\mathbf{JK} + \lambda_A \mathbf{I} + \lambda_I \mathbf{LK})^{-1}\mathbf{J}^T\mathbf{d} \tag{7.125}$$

其中使用 Gram 矩阵 $\mathbf{K}$ 的对称性和对角矩阵 $\mathbf{J}$、单位矩阵 $\mathbf{I}$。见习题 7.16。

当我们把内在正则化参数设定为零时 (即 $l = N$)，(请注意此条件下矩阵 $\mathbf{J}$ 成为标准对角阵的形式)，式(7.125)中的公式被简化到式(7.74)中普通正则化最小二乘算法。

表 7.1 给出了一个 LapRLS 算法的总结，其中包含四个设计者控制的参数：

1. 两个正则化参数：λ_A 和 λ_I；
2. 两个图参数 ϵ 和 σ^2，其中 ϵ 用于式(7.109)的邻接矩阵中，σ^2 用于式(7.118)中的核权值中。

注意到这个算法不需要计算 RLS 算法的权值向量。我们通过计算与表示定理相关的参数向量 $\mathbf{a}$，而避免了对此的计算。

在表 7.1 中总结了一个半监督学习算法的显著的特征，就是需要知道两个正则化参数 λ_A 和 λ_I。正如我们以前指出，推广第 7.8 节的交叉验证理论正适合对 λ_A 和 λ_I 进行估计。

表 7.1　拉普拉斯正则化最小二乘算法总结

给定量

向量样本 $\{\mathbf{x}_i, d_i\}_{i=1}^{l}$ 和 $\{\mathbf{x}_i\}_{i=l+1}^{N}$，分别是带类标的和无类标的。

l 是带类标样本的数量，$N-l$ 是无类标样本的数量。

设计的参数

ϵ 和 σ^2：　光谱图参数

λ_A 和 λ_I：　正则化参数，外部的和内在的

计算

1. 构造一个有 N 个结点的带权无向图 G，使用
   - 式(7.109)对图的邻接点进行辨认。
   - 式(7.117)和式(7.118)计算边权值。
2. 选择核函数 $k(\mathbf{x}, \cdot)$，并使用训练样本计算 Gram 矩阵 $K = \{k(\mathbf{x}_i, \mathbf{x}_j)\}_{i,j=1}^{N}$。
3. 使用式(7.110)和式(7.112)计算图 G 的拉普拉斯矩阵 $\mathbf{L}$。
4. 使用式(7.125)计算最优系数向量 $\mathbf{a}^*$。
5. 使用式(7.120)的表示理论计算优化逼近函数 $F_A^*(\mathbf{x})$。

7.16 用半监督学习对模式分类的实验

为了说明拉普拉斯 RLS 算法的模式分类能力，我们基于抽取自图 1.8 的双月图的合成数据来进行一个小的实验。特别地，我们把实验中的两个参数设置为固定不变的：

两个月亮之间的垂直分离，$d=-1$。

外围正则化参数，$\lambda_A=0.001$。

实验中唯一变化的参数是内正则化参数 λ_I。

当 λ_I 正好被设置为零时，拉普拉斯 RLS 算法简化成传统的 RLS 算法。其中带类标数据是提供学习信息的唯一来源。从实验的角度来看，我们关注的是在半监督学习的过程中，加入无类标信息是如何通过变化的参数 λ_I 影响由拉普拉斯 RLS 算法构造的决策边界的。在实验的第二部分中，λ_I 被赋予了一个足够大的值，以使得无类标样本对算法产生完全的影响。

对于两部分实验，每个类中只提供了两个类标数据点，一个类代表图 1.8 中上方的月亮，另一个类代表底部的月亮。训练样本的总和，包括类标样本和无类标样本有 $N=1000$ 个；测试样本的数量同样有 1000 个。

(a) 内在正则化参数，$\lambda_I=0.0001$。对于这个设置，图 7.7 给出了由拉普拉斯 RLS 算法构造的决策边界。尽管对 λ_I 赋了一个很小的值，这已显著地改变了由 RLS 算法（即 $\lambda_I=0$）所确定的决策边界。我们从图 2.2 和图 2.3 中回忆到 RLS 算法的决策边界是一条具有正坡度的直线。

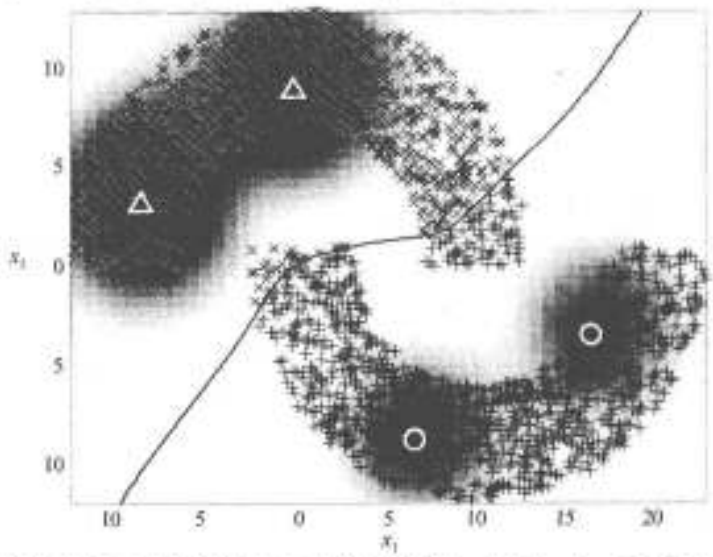


图 7.7 拉普拉斯 RLS 算法对图 1.8 中的双月分类，距离为 $d=-1$，每个类中的两个带类标数据点用符号△和○表示。内正则化参数 $\lambda_I=0.0001$

从效果上看，1 000 个测试数据中总共有 107 个错误分类；即分类错误率是 10.7%。

(b) 内正则化参数，$\lambda_I=0.1$。在实验的第二部分中，内正则化参数 λ_I 被赋值为 0.1，因此可以使得拉普拉斯 RLS 算法可以完全地利用无类标样本的内在信息内容。类标信息点的位置与实验的第一部分中的完全相同。

为了实现拉普拉斯 RLS 算法，我们在式(7.118)中设置了一个 $2\sigma^2=3$ 的 RBF 核。为了构造本身，我们使用了 20-最近邻图。实际上，为了此目的，RBF 网络有一个含 20 个计算结点的隐藏层。

第二部分实验的结果得到了图 7.8 所示的网络配置。与图 7.7 相比，我们看到在参数 $\lambda_I=0.1$ 和 $\lambda_I=0.00\,01$ 情况下，由拉普拉斯 RLS 算法构造的决策边界有显著的不同。特别地，两

个类（即上方的月亮和底部的月亮）现在被没有分类误差地分离了。这个结果在设置 $d=-1$ 时最为明显，两个类的样本线性地分离了，并且拉普拉斯 RLS 算法能够在每个类仅用两个带类标样本的情况下成功地分离它们。拉普拉斯 RLS 算法的这个显著的性能归因于能够充分地利用两个类的无类标数据中含有的信息。

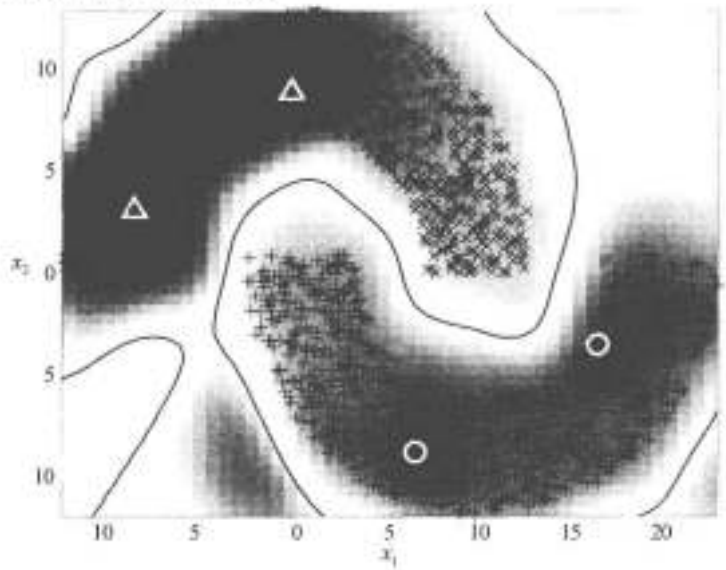


图 7.8　对图 1.8 中的双月图用拉普拉斯 RLS 分类，距离为 $d=-1$，每个类中的两个带类标数据点用符号△和○表示。内正则化参数 $\lambda_I=0.1$

两个部分的实验清楚地证明了正则化外部形式和内部形式的折中，其中由拉普拉斯 RLS 算法所示的半监督学习过程能够借助相对很少的带类标样本，从无类标样本完成泛化。

案例研究：使用 USPS 数据进行模式分类

图 7.9 指出了 RLS 和拉普拉斯 RLS 算法对于实际图像分类问题，使用美国邮政服务（USPS）的数据集的学习曲线。这些数据集包含 10 个手写数字类的 2007 个图像样本，其中每

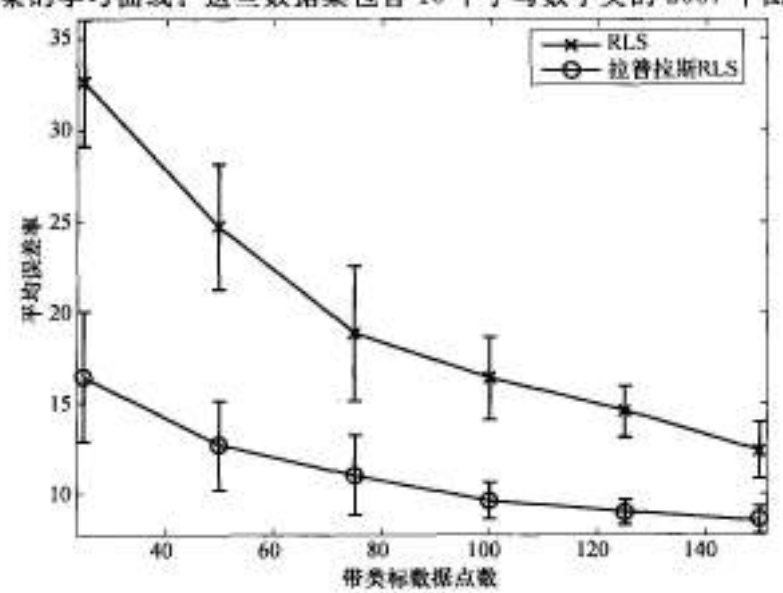


图 7.9　对 USPS 数据使用（a）RLS 算法和（b）拉普拉斯 RLS 算法（此图的复制得到 Vikas Sindhwani 博士的允许）

个图像样本用一个 256 维的像素矢量表示。对于此十个类中的每一个类，使用 RLS 和拉普拉斯 RLS 算法分别训练一个两类分类器。多类分类通过选取最大输出的类来实行，即用一类对剩余的多类的模式分类。图 7.9 描绘了平均分类误差率和两个算法作为由训练集中 2007 个样本所提供的带类标样本的函数的标准差。图 7.9 中的每一个点都由随机选择十个类标而获得。我们使用了一个高斯 RBF 核。对于式(7.118)中的指数，我们把 $2\sigma^2$ 设置为与随机从训练集中挑出的样本之间的欧几里得距离相同。对于拉普拉斯 RLS，我们使用 10 个近邻图来构造拉普拉斯；使用的正则化参数为 $\lambda_A = 10^{-6}$ 和 $\lambda_I = 0.01$。对于 RLS，在许多值上调试，使得其得到一个如图 7.9 所示的最优学习曲线。图 7.9 中所示的结果进一步证明了，与 RLS 算法相比，使用无类标数据显著提升了拉普拉斯 RLS 的性能。

7.17　小结和讨论

正则化理论是所有学习理论的核心。在本章，我们对正则化理论进行了详细的介绍。从 Tikhonov 的使用带类标样本用于监督学习的经典正则化理论开始，到应用于使用带类标样本和无类标样本的半监督学习的广义正则化理论结束。

Tikhonov 的正则化理论

在其最基本的形式中，用于 Tikhonov 的正则化理论的泛函由两项组成：一项是经验代价函数，用带类标训练样本的方式定义；另一项是正则化项，用应用于逼近函数的微分算子定义。此微分算子作为一个*光滑约束*，作用在由最小化代价函数而得到的解上。该代价函数与逼近函数的未知参数（权值）向量有关。这个最优解的重点是 Green 函数，其作为一个径向基函数网络的核。然而，要记住的是，对网络复杂性的约减成为决定光滑正则化算子的关键因素。

无论选择何种正则算子，为了使得 Tikhonov 的正则化理论的优点全被所设计的正则化网络所使用，我们需要一个估计正则化参数的原则性的方法。7.8 节中描述的广义交叉验证过程符合这个特定的要求。

半监督学习

随着对监督学习的正则化理论的完整学习，我们转而关注半监督学习的正则化。这是使用带类标数据和无类标数据来实现的。代价函数现在由三项组成：

- *经验代价函数*，由带类标实例定义。
- *外正则化项*，其反映了逼近函数的复杂度。该逼近函数需要使用带类标样本。
- *内正则化项*，其反映了用来产生无类标样本的输入空间的内在几何结构。

相应地，有两个正则化产生，一个是在外正则化项中，另一个在内正则化项中。

相应地，有两个正则化参数，一个用于外项，另一个用于内项。

作为广义正则化理论的一个重要实例，我们使用带类标实例和无类标实例来考虑最小二乘估计问题。通过使用一个包含拉普拉斯算子和表示理论泛化形式的应用的核光滑函数，我们可以推出一个半监督学习的正则化最小二乘估计算法；这个算法，称为*拉普拉斯正则化最小二乘算法*，有两个重要的使用特性：

1. 对于训练，算法可以处理带类标样本和无类标样本。故而该算法拓宽了其对更为困难的模式识别问题的应用。

2. 通过在算法公式中很基本的光滑函数的核化用最小二乘估计，对非线性可分模式的识别变得更为可行。

这个算法的实用性可由两个深刻的计算机实验证明，一个包括合成数据，另一个包括实际数据。

在 Belkin 等（2006）中，通过拉普拉斯支持向量机（LapSVM）推出了一个半监督学习算法。此算法能够成功地测试一些实际的数据集。然而，算法需要求一个稠密 Gram 矩阵的逆，因此会使得计算复杂度达到 N^3 阶，其中 N 是完全的训练样本数量（包括带类标样本和无类标样本）；另外，就像标准的支持向量机一样，我们仍然要解一个二次规划问题，其复杂度同样达到了 N^3 阶。LapRLS 算法的复杂度上要比 LapSVM 简单，因为在其公式中没有二次规划问题。更为重要的是，实验结果似乎显示了这两种半监督机器学习的性能很相近。因此，从实用的角度来看，LapRLS 算法对于求解半监督学习问题是一个更好的选择。

然而，LapRLS 算法的计算复杂度同样是 N^3 阶，这是因为在代价泛函中包括了内项。这个额外的高的计算复杂度使得 LapRLS 算法很难应用于包含大规模数据集的实际问题。所以研发可用于大规模数据的半监督学习算法在当前仍然是一个热门的话题。

注释和参考文献

1. *从诸如一个病态求逆问题的实例中学习。*通过实例的机器学习会违反一个或多个关于良态问题的 Hadamard 条件，这使我们把学习过程看作一个病态的求逆问题。然而，从严格的数学角度看，学习理论和病态求逆问题理论之间的联系并非直接的。这两个理论的数学基础是不同的；通常，学习理论自然上看是内在不确定的（不管我们是否显式地把概率理论加入其公式中），然而另一方面，逆问题理论可以被看作是一个几乎确定的问题。DeVito 等(2005) 提出了一个从诸如一个病态求逆问题的实例中学习的直观阐述。
2. *等式(7.46)的验证。*在基本项中，我们可以通过单位高斯函数来验证等式(7.46)的有效性：

$$G(x) = \exp(-\pi x^2) \tag{A}$$

其是一维的，$\sigma^2 = 1/2\pi$。基本上，我们所需要的是：

$$\sum_{n=0}^{\infty} (-1)^n \frac{(2\pi)^{-n}}{n!2^n} \frac{\partial^{2n}}{\partial x^{2n}} G(x) = \delta(x) \tag{B}$$

其中 $\delta(x)$是中心在 $x=0$ 点的 Dirac delta 函数。

要验证等式(B)，最方便的方法就是研究傅里叶变换（Kammler，2000）的基本属性。特别地，关于*微分属性*有：

$G(x)$在 x 域的微分等价于$\hat{G}(s)$的乘积。$G(s)$是 $G(x)$以 $i2\pi s$ 的傅里叶变换，其中 s 是空间频率，i 是 -1 的方根。

由傅里叶理论，我们同样可以知道在数学项中，单位高斯函数是其自身的傅里叶变换。特别地，对于等式(A)中的 $G(x)$，我们有：

$$\hat{G}(s) = \exp(-\pi s^2) \tag{C}$$

因此，通过等式(B)左边项的无穷级数求和的傅里叶变换，可得（简化后的项）：

$$\sum_{p=0}^{\infty} (-1)^p \frac{(2\pi)^{-p}}{p!2^p} (i2\pi s)^{2p} \exp(-\pi s^2) = \exp(-\pi s^2) \sum_{p=0}^{\infty} \frac{(\sqrt{\pi} s)^{2p}}{p!} \tag{D}$$

这个等式(D)右端的新的无穷级数现在可以被认为是指数函数 $\exp(\pi s^2)$的一系列的扩展。因此，等式(D)的右端项实际上等于 Dirac delta 函数 $\delta(x)$的单位逆变换。则等式(B)的验证就可以确立。

通过等式(B)的一维情况，我们可以通过考虑二维以及多维情况，引入归纳法而继续验证等式(7.46)。

3. *正则化精确插值。*在 Yee 和 Haykin(2001) 中，描述了一个设计 RBF 网络的方法，其包括两个严密的理论：
   - 在 7.3 节中描述的精确插值的正则化理论。
   - 在第 5 章中描述的核回归估计理论。

   由后一个理论，我们关注 Nadaraya-Watson 回归估计算子。这个方法提供了一个可简单编码且具有高效性能的解决回归和模式识别问题的基本策略。然而，此方法对计算量的要求较高，特别是训练集的规模较大时。
4. *广义交叉验证。*为了从通常的交叉验证得到广义交叉验证，我们先考虑在 Wahba(1990) 中的一个岭回归问题（ridge regression problem）：

$$\mathbf{y} = \mathbf{X}\boldsymbol{\alpha} + \boldsymbol{\varepsilon} \tag{A}$$

其中 $\mathbf{X}$ 是一个 $N\times N$ 阶的输入矩阵，噪声向量 $\boldsymbol{\varepsilon}$ 具有零均值，且其协方差矩阵等于 $\sigma^2\mathbf{I}$。对 $\mathbf{X}$ 进行奇异值分解有

$$\mathbf{X}=\mathbf{U}\mathbf{D}\mathbf{V}^T$$

式中 $\mathbf{U}$ 和 $\mathbf{V}$ 是正交矩阵，$\mathbf{D}$ 是对角阵。令

$$\tilde{\mathbf{y}}=\mathbf{U}^T\mathbf{y}$$
$$\boldsymbol{\beta}=\mathbf{V}^T\boldsymbol{\alpha}$$

和

$$\tilde{\boldsymbol{\varepsilon}}=\mathbf{U}^T\boldsymbol{\varepsilon} \tag{B}$$

可以用 $\mathbf{U}$ 和 $\mathbf{V}$ 将式(A)转变为

$$\tilde{\mathbf{y}}=\mathbf{D}\boldsymbol{\beta}+\tilde{\boldsymbol{\varepsilon}}$$

选择对角矩阵 $\mathbf{D}$(注意不要与微分算子混淆）使其奇异值成对出现。这样就有一个正交矩阵 $\mathbf{W}$，使 $\mathbf{W}\mathbf{D}\mathbf{W}^T$ 是轮换矩阵，即

$$\mathbf{A}=\mathbf{W}\mathbf{D}\mathbf{W}^T=\begin{bmatrix} a_0 & a_1 & \cdots & a_{N-1} \\ a_{N-1} & a_0 & \cdots & a_{N-2} \\ a_{N-2} & a_{N-1} & \cdots & a_{N-3} \\ \vdots & \vdots & & \vdots \\ a_1 & a_2 & \cdots & a_0 \end{bmatrix}$$

它的对角线元素为常数。令

$$\mathbf{z}=\mathbf{W}\tilde{\mathbf{y}}$$
$$\boldsymbol{\gamma}=\mathbf{W}\boldsymbol{\beta}$$

且

$$\boldsymbol{\xi}=\mathbf{W}\tilde{\boldsymbol{\varepsilon}}$$

则式(B)变换为

$$\mathbf{z}=\mathbf{A}\boldsymbol{\gamma}+\boldsymbol{\xi} \tag{C}$$

对角矩阵 $\mathbf{D}$ 具有矩阵“最大解耦”(maximally uncoupled) 行，而轮换矩阵 $\mathbf{A}$ 具有“最大耦合”(maximally coupled) 行。

按照上述变换，我们可以陈述广义交叉验证等价于将式(A)所示的岭回归问题变换为式(C)所示的最大耦合形式，然后对 $\mathbf{z}$ 进行一般的交叉验证，最后将其变换为原坐标系统 (Wahba, 1990)。

5. *维基百科验证*。对于一个咖啡杯变形为一个汽车轮胎的连续过程或反过程，可访问*维基百科网页*，并搜索“同构”。

习题

Green 函数

7.1 *薄板样条函数*由下式描述：

$$\varphi(r)=\left(\frac{r}{\sigma}\right)^2\log\left(\frac{r}{\sigma}\right).$$

对于某个 $\sigma>0$ 及 $r\in\mathbb{R}$。

可以验证使用此函数作为一个平移和旋转的变形 Green 函数。

7.2 高斯函数是仅有的可因式分解的径向基函数。利用高斯函数的这个性质证明定义为多元高斯分布的 Green 函数 $G(\mathbf{x},\mathbf{t})$可分解成：

$$G(\mathbf{x},\mathbf{t})=\prod_{i=1}^{m}G(x_i,t_i)$$

其中 x_i 和 t_i 是 $m\times 1$ 维向量 $\mathbf{x}$ 和 $\mathbf{t}$ 的第 i 个分量。

7.3 在第 5 章中，我们认为三种径向基函数：高斯函数、逆超二次函数和超二次函数，都满足 Micchelli 定理。但是，Green 函数类仅包含前两个径向基函数。解释为何 Green 函数类不包含超二次函数。

正则化网络

7.4 考虑代价泛函：

$$\mathscr{C}(F^*) = \sum_{i=1}^{N}\left[d_i - \sum_{j=1}^{m_1} w_i G(\| \mathbf{x}_j - \mathbf{t}_i \|)\right]^2 + \lambda \| \mathbf{D}F^* \|^2$$

它用到逼近函数：

$$F^*(\mathbf{x}) = \sum_{i=1}^{m_1} w_i G(\| \mathbf{x} - \mathbf{t}_i \|)$$

利用 Frèchet 微分，证明当

$$(\mathbf{G}^T\mathbf{G} + \lambda\mathbf{G}_0)\,\hat{\mathbf{w}} = \mathbf{G}^T\mathbf{d}$$

时，代价泛函$\mathscr{C}(F^*)$最小，其中 $N \times m_1$ 维矩阵 $\mathbf{G}$，$m_1 \times m_1$ 维矩阵 $\mathbf{G}_0$，$m_1 \times 1$ 向量$\hat{\mathbf{w}}$以及 $\mathbf{N} \times 1$ 向量 $\mathbf{d}$，分别由式(7.53)、式(7.56)、式(7.54) 及式(7.27) 定义。

7.5 考虑一个定义如下的正则化项：

$$\int_{\mathbb{R}^{m_0}} \| \mathbf{D}F(\mathbf{x}) \|^2 \mathrm{d}\mathbf{x} = \sum_{k=0}^{\infty} a_k \int_{\mathbb{R}^{m_0}} \| D^k F(\mathbf{x}) \|^2 \mathrm{d}\mathbf{x}$$

其中

$$a_k = \frac{\sigma^{2k}}{k!\,2^k}$$

线性微分算子 D 由梯度算子∇和拉普拉斯算子∇^2 定义如下：

$$D^{2k} = (\nabla^2)^k$$

且

$$D^{2k+1} = \nabla(\nabla^2)^k$$

证明：

$$\mathbf{D}F(\mathbf{x}) = \sum_{k=0}^{\infty} \frac{\sigma^{2k}}{k!\,2^k} \nabla^{2k} F(\mathbf{x})$$

7.6 在 7.3 节中，我们由式(7.46)的关系导出了关于 $F_\lambda(\mathbf{x})$ 的式(7.47)。在这个习题中我们希望从由式(7.46)开始利用多维傅里叶变换导出式(7.47)。利用 Green 函数 $G(\mathbf{x})$的多维傅里叶变换的定义

$$G(\mathbf{s}) = \int_{\mathbb{R}^{m_0}} G(\mathbf{x}) \exp(-i\mathbf{s}^T\mathbf{x}) d\mathbf{x}$$

完成推导，其中 $i = \sqrt{-1}$，$\mathbf{s}$ 是 m_0 维的变换变量。关于傅里叶变换的性质可以参考相关内容。

7.7 考虑式(7.78)所描述的非线性回归问题。令 a_{ik} 表示矩阵 $(\mathbf{G} + \lambda\mathbf{I})^{-1}$ 的第 ik 个元素。那么，由式(7.39)出发，证明回归函数 $f(\mathbf{x})$的估计可以表示为

$$\hat{f}(\mathbf{x}) = \sum_{k=1}^{N} \psi(\mathbf{x}, \mathbf{x}_k) d_k$$

其中 d_k 是对应于模型输入 $\mathbf{x}_k$ 的输出，且

$$\psi(\mathbf{x}, \mathbf{x}_k) = \sum_{i=1}^{N} G(\| \mathbf{x} - \mathbf{x}_i \|) a_{ik}, \qquad k = 1, 2, \cdots, N$$

上式中 $G(\|\cdot\|)$是 Green 函数。

7.8 样条函数是分段多项式逼近器的例子 (Schumaker, 1981)。样条方法的基本思想如下：将一个被逼近区域用节点分为有限个子区域；节点可以是固定的，这样逼近器就是线性参数化的；节点也可以是可变的，这样逼近器就是非线性参数化的。在这两种情况下，在每一个逼近区域中使用一个阶数最高为 n 的多项式，且要求整个函数必须是 $n-1$ 次可微的。多项式样条函数是相对光滑函数，容易在计算机上存储、操作及计算。

在实际使用的样条函数中，三次样条函数可能是应用最广泛的。一个一维输入的三次样条函数的代价泛函定义如下：

$$\mathscr{C}(f) = \frac{1}{2}\sum_{i=1}^{N}[d_i - f(\mathbf{x}_i)]^2 + \frac{\lambda}{2}\int_{x_1}^{x_N}\left[\frac{\mathrm{d}^2 f(x)}{\mathrm{d}x^2}\right]\mathrm{d}x$$

其中 λ 在样条函数中表示光滑性参数。

(a) 验证这个问题解 $f_\lambda(x)$的如下性质：

(1) 两个相续的 x 节点值之间 $f_\lambda(x)$是一个三次多项式。

(2) $f_\lambda(x)$及前两阶导数都是连续的，除其二阶导数值在边界点为零外。

(b) 因为$\mathscr{C}(f)$有唯一最小值，所以我们必须有：

$$\mathscr{C}(f_\lambda + \alpha g) \geqslant \mathscr{C}(f_\lambda)$$

其中 g 是与 f_λ 一类的二次可微函数，α 为任意实值常数。这意味着$\mathscr{C}(f_\lambda+\alpha g)$作为 α 的函数在$\alpha=0$局部最小。因此，证明：

$$\int_{x_1}^{x_N}\left(\frac{\mathrm{d}^2 f_\lambda(x)}{\mathrm{d}x^2}\right)\left(\frac{\mathrm{d}^2 g(x)}{\mathrm{d}x^2}\right)\mathrm{d}x = \frac{1}{2}\sum_{i=1}^{N}[d - f_\lambda(\mathbf{x}_i)]g(\mathbf{x}_i)$$

上式是关于三次样条问题的欧拉拉格朗日方程。

7.9 式(7.75)定义了最小二乘方法的 Gram 矩阵或核矩阵 $\mathbf{K}$。证明此矩阵 $\mathbf{K}$ 是非负定的。

正则化最小二乘估计

7.10 由式(7.57)推出用于正则化最小二乘估计的式(7.65)。

7.11 证明等式(7.70)，其中包括数据矩阵 $\mathbf{X}$ 和预期响应向量 $\mathbf{d}$。

半监督学习

7.12 从带类标样本和无类标样本中学习是一个可逆的问题。证明此论断的有效性。

光谱图理论

7.13 在 7.13 节中，我们作出了如下论断：拉普拉斯矩阵 L 的最小特征值是零。使用式(7.113)中的 Rayleigh 系数来证明此论断。

广义表示定理

7.14 在式(7.122)中的最后一行，我们使用了表示定理的如下性质：

$$\left\langle \sum_{i=1}^{N} a_i k(\cdot,\mathbf{x}_i), k(\cdot,\mathbf{x}_j)\right\rangle = \sum_{i=1}^{N} a_i \langle k(\cdot,\mathbf{x}_i), k(\cdot,\mathbf{x}_j)\rangle$$

证明此性质。

7.15 式(7.120)中用于带类标和无类标样本的表示定理和式(6.83)中仅用于带类标样本的表示定理具有相同的数学形式。解释用于半监督学习的表示定理如何包含了用于监督学习的表示定理，且后者是前者的一个特例。

拉普拉斯正则化最小二乘算法

7.16 (a) 推出式(7.124)中的代价泛函。然后使用此泛函去推导式(7.125)中的最优点 a^*。

(b) 详细解释此最小点如何包含用于带类标样本的式(7.74)的最小点。且后者是前者的一个特例。

7.17 比较拉普拉斯正则化最小二乘算法的计算复杂度和仅使用带类标样本的正则化最小二乘算法的计算复杂度。

7.18 在求解最小二乘方法时，我们可以选择使用常规等式，或是用 7.6 节中讨论的表示定理。然而，在解关于此方法的半监督学习的版本时，表示定理是正确的选择。解释此论断的合理性。

7.19 实现拉普拉斯 RLS 算法需要使用一个 RBF 网络。讨论无类标样本和带类标样本在设计此网络的隐藏层和输出层时的独特作用。

计算机实验

7.20 带类标数据点的集合可以看成拉普拉斯 RLS 算法的初始化条件。像这样，对于一个给定的无类标训练样本，我们预期由算法构造的决策边界依赖于带类标数据点的位置。在此实验中，我们使用从图 1.8 中的双月构造中抽取的合成数据研究此相关性。

(a) 每个类一个带类标数据点。用与过去相同的条件，重复 7.16 节中的计算机实验，但此次实验探求决策边界是如何被两个带类标数据点的位置所影响。其中这两个数据点分别属于两个类。

(b) 每个类两个带类标数据点。用于 (a) 相同的设置，每个类中用两个带类标数据点，重复该实验。评价你此次实验的结果。

第8章
Neural Networks and Learning Machines, 3E

主分量分析

本章组织

本章的目的是描述如何使用非监督学习来实现主分量分析。本章组织如下：

8.1 节给出简介，强调了非监督学习的本章。

8.2 节描述自组织的四个原则：自增强、竞争、协作和结构化信息。这些原则在学习理解神经网络中特别重要。在视觉系统中，这些自组织特征原则的作用在 8.3 节中讨论。

8.4 节通过使用扰动理论给出主分量分析的数学背景。

接下来的两节解决两个基于 Hebb 的在线学习算法，8.5 节关注最大化特征滤波器（最强的主分量的提取）的 Oja 规则，8.6 节关注 Oja 规则的泛化。8.7 节中，把泛化规则应用于图像压缩。

8.8 节讨论核 PCA 算法，使得提取输入信号的高阶统计量成为可能。高阶统计量包含 8.9 节中讨论的自然图像的内在属性。为了能够以较好的计算复杂度处理自然图像的模型，8.10 节中描述通过广义 Hebb 算法对核 PCA 算法自适应的修正。8.10 节给出一个对多块图像去噪的实例学习。

8.11 节是本章的总结和讨论。

8.1 引言

神经网络的一个重要性质就是它们从环境中学习的能力。通过训练，从统计的意义上来提升性能。除了在第 7 章中讨论的半监督学习之外，前面的章节都关注监督学习算法。在监督学习中，训练样本包含一组有输入输出的样本。在本章和下面三章中，我们采取一个新的方向：我们学习非监督学习算法。

在无监督学习中，我们的目的是使用无类标的样本，发现输入数据中的显著模式或特征。也就是说，神经网络依照如下规则：

从实例中学习且不带教师。

无监督学习可以从两个不同的方面考虑：

（i）*自组织学习*，其从神经生物的角度考虑。特别地，半监督学习算法伴随着一系列局部行为规则，要求使用这些规则计算带有预期性质的输入输出映射。这里的局部是指对于神经网络中每个神经元突触权值的调整受到神经元局部邻居的限制。在此环境下，用于自组织学习的神经网络模型趋于神经生物学结构，使得网络组织与人脑相似。

（ii）*统计机器学习理论*，这是传统机器学习的方法。在神经网络中强调的局部学习的概念在机器学习中起到次要的作用。相反，在统计机器学习中，更强调数学工具。

在本章中，我们从这两方面学习主分量分析（PCA）。[1]PCA 是可以广泛用于统计模式识别和信号处理中维数约减的标准工具。

8.2 自组织原则

原则 1 自增强

自组织第一个原则就是：

神经元突触权值的修正随着 Hebb 条件学习自增强，这使得突触可塑性有了可能。

在单个神经元中，自增强的过程，受到以下约束：对神经元突触权值的修正基于在局部区域可

获得前突触和后突触信号。特别地，自增强和局部的要求通过强突触导致前突触和后突触信号的发生规定了一个反馈机制。相应地，该突触的强度通过此也增强了。此机制是 Hebb 学习的本质。

基于 Hebb 假定的学习是所有学习规则中最老的和著名的。它是为了纪念神经生物学家 Hebb (1949)。Hebb 的书《自组织行为》(1949) 有如下的描述 (p. 62)：

当一个神经元细胞 A 足够近地反复且持续地激活细胞 B，一些增长过程或新陈代谢会发生，使得 A 作为其中一激活 B 的细胞，其有效性增强。

Hebb 假定此在相关学习的基础上 (在细胞层次上) 的变化，会导致对在空间上分布的相似的神经细胞的激活模式持续地修正。

这个关于 Hebb 假定的学习的论断是基于一个神经生物学背景下的。我们可以对其扩展成两个规则 (Stent, 1973; Changeux and Danchin, 1976)：

1. *如果两个神经元中的一个突触连接是同时激活的 (即同步的)，则突触的强度会选择性地增强。*

2. *如果两个神经元中的一个突触是异步激活的，则突触会选择性地减弱或消除。*

这样一个突触叫做 Hebb 突触。[2] (原始的 Hebb 规则不包括 2。) 更准确地说，我们定义一个 Hebb 突触，使用依赖时间的、高度局部性的、强交互的机制来提高作为前突触和后突触相关性的函数的突触的有效性。从此定义中，我们可以推断出以下四个表示 Hebb 学习特征的关键机制 (Brown 等, 1990)：

1. *依赖于时间的机制。*此机制表示对 Hebb 突触的修改依赖于前突触和后突触信号发生的准确时间。

2. *局部机制。*一个突触自然地提供了在时空连接中的信息信号的变换。局部的可获得的信息可通过 Hebb 突触产生一个与输入有关的局部突触的修正。

3. *交互机制。*Hebb 突触的变换的发生依赖于突触每边的信号。这就是说，Hebb 学习依赖于前突触和后突触信号之间的交互，在此意义下，我们不能通过这两个突触本身作出预测。注意到这种依赖或交互关系在本质上是确定的和静态的。

4. *共轭或相关机制。*对于基于 Hebb 假定的学习的一种解释就是对于突触有效性的改变的条件是前突触或后突触信号的共轭。因此，根据此解释，前突触和后突触信号的发生 (在一个短暂的间隔内)，已足够产生突触修正。基于此原因，Hebb 突触有时也被称为共轭突触。对于基于 Hebb 假定的学习的另一个解释，我们可以考虑交互机制在统计的意义上是 Hebb 突触的重要特征。特别地，前突触信号和后突触信号之间的相关性被认为与突触变化有关。相关性实际上是学习的基础 (Chen 等, 2007)。

在数学意义上推导 Hebb 学习的表达式，考虑神经元 k 关于前突触和后突触信号的突触权值，其分别记为 x_j 和 y_k。对于突触权值在 n 时间的调整由如下通用公式表达：

$$\Delta w_{kj}(n) = f(y_k(n), x_j(n)) \tag{8.1}$$

其中 $f(\cdot,\cdot)$是一个关于前突触和后突触信号的函数。信号 $x_j(n)$和 $y_k(n)$通常被认为是没有维数的。式(8.1)有许多的形式[3]，全部都是 Hebb 的。因此，在式(8.2)中，我们考虑最简单的 Hebb 学习形式：

$$\Delta w_{kj}(n) = \eta y_k(n) x_j(n) \tag{8.2}$$

其中 η 是正常量，用于确定学习率。式(8.2)清晰地强调了 Hebb 突触的相关性。这有时称为*激活乘法规则*。从式(8.2)中，我们看见输入信号 x_j (前突触激活) 的反复应用导致了 y_k 的增强。因此，*指数性的增长*最终导致了突触连接的饱和。在此时，在突触中并不存储新的信息，选择性也消失了。因此需要一些机制来稳定神经元的自组织行为。这就要考虑到第二个原则。

原则 2　竞争原则

第二个自组织原则如下：

可用资源的局限性，以一种或另一种的形式，导致单个或一群神经元的突触之间的竞争。这个结果使得最强健增长的突触或神经元是以其他神经元作为代价的。

第二个原则通过突触可塑性实现（即突触权值的可调整性）。

为了使一个给定的神经元趋于稳定，它的突触之间必须要有对有限资源（如能量）的竞争，以此方式，神经元中一些突触的增强被其他突触的减弱所弥补。相应地，只有最成功的突触的强度可用于增长。那些不成功的突触就会趋于减弱，而最终消失。

在神经网络的级别，一个相似的竞争过程可能会通过以下过程发生（Rumelhart and Zipser,1985）：

- 神经网络中的神经元都相同，除了一些随机分布的突触权值；因此，神经元对于给定的输入模式的响应是不同的。
- 在神经网络中的每个神经元的强度（即突触权值的总和）都被附加了一个特定的限制。
- 神经元之间对于一个输入集按照给定的规则互相竞争；因此，每一次只有一个输出神经元，或每组中只有一个神经元是激活的。那些赢得竞争的神经元叫做“*胜者全得*”神经元。

我们因此发现，通过这些竞争学习过程，网络中的个体神经元对于不同类的输入模式承担了特征探测的作用。

在 Hebb 学习中，神经网络中一些输出神经元可能会被同时激活，在竞争学习的任何时候仅一个输出神经元或每组中一个输出神经元是激活的。竞争学习中的这个特征使得其非常合适发现统计上突出的特征，这可以被用来分类输入模式。

原则 3　协作

自组织的第三个原则如下：

在神经网络和网络的神经元级别中，对突触权值的修改趋于互相合作。

由于突触可塑性或由于在外部环境中的适当条件的存在而使得前突触神经元同时激活，而产生协作。

考虑到单个神经元的第一种情况：单个的突触不能有效地产生有利的事件。因此，必须有神经元突触之间的协作，才能够产生足够强的激活神经元的信号。

在网络层次，协作可能会通过一组激活的神经元之间的侧向交互而产生。特别地，一个激活的神经元更有可能促进它近邻的神经元而不是离它较远的神经元。在一段时间后，我们发现通过一系列小的变化，一个协作系统会趋于稳定状态。

同样也要注意到在自组织系统中，会同时出现竞争与协作，然而竞争总是优先于协作。

原则 4　结构化信息

第四条，也即最后一条原则如下：

在一个输入信号中存在的潜在次序和结构代表了冗余的信息，其通过一个自组织系统以知识的形式获得。

因此，可以说包含在输入数据中的结构化信息是自组织学习的前提条件。同样，也需要注意，自增强、竞争、协作是在神经元或神经网络中的过程，结构化信息或冗余是输入信号的内在性质。

比如说，我们考虑一个声音或视频信号。当这样一个信号以高比率取样，则样本信号相应地会呈现出较高程度的相关性。这里的相关性是指平均的，信号从一个样本到另一个之间的变

化并不剧烈。这也意味着这些信号含有结构化的冗余信息。换句话说，相关性是结构化和冗余的同义词。

要评价结构化的重要性，我们假定包含在一个信号中所有冗余信息都完全去除了。所剩下的只是不可预测的非冗余信息，因此这些信息可能无法与噪声区分开。考虑到这种类型的输入，非自组织或非监督学习系统会起到作用。

总结和附注

基于神经生物的自组织规则适用于神经网络中的非监督训练，但对于更为通用的用于执行非监督学习任务的机器学习却并不是必要的。在任意的学习任务中，非监督学习的目标是建立一个模型，使其适合于一组无类标数据，使数据中的潜在结构能够很好地表示出来。但为了使模型能够实现，数据必须是结构化的。

8.3　自组织的特征分析

视觉系统中的信息处理是分阶段的。具体地，一些简单的特征如对比度和边缘方向是在系统的早期阶段分析的，而更精致复杂的特征则在后期阶段进行分析。图 8.1 表示与视觉系统相似的模型网络的整体结构。在 Linsker 的模型中，图 8.1 的网络神经元组织成二维层，从一层到下一层具有局部前馈连接。每个神经元只接受前一层位于一个覆盖区内有限数目神经元的信息，此区域称为接受域（receptive field）。网络接受域在突触的形成过程中起关键作用，因为它们使一层中的神经元对前一层神经活动的空间相关性的反应成为可能。假设下面两个结构特征：

1. 在整个神经元形成过程中，一旦突触连接被选择，其位置就固定了。

2. 每个神经元都是一个线性组合器。

模型结合 Hebb 型突触修改的协作和竞争学习的方面使得网络输出最优地区分输入总体，这需要通过自组织学习从一层到一层的基础上处理。即学习过程在处理下一层之前允许全面形成该层自身的自组织特征－分析（feature analyzing）特性。

在 Linsker 模型中模拟结果与猫和猴子的视觉形成的早期具有非常相似的性质。认识到视觉系统的高度复杂性，而 Linsker 考虑的非常简单的模型却能形成相似的特征-分析神经元，这的确值得注意。这并非意味着哺乳动物的视觉系统的特征－分析神经元形成的方式与上面的 Linsker 模型描述的方式完全相同。相反，它只能说明按照 Hebb 学习规则形成突触权值，再由这种相对简单的层状网络就可产生这种结构，因此对自组织原则提供了实用的证明。

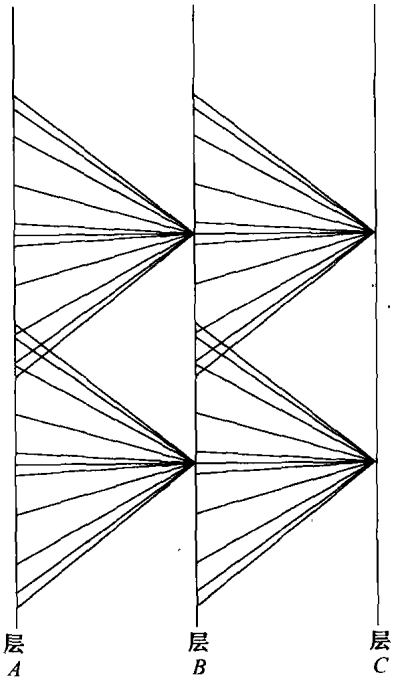


图 8.1　Linsker 模型的自适应层，各自的领域都有重叠

8.4　主分量分析：扰动理论

在统计模式识别中，一个常见的问题就是特征选择或特征提取。特征选择是指将数据空间变换到特征空间的过程，在理论上与原始数据空间具有相同的维数。然而，我们希望设计一种变换使得数据集由维数较少的“有效”特征来表示，而不减少原始数据所包含的内在信息内容；换句话说，数据集进行了维数压缩。具体来说，假设有一个 m 维的向量 $\mathbf{x}$，希望压缩到 l

维，其中 $l<m$。如果我们简单截断 $\mathbf{x}$，所带来的均方误差等于舍掉的各分量的方差之和。因此提出下面的问题：

是否存在一个可逆的线性变换 $\mathbf{T}$，使得对 $\mathbf{Tx}$ 的截断在均方误差意义下最优？

显然要求变换 $\mathbf{T}$ 后的某些分量具有较低的方差。*主分量分析*（principal components analysis，在通信理论中也叫 Karhunen-Loève 变换）能最大限度地减少方差，并因而是正确的选择。在本章我们讨论基于 Hebb 学习算法来完成数据向量的主分量分析。

令 $\mathbf{X}$ 为表示环境的 m 维随机向量。假设 $\mathbf{X}$ 均值为零，即：

$$\mathbb{E}[\mathbf{X}] = \mathbf{0}$$

其中$\mathbb{E}$是统计学习中的期望运算符。如果 $\mathbf{X}$ 的均值不是 0，在执行分析之前先减去其均值。令 $\mathbf{q}$ 表示 m 维单位向量，$\mathbf{X}$ 在其上投影。这个投影被定义为向量 $\mathbf{X}$ 和 $\mathbf{q}$ 的内积，表示为：

$$A = \mathbf{X}^T\mathbf{q} = \mathbf{q}^T\mathbf{X} \tag{8.3}$$

其满足约束条件：

$$\| \mathbf{q} \| = (\mathbf{q}^T\mathbf{q})^{1/2} = 1 \tag{8.4}$$

投影 A 也是随机变量，其均值和方差与 $\mathbf{X}$ 的统计有关。由假设 $\mathbf{X}$ 的均值为 0，推知 A 的均值也为 0：

$$\mathbb{E}[A] = \mathbf{q}^T\,\mathbb{E}[\mathbf{X}] = 0$$

方差与其均方值相同，可写为：

$$\sigma^2 = \mathbb{E}[A^2] = \mathbb{E}[(\mathbf{q}^T\mathbf{X})(\mathbf{X}^T\mathbf{q})] = \mathbf{q}^T\,\mathbb{E}[\mathbf{X}\mathbf{X}^T]\mathbf{q} = \mathbf{q}^T\mathbf{R}\mathbf{q} \tag{8.5}$$

$m\times m$ 矩阵 $\mathbf{R}$ 是随机向量 $\mathbf{X}$ 的*自相关矩阵*，定义为向量 $\mathbf{X}$ 和它自己的外积的期望，表示为：

$$\mathbf{R} = \mathbb{E}[\mathbf{X}\mathbf{X}^T] \tag{8.6}$$

我们观察到相关矩阵 $\mathbf{R}$ 是对称的，即

$$\mathbf{R}^T = \mathbf{R}$$

由这个性质知，如果 $\mathbf{a}$ 和 $\mathbf{b}$ 为任意 $m\times 1$ 向量，那么

$$\mathbf{a}^T\mathbf{R}\mathbf{b} = \mathbf{b}^T\mathbf{R}\mathbf{a} \tag{8.7}$$

由式(8.5)看出，投影 A 的方差 σ^2 是单位向量 $\mathbf{q}$ 的函数，可以写为：

$$\psi(\mathbf{q}) = \sigma^2 = \mathbf{q}^T\mathbf{R}\mathbf{q} \tag{8.8}$$

基于此我们可以认为 $\psi(\mathbf{q})$*为方差探针*（variance probe）。

主分量分析的特征结构

下面讨论的问题是在欧几里得范数的约束条件下，找出单位向量 $\mathbf{q}$ 沿 $\psi(\mathbf{q})$所具有的极值（extremal）或稳定值（stationary）（局部最大或最小）。这个问题的解决依赖于输入向量的相关矩阵 $\mathbf{R}$ 的特征结构。如果 $\mathbf{q}$ 为单位向量使得方差探针 $\psi(\mathbf{q})$具有极值，那么对单位向量 $\mathbf{q}$ 任意小的扰动 $\delta\mathbf{q}$，我们发现直到 $\delta\mathbf{q}$ 的一阶项将有

$$\psi(\mathbf{q}+\delta\mathbf{q}) = \psi(\mathbf{q})$$

现在，从式(8.8)给出的方差探针定义，我们有

$$\psi(\mathbf{q}+\delta\mathbf{q}) = (\mathbf{q}+\delta\mathbf{q})^T\mathbf{R}(\mathbf{q}+\delta\mathbf{q}) = \mathbf{q}^T\mathbf{R}\mathbf{q} + 2(\delta\mathbf{q})^T\mathbf{R}\mathbf{q} + (\delta\mathbf{q})^T\mathbf{R}\delta\mathbf{q}$$

在第 2 个等式中，已经利用式(8.7)。忽略项$(\delta\mathbf{q})^T\mathbf{R}\delta\mathbf{q}$ 并利用式(8.8)的定义，可以写成：

$$\psi(\mathbf{q}+\delta\mathbf{q}) = \mathbf{q}^T\mathbf{R}\mathbf{q} + 2(\delta\mathbf{q})^T\mathbf{R}\mathbf{q} = \psi(\mathbf{q}) + 2(\delta\mathbf{q})^T\mathbf{R}\mathbf{q} \tag{8.9}$$

$\psi(\mathbf{q}+\delta\mathbf{q})$是 $\psi(\mathbf{q})$的一阶近似；因此我们有：

$$(\delta\mathbf{q})^T\mathbf{R}\mathbf{q} = 0 \tag{8.10}$$

对 $\mathbf{q}$ 而言，任意扰动 $\delta\mathbf{q}$ 是不允许的；相反对扰动进行限制，仅使 $\mathbf{q}+\delta\mathbf{q}$ 的欧几里得范数为 1 的扰动是允许的，即：

$$\| \mathbf{q}+\delta\mathbf{q} \| = 1$$

或等价地：

$$(\mathbf{q}+\delta\mathbf{q})^T(\mathbf{q}+\delta\mathbf{q}) = 1$$

因此，根据式(8.4)，我们要求对 $\delta\mathbf{q}$ 的一阶项有：

$$(\delta\mathbf{q})^T\mathbf{q} = 0 \tag{8.11}$$

这意味着，扰动 $\delta\mathbf{q}$ 必须与 $\mathbf{q}$ 正交，因此仅在 $\mathbf{q}$ 的垂直方向上变化是允许的。

通常单位向量 $\mathbf{q}$ 在物理意义上是无量纲的。从而如果结合式(8.10)和式(8.11)，那么我们必须在式(8.11)中引入一个比例因子 λ 使得它和相关矩阵 $\mathbf{R}$ 中的元素有相同的量纲。于是可以写成：

$$(\delta\mathbf{q})^T\mathbf{R}\mathbf{q} - \lambda(\delta\mathbf{q})^T\mathbf{q} = 0$$

或等价于：

$$(\delta\mathbf{q})^T(\mathbf{R}\mathbf{q} - \lambda\mathbf{q}) = 0 \tag{8.12}$$

式(8.12)成立的充要条件为：

$$\mathbf{R}\mathbf{q} = \lambda\mathbf{q} \tag{8.13}$$

这个方程控制单位向量 $\mathbf{q}$ 使得方差探测值 $\psi(\mathbf{q})$ 有极值。

式(8.13)被认为是特征值问题，通常在线性代数中碰到（Strang,1980）。仅对特殊的 λ 值问题有非平凡解（即 $\mathbf{q}\neq\mathbf{0}$），λ 被称为相关矩阵 $\mathbf{R}$ 的特征值，对应的 $\mathbf{q}$ 被称为特征向量。相关矩阵的特征值必须是非负数。假设它的特征值互不相同，则对应的特征向量是唯一的。令 $m\times m$ 矩阵 $\mathbf{R}$ 的特征值为 $\lambda_1,\lambda_2,\cdots,\lambda_m$，对应的特征向量分别是 $\mathbf{q}_1,\mathbf{q}_2,\cdots,\mathbf{q}_m$。我们可写成：

$$\mathbf{R}\mathbf{q}_j = \lambda_j\mathbf{q}_j, \qquad j = 1,2,\cdots,m \tag{8.14}$$

令相应的特征值按降序排列，即：

$$\lambda_1 > \lambda_2 > \cdots > \lambda_j > \cdots > \lambda_m \tag{8.15}$$

这样 $\lambda_1=\lambda_{\max}$。令对应的特征向量用于构成一个 $m\times m$ 矩阵：

$$\mathbf{Q} = [\mathbf{q}_1,\mathbf{q}_2,\cdots,\mathbf{q}_j,\cdots,\mathbf{q}_m] \tag{8.16}$$

我们可以结合式(8.14)中的 m 个方程为一个方程组：

$$\mathbf{R}\mathbf{Q} = \mathbf{Q}\mathbf{\Lambda} \tag{8.17}$$

其中 $\mathbf{\Lambda}$ 为 $\mathbf{R}$ 的特征值构成的对角矩阵，即：

$$\mathbf{\Lambda} = \mathrm{diag}[\lambda_1,\lambda_2,\cdots,\lambda_j,\cdots,\lambda_m] \tag{8.18}$$

矩阵 $\mathbf{Q}$ 是正交（酉）矩阵，意味着它的列向量（即 $\mathbf{R}$ 的特征向量）满足正交条件：

$$\mathbf{q}_i^T\mathbf{q}_j = \begin{cases} 1, & j = i \\ 0, & j \neq i \end{cases} \tag{8.19}$$

式(8.19)要求不同的特征值。等价地，可写成：

$$\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$$

由此可以推导出矩阵 $\mathbf{Q}$ 的逆矩阵与它的转置矩阵相同，表示为：

$$\mathbf{Q}^T = \mathbf{Q}^{-1} \tag{8.20}$$

这意味着可以重写(8.17)为众所周知的正交相似变换形式：

$$\mathbf{Q}^T\mathbf{R}\mathbf{Q} = \mathbf{\Lambda} \tag{8.21}$$

或展开为：

$$\mathbf{q}_j^T\mathbf{R}\mathbf{q}_k = \begin{cases} \lambda_j, & k = j \\ 0, & k \neq j \end{cases} \tag{8.22}$$

式(8.21)的正交相似（酉）变换将相关矩阵 $\mathbf{R}$ 变成特征值对角阵。相关矩阵 $\mathbf{R}$ 可以用特征值和特征向量表示为：

$$\mathbf{R} = \sum_{i=1}^{m} \lambda_i \mathbf{q}_i \mathbf{q}_i^T = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^T \tag{8.23}$$

这称为谱定理。对所有 i，外积 $\mathbf{q}_i\mathbf{q}_i^T$ 的秩为 1。式(8.21)和式(8.23)是相关矩阵 $\mathbf{R}$ 的特征分解的两个等价表示。

主分量分析和矩阵 $\mathbf{R}$ 的特征分解从根本上来说是一致的，只是从不同的角度观察问题。从式(8.8)和式(8.22)可以看出方差探针和特征值的确相等，表示为：

$$\psi(\mathbf{q}_j) = \lambda_j, \qquad j = 1,2,\cdots,m \tag{8.24}$$

现在，从主分量分析的特征结构中可以概括两个重要发现：

- 零均值的随机向量 $\mathbf{X}$ 的相关矩阵 $\mathbf{R}$ 的特征向量定义为单位向量 $\mathbf{q}_j$，代表主方向，沿着它们方差探针 $\psi(\mathbf{q}_j)$取得极值。
- 相应的特征值定义方差探针 $\psi(\mathbf{q}_j)$的极值。

基本数据表示

令数据向量 $\mathbf{x}$ 为随机向量 $\mathbf{X}$ 的实例。用 a 表示随机变量 A 的一个实例。

由于单位向量 $\mathbf{q}$ 有 m 个可能的解，我们发现数据向量 $\mathbf{x}$ 有 m 个可能的投影需要考虑。特别地，从式(8.3)我们注意到：

$$a_j = \mathbf{q}_j^T\mathbf{x} = \mathbf{x}^T\mathbf{q}_j, \qquad j = 1,2,\cdots,m \tag{8.25}$$

其中 a_j 是 $\mathbf{x}$ 在单位向量 $\mathbf{q}_j$ 所表示的主方向上的投影。a_j 称作主分量，与向量 $\mathbf{x}$ 具有相同的物理量纲。式(8.25)的公式被看作是一个分析。

为了从投影 a_j 中准确重建原始数据向量 $\mathbf{x}$，我们可以采取下面的步骤。首先，将一组投影 $\{a_j | j = 1,2,\cdots,m\}$组合成一个单一的向量，表示为：

$$\mathbf{a} = [a_1, a_2, \cdots, a_m]^T = [\mathbf{x}^T\mathbf{q}_1, \mathbf{x}^T\mathbf{q}_2, \cdots, \mathbf{x}^T\mathbf{q}_m]^T = \mathbf{Q}^T\mathbf{x} \tag{8.26}$$

接着在式(8.26)的两边左乘矩阵 $\mathbf{Q}$，再利用式 $\mathbf{Q}\mathbf{Q}^T = \mathbf{I}$ 的关系。因此，原始数据向量 $\mathbf{x}$ 可重建为：

$$\mathbf{x} = \mathbf{Q}\mathbf{a} = \sum_{j=1}^{m} a_j \mathbf{q}_j \tag{8.27}$$

它可被看成合成公式。在这种意义上，单位向量 $\mathbf{q}_j$ 表示数据空间的一组基。确实，式(8.27)只是一个坐标变换，根据该变换数据空间中的点 $\mathbf{x}$ 变换到特征空间的点 $\mathbf{a}$。

维数约减

从统计模式识别的观点看，主分量分析的实际价值在于它为维数约减提供有效的方法。具体地讲，通过丢弃式(8.27)中方差小的项，保留方差大的项，可以减少有效数据表示所需的特征的数量。令 $\lambda_1, \lambda_2, \cdots, \lambda_l$ 表示相关矩阵 $\mathbf{R}$ 的前 l 个最大特征值。我们截断式(8.27)中的 l 项后面的展开式可以得到数据向量 $\mathbf{x}$ 的近似：

$$\hat{\mathbf{x}} = \sum_{j=1}^{l} a_j \mathbf{q}_j = [\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_l] \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_l \end{bmatrix}, \qquad l \leqslant m \tag{8.28}$$

对给定的原始数据向量 $\mathbf{x}$，可以用式(8.25)计算得到保留在式(8.28)中的主分量如下：

$$\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_l \end{bmatrix} = \begin{bmatrix} \mathbf{q}_1^T \\ \mathbf{q}_2^T \\ \vdots \\ \mathbf{q}_l^T \end{bmatrix} \mathbf{x}, \qquad l \leqslant m \tag{8.29}$$

从 $\mathbb{R}^m$ 到 $\mathbb{R}^l$ 的线性投影（即从数据空间到特征空间的映射）是对数据向量 $\mathbf{x}$ 近似表示的*编码器*，如图 8.2a 所示。相应地，从 $\mathbb{R}^l$ 到 $\mathbb{R}^m$ 的线性投影（即特征空间到数据空间的映射）表示为对原始数据向量 $\mathbf{x}$ 近似重构的解码器，如图 8.2b 所示。注意式(8.28)和式(8.29)中描述的优势（即最大）特征值 $\lambda_1,\lambda_2,\cdots,\lambda_l$ 并不参加计算，它们只是分别决定编码器和解码器所使用的主分量的数量。

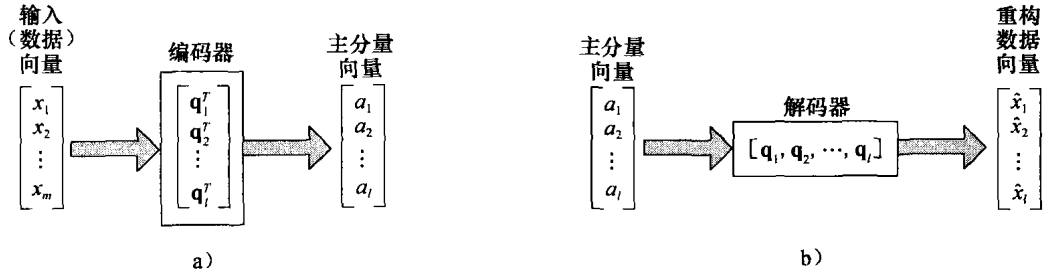


图 8.2　主分量分析的两阶段说明：a）编码；b）解码

逼近误差向量 $\mathbf{e}$ 等于原始数据向量 $\mathbf{x}$ 和逼近数据向量$\hat{\mathbf{x}}$的差，即：

$$\mathbf{e}=\mathbf{x}-\hat{\mathbf{x}} \tag{8.30}$$

将式(8.27)和式(8.28)代入式(8.30)得到：

$$\mathbf{e}=\sum_{i=l+1}^{m}a_i\mathbf{q}_i \tag{8.31}$$

误差向量 $\mathbf{e}$ 和*逼近数据向量*$\hat{\mathbf{x}}$是正交的，如图 8.3 所示。换句话说，$\hat{\mathbf{x}}$和 $\mathbf{e}$ 的内积为零。利用式(8.28)和式(8.31)，这个性质可以表示如下：

$$\mathbf{e}^T\hat{\mathbf{x}}=\sum_{i=l+1}^{m}a_i\mathbf{q}_i^T\sum_{j=1}^{l}a_j\mathbf{q}_j=\sum_{i=l+1}^{m}\sum_{j=1}^{l}a_ia_j\mathbf{q}_i^T\mathbf{q}_j=0 \qquad \text{对 } l<m \tag{8.32}$$

其中应用了式(8.19)的第二个条件。式(8.32)称作*正交性原理*。

由式(8.8)和式(8.22)的第一行，数据向量 $\mathbf{x}$ 的 m 个分量的总方差为：

$$\sum_{j=1}^{m}\sigma_j^2=\sum_{j=1}^{m}\lambda_j \tag{8.33}$$

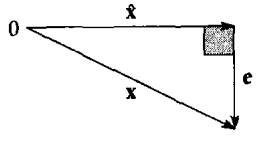


图 8.3　向量 $\mathbf{x}$、它的重建形式$\hat{\mathbf{x}}$和误差向量 $\mathbf{e}$ 的关系示例

其中 σ_j^2 是第 j 个主分量 a_j 的方差。逼近向量$\hat{\mathbf{x}}$的 l 个元素的总方差为：

$$\sum_{j=1}^{l}\sigma_j^2=\sum_{j=1}^{l}\lambda_j \tag{8.34}$$

在逼近误差向量 $\mathbf{x}-\hat{\mathbf{x}}$中的$(l-m)$个元素的总方差为：

$$\sum_{j=l+1}^{m}\sigma_j^2=\sum_{j=l+1}^{m}\lambda_j \tag{8.35}$$

特征值 $\lambda_{l+1},\cdots,\lambda_m$ 是相关矩阵 $\mathbf{R}$ 的特征值中最小的 $(m-l)$ 个特征值；在用于重构逼近向量$\hat{\mathbf{x}}$的式(8.28)中丢弃了它们所对应的项。这些特征值越接近 0，降维（对 $\mathbf{x}$ 进行主分量分析所导致的结果）后保存原始数据中的信息量就越有效。因此，为了对输入数据进行维数约减，我们：

计算输入数据向量的相关矩阵的特征值和特征向量，然后将原始向量投影到优势特征值对应的特征向量生成的子空间。

这种数据表示方法通常称为子空间分解（Oja,1983）。

例 1　双变量数据集

为了说明主分量分析的应用，考虑双变量（二维）数据集的例子，如图 8.4 所示，其中假设两个特征轴的标度近似相同。图中水平轴和垂直轴表示数据集的自然坐标轴。标号为 1 和 2 旋转坐标轴是应用这个数据集的主分量分析产生的结果。从图 8.4 可以看出数据集投影到 1 号轴上抓住了数据的主要特征，即具有双峰（即在它的结构上有两个聚类）的特点。的确，数据投影到轴 1 的方差比投影到其他轴上的要大。相反，当映射到轴 2 时，数据内在的双峰特征完全模糊。

从这个简单的例子中可以得到一个重要的结论。虽然，带有聚类结构的数据集在带有水平轴和垂直轴的二维平面图上很明显，但在实际中并不总是这样。在更一般的高维数据集中，可以想象数据固有的聚类结构被隐藏，要想看到它必须进行与主分量分析相似的统计分析（Linsker，1988a）。 ■

案例研究　数字图像压缩

主分量分析提供了数字图像压缩的一种简单有效的方法。对于存储容量、变换和特征提取的一个实用性上的要求就是图像是压缩的。图 8.5 所示的 PCA 使用实际数据，以验证此论断（Holmström 等，1997；Hyvärinen 等，2001）。

图 8.5 最左端显示了一组 10 个手写数字，即 0 到 9，每一个都用一个 32×32 的矩阵组成的二值图像表示。当每一个图像在一行一行的基础上扫描，就产生了一个 1 024×1 的向量。对于这 10 个数字中的每一个，大约 1 700 个手写字的样本被收集。样本均值（1 024×1 的向量）和协方差矩阵（1 024×1 024 的矩阵）使用标准方法估计。对于这 10 个手写字类的每一个，计算协方差矩阵的前 64 个主特征向量（分量）。图的第二行表示计算的样本均值。下面 6 列显示重构的图像，其指标 l 表示用式(8.28)重构图像时所使用的主分量的个数。在这些图像中，各自加上了样本均值，以合适的比例显示图像。

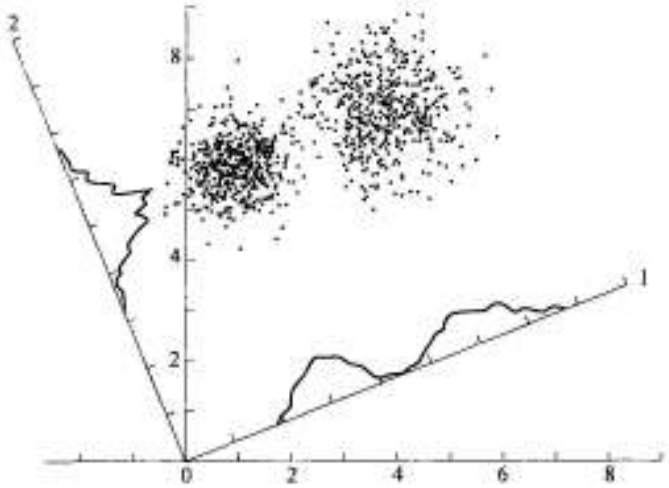


图 8.4　二维平面的一组数据，它们投影到两个轴 1 和 2 的密度图。投影到轴 1 有最大方差，清楚地表明数据的双峰或聚类特征



图 8.5　使用主分量分析的手写数字的压缩（这些图像的复制已得到了 Juha Karhunen 博士的允许）

由图 8.5 所示的 PCA 的结果，我们得到如下三点：

- 随着重构的大小 l 由 1，2，5，16，32，64 逐渐增长，重构的图像也与原始的 10 个手写字图像越来越相似。
- 当重构大小 $l=64$ 时，每一个重构的数字都非常清晰。

• 对于总共 1 024 个分量个数，最大的重构大小 $l=64$ 只是一个小的比例。

主分量个数的估计

在前面所讨论的数字图像压缩中，主分量的个数（即维数约减的大小）是在实验中确定的。对于这个估计问题的分析结果，我们可以把它看作一个模型选择问题。第 2 章所讨论的最小描述长度原则为解决此问题提供了一个好的测试方法。

在 Wax and Kailath (1985) 中，MDL 准则被用于阵列信号过程，即在有附加噪声的同时，确定一个达到信号的方向。为了解决这个问题，可以用 MDL 准则把输入数据空间分解成两个子空间，一个代表信号子空间，另一个代表噪声子空间。基本上，把输入数据空间分解成信号子空间和噪声子空间，同解一个维数约减问题是相同的。在此问题中，信号子空间的维数定义了响应于最大特征值的主特征向量（分量）的数量。

8.5　基于 Hebb 的最大特征滤波器

自组织神经网络的行为和主分量分析的统计方法之间存在密切的联系。在本节，我们将通过建立一个著名的结果来证实这个关系 (Oja, 1982)：

突触权值采用 Hebb 自适应规则的单个线性神经元能够形成关于输入分布第一个主分量的滤波器。

要继续这个证明，先考虑如图 8.6a 所示的简单神经元模型。该模型在模型输出为它的输入的线性组合这个意义下是线性的。神经元通过 m 个分别具有权值 $w_1, w_2, \cdots, w_m$ 的突触来接收 m 个输入信号 $x_1, x_2, \cdots, x_m$ 模型的输出结果：

$$y = \sum_{i=1}^{m} w_i x_i \tag{8.36}$$

注意这里描述的情形，我们仅处理单个神经元，所以不需要用双下标表示网络突触权值。

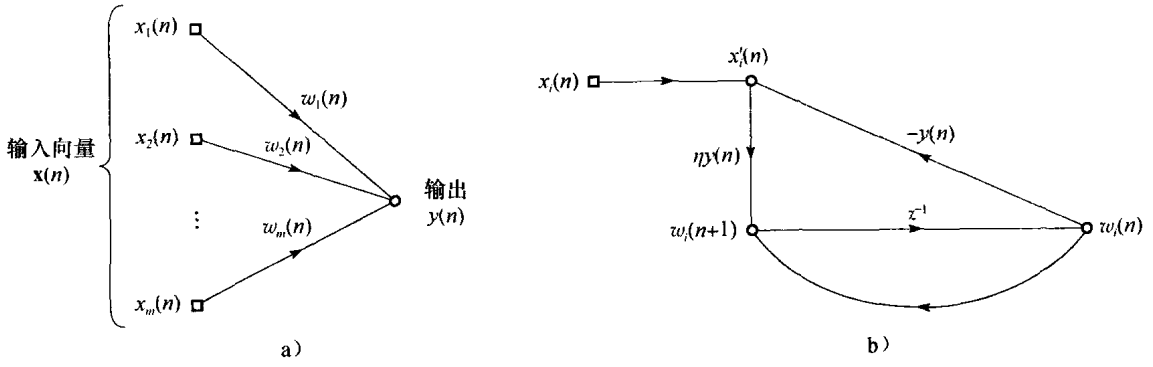


图 8.6　最大特征滤波器信号流图表示：a) 式(8.36)的图；b) 式(8.41)和式(8.42)的图

最大滤波器的推导

根据 Hebb 学习的假设，当前突触信号 x_i 和后突触信号 y 一致时，突触权值随时间逐步加强。具体可写成：

$$w_i(n+1) = w_i(n) + \eta y(n) x_i(n), \qquad i = 1, 2, \cdots, m \tag{8.37}$$

其中 n 表示离散时间，η 是*学习率参数*。但是，如 8.2 节所述的那样，这个学习规则的基本形式会导致突触权值 w_i 无限增大，这在现实上是不能接受的。在突触权值自适应学习规则中采用某种程度的饱和度或归一化，可以解决这个问题。利用归一化方法具有在神经元的突触权值间由于有限资源导致竞争的效果，从自组织的原则 2，这是稳定性的关键。从数学上来考

虑，式(8.37)方便的归一化形式描述如下：

$$w_i(n+1)=\frac{w_i(n)+\eta y(n)x_i(n)}{\left(\sum_{i=1}^{m}(w_i(n)+\eta y(n)x_i(n))^2\right)^{1/2}} \tag{8.38}$$

其中分母的求和是针对神经元的所有突触权值。假设学习率参数 η 很小，可以将式(8.38)展开成 η 的幂级数形式，所以写成：

$$\begin{aligned}\left(\sum_{i=1}^{m}(w_i(n)+\eta y(n)x_i(n))^2\right)^{1/2}&=\left(\sum_{i=1}^{m}(w_i^2(n)+2\eta w_i(n)y(n)x_i(n))\right)^{1/2}+O(\eta^2)\\&=\left(\sum_{i=1}^{m}w_i^2(n)+2\eta y(n)\sum_{i=1}^{m}w_i(n)x_i(n)\right)^{1/2}+O(\eta^2)\\&=(1+2\eta y^2(n))^{1/2}+O(\eta^2)\\&=1+\eta y^2(n)+O(\eta^2)\end{aligned} \tag{8.39}$$

在式(8.39)右边的第三行，我们使用以下约束：

$$\sum_{i=1}^{m}w_i^2(n)=\|\mathbf{w}(n)\|^2=1 \qquad \text{对所有 } n$$

以及此输入输出关系：

$$y(n)=\sum_{i=1}^{m}w_i(n)x_i(n)$$

另外，在式(8.39)的最后一行，我们在假定 η 较小的情况下使用如下的逼近公式：

$$(1+2\eta y^2(n))^{1/2}\approx 1+\eta y^2(n)$$

下面，用式(8.38)中的分子除以式(8.39)中分母的近似表示，再假定 η 很小，我们可以写出：

$$\begin{aligned}w_i(n+1)&=\frac{w_i(n)+\eta y(n)x_i(n)}{1+\eta y^2(n)+O(\eta^2)}\\&=(w_i(n)+\eta y(n)x_i(n))(1+\eta y^2(n)+O(\eta^2))^{-1}\\&=(w_i(n)+\eta y(n)x_i(n))(1-\eta y^2(n))+O(\eta^2)\\&=w_i(n)+\eta y(n)x_i(n)-\eta y^2(n)w_i(n)+O(\eta^2)\end{aligned}$$

合并常项，略去二阶项，最终写出：

$$w_i(n+1)=w_i(n)+\eta y(n)(x_i(n)-y(n)w_i(n)) \tag{8.40}$$

式(8.40)右端的项 $y(n)x_i(n)$表示突触权值通常的 Hebb 修改，这符合自组织原则 1 描绘的自放大效果。依据原则 2，该式中含有负项$-y(n)w_i(n)$导致稳定；它修改输入 $x_i(n)$成一种依赖于相应突触权值 $w_i(n)$和输出 $y(n)$的形式，表示为：

$$x_i'(n)=x_i(n)-y(n)w_i(n) \tag{8.41}$$

可以视为第 i 个突触的有效输入。由式(8.41)的定义可以重写式(8.40)的学习规则如下：

$$w_i(n+1)=w_i(n)+\eta y(n)x_i'(n) \tag{8.42}$$

神经元的整体操作可由两个信号流图的组合来表示，如图 8.6 所示。根据式(8.36)，图 8.6a 的信号流图表明输出 $y(n)$依赖于权值 $w_1(n),w_2(n),\cdots,w_m(n)$ 。图 8.6b 的信号流图提供式(8.41)和式(8.42)的图像；图中的传递参数 z^{-1}表示单位延迟操作符。在图 8.6a 中所产生的输出 $y(n)$在图 8.6b 中作为传递系数。图 8.6b 清楚地展示了作用于神经元的内部反馈的下列两种形式：

- 根据外部输入 $x_i(n)$，自放大的正反馈使得突触权值 $w_i(n)$增加。
- 由于$-y(n)$的负反馈控制 $w_i(n)$的增大，因此导致突触权值 $w_i(n)$的稳定。

乘积项$-y(n)w_i(n)$与在学习规则中经常用到的遗忘因子或泄漏因子有关，但存在差别：对

于较强的响应 y(n)，遗忘因子变得更加显著。这种控制现象有神经生物上的支持（Stent,1973）。

算法的矩阵形式

为了描述上的方便，令：

$$\mathbf{x}(n) = [x_1(n), x_2(n), \cdots, x_m(n)]^T \tag{8.43}$$

和

$$\mathbf{w}(n) = [w_1(n), w_2(n), \cdots, w_m(n)]^T \tag{8.44}$$

输入向量 $\mathbf{x}(n)$和突触权值向量 $\mathbf{w}(n)$通常都是随机向量的实现。用这个向量符号可以重写式(8.36)为内积形式如下：

$$y(n) = \mathbf{x}^T(n)\mathbf{w}(n) = \mathbf{w}^T(n)\mathbf{x}(n) \tag{8.45}$$

同样地，可以重写式(8.40)为：

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta y(n)[\mathbf{x}(n) - y(n)\mathbf{w}(n)] \tag{8.46}$$

将式(8.45)代入式(8.46)得：

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta[\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{w}(n) - \mathbf{w}^T(n)(\mathbf{x}(n)\mathbf{x}^T(n))\mathbf{w}(n)\mathbf{w}(n)] \tag{8.47}$$

式(8.47)所示的学习算法为*非线性随机差分方程*，这使得该算法的收敛性分析在数学上很难进行。为了得到收敛性分析，我们在假定学习参数 η 很小的前提下，先简单介绍随机逼近算法收敛分析的一般工具。

Kushner 的直接平均方法

检查自组织学习算法的式(8.47)右端，我们得到以下两点：

1. 输入向量 $\mathbf{x}(n)$以外积 $\mathbf{x}(n)\mathbf{x}^T(n)$的形式出现，表示协方差矩阵 $\mathbf{R}$ 的瞬时值，即式(8.6)中去掉期望算子且把 $\mathbf{x}(n)$当作随机向量 $\mathbf{X}(n)$的一个实现。实际上，$\mathbf{x}(n)\mathbf{x}^T(n)$可以表示此等式的随机行为。

2. 因为此算法是非监督的，故而此算法没有外部因素的作用。

由式(8.47)可知，算法的*特征均值*可以如下定义：

$$\mathbf{I} + \eta[(\mathbf{x}(n)\mathbf{x}^T(n)) - \mathbf{w}^T(n)(\mathbf{x}(n)\mathbf{x}^T(n))\mathbf{w}(n)\mathbf{I}] \tag{8.48}$$

其中 $\mathbf{I}$ 是单位矩阵。当此特征矩阵用旧的权值向量 $\mathbf{w}(n)$进行更新操作时，得到式(8.47)中新权值向量 $\mathbf{w}(n+1)$的更新公式。注意项 $\mathbf{w}^T(n)(\mathbf{x}(n)\mathbf{x}^T(n))\mathbf{w}(n)$是一个内积即标量，因此，我们必须用单位矩阵 $\mathbf{I}$ 乘以该项，以保证同式(8.48)剩下项之间的矩阵乘法的兼容性。

现在，请回忆第 3 章*最小均方算法*（LMS）中 Kushner *直接平均方法*，根据此方法，我们用以下的式来替换式(8.48)中的特征矩阵：

$$\mathbf{I} + \eta[\mathbf{R} - \mathbf{w}^T(n)\mathbf{R}\mathbf{w}(n)\mathbf{I}] \tag{8.49}$$

只要学习参数 η 很小，这个替换就是合理的。在 η 一定大的情况下，外积项，$\mathbf{x}(n)\mathbf{x}^T(n)$可以充当协方差矩阵 $\mathbf{R}$ 的角色。

因此，我们可以说只要 η 很小，式(8.47)的随机方程的解就与如下的非常简单的确定性差分解足够接近：

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta[\mathbf{R} - \mathbf{w}^T(n)\mathbf{R}\mathbf{w}(n)\mathbf{I}]\mathbf{w}(n) \tag{8.50}$$

我们令

$$\Delta\mathbf{w}(n) = \mathbf{w}(n+1) - \mathbf{w}(n)$$

用 t 表示连续时间，我们可以说权值在离散时间 n 的增量变化量 $\Delta\mathbf{w}(n)$与权值 $\mathbf{w}(t)$在连续时间 t 的变换率成比例。其比例关系如下式所示：

$$\frac{\mathrm{d}\mathbf{w}(t)}{\mathrm{d}t} \propto \Delta\mathbf{w}(n) \tag{8.51}$$

因此，把学习参数 η 代入式(8.51)中作为比例因子，并规范时间 t，我们可以通过如下的非线性常微分方程来描述最大特征滤波器的变化：

$$\frac{\mathrm{d}\mathbf{w}(t)}{\mathrm{d}t} = \mathbf{R}\mathbf{w}(t) - (\mathbf{w}^T(t)\mathbf{R}\mathbf{w}(t))\mathbf{w}(t) \tag{8.52}$$

其中二次项 $\mathbf{w}^T(t)\mathbf{R}\mathbf{w}(t)$是标量，使得在矩阵项中，方程的维数是正确的。

最大特征滤波器的渐近稳定性

根据相关矩阵 $\mathbf{R}$ 特征向量的完全正交集将 $\mathbf{w}(t)$展开成：

$$\mathbf{w}(t) = \sum_{k=1}^{m} \theta_k(t)\mathbf{q}_k \tag{8.53}$$

其中 $\mathbf{q}_k$ 是 $\mathbf{R}$ 的第 k 个归一化特征向量，系数 $\theta_k(t)$是向量 $\mathbf{w}(t)$在 $\mathbf{q}_k$ 上的时变投影。将式(8.53)代入式(8.52)，并应用 8.4 节中的基本定义：

$$\mathbf{R}\mathbf{q}_k = \lambda_k \mathbf{q}_k$$

和

$$\mathbf{q}_k^T \mathbf{R}\mathbf{q}_k = \lambda_k$$

其中 λ_k 是与 $\mathbf{q}_k$ 相关的特征值，最后得到：

$$\sum_{k=1}^{m} \frac{\mathrm{d}\theta_k(t)}{\mathrm{d}t}\mathbf{q}_k = \sum_{k=1}^{m} \lambda_k \theta_k(t)\mathbf{q}_k - \left[\sum_{l=1}^{m} \lambda_l \theta_l^2(t)\right] \sum_{k=1}^{m} \theta_k(t)\mathbf{q}_k \tag{8.54}$$

等价于

$$\frac{\mathrm{d}\theta_k(t)}{\mathrm{d}t} = \lambda_k \theta_k(t) - \theta_k(t) \sum_{l=1}^{m} \lambda_l \theta_l^2(t), \qquad k = 1,2,\cdots,m \tag{8.55}$$

从而我们将式(8.47)的随机逼近算法的收敛性分析归结为包含主模式 (principal mode) $\theta_k(t)$ 的常微分方程组 (8.55) 的系统稳定性分析。

修正 Langevin 公式

鉴于第 3 章中所谈论的自适应 LMS 滤波器，我们把与最大特征滤波器相关的式(8.55)看成不受外力驱动的 Langevin 公式的非线性修正形式，其理由如下：

(ⅰ) 我们说 Langevin 公式是修正的，是因为等式右端有正项 $\lambda_k\theta_k(t)$，其对等式放大；且这个放大项是基于 Hebb 规则的。

(ⅱ) Langevin 公式是非线性的，是因为第二项——$\theta_k(t)\sum_l \lambda_l\theta_l^2(t)$，其归因于最大滤波器突触之间的竞争。

(ⅲ) Langevin 公式没有外力驱动，是因为最大滤波器是自组织的。

因为没有外力驱动，故而不同于 LMS 滤波器，最大滤波器以渐进的方式绝对收敛。然而，非线性最大滤波器使得收敛行为的学习在数学上更为困难。

Langevin 等式的收敛性分析

依赖于对下标 k 所赋给的值，可分为两种情况。情况Ⅰ对应于 $1<k\leqslant m$。情况Ⅱ对应于 $k=1$；m 为 $\mathbf{x}(n)$和 $\mathbf{w}(n)$的维数。依次考虑这两种情况。

情况Ⅰ　$1<k\leqslant m$。

要处理这种情况我们定义：

$$\alpha_k(t) = \frac{\theta_k(t)}{\theta_1(t)}, \qquad 1 < k \leqslant m \tag{8.56}$$

首先假设 $\theta_1(t)\neq 0$，若初始值 $\mathbf{w}(0)$随机选取，概率 1 为真。对式(8.56)两边对时间 t 求导数得到：

$$\frac{\mathrm{d}\alpha_k(t)}{\mathrm{d}t}=\frac{1}{\theta_1(t)}\frac{\mathrm{d}\theta_k(t)}{\mathrm{d}t}-\frac{\theta_k(t)}{\theta_1^2(t)}\frac{\mathrm{d}\theta_1(t)}{\mathrm{d}t}$$

$$=\frac{1}{\theta_1(t)}\frac{\mathrm{d}\theta_k(t)}{\mathrm{d}t}-\frac{\alpha_k(t)}{\theta_1(t)}\frac{\mathrm{d}\theta_1(t)}{\mathrm{d}t},\qquad 1<k\leqslant m \tag{8.57}$$

其次，将式(8.55)代入式(8.57)，利用式(8.56)的定义并化简结果，得到

$$\frac{\mathrm{d}\alpha_k(t)}{\mathrm{d}t}=-(\lambda_1-\lambda_k)\alpha_k(t),\qquad 1<k\leqslant m \tag{8.58}$$

假设相关矩阵 **R** 的特征值互不相同且按降序排列，则有

$$\lambda_1>\lambda_2>\cdots>\lambda_k>\cdots>\lambda_m>0 \tag{8.59}$$

由此推知特征值之差 $\lambda_1-\lambda_k$ 为正，在式(8.58)中表示一个时间常数的倒数。所以，从情况 I 发现：

$$\alpha_k(t)\to 0,\quad t\to\infty,\quad \text{当 } 1<k\leqslant m \tag{8.60}$$

情况 II　$k=1$。

从式(8.57)可知，第二种情况由如下的微分方程描述：

$$\frac{\mathrm{d}\theta_1(t)}{\mathrm{d}t}=\lambda_1\theta_1(t)-\theta_1(t)\sum_{l=1}^{m}\lambda_l\theta_l^2(t)$$

$$=\lambda_1\theta_1(t)-\lambda_1\theta_1^3(t)-\theta_1(t)\sum_{l=2}^{m}\lambda_l\theta_l^2(t)$$

$$=\lambda_1\theta_1(t)-\lambda_1\theta_1^3(t)-\theta_1^3(t)\sum_{l=2}^{m}\lambda_1\alpha_l^2(t) \tag{8.61}$$

然而，从情况 I 我们知道，当 $t\to\infty$时，对于 $l\neq 1$，$\alpha_l\to 0$。因此，当 t 趋向无穷大时，式(8.61)右端的最后一项接近 0。忽略此项，式(8.61)简化为：

$$\frac{\mathrm{d}\theta_1(t)}{\mathrm{d}t}=\lambda_1\theta_1(t)[1-\theta_1^2(t)]\qquad \text{对 } t\to\infty \tag{8.62}$$

但是必须强调，只在渐进意义下式(8.62)才成立。

方程 (8.62) 表示*自治系统*（即系统不显式依赖于时间）。这样一种系统的稳定性最好由称为 Lyapunov 函数的*正定函数*处理，Lyapunov 函数的具体处理细节将在第 14 章介绍。令 **s** 表示自治系统的状态向量，$V(t)$表示系统的 Lyapunov 函数。如果满足下列条件，则系统的平衡状态是渐进稳定的：

$$\frac{\mathrm{d}}{\mathrm{d}t}V(t)<0,\quad \text{当 } \mathbf{s}\in\mathcal{U}-\bar{\mathbf{s}}$$

其中$\mathcal{U}$为 $\bar{\mathbf{s}}$ 的邻域。

对当前的问题，我们断言微分方程(8.62)有一个由下式所定义的 Lyapunov 函数：

$$V(t)=[\theta_1^2(t)-1]^2 \tag{8.63}$$

为了证实这个断言，必须证明 $V(t)$需要满足下面两个条件：

1. $\dfrac{\mathrm{d}V(t)}{\mathrm{d}t}<0$　　对于所有 t (8.64)

2. $V(t)$ 有最小值 (8.65)

在式(8.63)中对 t 求导得：

$$\frac{\mathrm{d}V(t)}{\mathrm{d}t}=4\theta_1(t)[\theta_1(t)-1]\frac{\mathrm{d}\theta_1(t)}{\mathrm{d}t}=-4\lambda_1\theta_1^2(t)[\theta_1^2(t)-1]^2,\quad t\to\infty \tag{8.66}$$

其中在第二个等式利用了式(8.62)。因为特征值 λ_1 是正的，从式(8.66)发现，当 t 趋近无穷大时，式(8.64)的条件为真。此外，从式(8.66)知 $V(t)$在 $\theta_1(t)=\pm 1$ 处具有最小值（即 $\mathrm{d}V(t)/\mathrm{d}t=0$），所以式(8.65)的条件也满足。因此我们可以用下列陈述结束情况 II 的分析：

$$\theta_1(t) \to \pm 1, \quad t \to \infty \tag{8.67}$$

根据式(8.67)中描述的结果和式(8.66)的定义，可以重新陈述式(8.60)中情况 I 的结果的最终形式：

$$\theta_k(t) \to 0, \quad t \to \infty \quad \text{当 } 1 < k \leqslant m \tag{8.68}$$

从情况 I 和 II 的分析作出的全面结论是两方面的：

- 式(8.47)描述的随机逼近算法仅主模式收敛于 $\theta_1(t)$，算法的其他所有模式将衰减为 0。
- 模式 $\theta_1(t)$收敛于 ± 1。

因此，渐进稳定性定理的条件 5 满足。特别地，依据式(8.53)的展开式，可以正式地如下陈述：

$$\mathbf{w}(t) \to \mathbf{q}_1, \quad t \to \infty \tag{8.69}$$

其中 $\mathbf{q}_1$ 是相关矩阵 $\mathbf{R}$ 的最大特征值 λ_1 对应的归一化特征向量。

最后，要确立式(8.69)的解只是式(8.52)的非线性常微分方程的一个局部渐进解（Lyapurnov 意义下的）。我们必须先满足如下的离散时间域的条件：

令 $\mathcal{B}(q)$表示式(8.52)的解附近的吸引域，则参数向量 $\mathbf{w}(n)$以概率 1 无限地进入吸引域 $\mathcal{B}(q)$的一个紧子集 $\mathcal{A}$。

（吸引域的概念在第 13 章中定义。）

为了满足此条件，我们必须证明对存在所有向量集合 $\mathcal{A}$的子集满足如下等式：

$$\lim_{n \to \infty} \mathbf{w}(n) = \mathbf{q}_1 \qquad \text{概率 1 经常是无穷的} \tag{8.70}$$

为了这样做，我们必须先证明参数向量列 $\mathbf{w}(n)$以概率 1 为界，这可通过硬性限制 $\mathbf{w}(n)$的项，使它们的幅度值小于阈值 a。我们可以定义 $\mathbf{w}(n)$的范数为

$$\| \mathbf{w}(n) \| = \max_j |w_j(n)| \leqslant a \tag{8.71}$$

令 $\mathcal{A}$是 $\mathbb{R}^m$ 的压缩子集，由一个范数小于等于 a 的向量集定义。可以直接证明（Sanger，1989b）。

如果 $\| \mathbf{w}(n) \| \leqslant a$，且常数 a 足够大，则 $\| \mathbf{w}(n+1) \| < \| \mathbf{w}(n) \|$ 以概率 1 成立。

于是，随着迭代次数 n 的增大，$\mathbf{w}(n)$将最终进入 $\mathcal{A}$内并以概率 1 留在内部。因为吸引域 $\mathcal{B}(\mathbf{q}_1)$包括所有有界范数的向量，因此有 $\mathcal{A} \in \mathcal{B}(\mathbf{q}_1)$。换句话说，条件 6 满足。

现在证明（在使用较小学习参数的情况下）随机逼近算法（8.47）将使 $\mathbf{w}(n)$以概率 1 收敛于特征向量 $\mathbf{q}_1$，$\mathbf{q}_1$ 是与相关矩阵 $\mathbf{R}$ 的最大特征值 λ_1 对应的特征向量。这不仅是算法的固定点，而且是唯一的渐进稳定点。

基于 Hebb 规则的最大特征滤波器的性质小结

刚才给出的收敛分析只证明，由式(8.40)或式(8.46)的自组织学习规则控制的单个线性神经元自适应地抽取平稳输入的第一个主分量。这第一个主分量对应于随机向量 $\mathbf{X}(n)$的相关矩阵的最大特征值 λ_1；事实上 λ_1 与模型输出 $y(n)$的方差有关，如下所示。

令 $\sigma^2(n)$表示随机变量 $Y(n)$的方差，$y(n)$表示 $Y(n)$的一次实现，即

$$\sigma^2(n) = \mathbb{E}[Y^2(n)] \tag{8.72}$$

其中由于输入均值为零，$Y(n)$具有 0 均值。在式(8.46)中令 $n \to \infty$并且利用 $\mathbf{w}(n)$趋向于 $\mathbf{q}_1$ 的事实，我们得到：

$$\mathbf{x}(n) = y(n)\mathbf{q}_1, \quad \text{当 } n \to \infty$$

利用这个关系，可以证明当迭代次数 n 趋向于∞时，方差 $\sigma^2(n)$趋向于 λ_1；参见习题 8.6。

总之，其运行由式(8.46)描述的基于 Hebb 的线性神经元以概率 1 收敛于一个固定点，它具有如下的特征（Oja，1982）：

1. 模型输出的方差趋向于相关矩阵 **R** 的最大特征值，表示为：

$$\lim_{n\to\infty}\sigma^2(n)=\lambda_1 \tag{8.73}$$

2. 模型的突触权值向量趋向相关的特征向量，表示为：

$$\lim_{n\to\infty}\mathbf{w}(n)=\mathbf{q}_1 \tag{8.74}$$

和

$$\lim_{n\to\infty}\|\mathbf{w}(n)\|=1 \tag{8.75}$$

这些结果均假设相关矩阵 **R** 是正定的，且 **R** 的最大特征值 λ_1 的重数为 1。这些结果也适用于具有 $\lambda_1>0$ 且重数为 1 的非负定相关矩阵 **R**。

例 2　匹配滤波器

考虑随机向量 **X**，**X** 的实现用 **x** 表示，令

$$\mathbf{X}=\mathbf{s}+\mathbf{V} \tag{8.76}$$

其中向量 **s** 为固定单位向量，**V** 表示噪声分量，具有零均值，协方差为 $\sigma^2\mathbf{I}$。**X** 的相关矩阵为：

$$\mathbf{R}=\mathbb{E}[\mathbf{X}(n)\mathbf{X}^T(n)]=\mathbf{s}\,\mathbf{s}^T+\sigma^2\mathbf{I} \tag{8.77}$$

因此相关矩阵 **R** 的最大特征值：

$$\lambda_1=1+\sigma^2 \tag{8.78}$$

对应的特征向量 $\mathbf{q}_1$ 为 $\mathbf{q}_1=\mathbf{s}$，容易证明在这种情况下这个解满足特征值问题：

$$\mathbf{R}\mathbf{q}_1=\lambda_1\mathbf{q}_1$$

因此，对于本例描述的情况，自组织线性神经元（收敛到它的稳定条件）充当一个匹配的滤波器，其冲击响应（由突触权值表示）与输入向量 $\mathbf{X}(n)$ 的信号分量 **s** 匹配。 ■

8.6　基于 Hebb 的主分量分析

上一节中基于 Hebb 的最大特征滤波器抽出输入的第一个主分量。这个单线性神经元模型的前馈网络型可以扩展到单层线性神经元的前馈网络，目的在于对输入进行任意大小的主分量分析（Sanger,1989b）。

广义 Hebb 算法

考虑如图 8.7 所示的前馈网络。假设具有下面两个结构属性：

1. 网络输出层的每个神经元是线性的。

2. 网络有 m 个输入和 l 个输出，它们都是指定的。另外，网络输出少于输入（即 $l<m$）。

网络接受训练的仅有突触权值集 $\{w_{ji}\}$，它们将输入层的源节点 i 和输出层计算节点 j 连接起来，其中 $i=1,2,\cdots,m$ 和 $j=1,2,\cdots,l$。

在时刻 n 神经元 j 对输入集 $\{x_i(n)\mid i=1,2,\cdots,m\}$ 的响应所产生的输出 $y_j(n)$ 由下式给出（参看图 8.8a）：

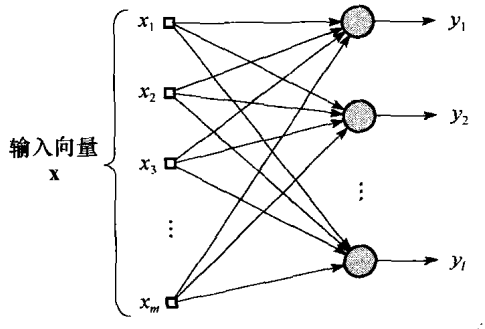


图 8.7　仅有单层计算节点的前向反馈网络

$$y_j(n)=\sum_{i=1}^{m}w_{ji}(n)x_i(n),\qquad j=1,2,\cdots,l \tag{8.79}$$

根据 Hebb 学习的广义形式，修改突触权值 $w_{ji}(n)$ 采用下式（Sanger,1989b）：

$$\Delta w_{ji}(n)=\eta\Big(y_j(n)x_i(n)-y_j(n)\sum_{k=1}^{j}w_{ki}(n)y_k(n)\Big),\qquad \begin{matrix}i=1,2,\cdots,m\\ j=1,2,\cdots,l\end{matrix} \tag{8.80}$$

其中 $\Delta w_{ji}(n)$是在时刻 n 对 $w_{ji}(n)$的修改，η 是学习率（Sanger，1989b）。注意在式(8.80)中，下标 i 指的是图 8.7 中网络的输入，而下标 j 指的是其输出。对于一层含有 l 个神经元的式(8.80)所示的广义 Hebb 算法（generalized Hebbian algorithm，GHA）包括上一节对单个神经元的式(8.40)的算法为其特殊情况（即 $j=1$）。

要对该算法的行为进行分析，将式(8.80)重新写成以下的形式：

$$\Delta w_{ji}(n) = \eta y_j(n)[x_i'(n) - w_{ji}(n)y_j(n)], \quad \begin{matrix} i = 1,2,\cdots,m \\ j = 1,2,\cdots,l \end{matrix} \tag{8.81}$$

其中 $x_i'(n)$为输入向量 $\mathbf{x}(n)$的第 i 个分量的修改形式；它是下标 j 的函数，表示为：

$$x_i'(n) = x_i(n) - \sum_{k=1}^{j-1} w_{ki}(n)y_k(n) \tag{8.82}$$

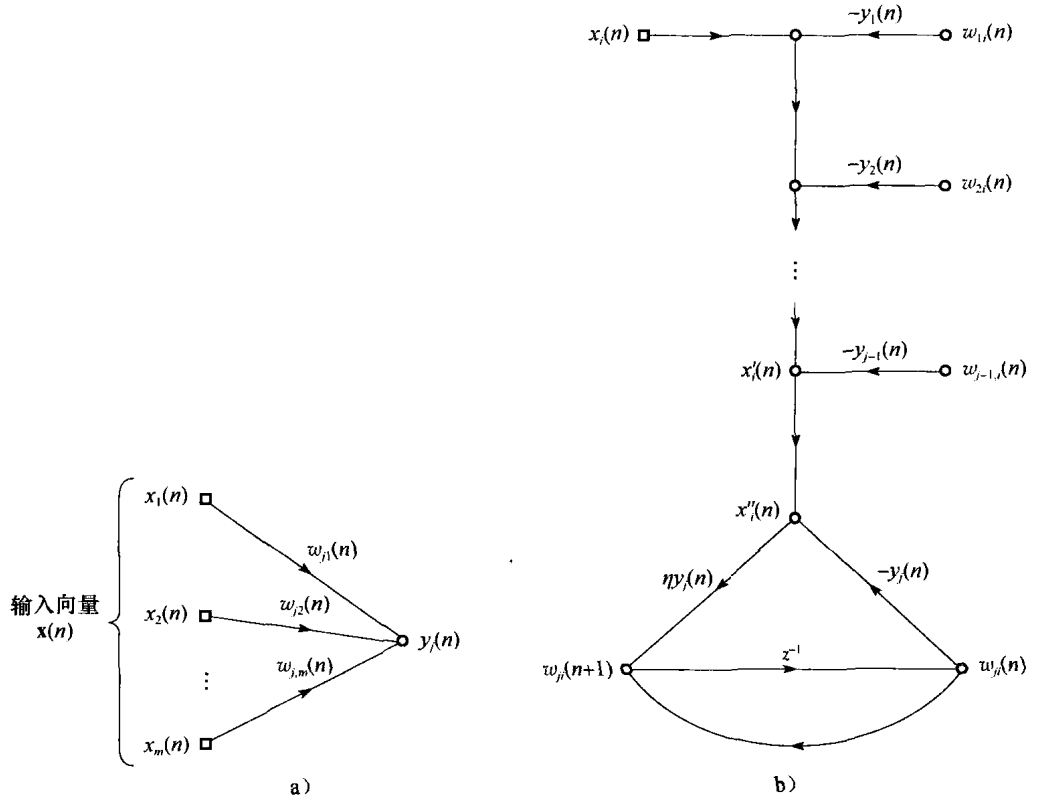


图 8.8　广义 Hebb 算法的信号流图表示：a) 式(8.79)的图；b) 式(8.80)到式(8.81)的图，其中 $x_i'(n)$和 $x_i''(n)$由式(8.82)和式(8.84)定义

对某个指定的神经元 j，式(8.81)表示的算法与式(8.40)表示的算法在数学形式上完全相同，只是将 $x_i(n)$变成了式(8.82)所定义的修改值 $x_i'(n)$。可以进一步将公式(8.81)重新写成 Hebb 的学习假设对应的形式，表示为：

$$\Delta w_{ji}(n) = \eta y_j(n)x_i''(n) \tag{8.83}$$

其中

$$x_i''(n) = x_i' - w_{ji}(n)y_j(n) \tag{8.84}$$

因此，注意到

$$w_{ji}(n+1) = w_{ji}(n) + \Delta w_{ji}(n) \tag{8.85}$$

和

$$w_{ji}(n) = z^{-1}[w_{ji}(n+1)] \tag{8.86}$$

其中 z^{-1}是单位延迟操作符，我们可以构建广义 Hebb 算法的信号流图，如图 8.8b 所示。从图中看出只要其公式由式(8.85)描述，则算法适合于该实现的局部形式。同时注意在图 8.8b 的信号流图中表示反馈的 $y_j(n)$由式(8.79)决定；它的信号流图表示在图 8.8a 给出。

为了帮助理解广义 Hebb 算法实际上如何操作，我们首先利用矩阵形式重写式(8.81)定义的算法如下：

$$\Delta \mathbf{w}_j(n) = \eta y_j(n)\mathbf{x}'(n) - \eta y_j^2(n)\mathbf{w}_j(n), \qquad j = 1,2,\cdots,l \tag{8.87}$$

其中 $\mathbf{w}_j(n)$是神经元 j 的突触权值向量，且：

$$\mathbf{x}'(n) = \mathbf{x}(n) - \sum_{k=1}^{j-1} \mathbf{w}_k(n) y_k(n) \tag{8.88}$$

向量 $\mathbf{x}'(n)$为输入向量 $\mathbf{x}(n)$的修正形式。基于式(8.87)给出的表示，我们得到下面的观察结果 (Sanger,1989b)：

1. 对于图 8.7 的前馈网络中的第一个神经元，我们有：

$$j = 1: \qquad \mathbf{x}'(n) = \mathbf{x}(n)$$

在这种情况下，广义 Hebb 算法相当于上一节的一个神经元的式(8.46)。由 8.5 节的描述，我们已经知道这个神经元将发现输入向量的第一个主分量。

2. 对于图 8.7 中的第 2 个神经元，我们写出：

$$j = 2: \qquad \mathbf{x}'(n) = \mathbf{x}(n) - \mathbf{w}_1(n) y_1(n)$$

如果第一个神经元已经收敛于第一个主分量，则第二个神经元看到一个输入向量 $\mathbf{x}'(n)$，从其中已经去掉相关矩阵 $\mathbf{R}$ 的第一个特征向量。因此第二个神经元抽取的是 $\mathbf{x}'(n)$的第一个主分量，相当于原来输入向量 $\mathbf{x}(n)$的第二个主分量。

3. 对于第 3 个神经元，我们写出：

$$j = 3: \qquad \mathbf{x}'(n) = \mathbf{x}(n) - \mathbf{w}_1(n) y_1(n) - \mathbf{w}_2(n) y_2(n)$$

假设前两个神经元已经分别收敛于第一个和第二个主分量，如前面两步的解释一样。第三个神经元的输入向量为 $\mathbf{x}'(n)$，从其中已经去掉相关矩阵 $\mathbf{R}$ 的前两个特征向量。因此第三个神经元抽取的是 $\mathbf{x}'(n)$的第一个主分量，相当于原来输入向量 $\mathbf{x}(n)$的第三个主分量。

4. 对于图(8.7)的前馈网络中剩下的神经元，继续执行上述过程。显然根据式(8.81)的广义 Hebb 算法训练的网络的每个输出代表对应于输入向量相关矩阵的某一特征向量的响应，并且这些输出按特征值递减排序。

这个计算特征向量的方法通称为 Hotelling 的紧缩技术 (Kreyszig,1988)；它类似于 Gram-Schmidt 正交化过程 (Strang,1980)。

收敛性考虑

令 $\mathbf{W}(n)=\{w_{ji}(n)\}$表示图 8.6 所示前馈网络的一个 $l\times m$ 的权值矩阵，即

$$\mathbf{W}(n) = [\mathbf{w}_1(n), \mathbf{w}_2(n), \cdots, \mathbf{w}_l(n)]^T \tag{8.89}$$

令广义 Hebb 算法的学习率参数 η 随着时间变化而变化，即 $\eta(n)$，限制条件为：

$$\lim_{n\to\infty} \eta(n) = 0 \quad \text{且} \quad \sum_{n=0}^{\infty} \eta(n) = \infty \tag{8.90}$$

可以将算法重新写成矩阵形式：

$$\Delta \mathbf{W}(n) = \eta(n)\{\mathbf{y}(n)\mathbf{x}^T(n) - \mathrm{LT}[\mathbf{y}(n)\mathbf{y}^T(n)]\mathbf{W}(n)\} \tag{8.91}$$

其中

$$\mathbf{y}(n) = \mathbf{W}(n)\mathbf{x}(n)$$

其中 LT[·]为下三角算子，它把矩阵对角线上方的所有元素置为 0，从而使矩阵成为下三角矩阵。在这些条件下以及采用 8.5 节的假设，则 GHA 算法收敛性证明的过程与上节关于最

大特征滤波器的收敛证明相似。因此有下面的定理（Sanger,1989b）：

如果权值矩阵 $\mathbf{W}(n)$ 在时间步 $n=0$ 时随机赋值，则式(8.91)所描述的广义 Hebb 算法以概率 1 收敛于固定点，且 $\mathbf{W}^T(n)$ 趋于一个矩阵，该矩阵的列分别为 $m\times 1$ 输入向量的 $m\times m$ 的相关矩阵 $\mathbf{R}$ 的前 l 个特征向量，按特征值的降序排列。

这个定理的实际价值在于，当对应特征值互不相同时它保证广义 Hebb 算法能够找到相关矩阵 $\mathbf{R}$ 的前 l 个特征向量。同样重要的是，我们不需要计算相关矩阵 $\mathbf{R}$，$\mathbf{R}$ 的前 l 个特征向量可直接由输入向量计算。特别是如果输入空间的维数 m 很大，而要求与 $\mathbf{R}$ 最大的 l 个特征值对应的特征向量的数目只是 m 的一小部分，则可以节省大量计算。

收敛定理是用时变学习率参数 $\eta(n)$ 表示的。实际上，学习率参数只能选择一个很小的固定常数 η，这样才能保证在 η 阶的突触权值的均方误差意义下收敛。

在 Chatterjee 等（1998）中，研究式(8.91)描述的 GHA 算法的收敛性质。那里给出的分析表明，η 增加将导致收敛速度加快，同时渐进均方误差也会增大；这在直观上也是符合的。除此之外，该论文对计算的精确性和学习速度之间的折中作了清楚的描述。

广义 Hebb 算法的最优性

假设在极限时写成：

$$\Delta\mathbf{w}_j(n)\to\mathbf{0}\quad 且\quad \mathbf{w}_j(n)\to\mathbf{q}_j,\quad n\to\infty,\quad 对\ j=1,2,\cdots,l \tag{8.92}$$

并且有

$$\|\mathbf{w}_j(n)\|=1,\quad 对所有\ j \tag{8.93}$$

那么在图 8.5 所示的前馈网络中，神经元的突触权值向量的极限值 $\mathbf{q}_1,\mathbf{q}_2,\cdots,\mathbf{q}_l$ 表示相关矩阵 $\mathbf{R}$ 的前 l 个特征值对应的归一化特征向量，按特征值的降序排列。在平衡时可写为：

$$\mathbf{q}_j^T\mathbf{R}\mathbf{q}_k=\begin{cases}\lambda_j, & k=j\\ 0, & k\neq j\end{cases} \tag{8.94}$$

其中 $\lambda_1>\lambda_2>\cdots>\lambda_l$。

对于神经元 j 的输出，我们有极限值：

$$\lim_{n\to\infty} y_j(n)=\mathbf{x}^T(n)\mathbf{q}_j=\mathbf{q}_j^T\mathbf{x}(n) \tag{8.95}$$

令 $Y_j(n)$ 表示一个随机变量，其实现记为输出 $y_j(n)$。在平衡时随机变量 $y_j(n)$ 和 $y_k(n)$ 的互相关为：

$$\lim_{n\to\infty}\mathbb{E}[Y_j(n)Y_k(n)]=\mathbb{E}[\mathbf{q}_j^T\mathbf{X}(n)\mathbf{X}^T(n)\mathbf{q}_k]=\mathbf{q}_j^T\mathbf{R}\mathbf{q}_k=\begin{cases}\lambda_j, & k=j\\ 0, & k\neq j\end{cases} \tag{8.96}$$

因此，我们可以说：在平衡时式(8.91)的广义 Hebb 算法充当输入数据的特征分析器。

令 $\hat{\mathbf{x}}(n)$ 表示输入向量 $\mathbf{x}(n)$ 的特定值，对于这个值，式(8.92)的极限条件对 $j=l-1$ 是满足的。因此，从式(8.80)的矩阵形式，我们发现在极限形式：

$$\hat{\mathbf{x}}(n)=\sum_{k=1}^{l}y_k(n)\mathbf{q}_k \tag{8.97}$$

这意味着给定两组值，即图 8.6 的前馈网络中神经元的突触权值向量的极限值 $\mathbf{q}_1,\mathbf{q}_2,\cdots,\mathbf{q}_l$ 和相应的输出 $y_1(n),y_2(n),\cdots,y_l(n)$，我们可以构造输入向量 $\mathbf{x}(n)$ 的线性最小平方估计 $\hat{\mathbf{x}}(n)$。实际上，如图 8.9 所描绘的式(8.97)的公式可视为一种数据重建。注意根据 8.4 节中的讨论，这种数据重建的方法导致逼近误差向量和估计 $\hat{\mathbf{x}}(n)$ 正交。

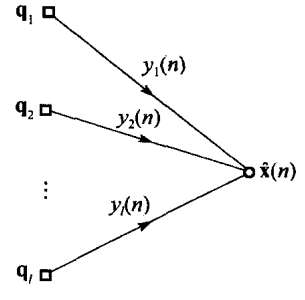


图 8.9　如何用 GHA 计算重建向量 $\hat{\mathbf{x}}$ 的信号流

GHA 小结

广义 Hebb 算法（GHA）所涉及的计算很简单，可以总结如下：

1. 在时间 $n=1$ 时，初始化网络突触权值 w_{ji}，使其取一个小的随机数。对学习率参数 η 赋给一个小的正数。

2. 对于 $n=1$，$j=1,2,\cdots,l$ 和 $i=1,2,\cdots,m$ 计算：

$$y_j(n)=\sum_{i=1}^{m}w_{ji}(n)x_i(n),\qquad j=1,2,\cdots,l$$

$$\Delta w_{ji}(n)=\eta\Big(y_j(n)x_i(n)-y_j(n)\sum_{k=1}^{j}w_{ki}(n)y_k(n)\Big),\qquad \begin{matrix}j=1,2,\cdots,l\\ i=1,2,\cdots,m\end{matrix}$$

其中，$x_i(n)$是 $m\times 1$ 输入向量 $x(n)$的第 i 个分量，l 是期望的主分量个数。

3. n 增加 1($n=n+1$)，转到第 2 步，并继续执行直到 w_{ji} 达到稳态值。对较大的 n，神经元 j 的突触权值 w_{ji} 收敛于输入向量 $\mathbf{x}(n)$的相关矩阵的第 j 个特征值对应特征向量的第 i 个分量。

8.7　计算机实验：图像编码

通过用广义 Hebb 学习算法解决图像编码问题完成对该算法的讨论。

图 8.10a 表示用于训练的一个 Lena 图像，该图像强调边缘信息。它被数字化为 256×256 的图像，分为 256 个灰度等级。利用一个具有 8 个神经元的单层线性前馈网络对图像编码，每个神经元有 64 个输入。利用 8×8 的非重叠图像块训练网络。试验扫描图像 2 000 次，学习率 $\eta=10^{-4}$。

图 8.10b 显示的 8×8 的掩模（mask）表示网络学习所得的突触权值。8 个掩模中的每一个为与某个特定的神经元相关的一组权值。具体地，兴奋（正）的权值用白色显示，抑制（负）的权值用黑色表示，灰色表示权值为 0。在我们的表示法中，掩模表示广义 Hebb 算法收敛后的 64×8 突触权值矩阵 $\mathbf{W}^{\mathrm{T}}$ 的列。

使用下面的步骤实现对图像编码：

- 图像的每个 8×8 块与图 8.10b 所示的 8 个掩模的每一个相乘，因此将产生 8 个系数作为图像编码；图 8.10c 显示没有量化的基于 8 个主分量的图像重建。
- 每个系数一律被量化为与该图像的系数方差的对数成正比的比特数。最大的 3 个掩模为每个 6 比特，其次的两个为每个 4 比特，再其次的两个为每个 3 比特，最小的一个为 2 比特。基于上述表示，需要 34 比特对每 8×8 的像素块编码，每个像素为 0.53 比特的数据率。

用量化系数重建图像，所有的掩模都用它们的量化系数加权，然后叠加重新构成的每块图像。以 11∶1 的压缩率重建的图像如图 8.10d 所示。



图 8.10　a) 用于图像编码试验的 Lena 图像；b) 8×8 的掩模表示由 GHA 学习的突触权值；c) 利用 8 个无量化主分量所得的 Lena 图像重建；d) 用量化的 11∶1 压缩比的 Lena 图像重建

作为第一个图像的变化，下面我们对图 8.11a 所示的辣椒图片应用广义 Hebb 算法。这幅图像强调纹理信息。图 8.11b 显示用前面描述的处理方式由网络学得的突触权值的 8×8 掩模图像；注意它们和 8.10b 的掩模的区别。图 8.11c 显示没有量化的基于 8 个主分量重建的海洋图像。为了研究量化的影响，令前两个掩模的输出每个为 5 比特，第 3 个为 3 比特，剩下的 5 个每个为 2 比特。这样需要 23 比特为每个 8×8 像素块编码，每个像素块的比特率为 0.36 比特每像素。图 8.11d 显示量化后重建的辣椒图像，使用其以刚才描述的方式量化的掩模。这幅图像的压缩比为 12∶1。

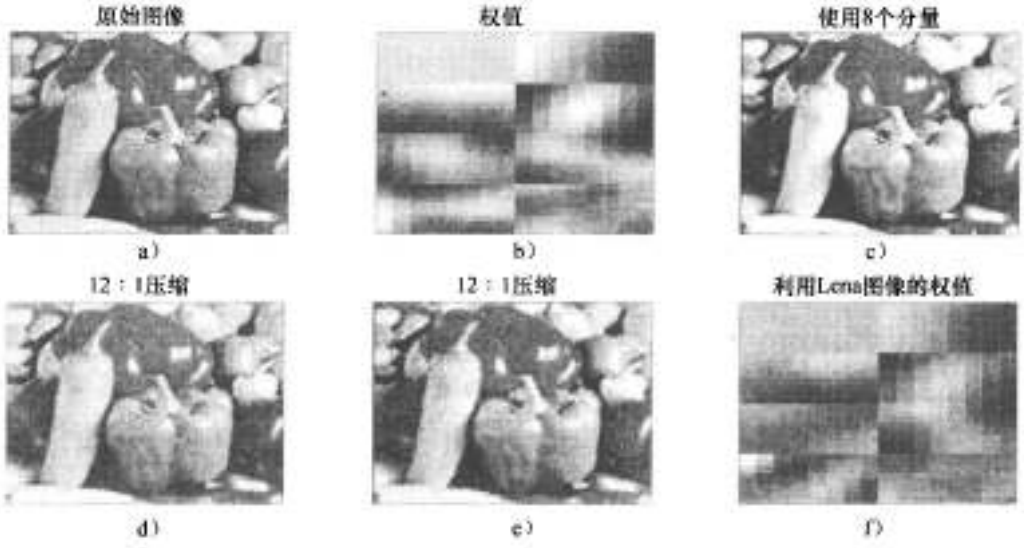


图 8.11　a）辣椒图像；b）8×8 的掩模表示由应用于辣椒图像的 GHA 学习到的突触权值；c）利用 8 个优势主分量重建的辣椒图像；d）利用 b）中的掩模以 12∶1 压缩比重建的；e）利用图 8.10b 中的掩模编码以 12∶1 压缩比量化重建的辣椒图像；f）复制图 8.10b 中的 Lena 掩模（即权值）

为了测试广义 Hebb 算法的“泛化”性能，最后用图 8.10b 的掩模分解图 8.11a 所示的辣椒图像，然后用与产生图 8.11d 所示重建图像一样的量化过程。这个图像重建结果如图 8.11e 所示，压缩比与 8.11d 一样，也为 12∶1。虽然在 8.11d 中的重建图像与在 8.11e 中的重建图像惊人地一致，但可以看到图 8.11d 比图 8.11e 更具有真实纹理信息而更少块状现象。产生这种情况的原因在于网络的权值。为了能够把对辣椒图像的在图 8.11b 中的掩模（权值）与对 Lena 图像的在图 8.10b 的掩模进行相比，我们在图 8.11f 中给出复制，并得到以下两个结论：

（1）它们的前 4 个突触权值很相似。

（2）然而，对 Lena 图像而言，后 4 个权值编码边缘信息，但在辣椒图像中，这 4 个权值编码纹理信息。

因此要点（2）解释了在图像（e）中与（d）相比的辣椒图像的块状现象。

8.8　核主分量分析

到目前为止本章讨论的 PCA 都是基于输入数据的二阶统计量（即相关性）；因此，标准 PCA 被称为线性维数压缩方法。然而，从实际的角度来看，我们需要把 PCA 的数据压缩能力拓展到结构中包含高阶统计量的输入数据。此拓展要求非线性的 PCA 算法。为了此目的，Scholkopf et al.（1998）设计了一种叫做核 PCA 的非线性版本的 PCA 算法。这个新的工具建立在第 6 章中所讨论的再生核 Hilbert 空间的基础之上。

在实现过程中，比较 GHA 和核 PCA 算法具有如下的指导意义：

1. GHA 使用了一个包含输入层和输出层的简单反馈网络；这个网络全部由线性神经元组成。核 PCA 同样使用一个反馈网络，但是这个网络包含了一个非线性的隐藏层和一个线性的输出层。

2. GHA 是一个在线学习算法，而核 PCA 是一个批量算法。

由于核 PCA 算法关联到隐藏层，此算法遵循第 6 章中在设计支持向量机中所讨论的理论。关于输出层，核 PCA 算法遵循标准 PCA 算法的维数压缩理论。因此，其名为“核 PCA”。

核 PCA 算法的推导

$\boldsymbol{\phi}$：$\mathbb{R}^{m_0} \rightarrow \mathbb{R}^{m_1}$ 表示从 m_0 维输入空间到 m_1 维特征空间的非线性映射。令向量 $\boldsymbol{\phi}(\mathbf{x}_i)$ 表示输入图像向量 $\mathbf{x}_i$ 在特征空间的特征向量。给定一组样本 $\{\mathbf{x}_i\}_{i=1}^N$，我们有一组相应的特征向量 $\{\boldsymbol{\phi}(\mathbf{x}_i)\}_{i=1}^N$。因此我们可以在特征空间定义由 $\widetilde{\mathbf{R}}$ 表示的外积形式为 $\boldsymbol{\phi}(\mathbf{x}_i)\boldsymbol{\phi}^T(\mathbf{x}_i)$ 的如下 $m_1 \times m_1$ 相关矩阵：

$$\widetilde{\mathbf{R}} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{\phi}(\mathbf{x}_i)\,\boldsymbol{\phi}^T(\mathbf{x}_i) \tag{8.98}$$

如同普通的 PCA，我们首先要做的就是确保特征向量 $\{\boldsymbol{\phi}(\mathbf{x}_i)\}_{i=1}^N$ 的集合具有零均值：

$$\frac{1}{N}\sum_{i=1}^{N} \boldsymbol{\phi}(\mathbf{x}_i) = \mathbf{0}$$

在特征空间上满足这个条件比在输入空间上更加困难；在习题 8.15 中我们描述一个过程来满足这个要求。假设特征向量已经聚集于中心，则可以在目前情况下改变式(8.14)，写成：

$$\widetilde{\mathbf{R}}\tilde{\mathbf{q}} = \tilde{\lambda}\tilde{\mathbf{q}} \tag{8.99}$$

其中 $\tilde{\lambda}$ 为相关矩阵 $\widetilde{\mathbf{R}}$ 的特征值，$\tilde{\mathbf{q}}$ 为对应的特征向量。我们注意对 $\tilde{\lambda} \neq 0$ 满足式(8.99)的所有特征向量落在特征向量 $\{\boldsymbol{\phi}(\mathbf{x}_j)\}_{j=1}^N$ 集合生成的空间中。因此存在一组相应的系数 $\{a_j\}_{j=1}^N$，用它们可写成：

$$\tilde{\mathbf{q}} = \sum_{j=1}^{N} a_j\,\boldsymbol{\phi}(\mathbf{x}_j) \tag{8.100}$$

由此将式(8.98)和式(8.100)代入式(8.99)得到：

$$\sum_{i=1}^{N}\sum_{j=1}^{N} a_j\,\boldsymbol{\phi}(\mathbf{x}_i)k(\mathbf{x}_i,\mathbf{x}_j) = N\tilde{\lambda}\sum_{j=1}^{N} a_j\,\boldsymbol{\phi}(\mathbf{x}_j) \tag{8.101}$$

其中 $k(\mathbf{x}_i, \mathbf{x}_j)$ 是**内积核**，通过特征向量由下式定义：

$$k(\mathbf{x}_i,\mathbf{x}_j) = \boldsymbol{\phi}^T(\mathbf{x}_i)\,\boldsymbol{\phi}(\mathbf{x}_j) \tag{8.102}$$

我们需要进一步计算式(8.101)以完全用内积核来表示此关系。在式(8.101)等号的两边左乘以转置向量 $\boldsymbol{\phi}^T(\mathbf{x}_s)$ 得：

$$\sum_{i=1}^{N}\sum_{j=1}^{N} a_j k(\mathbf{x}_s,\mathbf{x}_i)k(\mathbf{x}_i,\mathbf{x}_j) = N\tilde{\lambda}\sum_{j=1}^{N} a_j k(\mathbf{x}_s,\mathbf{x}_j), \qquad s = 1,2,\cdots,N \tag{8.103}$$

其中 $k(\mathbf{x}_s,\mathbf{x}_i)$，$k(\mathbf{x}_s,\mathbf{x}_j)$ 由式(8.102)定义。

现在引入下面两个矩阵定义：

- $N \times N$ 核矩阵 $\mathbf{K}$，称为核矩阵，它的第 ij 个元素为内积核 $k(\mathbf{x}_i,\mathbf{x}_j)$；
- $N \times 1$ 向量 $\boldsymbol{\alpha}$，第 j 个元素为参数 a_j。

因此，可以将式(8.103)写成紧凑的矩阵形式：

$$\mathbf{K}^2\boldsymbol{\alpha} = N\tilde{\lambda}\mathbf{K}\boldsymbol{\alpha} \tag{8.104}$$

其中矩阵的平方 $\mathbf{K}^2$ 表示 $\mathbf{K}$ 自身相乘。因为式(8.104)两端均有 $\mathbf{K}$，特征值问题感兴趣的全部解同样可用更为简单的特征值问题表示：

$$\mathbf{K}\boldsymbol{\alpha} = N\tilde{\lambda}\boldsymbol{\alpha} \tag{8.105}$$

令 $\lambda_1 \geqslant \lambda_2 \geqslant, \cdots, \geqslant \lambda_N$ 表示核矩阵 $\mathbf{K}$ 的特征值，即

$$\lambda_j = N\tilde{\lambda}_j, \qquad j = 1,2,\cdots,N \tag{8.106}$$

其中 j 是相关矩阵 λ_j 的第 j 个特征值。从而式(8.105)变成如下的标准形式：

$$\mathbf{K}\boldsymbol{\alpha} = \lambda\boldsymbol{\alpha} \tag{8.107}$$

其中系数向量 $\boldsymbol{\alpha}$ 起到核矩阵 $\mathbf{K}$ 的特征值 λ 的对应特征向量的作用。系数向量 $\boldsymbol{\alpha}$ 是归一化的，因为要求将相关矩阵 $\tilde{\mathbf{R}}$ 的特征向量 $\tilde{\mathbf{q}}$ 归一化为单位长度，即

$$\tilde{\mathbf{q}}_k^T\tilde{\mathbf{q}}_k = 1, \quad \text{当 } k = 1,2,\cdots,l \tag{8.108}$$

此处假设特征值 k 为降序排列，λ_l 为核矩阵 $\mathbf{K}$ 的特征值的最小非零值。利用式(8.100)和式(8.107)可以得到式(8.108)等价的归一化条件：

$$\boldsymbol{\alpha}_r^T\boldsymbol{\alpha}_r = \frac{1}{\lambda_k}, \qquad r = 1,2,\cdots,l \tag{8.109}$$

为了抽出主分量，需要计算特征向量 $\tilde{\mathbf{q}}_k$ 在特征空间上的投影如下：

$$\tilde{\mathbf{q}}_k^T\boldsymbol{\phi}(\mathbf{x}) = \sum_{j=1}^{N}\alpha_{k,j}\boldsymbol{\phi}^T(\mathbf{x}_j)\boldsymbol{\phi}(\mathbf{x}) = \sum_{j=1}^{N}\alpha_{k,j}k(\mathbf{x}_j,\mathbf{x}), \qquad k = 1,2,\cdots,l \tag{8.110}$$

其中向量 $\mathbf{x}$ 是“测试”点，$\boldsymbol{\alpha}_{k,j}$ 是矩阵 $\mathbf{K}$ 第 k 个特征值对应的特征向量 $\boldsymbol{\alpha}_k$ 的第 j 个系数。式(8.110)的投影是定义在 m_1 维的特征空间上的非线性主分量（nonlinear principal component）。

图 8.12 说明核 PCA 的基本思想，其中特征空间经过变换 $\boldsymbol{\phi}(\mathbf{x})$ 和输入空间是非线性相关的。图中的 a 和 b 部分分别称为输入空间和特征空间。图 8.12b 中的轮廓线表示在主特征向量上的投影为常数的线，特征向量用虚线箭头表示。在此图中，假设变换 $\boldsymbol{\phi}(\mathbf{x})$ 用下面的方式选择：在特征空间中数据点诱导的像聚集在特征向量沿线。图 8.12a 显示输入空间上对应特征空间的线性等值线的非线性等值线。注意我们有意没有在输入空间上画特征向量的原像，因为它甚至可能不存在（Scholkopf 等，1998）。

按照 Mercer 定理定义的内积核，我们在 m_1 维特征空间上执行普通的 PCA，维数 m_1 是设计参数。8.4 节描述的普通 PCA 的所有性质对核 PCA 均适用。尤其是，核 PCA 在特征空间上是线性的，但在输入空间上是非线性的。因此，所有可用普通 PCA 进行特征提取和数据压缩的领域，进行非线性扩展 PCA 也有意义。

在第 6 章我们提出了三种构造内积核的方法，它们是基于利用多项式、径向基函数和双曲函数，参见表 6.1。对给定的任务，如何选择最适合的核（即恰当的特征空间）是一个有待解决的问题。

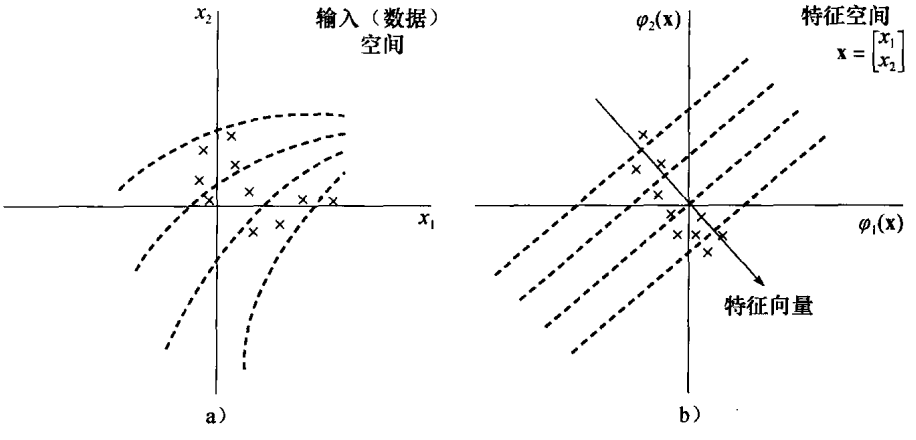


图 8.12　核 PCA 图例。a）二维输入空间，显示一组数据点；b）二维特征空间，显示数据点在一个主特征向量附近聚集的诱导像。在 b）中均匀排列的虚线表示在特征向量上投影为常数的等值线；它们在输入空间中的对应等值线是非线性的

核主分量分析小结

1. 给定训练样本$\{\mathbf{x}_i\}_{i=1}^N$，计算 $N\times N$ 核矩阵 $\mathbf{K}=\{k(\mathbf{x}_i,\mathbf{x}_j)\}$，其中

$$k(\mathbf{x}_i,\mathbf{x}_j)=\boldsymbol{\phi}^T(\mathbf{x}_i)\,\boldsymbol{\phi}(\mathbf{x}_j),\qquad i,j=1,2,\cdots,N$$

其中我们假定已经进行了数据预处理过程，使得训练样本的所有特征向量都满足零均值条件，即：

$$\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{\phi}(\mathbf{x}_i)=\mathbf{0}$$

2. 解特征值问题：

$$\mathbf{K}\boldsymbol{\alpha}=\lambda\boldsymbol{\alpha}$$

其中 λ 为 $\mathbf{K}$ 的特征值，$\boldsymbol{\alpha}$ 为对应的特征向量。

3. 归一化所计算的特征值，这要求

$$\boldsymbol{\alpha}_r^T\boldsymbol{\alpha}_r=\frac{1}{\lambda_r},\qquad r=1,2,\cdots,l$$

其中 λ_l 是矩阵 $\mathbf{K}$ 最小的非零特征值，假设特征值是按降序排列的。

4. 为了抽取测试点 $\mathbf{x}$ 的主分量，计算投影：

$$a_k=\tilde{\mathbf{q}}_r^T\,\boldsymbol{\phi}(\mathbf{x})=\sum_{j=1}^{N}\alpha_{r,j}k(\mathbf{x}_j,\mathbf{x}),\qquad r=1,2,\cdots,l$$

其中 $\alpha_{r,j}$ 是特征向量 $\boldsymbol{\alpha}_r$ 的第 j 个元素。

例 3　核 PCA 算法的事例试验

要对核 PCA 的运行有一个直观的了解，图 8.13 显示一个简单的实验结果（Schölkopf 等，

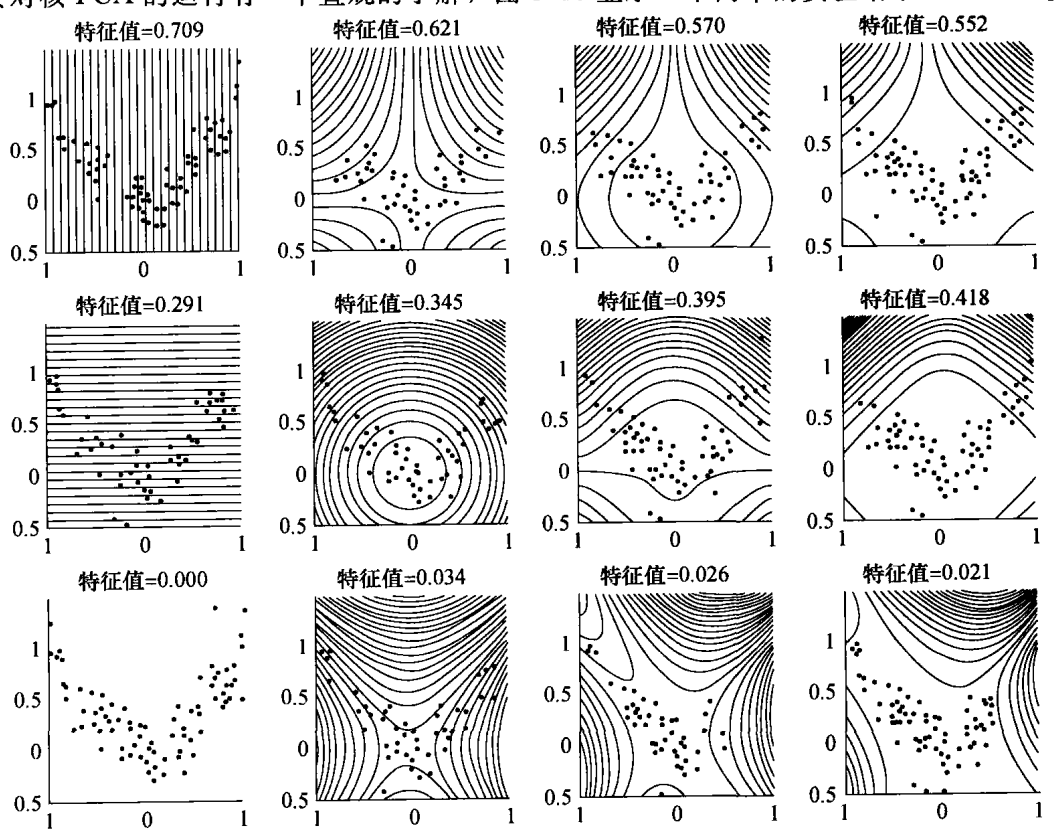


图 8.13　说明核 PCA 的二维示例。从左到右，核多项式的次数 $d=1$，2，3，4。从上到下，显示特征空间中的前面三个特征向量。第一列对应普通的 PCA，后三列对应多项式次数 $d=2$，3，4 的核 PCA（此图的复制经 Klaus-Robert Muller 博士允许）

1998)。二维数据由分量 x_1 和 x_2 组成，在这个试验中用下述方法产生：x_1 的值在区间 [−1, 1] 上均匀分布；x_2 的值与 x_1 非线性相关，由如下公式确定：

$$x_2 = x_1^2 + v$$

其中 v 是均值为 0 方差为 0.04 的附加高斯噪声。

图 8.13 所示的核 PCA 的结果可以由如下的核多项式得到：

$$k(\mathbf{x},\mathbf{x}_i) = (\mathbf{x}^T\mathbf{x}_i)^d, \qquad d = 1,2,3,4$$

其中 $d=1$ 对应线性 PCA，$d=2$，3，4 对应于核 PCA。线性 PCA 如图 8.13 左面所示，因为输入空间为二维，仅产生两个特征向量。相反，核 PCA 允许抽出高阶分量，结果如图 8.13 中的 2、3、4 列所示，分别与 $d=2$，3，4 对应。图中每部分显示的等值线（在线性 PCA 情形时除去零特征值）表示常数主值（即在与特征值相关联的特征向量上的投影为常数）。

根据图 8.13 显示的结果可得到如下结论：

- 如所期望的，线性 PCA 不能对非线性输入数据提供足够的描述。
- 在所有情况下，第一个主分量沿着构成输入数据的抛物线单调变化。
- 在核 PCA 中，对不同的多项式次数 d，第 2 和第 3 个主分量展示一定的相似性。
- 在多项式次数 $d=2$ 情况下，核 PCA 的第 3 个主分量显现出找到加性高斯噪声 v 的方差。消除这个主分量的影响，在效果上实际是执行某种形式的噪声消除。 ■

8.9　自然图像编码中的基本问题

在编码自然图像的过程中，有两个基本的策略。这两个策略都试图发觉图像的潜在结构中的内在冗余消息，以对潜在图像进行有效的表示。这两个策略是：

1. *压缩编码*。在这个编码策略中，图像变换成缩减数量的向量表示，并且编码受到规定的均方根误差的损失。主分量分析就是压缩编码的一个常见例子。

2. *稀疏分布编码*。在此第二个编码策略中，自然图像的维数并不约减。输入图像中的冗余信息以一种独特的方式变换，使得其与虚拟系统中神经元的激活模式的冗余信息相匹配。

在经典文章中（Field，1994），对这两种编码方法进行了对比。特别地，我们指出稀疏分布编码的特征可在自然图像潜在分布的*四阶矩*（即 kurtosis）中找到。PCA 是一个线性编码方法，依靠其函数的二阶统计量。因此它能够获得自然图像的四阶统计，这对于一个有效的编码策略是十分重要的。在 Field 的文章中提到了另一个关键的问题，就是形如*小波变换*[5] 的稀疏分布编码是有效的，因为对自然图像编码时，所得到的直方图呈现出高的峰度。另外，此文中指出，对于一阶逼近来说，自然图像的编码可以看作*自相似局部函数*的和（即小波变换的逆过程）。

现在，大都认为自然图像泛化的过程是非线性的（Ruderman，1997）。其中一个重要的因素是闭合，这本身是非线性的。在自然图像中的闭合图像轮廓有四个主要的来源（Richards，1998）：

- 外部闭合边；
- 皱褶或折叠；
- 阴影或亮度效果；
- 表面标记或纹理。

所有这四种图像轮廓依各自的方式提供了关于表面形状的信息。然而，关于推断哪一类边构造了图像轮廓的规则仍然有很大的不同。这给自然图像编码和解码的研究带来了挑战。

为了获得自然图像的高阶统计信息，显然，我们必须把非线性引入 PCA[6]。在下面的章节中，我们讨论一个能够实现此目标的有较高计算效率的自适应方法。

8.10　核 Hebb 算法

在前面几节的讨论中，我们知道：

高阶统计信息对于自然图像的结构编码（模型）是特别重要的。

另外，自然图像十分复杂，在这种情况下，包含在自然图像数字表示中的像素的数量会很高；这些像素的数量定义了图像空间的维数，其中每个样本图像仅被表示成一个点。因此，如果一个机器要学习自然图像模型，那么需要用大量的样本来训练此机器。

现在，回想到核 PCA 是一个批量学习算法，我们发现对于核矩阵的存储和操作达到 N^2 复杂度，其中 N 是训练样本的数量。因此，当需要模拟自然图像的时候，核 PCA 算法的时间复杂度是极大的。

为了降低计算复杂度，Kim 等(2005) 利用广义 Hebb 算法的非监督在线学习能力，设计了迭代算法，计算核 PCA。这个算法，叫核 Hebb 算法 (KHA)，能够在线性存储复杂度的条件下计算核主分量。不同于核 PCA，KHA 能够适应于非监督的大规模学习问题。

KHA 的推导

考虑训练样本 $\{\mathbf{x}_i\}_{i=1}^{N}$，我们可以推导 GHA 的更新规则，在特征空间中，如式(8.79)和式(8.80)所示，

$$y_j(n) = \mathbf{w}_j^T(n)\,\boldsymbol{\phi}(\mathbf{x}(n)), \qquad j = 1,2,\cdots,l \tag{8.111}$$

和

$$\Delta\mathbf{w}_j(n) = \eta\left[y_j(n)\,\boldsymbol{\phi}(\mathbf{x}(n)) - y_j(n)\sum_{p=1}^{j}\mathbf{w}_p(n)y_p(n)\right], \qquad j = 1,2,\cdots,l \tag{8.112}$$

我们选择 p 作为代替 k 的下标，以避免与核的标记 k 混淆。与前面相同，$\Delta\mathbf{w}_j(n)$和 $\mathbf{x}(n)$，分别在时间 n 时，对权值向量的更新和对输入向量的更新。η 是学习参数，下标 l 表示输出的数量。因为特征空间的高维数，我们可能无法直接使用式(8.112)。然而，从核 PCA 方法中，我们得知 $\mathbf{w}_j$ 可以由在特征空间中的训练样本展开，即：

$$\mathbf{w}_j = \sum_{i=1}^{N}\alpha_{ji}\,\boldsymbol{\phi}(\mathbf{x}_i) \tag{8.113}$$

其中 α_{ji} 是展开系数。使用式(8.111)和式(8.112)中的公式，可以得到如下两个更新规则：

$$y_j(n) = \sum_{i=1}^{N}\alpha_{ji}(n)\,\boldsymbol{\phi}^T(\mathbf{x}_i)\,\boldsymbol{\phi}(\mathbf{x}(n)), \qquad j = 1,2,\cdots,l \tag{8.114}$$

和

$$\sum_{i=1}^{N}\Delta\alpha_{ji}(n)\,\boldsymbol{\phi}(\mathbf{x}_i) = \eta\left[y_j(n)\,\boldsymbol{\phi}(\mathbf{x}(n)) - y_j(n)\sum_{i=1}^{N}\sum_{p=1}^{j}y_p(n)\alpha_{pi}\,\boldsymbol{\phi}(\mathbf{x}_i)\right], \qquad j = 1,2,\cdots,l \tag{8.115}$$

引入 Mercer 核的定义，有

$$k(\mathbf{x}_i,\mathbf{x}(n)) = \boldsymbol{\phi}^T(\mathbf{x}_i)\,\boldsymbol{\phi}(\mathbf{x}(n)), \qquad i = 1,2,\cdots,N$$

另外，我们可以规定如下两种可能的条件：

(ⅰ) 当 $\mathbf{x}(n)=\mathbf{x}_i$ 时，即训练集中输入向量 $\mathbf{x}(n)$的下标是 i。

(ⅱ) 在 $\mathbf{x}(n)\neq\mathbf{x}_i$ 时，即条件 (ⅰ) 不满足。

通过去除式(8.115)中外层关于下标 i 的求和，得到如下的关于系数$\{\alpha_{ji}\}$的更新规则 (Kim 等，2005)：

$$y_j(n) = \sum_{i=1}^{N}\alpha_{ji}(n)k(\mathbf{x}_i,\mathbf{x}(n)), \qquad j = 1,2,\cdots,l \tag{8.116}$$

和

$$\Delta\alpha_{ji}(n)=\begin{cases}\eta y_j(n)-\eta y_j(n)\sum_{p=1}^{j}\alpha_{pi}(n)y_p(n) & \text{如果 } \mathbf{x}(n)=\mathbf{x}_i\\ -\eta y_j(n)\sum_{p=1}^{j}\alpha_{pi}(n)y_p(n), & \text{如果 } \mathbf{x}(n)\neq\mathbf{x}_i\end{cases}\tag{8.117}$$

其中 $j=1,2,\cdots,l$ 且 $i=1,2,\cdots,N$。如同其他的核方法，我们在再生核 Hilbert 空间（RKHS）中实现 KHA。

对于核 PCA，必须保证核向量集$\{\boldsymbol{\phi}(\mathbf{x}_i)\}_{i=1}^{N}$具有零均值。习题 8.15 对于在批量算法中的问题给出了一种解决办法。对于如 KHA 之类的在线学习算法，必须使用一个滑动的均值去适应输入分布中的变化。

另一个就是关于 KHA 收敛性的问题。因为 KHA 是由 GHA 推导出来的，我们可以说，鉴于 8.6 节中关于收敛性的讨论，我们得知在学习参数 η 足够小的前提下，KHA 是局部收敛的。

案例研究：多块图像去噪

当说到复杂图像时，一个经常考虑的例子就是从自然场景图像中取块。当图像具有多个块时，对这样的图像建模就是极有挑战性的工作。事实上，在 8.7 节中讨论到的 Lena 图像就具有多个块，因此作为图像去噪学习中的基本实例。

这个实例学习由 Kim 等给出（2005）[7]，其把 KHA 与其余 6 种去噪方法比较。特别地，两种不同的 Lena 图像被构造：

（1）在 256×256 的 Lena 图像中加入白高斯噪声，产生了 7.72dB 的信噪比（SNR）。

（2）在同样的图像中加入椒盐噪声，制造出 4.92dB 的信噪比。

对于这两个图像中的每一个，我们在两个像素正则区间取 12×12 的有覆盖的图像块。

我们基于核 PCA 算法，假定高斯核的宽度为 1（$\sigma=1$），使用 KHA 算法（学习参数设立为 $\eta=0.05$）对通过带噪声的 Lena 图像数据大约 800 次扫描得到的样本建模。通过变化的参数 r，使用每一个核 PCA 模型中的前 r 个主分量去噪重构原始 Lena 图像。

为了方便比较，我们使用均值过滤方法[8]，Matlab 中的 Wiener 过滤法[9]，基于小波的方法和线性 PCA 算法来同去噪核 PCA 作比较。另外，以下两种方法被用来作比较：

- Pizurica 和 Philips 算法（Pizurica and Philips，2006），使用附加高斯噪声估计概率，此概率由小波子空间中的给定系数包含的无噪分量表出。
- Choi 和 Baraniuk 算法（Choi and Baraniuk，1999），通过把噪声信号投影到小波域的 Besov 空间[10]，获得原始信号的估计。

实验结果在图 8.14 和图 8.15 中给出，以下所有观察都由 Kim 等（2005）给出：

（1）对于图 8.14 中附加高斯白噪声（AWGN）情况下，由 Pizurica 和 Philips 算法得到的较好去噪效果和图 8.15 中附加椒盐噪声情况下，由均值过滤法得到的较好去噪效果可归因于使用了相关噪声源的统计信息这一先验知识。

（2）在另外一种情况下（即 Pizurica 和 Philips 算法在附加椒盐噪声情况下，以及均值过滤法在附加高斯白噪声情况下），这两种去噪方法的性能有所下降，这证明了依赖于先验知识的风险。

（3）如图 8.14 和图 8.15 所示，KHA 算法对于每种噪声效果都很好；这个结果说明如果没有关于附加噪声特点的信息时，KHA 算法是一个很好的选择。



图 8.14　对混入高斯噪声的图像去噪。a) 原始 Lena 图像；b) 加入噪声的图像；c) 均值过滤法；d) Matlab 中的小波去噪法；e) Matlab 中的 Wiener 过滤法；f) Choi 和 Baraniuk 算法；g) Pizurica 和 Philips 算法；h) PCA($r=20$)；i) KHA($r=40$)（此图片的复制得到了 K. I. Kim 博士的允许）

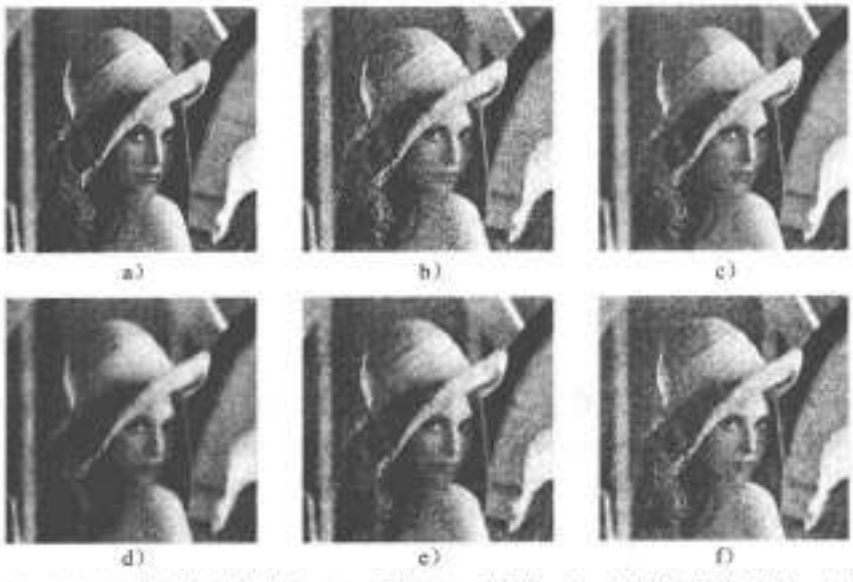


图 8.15　对混入椒盐噪声的图像去噪。a) 原始 Lena 图像；b) 加入噪声的图像；c) 均值过滤法；d) Matlab 中的小波去噪法；e) Matlab 中的 Wiener 过滤法；f) Choi 和 Baraniuk 算法；g) Pizurica 和 Philips 算法；h) PCA($r=20$)；i) KHA($r=20$)（此图片的复制得到了 K. I. Kim 博士的允许）



g)　　h)　　i)

图 8.15　(续)

最后，KHA 算法是一个在线非监督学习算法，因此具有两个额外的优点：

- 作为一个在线学习算法，计算复杂度较小。
- 作为非监督算法，不需要类标信息，避免了在监督学习中收集类标所耗费的时间和精力。

8.11　小结和讨论

在非监督学习中一个重要的问题就是，如何为学习过程设计一个性能评价或代价函数来产生一个起到监督作用的内部信号，使得网络能够预测或重建其本身的输入。在主分量分析中，代价函数是误差向量的均方值，这里误差向量定义为输入向量（假定为零均值）和重构向量之间的差别。我们的目标是在如下两个正交约束下关于一组自适应的系数最小化该代价函数：

(1) 规范化，即每个特征向量都是单位长度的；

(2) 正交性，即任意两个不同的特征向量相互正交。

习题 8.3 研究了用此方法来推导 PCA，作为 8.4 节中扰动理论的补充。

维数约简

PCA 算法最为重要的应用就是维数约简，其内容在式(8.28)和式(8.29)中总结了。为了讨论的方便，我们在此处重写这两个等式。

(1) 数据表示。给定一个 m 维的数据向量 $\mathbf{x}$，式(8.29)指出 $\mathbf{x}$ 可以由一个 l 维的主分量向量表示：

$$\mathbf{a}=\begin{bmatrix}a_1\\a_2\\\vdots\\a_l\end{bmatrix}=\begin{bmatrix}\mathbf{q}_1^T\\\mathbf{q}_2^T\\\vdots\\\mathbf{q}_l^T\end{bmatrix}\mathbf{x},\qquad l\leqslant m$$

其中 $\mathbf{q}_i$ 是如下的 $m\times m$ 协方差矩阵的第 i 个特征向量。

$$\mathbf{R}=\mathbb{E}[\mathbf{x}\mathbf{x}^T]$$

a_i 是向量 $\mathbf{a}$ 的第 i 个分量，是数据向量 $\mathbf{x}$ 在第 i 个特征向量 $\mathbf{q}_i$ 上的投影。如果 $l=m$，则新得到的向量 $\mathbf{a}$ 是原始数据向量 $\mathbf{x}$ 的旋转形式；且它们之间实质性的不同在于 $\mathbf{a}$ 有无关的分量，而 $\mathbf{x}$ 没有。如果 $l<m$，那么仅保留一个特征向量的子集，以用来近似地表示数据。在这种情况下，我们说是维数约简。

(2) 数据重构。给定主分量向量 $\mathbf{a}$，式(8.28)指出原始数据 $\mathbf{x}$ 可以由特征向量线性组合的形式被重构成：

$$\hat{\mathbf{x}}=\sum_{i=1}^{l}a_i\mathbf{q}_i,\qquad l\leqslant m$$

其中 $a_1, a_2, \cdots, a_l$ 是分量系数。这里，如果 $l=m$，则重构是准确的；而如果 $l<m$，那么重构是近似的。误差向量：

$$\mathbf{e} = \mathbf{x} - \hat{\mathbf{x}}$$

满足正交性的原则，即误差向量 $\mathbf{e}$ 与估计值 $\hat{\mathbf{x}}$ 正交。因此，我们说估计值 $\hat{\mathbf{x}}$ 是最小均方误差意义下的最优估计值（Haykin，2002）。确定约简维数 l 的一种最佳方法就是在第 2 章中讨论的最小描述长度（MDL）准则。

PCA 的一个在维数约简下的应用是去噪。在这个应用中，数据向量 $\mathbf{x}$ 由信号分量 $\mathbf{s}$ 和附加高斯白噪声 $\mathbf{v}$ 组成。我们的目标就是在最优意义下最小化噪声的影响。用 $\mathscr{X}$ 表示向量 $\mathbf{x}$ 所在的 m 维数据空间。给定 $\mathbf{x}$，PCA 把空间 $\mathscr{X}$ 分成两个互相正交的子空间：

- 信号子空间 $\mathcal{S}$。信号分量的估计值，由 $\hat{\mathbf{s}}$ 表示，位于此空间 $\mathcal{S}$ 中。估计值 $\hat{\mathbf{s}}$ 与 $\hat{\mathbf{x}}$ 在维数约简中起到相似的作用。
- 噪声子空间 $\mathcal{N}$。噪声分量的估计值，由 $\hat{\mathbf{v}}$ 表示，位于此空间 $\mathcal{N}$ 中。估计值 $\hat{\mathbf{v}}$ 与 $\mathbf{e}$ 在维数约简中起到相似的作用。

PCA 的另外一个应用是数据压缩。在此应用中，我们的目标是尽可能多地保存输入数据集中的信息。给定一个 m 维的数据向量 $\mathbf{x}$，PCA 通过对输入数据空间进行子空间分解实现此目标。此输入数据空间的前 l 个（小于 m）主分量提供一个线性映射。此映射在最小均方误差意义下是最优的，其对原始数据空间进行重构。另外，基于前 l 个（小于 m）主分量的表示比随意的子空间表示要好，因为输入空间的主分量是按特征值大小降序排列的，或者说按方差大小降序排列的。相应地，如在 8.7 节中的图像编码实例学习中讨论的那样，我们可以通过使用最大数值精度编码输入数据空间第一个主分量，而用较小的精度编码剩下的 $l-1$ 个分量，来最优地实现基于主分量分析的数据压缩。

关于无监督学习的两个观点

1. 自底向上的观点。局部性的概念在 8.2 节讨论的自组织的前三个原则（即自增强原则、竞争原则和合作原则）中起到重要的作用。这三个原则代表自下而上学习，其动机是形成一个学习过程模型。这样一个建模方法在无监督神经网络中已用到过。比如 8.5 节和 8.6 节分别讨论的 Hebb 最大特征过滤算法和广义 Hebb 算法。

另一方面，在导言里我们指出，机器学习中不强调局部性。这一缺乏自组织的特性反过来意味着非监督机器学习中自底向上的计算智能不起作用。

2. 自顶向下的观点。由通过自组织原则对无监督学习问题建模，我们转向自适应参数（即权值）的角度分析问题。具体地，给定一个无类标样本，我们在学习过程中施加的约束下最小化代价函数。第二阶段蕴含的理论就是自顶向下，就如同神经网络一样。最大特征过滤算法和广义 Hebb 算法的迭代公式就是无监督学习此观点的实例。

另一方面，机器学习限制无监督学习其本身为自顶向下的过程。为了弥补对自组织的强调的缺乏，在统计机器学习中分析工具已经被有效地使用。这个无监督学习方法的例子在 8.9 节中通过核 PCA 的方式给出。

不论无监督学习如何实现，它都是自顶向下的。其输入数据所包含的内在结构信息（自组织原则 4）可以被探查到。

神经生物的核算法

核算法，比如说核 PCA，是比较节省计算时间的。这是因为这些算法有能力处理包含在输入数据中的特定的高阶信息。但是，典型地，这些算法都遇到维数灾的问题。即此类问题的计算复杂度（由于各种各样的原因）随着输入数据空间的维数指数级增长。

例如，考虑图像压缩的问题。不幸的是，原始版本的核 PCA 的计算复杂度使得其在实际

图像问题（如人脸和自然图像）中应用受到局限。然而，通过对广义 Hebb 算法（GHA）核化，即在 8.10 节中讨论的核 Hebb 算法（KHA），我们得到一个迭代的无监督学习算法，可以在线性计算复杂度的条件下估计核主分量。正如 8.10 节中谈论到的图像去噪问题，证明了此类非监督算法的性能可以与监督学习算法的性能相比。因此，我们可以说，通过使用迭代的核化 PCA 算法，我们不仅在某种意义下避免了维数灾的问题，而且在仅使用无类标数据的情况下，就解决了图像去噪问题。

从此讨论中，我们得到一个有效的信息：

通过核化（基于统计学习理论）神经生物非监督信息算法，我们可以得到许多有用的东西。

在下一章中，关于神经生物学导出的自组织映射网络，我们将描述与本章不同的另一类核化的应用。

注释和参考文献

1. 在多元分析中，主分量分析（PCA）或许是最早的和最有名的方法（Jollife,1986；Preisendorfer,1988）。最早由 Pearson（1901）引入，在生物学背景下他用它来重建线性回归分析的新形式。后来 Hotelling（1933）在做心理测验时将它发展。Karhunen（1947）在概率论框架下再次独立地讨论了它；随后被 Loève（1963）推广。
2. *突触增强和抑制*。我们认识到正相关的行为有助于突触增强，而无关或负相关的行为导致突触减弱（Stent，1973）。基于此，我们推广 Hebb 修正的概念。突触减弱同样可能是一个非活性的类型。特别地，对于突触减弱的交互条件可能仅仅是非同时的先突触或后突触行为。

   我们可以进一步把突触修正规则分类成 Hebb 和逆 Hebb 规则（Palm，1992）。据此，一个 Hebb 突触随着正相关的前突触或后突触信号增强，随着无关或负相关信号而减弱。在 Hebb 和逆 Hebb 突触中，对突触修正的有效性依赖于一个与时间独立的、高度局部性的和有强交互性的机制。在此意义下，一个逆 Hebb 突触自然是 Hebb 突触的，尽管功能上说不是如此。另一方面，一个非 Hebb 突触，不包含 Hebb 机制中的任何一点。
3. *协方差假定*。一种克服 Hebb 假定局限性的方法就是使用在 Sejnowski（1977a，b）中介绍的协方差假定。在此假设中，在式(8.2)中的前突触和后突触信号被通过从它们各种在以特定时间段内的均值中分离出的前突触和后突触信号所替代。用 $\bar{x}$ 和 $\bar{y}$ 分别表示前突触信号 x_j 和后突触信号 y_k 的平均时间值。根据协方差假设，突触权值的调整值 w_{kj} 由此定义：

$$\Delta w_{kj} = \eta(x_j - \bar{x})(y_k - \bar{y}) \tag{A}$$

其中 η 是学习参数。均值 $\bar{x}$ 和 $\bar{y}$ 分别构成了前突触和后突触阈值，其决定了突触修正的符号。特别地，协方差假设有以下性质：

- 收敛到一个非频繁状态，即当 $x_k = \bar{x}$ 或 $y_j = \bar{y}$ 时达到收敛。
- 预测突触增强（即突触强度的增加）和突触衰弱（即突触强度的减少）。

图 A 给出了 Hebb 假定和协方差假定的不同。在这两种例子中，Δw_{kj} 关于 y_k 的依赖都是线性的；然而，在 Hebb 假定中，关于 y_k 轴的截距在原点，在协方差假定中，截距在 $y_k = \bar{y}$ 处。

从式(A)中得到如下的重要观点：

(1) 如果存在充分层次的前突触和后突触行为（即条件 $x_j > \bar{x}$ 和 $y_k > \bar{y}$ 同时满足），突触权值 w_{kj} 则会加强。

(2) 当以下两种条件之一成立时，突触权值 w_{kj} 衰减：

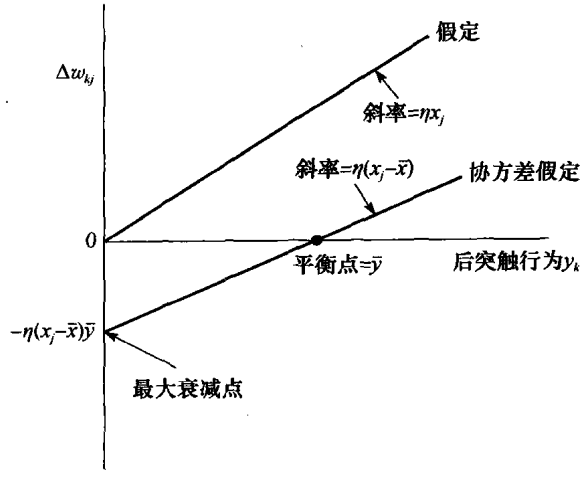


图 A　描述 Hebb 假定和协方差假定示例

- 一个前突触行为（即 $x_j > \bar{x}$）缺乏充分的后突触行为（即 $y_k < \bar{y}$）；
- 或一个后突触行为（即 $y_k > \bar{y}$）缺乏充分的先突触行为（即 $x_j < \bar{x}$）。

这种行为可能被认为在输入模式之间的暂时性竞争。

4. 历史注解。早在 1989 年 Sanger 的 GHA 发表之前，Karhunen 和 Oja（1982）发表了一篇会议论文，描述了一个叫做静态梯度算法（SGA），用于减少 PCA 的特征向量。后来有人证明 SGA 与 GHA 非常接近。
5. 小波。在本书的序言中，Mallat（1998）有以下的论断：

小波并非基于新的思想，而是基于在许多不同领域中不同形式的已有的概念。小波理论的形成和出现是多种学科努力的结果，其包括数学、物理、工程这三门被认为是独立发展相同思想的学科。对于信号处理，此关联创立了一系列的观点，其意义超出了新基或变换的构造。

用 $\psi(t)$ 表示一个零均值的函数，如下：

$$\int_{-\infty}^{\infty} \psi(t)\mathrm{d}t = 0$$

函数 $\psi(t)$ 表示一个带通滤波器的脉冲响应；这样一个函数可以称之为小波。此小波被规模参数 s 放大，且随时间参数 u 平移；我们可以写成：

$$\psi_{u,s}(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t-u}{s}\right)$$

给定一带傅里叶变换 $G(f)$ 的实值信号 $g(t)$，$g(t)$ 的连续小波变换由积分形式的内积所定义：

$$W_g(u,s) = \langle \psi_{u,s}(t), g(t) \rangle = \int_{-\infty}^{\infty} g(t)\psi_{u,s}(t)\mathrm{d}t$$

根据此公式，小波变换 $\psi_{u,s}(t)$ 与信号 $g(t)$ 相关。等价于

$$W_g(u,s) = \langle \Psi_{u,s}(f), G(f) \rangle = \int_{-\infty}^{\infty} G(f)\Psi_{u,s}^{*}(f)\mathrm{d}f$$

其中 $\psi_{u,s}(f)$ 是 $\psi_{u,s}(t)$ 的傅里叶变换，星号表示复共轭。因此，小波变换 $W_g(u,s)$ 依赖于信号 $g(t)$ 和其傅里叶变换 $G(f)$ 在时频域上的值。其中 $\psi_{u,s}(t)$ 的能量和它的傅里叶变换 $\psi_{u,s}(f)$ 是有关联的。

读者若想更深层次地了解小波变换，可以参考 Mallat（1998）和 Daubechies（1992，1993）的书。而 Meyer（1993）的简介性的书包括小波变换的历史发展过程。

6. 非线性 PCA 方法。

这些方法可以被归入四类网络：

（1）**Hebb 网络**，用非线性神经元代替基于 Hebb 规则的 PCA 算法的线性神经元得到（Karhunen and Joutsensalo，1995）。

（2）**复制器网络或自动编码器**，建立在多层感知器基础上，包括三层隐藏层（Kramer，1991）：

- 映射层；
- 瓶颈层；
- 逆映射层。

复制器网络在第 4 章讨论。

（3）**主曲线**，基于捕获数据结构的曲线或曲面的迭代估计（Hastie and Stuetzle，1989）。自组织映射可被看做发现主曲线离散逼近的计算过程；自组织映射在下一章讨论。

（4）**核 PCA**。源于 Schölkopf 等（1998），在本章的 8.8 节中讨论。

7. 在 Kim et al.（2005），图像去噪实验的结果，包括 KHA 算法，体现了如下几点：
   - 人脸（单块）图像的超限分辨和去噪；
   - 自然场景的多块超限分辨图像。
8. 中值滤波器是一个关于如下的绝对误差代价函数最小化贝叶斯风险的估计算子：

$$R(e(n)) = |e(n)|$$

其中 $e(n)$ 是误差信号，由滤波器的预期响应信号和实际响应之间的差别定义。结果表明此最小值就是后验概率密度函数的中值，此滤波器也因此得名。

9. 自适应 Wiener 滤波器。Wiener 滤波器在第 3 章讨论过。在自适应 Wiener 滤波器中，训练样本 $\{\mathbf{x}_i(n), \mathbf{d}_i(n)\}_{i=1}^{N}$ 被分为一系列连续的带类标数据块。并且滤波器参数在块乘块的基础上，使用规范等式（或离散形式的 Wiener-Hopf 等式）计算。实际上，在每一块内，数据被看成伪静态的。每一训练样本的统计变化显示了滤波器参数在每一块上发生变化。

10. Sobolev 空间由空间中所有具有 m 阶导数的函数组成。并且在此空间中，所有 m 阶导数都是绝对可积的（Vapnik，1998）。Bezov 空间包含第三个参数，并在 $m=1$ 和 $m=\infty$ 时，精简了光滑性条件。

习题

竞争和合作

8.1 在自组织系统中，包含竞争和合作。我们发现竞争先于合作。证明此论断的合理性。

主分量分析：约束优化方法

8.2 在 8.4 节中，我们使用扰动理论来推导 PCA。在此习题中，我们从一个约束最优化方法的角度，来解决同样的问题。

令 $\mathbf{x}$ 表示一个 m 维零均值的数据向量，$\mathbf{w}$ 表示同样 m 维的可调整的参数向量。令 σ^2 表示数据向量 $\mathbf{x}$ 在参数向量 $\mathbf{w}$ 上投影的方差。

(a) 证明在 $\|\mathbf{w}\|=1$ 的约束条件下，拉格朗日最大化方差 σ^2，由如下定义：

$$J(\mathbf{w}) = \mathbf{w}^T\mathbf{R}\mathbf{w} - \lambda(\mathbf{w}^T\mathbf{w} - 1)$$

其中 $\mathbf{R}$ 是数据向量 $\mathbf{x}$ 的协方差矩阵，λ 是拉格朗日乘子。

(b) 使用 8.2(a)的结果，证明关于 $\mathbf{w}$ 的拉格朗日函数 $J(\mathbf{w})$最大解由如下的特征方程所定义：

$$\mathbf{R}\mathbf{w} = \lambda\mathbf{w}$$

因此，说明 $\sigma^2=\mathbb{E}[(\mathbf{w}^T\mathbf{x})^2]=\lambda$。在特征分解中，$\mathbf{w}$ 是特征向量，λ 是相应的特征值。

(c) 令拉格朗日乘子 λ_i 表示对于第 i 个特征向量的规范化条件 $\|\mathbf{w}_i\|=1$。令拉格朗日乘子 λ_{ij} 表示正交化条件 $\mathbf{w}_i^T\mathbf{w}_j=0$。证明拉格朗日函数此时有如下的扩展形式：

$$J(\mathbf{w}_i) = \mathbf{w}_i^T\mathbf{R}\mathbf{w}_i - \lambda_{ii}(\mathbf{w}_i^T\mathbf{w} - 1) - \sum_{j=1}^{i-1}\lambda_{ij}\mathbf{w}_i^T\mathbf{w}_j, \qquad i = 1,2,\cdots,m$$

因此，证明最大化 $J(\mathbf{w}_i)$ 这 m 个等式的解正好为对应于特征值 λ_i 的特征向量 $\mathbf{w}_i$。

8.3 令 m 维零均值数据向量 $\mathbf{x}$ 的估计由如下的等式定义：

$$\hat{\mathbf{x}}_l = \sum_{i=1}^{l} a_i\mathbf{q}_i, \qquad l \leqslant m$$

其中 $\mathbf{q}_i$ 是如下协方差矩阵的第 i 个特征向量：

$$\mathbf{R} = \mathbb{E}[\mathbf{x}\mathbf{x}^T]$$

且 $a_1, a_2, \cdots, a_l$ 是系数，受制于如下条件：

$$\mathbf{q}_i^T\mathbf{q}_j = \begin{cases} 1 & \text{为 } j = i \\ 0 & \text{其他} \end{cases}$$

证明关于可变系数对如下均方误差的最小值：

$$J(\hat{\mathbf{x}}_i) = \mathbb{E}[\|\mathbf{x} - \hat{\mathbf{x}}_i\|^2]$$

就是所定义的第 i 个主分量

$$a_i = \mathbf{q}_i^T\mathbf{x}, \qquad i = 1,2,\cdots,l$$

即数据向量 $\mathbf{x}$ 在特征向量 $\mathbf{q}_i$ 上的投影。

8.4 根据问题 8.2 中讨论的约束最优化问题，考虑拉格朗日函数：

$$J(\mathbf{w}) = (\mathbf{w}^T\mathbf{x})^2 - \lambda(\mathbf{w}^T\mathbf{w} - 1)$$

其中$(\mathbf{w}^T\mathbf{x})^2$ 表示零均值数据向量 $\mathbf{x}$ 在权值向量 $\mathbf{w}$ 上的投影的方差的瞬时值。

(a) 估计拉格朗日函数 $J(\mathbf{w})$关于可变权值 $\mathbf{w}$ 的梯度，有

$$g(\mathbf{w}) = \frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = 2(\mathbf{w}^T\mathbf{x})\mathbf{x} - 2\lambda\mathbf{w}$$

(b) 对于在线学习的静态梯度下降算法，我们有如下权值更新公式：

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \frac{1}{2}\eta\mathbf{g}(\hat{\mathbf{w}}(n))$$

其中 η 是学习参数。因此，可以推出迭代公式：

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \eta[(\mathbf{x}(n)\mathbf{x}^T(n))\,\hat{\mathbf{w}}(n) - \hat{\mathbf{w}}^T(n)(\mathbf{x}(n)\mathbf{x}^T(n))\,\hat{\mathbf{w}}(n)\,\hat{\mathbf{w}}(n)]$$

这是式(8.47)的重写，定义了关于离散时间 n 的最大特征滤波，而$\hat{\mathbf{w}}(n)$代替了 $\mathbf{w}(n)$。

基于 Hebb 的最大特征滤波器

8.5 对于例 2 中考虑的匹配滤波器，特征值 λ_1 和对应的特征向量为 $\mathbf{q}_1$，定义为：

$$\lambda_1 = 1 + \sigma^2$$

且

$$\mathbf{q}_1 = s$$

证明这些参数满足基本的关系：

$$\mathbf{R}\mathbf{q}_1 = \lambda_1 \mathbf{q}_1$$

其中 $\mathbf{R}$ 为输入向量的相关矩阵。

8.6 考虑最大特征滤波器，其中权值向量 $\mathbf{w}(n)$按照式(8.46)演化。证明随着 n 趋向于无穷大，滤波器的输出方差趋向于 λ_{max}，其中 λ_{max}为输入向量相关矩阵的最大特征值。

8.7 次分量分析（minor components analysis，MCA）与主分量分析是相反的。在 MCA 中，我们寻找投影方差最小的方向。这样得到的方向对应于输入向量 $\mathbf{X}(n)$的相关矩阵 $\mathbf{R}$ 的最小特征值的特征向量。

在本题中，我们探讨怎样修改 8.4 节的单个神经元发现 $\mathbf{R}$ 的次分量。特别地，我们可以对式(8.40)的学习规则改变符号，得到（Xu 等，1992）：

$$w_i(n+1) = w_i(n) - \eta y(n)(x_i(n) - y(n)w_i(n))$$

证明如果相关矩阵 $\mathbf{R}$ 的最小特征值 λ_m 重数为 1，则：

$$\lim_{n\to\infty} \mathbf{w}(n) = \eta \mathbf{q}_m$$

其中 $\mathbf{w}(n)$是权值向量，第 i 个分量是 $\mathbf{w}_i(n)$，$\mathbf{q}_m$ 是与λ_m 对应的特征向量。

基于 Hebb 的主分量分析

8.8 构造一个信号流图表示向量值等式(8.87)和式(8.88)。

8.9 在 8.5 节描述的用于收敛性分析的常微分方程方法不能直接用于广义 Hebb 学习算法（GHA）。然而，通过将式(8.91)的突触权值矩阵 $\mathbf{W}(n)$用 $\mathbf{W}(n)$的列向量的组合来表示，则我们可以用通常的方式解释更新函数，然后继续应用渐进稳定性定理。因此，根据此处已有的说明，证明 GHA 算法的收敛性定理。

8.10 在这个习题中，我们可以探讨利用广义 Hebb 算法来研究随机输入向量产生的二维接收域（Sanger，1990）。随机输入包含独立于高斯噪声具有零均值和单位方差的二维域，它与高斯掩模（滤波器）作卷积，然后乘以一个高斯窗。高斯掩模有两个像素的标准偏差，高斯窗有 8 个像素的标准偏差。在位置(r,s)的结果随机输入 $x(r,s)$可以写成：

$$x(r,s) = m(r,s)[g(r,s) * w(r,s)]$$

其中 $w(r,s)$是独立和同分布的高斯噪声的域，$g(r,s)$是高斯掩模，$m(r,s)$是窗函数。$g(r,s)$和$w(r,s)$的循环卷积由下式定义：

$$g(r,s) * w(r,s) = \sum_{p=0}^{N-1}\sum_{q=0}^{N-1} g(p,q)w(r-p,s-q)$$

其中 $g(r,s)$和 $w(r,s)$均假设为周期的。

用随机输入 $x(r,s)$的 2 000 个样本训练基于 GHA 算法的单层前馈网络。网络有 4 096 个输入，排列成 64×64 像素网格，具有 16 个输出。训练网络的结果突触权值用 64×64 阵列的数表示。执行上述计算并显示突触权值作为二维掩模的 16 个阵列。评价你的结果。

8.11 在仅需要主子空间（即主特征向量张成的空间）的情况下，我们可以使用由此式定义的对称算法：

$$\hat{\mathbf{w}}_j(n+1) = \hat{\mathbf{w}}_j(n) + \eta y_j[\mathbf{x}(n) - \hat{\mathbf{x}}_j(n)]$$

$$\hat{\mathbf{x}}(n) = \sum_{j=1}^{l} \hat{\mathbf{w}}_j(n)\mathbf{y}_j(n)$$

(a) 讨论此对称算法和 GHA 之间的异同点。

(b) 主子空间可以看成式(8.46)定义的 Oja 规则的泛化。解释此泛化的合理性。

特征提取：习题 8.12 和习题 8.13 的导言

在表示一个由许多聚类组成的数据集时，我们可以说，为了使这些聚类可见，它们之间的分割应当大于每个聚类内的差异。如果以上条件成立，则数据集中只有少量数目的聚类，那么我们用 PCA 所求出的主分量来投影聚类的话就会得到好的分离效果。这样对于特征提取问题来说就会是有效的一组基底。

8.12 在 4.19 节中，我们描述了结构风险最小化，此方法通过为机器学习匹配合适大小的训练样本集，来系统地获得最佳的泛化性能。

把目标作为约简输入数据空间维数的主分量分析看成机器学习的预处理过程，讨论这个预处理过程如何能够通过对一组模式分类器排序，而把结构信息嵌入学习过程。

8.13　作为预处理过程的主分量分析的另一个应用是使用反向传播算法监督式地训练一多层感知器。此应用的目的是通过对输入数据相关联而加速学习过程的收敛。试讨论此目的如何实现。

自适应主分量提取

8.14　广义 Hebb 学习算法（GHA）依赖于对主分量分析使用反馈连接。在此问题中，我们使用一种叫做*自适应主分量提取算法*（Kung and Diamantaras，1990；Diamantaras and Kung，1996）。

APEX 算法使用前馈和反馈连接，如图 P8.14 所示。输入向量 $\mathbf{x}$ 是 m 维，网络中的每一个神经元都是线性的。

在此网络中有两种突触连接：

(1) 从输入结点到 $1,2,\cdots,j$ 每个神经元的前馈连接，其中 $j<m$。这些连接由前馈权值向量表示：

$$\mathbf{w}_j(n)=[w_{j1}(n),w_{j2}(n),\cdots,w_{jm}(n)]^T$$

其中 n 代表离散的时间。

(2) 由从 $1,2,\cdots,j-1$ 单个神经输出到神经元 j 的侧向连接；这些连接由反馈神经权值表示：

$$\mathbf{a}_j(n)=[a_{j1}(n),a_{j2}(n),\cdots,a_{j,j-1}(n)]^T$$

这些前馈突触连接是 Hebb 的，但是反馈突触连接是反 Hebb 的，因此是抑制的。神经元 j 的输出由以下给出：

$$y_j(n)=\mathbf{w}_j^T(n)\mathbf{x}(n)+\mathbf{a}_j^T(n)\mathbf{y}_{j-1}(n)$$

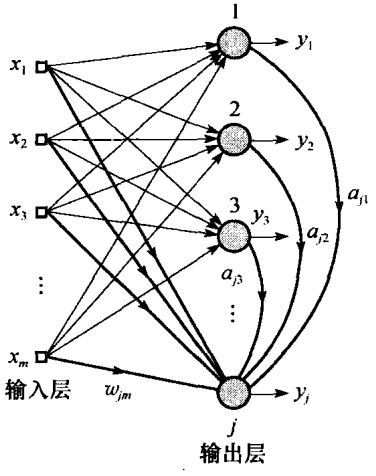


图 P8.14　APEX 算法中的前馈和后向连接网络

由以上分析，我们假定网络中所有神经元已收敛到它们各自的稳定状态，即

$$\mathbf{w}_k(0)=\mathbf{q}_k,\quad k=1,2,\cdots,j-1$$

$$\mathbf{a}_k(0)=\mathbf{0},\quad k=1,2,\cdots,j-1$$

其中 $\mathbf{q}_k$ 是相应于协方差矩阵的第 k 个特征值的特征向量。

$$\mathbf{R}=\mathbb{E}[\mathbf{x}(n)\mathbf{x}^T(n)]\text{ 在时间阶段 }n=0$$

(a) 基于式(8.40)，写出对于神经元 j 的关于向量 $\mathbf{w}_j(n)$ 和 $\mathbf{a}_j(n)$ 的更新公式。

(b) 假定协方差矩阵 $\mathbf{R}$ 的特征值按降序排列，其中 λ_1 是最大的。记为关于特征值 λ_k 的特征向量 $\mathbf{q}_k$。要表达前馈权值向量 $\mathbf{w}_j(n)$ 的时变特性，可以使用如下式表示：

$$\mathbf{w}_j(n)=\sum_{k=1}^{m}\theta_{jk}(n)\mathbf{q}_k$$

其中 $\theta_{jk}(n)$ 是时变系数。因此，试证：

(i) $$\sum_{k=1}^{m}\theta_{jk}(n+1)\mathbf{q}_k=\sum_{k=1}^{m}\{1+\eta[\lambda_k-\sigma_j^2(n)]\}\theta_{jk}(n)\mathbf{q}_k+\eta\sum_{k=1}^{j-1}\lambda_k a_{jk}(n)\mathbf{q}_k$$

其中 η 是学习参数，$a_{jk}(n)$ 是反馈权值向量 $\mathbf{a}_j$ 的第 k 个分量，且 $\sigma_j^2(n)=\mathbb{E}[y_j^2(n)]$ 是神经元 j 的平均输出。

(ii) $$\mathbf{a}_j(n+1)=-\eta\lambda_k\theta_{jk}(n)\mathbf{1}_k+\{1-\eta[\lambda_k+\sigma_j^2(n)]\}\mathbf{a}_j(n)$$

其中 $\mathbf{1}_k$ 是其所有分量都为 0，仅第 k 个分量为 1 的向量。

(c) 为了进一步讨论，需要考虑两种情况：

情况 I：$1\leqslant k\leqslant j-1$

在此情况下，有：

$$\begin{bmatrix}\theta_{jk}(n+1)\\ a_{jk}(n+1)\end{bmatrix}=\begin{bmatrix}1+\eta(\lambda_k-\sigma_j^2(n)) & \eta\lambda_k\\ -\eta\lambda_k & 1-\eta(\lambda_k+\sigma_j^2(n))\end{bmatrix}\begin{bmatrix}\theta_{jk}(n)\\ a_{jk}(n)\end{bmatrix}$$

此 2×2 的矩阵具有二重特征值：

$$\rho_{jk}=[1-\eta\sigma_j^2(n)]^2$$

考虑到 $\rho_{jk}<1$，证明 $\theta_{jk}(n)$ 和 $a_{jk}(n)$ 在 n 不断增大时渐近地趋于 0。

情况 II：$j\leqslant k\leqslant m$

对于此种情况，反馈权值 $a_{jk}(n)$ 对于网络的模型没有影响；因此，

$$a_{jk}(n)=0\qquad\text{对于 }j\leqslant k\leqslant m$$

故而，对于每个 $k\geqslant j$ 的主要模型，有

$$\theta_{jk}(n+1) = \{1 + \eta[\lambda_k - \sigma_j^2(n)]\}\theta_{jk}(n)$$

因此 n 不断增大时 $\theta_{jk}(n)$渐进地收敛于 0。

神经元 j 的平均输出表达如下：

$$\sigma_j^2(n) = \sum_{k=j}^{m} \lambda_k \theta_{jk}^2(n)$$

最终有：

$$\lim_{n\to\infty} \sigma_j^2(n) = \lambda_j$$

和

$$\lim_{n\to\infty} \mathbf{w}_j(n) = \mathbf{q}_j$$

核 PCA

8.15 令 $\bar{k}_{ij}$ 表示核矩阵 $\mathbf{K}$ 的第 ij 个元素 k_{ij} 中心化后所对应的部分。证明以下等式（Schölkopf，1997）：

$$\bar{k}_{ij} = k_{ij} - \frac{1}{N}\sum_{m=1}^{N} \boldsymbol{\phi}^T(\mathbf{x}_m)\boldsymbol{\phi}(\mathbf{x}_j) - \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{\phi}^T(\mathbf{x}_i)\boldsymbol{\phi}(\mathbf{x}_n) + \frac{1}{N^2}\sum_{m=1}^{N}\sum_{n=1}^{N} \boldsymbol{\phi}^T(\mathbf{x}_m)\boldsymbol{\phi}(\mathbf{x}_n)$$

建议用紧凑的矩阵形式表示这个关系。

8.16 证明核矩阵 $\mathbf{K}$ 的特征向量 $\boldsymbol{\alpha}$ 的归一化与式(8.109)的条件等价。

计算机实验

8.17 我们继续 8.7 节中图像编码的实验，有两个特别有趣的问题：

(a) 描绘 GHA 的学习曲线，其中算法是训练 Lena 图像（即描绘均方误差随着训练轮数的变化的变化量）。

(b) 同样，描绘在椒盐图像中算法的学习曲线。

8.18 在此实验中，我们重新提到核 PCA 中的例 3。我们对于二维数据用以下公式计算核 PCA 分量：

$$x_2 = x_1^2 + v$$

其中 v 是均值为 0 方差 0.04 的附加高斯噪声。然而，此处我们要求用核 Hebb 算法计算。比较此处和例 3 中的实验结果。

第9章
Neural Networks and Learning Machines, 3E

自组织映射

本章组织

这一章学习运用自组织原则来产生“拓扑映射”。关于这一主题的处理方案组织如下：

9.1 节是引言，用于激发运用自组织映射网的兴趣。

9.2 节描述两个基本特征模型，它们两个都用自己的方式受到神经生物学考虑的激发。

9.3 节和 9.4 节处理高度流行和广泛使用的自组织（特征）映射（SOM）及其特性。9.5 节介绍计算机实验，突出了 SOM 的独有的特征。9.6 节介绍 SOM 用于构造上下文映射的应用。

9.7 节讨论分层矢量量化，利用自组织映射将简化其执行。

9.8 节讲述基于核的自组织映射（kernel SOM），紧接着通过 9.9 节的计算机实验来例示这一新算法的改进的拓扑映射能力。9.10 节讨论核 SOM 和 Kullback-Leibler 发散之间的关系。

9.11 节通过小结和讨论本章的主要内容而结束本章。

9.1 引言

在这一章我们通过考虑一种称为自组织映射的特殊人工神经网络继续学习自组织系统。这类网络基于竞争学习（competitive learning）；网络的输出神经元之间互相竞争以求被激活或点火，结果在每一时刻只有一个输出神经元，或者每组只有一个输出神经元被激活。赢得竞争的一个输出神经元被称作胜者全得（winner-takes-all）神经元或简称获胜神经元（winning neuron）[1]。在输出神经元中导出胜者全得的竞争方法是在它们之间使用侧抑制连接（即负反馈路径）；这个思想是由 Rosenblatt（1958）最先提出的。

在自组织映射里，神经元被放置在网格节点上，这个网格通常是一维或是二维的。更高维映射也可以，但是不常见。在竞争学习过程中，神经元变化依不同输入模式（刺激）或者输入模式的类别而选择性地调整。这样调整后神经元（即获胜神经元）的位置彼此之间成为有序的，使得对于不同的输入特征，在网格上建立起有意义的坐标系。因此自组织映射由输入模式的拓扑映射（topographic map）结构所表征，其中网格神经元的空间位置（即坐标）表示输入模式包含的内在统计特征，“自组织映射”因此得名。

作为一个神经模型，自组织映射在两个自适应层次之间提供桥梁：

- 在单个神经元的微观层次形成自适应规则。
- 在神经元层次的微观层上形成特征选择的在实验上更好和具体可实现的模式。

自组织映射本质上是非线性的。

发展自组织映射作为神经模型是由人脑的一个突出特征所激发：

人脑在许多地方以这样一种方式组织起来，使得不同的感觉输入由拓扑有序的计算映射（topologically ordered computational map）来表示。

特别地，感觉输入如触觉（Kaas 等，1983）、视觉（Hubel and Wiesel，1962，1977）和听觉（Suga，1985）用拓扑有序的方式映射到人脑皮层的不同区域。这样在神经系统的信息处理基本结构中，计算映射组成一个基本构件。一个计算映射由神经元阵列定义，这些神经元表示略微不同调制的处理器和滤波器，它们并行处理携带信息的传感信号。所以，神经元将输入信号转变为空间位置编码的概率分布，分布通过映射中最大相关激活的位置表示参数的计算值（Knudsen 等，1987）。用这种方式导出的信息属于这样一种形式，它可以用于使用相对简单的

连接模式的高阶处理器。

9.2　两个基本的特征映射模型

任何人只要检查人脑就会禁不住对人脑被大脑皮质所占据的范围留下深刻印象。人脑几乎完全被大脑皮质所包围，它遮蔽了其他部分。由于惊人的复杂性，大脑皮质也许超过了宇宙中任何已知的结构（Hubel and Wiesel，1977）。同样给我们深刻印象的是将不同的感觉输入（运动、身体的体觉、视觉、听觉等）以一种有序的方式映射到相应的大脑皮质区域的方法；为了说明这一点，参看导言中图 4 的大脑皮质的细胞结构图。计算映射的使用提供下面的特性（Knudsen 等，1987；Durbin and Michison，1990）：

1. *在每次映射中，神经元并行地处理自然相似的信息片断，但这些信息片断来自于感知输入空间的不同区域。*

2. *在表示的每一阶段，每一个新来的信息片段保持在它合适的位置中。*

3. *处理高度相关的信息片段的神经元被紧密地联系到一起，通过短的突触连接使得它们能够交互。*

4. *上下文映射能通过从高维参数空间到皮质表面的决策-衰减映射*（decision-reducing mapping）*来理解。*

我们的兴趣在于建立人工拓扑映射，它以神经生物学激励的方式通过自组织来学习。在这段文字中，从人脑计算映射的非常简短的讨论所体现的重要一点是拓扑映射构成原则，它可以陈述如下（Kohonen，1990）：

在拓扑映射中输出神经元的空间位置对应于特殊的定义域或从输入空间抽取数据的特征。

这个原则提供了这里描述的两个基本不同的特征映射模型[2] 的神经学生物基础。

图 9.1 展现两个模型的布局。在这两种情况下输出神经元被安排在二维的网格中。这种拓扑确保每个神经元都有一组邻域。模型间的区别在于输入模式的指定方式。

图 9.1a 的模型由 Willshaw and von der Malsburg（1976）在生物学基础上首先提出，用以解释（在高级脊椎动物中）从视网膜到视觉皮质的视觉映射的问题。具体地，有两个不同的二维网格神经元连接在一起，一个投射到另一个。一个网格代表前突触（输入）神经元，另一个网格代表后突触（输出）神经元。后突触网格使用*短程兴奋机制*（short-range excitatory mechanism）和*长程抑制机制*（long-range inhibitory mechanism）。这两种机制本质上都是局部的且对自组织特别重要。这两个网格由 Hebb 型的可调突触相互连接。因此严格地说，后突触神经元并不是胜者全得；相反使用阈值确保在任一时刻仅有一些后突触神经元点火。更进一步，为了防止可能导致网络不稳定性的突触权值的稳定建立，每个后突触神经元

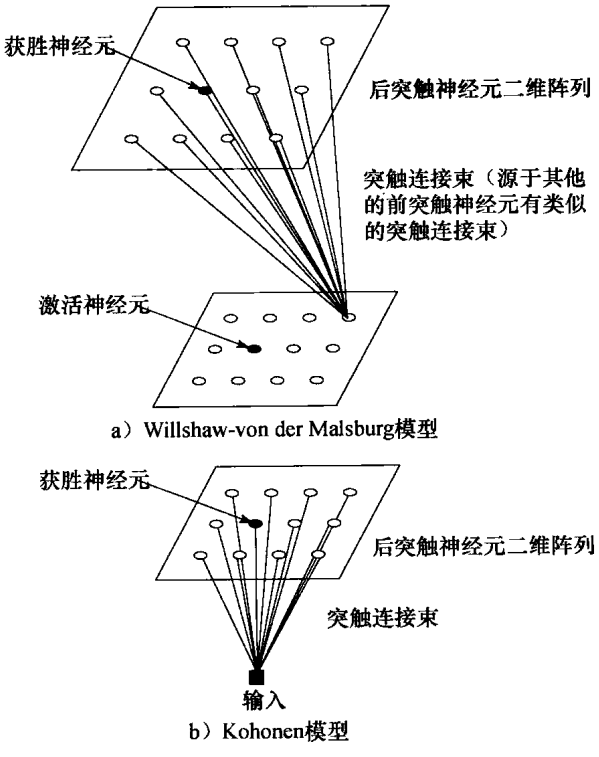


图 9.1　两个自组织特征映射

的总权值有一个上界[3]。因此对每个神经元一些突触权值上升伴随着另外的神经元下降。Willshaw-von der Malsburg 模型的基本思想是对前突触神经元的几何邻近编码为它们电位活动的相关形式，并且在后突触网格中利用这些相关使得相邻的前突触神经元连接到相邻的后突触神经元。从而由自组织产生拓扑有序的映射。但需注意 Willshaw-von der Malsburg 模型限制为输入和输出维数相同的映射。

图 9.1b 的第二个模型，由 Kohonen（1982）引入，并不在说明神经生物学的细节。模型抓住人脑中计算映射的本质特征而且保留计算的易行性。Kohonen 模型看起来比 Willshaw-von der Malsburg 模型更为一般，前者能进行数据压缩（即输入维数的缩减）。

现实中，Kohonen 模型属于*向量—编码*（vector-coding）算法的类型。模型提供一个拓扑映射，它最优地设置固定数目的向量（即编码字）到高维输入空间，因此有利于数据压缩。Kohonen 模型因此可由两种方式导出。首先，我们可以用由神经生物学考虑所激发的自组织的基本思想导出模型，这是传统的方法（Kohonen，1982，1990，1997）。另外，可以用向量量化的方法，使用包含编码器和解码器的模型，这由通信理论的考虑所激发（Luttrell，1989b，1991a）。在本章我们考虑这两种方法。

在文献中 Kohonen 模型比 Willshaw-von der Malsburg 模型受到更多的关注。它拥有在本章后面讨论的一些性质，这使得 Kohonon 模型可能用于捕捉皮质映射的本质特征。

9.3　自组织映射

自组织映射（self-organizing map，SOM）的主要目的是将任意维数的输入信号模式转变为一维或二维的离散映射，并且以拓扑有序的方式自适应实现这个变换。图 9.2 给出常用作离散映射的二维神经元网格的简要图表。网格中每个神经元和输入层的源节点全连接。这个网络表示具有神经元按行和列排列的单一计算层的前馈结构。一维网格是图 9.2 描绘的构形的一个特例：在这种特殊情形中，计算层仅由单一的列或行神经元构成。

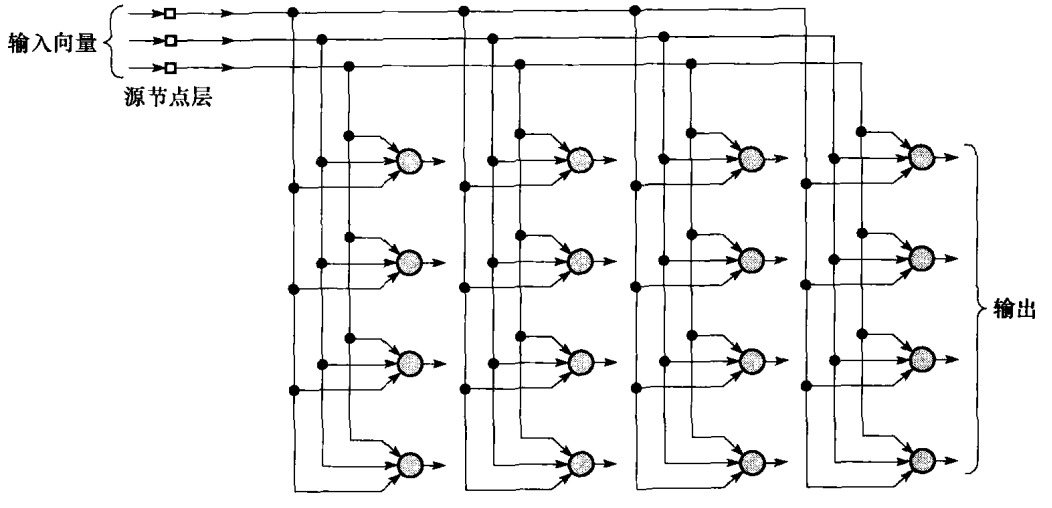


图 9.2　神经元的二维网格，以三维输入和 4×4 维输出为例说明

呈现给网络的每个输入模式，通常包含面对平静背景的一个局部化活动区域或“点”。这个点的位置和性质通常随输入模式的实现不同而不同。因此网络中所有神经元应经历输入模式的足够次数的不同实现，确保有机会完成恰当的自组织过程。

负责形成自组织映射的算法，第一步进行网络突触权值的*初始化*。这个工作可以从*随机数产生器*中挑选较小的值赋予它们；这样做，在特征映射上没有加载任何先验的序。一旦网络被

恰当初始化，在自组织映射的形成中有三个主要过程，小结如下：

1. 竞争。对每个输入模式，网络中的神经元计算它们各自的判别函数的值。这个判别函数为神经元之间的竞争提供基础。具有判别函数最大值的特定神经元成为竞争的胜利者。

2. 合作。获胜神经元决定兴奋神经元的拓扑邻域的空间位置，从而提供这样的相邻神经元合作的基础。

3. 突触调节。最后的这一机制使兴奋神经元通过对它们突触权值的适当调节以增加它们关于该输入模式的判别函数值。所做的调节使获胜神经元对以后相似输入模式的响应增强了。

竞争和合作的过程符合第8章描述的四个自组织原则中的两个。对于自增强原则，它来源于自适应过程的Hebb学习的修正形式。如第8章的解释，输入数据中的冗余（虽然在描述SOM算法时没有明显提及）对学习是必要的，因为它提供了输入激活模式中所隐含的结构知识。下面给出竞争、合作和突触调节过程的详细描述。

竞争过程

令 m 表示输入（数据）空间的维数。从输入空间中随机选择输入模式（向量）记为

$$\mathbf{x} = [x_1, x_2, \cdots, x_m]^T \tag{9.1}$$

网络中每个神经元的突触权值向量和输入空间的维数相同。神经元 j 的突触权值向量记为

$$\mathbf{w}_j = [w_{j1}, w_{j2}, \cdots, w_{jm}]^T, \qquad j = 1, 2, \cdots, l \tag{9.2}$$

其中 l 是网络中神经元的总数。为了找到输入向量 $\mathbf{x}$ 与突触权值向量 $\mathbf{w}_j$ 的最佳匹配，对 $j=1,2,\cdots,l$ 比较内积 $\mathbf{w}_j^T\mathbf{x}$ 并选择最大者。这里假定所有的神经元有相同的阈值；阈值偏置取负。这样，通过选择具有最大内积 $\mathbf{w}_j^T\mathbf{x}$ 的神经元，我们实际上决定了兴奋神经元的拓扑邻域中心的位置。

从导言中我们回想基于内积 $\mathbf{w}_j^T\mathbf{x}$ 最大化的最优匹配准则，在数学上等价于向量 $\mathbf{x}$ 和 $\mathbf{w}_j$ 的欧几里得距离的最小化。如果用索引 $i(\mathbf{x})$ 标识最优匹配输入向量 $\mathbf{x}$ 的神经元，我们可以通过下列条件决定 $i(\mathbf{x})$[4]：

$$i(\mathbf{x}) = \arg\min_j \| \mathbf{x} - \mathbf{w}_j \|, \qquad j \in \mathcal{A} \tag{9.3}$$

这概括了神经元中竞争过程的本质。这里 $\mathcal{A}$ 定义了神经元网格。根据式(9.3)，$i(\mathbf{x})$ 是注意的目标，因为我们要识别神经元 i。满足这个条件的特定神经元 i 被称为输入向量 $\mathbf{x}$ 的*最佳匹配*或*获胜神经元*。式(9.3)导出这样的观察：

激活模式的连续输入空间通过网络中神经元之间的竞争过程映射到神经元的离散输出空间。

根据应用的不同，网络的响应可能是获胜神经元的标号（即它在网格中的位置）或者是在欧几里得距离意义下距输入向量最近的突触权值向量。

合作过程

获胜神经元位于合作神经元的拓扑邻域的中心。关键问题是：我们怎样定义一个在神经生物学上正确的拓扑邻域？

要回答这个问题，记住对于人类大脑中一组兴奋神经元的侧向相互作用有神经生物学的证据。特别地，一个点火的神经元倾向于激活它紧接的邻域内的神经元而不是和它隔得远的神经元，这在直观上是满足的。这个观察引导我们对获胜神经元的拓扑邻域按侧向距离光滑地缩减(Lo等，1991，1993；Ritter等，1992)[5]。具体地，设 $h_{j,i}$ 表示以获胜神经元 i 为中心的拓扑邻域且包含一组兴奋（合作）神经元，其中一个神经元记为 j。设 $d_{i,j}$ 表示在获胜神经元 i 和兴奋神经元 j 的侧向距离。然后我们可以假定拓扑邻域 $h_{j,i}$ 是侧向距离 $d_{j,i}$ 的单峰函数使得它满足两个不同的要求：

1. 拓扑邻域 $h_{j,i}$ 关于 $d_{i,j}=0$ 定义的最大点是对称的；换句话说，在距离 $d_{j,i}$ 为零的获胜神经元 i 处达到最大值。

2. 拓扑邻域 $h_{j,i}$ 的幅度值随侧向距离 $d_{i,j}$ 的增加而单调递减，当 $d_{j,i}\to\infty$ 时趋于零；对收敛来说这是一个必要条件。

满足这些要求的一个 $h_{j,i}$ 的好的选择为高斯函数[6]：

$$h_{j,i(\mathbf{x})}=\exp\left(-\frac{d_{j,i}^2}{2\sigma^2}\right),\qquad j\in\mathcal{A} \tag{9.4}$$

它是平移不变的（即不依赖于获胜神经元 i 的位置）。参数 σ 是拓扑邻域的“有效宽度”，如图 9.3 所示；它度量靠近获胜神经元的兴奋神经元在学习过程中参与的程度。就量化来说，式(9.4)所示的高斯拓扑邻域比矩形形式的拓扑邻域在生物上更合适。它的使用使 SOM 算法的收敛速度比矩形拓扑邻域更快（Lo 等，1991，1993；Erwin 等，1992a）。

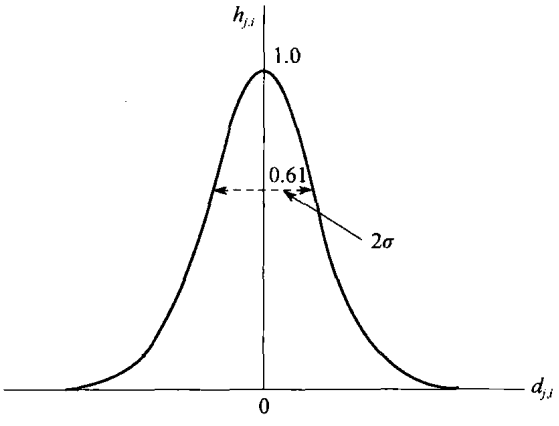


图 9.3　高斯邻域函数

对于邻域函数神经元之间的合作，必然要求拓扑邻域函数 $h_{j,i}$ 依赖获胜神经元 i 和兴奋神经元 j 在输出空间的侧向距离 $d_{j,i}$ 而不是依赖于原始输入空间的某种距离度量。这正是在式(9.4)中我们所表达的意义。就一维网格来说，$d_{j,i}$ 是整数且等于 $|j-i|$。另一方面，在二维网格的情况下它定义为：

$$d_{j,i}^2=\|\mathbf{r}_j-\mathbf{r}_i\|^2 \tag{9.5}$$

其中离散向量 $\mathbf{r}_j$ 定义兴奋神经元 j 的位置，而 $\mathbf{r}_i$ 定义获胜神经元 i 的离散位置，两者都是在离散输出空间中度量的。

SOM 算法的另一个独有特征是拓扑邻域的大小随时间收缩。这要求通过使拓扑邻域函数 $h_{j,i}$ 的宽度 σ 随时间而下降来满足。对于 σ 依赖于离散时间 n 的流行选择是由

$$\sigma(n)=\sigma_0\exp\left(-\frac{n}{\tau_1}\right)\qquad n=0,1,2,\cdots, \tag{9.6}$$

描述的指数衰减，其中 σ_0 是 SOM 算法中 σ 的初值，τ_1 是由设计者选择的时间常数（Ritter 等，1992；Obermayer 等，1991）。因此，拓扑邻域假定具有时变形式，表示如下：

$$h_{j,i(\mathbf{x})}(n)=\exp\left(-\frac{d_{j,i}^2}{2\sigma^2(n)}\right),\qquad n=0,1,2,\cdots, \tag{9.7}$$

其中 $\sigma(n)$ 由式(9.6)定义。于是随着 n（即迭代次数）的增加，宽度 $\sigma(n)$ 以指数下降，拓扑邻域以相应的方式缩减。然而，需要重点指出的是，邻域函数对于获胜神经元 i 最终仍然具有单位值，因为对于神经元 j 的距离 $d_{j,i}$ 是在网格空间中计算并和获胜神经元 i 相比较的。

存在着另一种关于邻域函数 $h_{j,i(\mathbf{x})}(n)$ 在获胜神经元 $i(\mathbf{x})$ 周围随时间 n 变动的有用观点。宽度 $h_{j,i(\mathbf{x})}(n)$ 的目标是使网格中大量兴奋神经元的权值更新方向相关。随着 $h_{j,i(\mathbf{x})}(n)$ 宽度减少，更新方向相关的神经元数量也在减少。当自组织映射的训练在计算机图形屏幕显示时，这个现象尤其明显。以相关形式在获胜神经元周围移动大量自由度是相当耗费计算机资源的，就像标准 SOM 算法一样。相反，使用重正规化（renormalized）SOM 的训练形式会更好，根据这一情况，我们选用更小数量的正规化自由度。通过使用恒定宽度的邻域函数 $h_{j,i(\mathbf{x})}(n)$，但逐渐增加领域函数中神经元的数量，这个操作很容易以离散形式完成。新的神经元被插到已有的神经

元之间，而 SOM 算法的平滑性保证新的神经元以很好的方式参与突触自适应（Luttrell，1989a）。重正规化 SOM 算法的概述在习题 9.15 给出。

自适应过程

现在我们来讨论特征映射自组织形成过程的最后一个过程，即突触自适应过程。为了使网络成为自组织的，要求神经元 j 的突触权值向量 $\mathbf{w}_j$ 随输入向量 $\mathbf{x}$ 改变。问题是如何进行改变。在 Hebb 学习假设中，突触权值随着前突触和后突触的激活同时发生而增加。此方法非常适合联想学习（例如，主分量分析）。然而对于这里考虑的无监督学习，Hebb 假设的基本形式是不能令人满意的，原因如下：连接的改变仅发生在一个方向上，这样最终使所有的突触权值都趋于饱和。为了克服这个问题，我们通过包括一个遗忘项 $g(y_j)\mathbf{w}_j$ 来改变 Hebb 假定，其中 $\mathbf{w}_j$ 是神经元 j 的突触权值向量，$g(y_j)$是响应 y_j 的正的标量函数。对 $g(y_j)$的唯一强制要求是它的 Taylor 级数展开的常数项为零，这样我们可写成：

$$g(y_j) = 0 \qquad \text{当 } y_j = 0 \tag{9.8}$$

这个要求的意义很快就会变得明显。给定这样一个函数，我们可以把网格中神经元 j 的权值向量改变表示成

$$\Delta\mathbf{w}_j = \eta y_j \mathbf{x} - g(y_j)\mathbf{w}_j \tag{9.9}$$

其中 η 是算法的学习率参数。式(9.9)右端第一项是 Hebb 项，第二项是遗忘项。为了满足式(9.8)，对 $g(y_j)$选择线性函数如下：

$$g(y_j) = \eta y_j \tag{9.10}$$

对于获胜神经元 $i(\mathbf{x})$，我们可以进一步简化式(9.9)，设：

$$y_j = h_{j,i(\mathbf{x})} \tag{9.11}$$

用式(9.10)和式(9.11)代入式(9.9)得到：

$$\Delta\mathbf{w}_j = \eta h_{j,i(\mathbf{x})}(\mathbf{x} - \mathbf{w}_j), \qquad \begin{cases} i: \text{获胜神经元} \\ j: \text{兴奋（激活）神经元} \end{cases} \tag{9.12}$$

最后使用离散时间形式，假定在时间 n 神经元 j 的权值向量为 $\mathbf{w}_j(n)$，更新权值向量 $\mathbf{w}_j(n+1)$ 在时间 $n+1$ 被定义为：

$$\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \eta(n)h_{j,i(\mathbf{x})}(n)(\mathbf{x}(n) - \mathbf{w}_j(n)) \tag{9.13}$$

它被应用到网格中获胜神经元 i 的拓扑邻域中的所有神经元（Kohonen，1982；Ritter 等，1992；Kohonen，1997a）。式(9.13)具有将获胜神经元 i 的突触权值向量 $\mathbf{w}_i$ 向输入向量 $\mathbf{x}$ 移动的作用。随着训练数据的重复出现，由于邻域更新使得突触权值向量趋于服从输入向量的分布。因此算法导致在输入空间中特征映射的拓扑排序，这意味着网格中相邻神经元会有相似的突触权值向量。关于这一点在 9.4 节中，我们将进一步详述。

式(9.13)为计算特征映射突触权值所期望的公式。除了这个公式之外，我们还需要用于选择邻域函数 $h_{j,i(\mathbf{x})}(n)$的启发式规则式(9.7)。

学习率参数 $\eta(n)$应如式(9.13)所示的时变形式，这也是它用于随机逼近的要求。特别地，它应从初始值 η_0 开始，然后随时间 n 增加而逐渐下降。这个要求可以通过下面的启发式而满足：

$$\eta(n) = \eta_0 \exp\left(-\frac{n}{\tau_2}\right), \qquad n = 0,1,2,\cdots, \tag{9.14}$$

其中，τ_2 是 SOM 算法的另一个时间常数。即使在式(9.6)和式(9.14)中描述的邻域函数宽度和学习率参数分别以指数衰减的公式可能不是最优的，但它们对于以自组织方式构成特征映射是足够的。

自适应过程的两个阶段：排序和收敛

假定算法的参数是正确选择的，从完全无序的初始状态开始，SOM 算法令人惊奇地逐步导致一个从输入空间抽取的激活模式有组织地表示。我们可以把根据式(9.13)计算的网络权值的自适应分解为两个阶段：排序或自组织阶段及其后的收敛阶段。自适应过程的这两个阶段描述如下（Kohonen，1982，1997a)：

1. *自组织或排序阶段*。在自适应过程的第一阶段形成权值向量的拓扑排序。这个排序阶段可能需要 SOM 算法的 1000 次迭代，也许会更多。要仔细考虑学习率参数和邻域函数的选择。

- 学习率参数 $\eta(n)$初始值应接近 0.1；然后逐渐减少，但应保持在 0.01 以上（即它将不被允许为 0)。这些要求的值可以在公式(9.14)中选择

$$\eta_0 = 0.1$$
$$\tau_2 = 1000$$

而得到满足。

- 邻域函数 $h_{j,i}(n)$的初始化应包括以获胜神经元 i 为中心的几乎所有神经元，然后随时间慢慢收缩。

具体来说，排序阶段可能需要 1000 次迭代或更多，允许 $h_{j,i}(n)$减少到仅有围绕获胜神经元的少量邻居神经元的小的值或者减少到获胜神经元自身。假定对离散映射使用神经元二维网格，则我们可以设定邻域函数的初始值 σ_0 等于网格的半径。相应地我们设定式(9.6)的时间常数：

$$\tau_1 = \frac{1000}{\log\sigma_0}$$

2. *收敛阶段*。自适应过程的第二阶段需要微调特征映射从而提供输入空间的准确统计量。而且，达到收敛所需要的迭代次数强烈依赖于输入空间的维数。作为一般性规则，组成收敛阶段的迭代次数至少是网络中神经元数目的 500 倍。这样收敛阶段可能进行几千次以至上万次的迭代。学习率参数的选择和邻域函数可以如下实现。

- 对于好的统计精度，在收敛阶段学习参数 $\eta(n)$应该保持在较小的值上，为 0.01 数量级。无论如何，不允许它下降到零；否则，网络会陷入到亚稳定状态。*亚稳定状态*（metastable state）属于有拓扑缺陷的特征映射结构。式(9.14)的指数衰减保证不可能进入亚稳定状态。
- 邻域函数 $h_{j,i(x)}$应该仅包括获胜神经元的最近邻域，最终减到一个或零个邻域神经元。

作为另一个评论：在讨论排序和收敛问题时，我们强调了完成这一过程需要的迭代次数。然而，在一些软件包中，回合（而不是迭代）被用于描述这两个问题。

SOM 算法小结

Kohonen 的 SOM 算法的本质是用一个简单的几何计算代替类 Hebb 规则的复杂性质和侧向相互作用。算法的主要构成/参数有：

- 根据一定概率分布产生激活模式的连续输入空间。
- 以神经元的网格形式表示的网络拓扑，它定义一个离散输出空间。
- 在获胜神经元 $i(x)$周围定义随时间变化的邻域函数 $h_{j,i(x)}(n)$。
- 学习率参数 $\eta(n)$的初始值是 η_0，然后随着时间 n 递减，但永不为零。

对于邻域函数和学习率参数，在排序阶段（即开始的大约 1000 次迭代）我们分别使用式(9.7)和式(9.14)。为了好的统计精度，在收敛阶段 $\eta(n)$的相当长的时间内应该保持一个较小值（0.01 或更小)，一般为几千次迭代。对于邻域函数，在收敛阶段之初，它应仅包含获胜

神经元的最近的邻域，并且最终缩减到一个或零个邻域神经元。

在初始化后算法的应用中涉及三个基本步骤：取样、相似性匹配和更新。重复这三个步骤直到完成特征映射的形成。算法总结如下：

1. *初始化*。对初始权值向量 $\mathbf{w}_j(0)$ 选择随机值。这里唯一的限制是 $j=1,2,\cdots,l$，$\mathbf{w}_j(0)$ 互不相同，其中 l 是网格中神经元的数目。可能希望保持较小的权值。

另一种算法初始化方法是从输入向量 $\{\mathbf{x}_i\}_{i=1}^{N}$ 的可用集里随机选择权值向量 $\{\mathbf{w}_j(0)\}_{j=1}^{l}$。这一不同选择的优势在于初始映射将在最终映射的范围内。

2. *取样*。以一定概率从输入空间取样本 $\mathbf{x}$；向量 $\mathbf{x}$ 表示应用于网格的激活模式。向量 $\mathbf{x}$ 的维数等于 m。

3. *相似性匹配*。在时间步 n 使用最小距离准则寻找最匹配（获胜）的神经元 $i(\mathbf{x})$：

$$i(\mathbf{x}) = \arg\min_j \| \mathbf{x}(n) - \mathbf{w}_j \|, \qquad j = 1,2,\cdots,l$$

4. *更新*。通过用更新公式调整所有神经元的权值向量：

$$\mathbf{w}_j(n+1) = \mathbf{w}_j(n) + \eta(n)h_{j,i(\mathbf{x})}(n)(\mathbf{x}(n) - \mathbf{w}_j(n))$$

其中 $\eta(n)$ 是学习率参数，$h_{j,i(\mathbf{x})}(n)$ 是获胜神经元 $i(\mathbf{x})$ 周围的邻域函数；为了获得最好的结果，$\eta(n)$ 和 $h_{j,i(\mathbf{x})}(n)$ 在学习过程中是动态变化的。

5. *继续*。继续步骤 2 直到在特征映射里观察不到明显的变化为止。

9.4　特征映射的性质

一旦 SOM 算法收敛，由算法计算的特征映射显示输入空间的重要统计特性。

开始令 $\mathcal{X}$ 表示空间的连续输入（数据）空间，它的拓扑由向量 $\mathbf{x}\in\mathcal{X}$ 的度量关系定义。令 $\mathcal{A}$ 表示空间的离散输出空间，其拓扑由安排一组神经元作为网格的计算节点来赋予。令 Φ 表示称为特征映射的非线性变换，它映射输入空间 $\mathcal{X}$ 到输出（即网格）空间 $\mathcal{A}$，表示为

$$\Phi:\mathcal{X}\rightarrow\mathcal{A} \tag{9.15}$$

式(9.15)可看成式(9.3)的抽象，式(9.3)定义为响应输入向量 $\mathbf{x}$ 而产生的获胜神经元 $i(\mathbf{x})$ 的位置。例如，在神经生物学中输入空间 $\mathcal{X}$ 可以表示密布于整个体表面的体感觉接收器的坐标集。相应地，输出空间 $\mathcal{A}$ 表示体感觉接收器投影到的人脑皮层中的神经元集。

给定输入向量 x，SOM 算法首先根据特征映射 Φ 确定在输出空间 $\mathcal{A}$ 中的最佳匹配或获胜神经元 $i(\mathbf{x})$。神经元 $i(\mathbf{x})$ 的突触权值向量 $\mathbf{w}_i$ 可以视为神经元指向输入空间 $\mathcal{X}$ 的指针。

因此，如图 9.4 所示，SOM 算法包含了两个定义了该算法的成分：

- 从连续输入空间 $\mathcal{X}$ 到离散输出神经元空间 $\mathcal{A}$ 的投影。根据 9.3 节中算法小结的相似性匹配步（即第三步），输入向量被映射到网格结构的“获胜神经元”。
- 从输出空间回到输入空间的指针。实际上，由获胜神经元的权重向量所定义的指针表示输入数据空间中的一个特别点，这个点可作为获胜神经元的映像；这一操作是根据算法小结中的更新步（即第 4 步）迭代完成的。

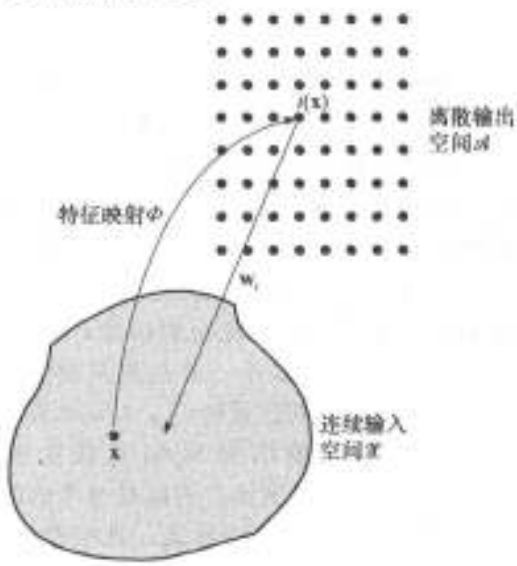


图 9.4　特征映射 Φ 和获胜神经元 i 权值向量 $\mathbf{w}_i$ 的关系图

换句话说，在存在着网格神经元的输出空间和产生样例的输入空间之间有着反向或前向的通信。

SOM 算法有下面将要讨论的一些重要性质。

性质 1　输入空间的近似

由输出空间$\mathcal{A}$的突触权值向量$\{\mathbf{w}_j\}$的集合表示的特征映射Φ对输入空间$\mathcal{X}$提供一个好的近似。

SOM 算法的基本目标是通过寻找原型 $\mathbf{w}_j \in \mathcal{A}$ 的一个较小的集合存储输入向量 $x \in \mathcal{X}$ 的一个大集合，从而对原始输入空间$\mathcal{X}$提供一个好的近似。刚才描述的思想的理论基础植根于向量量化理论（vector quantization theory），它的动机是维数的削减或者是数据的压缩（Gersho and Gray，1992）。因此给出这个理论的简要讨论是适宜的。

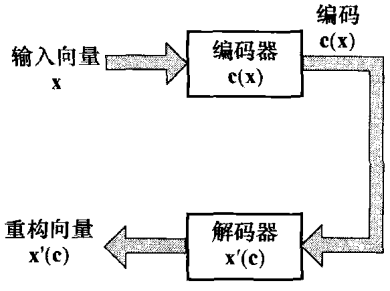


图 9.5　描述 SOM 模型性质 1 的编码器－解码器模型

考虑图 9.5，其中 $\mathbf{c}(\mathbf{x})$作为输入向量 $\mathbf{x}$ 的编码器而$\mathbf{x}'(\mathbf{c})$作为 $\mathbf{c}(\mathbf{x})$的解码器。向量 $\mathbf{x}$ 从满足固有概率密度函数 $p_{\mathbf{x}}(\mathbf{x})$的训练样本（即输入空间$\mathcal{X}$）中随机选择。通过变化函数$\mathbf{c}(\mathbf{x})$和 $\mathbf{x}'(\mathbf{c})$决定最优编码－解码方案使得极小化由

$$D = \frac{1}{2}\int_{-\infty}^{\infty} p_{\mathbf{x}}(\mathbf{x})\mathrm{d}(\mathbf{x},\mathbf{x}')\mathrm{d}\mathbf{x} \tag{9.16}$$

定义的期望失真，其中引入因子$\frac{1}{2}$是为了表达方便，$\mathrm{d}(\mathbf{x},\mathbf{x}')$ 是失真（distortion）度量。积分在假定维数为 m 的整个输入空间$\mathcal{X}$上进行，因此在式(9.16)中使用了微分变量 $\mathrm{d}\mathbf{x}$。失真度量 $\mathrm{d}(\mathbf{x},\mathbf{x}')$的一个常用选择是输入向量 $\mathbf{x}$ 和重建向量 $\mathbf{x}'$之间的欧几里得距离的平方；即

$$\mathrm{d}(\mathbf{x},\mathbf{x}') = \| \mathbf{x}-\mathbf{x}' \|^2 = (\mathbf{x}-\mathbf{x}')^T(\mathbf{x}-\mathbf{x}') \tag{9.17}$$

这样我们可把式(9.16)重写为：

$$D = \frac{1}{2}\int_{-\infty}^{\infty} p_{\mathbf{x}}(\mathbf{x}) \| \mathbf{x}-\mathbf{x}' \|^2 \mathrm{d}\mathbf{x} \tag{9.18}$$

期望失真 D 最小化的必要条件包含在广义 Lloyd 算法中[7]（Gersho and Gray，1992）。条件是两方面的：

条件 1　给定输入向量 $\mathbf{x}$，选择编码 $\mathbf{c}=\mathbf{c}(\mathbf{x})$使其最小化平方误差失真 $\| \mathbf{x}-\mathbf{x}'(\mathbf{c}) \|^2$。

条件 2　给定编码 $\mathbf{c}$，计算重构向量 $\mathbf{x}'=\mathbf{x}'(\mathbf{c})$作为满足条件 1 的输入向量 $\mathbf{x}$ 的中心。

条件 1 称为最近邻编码规则。条件 1 和 2 意味着平均失真 D 关于编码器 $\mathbf{c}(\mathbf{x})$和解码器 $\mathbf{x}'(\mathbf{c})$各自的变化是稳定的（即在局部极小）。为了实现向量量化，广义 Lloyd 算法以批量训练方式进行。基本上，算法是交替按照条件 1 优化编码器 $\mathbf{c}(\mathbf{x})$和按照条件 2 优化解码器$\mathbf{x}'(\mathbf{c})$，直到期望失真 D 达到一个最小。要克服局部最小化问题，可能需要以不同初值运行广义 Lloyd 算法若干次。

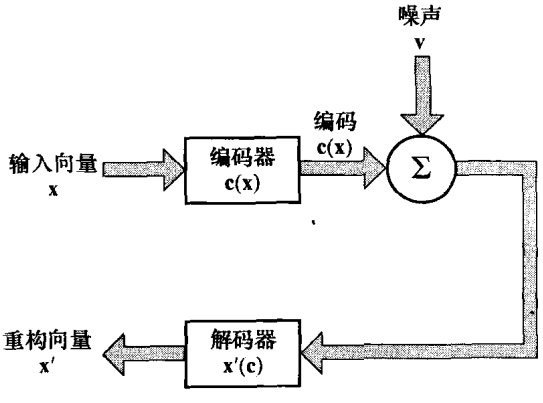


图 9.6　噪声编码器－解码器模型

广义 Lloyd 算法和 SOM 算法紧密相关，如 Luttrell（1989b）所示。可以通过考虑图 9.6 所示的系统描述这种关系的形式，其中在编码器 $\mathbf{c}(\mathbf{x})$之后我们引入了独立于数据的噪声过程。噪声 $\mathbf{v}$ 附加在编码器和解码器之间的虚构的“通信信道”

上，它的目的是说明输出编码 $\mathbf{c}(\mathbf{x})$可能失真的可能性。在图 9.6 所示模型的基础上，可以考虑期望失真的一种修正形式

$$D_1 = \frac{1}{2}\int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x})\int_{-\infty}^{\infty} \pi(\mathbf{v}) \| \mathbf{x} - \mathbf{x}'(\mathbf{c}(\mathbf{x}) + \mathbf{v}) \|^2 \mathrm{d}\mathbf{v}\mathrm{d}\mathbf{x} \tag{9.19}$$

其中 $\pi(\mathbf{v})$为加性噪声 $\mathbf{v}$ 的概率密度函数（pdf），内部积分是对这个噪声的所有可能实现之上的积分，因而在式(9.19)中使用了增加变量 $\mathrm{d}\mathbf{v}$。

根据广义 Lloyd 算法描述的策略，对图 9.6 所示的模型可考虑两个不同的优化，一个属于编码器而另一个属于解码器。为了找到给定 $\mathbf{x}$ 的最优编码器，我们需要期望失真度量 D_1 对编码向量 $\mathbf{c}$ 的偏导数。利用式(9.19)，可得

$$\frac{\partial D_1}{\partial \mathbf{c}} = \frac{1}{2} p_{\mathbf{X}}(\mathbf{x})\int_{-\infty}^{\infty} \pi(\mathbf{v}) \frac{\partial}{\partial \mathbf{c}} \| \mathbf{x} - \mathbf{x}'(\mathbf{c}) \|^2 |_{\mathbf{c}=\mathbf{c}(\mathbf{x})+\mathbf{v}} \mathrm{d}\mathbf{v} \tag{9.20}$$

为了找到给定 $\mathbf{c}$ 的最优解码器，我们需要期望失真度量 D_1 对解码向量 $\mathbf{x}'(\mathbf{c})$的偏导数。利用式(9.19)，可得

$$\frac{\partial D_1}{\partial \mathbf{x}'(\mathbf{c})} = -\int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x})\pi(\mathbf{c} - \mathbf{c}(\mathbf{x}))(\mathbf{x} - \mathbf{x}'(\mathbf{c}))\mathrm{d}\mathbf{x} \tag{9.21}$$

因此，根据式(9.20)和式(9.21)，以前陈述的广义 Lloyd 算法的条件 1 和条件 2 必须修改如下(Luttrell，1989b)：

条件 Ⅰ　*给定输入向量 $\mathbf{x}$，选择编码 $\mathbf{c}=\mathbf{c}(\mathbf{x})$使其最小化失真度量*

$$D_2 = \int_{-\infty}^{\infty} \pi(\mathbf{v}) \| \mathbf{x} - \mathbf{x}'(\mathbf{c}(\mathbf{x}) + \mathbf{v}) \|^2 \mathrm{d}\mathbf{v} \tag{9.22}$$

条件 Ⅱ　*给定编码 $\mathbf{c}$，计算重构向量 $\mathbf{x}'(\mathbf{c})$使其满足条件*

$$\mathbf{x}'(\mathbf{c}) = \frac{\int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x})\pi(\mathbf{c} - \mathbf{c}(\mathbf{x}))\mathbf{x}\mathrm{d}\mathbf{x}}{\int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x})\pi(\mathbf{c} - \mathbf{c}(\mathbf{x}))\mathrm{d}\mathbf{x}} \tag{9.23}$$

令式(9.21)中的偏导数 $\partial D_1/\partial \mathbf{x}'(\mathbf{c})$为 0，然后解出 $\mathbf{x}'(\mathbf{c})$可得式(9.23)。

图 9.5 描述的模型可作为图 9.6 描述的模型的特殊情形。特别地，如果令噪声 $\mathbf{v}$ 的概率密度函数 $\pi(\mathbf{v})$等于 Dirac delta 函数 $\delta(\mathbf{v})$，条件 Ⅰ 和条件 Ⅱ 分别退化为广义 Lloyd 算法的条件 1 和条件 2。

为了简化条件 Ⅰ，假定 $\pi(\mathbf{v})$为 $\mathbf{v}$ 的光滑函数。可以证明式(9.22)定义的失真度量 D_2 的二阶近似包含两项（Luttrell，1989b）：

- 常规失真项，由平方误差失真 $\| \mathbf{x} - \mathbf{x}'(\mathbf{c}) \|^2$ 定义。
- 由噪声模型 $\pi(\mathbf{v})$引起的曲率（curvature）项。

假设曲率项小，对于图 9.6 的模型条件 Ⅰ 可以近似为图 9.5 的无噪声模型的条件 Ⅰ。这样又使条件 Ⅰ 变成以前的最近邻编码规则。

至于条件 Ⅱ，可以使用随机下降学习来实现。具体地，根据 $p_{\mathbf{X}}(\mathbf{x})$从输入空间$\mathscr{X}$随机选择输入向量 $\mathbf{x}$，并且更新重构向量 $\mathbf{x}'(\mathbf{c})$如下：

$$\mathbf{x}'_{\mathrm{new}}(\mathbf{c}) \leftarrow \mathbf{x}'_{\mathrm{old}}(\mathbf{c}) + \eta\pi(\mathbf{c} - \mathbf{c}(\mathbf{x}))[\mathbf{x} - \mathbf{x}'_{\mathrm{old}}(\mathbf{c})] \tag{9.24}$$

其中 η 为学习率参数，$\mathbf{c}(\mathbf{x})$为条件 Ⅰ 的最近邻编码近似。更新式(9.24)由检查式(9.21)的偏导数可得。这个更新应用于所有的 $\mathbf{c}$，对此我们有

$$\pi(\mathbf{c} - \mathbf{c}(\mathbf{x})) > 0 \tag{9.25}$$

可以认为式(9.24)描述的梯度下降过程为式(9.19)的失真度量 D_1 的一种最小化方法。也就是说，式(9.23)和式(9.24)本质上是同类型的，区别在于式(9.23)为批量方式的而式(9.24)为连续的方式（即经过流的方式）。

更新式(9.24)等同于式(9.13)的（连续）SOM 算法，记住在表 9.1 中所列的对应关系。因此，可以说用于向量量化的广义 Lloyd 算法为具有零邻域大小的 SOM 算法的批量训练模式；对零邻域，$\pi(0)=1$。注意，为了从 SOM 算法的批量方式得到广义 Lloyd 算法我们无需作任何近似，因为当邻域为 0 宽度时曲率项（和所有高阶项）不起任何作用。

表 9.1　在 SOM 算法和图 9.6 的模型之间的对应关系

图 9.6 的编码器—解码器模型	SOM 算法
编码器 $\mathbf{c}(\mathbf{x})$	最佳匹配神经元 $i(\mathbf{x})$
重构向量 $\mathbf{x}'(\mathbf{c})$	突触权值向量 $\mathbf{w}_j$
概率密度函数 $\pi(\mathbf{c}-\mathbf{c}(\mathbf{x}))$	邻域函数 $h_{j,i(\mathbf{x})}$

下面给出这里的讨论所需注意的重要之处：

1. SOM 算法为向量量化算法，它提供输入空间$\mathcal{X}$的良好近似。这个观点提供了导出 SOM 算法的另一种途径，如式(9.24)的示例。

2. 根据这个观点，SOM 算法中的邻域函数 $h_{j,i(\mathbf{x})}$ 有一个概率密度函数的形式。在 Luttrell (1991a)，考虑对图 9.6 的模型中噪声 $\mathbf{v}$ 而言合适的零均值高斯模型。因此我们对采用式(9.4)的高斯邻域函数又有了一个理论依据。

用求和作为对式(9.23)右端的分子和分母的积分的近似，批量 SOM[8] 仅仅是式(9.23)的重写。注意在 SOM 算法的这种形式中，输入模式呈现给网络的顺序对特征映射的最终形式没有影响，且无需学习率调度。但算法仍需利用邻域函数。

性质 2　拓扑排序

通过 SOM 算法计算的特征映射 Φ 是拓扑有序的，意味着网格中神经元的空间位置对应于输入模式的特定区域或特征。

拓扑排序的特性[9] 是更新公式(9.13)的直接结果，它使获胜神经元 $i(\mathbf{x})$ 的权值向量 $\mathbf{w}_i$ 移向输入向量 $\mathbf{x}$。它同样对于获胜神经元 $i(\mathbf{x})$ 近邻的神经元 j 的突触权值向量 $\mathbf{w}_j$ 的移动有作用。因此我们可以将特征映射 Φ 看成一个弹性网或虚拟网，它有在输出空间$\mathcal{A}$中描述的一维或二维的网格，并且它的节点具有权值作为输入空间$\mathcal{X}$中的坐标（Ritter，1995）。因此算法的总的目标可以陈述如下：

指针或原型以突触权值向量 $\mathbf{w}_j$ 的形式逼近输入空间$\mathcal{X}$，使得特征映射 Φ 以这样一种方式提供根据某个统计准则而言表征输入向量 $\mathbf{x}\in\mathcal{X}$ 的重要特征的可信赖表示。

特征映射 Φ 通常在输入空间$\mathcal{X}$中显示。具体地，所有的指针（即突触权向量）显示为点，相邻神经元的指针按照网格的拓扑用线相连。因此，使用连线将两个指针 $\mathbf{w}_i$ 和 $\mathbf{w}_j$ 连起来，表示相应神经元 i 和 j 在网格中是相邻神经元。

性质 3　密度匹配

特征映射 Φ 反映输入分布在统计上的变化：在输入空间$\mathcal{X}$中样本向量 $\mathbf{x}$ 以高的概率抽取的区域映射到输出空间的更大区域，从而比在$\mathcal{X}$中样本向量 $\mathbf{x}$ 以低的概率抽取的区域有更好的分辨率。

令 $p_{\mathbf{x}}(\mathbf{x})$ 表示随机输入向量 $\mathbf{x}$ 的多维概率密度函数。由定义，这个 pdf 在整个输入空间上的积分必须等于 1：

$$\int_{-\infty}^{\infty} p_{\mathbf{x}}(\mathbf{x})\mathrm{d}\mathbf{x} = 1$$

令 $m(\mathbf{x})$ 表示映射放大（magnification）因子，定义为输入空间$\mathcal{X}$的小体积 $\mathrm{d}\mathbf{x}$ 中的神经元个数。放大因子在整个输入空间$\mathcal{X}$的积分一定等于网络中的神经元总数 l，即

$$\int_{-\infty}^{\infty} m(\mathbf{x})\mathrm{d}\mathbf{x} = l \tag{9.26}$$

对于准确匹配输入密度的 SOM 算法，我们要求（Amari，1980）

$$m(\mathbf{x}) \propto p_{\mathbf{x}}(\mathbf{x}) \tag{9.27}$$

这个性质意味着，如果输入空间中的一个特殊区域包含经常发生的刺激，那么与刺激出现较少的输入空间的区域相比，它将用特征映射中更大的区域表示。

一般地，在二维特征映射中放大因子 $m(\mathbf{x})$不能表示为输入向量 $\mathbf{x}$ 的概率密度函数 $p_{\mathbf{x}}(\mathbf{x})$的一个简单函数。只有在一维特征映射时才可能导出这样的关系。对这种特殊情况，我们发现与早些的推测（Kohonen，1982）相反，它的放大因子 $m(\mathbf{x})$并不与 $p_{\mathbf{x}}(\mathbf{x})$成比例。基于采用的编码方法，在文献中报告了两种不同的结果：

1. *最小失真编码*，根据这个编码，式(9.22)的失真测度中的曲率项和高阶项由于噪声模型 $\pi(\mathbf{v})$仍然保留。这种编码方法可以产生结果：

$$m(\mathbf{x}) \propto p_{\mathbf{x}}^{1/3}(\mathbf{x}) \tag{9.28}$$

这与标准的向量量化器得到的结果相同（Luttrell，1991a）。

2. *最近邻编码*，如同在 SOM 算法的标准形式中，它出现在忽略曲率项的时候。这个编码方法产生结果（Ritter，1991）

$$m(\mathbf{x}) \propto p_{\mathbf{x}}^{2/3}(\mathbf{x}) \tag{9.29}$$

我们前面关于一族经常发生的刺激可以在特征映射中由更大的区域来表示的陈述仍然成立，虽然是用式(9.27)中描述的理想条件的失真形式。

作为一个一般规则（被计算机仿真确认），由 SOM 算法计算的特征映射往往趋向于过高表示低输入密度区域和过低表示高输入密度区域。换句话说，SOM 算法不能为输入数据固有的概率分布提供可信赖的表示[10]。

性质 4　特征选择

从输入空间中给定数据，自组织映射能够为逼近固有分布选择一组最好的特征。

这个性质是性质 1 至性质 3 的自然结论。性质 4 使人想起前一章讨论的主分量分析的思想，但是如图 9.7 所示，它们有一个重要的区别。在图 9.7a 中展示被加性噪声损坏的线性输入—输出映射导出的零均值数据点的二维分布。这种情况下，主分量分析工作得很好：它告诉我们，在图 9.7a 中的“线性”分布的最好描述是，定义成通过原点且平行于数据相关矩阵的最大特征值对应的特征向量平行的直线（即一维的“超平面”）。接下去考虑图 9.7b 所描述的情况，这是受零均值加性噪声损坏的非线性输入—输出映射的结果。在这第二种情形从主分量分析计算的直线逼近不可能提供可接受的数据描述。另一方面，利用建立在一维神经元网格的自组织映射则由于它的拓扑有序性质能够克服这个逼近问题。在图 9.7b 中说明的后一个逼近仅仅当网格的维数和分布的固有维数匹配时工作良好。

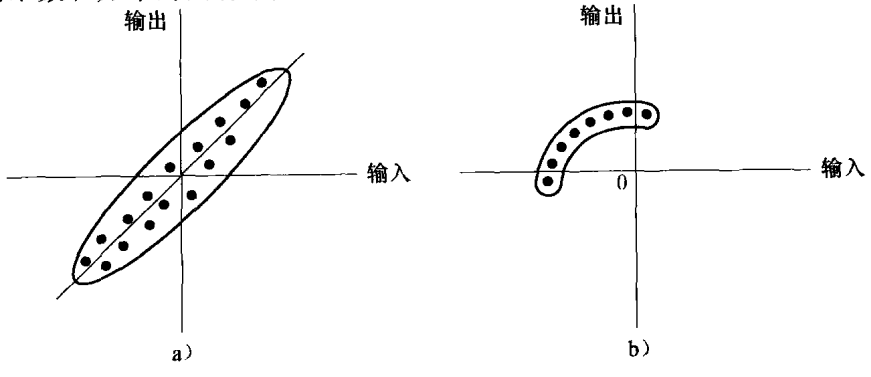


图 9.7　a）线性输入-输出映射产生的二维分布；b）非线性输入-输出映射产生的二维分布

9.5　计算机实验Ⅰ：利用 SOM 解网格动力学问题

Ⅰ. 由二维分布驱动的二维网格

我们使用计算机仿真来说明 SOM 算法的行为，通过研究 576 个神经元组成的网络，排列成 24 行和 24 列的二维网格。网络用二维输入向量 $\mathbf{x}$ 训练，它的分量 x_1 和 x_2 均匀分布在区域 $\{(-1 < x_1 < +1);(-1 < x_2 < +1)\}$ 上。为了初始化网络，突触权值从一个随机集合抽取。

图 9.8 显示训练网络学习表示输入分布的三个阶段。图 9.8a 显示用来训练特征映射的数据的均匀分布。图 9.8b 显示随机抽取的突触权值的初始值。图 9.8c 和图 9.8d 分别表示了在排序阶段和收敛阶段完成后相应的由 SOM 算法计算得到的 24×24 映射。如前面性质 2 所讨论的那样，在图 9.8 中将网络中相邻神经元用线连起来（通过行和列）。

图 9.8 所示的结果展现表征 SOM 算法学习过程特点的排序阶段和收敛阶段。图 9.8c 显示排序阶段，映射展开形成的网格。在这个阶段之后神经元映射为正确的排序。在收敛阶段映射散开充满输入空间。在第二阶段结束后，如图 9.8d 所示，映射中神经元的统计分布接近输入向量的分布，除了一些变形之外。比较图 9.8d 中特征映射的最终状态和图 9.8a 的输入均匀分布，我们看出收敛阶段映射的调整抓住了可在输入分布中看到的局部不规则性。

SOM 算法的拓扑排序性质在图 9.8d 得到很好说明。尤其观察到算法（在收敛之后）抓住了输入中均匀分布的固有拓扑。图 9.8 所示的计算机仿真中输入空间$\mathscr{X}$和输出空间$\mathscr{A}$都是二维的。

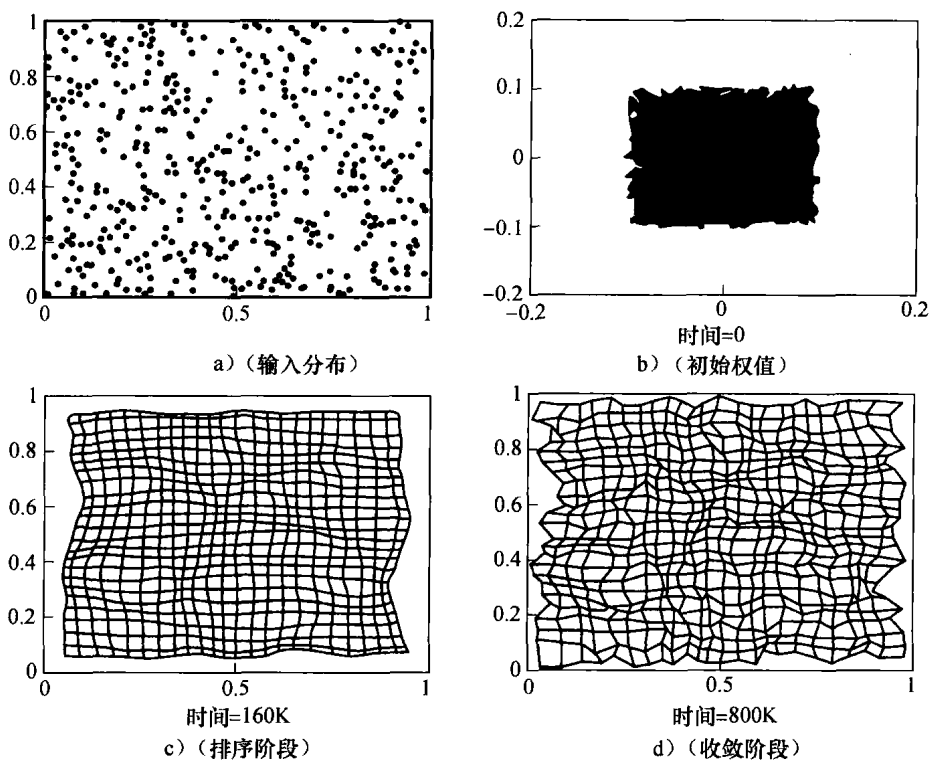


图 9.8　a）输入数据分布。b）二维网格初始情况。c）排序阶段之后网格情况。d）收敛阶段之后网格情况。在映射 b）c）d）之下的时间表示迭代次数

Ⅱ. 由二维刺激驱动的一维网格

我们现在考查当输入空间$\mathscr{X}$的维数大于输出空间$\mathscr{A}$的维数的情况。尽管不匹配，特征映射 Φ 常常能形成输入分布的拓扑表示。图 9.9 显示在特征映射演化过程中的三个不同的阶段，它

的初始化如图 9.9b 所示，从如图 9.9a 所示矩形中抽取数据进行训练，但是，这一次计算是在 100 个神经元的一维网格中进行的。图 9.9c 和图 9.9d 分别表示排序和收敛之后的特征映射。这里我们看到为了尽可能紧密地填充矩形从而提供二维输入空间$\mathcal{X}$的固有拓扑的良好近似，用算法计算的特征映射是非常失真的。在图 9.9d 所示的近似曲线类似于 Peano 曲线（Peano curve)(Kohonon，1990a)。以图 9.9 的特征映射为例的这种运算被称为*维数削减*（dimensionality reduction)，其中输入空间$\mathcal{X}$由将它投影到的低维输出空间$\mathcal{A}$来表示。

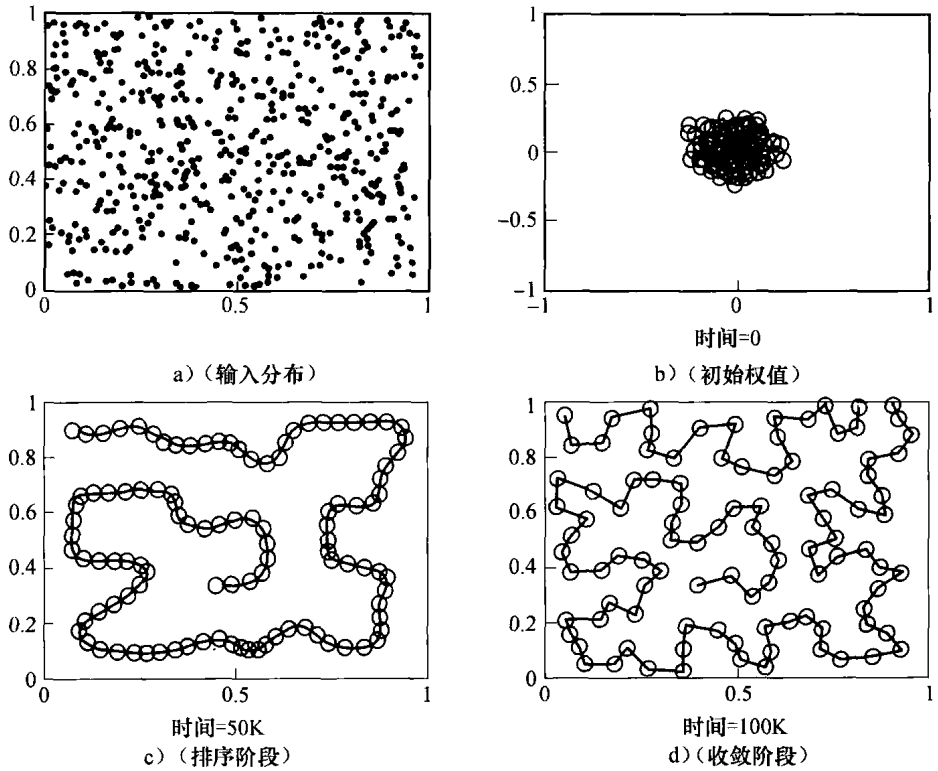


图 9.9　a) 二维输入数据分布；b) 一维网格初始情况；c) 排序阶段之后的网格情况；d) 收敛阶段之后的网格情况。在映射 b)、c)、d) 之下的时间表示迭代次数

9.6　上下文映射

自组织特征映射有两种明显不同的可视化方法。在一种可视化方法中，特征映射被视为有弹性的网络，此时向量权值被视为对应神经元的指针，指向输入空间。这种可视化方法特别适用于显示 SOM 算法的拓扑排序属性，如 9.5 节给出的计算机仿真实验结果所说明。

在第二种可视化方法中，对二维网格（表示网络的输出层）的神经元赋予类别标号，它取决于每个*测试模式*（以前未见过）如何激活自组织网络中的特定神经元。作为仿真第二阶段的结果，二维网格中的神经元被剖分成许多*相干区域*（coherent region)，相干的含义是神经元每个分组表示邻接符号或标号的一个独特的集合（Ritter，2003)。这里首先假定产生良序的特征映射的正确条件成立。

例如，考虑表 9.2 中给出的数据集合，它们是关于 16 种不同动物的。表的每一列是对动物的示意性描述，它是根据左边 13 个不同的属性的出现（=1）或不出现（=0）而描述。一些属性例如“羽毛”和“两条腿”是相关的，而其他许多属性是不相关的。对表头给出的每个动物，它的属性代码 $\mathbf{x}_a$ 是由 13 个属性构成。动物本身由符号代码 $\mathbf{x}_s$ 指定，符号代码的组成必

须不表达动物的任何信息或它们之间已知的相似点。例如当前的例子，$\mathbf{x}_s$ 是由一个列向量构成，它的第 k 个元素，表示动物 $k=1,2,\cdots,16$，赋予一个固定值 a；剩下的元素都置成 0。参数 a 与属性代码比较而言决定符号代码之间的相关影响。为了确定属性代码是重要的，a 选择为 0.2。每个动物的输入向量 $\mathbf{x}$ 是 29 个元素的向量，表示属性代码 $\mathbf{x}_a$ 和符号代码 $\mathbf{x}_s$ 的联合，表示为

$$\mathbf{x}=\begin{bmatrix}\mathbf{x}_s\\ \mathbf{x}_a\end{bmatrix}=\begin{bmatrix}\mathbf{x}_s\\ \mathbf{0}\end{bmatrix}+\begin{bmatrix}\mathbf{0}\\ \mathbf{x}_a\end{bmatrix}$$

最后，每个数据向量都被归一化为单位长度。这样产生的数据集的模式被呈现给 10×10 的二维神经元网格，神经元的权值按照 9.3 节中阐述的 SOM 算法调整。训练连续进行 2 000 次迭代，此时特征映射应该达到一个稳定状态。接着，由一个动物包含的符号代码 $\mathbf{x}=[\mathbf{x}_s,0]^T$ 定义的测试模式呈现给自组织网络，并且确定具有最强响应的神经元。对所有的 16 种动物都重复这样做。

表 9.2　动物的名称和它们的属性

	动物	鸽子	母鸡	鸭	鹅	猫头鹰	隼	鹰	狐狸	狗	狼	猫	虎	狮	马	斑马	母牛
为	小型	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0	0
	中型	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
	大型	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
有	2 条腿	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
	4 条腿	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
	毛发	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
	蹄	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
	鬃	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	0
	羽毛	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
擅长	猎食	0	0	0	0	1	1	1	1	0	1	1	1	1	0	0	0
	奔跑	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0
	飞翔	1	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0
	游泳	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

按刚才陈述的方法处理，我们得到如图 9.10 所示的映射，其中标记名称的神经元代表它们对各自的测试模式有最强的响应，图中未被占据的矩形空间表示有较弱的响应的神经元。

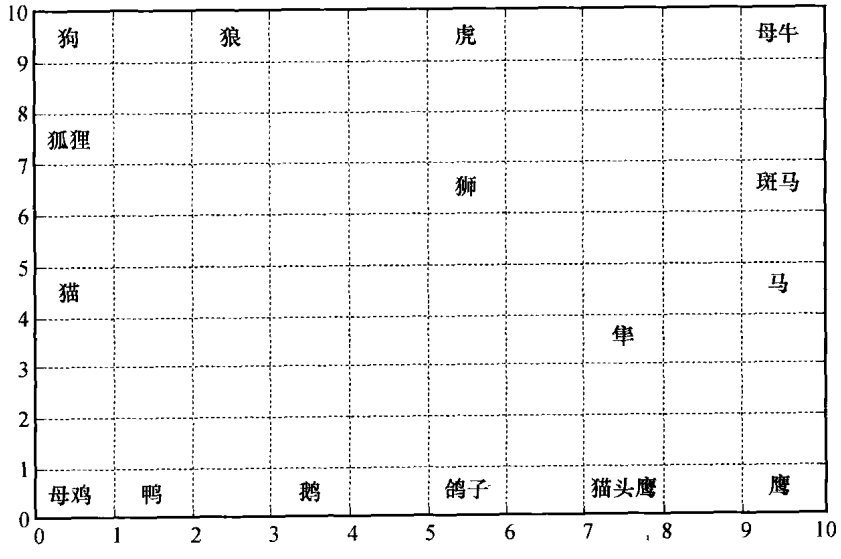


图 9.10　包含对它们各自输入具有最强响应的标定神经元的特征映射

图 9.11 对相同的自组织网络显示“模拟电极渗透映射”的结果。但是，图中网络的每个神经元用使之产生最好响应的特定动物名称标记。图 9.11 清楚地表明在 16 个不同的动物中特

征映射能抓住“种属关系”。这里有三个不同的聚类，第一个表示“鸟类”，第二个表示“平和的种属”，第三个表示“猎手”。

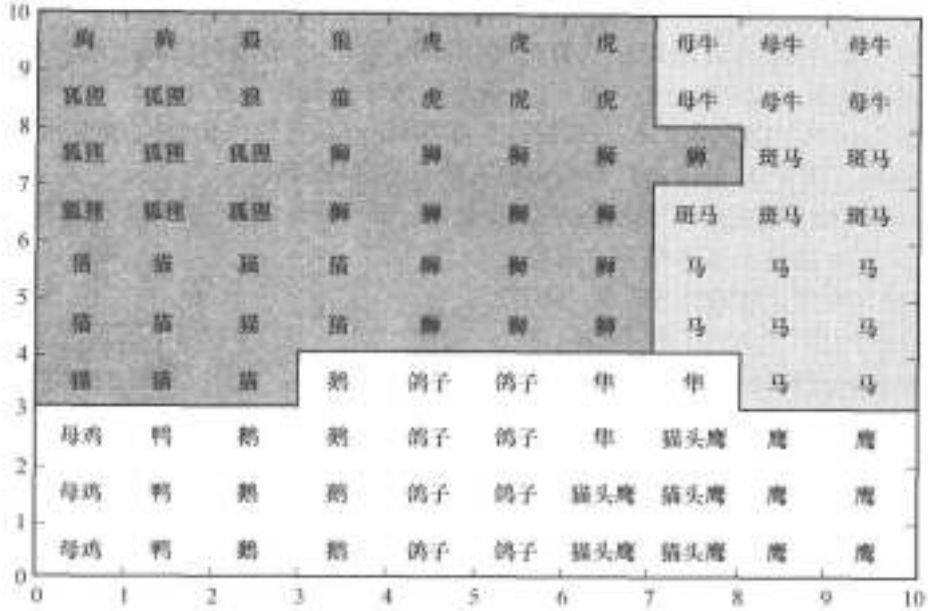


图 9.11 利用“模拟电极渗透映射”的语义映射。映射被分成三个不同区域，分别代表鸟类（白色）、平和种属（浅灰色）及猎手（灰色）

图 9.11 表示的特征映射类型称为上下文映射或语义映射（Ritter,2003）。这个映射与大脑皮质的映射相似（即在大脑皮质里形成的计算映射），这在 9.2 节中做过简要讨论。作为利用 SOM 算法产生的结果，上下文映射在众多领域都有应用，诸如文本的音素类别的无监督分类，遥感（Kohonen，1997a），数据探测或数据挖掘（Kohonen，1997b）。

9.7 分层向量量化

在 9.4 节自组织特征映射的性质 1 的讨论中，我们指出在向量量化方面它与广义 Lloyd 算法紧密相关。向量量化是有损（lossy）数据压缩的一种形式，有损是指一些包含在输入数据中的信息由于压缩的结果丢失了。数据压缩植根于香农信息论的一个分支，称为率失真（rate distortion）理论（Cover and Thomas，2002）。目前要处理分层向量量化，以陈述下面率失真理论的基本结果作为开始是很适合的（Gray，1984）：

通过获得向量编码而不是标量编码，总是能够取得好的数据压缩性能，即使数据源是无记忆的（例如，它提供一系列独立随机变量），或者数据压缩系统有记忆（即编码器的动作依赖于编码器以前的输入或输出）。

这一基本结果成为数十年来对向量量化的广泛研究工作的基础。

然而，传统的向量量化算法要求大量的计算。向量量化最费时的部分是编码操作。在编码过程中，输入向量必须与每一个在码书中的代码向量作比较，以便决定哪一个特别的代码产生最小失真度。例如对于码书包含 N 个码向量，编码所花的时间依赖于 N 的阶，这样对大的 N 值所花时间就多。在 Luttrell(1989a) 中描述了一个多阶段分层（multistage hierarchical）向量量化器，它用精度换取编码速度。多阶段分层向量量化器试图将所有的向量量化过程分解成许多子操作，每个子操作仅要求少量的计算。理想的分解对每个子操作简化为简单的查表。通过巧妙地使用 SOM 算法来训练量化器的每一阶段，准确性的丢失可能很少（低到几分之一分

贝（decibel)），同时计算速度的增益可能很大。

考虑两个向量量化器 VQ_1 和 VQ_2，其中 VQ_1 将它的输出送到 VQ_2 作为其输入。VQ_2 的输出是应用于 VQ_1 的原输入信号的最终编码形式。在运行它的量化过程中，VQ_2 不可避免地抛弃一些信息。就 VQ_1 而言，VQ_2 仅有的作用是扭曲 VQ_1 输出的信息。这样很明显对 VQ_1 的正确的训练方法是 SOM 算法，它对 VQ_2 诱导的信号失真负责（Luttrell,1989a)。要使用广义 Lloyd 算法来训练 VQ_2，我们仅需要假定 VQ_2 的输出在重建之前没有被损坏。从而我们无需引入噪声模型（在 VQ_2 的输出）及相应的有限宽度邻域函数。

我们可以推广这个启发式的结论到多阶段量化器。必须设计每一阶段使之考虑所有的后面阶段导致的失真并且为它建立噪声模型。因此，使用 SOM 算法训练量化器的所有阶段，除了最后一个阶段适宜用广义 Lloyd 算法训练。

分层向量量化过程是多阶段向量量化的特例。作为一种例证，考虑 4×1 的输入向量

$$\mathbf{x} = [x_1, x_2, x_3, x_4,]^T$$

的量化。在图 9.12a 中给出用于 $\mathbf{x}$ 的单阶段向量量化器。另外，可以使用如图 9.12b 所描绘的两阶段分层量化器。这两个模式的重要区别是在图 9.12a 的量化器输入维数为 4 而在图 9.12b 中它是 2。因此，图 9.12b 的量化器要求小规模的查找表，因此比图 9.12a 的量化器实现简单。这是分层量化器比传统量化器优越之处。

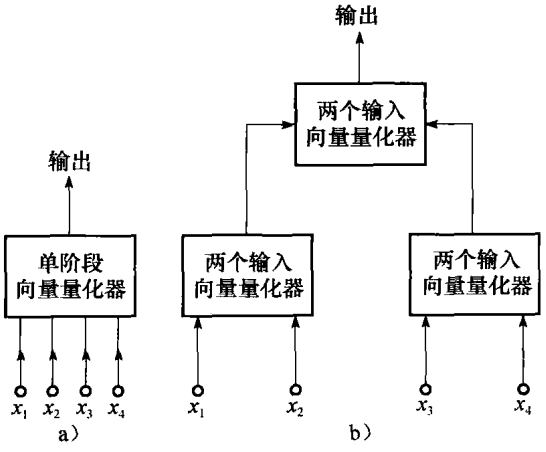


图 9.12　a）具有四维输入的单阶段向量量化器；b）使用两个输入的两阶段分层向量量化器（摘自 S. P. Luttrell（1989a)，British Crown 版权)

案例研究　一阶自回归模型

Luttrell（1989a）展示了多阶段分层向量量化器应用到不同的随机时间序列的性能，编码准确度丢失很少。在图 9.13 中我们利用一阶自回归（AR）模型

$$x(n+1) = \rho x(n) + \nu(n) \tag{9.30}$$

产生了具有相关高斯噪声过程的 Luttrell 的结果，其中 ρ 为 AR 系数，$\nu(n)$为具有零均值和单位方差的统计独立同分布（iid）高斯随机变量集合中取得。因此我们可以证明 $x(n)$的统计特征如下：

$$\mathbb{E}[x(n)] = 0 \tag{9.31}$$

$$\mathbb{E}[x^2(n)] = \frac{1}{1-\rho^2} \tag{9.32}$$

$$\frac{\mathbb{E}[x(n+1)x(n)]}{\mathbb{E}[x^2(n)]} = \rho \tag{9.33}$$

因此 ρ 也可看成时间序列 $\{x(n)\}$ 的相关系数。要按照式(9.30)初始化生成的时间序列，对 $x(0)$使用均值为零和方差为 $1/(1-\rho^2)$的高斯随机变量，并且相关系数使用 $\rho=0.85$。

对于向量量化使用类似于图 9.12b 中的二分树一样具有四维输入空间的分层编码器。对于 AR 时间序列$\{x(n)\}$，平移对称意味着仅需两个不同的查找表。每张表的大小按指数依赖于输入比特数，而线性依赖于输出比特数。在训练过程中，需要大量比特数表示式(9.24)描述的更新的正确计算数；这样在训练期间不使用查找表。但是一旦训练完成，比特数可降低至它们的正常水平，并且按要求填充表项。对于如图 9.12b 显示的编码器，每个输入样本用 4 比特近似。对解码器的各个阶段，使用 $N(=17)$个码字向量，这样从每个查找表的输出比特数也近似

为 4。因此第一阶段和第二阶段的查找表的地址空间的大小为 256($=2^{4+4}$)，这意味着查找表的表示所需存储要求是适中的。

图 9.13 显示用 $x(n)$作为输入得到的编码-解码结果。图 9.13a 的下半部分显示两阶段中每个阶段的编码向量为一条嵌入二维输入空间的曲线；图 9.13a 的上半部分表示相应的用 16×16 比特的共生（co-occurrence）矩阵的估计。图 9.13b 表示如下时间序列片段。

- 由第一个编码阶段计算的编码向量。
- 保持其他变量固定，由第二阶段最小化均值平方失真计算出的重构向量。

图 9.13c 显示 512 个样本，包括原始时间序列（顶部曲线）和从最后一个编码器阶段的输出得到的它的重构（底部曲线）；图 9.13c 的水平方向的刻度是图 9.13b 的一半。最后，图 9.13d 表示从一对样本（原始时间序列样本和它的相应重构）产生的共生矩阵。图 9.13d 中的带宽指示由分层向量量化产生的失真程度。

检查图 9.13c 的波形，可以看出除了一些正的和负的峰值被剪除之外重构是对原始时间序列的好的表示。根据 Luttrell(1989a)，计算得到的归一化后的均值平方失真同每个样本用一比特的单阶段 4－样本块编码器所获得的几乎一样好（Jayant and Noll，1984）。

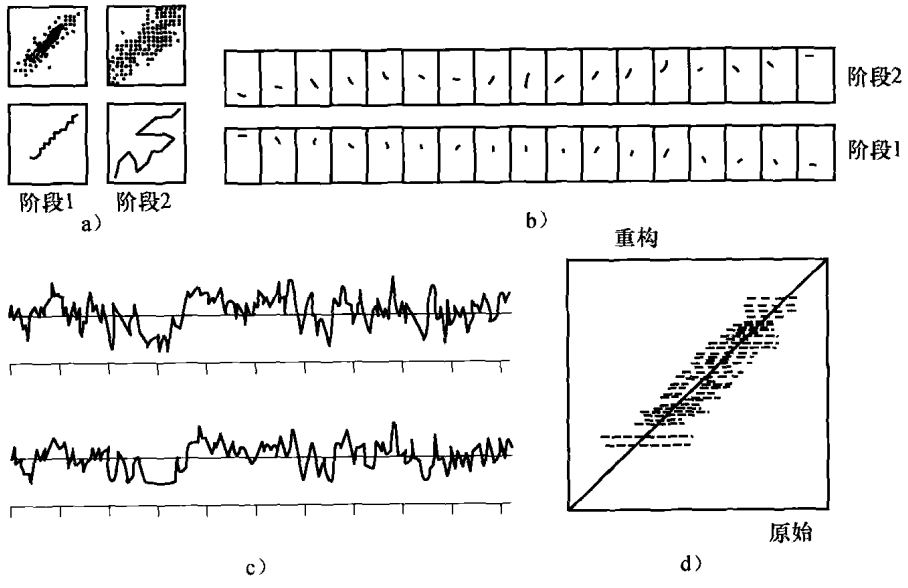


图 9.13　用于相关高斯噪声输压缩的两阶段编码/解码结果。相关系数 $\rho=0.85$
（摘自 S. P. Luttrell(1989a)，British Crown 版权）

9.8　核自组织映射

Kohonen 的自组织映射算法对于探测大量高维数据是很强大的工具，这从多个大规模视觉和数据挖掘应用中得到了例证。然而，从理论的角度，自组织映射存在着两个基本的局限：

1. 由算法提供的输入空间概率密度函数的估计缺少精度。实际上，在图 9.8 的实验结果中已经说明了算法的这一缺点。这一缺点也从理论上是存在的，在式(9.28)或式(9.29)中，无论哪一个，算法的密度匹配性质都是不完美的。

2. 算法的构成中不存在可以用于最优化的目标函数。考虑算法的非线性随机特征，缺少目标函数使得对于收敛性的证明这一问题变得更加困难。

实际上，很大程度上是因为自组织映射的这两个局限，尤其是后者，促使很多研究者设计不同的途径来构成特征映射模型。在本节中我们描述由 Van Hulle(2002b) 提出的基于核的自

组织映射形式，其动机在于改善拓扑映射。

目标函数

在我们前面讨论的核方法的应用中，以支持向量机（SVM）和核主分量分析为例，核参数通常是固定的。与之相比，在核自组织映射中，网格结构的每个神经元作为一个核。这样使得核参数根据预定义的目标函数各自调整，而目标函数迭代性地最大化以便形成满意的拓扑映射。

在本节中，我们集中注意力于核（即神经元）输出的联合熵（joint entropy），称之为*目标函数*。熵的记号在第 10 章中详细讨论。对于目前而言，足够通过新概念的定义而开始讨论。考虑连续随机变量 Y_i，其概率密度函数定义为 $p_{Y_i}(y_i)$，其中样本值 y_i 位于范围 $0 \leqslant y_i < \infty$。Y_i 的微分熵（differential entropy）定义为：

$$H(Y_i) = -\int_{-\infty}^{\infty} p_{Y_i}(y_i) \log p_{y_i}(y_i) \mathrm{d}y_i \tag{9.34}$$

这里用 log 来定义对数以便和第 10 章的术语相一致。对于核 SOM，随机变量 Y_i 与网格中第 i 个核的输出相关联，y_i 是 Y_i 的一个样本值。

在下面，我们将进行自底向上方式：

- 首先最大化给定核的微分熵。
- 然后，当已经达到最大化时，调整核参数来最大化核输出和输入之间的交互信息。我们将在后面对第二个新概念作进一步说明。

核的定义

记核为 $k(\mathbf{x}, \mathbf{w}_i, \sigma_i)$，其中 $\mathbf{x}$ 是 m 维输入向量，$\mathbf{w}_i$ 是第 i 个核的权值（参数）向量，σ_i 是宽；索引 $i = 1,2,\cdots,l$，其中 l 是构成映射的网格结构的神经元总个数。分配索引 i 给核宽以及权向量的基本原理是这两个参数将被迭代性地调整。由于核呈放射状地围绕其中心对称，定义为 $\mathbf{w}_i$，我们有

$$k(\mathbf{x}, \mathbf{w}_i, \sigma_i) = k(\| \mathbf{x} - \mathbf{w}_i \|, \sigma_i), \quad i = 1,2,\cdots,l \tag{9.35}$$

其中 $\| \mathbf{x} - \mathbf{w}_i \|$ 是输入向量 $\mathbf{x}$ 和权值向量 $\mathbf{w}_i$ 之间的欧几里得距离，这两者具有相同的维数。

现在，正如 SVM 和核 PCA 的例子中所示，我们期望用概率分布（即某种高斯形式）来定义核。我们也将寻找概率分布但采用核的不同定义，这将在下面解释。

设核输出 y_i 具有“有界”支撑。则由式(9.34)定义的微分熵 $H(Y_i)$将在 Y_i 服从均匀分布时达到最大。（关于这一陈述的证明在于熵是随机性的测量，而均匀分布是随机性的极端形式。）刚提到的最优性的条件在当输出分布和输入空间的累积分布函数相匹配时发生。对于高斯分布输入向量 $\mathbf{x}$，我们发现相应的欧几里得距离 $\mathbf{x} - \mathbf{w}_i$ 的累积分布函数是不完全 gamma 分布（incomplete gamma distribution）。将在后面加以定义的这一分布是所期望的核的定义。

令输入向量 $\mathbf{x}$ 的 m 个元素是统计独立同分布（iid）的，第 j 个元素服从均值为 μ_j 方差为 σ^2 的高斯分布。令 v 定义输入向量 $\mathbf{x}$ 和均值向量 $\boldsymbol{\mu} = [\mu_1, \mu_2, \cdots, \mu_m]^T$ 之间的欧几里得距离的平方，如下所示：

$$v = \| \mathbf{x} - \boldsymbol{\mu} \|^2 = \sum_{j=1}^{m} (\mathbf{x}_j - \boldsymbol{\mu}_j)^2 \tag{9.36}$$

随机变量 V，由样本值 v 表示，具有卡方分布（chi-square distribution），如下所示（Abramowitzand Stegun，1965）：

$$p_V(v) = \frac{1}{\sigma^m 2^{m/2} \Gamma(m/2)} v^{(m/2)-1} \exp\left(-\frac{v}{2\sigma^2}\right), \quad v \geqslant 0 \tag{9.37}$$

其中 m 是分布的自由度个数（number of degrees of freedom），$\Gamma(\cdot)$是 gamma 函数，定义为：

$$\Gamma(\alpha) = \int_0^{\infty} z^{\alpha-1} \exp(-z) \mathrm{d}z \tag{9.38}$$

令 r 记到核中心的半径距离，定义为：

$$r = v^{1/2} = \| \mathbf{x} - \boldsymbol{\mu} \| \tag{9.39}$$

这表示了新的随机变量 R 的样本值。然后，利用将随机变量 V 变换为随机变量 R 的规则，我们写成：

$$p_R(r) = \frac{p_V(v)}{\left| \frac{\partial r}{\partial v} \right|} \tag{9.40}$$

利用这一变换，我们发现经过一些合适的代数操作之后，由样本值 r 表示的随机变量 R 的概率密度函数由下式给出（参看习题 9.8）：

$$p_R(r) = \begin{cases} \dfrac{1}{2^{(m/2)-1}\Gamma(m/2)} \left(\dfrac{r}{\sigma}\right)^{m-1} \exp\left(-\dfrac{r^2}{2\sigma^2}\right), & r \geqslant 0 \\ 0, & r < 0 \end{cases} \tag{9.41}$$

图 9.14 中的连续曲线是 $p_R(r)$ 对于距离 r 的单位方差及 $m = 1,2,3\cdots$ 的概率密度函数图。从这些图中我们看出随着输入空间维数 m 的增加，$p_R(r)$ 快速接近高斯函数。更具体地说，逼近高斯函数的二阶统计参数定义为（Van Hulle，2002b）

$$\left.\begin{aligned} \mathbb{E}(R) &\approx \sqrt{m}\sigma \\ \mathrm{Var}[R] &\approx \frac{\sigma^2}{2} \end{aligned}\right\} \text{对于大的 } m \tag{9.42}$$

随机变量 R 的累积分布函数将在习题 9.9 的(a)中提及，其解由不完全 gamma 分布定义（Abramowitz and Stegun，1965）：

$$p_R(r|m) = \underbrace{\sigma\left(1 - \frac{\Gamma\left(\frac{m}{2}, \frac{r^2}{2\sigma^2}\right)}{\Gamma\left(\frac{m}{2}\right)}\right)}_{\text{完全gamma分布}} \tag{9.43}$$

因子 $\Gamma\left(\frac{m}{2}, \frac{r^2}{2\sigma^2}\right)/\Gamma\left(\frac{m}{2}\right)$ 是不完全 gamma 分布的补（complement of the incomplete gamma distribution），其对单位方差和增长的 m 关于距离 r 的图也包含在图 9.14 的短划线中。这些曲线也提供了期望核的图形。具体来说，将 r^2 看成是输入向量 $\mathbf{x}$ 和第 i 个神经元的权值向量 $\mathbf{w}_i$ 之间欧几里得距离的平方，最后相应的核 $k(\mathbf{x}, \mathbf{w}_i, \sigma_i)$ 定义如下（Van Hulle，2002b）：

$$k(\mathbf{x}, \mathbf{w}_i, \sigma_i) = \frac{1}{\Gamma\left(\frac{m}{2}\right)} \Gamma\left(\frac{m}{2}, \frac{\| \mathbf{x} - \mathbf{w}_i \|^2}{2\sigma_i^2}\right), \quad i = 1,2,\cdots,l \tag{9.44}$$

注意以 $r = \| \mathbf{x} - \mathbf{w}_i \|$ 为中心的核对于所有的 i 是放射状对称的。更重要的是，不完全 gamma 分布的采用保证了当输入分布是高斯时核的微分是最大的。

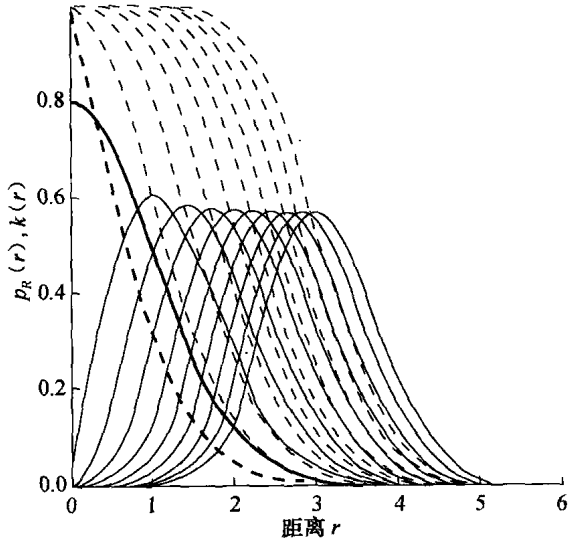


图 9.14 显示了对距离 r 的两个不同图集的图，对于单位方差和增长的维数 $m=1, 2, 3, \cdots$：
- 连续曲线是式(9.41)的概率密度函数
- 短划线是不完全 gamma 分布的补图，或者等价于式(9.44)中 $r = \| \mathbf{x} - \mathbf{w} \|$ 的核 $k(r)$ 的图

(这个图的复制得到了 Dr. Marc Van Hulle 的许可)

映射构造的学习算法

有了式(9.44)的核函数，我们现在为构成自组织拓扑公式的算法做好了准备，在映射中利用核函数来描述每个神经元。

我们通过推导由(9.34)式定义的目标函数对于核参数（权值向量 $\mathbf{w}_i$ 和核宽 σ_i ，$i=1,2,\cdots,l$）的梯度公式来开始。然而，如目前的状况，目标函数 $H(Y_i)$是定义在第 i 个神经元输出之上：

$$y_i = k(\mathbf{x},\mathbf{w}_i,\sigma_i), \quad i=1,2,\cdots,l \tag{9.45}$$

另一方面，式(9.41)的分布是定义在到核的中心的半径距离 r 之上的。因而我们需要将随机变量 R 变换到 Y_i，且相应地得到：

$$p_{Y_i}(y_i) = \frac{p_R(r)}{\left|\frac{\mathrm{d}y_i}{\mathrm{d}r}\right|} \tag{9.46}$$

这里右端的分母部分说明 y_i 对于 r 的依赖性。因此，将式(9.46)代入式(9.34)，可以重新定义目标函数 $H(Y_i)$为：

$$H(Y_i) = -\int_0^\infty p_R(r)\log\, p_R(r)\mathrm{d}r + \int_0^\infty p_R(r)\log\left|\frac{\partial y_i(r)}{\partial r}\right|\mathrm{d}r \tag{9.47}$$

为了进一步做下去，首先考虑 $H(Y_i)$关于权值向量 $\mathbf{w}_i$ 的梯度。等式右端的第一项独立于 $\mathbf{w}_i$。第二项是偏导数 $\log|(\partial y_i(r))/dr|$ 的期望。因此可以将 $H(Y_i)$对于 $\mathbf{w}_i$ 的导数表达为：

$$\frac{\partial H(Y_i)}{\partial \mathbf{w}_i} = \frac{\partial}{\partial \mathbf{w}_i}\mathbb{E}\left[\log\left|\frac{\partial y_i(r)}{\partial r}\right|\right] \tag{9.48}$$

现在假设对于每个核我们从 r 的一个训练样本开始来逼近概率密度函数 $p_R(r)$以最大化核输出 $y_i(r)$的微分熵。然后将式(9.48)的右端项的期望用确定量来代替，如下所示：

$$\mathbb{E}\left[\log\left|\frac{\partial y_i(r)}{\partial r}\right|\right] = \log\left|\frac{\partial \bar{y}_i(r)}{\partial r}\right| \tag{9.49}$$

其中 $\bar{y}_i(r)$是 $y_i(r)$在 r 的训练样本之上的平均值。相应地，重写式(9.48)为简单形式：

$$\begin{aligned}\frac{\partial H(\bar{y}_i)}{\partial \mathbf{w}_i} &= \frac{\partial}{\partial \mathbf{w}_i}\left(\log\left|\frac{\partial \bar{y}_i(r)}{\partial r}\right|\right)\\ &= \frac{\partial r}{\partial \mathbf{w}_i}\frac{\partial}{\partial r}\left(\log\left|\frac{\partial \bar{y}_i(r)}{\partial r}\right|\right)\end{aligned} \tag{9.50}$$

平均值 $\bar{y}_i(r)$具有和式(9.44)定义的不完全 gamma 分布相似的形式，它的使用产生了（参看习题 9.9 的(b)）：

$$\frac{\partial \bar{y}_i(r)}{\partial r} = \frac{-2}{\Gamma(m/2)(\sqrt{2}\sigma_i)^{m-1}} r^{m-1}\exp\left(-\frac{r^2}{2\sigma^2}\right) \tag{9.51}$$

回忆核是以下面的点为中心而对称的：

$$r = \|\mathbf{x}-\mathbf{w}_i\|$$

因而，实现式(9.51)中的 $\partial \bar{y}_i(r)/\partial r$ 对 $\mathbf{w}_i$ 的偏微分且将其结果代入式(9.50)，得到（经过简化）

$$\frac{\partial H(\bar{y}_i)}{\partial \mathbf{w}_i} = \frac{\mathbf{x}-\mathbf{w}_i}{\sigma_i^2} - (m-1)\left(\frac{\mathbf{x}-\mathbf{w}_i}{\|\mathbf{x}-\mathbf{w}_i\|^2}\right) \tag{9.52}$$

下面关于式(9.52)的两个备注是值得注意的：

（ⅰ）等式的右端两项对于大的迭代次数收敛到输入向量 $\mathbf{x}$ 的中心。

（ⅱ）对于维数 m 的高斯分布输入向量 $\mathbf{x}$，从前面的讨论中我们知道期望为：

$$\mathbb{E}[\|\mathbf{x}-\mathbf{w}_i\|^2] = m\sigma_i^2 \tag{9.53}$$

因此，对于所有 m，等式的右端第二项希望比第一项更小。

从计算的观点看，高度期望简化式(9.52)使得我们可以对关于权值向量 $\mathbf{w}_i$ 的更新规则利用单一的学习率参数来完成[11]。对此我们选择一个启发式建议：将平方欧几里得项 $\|\mathbf{x}-\mathbf{w}_i\|^2$ 用式(9.53)的期望值来代替，因而可通过如下方式来逼近式(9.52)：

$$\frac{\partial H(\bar{y}_i)}{\partial \mathbf{w}_i} \approx \frac{\mathbf{x}-\mathbf{w}_i}{m\sigma_i^2}, \quad \text{对于所有 } i \tag{9.54}$$

最大化目标函数，权值更新很自然地作用在式(9.54)的梯度向量的相同方向上，与梯度上升(gradient ascent) 相一致。我们可以写：

$$\Delta \mathbf{w}_i = \eta_w \left(\frac{\partial H(\bar{y}_i)}{\partial \mathbf{w}_i} \right)$$

其中 η_w 是小的学习率参数。将输入向量 $\mathbf{x}$ 的固定维数 m 吸收到 η_w，我们最后可以表示权值更新为：

$$\Delta \mathbf{w}_i \approx \eta_w \left(\frac{\mathbf{x}-\mathbf{w}_i}{\sigma_i^2} \right) \tag{9.55}$$

因此关于核 SOM 算法的第一个更新公式为：

$$\mathbf{w}_i^+ = \mathbf{w}_i + \Delta \mathbf{w}_i = \mathbf{w}_i + \eta_w \left(\frac{\mathbf{x}-\mathbf{w}_i}{\sigma_i^2} \right) \tag{9.56}$$

其中 $\mathbf{w}_i$ 和 $\mathbf{w}_i^+$ 分别表示老的和更新后的神经元 i 的权值向量的值。

下面考虑目标函数 $H(\bar{y}_i)$对于核宽 σ_i 的梯度向量。同以前所讲述的梯度向量 $\partial H(\bar{y}_i)/(\partial \mathbf{w}_i)$ 相似的方式进行，得到：

$$\frac{\partial H(\bar{y}_i)}{\partial \sigma_i} = \frac{1}{\sigma_i} \left(\frac{\|\mathbf{x}-\mathbf{w}_i\|^2}{m\sigma_i^2} - 1 \right) \tag{9.57}$$

然后定义核宽的调整为：

$$\Delta \sigma_i = \eta_\sigma \frac{\partial H(\bar{y}_i)}{\partial \sigma_i} = \frac{\eta_\sigma}{\sigma_i} \left(\frac{\|\mathbf{x}-\mathbf{w}_i\|^2}{m\sigma_i^2} - 1 \right) \tag{9.58}$$

其中 η_σ 为第二个学习率参数。对于核 SOM 算法的第二个更新公式，我们有

$$\sigma_i^+ = \sigma_i + \Delta \sigma_i = \sigma_i + \frac{\eta_\sigma}{\sigma_i} \left(\frac{\|\mathbf{x}-\mathbf{w}_i\|^2}{m\sigma_i^2} - 1 \right) \tag{9.59}$$

由式(9.56)和式(9.59)给出的两个更新规则对于单一神经元工作良好。下面我们考虑对于多个神经元的网络的扩展。

目标函数的联合最大化

在一个神经元接着一个神经元的基础上最大化目标函数 $H(\bar{y}_i)$对于可使用的算法而言是不充分的。为了了解为什么这是真的，考虑由两个神经元组成的网格，其相应的核输出记为 y_1 和 y_2。当使用式(9.56)和式(9.59)的更新公式时，例如假设高斯输入分布，这两个神经元核最终将相互一致；换句话说，两个核输出 y_1 和 y_2 成为统计相关。为了预防这一不满意的可能性（为了尽可能保持 y_1 和 y_2 之间的统计独立性），我们需要通过将核自适应放入竞争学习框架来最大化目标函数 $H(\bar{y}_i)$，这和我们推导 Kohonen 的 SOM 算法时是一样的。则在竞争中获胜的神经元的核将要降低其和邻域神经元交互作用的范围，尤其当获胜神经元是强烈活跃时；因此，邻域神经元之间的覆盖减少了。而且，正如在 Kohonen 的 SOM 算法中那样，为了对输入空间的数据分布拓扑保持其神经元网格，我们对学习过程强加一个邻域函数。相应地，竞争学习和邻域函数的组合使用使得我们能够对多个神经元运用两个更新规则，这将在下面讨论。

拓扑映射构造

考虑由 l 个神经元组成的网格$\mathcal{A}$，这些神经元是由相应的核集（不完全 gamma 分布的补）

刻画的：

$$k(\mathbf{x},\mathbf{w}_i,\sigma_i), \qquad i=1,2,\cdots,l \tag{9.60}$$

带着拓扑映射构造的目的，我们引入基于活跃程度的在网格$\mathcal{A}$的 l 个神经元之间的竞争，获胜神经元被定义为：

$$i(\mathbf{x}) = \arg\max_i y_j(\mathbf{x}), \quad \text{当 } j \in \mathcal{A} \tag{9.61}$$

注意这里的相似性匹配准则和式(9.3)的形式不同，式(9.3)是基于最短距离神经元竞争的。式(9.3)和式(9.61)这两个准则仅仅在当所有的神经元核都具有相同的宽（半径）时才等价。

为了提供拓扑映射构造所需要的信息，正如 Kohonen 的 SOM 那样，我们引入邻域函数 $h_{j,i(\mathbf{x})}$，以获胜神经元 $i(\mathbf{x})$为中心。而且，根据 9.3 节的讨论，我们采用距获胜神经元 $i(\mathbf{x})$ 的网格距离的单调减函数。特别地，选择式(9.4)的高斯函数，这里复制如下：

$$h_{j,i(\mathbf{x})} = \exp\left(-\frac{\|\mathbf{x}_j-\mathbf{w}_i\|^2}{2\sigma^2}\right), \qquad j \in \mathcal{A} \tag{9.62}$$

这里 σ 记邻域函数 $h_{j,i(\mathbf{x})}$ 的范围；不要将邻域范围 σ 和核宽 σ_i 相混淆。

核 SOM 算法小结

现在我们为描述核自组织映射的步骤做好了准备：

1. *初始化*。对初始权值向量 $\mathbf{w}_i(0)$和核宽 $\sigma_i(0)(i=1,2,\cdots,l)$ 选择随机值，这里 l 是网格结构中神经元的总个数。这里仅有的限制是对不同的神经元 $\mathbf{w}_i(0)$ 和 $\sigma_i(0)$ 也不同。

2. *取样*。从输入分布中按一定的概率取出一个样本 $\mathbf{x}$。

3. *相似性匹配*。在算法的时间步 n，用下面的准则来确定获胜神经元 $i(\mathbf{x})$：

$$i(\mathbf{x}) = \arg\max_j y_j(\mathbf{x}), \qquad j=1,2,\cdots,l$$

4. *自适应*。调整权值向量和每个核的宽，使用相应的更新公式：

$$\mathbf{w}_j(n+1) = \begin{cases} \mathbf{w}_j(n) + \dfrac{\eta_w h_{j,i(\mathbf{x})}}{\sigma_j^2}(\mathbf{x}(n)-\mathbf{w}_j(n)), & j \in \mathcal{A} \\ \mathbf{w}_j(n), & \text{否则} \end{cases} \tag{9.63}$$

$$\sigma_j(n+1) = \begin{cases} \sigma_j(n) + \dfrac{\eta_\sigma h_{j,i(\mathbf{x})}}{\sigma_j(n)}\left[\dfrac{\|\mathbf{x}(n)-\mathbf{w}_j(n)\|^2}{m\sigma_j^2(n)} - 1\right], & j \in \mathcal{A} \\ \sigma_j(n), & \text{否则} \end{cases} \tag{9.64}$$

这里 η_w 和 η_σ 为学习算法的两个学习率参数，$h_{j,i(\mathbf{x})}$ 是以获胜神经元 $i(\mathbf{x})$为中心的邻域函数，根据式(9.61)定义。如 Kohonen 的 SOM，邻域范围 σ 允许随时间指数衰减。

9.9　计算机实验Ⅱ：利用核 SOM 解点阵动力学问题

在这一试验中，我们回顾二维网格，这已经在 9.5 节的计算机实验Ⅰ中进行了研究。这一次实验中我们采用核 SOM。选择算法中的两个学习率参数为：

$$\eta_w = 0.01$$

和

$$\eta_\sigma = 10^{-4}\eta_w$$

二维网格是由 24×24 神经元组成的方格，输入数据是均匀分布的。权值的初始化是从同样的输入分布中取样的，半径的初始化是从均匀分布 [0,0.1] 中取样的。用高斯函数作为邻域函数，其宽为

$$\sigma(n) = \sigma_0\exp\left(-2\sigma_0\left(\frac{n}{n_{\max}}\right)\right)$$

这里 n_{max} 记为最大时间步次数，σ_0 记时间 $n=0$ 时邻域函数张开的范围。实验中使用的值为

$$n_{max} = 2 \times 10^6$$

和

$$\sigma_0 = 12$$

做这样的选择是为了确保在学习过程结束时邻域函数将消失。在那一点上近似值为 4.5×10^{-10}，这实际上是 0。当最终达到这一条件时，邻域函数仅仅围绕获胜神经元。

图 9.15 表示的两个序列图示了核 SOM 算法产生的拓扑映射。注意到：

- 图左边列显示的图片说明了核权值随时间 n 演化的过程。
- 图右半列显示的图片说明了相应的核宽随时间 n 演化的过程。

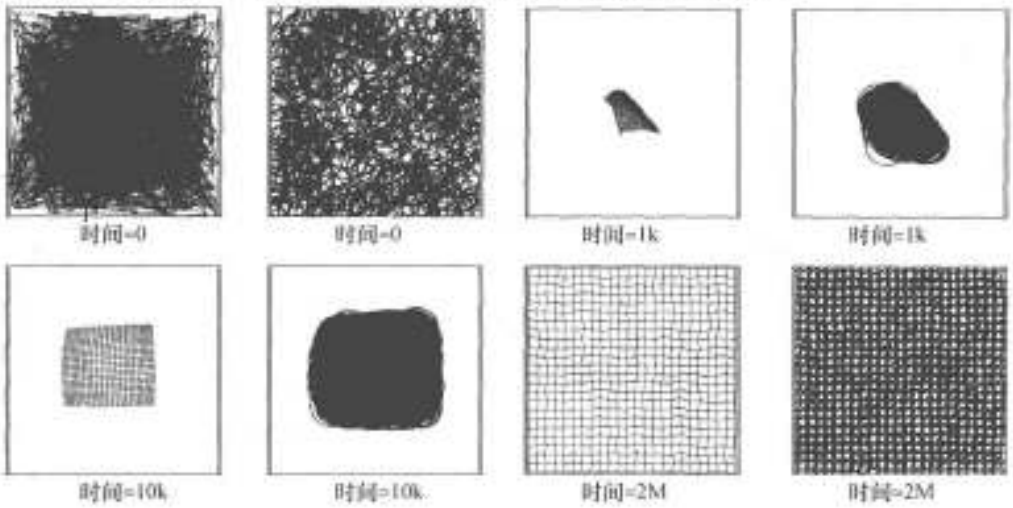


图 9.15　随时间而演化的 24×24 网格，时间值（迭代次数）在每个图片的下方给出。左列：核权值的演化。右列：核宽的演化。图中每一个方块描画了均匀输入分布的结果。在每一个映射下给出的时间表示迭代次数（这个图的复制得到了 Dr. Marc Van Hulle 的许可）

对于大致相同的迭代次数在 24×24 网格上分别通过核 SOM 和传统 SOM 计算的结果，比较图 9.15 左边列的拓扑映射的最终形式以及图 9.8 的映射结果，我们可以作出如下重要的观察：

由核 SOM 计算得到的拓扑映射分布比传统 SOM 计算得到的拓扑映射更加接近于分配给输入数据空间的均匀分布。

相应地，我们可以继续说由核 SOM 计算的放大因子 $m(\mathbf{x})$ 比传统 SOM 的能更好地匹配输入密度 $p_{\mathbf{x}}(\mathbf{x})$；即核 SOM 可以更接近于式(9.27)的理想条件。

9.10　核 SOM 和相对熵之间的关系

我们发现讨论核 SOM(采用不完全 gamma 分布核）和相对熵（Kullback-Leibler divergence，KLD）之间的关系可以提供很多信息。将在下一章讨论细节的 KLD 为评估对真实概率的概率估计质量提供了共识。记真实概率为 $p_{\mathbf{x}}(\mathbf{x})$，其估计记为 $\hat{p}_{\mathbf{x}}(\mathbf{x})$。则我们定义这两个密度之间的 KLD 为：

$$D_{p\|\hat{p}} = \int_{-\infty}^{\infty} p_{\mathbf{x}}(\mathbf{x}) \log\left(\frac{p_{\mathbf{x}}(\mathbf{x})}{\hat{p}_{\mathbf{x}}(\mathbf{x})}\right) d\mathbf{x} \tag{9.65}$$

这里我们采用了信息论中常用的术语。如此定义的 KLD 总是非负的，当且仅当 $\hat{p}_{\mathbf{x}}(\mathbf{x})$ 和 $p_{\mathbf{x}}(\mathbf{x})$ 完全匹配时其值为 0。

对于当前的讨论，假设密度估计被表达为具有相等混频的高斯密度函数的混合，如下所示：

$$\hat{p}_{\mathbf{X}}(\mathbf{x}|\mathbf{w}_i,\sigma_i)=\frac{1}{l}\sum_{i=1}^{l}\frac{1}{(2\pi)^{m/2}\sigma_i^m}\exp\left(-\frac{1}{2\sigma_i^2}\|\mathbf{x}-\mathbf{w}_i\|^2\right) \tag{9.66}$$

这是以权值向量 $\mathbf{w}_i$ 和宽 σ_i，$i=1,2,\cdots,l$ 为条件的。最优密度估计 $p_{\mathbf{X}}(\mathbf{x})$ 是通过最小化它和密度估计 $\hat{p}_{\mathbf{X}}(\mathbf{x}|\mathbf{w}_i,\sigma_i)$ 之间的 KLD 来获得的。实际上，最优密度函数 $p_{\mathbf{X}}(\mathbf{x})$ 被看成是真实密度。作为感兴趣问题的最优化，我们需要对式(9.66)的 KLD 关于可调整参数 $\mathbf{w}_i$ 和 σ_i 微分。最后，得到如下的对 $\mathbf{w}_i$ 的偏导数对：

$$\begin{aligned}\frac{\partial}{\partial\mathbf{w}_i}(D_{p_{\mathbf{X}}\|\hat{p}_{\mathbf{X}}})&=\frac{\partial}{\partial\mathbf{w}_i}\int_{-\infty}^{\infty}p_{\mathbf{X}}(\mathbf{x})\log\left(\frac{p_{\mathbf{X}}(\mathbf{x})}{\hat{p}_{\mathbf{X}}(\mathbf{x}|\mathbf{w}_i,\sigma_i)}\right)\mathrm{d}\mathbf{x}\\&=\int_{-\infty}^{\infty}\frac{\partial}{\partial\mathbf{w}_i}(p_{\mathbf{X}}(\mathbf{x})\log\,p_{\mathbf{X}}(\mathbf{x})-p_{\mathbf{X}}(\mathbf{x})\log\,\hat{p}_{\mathbf{X}}(\mathbf{x}|\mathbf{w}_i,\sigma_i))\mathrm{d}\mathbf{x}\\&=-\int_{-\infty}^{\infty}p_{\mathbf{X}}(\mathbf{x})\frac{\partial}{\partial\mathbf{w}_i}(\log\,\hat{p}_{\mathbf{X}}(\mathbf{x}|\mathbf{w}_i,\sigma_i))\mathrm{d}\mathbf{x}\\&=-\int_{-\infty}^{\infty}p_{\mathbf{X}}(\mathbf{x})\left(\frac{1}{\hat{p}_{\mathbf{X}}(\mathbf{x}|\mathbf{w}_i,\sigma_i)}\frac{\partial}{\partial\mathbf{w}_i}\,\hat{p}_{\mathbf{X}}(\mathbf{x}|\mathbf{w}_i,\sigma_i)\right)\mathrm{d}\mathbf{x}\end{aligned} \tag{9.67}$$

相似地，我们可将对 σ_i 的偏导数表示为：

$$\frac{\partial}{\partial\sigma_i}(D_{p_{\mathbf{X}}\|\hat{p}_{\mathbf{X}}})=-\int_{-\infty}^{\infty}p_{\mathbf{X}}(\mathbf{x})\left(\frac{1}{\hat{p}_{\mathbf{X}}(\mathbf{x}|\mathbf{w}_i,\sigma_i)}\frac{\partial}{\partial\sigma_i}\,\hat{p}_{\mathbf{X}}(\mathbf{x}|\mathbf{w}_i,\sigma_i)\right)\mathrm{d}\mathbf{x} \tag{9.68}$$

令 KLD 的这两个偏导数为 0，然后通过随机逼近理论（Robbins and Monro，1951），我们获得学习规则对（Van Hulle，2002b）

$$\Delta\mathbf{w}_i=\eta_{\mathrm{w}}\,\hat{p}_{\mathbf{X}}(\mathbf{x}|\mathbf{w}_i,\sigma_i)\left(\frac{\mathbf{x}-\mathbf{w}_i}{\sigma_i^2}\right), \tag{9.69}$$

和

$$\Delta\sigma_i=\eta_{\mathrm{w}}\,\hat{p}_{\mathbf{X}}(\mathbf{x}|\mathbf{w}_i,\sigma_i)\cdot\frac{m}{\sigma_i}\left(\frac{\|\mathbf{x}-\mathbf{w}_i\|^2}{m\sigma_i^2}-1\right) \tag{9.70}$$

对 $i=1,2,\cdots,l$；$\hat{p}_{\mathbf{X}}(\mathbf{x}|\mathbf{w}_i,\sigma_i)$ 为由权值向量 $\mathbf{w}_i$ 和宽 σ_i 表示的第 i 个神经元的条件后验密度。

假设我们令条件后验密度为：

$$\hat{p}_{\mathbf{X}}(\mathbf{x}_j|\mathbf{w}_i,\sigma_i)=\delta_{ji},\quad \text{当 } j=1,2,\cdots,l \tag{9.71}$$

其中

$$\delta_{ji}=\begin{cases}1, & \text{当 } j=i\\0, & \text{当 } j\neq i\end{cases}$$

当这一理想条件得到满足时，神经元 i 是在神经元 $j=1,2,\cdots,l$ 中竞争的获胜神经元。因此我们可以将条件后验密度函数 $\hat{p}_{\mathbf{X}}(\mathbf{x}|\mathbf{w}_i,\sigma_i)$ 看成是扮演着核 SOM 构成中引入的拓扑邻域函数 $h_{j,i(\mathbf{x})}$。事实上，令

$$\hat{p}_{\mathbf{X}}(\mathbf{x}|\mathbf{w}_j,\sigma_j)=h_{j,i(x)} \tag{9.72}$$

我们找到了由相对熵推导的更新规则对，即式(9.69)和式(9.70)，它们和 9.9 节中导出的核 SOM 的更新规则对式(9.63)和式(9.64)具有相似的数学形式。

因而我们可以给出如下结论（Van Hulle，2002b）：

在高斯混合模型的假设下，最小化相对熵和最大化定义为不完全 gamma 分布核及基于活跃度的邻域函数上的联合熵等价，后者是核 SOM 的核心。

这一结论在密度估计的背景中尤其重要，此时给定一个数据集 $\{\mathbf{x}_i\}_{i=1}^{N}$，要求对于产生这个

数据的内在固有分布计算一个估计。

9.11　小结和讨论

自组织映射

由 Kohonen(1982) 提出的自组织映射是一个简单但强大的算法，它建立在一维或二维的神经元网格上，用于捕获包含在输入（数据）空间中感兴趣的重要特征。为此，它利用神经元权值向量作为原型提供一个输入数据的结构表示。SOM 算法受到神经生物学的激发，综合第 8 章中讨论的所有自组织的基本机制：竞争、合作、自增强以及结构化信息。因此它可以作为退化但一般的模型，描述在复杂系统中从完全混乱开始最终出现整体有序的现象。换句话说，SOM 具有通过时间进程的演化过程从无序中产生有序的内在能力。

自组织映射也可以被看作向量量化器，从而提供一个导出调整权值向量的更新规则的原理性方法 (Luttrell, 1989b)。后一种方法明确地强调邻域函数作为概率密度函数的作用。

然而应该强调的是，基于使用在式(9.19)中的平均分布 D_1 作为极小化代价函数的后一种方法中，仅当特征映射被很好地排序后才是合理的。在 Erwin 等 (1992b) 中，证明在自适应过程的排序阶段（即在初始是高度混乱的特征映射的拓扑排序期间）自组织映射的学习动态系统不能用一个代价函数的随机梯度下降描述。但就一维网格的情况来说，它可以用一组代价函数描述，对于网络中每个神经元，一个对应的代价函数随随机梯度下降独立地被最小化。

自组织映射的收敛考虑

关于 Kohonen 的 SOM 算法，令人惊奇的是它的实现如此简单，但在一般设置下分析它的性质数学上却如此困难。虽然几个研究者使用相当有力的方法来分析它，但是，他们仅获得有限的应用性结果。在 Cottre Ⅱ 等 (1997) 中给出关于 SOM 算法理论方面的结果的综述。尤其由 Forte and Pagés(1995, 1996)得出的结果引人注目，结果表明就一维网格情况而言，可严格证明：在自组织阶段结束后，SOM 算法“几乎确定”收敛到一个唯一状态。这个重要的结果已被证明对一大类邻域函数成立。然而，在多维情况下尚未得到同样的结论。

神经生物学考虑

既然自组织映射是由大脑皮质映射的思想所激发的，很自然会问是否这种模型可以实际解释皮质映射的形成。Erwin 等 (1995) 进行了这项研究。他们发现自组织映射可以解释猕猴初级视觉皮质中计算映射的形成。这项研究的输入空间的维数是 5 维：二维为视觉空间接收域的位置，剩下的三维代表方向优先、方位选择和视觉优势。皮质表面被分成小块，每块被视为二维网格的计算单元（即人工神经元）。在一定假设下，表明 Hebb 学习导致空间模式的定位和视觉优势与在猕猴中发现的非常相似。

自组织映射的应用

SOM 算法的简单性和强大的可视能力的组合促使该算法在多个大规模应用中得到使用。典型地，算法在非监督模式下训练，使用大量的训练数据样本。特别地，如果数据包含语义相关目标群 (类)(semantically related object groupings)，属于用户定义的类的向量子集被 SOM 通过如下方式映射：算法计算的映射上数据向量的分布提供了原始数据空间固有分布的二维离散逼近。基于这一思想，在 Laaksonen 等 (2004) 和 Laaksonen and Viitaniemi(2007) 中，SOM 被成功用于检测和描述语义目标和目标类之间的存在关系 (ontological relations)，语义目标在一个包含 2618 个图像的视觉数据库中，每个图像属于一个或多个预定义的语义类。在这个研究中使用的存在关系包括如下几点：

- 在一个图像中同时存在从两个或更多目标类而来的目标。

- 视觉相似性的分类。
- 在一个图像中不同目标类型的空间关系。

在另一个不同的应用中，Honkela 等（1995）利用 SOM 算法来研究自然语言单词的语义规则，这里规则是在它们发生的上下文关系中反映的。这一研究的目的是对这些规则的确切形象计算上下文映射。在这一研究完成的实验中，源数据库由 Brothers Grimm 的童话故事的英语翻译组成，对于单词没有任何先验句法或语义分类；单词数总共大约 250 000，词汇的大小超过 7 000 个单词。SOM 算法能够建立上下文映射并且看起来很好地遵守传统的语义分类以及关于单词语义的人类直觉。关于文本内容的分析被扩展到收集上百万的文档；在这一类应用中，网格的神经元个数可以达到上百万，输入数据空间的维数也可能达到上千维（Honkela，2007）。这一类大规模的应用使得自组织映射成为强有力的工具。

核 SOM

在本章的后面部分，我们描述了 Van Hulle(2002b)的核 SOM 算法，这一算法的主要目的是提供改进的拓扑映射和逼近分布能力。核 SOM 的一个出众的特征是其推导是从构造一个熵目标函数开始的。更重要的是，核 SOM 是在线的基于随机梯度的算法。

比较本章中学习的两个自组织映射，我们可以说对于神经元网格中的权值向量标准 SOM 和核 SOM 具有相似的更新规则。而且，他们在同一方向上对权值更新，但采用不同的学习率参数。和标准 SOM 不同，核 SOM 具有对网格中每个神经元 i 自动调整核宽 σ_i 的内在能力，从而最大化核（神经元）输出的联合熵。

然而，核 SOM 需要对两个学习率参数 η_w 和 η_σ 进行仔细的调整以保证权值和宽的更新不发生爆炸性的增长。如果当核宽的方差 σ_i^2 的逆比学习率参数 η_w 和 η_σ 大时就会发生爆炸性增长。这一不希望的行为是由于这样的事实：在式(9.56)和式(9.59)的更新式中，学习率参数 η_w 和 η_σ 分别被 σ_i^2 和 σ_i 除。为了避免 $\mathbf{w}_i$ 和 σ_i 的爆炸性增长的可能性，我们可以将 σ_i^2 用和 $\sigma_i^2+\alpha$ 来代替，这里 σ 是预先给定的小常数。

注释和参考文献

1. 存在其他类型的没有胜利者的竞争学习，如在 Heskes(2001) 和 Van Hulle(2005) 中讨论的那样。
2. 图 9.1 的两个特征映射模型是由 von der Malsburg(1973) 的自组织的先驱性研究所激发，Malsburg 注意到视觉皮质的模型不能整体地被基因预先确定；相反涉及突触学习的自组织过程可能导致特征敏感的皮质细胞的局部排序，但是在 von der Malsburg 的模型中不能取得全局拓扑序，因为模型使用固定的（很小的）邻域，von der Malsburg 的计算机仿真也许是第一次展示自组织。
3. Amari(1980) 在某种程度上放松对后突触神经元的突触权值的限制。Amari 给出的数学分析阐明了由自组织形成的皮质映射的动态稳定性。
4. Grossberg(1969) 在神经网络文献中第一次引入式(9.3)描述的竞争学习规则。
5. 在 Kohonen(1982) 导出的 SOM 算法的原始形式中，拓扑邻域假定为有固定的范围。令 $d_{j,i}$ 表示在邻域函数内获胜神经元 i 和兴奋神经元 j 的侧向距离。一维网格情形的拓扑邻域定义为：

$$h_{j,i}=\begin{cases}1, & -K\leqslant d_{j,i}\leqslant K\\ 0, & \text{否则}\end{cases} \tag{A}$$

其中 $2K$ 为兴奋神经元一维邻域的总长度。与神经生物学考虑相反，式（A）描述的模型意味着在拓扑邻域内所有神经元以相同的速度点火，且这些神经元内部的相互作用与它们到获胜神经元 i 的侧向距离无关。

6. Erwin 等(1992b) 表明当 SOM 算法利用非凸的邻域函数时会出现亚稳定状态，它表示在特征映射设置中的拓扑缺陷。一个宽的凸邻域函数，如宽高斯函数，形成拓扑排序的时间比非凸邻域函数所花的时间短，这是因为没有亚稳定状态。
7. 在第 5 章的注释中指出在通信和信息论的文献中，提出了著名的标量量化的早期方法，即 Lloyd 算法。这个算法首先由 Lloyd 在 Bell 实验室 1957 年未发表的报告中描述（Lloyd，1957），很久以后才发表（Lloyd，

1982)。Lloyd 算法有时也称为“最大量化器”。用于向量量化的广义 Lloyd 算法（generalized Lloyd algorithm，GLA）是 Lloyd 算法的直接推广。广义 Lloyd 算法在 McQueen(1967) 将其作为如第 5 章讨论的统计聚类的工具之后有时称为 k-均值算法。在前面的这一章中我们确实指出 k-均值算法以和期望最大（EM）算法相似的方式进行；这两者之间基本的区别是 k-均值算法的目标函数（和 GLA 相似）被最小化，而 EM 算法的目标函数被最大化。EM 算法在第 11 章中讨论。Lloyd 算法及广义 Lloyd 算法的历史评述可参看 Gersho and Gray(1992)。

8. Kohonen(1993) 给出的实验结果表明，SOM 算法的批量方式比它的在线方式快。但是使用批量方式时 SOM 算法失去自适应能力。
9. 自组织映射的拓扑性质可由不同方法定量评价。一种这样的定量度量称为*地形图产品*（topographic product），它在 Bauer and Pawelzik(1992) 中描述，它可用于比较属于不同维数的不同特征映射的真实行为。但是只有当网格维数和输入空间维数匹配时这种度量才是可量化的。
10. SOM 算法无能力提供输入数据的固有分布的可信表示，这一点促使对算法的修正和能真实表示输入的新自组织算法的发展。

    在文献中有两类 SOM 算法修正的报道。

    （ⅰ）*修改竞争过程*。DeSieno(1988) 在网格中用记忆形式跟踪单个神经元的累计激活量。具体地，添加“良心”机制影响 SOM 算法的竞争过程。这样做使得每个神经元不管它在网格中的位置如何都有机会以接近于理想值 $1/l$ 的概率获胜，其中 l 为总的神经元数。习题 9.7 给出具有良心机制的 SOM 算法的描述。

    （ⅱ）*修改自适应过程*。在这第二种方法中，对用于调整邻域函数内每个神经元权值向量的更新规则进行修改，以控制特征映射的放大性质。在 Bauer 等（1996）中，表明通过对更新规则添加可调步长参数，可以为特征映射提供输入数据的可信表示。Lin 等（1997）遵循相似的途径引入 SOM 算法的两种修改：

    - 修改更新规则，抽取输入向量 x 和问题中神经元 j 的权值向量 $\mathbf{w}_j$ 的直接依赖性。
    - 利用为可分输入分布特别设计的等变化（equivariant）剖分替代 Voronoi 剖分。

    这第二种修改使得 SOM 算法能进行盲源分离。（盲源分离在第 10 章详细讨论。）

    这里所提到的修改建立在标准 SOM 算法的各种形式上。Linsker(1989b) 采用一种完全不同的方法。具体地，利用最大化输出信号和带加性噪声的输入信号之间的互信息的方法，导出用于地形图映射形成的全局学习规则（植根于香农信息论的互信息的定义在第 10 章讨论）。Linsker 的模型产生与输入分布精确匹配的神经元分布。利用信息论的方法以自组织方式处理地形图映射形成也在 Van Hulle (1996，1997) 中有所讨论。
11. 在 Van Hulle(2002) 中对式(9.52)右端第二项的忽视是基于下面的讨论：
    - 对高斯分布输入向量 $\mathbf{x}$ 所获得的期望值 $\|\mathbf{x}-\mathbf{w}_i\|^2$ 在式(9.53)中定义。
    - 在 m-维放射状对称的高斯分布中，分布可以通过取 m 个样本来建立，每个样本对应于一个输入维数。则在具有相同半径的一维高斯分布中，当权值更新量 Δw_{ij} 小（这假定了使用小的学习率参数 η_w）且更新是对每个输入维数分别（即以随机顺序）更新时，可以忽略式(9.52)的第二项。

习题

SOM 算法

9.1 函数 $g(y_j)$ 表示响应 y_j 的非线性函数，它如同在式(9.9)中那样用于 SOM 算法。如果 $g(y_j)$ 的 Taylor 展开的常数项不为零，讨论这会产生什么结果？

9.2 假设 $\pi(\mathbf{v})$ 为图 9.6 模型的噪声 $\mathbf{v}$ 的光滑函数，利用式(9.19)的失真度量的 Taylor 展开，确定噪声模型 $\pi(\mathbf{v})$ 导致的曲率项。

9.3 有时说 SOM 算法保持输入空间中存在的拓扑关系。严格地说，这种性质只有输入空间的维数与神经元网格的维数相等或再低时才能保证。讨论这个陈述的正确性。

9.4 一般说基于竞争学习的 SOM 算法对硬件故障不具有容错性，但是算法对输入的小的扰动引起输出从获胜神经元跳到相邻的神经元具有容错性。讨论这两个陈述的含义。

9.5 考虑由式(9.23)表示的 SOM 算法的离散形式所获得的批量方式，表示为：

$$\mathbf{w}_j = \frac{\sum_i \pi_{j,i}\mathbf{x}_i}{\sum_i \pi_{j,i}}, \quad j = 1,2,\cdots,l$$

证明 SOM 算法的这种形式可以表示成和 Nadaraya-Watson 回归估计器相似的形式（Cherkassky and Mulier，1995）；估计器在第 5 章已经讨论过。

学习向量量化

9.6 第 8 章讨论的最大特征滤波器和自组织特征映射的更新规则都利用 Hebb 学习假设的修正。比较这两个修正，说明它们的异同点。

9.7 良心算法是 SOM 算法的修正，它迫使密度匹配是精确的匹配（DeSieno，1988）。在表 P9.7 所总结的良心算法中，每个神经元保存它竞争获胜的次数（即它的突触权值向量在欧几里得距离下成为距离输入向量最近的神经元的次数）。这里使用的概念，就是如果一个神经元获胜太频繁，它“感到有罪”从而退出竞争。

为了研究利用良心算法在密度匹配上产生的改善，考虑利用图 P9.7 画出的线性输入密度训练由 20 个神经元组成的一维网格（即线性排列）。

(a) 利用计算机仿真比较由良心算法和 SOM 算法产生的密度匹配，对 SOM 算法使用 $\eta=0.05$ 而良心算法使用 $B=0.0001$，$C=1.0$ 和 $\eta=0.05$。

(b) 作为这个比较的参考框架，包括输入密度的“精确”匹配。

讨论你的计算机仿真结果。

表 P9.7　良心算法小结

1. 寻找和输入向量 $\mathbf{x}$ 最近的突触权值向量 $\mathbf{w}_i$：

$$\|\mathbf{x}-\mathbf{w}_i\| = \min_j \|\mathbf{x}-\mathbf{w}_j\|, \quad j=1, 2, \cdots, N$$

2. 保持一轮神经元 j 竞争获胜的总时间部分 p_j：

$$p_j^{\text{new}} = p_j^{\text{old}} + B(y_j - p_j^{\text{old}})$$

其中 B 是小的正数，且

$$y_j = \begin{cases} 1, & \text{如果神经元 } j \text{ 是获胜神经元} \\ 0, & \text{否则} \end{cases}$$

在算法开始时，p_j 初始化为零

3. 利用良心机制

$$\|\mathbf{x}-\mathbf{w}_i\| = \min_j(\|\mathbf{x}-\mathbf{w}_j\| - b_j)$$

寻找新的获胜神经元，其中 b_j 是为了修改竞争而引入的偏置项；它定义为

$$b_j = C\left(\frac{1}{N} - p_j\right)$$

其中 C 为偏置因子而 N 为网络中神经元的总数。

4. 更新获胜神经元的突触权值向量：

$$\mathbf{w}_i^{\text{new}} = \mathbf{w}_i^{\text{old}} + \eta(\mathbf{x} - \mathbf{w}_i^{\text{old}})$$

其中 η 为通常在 SOM 算法中使用的学习率参数。

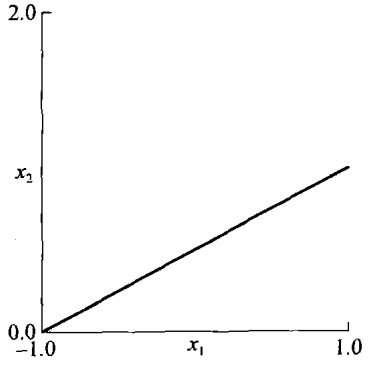


图 P9.7　习题 9.7 的图

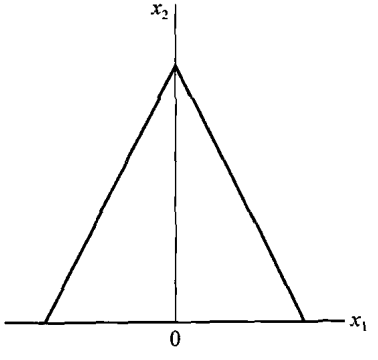


图 P9.11　习题 9.11 的图

核 SOM

9.8　利用作用于式(9.37)的式(9.40)的变换公式，推导式(9.41)的概率密度函数。

9.9　这一习题包括两部分，用于解决推导属于核 SOM 算法的多个等式的问题：

(a) 随机变量 X 的不完全 gamma 分布（样本值为 X）由下式定义（Abramowitz and Stegun，1965，p. 260）：

$$P_X(x|\alpha) = \frac{1}{\Gamma(\alpha)}\int_0^x t^{\alpha-1}\exp(-t)\mathrm{d}t$$

这里 $\Gamma(\alpha)$是 gamma 函数。相应地定义不完全 gamma 分布的补为：

$$\Gamma(d,x) = \int_x^\infty t^{\alpha-1}\exp(-t)\mathrm{d}t$$

利用这两个公式推导式(9.43)定义的随机变量 R 的累积分布函数。

(b) 利用不完全 gamma 分布作为平均神经元输出 $\overline{y}_i$ 的定义，对偏导数 $\partial\overline{y}_i(r)/\partial r$ 推导式(9.51)。

9.10　对核 SOM 算法的权值向量开发式(9.55)的近似更新公式时，我们证明了对式(9.52)的第二项的忽略。然而，对于核宽 σ_i 推导式(9.58)的更新公式时没有做任何近似。验证后一个选择。

计算机实验

9.11　在这个试验中我们用计算机仿真研究 SOM 算法应用于具有二维输入的一维网格。网格由 65 个神经元组成。输入由图 P9.11 所示的三角形内均匀分布的随机点构成。计算由 SOM 算法在 0,20,100,1000，10 000 和 25 000 次迭代后产生的映射。

9.12　考虑一个用三维输入分布训练的二维神经元网格，网格由 10×10 神经元构成。

(a) 在由下式定义的小区域内输入是均匀分布的。

$$\{(0 < x_1 < 1),(0 < x_2 < 1),(0 < x_3 < 0.2)\}$$

利用 SOM 算法计算输入空间在 50，1 000 和 10 000 次算法迭代后的二维投影。

(b) 当输入在如下定义的一个更大的区域内均匀分布时重复你的计算。

$$\{(0 < x_1 < 1),(0 < x_2 < 1),(0 < x_3 < 0.4)\}$$

(c) 当输入在如下定义的立方体内均匀分布时再一次重复你的计算。

$$\{(0 < x_1 < 1),(0 < x_2 < 1),(0 < x_3 < 1)\}$$

讨论你的计算机仿真结果的含义。

9.13　在 SOM 算法应用中经常出现的问题是不能形成拓扑排序而产生“折叠”映射。当允许邻域体积衰减太快时就会发生这个问题。折叠映射的产生可以看作拓扑排序过程形成某种形式的“局部最小”。

为了研究这个现象，考虑一个 10×20 神经元的二维网格，用在正方形 $\{(-1 < x_1 < +1),(-1 < x_2 < +1)\}$ 内均匀分布的二维输入训练。计算由 SOM 算法产生的映射，允许获胜神经元周围的邻域函数比正常使用的衰减快得多。你可能需要重复几次试验才能看到排序过程的失败。

9.14　SOM 算法的拓扑排序性质可以用于形成高维输入空间的一种抽象的二维表示形式。为了研究这种表示形式，考虑由 10×10 神经元组成的二维网格，它的训练输入空间由 8 维空间的 4 个高斯云𝒞1，𝒞12，𝒞13 和𝒞14 构成。所有云具有单位方差但其中心不同。它们的中心位置分别为 (0,0,0,…,0)，(4,0,0,…,0)，(4,4,0,…,0) 和(0,4,0,…,0)。计算由 SOM 算法产生的映射，在映射中每个神经元的类别和在该神经元周围输入点中具有最多输入点的类别相同。

9.15　表 P9.15 给出重正规化 SOM 算法的小结；在第 9.3 节给出了算法的简要描述。比较常规的和重正规化的 SOM 算法，注意以下两个问题：

1. 算法实现所涉及的编码复杂性。
2. 训练花费的计算机时间。

利用从一个正方形内的均匀分布中抽取的数据，且按照下列两个网络配置来说明这两种算法的比较：

(a) 257 个神经元的一维网格。

(b) 2 094 个神经元的一维网格。

在这两种情形都以 2 个编码向量开始。

表 P9.15 重正规化训练算法小结（一维形式）

1. *初始化*。置编码向量的数目为一小整数（例如，为简单起见使用 2 或对所求问题更具代表性的其他数目）。从训练集中随机选择相应数目的训练向量初始化它们的位置。

2. *选择一个输入向量*。从训练集中随机选择一个输入向量。

3. *输入向量编码*。确定获胜编码向量（即获胜神经元的突触权值向量）。为了做到这一点，在需要时使用“最近邻”或“最小失真”编码方法。

4. *码书更新*。执行通常的“获胜者和它的拓扑邻域”更新。你会发现保持学习率参数 η 固定（如 0.125）就足够了。例如更新获胜神经元使用 η 而它的最近邻使用 $\eta/2$。

5. *码书分裂*①。继续码书更新（第 4 步），每次使用随机训练集中挑选的新输入向量直到码书更新的次数是码字向量数目的 10～30 倍。这时码书大概已经稳定，应该进行码书分裂。为做到这一点你既可以采用你所有的码字向量的 Peano 串，且对它们的位置进行插值以产生对 Peano 串的更小粒度的逼近；也可以简单向每两个已有的码字向量连线添加另外码字向量。

6. *训练完成*。继续进行码书更新和码书分裂直到码字向量总数达到某一预定值（如 100），这时整个训练结束。

①码书分裂近似在每一回合时加倍码字向量的数目，所以达到任何预定的码字数目无需花费许多的回合。

9.16 考虑图 P9.16 所示的信号空间图对应的 M *行脉冲幅度调制*（M-level pulse-amplitude modulation，PAM），$M=8$。信号点对应于 Gray 编码数据块。每个信号点由具有合适幅度尺度的矩形冲击信号表示：

$$p(t) = \pm\frac{7}{2}, \pm\frac{5}{2}, \pm\frac{3}{2}, \pm\frac{1}{2}, \quad 0 \leqslant t \leqslant T$$

其中 T 为信号区间。在接收器输入端，对具有变化的信噪比（signal-to-noise ratio，SNR）的传输信号添加零均值的高斯噪声。SNR 定义为传输信号能量平均和噪声能量平均的比值。

(a) 利用随机二值序列作为发送器输入，产生表示 SNR=10，20，30 分贝的接收信号数据。

(b) 对这些 SNR，建立自组织特征映射。你可使用的典型值为：

- 对接受信号以 8 倍信号率采样获得的 8 个元素构成输入向量（即每个信号区间 8 个样本）。假设不知道时间信息。
- 64 个神经元的一维网格（即输入向量大小的 8 倍）。

(c) 对三个 SNR 显示特征映射，由此表示 SOM 算法的拓扑排序性质。

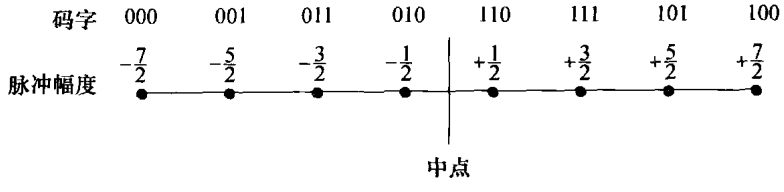


图 P9.16 习题 9.16 的图

第10章
Neural Networks and Learning Machines, 3E

信息论学习模型

本章组织

本章的主题是学习以一种或多种根植于信息论的方式构成非监督学习模型。

本章的组织如下：

10.1 节介绍信息论的引导素材以及其对于神经元处理的深刻影响。

10.2 到 10.6 节回顾香农信息论的基本概念。从 10.2 节的熵的概念开始，接着在 10.3 节中介绍最大熵原则。10.4 节讨论在连续随机变量对之间的互信息概念并检测其关联性。相对熵的相关概念提供了对于一对不同的概率密度函数之间相匹配的程度的度量，这在 10.5 节中讨论。10.6 节通过描述系词来完成整个回顾，已经发现了几十年的系词是一个有用的概念但很大程度上被忽视了。

10.7 节讨论作为非监督学习目标函数的互信息规则，从而为讨论下面的五个原则及其应用铺垫，这在 10.8 节到 10.12 节阐述：

- 最大互信息（Infomax）原则
- 最小冗余原则
- 处理空间相干特征的 Imax 原则
- 处理空间不相干特征的 Imin 原则
- 独立分量分析（ICA）原则

10.13 节讨论稀疏问题，这是自然图像的内在特点；这一节也通过描述其和稀疏的关系为 ICA 提供了动机。

10.14 节到 10.17 节描述不同的 ICA 算法，强调其实际优点和局限性：

- 自然梯度学习算法
- 最大似然估计
- 最大熵学习算法
- 通过最大化所熟知的负熵非高斯准则的 FastICA

10.18 节讨论称为相关 ICA 的新概念，这是建立在对系词的运用之上的。

10.19 节介绍另一个新的且吸引人的方法叫做信息瓶颈（IB）方法，这是建立在香农信息论的另一个概念之上的：速率失真理论。IB 方法为数据的最优流形表达的讲述铺平了道路，这将在 10.20 节讨论，紧接着在 10.21 节给出一个计算机实验。

本章通过 10.22 节的总结和结论来结束。

10.1 引言

香农在 1948 年发表的经典论文中，为信息论奠定了基础。香农在信息论方面的开创性工作[1]和其他的研究工作者对它的补充，是对电子工程师设计高效可靠通信系统的需求的直接回应。无论它的实际起源是什么，如我们今天所知道的信息论正是关于通信过程本质的深刻数学理论。这个理论提供一个对根本问题研究的总体框架，例如，信息表示的效率以及通信信道可靠信息传输的极限问题。而且该理论包括很多有力的定理用以计算最佳表示和信号所携带信息的传输的理想界限。这些界限非常重要，因为它们为提高信息处理系统的设计提供了标准。

本章的主要目的是讨论以一种原则性方式导致自组织的信息论模型。在这个背景下，特别值得注意的模型是由 Linsker(1988a，b）提出的最大互信息原则（maximum mutual informa-

tion principle)。该原则表明：

多层神经网络的突触连接以这样一种方式进行：在网络的每个处理阶段，当进行信号变换时，为保留的信息量达到最大，要遵从一定的约束条件。

利用信息论来解释人们的感知过程并不是什么新的想法[2]。例如，在 Attneave(1954) 的一篇早期论文中提出了关于感知系统的信息论作用：

感知机制的一个主要功能是减少刺激的冗余，以一种比它冲击接收器的形式更经济的方式对信息进行描述或编码。

在 Attneave 的论文背后的主要思想在于认识到为减少冗余对场景数据编码和确认场景中特定特征是相关的。这种重要认识和在 Craik(1943) 描述的关于人脑的观点相关，在该论文中构造了一个外部世界的模型以便结合现实的规则和约束。

10.2　熵

对于一个随机变量 X，它的每一个实现（出现）可看作一个消息。严格地说，如果随机变量 X 的幅度值是连续的，则它带有无穷的信息。但是，从物理和生物的角度来看，我们认识到讨论具有无限精度的幅度度量的信息是没有意义的，这就是说可以把 X 的值一致量化到有限的离散水平。这样我们可以把 X 看成是离散的随机变量，其模型为

$$X = \{x_k \mid k = 0, \pm 1, \cdots, \pm K\} \tag{10.1}$$

其中 x_k 是一个离散的数值且 $(2K+1)$ 是总的离散水平。假设离散水平之间的间隔 δ_x 非常小，能够以足够的精度来描述我们感兴趣的变量。当然我们能够接近连续的极限，只要 $\delta_x \to 0$ 且 K 趋于无穷，在这种情况下就得到连续变量而且求和变成积分。

为完善模型，让事件 $X=x_k$ 以概率

$$p_k = P(X = x_k) \tag{10.2}$$

发生，其中要求

$$0 \leqslant p_k \leqslant 1 \quad \text{和} \quad \sum_{k=-K}^{K} p_k = 1 \tag{10.3}$$

假如事件 $X=x_k$ 发生的概率 $p_k=1$，因此要求对所有 $i \neq k$ 且有 $p_i=0$。在这种情况下，如果事件 $X=x_k$ 发生就没有什么“惊奇”的了，并且不传达任何“信息”，因为我们知道消息必须是什么。在另一种情况下，如果各种离散水平发生的概率不同，概率 p_k 特别小，那么当 X 取值 x_k 而不是具有更高概率 p_i 的离散水平 $x_i (i \neq k)$ 时，这就有更大的“惊奇”和有“信息”了。因此“不确定”、“惊奇”和“信息”是相关的。在 $X=x_k$ 发生之前，有一定的不确定性。在 $X=x_k$ 发生之后，有一定惊奇。在 $X=x_k$ 发生之后，信息量增加了。这里的三个量很显然是一样的，而且信息量与事件发生的概率成反比。

我们定义观察到具有概率 p_k 的事件 $X=x_k$ 后所获得的信息增益量为对数函数

$$I(x_k) = \log\left(\frac{1}{p_k}\right) = -\log p_k \tag{10.4}$$

其中对数函数的底是任意的。当以自然对数为底时，信息的单位是奈特 (nat)，当以 2 为底时，单位是比特 (bit)。在任何情况下以式(10.4)定义的信息量都有以下的性质：

1.　$I(x_k) = 0$，　当 $p_k = 1$ 　　(10.5)

显然，如果我们绝对肯定将发生的事件，则当其发生时就没有获得信息。

2.　$I(x_k) \geqslant 0$，　当 $0 \leqslant p_k \leqslant 1$ 　　(10.6)

也就是说，当事件 $X=x_k$ 发生时，或提供一些信息或不提供信息，但不会导致信息损失。

3. $$I(x_k) > I(x_i), \quad \text{当 } p_k < p_i \tag{10.7}$$

也就是说，小概率事件发生时携带的信息量比大概率事件发生时携带的信息量多。

信息量 $I(x_k)$也是一个具有概率 p_k 的离散随机变量。$I(x_k)$在全部 $2K+1$ 个离散数值上的平均值定义为：

$$H(X) = \mathbb{E}[I(x_k)] = \sum_{k=-K}^{K} p_k I(x_k) = -\sum_{k=-K}^{K} p_k \log\, p_k \tag{10.8}$$

量 $H(X)$叫做一个可取有限离散值的随机变量 X 的熵；之所以称为熵是因为式(10.8)给出的定义与统计热力学中的熵非常相似[3]。熵 $H(X)$表示每一个消息所携带的信息的平均量。注意在 $H(X)$中 X 不是 $H(X)$的变量，而是一个随机变量的标记。同时注意到在式(10.8)中我们取 $0 \log 0$ 为 0。

熵 $H(X)$被限定如下：

$$0 \leqslant H(X) \leqslant \log(2K+1) \tag{10.9}$$

其中 $(2K+1)$ 是总的离散水平的数目。进一步，我们做如下说明：

1. $H(X)=0$ 当且仅当对于某一个 k 概率 $p_k=1$ 时，而集合中其他的概率为 0；熵的这个下界不对应不确定性。
2. $H(X)=\log(2K+1)$当且仅当对所有的 k，$p_k=1/(2K+1)$(即所有的离散值的概率相等)；这个上界对应最大不确定性。

连续随机变量的微分熵

信息论概念的讨论现在只涉及它们的幅度离散的随机变量总体。现在我们将这些概念中的一些扩展到连续随机变量。

假设连续随机变量 X 的概率密度函数是 $p_X(x)$，与离散随机变量的熵的定义类似，我们定义如下：

$$h(X) = -\int_{-\infty}^{\infty} p_X(x) \log\, p_X(x) \mathrm{d}x = -\mathbb{E}[\log\, p_X(x)] \tag{10.10}$$

将 $h(X)$定义为 X 的微分熵 (differential entropy)，与一般的或绝对熵相区别。

我们对使用式(10.10)的合理性可以解释如下。开始将连续随机变量 X 看成离散随机变量的极限形式，设 $x_k = k\delta x$，其中 $k = 0, \pm 1, \pm 2, \cdots$，且 δx 趋于 0。由定义，连续随机变量 X 取值在 $[x_k, x_k + \delta x]$ 之间的概率为 $p_X(x_k)\delta x$。所以，当 δx 趋于 0 时连续随机变量 X 的普通熵可以写成如下极限的形式：

$$\begin{aligned} H(X) &= -\lim_{\delta x \to 0} \sum_{k=-\infty}^{\infty} p_X(x_k)\delta x \log(p_X(x_k)\delta x) \\ &= -\lim_{\delta x \to 0} \left[\sum_{k=-\infty}^{\infty} p_X(x_k)(\log\, p_X(x_k))\delta x + \log\delta x \sum_{k=-\infty}^{\infty} p_X(x)\delta x \right] \\ &= -\int_{-\infty}^{\infty} p_X(x) \log\, p_X(x) \mathrm{d}x - \lim_{\delta x \to 0} \log\delta x \int_{-\infty}^{\infty} p_X(x) \mathrm{d}x \\ &= h(X) - \lim_{\delta x \to 0} \log\delta x \end{aligned} \tag{10.11}$$

其中最后一行用到了式(10.10)以及在概率密度函数 $p_X(x)$下方的总面积为 1 这个事实。当 δx 趋于 0 时，$-\log\, \delta x$ 趋于无穷大。这意味着连续随机变量的熵是无穷大。直观上，我们也期望这是真的，因为随机变量可以在 $(-\infty, \infty)$ 上任意取值，和随机变量相关联的不确定性是无穷大的。为了避免随着项 $\log\, \delta x$ 所带来的问题，我们采用 $h(X)$作为微分熵，项 $-\log\, \delta x$ 作为参考。而且，由于熵作为一个随机系统处理的信息实体，我们感兴趣的实际上是具有相同参考的两个熵项的差，信息将和相应微分熵项之间的差是一样的。所以我们完全有理由采用在式

(10.11)所定义的项 $h(X)$作为连续随机变量 X 的微分熵。

当有一个由 n 个随机变量 $X_1, X_2, \cdots, X_n$ 组成的随机连续向量 $\mathbf{X}$ 时，我们定义 $\mathbf{X}$ 的微分熵为 n 重积分

$$h(\mathbf{X}) = -\int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x}) \log\, p_{\mathbf{X}}(\mathbf{x}) \mathrm{d}\mathbf{x} = -\mathbb{E}[\log\, p_{\mathbf{X}}(\mathbf{x})] \tag{10.12}$$

其中 $p_X(\mathbf{x})$是 $\mathbf{X}$ 的联合概率密度函数，$\mathbf{x}$ 是 $\mathbf{X}$ 的一个样本。

例 1　均匀分布

考虑在 [0,a] 区间上均匀分布的随机变量 X，其概率密度函数为：

$$p_X(x) = \begin{cases} \dfrac{1}{a}, & 0 \leqslant x \leqslant a \\ 0, & \text{否则} \end{cases} \tag{10.13}$$

X 的微分熵为：

$$h(X) = -\int_0^a \frac{1}{a} \log\left(\frac{1}{a}\right) \mathrm{d}x = \log a$$

当 $a < 1$, $\log a$ 为负，这意味着熵 $h(X)$是负的。因而我们可以说，和离散随机变量的微分熵不同，连续随机变量的微分熵可以假设为负值。

当 $a=1$，微分熵 $h(X)$设为 0。可以说一致分布随机变量在所有随机变量中包含最小量的信息。 ■

微分熵的性质

从式(10.10)给出的微分熵 $h(X)$的定义中容易看出变换不会改变它的值，即

$$h(X + c) = h(X) \tag{10.14}$$

其中 c 为常量。

$h(X)$另一个有用的性质是：

$$h(aX) = h(X) + \log|a| \tag{10.15}$$

其中 a 为比例系数。要证明该式，我们首先知道概率密度函数曲线下方的面积是 1，故

$$p_Y(y) = \frac{1}{|a|} p_Y\left(\frac{y}{a}\right) \tag{10.16}$$

接着应用式(10.10)，可写成

$$\begin{aligned} h(Y) &= -\mathbb{E}[\log\, p_Y(y)] = -\mathbb{E}\left[\log\left(\frac{1}{|a|} p_Y\left(\frac{y}{a}\right)\right)\right] \\ &= -\mathbb{E}\left[\log\, p_Y\left(\frac{y}{a}\right)\right] + \log|a| \end{aligned} \tag{10.17}$$

代入 $Y=aX$ 得到

$$h(aX) = -\int_{-\infty}^{\infty} p_X(x) \log\, p_X(x) \mathrm{d}x + \log|a|$$

由此立刻得出式(10.15)。

式(10.15)用于标量的随机变量，也可以推广用于随机向量 $\mathbf{X}$ 乘以矩阵 $\mathbf{A}$ 的情况如下：

$$h(\mathbf{AX}) = h(\mathbf{X}) + \log|\det(\mathbf{A})| \tag{10.18}$$

其中 $\det(\mathbf{A})$是矩阵 $\mathbf{A}$ 的行列式。

10.3　最大熵原则

假设有一个随机系统，已知一组状态，但不知其概率，而且我们知道这些状态的概率分布的一些限制条件。这些条件或者是已知一定的总体平均值，或者是它们的一些界限。在给定关

于模型的先验知识的条件下，问题是选择一个在某种意义下最佳的概率模型。我们经常发现有无穷多种模型可以满足该条件。应该选择哪个模型呢？

这个基本问题的答案基于 Jaynes(1957) 提出的最大熵原则[4]。最大熵原则可以陈述如下(Jaynes，1957，2003)：

当根据不完整的信息作为依据进行推断时，应该由满足分布限制条件的具有最大熵的概率分布推得。

实际上，熵的概念在概率分布空间定义一种度量，使得具有较高熵的分布比其他的分布具有更大的值。

从上面陈述，很明显“最大熵问题”是一个约束最优化问题。要说明解这个问题的步骤，考虑最大微分熵

$$h(X) = -\int_{-\infty}^{\infty} p_X(x)\log\ p_X(x)\mathrm{d}x$$

对所有随机变量 X 的概率密度函数 $p_X(x)$，并满足以下约束条件：

1. $p_X(x)\geqslant 0$，在 x 的支持集之外等式成立

2. $\int_{-\infty}^{\infty} p_X(x)\mathrm{d}x = 1$

3. $\int_{-\infty}^{\infty} p_X(x)g_i(x)\mathrm{d}x = \alpha_i$，对 $i = 1,2,\cdots,m$

其中 $g_i(x)$是 x 的一部分函数。约束 1 和约束 2 描述概率密度函数的基本属性，约束 3 定义变量 X 的矩，它随函数 $g_i(x)$的表达式不同而发生变化。实际上，约束 3 综合随机变量 X 的可用先验知识。为了解这个约束最优化问题，我们利用第 6 章讨论过的拉格朗日乘子法。具体来说，首先形成拉格朗日函数

$$J(p) = \int_{-\infty}^{\infty}\left[-p_X(x)\log\ p_X(x) + \lambda_0 p_X(x) + \sum_{i=1}^{m}\lambda_i g_i(x)p_X(x)\right]\mathrm{d}x \tag{10.19}$$

其中 $\lambda_0,\lambda_1,\cdots,\lambda_m$ 是拉格朗日乘子。对式(10.19)的被积函数求 $p_X(x)$的微分，并使其为 0，得到

$$-1-\log\ p_X(x) + \lambda_0 + \sum_{i=1}^{m}\lambda_i g_i(x) = 0$$

解此方程得

$$p_X(x) = \exp\left(-1+\lambda_0 + \sum_{i=1}^{m}\lambda_i g_i(x)\right) \tag{10.20}$$

在式(10.20)的拉格朗日乘子根据约束条件 2 和 3 选择。式(10.20)定义这个问题的最大熵分布。

例 2　一维高斯分布

假设我们可用的先验知识为随机变量 X 的均值 μ 和方差 σ^2。根据定义，随机变量 X 的方差由下式给出：

$$\int_{-\infty}^{\infty}(x-\mu)^2 p_X(x)dx = \sigma^2 = \text{常数}$$

将此式与约束条件 3 作比较，看出

$$g_1(x) = (x-\mu)^2$$

和

$$\alpha_1 = \sigma^2$$

所以应用式(10.20)可得

$$p_X(x) = \exp[-1 + \lambda_0 + \lambda_1 (x-\mu)^2]$$

注意如果 $p_X(x)$ 和 $(x-\sigma)^2 p_X(x)$ 对 x 的积分是收敛的，则 λ_1 为负数。将此等式代入约束条件 2 和 3，解出 λ_0 和 λ_1，得到：

$$\lambda_0 = 1 - \log(2\pi\sigma^2)$$

和

$$\lambda_1 = -\frac{1}{2\sigma^2}$$

所以希望的 $p_X(x)$ 的分布形式为

$$p_X(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{10.21}$$

这是一个均值为 μ 和方差为 σ^2 的高斯随机变量 X 的概率密度函数。这样的随机变量的微分熵的最大值为：

$$h(X) = \frac{1}{2}[1 + \log(2\pi\sigma^2)] \tag{10.22}$$

对这个例子我们总结如下：

1. 对于给定的方差 σ^2，在任意的随机变量中高斯随机变量取得微分熵的最大值。也就是说，如果 X 是一个高斯随机变量，Y 是其他具有相同均值和方差的随机变量，则对所有的 Y

$$h(X) \geqslant h(Y)$$

只有当随机变量 Y 也是高斯时等式成立。

2. 高斯随机变量 X 的熵值唯一取决于 X 的方差（即与 X 的均值无关）。 ■

例 3　多维高斯分布

在这第二个例子中，我们想在例 2 的结果基础上，建立计算多维高斯分布的微分熵的计算公式。由于高斯分布的熵与随机变量 X 的均值无关，为简化讨论，我们可以仅讨论具有均值为 0 的随机变量 $\mathbf{X}$。这样 $\mathbf{X}$ 的二阶统计性质由其协方差矩阵 Σ 决定，它为 $\mathbf{X}$ 同自身的外积的期望所定义。这样 $\mathbf{X}$ 的联合概率密度函数由

$$p_{\mathbf{X}}(\mathbf{x}) = \frac{1}{(2\pi)^{m/2}(\det(\boldsymbol{\Sigma}))^{1/2}} \exp\left(-\frac{1}{2}\mathbf{x}^T\boldsymbol{\Sigma}^{-1}\mathbf{x}\right) \tag{10.23}$$

给出（Wilks，1962），其中 $\det(\boldsymbol{\Sigma})$ 是 $\boldsymbol{\Sigma}$ 的行列式。式(10.12)定义 $\mathbf{X}$ 的微分熵。因此将式(10.23)代入式(10.12)，得到

$$h(\mathbf{X}) = \frac{1}{2}[m + m\log(2\pi) + \log|\det(\boldsymbol{\Sigma})|] \tag{10.24}$$

这包括式(10.22)作为其特例。按最大熵原则的观点，我们可以这样说：

对于给定的一个协方差矩阵 $\boldsymbol{\Sigma}$，在所有零均值随机向量可达到的微分熵中，式(10.23)定义的多元高斯分布具有最大的微分熵，此最大微分熵由式(10.24)定义。术语“元”是对随机向量 $\mathbf{X}$ 的分量的另一种称呼。 ■

10.4　互信息

考虑一对连续随机变量 X 和 Y，这两者是相关的。由概率理论，可将 X 和 Y 的联合概率密度表示为：

$$p_{X,Y}(x,y) = p_Y(y|x)p_X(x) \tag{10.25}$$

因此，根据微分熵的定义，有

$$h(X,Y) = h(X) + h(Y|X) \tag{10.26}$$

这里 $h(X,Y)$称为 X 和 Y 的联合微分熵，且 $h(Y|X)$称为给定 X，Y 的条件微分熵。用文字来描述，可以说关于 X 和 Y 的不确定性等于关于 X 的不确定性加上给定 X 时 Y 的不确定性。相似地，可以说关于 X 和 Y 的不确定性等于 Y 的不确定性加上给定 Y 时 X 的不确定性，如下所示：

$$h(X,Y) = h(Y) + h(X|Y) \tag{10.27}$$

下面考虑一个更加结构化的状况，这包含了一个随机神经系统，其中连续随机变量 X 应用到系统的输入，在系统的输出端产生了一个连续随机变量 Y。通过定义，微分熵 $h(X)$是在观察系统输出 Y 之前关于系统输入 X 的不确定性，而条件微分熵 $H(X|Y)$是在观察了系统输出 Y 之后的系统输入 X 的不确定性。其差 $H(X)-H(X|Y)$就是由观察系统输出 Y 所决定的系统输入 X 的不确定性。这一熵差称为系统输入 X 和系统输出 Y 之间的互信息，记为 $I(X;Y)$，因此可以写为：

$$\begin{aligned} I(X;Y) &= h(X) - h(X|Y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} p_{X,Y}(x,y)\log\left(\frac{p_{X,Y}(x,y)}{p_X(x)p_Y(y)}\right)\mathrm{d}x\mathrm{d}y \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \underbrace{p_{X|Y}(x|y)p_Y(y)}_{p_{X,Y}(x,y)}\log\left(\frac{p_{X,Y}(x,y)}{p_Y(y)}\right)\mathrm{d}x\mathrm{d}y \end{aligned} \tag{10.28}$$

对于式(10.28)中第一行到第二行的转变，参看习题 10.2。微分熵是互信息的一个特例，因为有

$$h(X) = I(X;X)$$

式(10.28)中互信息 $I(X;Y)$的公式表示为微分熵 $h(X)$。相应地，互信息 $I(Y;X)$可以表示为微分熵 $h(Y)$

$$I(Y;X) = h(Y) - h(Y|X) \tag{10.29}$$

其中 $h(Y|X)$是给定 X 时 Y 的条件微分熵。互信息 $I(Y;X)$是通过观察系统输入 X 得到的关于系统输出 Y 的不确定性。

两个连续随机变量 X 和 Y 之间的互信息具有三个重要性质：

性质 1　非负性

互信息 $I(X;Y)$总是非负的，即

$$I(X;Y) \geqslant 0 \tag{10.30}$$

这个性质说明，通过观测系统的输出 Y，平均说来我们不可能丢失系统输入 X 的信息。而且，当且仅当输入和输出统计独立时互信息为 0。

性质 2　对称性

这第二个性质说明

$$I(Y;X) = I(X;Y) \tag{10.31}$$

性质 1 和性质 2 可由式(10.28)的定义公式直接得到。

将式(10.26)到式(10.31)综合起来，我们有

$$I(X;Y) = h(X) - h(X|Y) = h(Y) - h(Y|X) = (h(X) + h(Y)) - h(X,Y) \tag{10.32}$$

由此我们可以构造图 10.1(MacKay, 2003)。系统输入 X 的微分熵通过图中第 2 个矩形来表示，系统输出 Y 的微分熵通过图中第 3 个矩形来表示。X 和 Y 之间的互信息表示为图中的阴影区域，通过这两个矩形之间的覆盖来表示。图中也包含了联合熵 $h(X,Y)$的表示以及两个条件熵 $h(X|Y)$和 $h(Y|X)$。

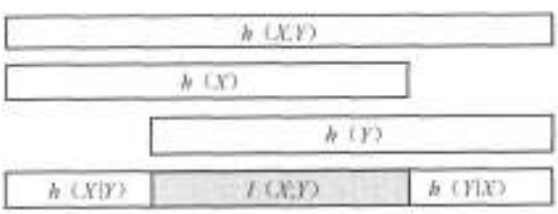


图 10.1　式(10.32)中包含的关系，包括互信息 $I(X;Y)$

性质 3　不变性

在随机变量的可逆变换下互信息是不变的。

考虑可逆变换

$$u = f(x)$$

和

$$v = g(y)$$

其中 x 和 y 是随机变量 X 和 Y 的样本，u 和 v 是变换后的随机变量 U 和 V 的样本值。互信息的不变性说明：

$$I(X;Y) = I(U;V) \tag{10.33}$$

由于从 x 到 u 的变换以及从 y 到 v 的变换都是可逆的，在这两个变换的过程中没有损失信息。从直觉上，这一结果验证了互信息的不变性。

互信息的一般性

在式(10.28)中给出的互信息 $I(X;Y)$ 的定义应用于标量随机变量 X 和 Y。这个定义也易于扩展至随机向量 $\mathbf{X}$ 和 $\mathbf{Y}$，因此可以写成 $I(\mathbf{X};\mathbf{Y})$。具体地，定义互信息 $I(\mathbf{X};\mathbf{Y})$ 为

$$\begin{aligned} I(\mathbf{X};\mathbf{Y}) &= h(\mathbf{X}) - h(\mathbf{X}|\mathbf{Y}) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} p_{\mathbf{X},\mathbf{Y}}(\mathbf{x},\mathbf{y})\log\left(\frac{p_{\mathbf{X},\mathbf{Y}}(\mathbf{x},\mathbf{y})}{p_{\mathbf{X}}(\mathbf{x})p_{\mathbf{Y}}(\mathbf{y})}\right)\mathrm{d}\mathbf{x}\mathrm{d}\mathbf{y} \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \underbrace{p_{\mathbf{X}|\mathbf{Y}}(\mathbf{x}|\mathbf{y})p_{\mathbf{Y}}(\mathbf{y})}_{p_{\mathbf{X},\mathbf{Y}}(\mathbf{x},\mathbf{y})}\log\left(\frac{p_{\mathbf{X}|\mathbf{Y}}(\mathbf{x}|\mathbf{y})}{p_{\mathbf{Y}}(\mathbf{y})}\right)\mathrm{d}\mathbf{x}\mathrm{d}\mathbf{y} \end{aligned} \tag{10.34}$$

互信息 $I(\mathbf{X};\mathbf{Y})$ 同样具有与式(10.30)和式(10.31)的关于标量随机变量性质平行的性质，这直观上是满足的。

10.5　相对熵

在式(10.34)中定义的互信息 $I(\mathbf{X};\mathbf{Y})$，作用于随机神经系统，其输入和输出相应地记为多维向量 X 和 Y。下面考虑同样的系统，但这一次我们有两个不同的概率密度函数 $p_{\mathbf{X}}(\mathbf{x})$ 和 $g_{\mathbf{X}}(\mathbf{x})$ 作为输入向量 $\mathbf{X}$ 的潜在的可能说明。然后我们可以定义在 $p_{\mathbf{X}}(\mathbf{x})$ 和 $g_{\mathbf{X}}(\mathbf{x})$ 之间的相对熵(KLD) 如下 (Kullback，1968；Shore and Johnson，1980)：

$$D_{p\|g} = \int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x})\log\left(\frac{p_{\mathbf{X}}(\mathbf{x})}{g_{\mathbf{X}}(\mathbf{x})}\right)\mathrm{d}\mathbf{x} = \mathbb{E}\left[\log\left(\frac{p_{\mathbf{X}}(\mathbf{x})}{g_{\mathbf{X}}(\mathbf{x})}\right)\right] \tag{10.35}$$

其中的期望是对概率密度函数 $p_{\mathbf{X}}(\mathbf{x})$。

KLD 有两个其自身特有的性质：

性质 1　非负性

这个性质表明

$$D_{p\|g} \geqslant 0 \tag{10.36}$$

对于 $g_{\mathbf{X}}(\mathbf{x}) = p_{\mathbf{X}}(\mathbf{x})$ 的特例，两个分布完全重合，而 KLD 正好为零。

性质 2　不变性

考虑可逆变换

$$\mathbf{y} = \mathbf{f}(\mathbf{x})$$

其中 $\mathbf{x}$ 和 $\mathbf{y}$ 是相应的随机变量 $\mathbf{X}$ 和 $\mathbf{Y}$ 的样本。相应地，KLD 在这个变换下是不变的，这意味着

$$D_{p_{\mathbf{X}}\|g_{\mathbf{X}}} = D_{p_{\mathbf{Y}}\|g_{\mathbf{Y}}}$$

$D_{p_{\mathbf{X}}\|g_{\mathbf{X}}}$ 是相应于输入向量 $\mathbf{X}$ 的 KLD，$D_{p_{\mathbf{Y}}\|g_{\mathbf{Y}}}$ 是相应于变换后输出向量 $\mathbf{Y}$ 的 KLD。

相对熵和互信息之间的关系

一对向量 $\mathbf{X}$ 和 $\mathbf{Y}$ 之间的互信息 $I(\mathbf{X};\mathbf{Y})$用相对熵有一个有趣的解释。为了表述的方便重写式(10.34)的第二行，有

$$I(\mathbf{X};\mathbf{Y}) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} p_{\mathbf{X},\mathbf{Y}}(\mathbf{x},\mathbf{y})\log\left(\frac{p_{\mathbf{X},\mathbf{Y}}(\mathbf{x},\mathbf{y})}{p_{\mathbf{X}}(\mathbf{x})p_{\mathbf{Y}}(\mathbf{y})}\right)\mathrm{d}\mathbf{x}\mathrm{d}\mathbf{y}$$

然后将这一式子和式(10.35)作比较。我们立即推得

$$I(\mathbf{X};\mathbf{Y}) = D_{p_{\mathbf{x},\mathbf{y}}\|p_{\mathbf{x}}p_{\mathbf{y}}} \tag{10.37}$$

总的来说，$\mathbf{X}$ 和 $\mathbf{Y}$ 之间的互信息 $I(\mathbf{X};\mathbf{Y})$等于联合概率密度函数 $p_{\mathbf{X},\mathbf{Y}}(\mathbf{x},\mathbf{y})$以及概率密度函数 $p_{\mathbf{X}}(\mathbf{x})$和 $p_{\mathbf{Y}}(\mathbf{y})$的乘积的相对熵。

相对熵的熵解释

式(10.37)描述的后一结果的特例是 $m\times 1$ 的随机向量 $\mathbf{X}$ 的概率密度函数 $p_{\mathbf{X}}(\mathbf{x})$和它的 m 个边缘概率密度函数的积的相对熵。令 $\tilde{p}_{X_i}(x_i)$ 记分量 X_i 的第 i 个边缘概率密度函数，定义为：

$$\tilde{p}_{X_i}(x_i) = \int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x})\mathrm{d}\mathbf{x}^{(i)}, i = 1,2,\cdots,m \tag{10.38}$$

其中 $\mathbf{x}^{(i)}$是一个从向量 x 中除去第 i 个元素后的 $(m-1)\times 1$ 向量。定义级乘分布为

$$\tilde{p}_{\mathbf{X}}(\mathbf{x}) = \prod_{i=1}^{m}\tilde{p}_{\mathbf{X}_i}(x_i)$$

这表示一个随机变量的独立集合。这个集合中的第 i 个分量 $\mathbf{X}_i$ 的分布是和原始随机向量 $\mathbf{X}$ 的第 i 个边缘分布相同的。通常概率分布 $p_{\mathbf{X}}(\mathbf{x})$和级乘配对 $\tilde{p}_{\mathbf{X}}(\mathbf{x})$之间的 KLD 定义为

$$\begin{aligned} D_{p_{\mathbf{x}}\|\tilde{p}_{\mathbf{x}}} &= \int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x})\log\left\{\frac{p_{\mathbf{X}}(\mathbf{x})}{\prod_{i=1}^{m}\tilde{p}_{\mathbf{X}i}(x_i)}\right\}\mathrm{d}\mathbf{x} \\ &= \int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x})\log\ p_{\mathbf{X}}(\mathbf{x})\mathrm{d}\mathbf{x} - \sum_{i=1}^{m}\int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x})\log\ \tilde{p}_{X_i}(x_i)\mathrm{d}\mathbf{x} \end{aligned} \tag{10.39}$$

根据定义，式(10.39)第二行右边第一个积分等于 $-h(\mathbf{X})$，其中 $h(\mathbf{X})$是 $\mathbf{X}$ 的微分熵。为了处理等式右端第二项，我们首先注意到微分 $\mathrm{d}\mathbf{x}$ 可以表示为

$$\mathrm{d}\mathbf{x} = \mathrm{d}\mathbf{x}^{(i)}\mathrm{d}x_i$$

因此，可以写

$$\int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x})\log\ \tilde{p}_{X_i}(x_i)\mathrm{d}\mathbf{x} = \int_{-\infty}^{\infty}\log\ \tilde{p}_{X_i}(x_i)\int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x})\mathrm{d}\mathbf{x}^{(i)}\mathrm{d}x_i \tag{10.40}$$

其中右端内层积分是对 $(m-1)\times 1$ 向量 $\mathbf{x}^{(i)}$积分，而外层积分是对标量 x_i 积分。但从(10.38)我们发现内层积分实际上等于边缘概率密度函数 $\tilde{p}_{X_i}(x_i)$。由此可以将式(10.40)重写为等价形式：

$$\int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x})\log\ \tilde{p}_{X_i}(x_i)\mathrm{d}\mathbf{x} = \int_{-\infty}^{\infty}\tilde{p}_{X_i}(x_i)\log\ \tilde{p}_{X_i}(x_i)\mathrm{d}x_i = -\tilde{h}(X_i), \qquad i = 1,2,\cdots,m \tag{10.41}$$

其中 $\tilde{h}(X_i)$是第 i 个边缘熵（即边缘概率密度函数 $\tilde{p}_{X_i}(X_i)$的微分熵）。最后将式(10.41)代入式(10.39)，并注意式(10.39)中的第一个积分为 $-h(\mathbf{X})$，我们将式(10.39)的相对熵简化为

$$D_{p_{\mathbf{x}}\|\tilde{p}_{\mathbf{x}}} = -h(\mathbf{x}) + \sum_{i=1}^{m}\tilde{h}(X_i) \tag{10.42}$$

在本章后面，我们将要利用这一公式来学习独立分量分析。

Pythagorean 分解

下面我们考虑概率密度函数 $p_{\mathbf{X}}(\mathbf{x})$和 $p_{U}(\mathbf{x})$之间的相对熵，其中向量 $\mathbf{x}$ 是随机向量 $\mathbf{X}$ 和 $\mathbf{U}$

的共同样本，x_i 是 $\mathbf{x}$ 的第 i 个分量。$m\times 1$ 随机向量 $\mathbf{U}$ 是由独立的变量组成：

$$p_{\mathbf{U}}(\mathbf{x}) = \prod_{i=1}^{m} pu_i(x_i)$$

而 $m\times 1$ 的随机变量 $\mathbf{X}$ 通过 $\mathbf{U}$ 定义为

$$\mathbf{X} = \mathbf{AU}$$

其中 $\mathbf{A}$ 是一个非对角矩阵。令 $\tilde{p}_{X_i}(x_i)$ 表示从 $p_{\mathbf{X}}(\mathbf{x})$ 导出的每一个 X_i 的边缘概率密度，则 $p_{\mathbf{X}}(\mathbf{x})$ 和 $p_{\mathbf{U}}(\mathbf{x})$ 之间的相对熵可以作如下的 Pythagorean 分解：

$$D_{p_{\mathbf{X}}\|p_{\mathbf{U}}} = D_{p_{\mathbf{X}}\|\tilde{p}_{\mathbf{X}}} = D_{p_{\mathbf{X}}\|\tilde{p}_{\mathbf{U}}} \tag{10.43}$$

我们之所以称这个经典的关系为 Pythagorean 分解，是因为它具有信息-几何解释（Amari，1985）[5]。

10.6　系词

互信息 $I(X;Y)$ 提供了两个随机变量 X 和 Y 之间的统计独立性测量。对这一依赖的图解，我们可以参看基于式(10.32)的图解图 10.1。然而，这一等式缺少数学上的洞察力。具体地，如果互信息 $I(X;Y)$ 是 0，它告诉我们随机变量 X 和 Y 是统计独立的。但是，如果 $I(X;Y)$ 大于 0，确认在 X 和 Y 之间的统计依存性，却没有提供给我们这一依存的统计测量。

为了详细说明，考虑一对随机变量，其样本值相应地记为 x 和 y。感兴趣的问题是形成在 X 和 Y 之间的统计依存的测量，其不受其尺度变换或变化的影响。为了完成这一目标，我们将 X 和 Y 变换为相应的两个新的随机变量 U 和 V，使得 U 和 V 在区间 $[0,1]$ 上一致分布。这一变换是一种非线性尺度变换，可用累积分布函数 $P_X(x)$ 和 $P_Y(y)$ 来表示；它是通过设

$$u = P_X(x)$$

和

$$v = P_Y(y)$$

来完成的。其中 u 和 v 是随机变量 U 和 V 的相应的样本值。(U,V) 的联合概率分布函数在单位正方形 $[0,1]\times[0,1]$ 上分布；当且仅当原始随机变量 X 和 Y（或者，等价于新的随机变量 U 和 V）是统计独立时这个分布是一致的。X 和 Y 的联合分布因此转换为 U 和 V 在单位正方形上的联合分布，这里边缘分布是一致的。

新的随机变量对 (U,V) 是唯一决定的，它被称为系词（copula）[6]。正式地，

系词是在自由分布方式下模型化 U 和 V 之间统计依存的函数，包含随机变量对 (U,V)。

我们可以继续说明关于系词的 Sklar 定理如下（Sklar，1959）：

给定累积分布函数 $P_{X,Y}(x,y)$，$P_X(x)$ 和 $P_Y(y)$，存在唯一的系词 $C_{U,V}(u,v)$ 满足下面的关系：

$$P_{X,Y}(x,y) = C_{U,V}(P_X(x),P_Y(y)) \tag{10.44}$$

和

$$C_{U,V}(u,v) = P(P_X^{-1}(x),P_Y^{-1}(y)) \tag{10.45}$$

其中两个新的随机变量 U 和 V 是原始随机变量 X 和 Y 对应的非线性变换，其样本 u 和 v 定义为

$$u = P_X(x) \tag{10.46}$$

和

$$v = P_Y(y) \tag{10.47}$$

随机变量对 (U,V) 的联合分布在单位正方形上分布。

系词的性质

性质 1. 系词的有限值

由于样本 u 和 v 局限于范围 $[0,1]$，系词值自身就局限于

$$C_{U,V}(u,0)=C_{U,V}(0,v)=0 \quad C_{U,V}(u,1)=u \quad C_{U,V}(1,v)=v$$

性质 2. 利用系词表示联合密度 $p_{X,Y}(x,y)$

用系词来将联合概率密度函数 $p_{X,Y}(x,y)$表示为三项的积：

- 边缘概率密度函数 $p_X(x)$和 $p_Y(y)$。
- 系词的联合概率密度函数 $c_{U,V}(u,v)$。

为了建立这一关系，我们从联合概率密度函数的基本定义开始：

$$p_{X,Y}(x,y)=\frac{\partial^2}{\partial x\partial y}P_{X,Y}(x,y)$$

然后，利用式(10.44)，我们写

$$p_{X,Y}(x,y)=\frac{\partial^2}{\partial x\partial y}C_{U,V}(P_X(x),P_Y(y))=\frac{\partial}{\partial x}\frac{\partial}{\partial y}C_{U,V}(P_X(x),P_Y(y))$$

$$=\frac{\partial}{\partial x}\left[\frac{\partial P_Y(y)}{\partial y}\frac{\partial}{\partial P_Y(y)}C_{U,V}(P_X(x),P_Y(y))\right]=\frac{\partial}{\partial x}[p_Y(y)C'_{U,V}(P_X(x),v)]$$

其中，在最后一行，利用定义 $P_Y(y)=v$，$C'_{U,V}(P_X(x),P_Y(y))$ 表示系词对 $P_Y(y)$的微分。边缘 $P_Y(y)$是独立于 x 的，我们继续写

$$p_{X,Y}(x,y)=p_Y(y)\frac{\partial}{\partial x}C'_{U,V}(P_X(x),v)=p_Y(y)\frac{\partial P_X(x)}{\partial x}\frac{\partial}{\partial P_X(x)}C'_{U,V}(P_X(x),v)$$

$$=p_Y(y)p_X(x)C''_{U,V}(P_X(x),v)$$

这里 $C''_{U,V}(P_X(x),v)$表示导数 $C'_{U,V}(P_X(x),v)$对 $P_X(x)$的微分。最后，认识到 $P_X(x)=u$，通过定义，系词的联合概率密度函数表示为

$$c_{U,V}(u,v)=\frac{\partial^2}{\partial u\partial v}C_{U,V}(u,v) \tag{10.48}$$

我们获得下面的关系：

$$p_{X,Y}(x,y)=p_X(x)p_Y(y)c_{U,V}(u,v) \tag{10.49}$$

式(10.49)引导我们给出如下的说明：

如果两个随机变量 X 和 Y 是统计依存的，则系词的联合密度 $c_{U,V}(u,v)$清晰地说明了 X 和 Y 之间的统计依存。

例 4　两个统计独立随机变量的系词

令随机变量 X 和 Y 为统计独立的。我们就有

$$p_{X,Y}(x,y)=p_X(x)p_Y(y)$$

在这一条件下，式(10.49)衰减为

$$c_{U,V}(u,v)=1, \qquad \text{当 } 0\leqslant u,v\leqslant 1$$

相应地有

$$C_{U,V}(u,v)=\int_0^u\int_0^v C_{U,V}(u,v)\mathrm{d}u\mathrm{d}v=\int_0^u\int_0^v 1\mathrm{d}u\mathrm{d}v=uv$$

因此，当相应的随机变量 X 和 Y 是统计独立的时候，系词的密度 $C_{U,V}(u,v)=uv$ 将 U 和 V 连接起来。 ■

互信息和系词的熵之间的关系

有了刚刚介绍过的系词的背景，我们现在可以给出另一个说明：

两个随机变量 X 和 Y 之间的互信息是相应的非线性变换随机变量对 U 和 V 的系词联合熵的相反数。

为了说明这一关系，我们讲述如下：

1. 由于随机变量 U 和 V 是作用于原始随机变量 X 和 Y 上的可逆变换，根据第 10.4 节讲述的互信息的不变性立即可得

$$I(X;Y) = I(U;V)$$

2. 将式(10.32)的最后一行作用到互信息 $I(U;V)$ 上有

$$I(U;V) = h_C(U) + h_C(V) - h_C(U,V)$$

由于随机变量 U 和 V 都在区间 $[0,1]$ 上一致分布，因此微分熵 $h(U)$ 和 $h(V)$ 是 0。因此，$I(U;V)$ 衰减为

$$I(U;V) = -h_C(U,V) = \mathbb{E}[\log c_{U,V}(u,v)] \tag{10.50}$$

这是所需要的关系。

在式(10.50)中定义的互信息直观上比式(10.32)中给出的三个标准公式更让人满意，这是因为以下两种原因：

1. 给定一对随机变量，它们之间的互信息直接表示为系词的函数，而系词是和两个随机变量之间依存性相匹配的潜在分布的部分。

2. 互信息不是两个随机变量边缘分布的函数。

此外，根据式(10.49)，可以有两个更加深刻的备注：

$$I(X;Y) = 0 \text{ 对应于 } c_{U,V}(u,v) = 1$$
$$I(X;Y) > 0 \text{ 对应于 } c_{U,V}(u,v) > 1$$

10.7　互信息作为最优化的目标函数

现在我们对香农的信息论已经有了足够的了解，可以讨论它在研究自组织系统中的作用。

为了进行讨论，设有一个多输入/多输出的神经网络系统。在这里主要目标是为一个特定任务（例如，建模、抽取统计突出特征或信号分离）而设计的系统进行自组织。通过选择某些系统变量间的互信息作为优化的目标函数，这个要求可以满足。这种特定的选择由下面两个考虑得到证明：

1. 如同第 10.4 节到第 10.6 节的讨论，互信息具有一些独特的性质。

2. 无需教师也可确定互信息，这样自然就完成了自组织的准备。

问题变成了系统调整自由参数之一（即突触权值）以优化互信息的问题。

根据感兴趣的应用的不同，我们能够确定如图 10.2 所示的 4 种不同情况。这些情况可以描述如下：

- 在图 10.2a 描绘的情况 1，输入向量 X 由分量 $X_1, X_2, \cdots, X_m$ 组成，输出向量 $\mathbf{Y}$ 由分量 $Y_1, Y_2, \cdots, Y_l$ 组成。需求是最大化传送到系统输出 $\mathbf{Y}$ 的关于系统输入 $\mathbf{X}$ 的信息（即通过系统的信息流）。
- 在图 10.2b 描绘的情况 2，一对输入向量 $\mathbf{X}_a$ 和 $\mathbf{X}_b$ 是从相邻但不重叠的图像区域截取而来。各自产生的标量输出分别是 $\mathbf{Y}_a$ 和 $\mathbf{Y}_b$。需求是最大化传送到 $\mathbf{Y}_a$ 的关于 $\mathbf{Y}_b$ 的信息，以及相反的需求。
- 在图 10.2c 描绘的情况 3，输入向量 $\mathbf{X}_a$ 和 $\mathbf{X}_b$ 是从两幅不同但相关的图像相应部分截取而来。各自产生的输出分别是 $\mathbf{Y}_a$ 和 $\mathbf{Y}_b$，需求是最小化传送到 $\mathbf{Y}_a$ 的关于 $\mathbf{Y}_b$ 的信息，以及相反的需求。
- 在图 10.2d 描绘的情况 4，输入向量 $\mathbf{X}$ 和输出向量 $\mathbf{Y}$ 与图 10.2a 定义的形式相似，但有相

同的维数（即 $l=m$）。这里的目标是使输出向量 $\mathbf{Y}$ 的各分量之间的统计相关最小化。

在所有的这 4 种情况下，互信息起核心作用。但是，它的推导过程还是要根据所考虑的具体情况而定。在本章余下的部分将以刚才罗列的顺序讨论涉及这些情况的问题以及它们的实际含义。更重要的是，必须指出情况 4 包含了本章中讲述的理论、计算算法、应用的多个素材，这反映了信息论模型的实践关系。

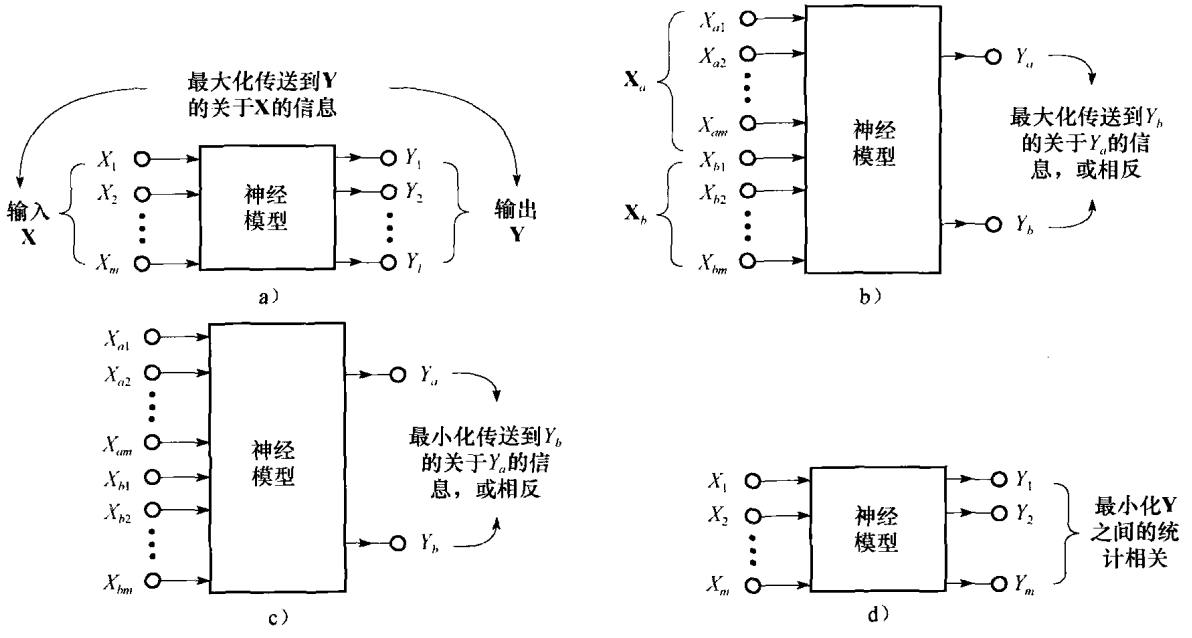


图 10.2　适用于信息最大化的应用及其三个变体的四种基本情况

10.8　最大互信息原则

设计一个神经处理器，使互信息 $I(\mathbf{Y};\mathbf{X})$ 最大的思想是吸引人的，这一思想是统计信号处理的基础。这种优化方法在 Linsker(1987，1988a，1989a) 提出的最大互信息（maximum mutual information (Infomax)）原则中得以体现，它可正式陈述如下：

从神经系统的输入层观测到的随机向量 $\mathbf{X}$ 到系统的输出层得到的随机向量 $\mathbf{Y}$ 之间的变换应该这样选择，这种变换使得输出层神经元的活动共同最大化关于输入层神经元的活动的信息。最大化的目标函数是向量 $\mathbf{X}$ 和 $\mathbf{Y}$ 之间的互信息 $I(\mathbf{Y};\mathbf{X})$。

最大互信息原则提供了一个解决如图 10.2a 所描述的信息传输系统自组织的数学框架，它独立于实现它所使用的规则，假设输出向量 Y 的分量数 l 小于输入向量 $\mathbf{x}$ 的分量数 m。同样，这个原则也可以看作信道容量这个概念在神经网络中的对应物，信道容量定义为通过一个通信信道的信息传输率的香农极限。

接下来，我们给出两个涉及有噪声的单神经元的例子说明最大互信息原则的应用。在一个例子中噪声出现在输出端，而在另一个例子中噪声出现在输入端。

例 5　被过程噪声破坏的单神经元

考虑线性神经元的简单情形，假设系统从 m 个源节点接受输入。令该神经元的输出中出现过程噪声，可表示为

$$Y=\left(\sum_{i=1}^{m}w_iX_i\right)+N \tag{10.51}$$

其中 w_i 为第 i 个突触权值，N 为过程噪声，如图 10.3 所示的模型。假设：

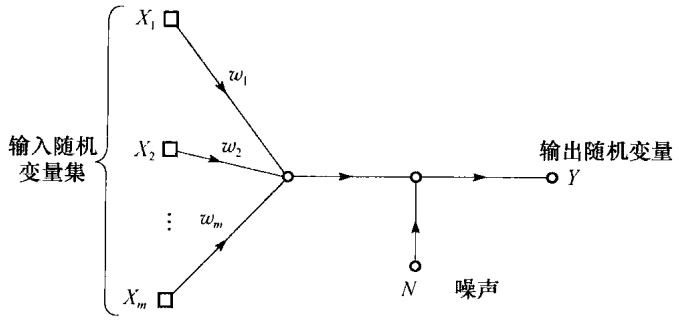


图 10.3　噪声神经元的信号流图

- 输出 Y 是一个零均值以方差为 σ_Y^2 的高斯随机变量。
- 过程噪声 N 也是一个高斯随机变量，均值为 0，方差为 σ_N^2。
- 过程噪声 N 与输入向量的任何一个分量都不相关，也即

$$\mathbb{E}[NX_i] = 0, \quad \text{对于所有 } i$$

输出 Y 的高斯性可以用两种方法之一得到满足。首先，输入 $X_1, X_2, \cdots, X_m$ 全部是高斯分布的。再假设附加的噪声 N 也是高斯的，则 Y 的高斯性可以保证，这是由于一组高斯分布的随机变量的加权和仍是高斯的。或者，输入 $X_1, X_2, \cdots, X_m$ 是统计独立的，且在温和的条件下，利用概率论的中心极限定理，它们的加权和在 m 很大时趋近于高斯分布。

为了进行分析，我们首先注意到在式(10.32)的第二行，神经元的输出向量 Y 和输入向量 $\mathbf{X}$ 之间的互信息 $I(\mathbf{Y};\mathbf{X})$ 是

$$I(Y;\mathbf{X}) = h(Y) - h(Y|\mathbf{X}) \tag{10.52}$$

根据式(10.51)，注意在已知输入向量 $\mathbf{X}$ 的情况下，输出 Y 的概率密度函数等于一个常数加上一个高斯分布的随机变量的概率密度函数。因此，条件熵 $h(Y|\mathbf{X})$ 是由输出神经元传送的关于过程噪声 N 而不是信号向量 $\mathbf{X}$ 的“信息”。我们可以设置：

$$h(Y|\mathbf{X}) = h(N)$$

因此式(10.52)可以重新简化为

$$I(Y;\mathbf{X}) = h(Y) - h(N) \tag{10.53}$$

应用式(10.22)关于高斯随机变量的微分熵到当前的问题，我们得到

$$h(Y) = \frac{1}{2}[1 + \log(2\pi\sigma_Y^2)] \tag{10.54}$$

和

$$h(N) = \frac{1}{2}[1 + \log(2\pi\sigma_N^2)] \tag{10.55}$$

经过简化，将式(10.54)和式(10.55)代入式(10.53)得

$$I(Y;\mathbf{X}) = \frac{1}{2}\log\left(\frac{\sigma_Y^2}{\sigma_N^2}\right) \tag{10.56}$$

其中 σ_Y^2 依赖于 σ_N^2。

比值 σ_Y^2/σ_N^2 可看作信噪比。假设噪声方差 σ_N^2 为固定的约束条件，从式(10.56)看出互信息 $I(Y;X)$ 是通过神经元输出 Y 的方差 σ_Y^2 的最大化而最大化的。因此可以这样说，在一定的条件下，使神经元输出的方差最大化也就是使神经元的输出信号和它的输入之间的互信息最大化。

最后，由附加的过程噪声破坏的单一神经元的处理基于最小化输出方差，产生了由第 8 章讨论过的 Oja 规则训练的 PCA 神经元的一个解。 ■

例 6　受附加输入噪声破坏的单个神经元

假设噪声影响在每一个输入节点的突触末端的线性神经元的行为，如图 10.4 所示。根据第二个噪声模型得出：

$$Y = \sum_{i=1}^{m} w_i(X_i + N_i) \tag{10.57}$$

其中假设每个噪声分量 N_i 是一个独立高斯随机变量，其均值为 0，共同方差为 σ_N^2。将式(10.57)改写成类似式(10.51)的形式：

$$Y = \left(\sum_{i=1}^{m} w_i X_i\right) + N'$$

其中 N'是噪声分量的组合，定义为

$$N' = \sum_{i=1}^{m} w_i N_i$$

噪声 N'是一个高斯分布，其均值为 0，方差为所有独立噪声分量方差的加权和，即

$$\sigma_{N'}^2 = \sum_{i=1}^{m} w_i^2 \sigma_N^2$$

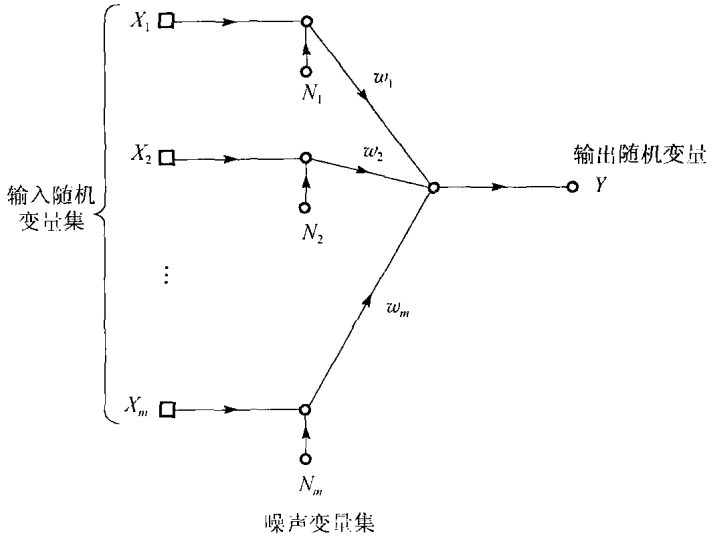


图 10.4　神经元的另一个噪声模型

与前类似，我们假设神经元的输出变量 Y 是方差为 σ_Y^2 的高斯分布。Y 和 $\mathbf{X}$ 之间的互信息 $I(Y;\mathbf{X})$同样由式(10.52)给出。但是，这一次条件熵 $h(Y|\mathbf{X})$定义如下：

$$h(Y|\mathbf{X}) = h(N') = \frac{1}{2}(1 + 2\pi\sigma_{N'}^2) = \frac{1}{2}\left[1 + 2\pi\sigma_N^2 \sum_{i=1}^{m} w_i^2\right] \tag{10.58}$$

这样，将式(10.54)和式(10.58)代入式(10.52)并简化，可得

$$I(Y;\mathbf{X}) = \frac{1}{2}\log\left(\frac{\sigma_Y^2}{\sigma_N^2 \sum_{i=1}^{m} w_i^2}\right) \tag{10.59}$$

在约束噪声方差 σ_N^2 保持常量的条件下，$I(Y;\mathbf{X})$的最大化就是比值 $\sigma_Y^2 / \sum_{i=1}^{m} w_i^2$ 的最大化，其中 σ_Y^2 是 w_i 的函数。 ■

我们可从例 5 和例 6 推出什么结论？ 首先，从给出的两个例子可以看出，应用最大熵原则的结果依赖于问题。对于给定噪声方差 σ_N^2，最大化互信息 $I(Y;\mathbf{X})$和应用于图 10.3 的模型

输出的方差之间的比等价，并不能直接转到图 10.4 的模型。只有当对图 10.4 的模型加上 $\sum_i w_i^2 = 1$ 的约束时，图 10.4 和图 10.3 所代表的模型才有相似的行为。

一般说来，确定输入向量 $\mathbf{X}$ 与输出向量 $\mathbf{Y}$ 的互信息 $I(Y;X)$ 是很困难的。在例 5 和例 6 中，为了数学上分析的方便，我们假设系统噪声分布是一个或多个噪声源的多元高斯分布。这个假设需要在最大互信息原则的实际应用中验证。

当采用高斯噪声模型时，本质上是采用互信息的一个替代，其计算的前提是神经元的输出向量 $\mathbf{Y}$ 是一个均值向量和协方差矩阵都与实际情况相同的多元高斯分布。在 Linsker (1993) 中，利用相对熵提供对于这种条件下替代互信息的一个原则性理由，这些都假设网络已经存储关于输出向量 $\mathbf{Y}$ 的均值向量和协方差矩阵而不包含更高阶统计。

最后，在例 5 和例 6 给出的分析情况只是对于一个神经元进行的。有意这样做是为了最大互信息原则在数学上易于处理，最优化应该在局部神经元级进行。这种优化符合自组织的本质。

例 7　无噪声网络

在例 5 和例 6 中，考虑了带有噪声的神经元。在本例中我们研究一个无噪声的网络，它将任意分布的随机向量 $\mathbf{X}$ 变换为新的具有不同分布的随机向量 $\mathbf{Y}$。注意 $I(\mathbf{X};\mathbf{Y}) = I(\mathbf{Y};\mathbf{X})$，并且在这里展开式(10.32)的第 2 行，可以将输入向量 $\mathbf{X}$ 和输出向量 $\mathbf{Y}$ 之间的互信息表达为：

$$I(\mathbf{Y};\mathbf{X}) = h(\mathbf{Y}) - h(\mathbf{Y}|\mathbf{X})$$

其中 $h(\mathbf{Y})$是 $\mathbf{Y}$ 的熵，$h(\mathbf{Y}|\mathbf{X})$是在给定 $\mathbf{X}$ 的条件下 $\mathbf{Y}$ 的条件熵。假设从 $\mathbf{X}$ 到 $\mathbf{Y}$ 的映射是无噪声的，条件熵 $h(\mathbf{Y}|\mathbf{X})$取其最小的可能值：它发散到 $-\infty$。这是由于在 10.2 节讨论的连续随机变量熵的微分特性的必然结果。但是，当我们考虑互信息 $I(\mathbf{Y};\mathbf{X})$对参数化映射网络的权值矩阵 $\mathbf{W}$ 的梯度时，这个困难并不造成什么后果。具体地，可以写成

$$\frac{\partial I(\mathbf{Y};\mathbf{X})}{\partial \mathbf{W}} = \frac{\partial h(\mathbf{Y})}{\partial \mathbf{W}} \tag{10.60}$$

因为条件熵 $h(\mathbf{Y}|\mathbf{X})$与 $\mathbf{W}$ 独立。式(10.60)表明：

对于一个无噪声映射网络，最大化网络输出 $\mathbf{Y}$ 的微分熵就等于最大化 $\mathbf{Y}$ 和网络输入 $\mathbf{X}$ 之间的互信息，两者都是关于映射网络的权矩阵 $\mathbf{W}$ 最大化。 ■

10.9　最大互信息和冗余减少

在香农的信息论框架中，序和结构代表冗余，它减少接受方对信息分辨的不确定性。在固有过程中我们拥有的序和结构越多，则观察这个过程获得的信息量就越少。例如考虑高度结构化和冗余的序列 $aaaaaa$。一旦得到第一个样本 a，则我们就可以立即知道其余后面五个都是一样的 a。这样的一个序列所传递的信息的极限是单个符号传递的信息量。换句话说，样本序列的冗余越大，序列中所含的信息内容也就越少，但是该信息内容的结构越多。

从互信息 $I(\mathbf{Y};\mathbf{X})$的定义，我们知道这是通过观察系统输入 $\mathbf{X}$ 来决定输出 $\mathbf{Y}$ 的不确定性的度量。最大互信息原则是使互信息 $I(\mathbf{Y};\mathbf{X})$最大，其结果是我们在观测到输入为 $\mathbf{X}$ 时，对系统输出 $\mathbf{Y}$ 增加确定性。考虑到前面提到的信息与冗余之间的关系，因此我们可以说：

最大互信息原则导致与在输入 $\mathbf{X}$ 中的冗余比较而言减少输出 $\mathbf{Y}$ 中的冗余。

噪声的出现是推动使用冗余以及相异性（diversity）相关方法的一个因素，相异性的定义如下：通过一个处理器产生不同性质的两个或多个输出。而且，当输入信号的附加性噪声很高时，我们可以利用冗余来减少噪声的效果。在这种环境下，输入信号之间的更多（相关）分量都由处理器组合起来，以提供输入的精确表示。同样，当输出端的噪声（即处理器噪声）很高时，给出更多的输出分量以提供冗余信息。在处理器输出端观测到的相互独立的属性也相应地

减少了，但各个属性表示的精确度反而提高了。因此，高水平的噪声有利于表示的冗余。但是，当噪声水平很低时，表示的相异性比冗余更有利。

感知系统建模

自从信息论的早期，就提出了感觉消息（刺激）的冗余对感知理解非常有用（Attneave，1954；Barlow，1959）。感觉消息的冗余提供了人脑建立其周围环境的“认知映射”或“工作模型”。在感觉消息中规则必须以某种方式被人脑编码，使它知道什么经常发生。但是，冗余减少是 Barlow 假设的特定形式。这个假设说明：

早期处理的目的是将高冗余的感觉输入转化成更有效的析因码（factorial code）。

换句话说，在输入的条件下使神经元输出统计独立。

受 Barlow 假设的启发，Atick and Redlich(1990) 提出把最小冗余原则作为如图 10.5 所示的感知系统的信息论模型的基础。系统由三个部分组成：输入通道、重编码系统和输出通道。输入通道的输出可以表示为：

$$\mathbf{X} = \mathbf{S} + \mathbf{N}_i$$

其中 $\mathbf{S}$ 是输入通道接收到的理想信号，$\mathbf{N}_i$ 假设为输入中所有噪声的源。随后信号 $\mathbf{X}$ 被线性矩阵算子 $\mathbf{A}$ 变换（重编码），然后通过视觉神经或输出通道传输，产生输出 $\mathbf{Y}$，表示为

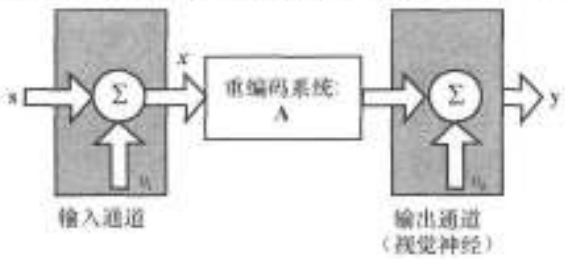


图 10.5　感知系统模型。信号向量 $\mathbf{s}$ 和噪声向量 v_i 和 v_o 分别是随机向量 $\mathbf{S}$，$\mathbf{N}_i$ 和 $\mathbf{N}_o$ 的值

$$\mathbf{Y} = \mathbf{AX} + \mathbf{N}_o$$

其中 $\mathbf{N}_o$ 表示后编码本身的噪声。在 Atick 和 Redlich 的方法中，观察到达视网膜的光信号包含一些非常有用的高冗余形式的感觉信息。进一步假设在信号沿视觉神经发送以前视网膜信号处理的目的就是减少或消除由于关联性和噪声所带来的数据冗余。为了量化描述这种观点，冗余度度量定义如下：

$$R = 1 - \frac{I(\mathbf{Y};\mathbf{S})}{C(\mathbf{Y})} \tag{10.61}$$

其中 $I(\mathbf{Y};\mathbf{S})$ 是 $\mathbf{Y}$ 和 $\mathbf{S}$ 之间的互信息，$C(\mathbf{Y})$ 是视觉神经（输出通道）的信道容量。式(10.61)的合理性基于人脑感兴趣的信息是理想的输入信号 $\mathbf{S}$，但是信息必须经过的物理信道实际上是视觉神经。假设在感知系统完成的输入与输出映射之间没有维数减少，这意味着 $C(\mathbf{Y}) > I(\mathbf{Y};\mathbf{S})$。要求找到一个输入-输出映射（即矩阵 $\mathbf{A}$）使冗余度度量 R 达到最小且满足不丢失信息的约束，可以表示为

$$I(\mathbf{Y};\mathbf{X}) = I(\mathbf{X};\mathbf{X}) - \epsilon$$

其中ϵ是一些很小的正参数。式(10.61)中信道容量 $C(\mathbf{Y})$ 定义为保持平均输入能量固定的条件下对所有应用于它的输入的概率分布，可能流过视觉神经的最大信息率。

当信号向量 $\mathbf{S}$ 和输出向量 $\mathbf{Y}$ 有相同的维数和系统存在噪声时，最小冗余度原则和最大互信息原则数学上是等价的，只要假设在两种情况下输出神经元计算能力的约束相同。具体地，假设根据图 10.5 的模型中信道容量的度量取决于每一个神经元输出的动态范围。那么，根据最小冗余度原则，对于一个给定的允许信息丢失，以及从而对于一个给定的 $I(\mathbf{Y};\mathbf{S})$，需要最小化的量定义为：

$$1 - \frac{I(\mathbf{Y};\mathbf{S})}{C(\mathbf{Y})}$$

因此，对于参数 λ，这样最小化的量本质上为

$$F_1(\mathbf{Y};\mathbf{S}) = C(\mathbf{Y}) - \lambda I(\mathbf{Y};\mathbf{S}) \tag{10.62}$$

另一方面，根据最大互信息原则，在图 10.5 的模型中需要最大化的量为：

$$F_2(\mathbf{Y};\mathbf{S}) = I(\mathbf{Y};\mathbf{S}) + \lambda C(\mathbf{Y}) \tag{10.63}$$

虽然函数 $F_1(\mathbf{Y};\mathbf{S})$和 $F_2(\mathbf{Y};\mathbf{S})$并不相同，但是它们的最优化产生相同的结果：它们都是拉格朗日乘子法的公式，仅仅是 $I(\mathbf{Y};\mathbf{S})$和 $\mathbf{C}(\mathbf{Y})$简单地互换了角色。

从这些讨论中注意到这样一个重要的观点：虽然公式不同，但是这两个信息论的原则产生相似的结果[7]：

一个神经系统输出和输入之间的互信息的最大化确实可以导致冗余削减。

10.10　空间相干特征

在 10.8 节中提出的最大互信息原则，主要应用于如图 10.2a 所示的情况下，神经系统的输出向量 $\mathbf{Y}$ 和输入向量 $\mathbf{X}$ 之间的互信息 $I(\mathbf{Y};\mathbf{X})$作为一个求最大值的目标函数。在术语上作适当改变，我们可以将其扩展到自然景物图像的无监督处理中（Becker and Hinton，1992）。一个未处理的图像的像素，虽然形式很复杂，但是包含我们感兴趣的景物的丰富信息。特别是，每个像素的密集度受内在参数的影响，例如深度、反射、表面方向和背景噪声以及照明度。目的就是设计一个自组织系统，能够学习将这种复杂的信息编码成一种简单的形式。更具体一点，目标就是从这个图像中提取能够展现该图像空间相干的高阶特征，使得在图像的空间局部区域的信息表示很容易产生邻近区域的信息表示；区域是指图像中的一组像素的集合。这里描述的情况属于图 10.2b 的场景。

因此我们可以将 Imax 原则的情况 2 说明如下（Becker，1996；Becker and Hinton，1992）：

两个向量 $\mathbf{X}_a$ 和 $\mathbf{X}_b$（代表一个神经系统相邻的无重叠的图像区域）的变换应该如此选择，使得输入 $\mathbf{X}_a$ 对应的标量输出 $\mathbf{Y}_a$ 最大化输入 $\mathbf{X}_b$ 对应的标量输出 $\mathbf{Y}_b$ 的信息，反之亦然。最大化的目标函数就是输出 $\mathbf{Y}_a$ 和 $\mathbf{Y}_b$ 之间的互信息 $I(Y_a;Y_b)$。

尽管 Imax 原则并不和最大互信息原则相等价或能够从其推导出来，但它必定按相似的思想起作用。

例 8　相干图像处理

考虑图 10.6 所示的例子，有两个神经网络（模型）a 和 b，分别接受输入为 $\mathbf{X}_a$ 和 $\mathbf{X}_b$，来自同一图像中相邻的不重叠区域，各自的标量输出分别是 $\mathbf{Y}_a$ 和 $\mathbf{Y}_b$。令 S 表示 Y_a 和 Y_b 中共同信号分量，它是原始图像的两个相关区域的空间相干性的表示。我们可以将 Y_a 和 Y_b 看成共同信号 S 的带噪声形式，表示为：

$$Y_a = S + N_a$$

和

$$Y_b = S + N_b$$

N_a 和 N_b 是加性噪声分量，假设为统计独立的零均值高斯分布随机变量。信号分量 S 也假设为高斯分布的。根据这两个式子，在图 10.6 中假设模块 a 和 b 彼此相容。

利用式(10.32)的最后一行，Y_a 和 Y_b 的互信息定义为：

$$I(Y_a;Y_b) = h(Y_a) + h(Y_b) - h(Y_a,Y_b) \tag{10.64}$$

根据式(10.22)的公式，关于高斯随机变量的微分熵，Y_a 的微分熵 $h(Y_a)$ 为：

$$h(Y_a) = \frac{1}{2}[1 + \log(2\pi\sigma_a^2)] \tag{10.65}$$

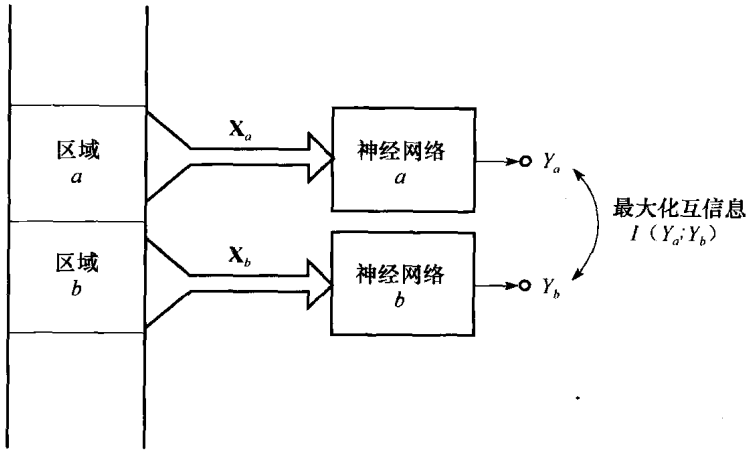


图 10.6　按照 Imax 原则处理图像的两个邻近区域

其中 σ_a^2 是 Y_a 的方差。同理得 Y_b 的微分熵为：

$$h(Y_b) = \frac{1}{2}[1 + \log(2\pi\sigma_b^2)] \tag{10.66}$$

其中 σ_b^2 是 Y_b 的方差。至于联合微分熵 $h(Y_a, Y_b)$，利用式 (10.24) 得

$$h(Y_a;Y_b) = 1 + \log(2\pi) + \frac{1}{2}\log|\det(\boldsymbol{\Sigma})|$$

2×2 的矩阵 Σ 是 Y_a 和 Y_b 的协方差矩阵，定义为：

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_a^2 & \rho_{ab}\sigma_a\sigma_b \\ \rho_{ab}\sigma_a\sigma_b & \sigma_b^2 \end{bmatrix} \tag{10.67}$$

和

$$\det(\boldsymbol{\Sigma}) = \sigma_a^2\sigma_b^2(1-\rho_{ab}^2)$$

其中 ρ_{ab} 是 Y_a 和 Y_b 的相关系数；也就是

$$\rho_{ab} = \frac{\mathbb{E}[(Y_a - \mathbb{E}[Y_a])(Y_b - \mathbb{E}[Y_b])]}{\sigma_a\sigma_b}$$

所以可以重写 Y_a 和 Y_b 的联合微分熵为：

$$h(Y_a;Y_b) = 1 + \log(2\pi) + \frac{1}{2}\log[\sigma_a^2\sigma_b^2(1-\rho_{ab}^2)] \tag{10.68}$$

将式(10.65)、式(10.66)和式(10.68)代入式(10.64)，并简化得

$$I(Y_a;Y_b) = -\frac{1}{2}\log(1-\rho_{ab}^2) \tag{10.69}$$

由式(10.69)立即推出，最大化互信息 $I(Y_a;Y_b)$等价于最大化相关系数 ρ_{ab}。这从直观上看也是满足的。注意，由 ρ_{ab}的定义知，$|\rho_{ab}| \leqslant 1$。 ■

式(10.69)的结果是由图 10.6 的随机系统的输出产生的两个随机变量 Y_a 和 Y_b 的例子推导的，这两者都被假设为高斯分布。然而，在更一般的非高斯分布情形下，相关系数 ρ_{ab} 的利用不能作为 Imax 原则的合适测量。为了一般化 Imax 的运用，我们提出由式(10.50)的公式启发的系词的运用。具体地，考虑图 10.2b 的情形。令 $\mathbf{W}$ 为响应于产生输出 Y_a 和 Y_b 的系统的权重矩阵，而 Y_a 和 Y_b 分别响应于对应的输入向量 $\mathbf{X}_a$ 和 $\mathbf{X}_b$ 的组合影响。然后利用式(10.50)的第一行来形成 Imax 原则的简化：

$$\max_{\mathbf{W}} I(Y_a;Y_b) = \min_{\mathbf{W}} h_C(U_a, U_b; \mathbf{W}) \tag{10.70}$$

其中，根据相关的累积概率分布，得出：

$$u_a = P_{Ya}(y_a)$$

和

$$u_b = P_{Yb}(y_b)$$

且 $h_C(U_a, U_b, \mathbf{W})$ 是随机变量 U_a 和 U_b 的联合微分熵，其对应的样本值是 u_a 和 u_b。等价地，根据式(10.50)的第二行，也可以写成：

$$\max_{\mathbf{w}} I(Y_a; Y_b) = \max_{\mathbf{w}} \mathbb{E}[\log c_{U_a,U_b}(u_a, u_b; \mathbf{W})] \tag{10.71}$$

其中 $c_{U_a,U_b}(u_a, u_b; \mathbf{W})$ 是随机变量 U_a 和 U_b 的系词的联合概率密度函数。式(10.71)的公式包含了式(10.69)的结果作为一个特例；这一公式的重要性将在本章后面讲述。

Imax 和标准相关分析之间的关系

再次考虑两个输入向量 $\mathbf{X}_a$ 和 $\mathbf{X}_b$，他们不必有相同的维数。相应的有两个权（基）向量 $\mathbf{w}_a$ 和 $\mathbf{w}_b$，他们和对应的 $\mathbf{X}_a$ 和 $\mathbf{X}_b$ 具有相同的维数。统计中常用的标准相关分析（canonical correlation analysis（CCA））的目的就是指找到两个线性组合

$$Y_a = \mathbf{w}_a^T \mathbf{X}_a$$

和

$$Y_b = \mathbf{w}_b^T \mathbf{X}_b$$

使它们之间的关联性最大。将这里的问题和 Imax 相比较，我们可以看到实际上 Imax 是 CCA 的非线性副本。对于更详细的 CCA 的说明，读者可以参考注释和参考中的注释 8。

10.11　空间非相干特征

在前面一节里我们讨论了一个无监督的图像处理过程，它从一个图像中提取空间相干特征。现在我们将讨论与那里相反的问题。具体地说，考虑图 10.2c，其中目的是增强从两个不同图像中抽取相应区域的空间差异。在图 10.2b 中，我们是求模块输出间的互信息最大化，在图 10.2c 中我们做相反的工作。

因此我们可以将情况 3 的 Imin 原则[9] 陈述如下（Ukrainec and Haykin，1992，1996）：

从两幅不同图像对应的区域得到的数据作为两个输入向量 $\mathbf{X}_a$ 和 $\mathbf{X}_b$，神经系统对它们的变换的选择应该使得输入 $\mathbf{X}_a$ 对应的系统标量输出 Y_a 关于输入 X_b 对应的系统标量输出 Y_b 信息最小，反之亦然。最小化的目标函数是输出 Y_a 和 Y_b 之间的互信息 $I(Y_a; Y_b)$。

案例研究：雷达偏振测定

例如，Imin 原则可以在雷达偏振测定（radar polarimetry）方面有所应用。雷达监视系统产生一对我们感兴趣的环境的图像，利用在一个偏振方向上传送，在相同或不同偏振方向接收得到反向散射。偏振可以在垂直方向，也可以在水平方向上。例如，我们可能有两幅雷达图像，一幅图像代表相同方向（水平-水平）的偏振，而另一幅为交叉方向（水平发送-垂直接受）的偏振。这样的应用由 Ukrainec and Haykin(1992，1996) 提出，属于在一个双偏振雷达系统中的偏振目标增强。研究中雷达景物的采样描述如下。在一个非相干雷达以水平偏振方式传播，在垂直和水平偏振频道接收雷达返回。感兴趣的目标就是设计一个协件偏振扭曲反射器来将偶然偏振旋转 90 度。在普通的雷达系统操作中，这样一个目标的探测是非常困难的，既因为雷达系统的缺陷也因为地面目标会发生意想不到的偏振，并反射回来产生杂波（clutter）。我们发现需要用一个非线性映射来解释普通雷达返回结果的非高斯分布。目标增强问题变为涉

及约束二次函数最小化的求解问题。最终结果是一个处理后的交叉偏振图像，它在目标可见度方面表现出极大的提高，而且远比我们应用诸如主分量分析之类的线性技术得到的效果要好得多。因为模型无关的概率密度函数估计是一个计算量非常大的工作，所以 Ukraire 和 Haykin 提出的模型对变换后的数据假设是高斯统计分布的。两个高斯变量 Y_a 和 Y_b 的互信息由式(10.69)定义。为了学习两个模型的突触权值，采用了变通的方法。要求是抑制雷达杂波，对水平偏振和垂直偏振的雷达图像这是常见的。为了满足该要求，最小化互信息 $I(Y_a;Y_b)$，满足下面加在权值向量上的约束条件：

$$C = (\mathrm{tr}[\mathbf{W}^T\mathbf{W}] - 1)^2$$

其中 $\mathbf{W}$ 是网络总的权值矩阵，$\mathrm{tr}[\cdot]$是括号内矩阵的迹。如果

$$\nabla_w I(Y_a;Y_b) + \lambda \nabla_w C = 0 \tag{10.72}$$

成立，我们可以得到一个稳定点，其中 λ 是拉格朗日乘子。利用拟牛顿最优化程序寻找最小值。在第 3 章和第 4 章中讨论过拟牛顿方法。

图 10.7 显示 Ukrairec and Haykin(1992，1996) 所用的神经网络结构。对每个模型选择一个高斯径向基函数网络 (RBF)，这是因为它可以提供一系列的固定基函数的好处（即有一个非自适应隐藏层)。输入数据在基函数上展开，然后通过线性权值层相结合；在图 10.7 中的虚线代表两个模块间的交叉耦合连接。高斯函数的中心在区间内均匀选择以便能完整覆盖全部输入区域，它们的宽度选择应用启发式规则。图 10.8a 显示一个在安大略湖岸边的一个公园的水平极化和垂直极化的雷达图像。每一幅图像的范围坐标是沿水平轴的，从左到右递增；方位角坐标沿垂直轴。图 10.8b 显示采用最小化水平极化和垂直极化的雷达图像的互信息的组合图像。一个非常清晰的亮点在图像中可以看出来，它是根据雷达从放在湖边的一个协作偏振扭曲反射器返回的。这里所讨论的例子说明了将 Imin 原则应用于处理空间非相干图像的实际好处[10]。

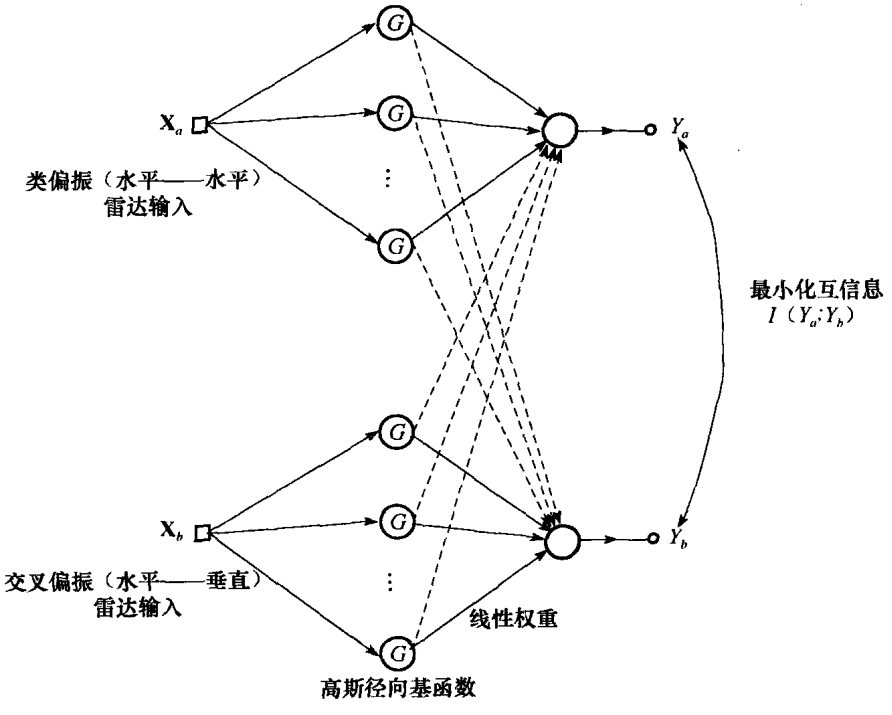
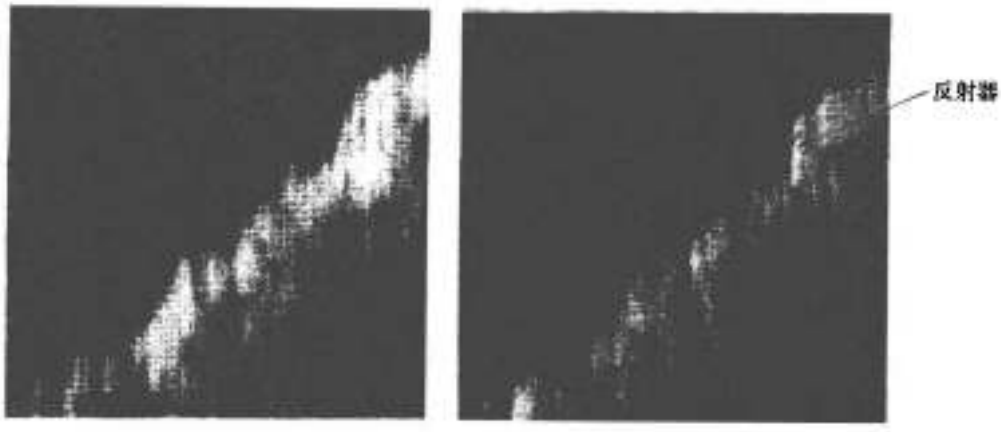
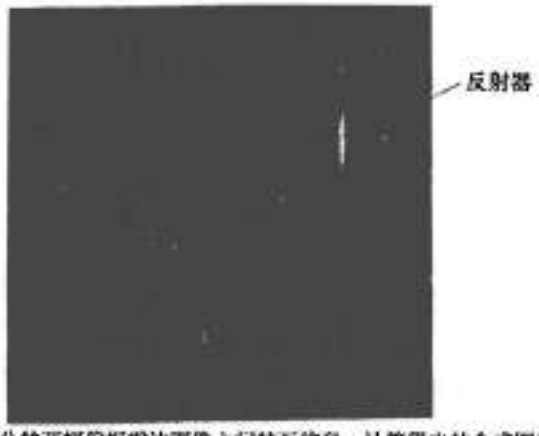


图 10.7　神经处理器框图，其目标是利用一对偏振测定的非相干雷达输入抑制背景杂波；杂波抑制由最小化两个模型输出的互信息来达到



a）未处理的β-扫描雷达图像（方位角和范围对比），水平-水平偏振（上）和水平-垂直（下）偏振



b）最小化a部分的两幅偏振雷达图像之间的互信息，计算得出的合成图像

图 10.8　Imin 原则应用于雷达偏振

Imax 和 Imin 原则的推广

在 10.10 节中构成 Imax 原则以及本节中构成 Imin 原则中，我们论述了对于一对输出终端的互信息 $I(Y_a;Y_b)$ 进行最大化或者最小化。Imax 和 Imin 原则都可以被推广到多个终端的情形，其输出为 $Y_a, Y_b, Y_c, \cdots$，相应地通过最大化或者最小化多元互信息 $I(Y_a;Y_b;Y_c;\cdots)$ 来做。

10.12　独立分量分析

现在我们将注意力集中在由图 10.2d 描述的最后一种情况。为了使那里陈述的信号处理问题更加具体化，考虑图 10.9 的方框图。操作从一个随机源向量 $\mathbf{S}$ 开始，其定义为

$$\mathbf{S} = [S_1, S_2, \cdots, S_m]^T$$

构成 $\mathbf{S}$ 的 m 个随机变量的样本值分别记为 $s_1, s_2, \cdots, s_m$。随机源向量 $\mathbf{S}$ 被作用于一个混合器（mixer），其输入输出之间的关系由一个非奇异的 $m \times m$ 的称为混合矩阵的 $\mathbf{A}$ 决定。由源向量 $\mathbf{S}$ 构成的线性系统和混合器 $\mathbf{A}$ 对于观测者是完全未知的。系统的输出由如下的随机向量定义：

$$\mathbf{X} = \mathbf{AS} = \sum_{i=1}^{m} \mathbf{a}_i S_i \tag{10.73}$$

其中 $\mathbf{a}_i$ 是混合矩阵 $\mathbf{A}$ 的第 i 个列向量，S_i 是由第 i 个源产生的随机信号，$i = 1, 2, \cdots, m$。随机向量 $\mathbf{X}$ 相应地记为

$$\mathbf{X} = [X_1, X_2, \cdots, X_m]^T$$

X_j 的样本值记为 x_j，其中 $j = 1, 2, \cdots, m$。

式(10.73)所述的模型称为生成模型（generative model），是在其负责生成随机变量 $X_1, X_2, \cdots, X_m$ 的意义上而言的。相应地，组成源向量 $\mathbf{S}$ 的随机变量 $S_1, S_2, \cdots, S_m$ 被称为潜在变量，意思是他们不能被直接观察。

盲源分离问题[1]

图 10.9 的方框图包含一个分离器，由 $m \times m$ 分离矩阵 $\mathbf{W}$ 来表示。响应于观测向量 $\mathbf{X}$，分离器产生一个由下面的随机向量定义的输出：

$$\mathbf{Y} = \mathbf{WX}$$

根据此我们现在可以给出如下声明：

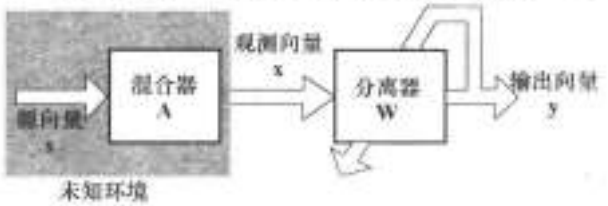


图 10.9　用于解盲源分离问题的处理器方框图。向量 s，x 和 y 是相应的随机向量 S，X 和 Y 的值

给定由潜在（源）变量 $S_1, S_2, \cdots, S_m$ 的未知线性混合所得到的观测向量 $\mathbf{X}$ 的独立实现集合，估计分离矩阵 $\mathbf{W}$ 使得得到的输出向量 $\mathbf{Y}$ 的分量尽可能地统计独立；这里，术语“独立”应该被理解为其强烈的统计意义。

这一声明说明了盲源分离问题的本质。这个问题称为盲的是为了强调这样的事实：对于分离矩阵 $\mathbf{W}$ 的估计是在非监督方式下进行的。而且，用于恢复原始源信息 $\mathbf{S}$ 的仅有信息是包含在观测向量 $\mathbf{X}$ 中的。这种包含于解盲源分离（BSS）问题中的内在原则被称为独立分量分析（Comon，1994）。独立分量分析（ICA）可看作主分量分析（PCA）的延伸，他们有如下基本上的不同：PCA 仅仅强制到至多为二阶独立的，而且向量的方向限制为正交的，而 ICA 对于输出向量 $\mathbf{Y}$ 的所有单个分量限制为统计独立，并且没有正交性的限制。

基本假设

为了简化主分量分析的研究，我们做下面的四个基本假设：

1. 统计独立性。构成源向量 $\mathbf{S}$ 的潜在变量假设为统计独立的。然而，注意由于观测向量 $\mathbf{X}$ 是由潜在变量的线性组合组成的，因此观测向量 $\mathbf{X}$ 的各个分量是统计相关的。

2. 混合矩阵的维数。混合矩阵是方阵，这意味着观测数和源数相同。

3. 无噪模型。假设生成模型为无噪的，这意味着在模型中仅有的随机源是源向量 $\mathbf{S}$。

4. 零均值。假设源向量 $\mathbf{S}$ 具有 0 均值，这意味着观测向量 $\mathbf{X}$ 也具有 0 均值。如果不是，则从 $\mathbf{X}$ 中减去均值向量 $\mathbf{E}[\mathbf{X}]$ 以使得其假设为 0 均值。

有时候另一个假设也是需要的：

5. 白噪化。假设观测向量 $\mathbf{X}$ 被“白噪化”。这意味着其各个分量是不相关的，但不是必须独立的。白噪化是通过对观测向量的线性变换使得相关矩阵 $\mathbf{E}[\mathbf{X}\mathbf{X}^T]$ 等于单位矩阵来完成的。

认识到解 BSS 问题除了对于每个源输出（即潜在变量）的估计的任意拉伸和置换之外是可行的这一点也是重要的。为了详细说明，可能找到一个分离矩阵 $\mathbf{W}$，其各行是混合矩阵 $\mathbf{A}$ 的重新拉伸和置换。换句话说，通过 ICA 算法得到的 BSS 问题的解可以表示为下面的形式：

$$\mathbf{y} = \mathbf{Wx} = \mathbf{WAs} = \mathbf{DPs}$$

其中 $\mathbf{D}$ 是一个非奇异对角矩阵，$\mathbf{P}$ 是置换矩阵；$\mathbf{s}$、$\mathbf{x}$ 和 $\mathbf{y}$ 是相应的随机向量 $\mathbf{S}$、$\mathbf{X}$ 和 $\mathbf{Y}$ 的实现。

源的非高斯性：可能除了一个源外，这对 ICA 是必然要求。

为了 ICA 算法能够尽可能地在分离器输出端分离给定的源信号集合，需要对于由生成模型的输出产生的观测向量 $\mathbf{X}$ 的充分信息。这一关键问题如下所述：

观测向量 $\mathbf{X}$ 中的信息内容是如何证明其对于分离源信号是可行的？

我们将通过一个简单但有洞察力的例子来回答这个基本问题。

例 9　一对独立源的两个不同特性

考虑包含一对独立随机源信号 S_1 和 S_2 的生成模型，这两者都具有 0-均值和单位方差。混合矩阵由下面的非奇异矩阵定义：

$$A = \begin{bmatrix} 1 & -1 \\ 1 & 2 \end{bmatrix}$$

该例包含两个部分：在第一部分，两个源都是高斯分布；在第二部分，一个源是高斯分布，另一个是一致分布。

由概率理论，我们知道下面的高斯分布的两个性质（Bertsekas and Tsitsiklis，2002）：

1. 0-均值高斯随机变量的高阶矩都是均等的且由方差唯一定义（即对于 0-均值特例的二阶矩）。

2. 两个线性拉伸（加权）高斯随机变量也是高斯型。

因而就有当两个源信号 S_1 和 S_2 都是 0-均值高斯型时，观测 X_1 和 X_2 也是 0-均值的高斯型。而且，对于规定的混合矩阵，X_1 具有方差

$$(1)^2\sigma_1^2 + (-1)^2\sigma_2^2 = 17,$$

且 X_2 具有方差

$$(1)^2\sigma_1^2 + (2)^2\sigma_2^2 = 65,$$

这里，$\sigma_1^2 = 1$，$\sigma_2^2 = 16$。

图 10.10a 画出了源信号 S_1 和 S_2 的直方图，而 b 画出了观测 X_1 和 X_2 的相应的二维分布。检查图 10.10b，我们发现二维分布是关于原点对称的，其信息内容对于在原始源信号 S_1 和 S_2 的各个方向之间区分是不充分的。

下面考虑源 S_1 是 0-均值和单位方差的高斯分布，源 S_2 是区间 $[-2, 2]$ 上的一致分布。图 10.11a 画出了 S_1 和 S_2 的直方图，b 画出了相应的观测 X_1 和 X_2 的二维分布。和第一种情形的图 10.10b 相似，图 10.11b 的二维分布关于原点对称。然而，对于图 10.11b 分布的深入检查揭示了两个特点：

1. 高斯分布源信号 S_1（无限支持），沿着斜率为 1 的正向显示。

2. 均匀分布源信号 S_2（无限支持），沿着斜率为 -2 的负方向显示。

此外，这两个斜率与混合矩形的元素值相关。

由第二种情况得到的结论是，观测 X_1，X_2 的二维分布包含了足够的方向信息，这些信息是跟源信号 S_1、S_2 是线性可分有关的。这个非常理想的条件，只在允许单个源信号有高斯分布时才出现。 ■

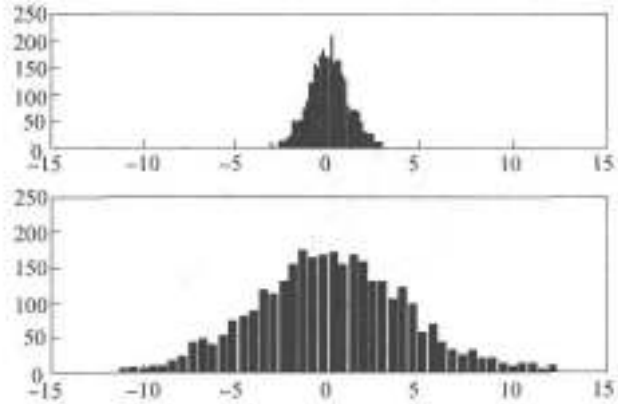
以这个例子的结果为基础，现在可以继续回答我们提出的基础问题，源信号在分离器输出的可行的可分性。

1. 观测 $X_1, X_2, \cdots, X_m$ 必须具有和相应的二阶矩不相关的高阶矩。相应地，源信号 S_1，$S_2, \cdots, S_m$ 必须是非高斯的。

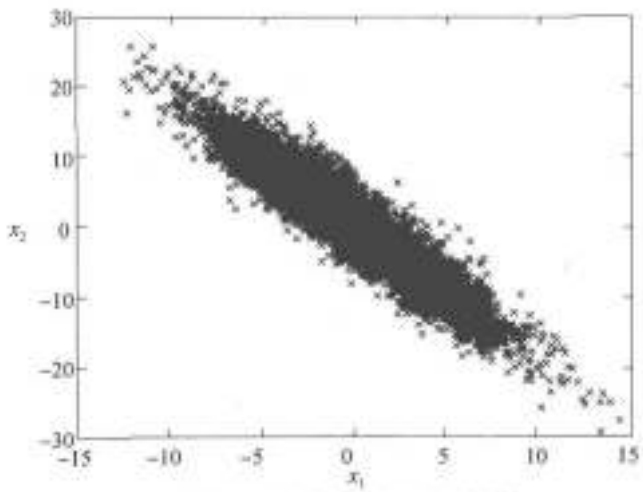
2. 仅有一个源被允许具有高斯分布。

作为小结，源分离的必要条件是源是非高斯的，混合矩阵是非奇异的，生成模型必须满足这两个条件。特别地，我们可以有如下声明（Cardoso，2003）：

独立分量分析（ICA）是随机向量分解为尽可能统计独立的线性分量，这里术语“独立”理解为强烈的统计意义；ICA 超出（二阶）了去相关因此需要表示数据向量的观测是非高斯的。

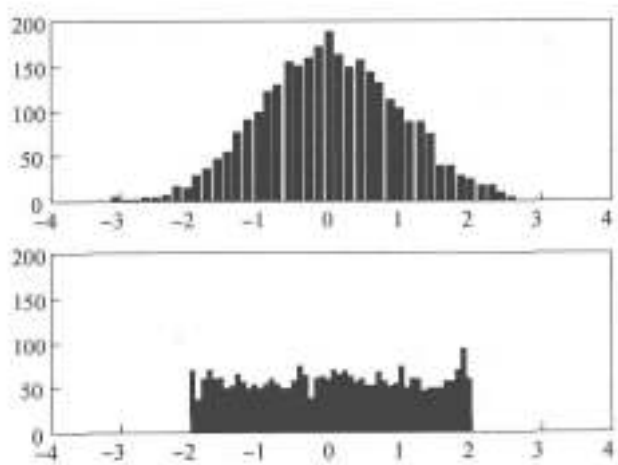


a）两个过程的直方图：上面的直方图表示0-均值和方差$\sigma_1^2=1$的高斯源信号S_1；下面的直方图表示0-均值和方差$\sigma_2^2=16$的高斯源信号S_2



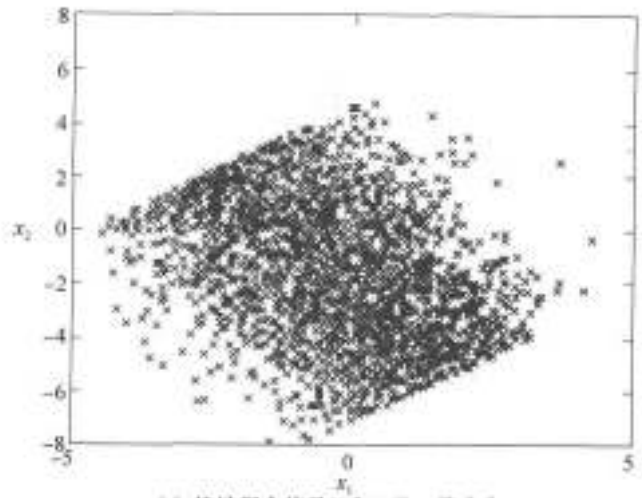
b）线性混合信号X_1和X_2的二维分布

图 10.10　两个高斯分布过程



a）两个过程的直方图：上面的直方图表示0-均值和方差σ^2的高斯源信号S_1；下面的直方图表示在区间[-2,2]上一致分布的源信号S_2

图 10.11　高斯和一致分布过程



b) 线性混合信号X_1和X_2的二维分布

图 10.11　(续)

ICA 算法的分类

现在我们建立了线性混合源信号分离的必要条件，我们可以继续给出两个概括定义的 ICA 算法家族：

1. 根植于最小化互信息的 ICA 算法

最小化图 10.9 的框图中分离器输出之间的互信息为 ICA 算法的设计提供了一个自然的基础。ICA 算法的第一个家族包含如下：

1.1　由 Amari 等 (1996) 提出的算法，这一算法基于相对熵。这一算法在第 10.14 中讲述。

1.2　由 Pham 等 (1992) 提出的算法，这一算法基于最大似然估计。这一算法归于贝叶斯理论的边缘，其忽略了先验信息。这将在 10.15 节讨论。

1.3　由 Bell and Sejnowski(1995)提出的最大互信息 (Infomax) 算法，是基于最大熵原则。这一算法在 10.16 节讲述。在 Cardos(1997) 中，证明了 Infomax 算法和最大似然估计算法等价。

实际上，尽管这些 ICA 算法的形式不同，但它们都是最小化互信息的基本变形。

2. 根植于最大化非高斯性的 ICA 算法

算法的第二家族包括 fastICA 算法 (Hyvarinen and Oja，1997)，它利用负熵作为非高斯型的测量。而且，这一算法不仅代表了它这一类，而且和其他 ICA 算法相比计算速度更快。fastICA 在 10.18 节讨论。

在讨论前述的 ICA 算法之前，我们下面通过考虑自然图像来探索 ICA 的信号处理能力。

10.13　自然图像的稀疏编码以及与 ICA 编码的比较

在第 8 章，我们强调了自然图像高阶统计的重要性以及那些统计量对图像模型化的影响。在本节中，我们强调自然图像的另一个重要特性（名为稀疏）以及捕捉它的 ICA 的角色。在这样做的时候，我们给出了 ICA 在实际应用中的重要性。

10.9 节讨论了如何将最小冗余准则应用于模型化视觉系统 (Atick and Redlich，1990)。在 Dong and Atick(1995)以及 Dan 等 (1996) 中，这一原则的应用延伸到去看视觉系统中视网膜神经节细胞的性质是如何通过白噪化或者去相关由这些细胞根据自然图像的 1/f 振幅功率谱产生的输出集来解

释的。随后，Olshausen and Field(1997) 指出 Atick 和合作者研究的模型的基本局限：那里所考虑的减少冗余局限于自然图像中的像素中的线性两两相关；这些相关可由 PCA 捕获。然而，实际上，自然图像由于以线和边为方向展示了高阶相关（尤其弯曲的变种）在自然图像中是普遍存在的。

在 Olshausen and Field(1997) 中描述了一个概率模型用于捕捉自然图像中的高阶相关结构。更重要的是，这一模型是用基函数的线性重叠来描述的，如下所示：

$$I(\mathbf{x}) = \sum_i a_i \phi_i(\mathbf{x}) \tag{10.74}$$

其中向量 $\mathbf{x}$ 记二维图像 $I(\mathbf{x})$ 中的离散空间位置，$\phi_i(\mathbf{x})$ 记基函数，a_i 记混合振幅。A_i 的计算值构成了编码方案的输出。而且，基函数被选择为自适应的，是为了说明以可能的最佳方式下统计独立事件收集的观点下图像的内在结构。因此，建立在 Field(1994) 的工作基础之上，Olshausen and Field(1997) 作了如下的推测：

稀疏是式(10.74)中混合振幅 a_i 的合适的先验，式(10.74)是基于这样的直觉：自然图像可以通过相关小数目的结构单元来描述，这样的结构单元由边、线以及其他基本特征来例证。

为了验证这一推测，Olshausen 和 Field 实现了下面的两个任务：

1. 构成*稀疏编码算法*，目的是最大化根植于图像处理和信息论的稀疏。这一算法设计用来学习图像模型的基函数集合，基于式(10.74)的图像模型将最好地用稀疏、统计独立分量的方式说明自然图像。已经证明了稀疏编码算法最小化和 ICA 同样的目标函数，但是由于过完备表示引入的难解性需要做一个逼近。

2. 生成数据，从 10 个 512×512 像素的自然环境（树木、岩石、山脉等）图像中取得；这些数据用于训练算法。

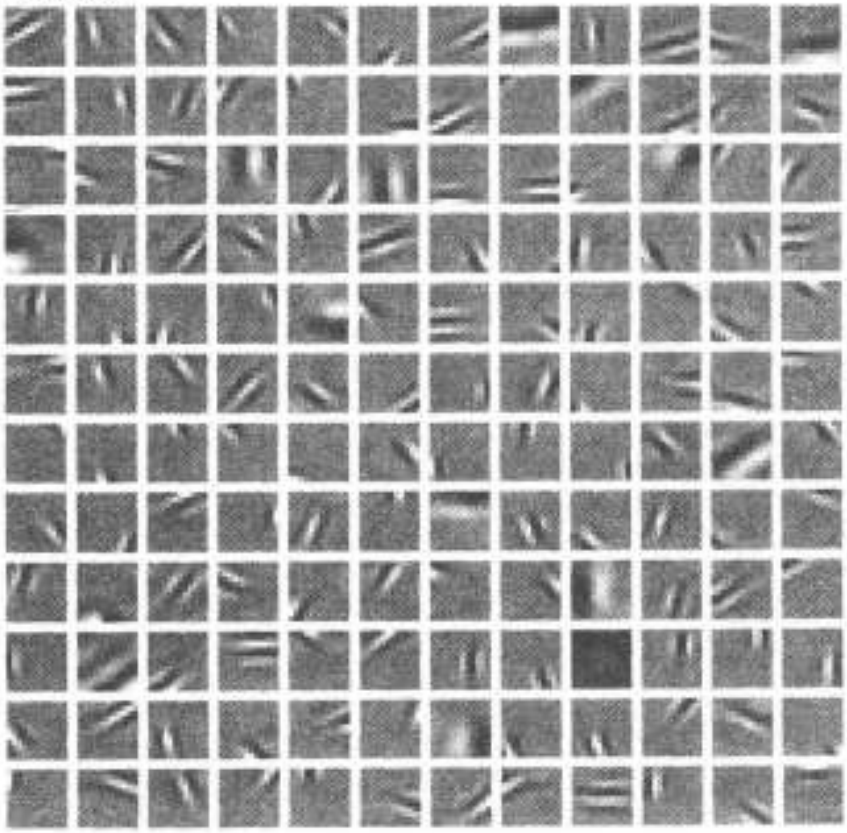


图 10.12　对自然图像应用稀疏编码算法的结果（这个图的复制得到了 Bruno Olshausen 博士的允许）

由稀疏编码算法计算得到的一个稳定解通常在大概 2 000 次更新（即大概 20 000 次图像表示）后获得。训练过程的结果在图 10.12 中给出，其中基函数的大多数被局限在各个像素中。

在一个独立的研究中，Bell and Sejnowski(1997) 将 ICA 应用到包含树木、树叶等的四个自然场景中，它们被转换为灰值图像，其值在 0 到 255 的范围内。将在 10.16 节介绍的 ICA 的 Infomax 算法，在这一研究中被使用。其结果在图 10.13 中给出。

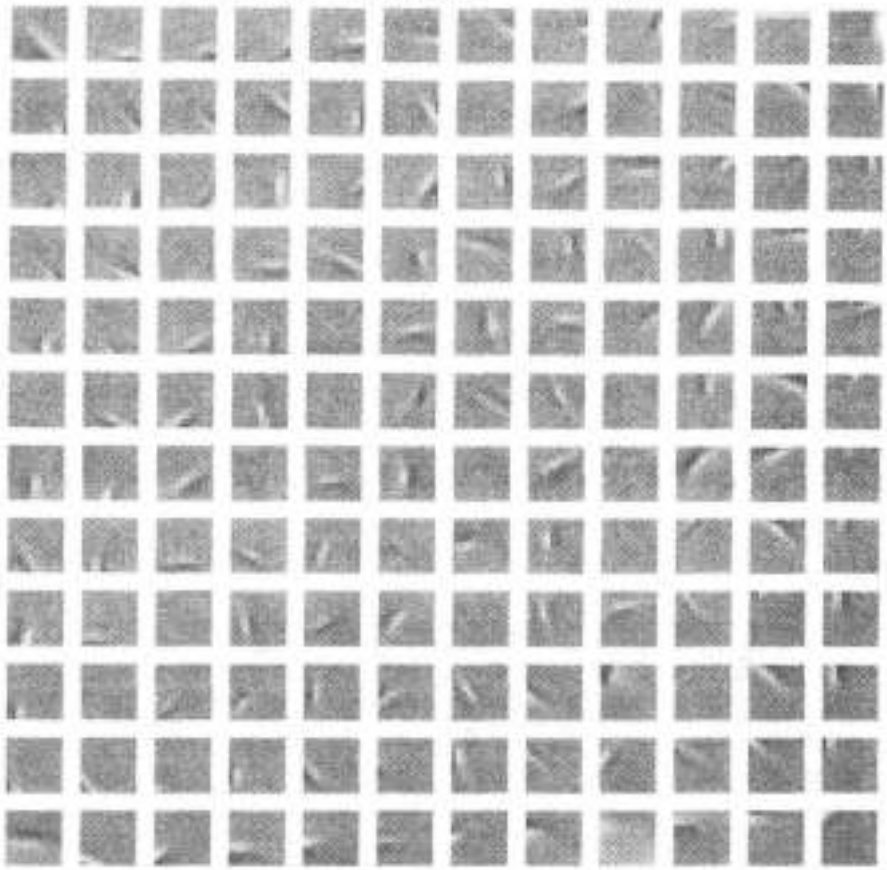


图 10.13　对另一个自然图像应用 ICA 的 Infomax 算法的结果（这个图的复制得到了 Anthony Bell 博士的允许）

比较图 10.12 中利用稀疏编码算法的解和图 10.13 中利用 ICA 的 Infomax 算法的解，值得关注的是这两个解有多么相似。当我们认识到完全不同的自然图像被用于独立地训练这两个算法时相似性是所有更值得注意的。

这两个完全独立的研究告诉我们下面两个重要的教训：

1. 自然图像是内在稀疏的，它们可以通过相关的小数目的不同结构单元来描述，其例子包括边和线。

2. 最基本的是，独立分量分析的算法具有捕捉这些结构单元的内在能力。

因此，图 10.12 和图 10.13 的结果给了我们研究 ICA 学习算法的动机，我们将在下面的四个小节里面实现这一点。

10.14　独立分量分析的自然梯度学习

考虑输入-输出关系

$$\mathbf{Y} = \mathbf{W}\mathbf{X} \tag{10.75}$$

这里随机向量 $\mathbf{X}$ 记观测（即分离器输入），$\mathbf{W}$ 记分离矩阵，随机变量 $\mathbf{Y}$ 记结果响应（即分离器

输出)。将输出 $\mathbf{Y}$ 的各个分量中的统计独立作为盲源分离的期望性质，我们能采用什么样的实际测量来实现该性质呢？为了对这一基础问题的回答作准备，令 $p_{\mathbf{Y}}(\mathbf{y},\mathbf{W})$ 记输出 $\mathbf{Y}$ 的概率密度函数，其参数为分离矩阵 $\mathbf{W}$，令相应的析因分布定义为

$$\tilde{p}_{\mathbf{Y}}(\mathbf{y}) = \prod_{i=1}^{m} \tilde{p}_{Y_i}(y_i) \tag{10.76}$$

其中 $\tilde{p}_{Y_i}(y_i)$ 是随机变量 Y_i(即 $\mathbf{Y}$ 的第 i 个分量) 的边缘概率密度函数；基于明显的理由，析因分布 $\tilde{p}_{\mathbf{Y}}(y)$ 是非参数的。实际上，式(10.76)可以看成学习规则 (将要说明的) 的约束，迫使其将 $p_{\mathbf{Y}}(\mathbf{y},\mathbf{W})$ 和析因分布 $\tilde{p}_{\mathbf{Y}}(\mathbf{y})$ 对比，理想情况下，它将和原始源相匹配。在我们的配置下，集中于作为仅有的两个分布的分布 $p_{\mathbf{Y}}(\mathbf{y},\mathbf{W})$ 和 $\tilde{p}_{\mathbf{Y}}(\mathbf{y})$，我们现在可以给出我们问题的答案，这隐含在 ICA 的原则里：

给定一个 $m\times 1$ 的随机向量 $\mathbf{X}$ 来表示 m 个独立信号源的线性组合，通过这样的方法将观测向量 $\mathbf{X}$ 转换到新的随机向量 $\mathbf{Y}$：对未知参数矩阵 $\mathbf{W}$ 最小化参数概率密度函数 $p_{\mathbf{Y}}(\mathbf{y},\mathbf{W})$ 和相应的析因分布 $\tilde{p}_{\mathbf{Y}}(\mathbf{y})$ 之间的相对熵。

从这一声明中可知，很清楚相对熵是期望反差函数的自然基，其形成构成了 ICA 学习算法推导的最开始一步。作为 ICA 的未知参数的分离矩阵 $\mathbf{W}$，期望反差函数是 $\mathbf{W}$ 的函数。从现在开始，我们用 $R(\mathbf{W})$ 来记反差函数，根据式(10.39)第一行的相对熵，现在可以给出 $R(\mathbf{W})$ 的正式定义：

$$R(\mathbf{W}) = \int_{-\infty}^{\infty} p_{\mathbf{Y}}(\mathbf{y},\mathbf{W}) \log\left[\frac{p_{\mathbf{Y}}(\mathbf{y},\mathbf{W})}{\prod_{i=1}^{m} \tilde{p}_{Y_i}(y_i)}\right] \mathrm{d}\mathbf{y} \tag{10.77}$$

关于这一公式真正值得注意的是：它作为令人鼓舞的框架被用于推导 ICA 和盲源分离相关文献中提案的多个学习算法 (Cichocki and Amari, 2002)。

根据 10.5 节对于相对熵的讨论，我们可以以我们配置的两个熵的方式重新构造期望反差函数 $R(\mathbf{W})$，如下所示：

$$R(\mathbf{W}) = -h(\mathbf{Y}) + \sum_{i=1}^{m} \tilde{h}(Y_i) \tag{10.78}$$

这里 $h(\mathbf{Y})$ 是分离器输出端即向量 $\mathbf{Y}$ 的熵，$\tilde{h}(Y_i)$ 是 $\mathbf{Y}$ 的第 i 个元素的边缘熵。$R(\mathbf{W})$ 是用于对 $\mathbf{W}$ 最小化的目标函数。

微分熵 h(Y)的确定

输出向量 $\mathbf{Y}$ 与式(10.75)和输入向量 $\mathbf{X}$ 相关，这里 $\mathbf{W}$ 是分离矩阵。根据式(10.18)，可以将 $\mathbf{Y}$ 的微分熵表示如下：

$$h(\mathbf{Y}) = h(\mathbf{WX}) = h(\mathbf{X}) + \log|\det(\mathbf{W})| \tag{10.79}$$

这里 $h(\mathbf{X})$ 是 $\mathbf{X}$ 的微分熵，$\det(\mathbf{W})$ 是 $\mathbf{W}$ 的行列式。将这一表达式用于式(10.77)，可以再一次重新构造期望反差函数：

$$\begin{aligned} R(\mathbf{W}) &= -h(\mathbf{X}) - \log|\det(\mathbf{W})| + \sum_{i=1}^{m} \tilde{h}(Y_i) \\ &= -h(\mathbf{X}) - \log|\det(\mathbf{W})| - \sum_{i=1}^{m} \mathbb{E}[\log \tilde{p}_{Y_i}(y_i)] \end{aligned} \tag{10.80}$$

这里，对于方程第二行最右端项，利用式 (10.10) 以及对 Y_i 的期望。注意熵 $h(\mathbf{X})$ 是独立于分离矩阵 $\mathbf{W}$ 的；从此以后，在推导 ICA 的学习算法时我们忽略这一项。

ICA 随机梯度算法的推导

带着随机梯度下降的思想，实际上通常的做法是忽略期望算子 $\mathbb{E}$ 并仅仅集中注意力于瞬时

值。对于手头的问题，仅有一个需要考虑的瞬时值，即 $\tilde{p}_{Y_i}(y_i)$。令 $\rho(\mathbf{W})$ 记期望反差函数 $R(\mathbf{W})$，从此之后我们将之简单称为*反差函数* (contrast function)，即

$$R(\mathbf{W}) = \mathbb{E}[\rho(\mathbf{W})]$$

因此，忽略熵 $h(\mathbf{X})$，我们可以利用式(10.80)来写：

$$\rho(\mathbf{W}) = -\log|\det(\mathbf{W})| - \sum_{i=1}^{m}\log\tilde{p}_{Y_i}(y_i) \tag{10.81}$$

随机梯度矩阵定义为：

$$\nabla_\rho(\mathbf{W}) = -\frac{\partial}{\partial\mathbf{W}}\log|\det(\mathbf{W})| - \frac{\partial}{\partial\mathbf{W}}\sum_{i=1}^{m}\log\tilde{p}_{Y_i}(y_i) \tag{10.82}$$

其中∇为对分离矩阵 $\mathbf{W}$ 的梯度算子。这一梯度矩阵的两个部分被分别考虑：

1. 第一个部分定义为

$$\frac{\partial}{\partial\mathbf{W}}\log|\det(\mathbf{W})| = \mathbf{W}^{-T} \tag{10.83}$$

其中 $\mathbf{W}^{-T}$ 是逆矩阵 $\mathbf{W}^{-1}$ 的转置。

2. 随机梯度矩阵的第二个部分的第 i 个分量定义为

$$\frac{\partial}{\partial\mathbf{w}_i}\log\tilde{p}_{Y_i}(y_i) = \frac{\partial y_i}{\partial\mathbf{w}_i}\frac{\partial}{\partial y_i}\log\tilde{p}_{Y_i}(y_i) \tag{10.84}$$

其中 $\mathbf{w}_i$ 是分离矩阵 $\mathbf{W}$ 的第 i 个列向量，y_i 是输出向量 Y_i 的样本值。因此，取式(10.75)第 i 个分量的样本值，我们有

$$y_i = \mathbf{w}_i^T\mathbf{x}, i = 1,2,\cdots,m \tag{10.85}$$

其中 $\mathbf{x}$ 是输入向量 $\mathbf{X}$ 的样本值，y_i 是 Y_i 的样本值。对 $\mathbf{w}_i$ 微分式(10.85)，得到：

$$\frac{\partial y_i}{\partial\mathbf{w}_i} = \mathbf{x} \tag{10.86}$$

而且，

$$\frac{\partial}{\partial y_i}\log\tilde{p}_{Y_i}(y_i) = \frac{1}{\tilde{p}_{Y_i}(y_i)}\frac{\partial}{\partial y_i}\tilde{p}_{Y_i}(y_i) = \frac{\tilde{p}'_{Y_i}(y_i)}{\tilde{p}_{Y_i}(y_i)} \tag{10.87}$$

其中偏导数

$$\tilde{p}'_{Y_i}(y_i) = \frac{\partial}{\partial y_i}\tilde{p}_{Y_i}(y_i)$$

在所讨论的这一点，我们发现为构造分离器而引入*激活函数* φ 是便利的；具体来说，我们定义

$$\varphi_i(y_i) = -\frac{\tilde{p}'_{Y_i}(y_i)}{\tilde{p}_{Y_i}(y_i)}, i = 1,2,\cdots,m \tag{10.88}$$

相应地，将式(10.85)代入式(10.88)，得到：

$$\frac{\partial}{\partial\mathbf{w}_i}\log\tilde{p}_{Y_i}(y_i) = -\varphi_i(y_i)\mathbf{x}, i = 1,2,\cdots,m \tag{10.89}$$

由这一表达式，我们可以将式(10.82)中随机梯度矩阵的和项部分表示为：

$$\frac{\partial}{\partial\mathbf{W}}\sum_{i=1}^{m}\log\tilde{p}_{Y_i}(y_i) = -\boldsymbol{\phi}(\mathbf{y})\mathbf{x}^T = -\mathbf{x}\boldsymbol{\phi}^T(\mathbf{y}) \tag{10.90}$$

其中激活函数向量表示为输出向量 $\mathbf{y}$ 的函数：

$$\boldsymbol{\phi}(\mathbf{y}) = [\varphi_1(y_1),\varphi_2(y_2),\cdots,\varphi_m(y_m)]^T$$

下面将式(10.83)和式(10.90)代入式(10.82)，得到需要的随机梯度矩阵：

$$\nabla\rho(\mathbf{W}) = -\mathbf{W}^{-T} + \boldsymbol{\phi}(\mathbf{y})\mathbf{x}^T \tag{10.91}$$

现在，令 η 记学习率参数，假设为一个小的正常数。然后，给定式(10.91)的梯度矩阵，

对于分离矩阵的增量调整是：

$$\Delta \mathbf{W} = -\eta \nabla \rho(\mathbf{W}) = \eta [\mathbf{W}^{-T} - \boldsymbol{\phi}(\mathbf{y})\mathbf{x}^T] \tag{10.92}$$

由于立刻将变得明显的原因，我们发现通过首先转置式(10.85)来重新构造(10.92)是便利的，这产生

$$\mathbf{y}^T = \mathbf{x}^T \mathbf{W}^T$$

因此，可以重写式(10.92)为下面的新的等价形式：

$$\Delta \mathbf{W} = \eta [\mathbf{I} - \boldsymbol{\phi}(\mathbf{y})\mathbf{x}^T \mathbf{W}^T]\mathbf{W}^{-T} = \eta [\mathbf{I} - \boldsymbol{\phi}(\mathbf{y})\mathbf{y}^T \mathbf{W}^{-T}] \tag{10.93}$$

其中 $\mathbf{I}$ 是单位矩阵。相应地，更新分离矩阵的在线学习规则有如下的形式：

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \eta(n) \underbrace{[\mathbf{I} - \boldsymbol{\phi}(\mathbf{y}(n))\mathbf{y}^T(n)]}_{\text{校正项}} \mathbf{W}^{-T}(n) \tag{10.94}$$

其中参数都用其随时间变化的形式来表示。

这一算法不良的性质是通过对权重矩阵 $\mathbf{W}$ 的转置的逆对调整项的后乘。我们下一个任务是找到一个方法来消去逆的计算。

等变异性质

ICA 算法的目的是更新分离矩阵 $\mathbf{W}(n)$ 使得输出向量

$$\mathbf{y}(n) = \mathbf{W}(n)\mathbf{x}(n) = \mathbf{W}(n)\mathbf{A}\mathbf{s}(n)$$

尽可能地在某种统计意义下和原始源信号 $\mathbf{s}(n)$ 相近。更具体来说，考虑由系统矩阵 $\mathbf{C}(n)$ 刻画的全局系统，$\mathbf{C}(n)$ 是通过将混合矩阵 $\mathbf{A}$ 和分离矩阵 $\mathbf{W}(n)$ 相乘而得到的，即

$$\mathbf{C}(n) = \mathbf{W}(n)\mathbf{A} \tag{10.95}$$

理想情况下，这一全局系统满足两个条件：

1. 调整 $\mathbf{C}(n)$ 的算法收敛到等于交换矩阵的最优值。(注意，一个有符号交换矩阵，在每一行和列仅有一次 $+1$ 或 -1，也是最优的。)

2. 这一算法的自身描述为：

$$\mathbf{C}(n+1) = \mathbf{C}(n) + \eta(n)\mathbf{G}(\mathbf{C}(n)\mathbf{s}(n))\mathbf{C}(n) \tag{10.96}$$

其中 $\mathbf{G}(\mathbf{C}(n)\mathbf{s}(n))$ 是矩阵积 $\mathbf{C}(n)s(n)$ 的矩阵值函数。这一算法的性能是由系统矩阵 $\mathbf{C}(n)$ 完全刻画的，而不是由混合矩阵 $\mathbf{A}$ 以及分离矩阵 $\mathbf{W}(n)$ 的各个值刻画。这样的自适应系统称之为等变异 (equivariant) (Cardoso and Laheld，1996)。

式(10.94)的在线学习算法当然能够近似满足第一个条件。然而，如其所表明的，它不能满足第二个条件。为了说明确实如此，我们用混合矩阵 $\mathbf{A}$ 来乘式(10.94)，然后利用式(10.95)来写：

$$\mathbf{C}(n+1) = \mathbf{C}(n) + \eta(n)\mathbf{G}(\mathbf{C}(n)\mathbf{s}(n))\mathbf{W}^{-T}(n)\mathbf{A} \tag{10.97}$$

其中

$$\mathbf{G}(\mathbf{C}(n)\mathbf{s}(n)) = \mathbf{I} - \boldsymbol{\phi}(\mathbf{C}(n)\mathbf{s}(n))(\mathbf{C}(n)\mathbf{s}(n))^T \tag{10.98}$$

显然，式(10.94)的算法不满足式(10.96)描述的等变异条件，因为矩阵值函数 $\mathbf{G}(\mathbf{C}(n)\mathbf{s}(n))$ 是被 $\mathbf{W}^{-T}(n)\mathbf{A}$ 右乘，这通常是和 $\mathbf{C}(n)$ 不同的。为了校正这一状况，我们在式(10.97)中的函数 $\mathbf{G}(\mathbf{C}(n)\mathbf{s}(n))$ 和矩阵积 $\mathbf{W}^{-T}(n)\mathbf{A}$ 之间插入矩阵积 $\mathbf{W}^{-T}(n)\mathbf{W}(n)$。由矩阵 $\mathbf{W}$ 及其转置的积组成的项 $\mathbf{W}^T\mathbf{W}$ 总是正定的。这就是乘以 $\mathbf{W}^T\mathbf{W}$ 不会改变学习算法极小点的符号的原因。

重要的问题是，这一修正暗示着什么来达到等变异条件？答案在于参数空间的梯度方向是如何形成的。理想情况下，可以利用反差函数 $\rho(\mathbf{W})$ 的自然梯度[12]，由通常的梯度 $\nabla \rho(\mathbf{W})$ 定义为

$$\nabla^* \rho(\mathbf{W}) = (\nabla \rho(\mathbf{W}))\mathbf{W}^T\mathbf{W} \tag{10.99}$$

通常的梯度矩阵由式(10.91)定义。在潜在意义下，梯度 $\nabla \rho(\mathbf{W})$ 仅在参数空间 $\mathcal{W} \in \{\mathbf{W}\}$ 是具有正交坐标系统的欧几里得空间时是下降的最优方向。然而，在包含神经网络的典型状况下，参数空间 $\mathcal{W}$ 的坐标系统不是正交的。在后一种状况下自然梯度 $\nabla^* \rho(\mathbf{W})$ 将提供最速下降——因此优先使用它来替代通常的梯度以构造 ICA 的随机梯度算法。对于要定义的自然梯度空间，必须满足两个条件：

1. 参数空间$\mathcal{W}$是黎曼的（Riemannian）[13]。黎曼结构是可微流形（可微流形的概念在第 7 章已经讨论过了）。

2. 矩阵 $\mathbf{W}$ 是非奇异的（即可逆的）。

对于当前的问题，这两个条件都是满足的。

相应地，现在我们通过刚刚描述的方式来修正式(10.94)的算法，允许我们写

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \eta(n)[\mathbf{I} - \boldsymbol{\phi}(\mathbf{y}(n))\mathbf{y}^T(n)]\mathbf{W}(n)(W^T(n)W^{-T}(n))$$

最后，认识到矩阵乘积 $\mathbf{W}^{-T}(n)\ \mathbf{W}(n)$ 等于单位矩阵，最后写

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \eta(n)[\mathbf{I} - \boldsymbol{\phi}(\mathbf{y}(n))\mathbf{y}^T(n)]\mathbf{W}(n) \tag{10.100}$$

这导致带有期望等变异性质的盲源分离。由于式(10.100)的在线学习算法的推导基于自然梯度，这一算法在文献中通常称之为独立分量分析的自然梯度学习算法（Cichocki and Amari，2002）。很明显，这一算法的一个完整图必须也包括式(10.85)的输入输出关系在整个输出集上的矩阵表示：

$$\mathbf{y} = \{y_i\}_{i=1}^{M} = \mathbf{W}\mathbf{x}$$

算法的这一完整的输入输出图在图 10.14 的信号流图中画出。

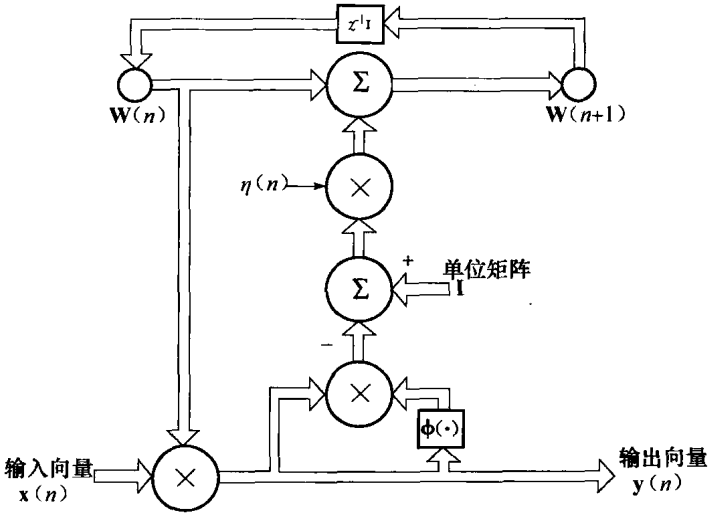


图 10.14　式(10.85)和式(10.104)的盲源分离学习算法的信号流图：标志为 $z^{-1}\mathbf{I}$ 的块表示单位时间延迟单元。这一图包含多个反馈循环

自然梯度学习算法的重要优势

作为拥有等变异性质的补充，在式(10.100)中描述的自然梯度学习算法具有四个重要优势：

1. 这个算法是计算高效的，因为它避免了转化分离矩阵 $\mathbf{W}$。
2. 算法的收敛速率是相对快的。
3. 这个算法的执行以一种自适应神经系统的形式。
4. 作为一个随机梯度算法，这一算法具有追迹不稳定环境的统计变化的内在能力。

ICA 理论的鲁棒性

式(10.100)的自然梯度学习算法需要式(10.88)定义的激活函数 $\varphi(y)$ 的知识，这证明了 $\varphi(y)$ 是依赖于边缘分布 $\tilde{p}_Y(y)$ 的。相应地，为了使这个算法能够为盲源分离问题提供满意的解，边缘分布 $\tilde{p}_Y(y)$ 的任意数学描述必须和原始独立分量（即源）的真正分布相近；否则，就有严重的模型不匹配。

然而，实际上，我们发现仅考虑两个关于每个独立分量的内在概率分布的可能逼近是足够的[14]：

1. **超高斯分布（Super-Gaussian distribution）**。这一分布具有和拉普拉斯分布相似的形式，定义为：

$$p_Y(y) = \frac{\alpha}{2}\exp(-\alpha|y|), -\infty < \alpha < \infty$$

这里绝对值 $|y|$ 以速率 α 指数延迟。例如，语音信号的振幅样本倾向于服从拉普拉斯分布。

2. **亚高斯分布（Sub-Gaussian distribution）**。这第二个分布类似于 log-高斯分布，其在原点附近有点平坦。

在之前关于“逼近”的陈述是 ICA 理论的鲁棒性的证明：

(i) 内在分布的简单模型对于估计独立分量是足够的。

(ii) 在对每个独立分量测试超高斯和亚高斯逼近时的小的模型误差是允许的。

更具体地，ICA 理论的鲁棒性由下面重要的定理所证实（Hyvärinen 等，2001）：

令 $\tilde{p}_{Y_i}(y_i)$ 记由分离器输出 y_i 表示的第 i 个独立分量（源信号）的假设概率密度函数。定义激活函数：

$$\varphi(y_i) = -\frac{\partial}{\partial y_i}\log\tilde{p}_{Y_i}(y_i) = -\frac{\tilde{p}'_{Y_i}(y_i)}{\tilde{p}_{Y_i}(y_i)}, \tilde{p}'_{Y_i}(y_i) = \frac{\partial}{\partial y_i}\tilde{p}_{Y_i}(y_i)$$

假设独立分量 $\{y_i\}_{i=1}^m$ 的估计约束为彼此不相关，且对所有 i 随机变量 Y_i 具有单位方差。则独立分量的自然梯度估计为局部一致的。设假定的分布满足如下条件：

$$\mathbb{E}[y_i\varphi(y_i) - \varphi'(y_i)] > 0, \quad \text{对于所有 } i \tag{10.101}$$

其中

$$\varphi'(y_i) = \frac{\partial}{\partial y_i}\varphi(y_i)$$

这一定理从此之后称为 ICA 鲁棒定理（Hyvärinen 等，2001），它严格地证明了只要不等式条件(10.101)的符号对所有 i 保持不变，在逼近分布 $\tilde{p}_{Y_i}(y_i)$ 中小的差异不影响利用自然梯度学习算法计算的独立分量的估计的局部一致性[15]。

对于自然梯度学习的 ICA 鲁棒定理可以等价地应用于第 10.15 节讲述的最大似然估计过程。而且，ICA 鲁棒定理告诉我们，如何基于式(10.101)的不等式构造函数族，族中的每一对由属于超高斯分布和其亚高斯分布副本的 log-高斯密度函数构成。实际上，我们因此在两个候选分布之间具有一个简单的二位选择。下面的例子解释了这样的一个选择。

例 10　超高斯和亚高斯函数

考虑一对 log -密度函数

$$\log p_Y^+(y) = \alpha_1 - 2\log\cosh(y)$$

$$\log p_Y^-(y) = \alpha_2 - \left(\frac{1}{2}y^2 - \log\cosh(y)\right)$$

其中 α_1 和 α_2 是正常数，用于确认每一个函数满足概率密度函数的基本性质。正和负的上标用于分别强调考虑中的函数参照超高斯或者亚高斯概率密度函数。

将式(10.88)的公式作用于激活函数 $p_Y^+(y)$，得到双曲正切函数：

$$\varphi^+(y) = \tanh(y)$$

这里为了数学上的方便我们忽略了乘积因子 2。将这一结果再次对 y 求微分，得到激活函数的梯度

$$\varphi^{+'} = \operatorname{sech}^2(y)$$

因此，对于超高斯函数，式(10.101)的左边生成以下的结果（不考虑伸缩因子 2）

$$\mathbb{E}[y\tanh(y) - \operatorname{sech}^2(y)]$$

对于 $p_Y^-(y)$ 进行同样的两个操作，得到

$$\varphi^{-}(y) = y - \tanh(y)$$
$$\varphi^{-\prime}(y) = 1 - \operatorname{sech}^2(y)$$

因此，对于亚高斯函数，式(10.101)的左边产生

$$\mathbb{E}[y^2 - y\tanh(y) - 1 + \operatorname{sech}^2(y)] = \mathbb{E}[-y\tanh(y) + \operatorname{sech}^2(y)]$$

其中我们调用了 0 -均值随机变量 Y(由样本值 y 表示) 的方差是 1 的假设，即$\mathbb{E}[Y^2]=1$。

检查刚刚获得的超高斯和亚高斯函数的结果，我们发现它们实际上具有对立的代数符号。于是，仅有其中一个满足式(10.101)的不等式；对于 ICA 的数据集满足这一不等式的特别的激活函数是被用于根植于独立分量分析原则的算法类（如自然梯度学习算法）的函数。 ■

10.15　独立分量分析的最大似然估计

前面一节所讨论的独立分量分析的原则只是诸多盲源分离方法中的一种。但在这一原则的背景中，有其他两种方法能够以无监督方式解决源分离问题：最大似然法和最大熵法。在本节中我们讨论最大似然法，在下一节中讨论最大熵法。

最大似然法是一个统计估计的良好建立的过程，具有一些良好的性质[16]。在这个过程中，我们首先建立对数似然函数，然后根据考虑的概率模型的参数向量对它进行最优化。从第 2 章的讨论中，我们知道似然函数是一个给定模型中的数据集的概率密度函数，但只是作为模型未知参数的一个函数。根据图 10.9，令 $p_{\mathbf{S}}(\mathbf{s})$表示样本值是 $\mathbf{s}$ 的随机源向量 $\mathbf{S}$ 的概率密度函数。那么在混合器输出端的观测向量 $\mathbf{X}=\mathbf{AS}$ 的概率密度函数定义为：

$$p_{\mathbf{X}}(\mathbf{x},\mathbf{A}) = |\det(\mathbf{A})|^{-1} p_{\mathbf{S}}(A^{-1}\mathbf{x}) \tag{10.102}$$

其中 $\det(\mathbf{A})$是混合矩阵 $\mathbf{A}$ 的行列式。令$\mathcal{T}=\{\mathbf{x}_k\}_{k=1}^{N}$表示随机向量 $\mathbf{X}$ 的 N 次独立实现组成的训练样本。于是可以写成

$$p_{\mathbf{X}}(\mathcal{T},\mathbf{A}) = \prod_{k=1}^{N} p_{\mathbf{X}}(\mathbf{x}_k,\mathbf{A}) \tag{10.103}$$

我们发现用归一化（除以样本数目 N）后的对数似然函数更方便，表示为

$$\frac{1}{N}\log p_{\mathbf{X}}(\mathcal{T},\mathbf{A}) = \frac{1}{N}\sum_{k=1}^{N}\log p_{\mathbf{X}}(\mathbf{x}_k,\mathbf{A}) = \frac{1}{N}\sum_{k=1}^{N}\log p_{\mathbf{S}}(\mathbf{A}^{-1}\mathbf{x}_k) - \log|\det(\mathbf{A})|$$

令 $\mathbf{y}=\mathbf{A}^{-1}\mathbf{x}$ 为分离器输出端的随机向量 $\mathbf{Y}$ 的一个实现，这样可写成

$$\frac{1}{N}\log p_{\mathbf{X}}(\mathcal{T},\mathbf{A}) = \frac{1}{N}\sum_{k=1}^{N}\log p_{\mathbf{S}}(\mathbf{y}_k) - \log|\det(\mathbf{A})| \tag{10.104}$$

令 $\mathbf{A}^{-1}=\mathbf{W}$ 且 $p_{\mathbf{Y}}(\mathbf{y},\mathbf{W})$表示以 $\mathbf{W}$ 为参数的 $\mathbf{Y}$ 的概率密度函数。注意式(10.104)中的求和是$\log p_{\mathbf{S}}(\mathbf{y}_k)$的样本平均值。从大数定律发现，当 N 趋于无穷时，

$$\begin{aligned} L(\mathbf{W}) &= \lim_{N\to\infty}\frac{1}{N}\sum_{k=1}^{N}\log p_{\mathbf{S}}(\mathbf{y}_k) + \log|\det(\mathbf{W})| = \mathbb{E}[\log p_{\mathbf{S}}(\mathbf{y})] + \log|\det(\mathbf{W})| \\ &= \int_{-\infty}^{\infty} p_{\mathbf{Y}}(\mathbf{y},\mathbf{W})\log p_{\mathbf{S}}(\mathbf{y})\,\mathrm{d}\mathbf{y} + \log|\det(\mathbf{W})| \end{aligned} \tag{10.105}$$

以概率 1 成立，其中第二等式是关于 $\mathbf{Y}$ 的求期望。函数 $L(\mathbf{W})$是希望的对数似然函数。通过以下公式

$$p_{\mathbf{S}}(\mathbf{y}) = \left(\frac{p_{\mathbf{S}}(\mathbf{y})}{p_{\mathbf{Y}}(\mathbf{y},\mathbf{W})}\right) p_{\mathbf{Y}}(\mathbf{y},\mathbf{W})$$

我们可以将 $L(\mathbf{W})$表示为等价形式：

$$\begin{aligned} L(\mathbf{W}) &= \int_{-\infty}^{\infty} p_{\mathbf{Y}}(\mathbf{y},\mathbf{W})\log\left(\frac{p_{\mathbf{S}}(\mathbf{y})}{p_{\mathbf{Y}}(\mathbf{y},\mathbf{W})}\right)\mathrm{d}\mathbf{y} + \int_{-\infty}^{\infty} p_{\mathbf{Y}}(\mathbf{y},\mathbf{W})\log p_{\mathbf{Y}}(\mathbf{y},\mathbf{W})\,\mathrm{d}\mathbf{y} + \log|\det(\mathbf{W})| \\ &= -R(\mathbf{W}) - h(\mathbf{Y},\mathbf{W}) + \log|\det(\mathbf{W})| \end{aligned} \tag{10.106}$$

其中我们运用了如下的定义：

- 和相对熵有相同公式的期望反差函数 $R(\mathbf{W})$，如式(10.77)所定义。
- 如式(10.12)第一行所定义的微分熵 $h(\mathbf{Y}, \mathbf{w})$。

下面，利用式(10.78)，最后重写式(10.79)为所期望的形式

$$L(\mathbf{W}) = -R(\mathbf{W}) - h(\mathbf{X}) \tag{10.107}$$

其中 $h(\mathbf{X})$是分离器输入端的随机向量 $\mathbf{X}$ 的微分熵 (Cardoso, 1998a)。在式(10.107)中，唯一依赖于分离器的权值向量 $\mathbf{W}$ 的是期望反差函数 $R(\mathbf{W})$。因此从式(10.107)可以得到如下结论：最大化对数似然函数 $L(\mathbf{W})$就等于最小化 $R(\mathbf{W})$，即使分离器的输出 $\mathbf{Y}$ 的概率分布与初始源向量 $\mathbf{S}$ 的概率分布匹配。

最大似然估计与独立分量分析原则之间的关系

对目前问题应用式(10.43)所描述的 Pythagoream 分解，可以将期望反差函数表示为极大似然

$$R(\mathbf{W}) = D_{p_{\mathbf{Y}} \| \tilde{p}_{\mathbf{Y}}} + D_{\tilde{p}_{\mathbf{Y}} \| p_{\mathbf{S}}} \tag{10.108}$$

式(10.108)右边的第一个相对熵 $D_{p_{\mathbf{Y}} \| \tilde{p}_{\mathbf{Y}}}$是表征独立分量分析方法的结构失配的度量，第二个相对熵 $D_{\tilde{p}_{\mathbf{Y}} \| p_{\mathbf{S}}}$是描述初始源向量 S 的分布和分离器输出 Y 的边缘分布之间的边缘失配的度量。因此可以将用于最大似然的全局分布匹配准则表达如下：

$$\left(\text{全局失配}\right) = \underbrace{\left(\text{结构失配}\right)}_{D_{p_{\mathbf{Y}} \| \tilde{p}_{\mathbf{Y}}}} + \underbrace{\left(\text{边缘失配}\right)}_{D_{\tilde{p}_{\mathbf{Y}} \| p_{\mathbf{S}}}} \tag{10.109}$$

在所关心的式(10.109)的右边，“结构失配”是指一组独立变量的一个分布的结构，而“边缘失配”是指各边缘分布之间的不匹配。

在理想情况下 $\mathbf{W}=\mathbf{A}^{-1}$（即完全盲源分离），结构失配和边缘失配都消失。在这种情况下，最大似然与独立分量分析产生完全相同的结果，理想情况下的两者的关系描绘在图 10.15 中。

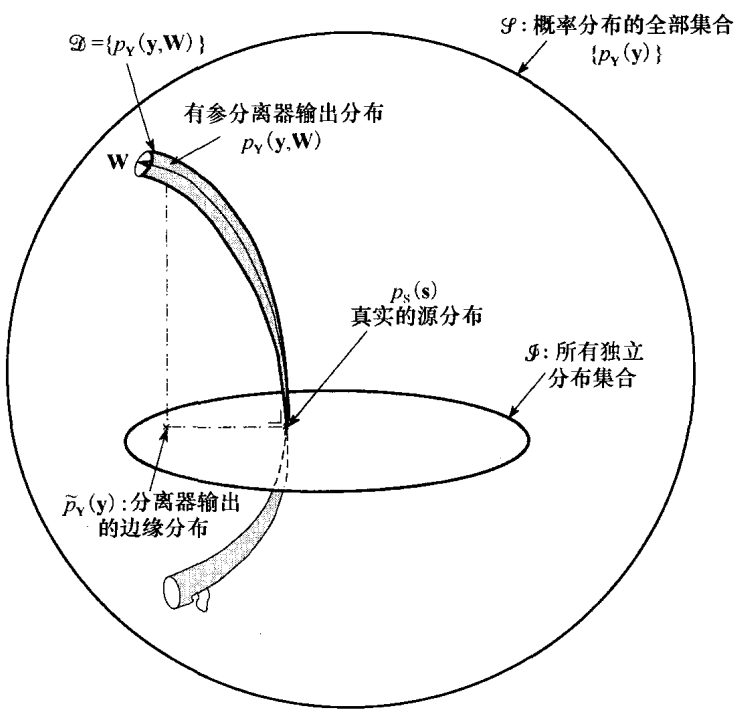


图 10.15　用于盲源分离的最大似然估计与独立分量分析之间的关系示意图。
最大似然最小化 $D_{p_{\mathbf{Y}} \| p_{\mathbf{S}}}$，而独立分量分析最小化 $D_{p_{\mathbf{Y}} \| \tilde{p}_{\mathbf{Y}}}$

在这个图中，$\mathcal{S}$是分离器输出端随机向量 Y 的所有概率密度函数 $p_{\mathbf{Y}}(\mathbf{y})$的集合；$\mathcal{I}$是所有独立概率分布的集合，也就是那些乘积形式。$\mathcal{S}$和$\mathcal{I}$都是无穷维的。集$\mathcal{Q}=\{p_{\mathbf{Y}}(\mathbf{y}, \mathbf{W})\}$是在分离器的输出端测量得到的概率分布的有限集。$\mathcal{Q}$是 m^2 维的，其中 m 表示 Y 的维数，权值向量 W 是其中的一个坐标系。从图 10.15 中，可以清楚看出 $D_{\mathcal{Q}\|\mathcal{I}}$ 和 $D_{\mathcal{Q}\|\mathcal{S}}$ 在 $\mathbf{W}=\mathbf{A}^{-1}$ 时最小化。而且，如图 10.15 所示，集合$\mathcal{Q}$和$\mathcal{I}$在交点处正交，该交点由真实概率密度函数 $p_{\mathbf{S}}(\mathbf{s})$所定义。

对于一个基于最大似然原则的盲源分离算法必须包括对固有的未知源分布的估计，而这些源分布通常就是未知的。这个估计的参数正如调节分离权值矩阵 W 一样是可以调节的。换句话说，我们应该进行混合矩阵和源分布（一些特征）的联合估计（Cardoso，1997，1998）；这种联合估计的一种巧妙和成熟的方法已经在 Pham 等（1992，1997）中给出。

10.16　盲源分离的最大熵学习

在本节中，我们寻求用第 10.3 节讨论过的最大熵原则来作为解决盲源分离问题的另一种方法。考虑图 10.16，它给出了基于这种方法的系统方框图。与以前一样，分离器对观察向量 x 进行操作，产生输出 $\mathbf{y}=\mathbf{W}\mathbf{x}$，它是初始源向量 s 的估计。向量 y 经过每个分量为非线性的变换 $G(\cdot)$变成 z，且 $\mathbf{G}(\cdot)$是一个单调可逆函数。因此，与 y 不同，对一个任意大的分离器 z 的微分熵 $h(\mathbf{Z})$保证都是有界的。对于给定的非线性 $\mathbf{G}(\cdot)$，最大熵方法通过对 W 求 $h(\mathbf{z})$的最大值，得到初始源向量 s 的一个估计。根据在例 7 中导出的式(10.60)，对于无噪声网络，我们回忆到最大熵方法与最大互信息原则是紧密相关的。实际上，这是由于基于图 10.16 的方案的算法在文献中被称为 ICA 的 Infomax 算法（Bell and Sejnowski，1995）。

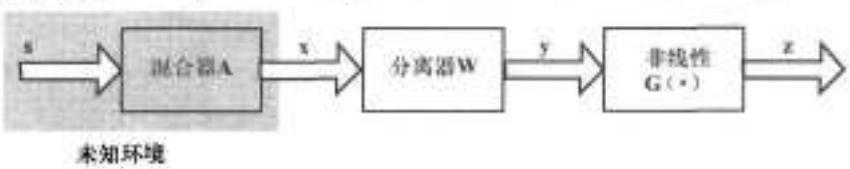


图 10.16　用于盲源分离的最大熵原则方框图。向量 s，x，y 和 z 分别是随机向量 S，X，Y 和 Z 的样本值

非线性 G 是一个对角映像，表达为

$$\mathbf{G}:\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \rightarrow \begin{bmatrix} g_1(y_1) \\ g_2(y_2) \\ \vdots \\ g_m(y_m) \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_m \end{bmatrix} \tag{10.110}$$

也可以写成

$$\mathbf{z}=\mathbf{G}(\mathbf{y})=\mathbf{G}(\mathbf{W}\mathbf{A}\mathbf{s}) \tag{10.111}$$

由于非线性 $\mathbf{G}(\cdot)$是可逆的，可以将初始源向量 s 利用分离器输出向量 z 表示成

$$\mathbf{s}=\mathbf{A}^{-1}\mathbf{W}^{-1}\mathbf{G}^{-1}(\mathbf{z})=\boldsymbol{\psi}(\mathbf{z}) \tag{10.112}$$

其中 $\mathbf{G}^{-1}$是一个非线性的逆：

$$\mathbf{G}^{-1}:\begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_m \end{bmatrix} \rightarrow \begin{bmatrix} g_1^{-1}(z_1) \\ g_2^{-1}(z_2) \\ \vdots \\ g_m^{-1}(z_m) \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \tag{10.113}$$

输出向量 z 的概率密度函数利用源向量 s 的概率密度函数定义为：

$$p_{\mathbf{Z}}(\mathbf{z}) = \frac{p_{\mathbf{S}}(\mathbf{s})}{|\det(\mathbf{J}(\mathbf{s}))|}\Big|_{\mathbf{s}=\boldsymbol{\psi}(\mathbf{z})} \tag{10.114}$$

其中 $\det(\mathbf{J}(\mathbf{s}))$是 Jacobi 矩阵 $\mathbf{J}(\mathbf{s})$的行列式（Papoulis，1984）。$\mathbf{J}(\mathbf{s})$的第 ij 元素定义如下：

$$J_{ij} = \frac{\partial z_i}{\partial s_j} \tag{10.115}$$

所以非线性 $\mathbf{G}$ 的输出端的随机向量 $\mathbf{Z}$ 的熵为

$$\begin{aligned} h(\mathbf{Z}) &= -\mathbb{E}[\log p_{\mathbf{Z}}(\mathbf{z})] \\ &= -\mathbb{E}\left[\log\left(\frac{p_{\mathbf{S}}(\mathbf{s})}{|\det(\mathbf{J}(\mathbf{s}))|}\right)\right]_{\mathbf{s}=\boldsymbol{\psi}(\mathbf{z})} \\ &= -D_{p_{\mathbf{S}}\|\,|\det \mathbf{J}|} \quad \text{在 } \mathbf{s}=\boldsymbol{\psi}(\mathbf{z}) \text{ 估值} \end{aligned} \tag{10.116}$$

因此可以看出最大化微分熵 $h(\mathbf{Z})$等价于最小化 $p_{\mathbf{S}}(\mathbf{s})$和由 $|\det(\mathbf{J}(\mathbf{s}))|$定义的 $\mathbf{s}$ 的概率密度函数之间的相对熵；参见式(10.35)的最后一行。

假设对所有的 i，随机变量 $\mathbf{Z}_i$(即 $\mathbf{Z}$ 的第 i 个元素）在 [0，1] 上均匀分布。根据例 1，那么熵 $h(\mathbf{Z})$为 0。相应地，从式（10.116）得到

$$p_{\mathbf{S}}(\mathbf{s}) = |\det(\mathbf{J}(\mathbf{s}))| \tag{10.117}$$

在理想情况 $\mathbf{W}=\mathbf{A}^{-1}$时，这种关系化简为

$$p_{S_i}(s_i) = \frac{\partial z_i}{\partial y_i}\Big|_{z_i=g(s_i)}, \quad \text{对于所有 } i \tag{10.118}$$

相反，如果式(10.118)满足，则最大化 $h(\mathbf{Z})$得到 $\mathbf{W}=\mathbf{A}^{-1}$，从而盲源分离问题得到解决。

现在我们可以总结用于盲源分离的最大熵原则思想如下（Bell and Sejnowski,1995）：

如图 10.16 所示，令在分离器输出的非线性由初始源分布定义为

$$z_i = g_i(y_i) = \int_{-\infty}^{z_i} p_{s_i}(s_i)\mathrm{d}s_i, \quad \text{当 } i = 1,2,\cdots,m \tag{10.119}$$

最大化在非线性 G 输出端的随机向量 $\mathbf{Z}$(其第 i 个元素具有样本值 z_i）的微分熵等价于 $\mathbf{W}=\mathbf{A}^{-1}$，这将产生完全的盲源分离。

最大熵和最大似然方法的等价性

对所有的 i，在随机变量 Z_i 是区间 [0,1] 上均匀分布的条件下，最大熵方法和最大似然方法对盲源分离问题是等价的（Cardoso，1997）。为了证明这个关系，我们首先利用微分的链式规则将式(10.115)改写为等价形式：

$$J_{ij} = \sum_{k=1}^{m} \frac{\partial z_i}{\partial y_i}\frac{\partial y_i}{\partial x_k}\frac{\partial x_k}{\partial s_j} = \sum_{k=1}^{m} \frac{\partial z_i}{\partial y_i} w_{ik} a_{kj} \tag{10.120}$$

其中偏导数 $\partial z_i/\partial y_i$ 是需要定义的。因此 Jacobi 矩阵 $\mathbf{J}$ 可以表达为

$$\mathbf{J} = \mathbf{DWA}$$

其中 D 是对角矩阵

$$\mathbf{D} = \mathrm{diag}\left(\frac{\partial z_1}{\partial y_1}, \frac{\partial z_2}{\partial y_2}, \cdots, \frac{\partial z_m}{\partial y_m}\right)$$

所以

$$|\det(\mathbf{J})| = |\det(\mathbf{WA})| \prod_{i=1}^{m} \frac{\partial z_i}{\partial y_i} \tag{10.121}$$

对于由权值矩阵 $\mathbf{W}$ 和非线性函数 $\mathbf{G}$ 参数化的概率密度函数 $p_{\mathbf{S}}(\mathbf{s})$，根据式(10.121)，它的估计可以形式地表示为（Roth and Baram，1996）：

$$p_{\mathbf{S}}(\mathbf{s}|\mathbf{W},\mathbf{G}) = |\det(\mathbf{WA})| \prod_{i=1}^{m} \frac{\partial g_i(y_i)}{\partial y_i} \tag{10.122}$$

因此在这种条件下，可以看出盲源分离最大化对数似然函数 $\log p_{\mathbf{S}}(\mathbf{s}|\mathbf{W},\mathbf{G})$ 等价于最大化熵 $h(\mathbf{Z})$。也就是说，最大熵方法与最大似然方法是等价的。

盲源分离的学习算法

查看式（10.116）的第二行，注意到由于源的分布通常是固定的，最大化熵 $h(\mathbf{Z})$ 要求对权矩阵 W 求分母项 $\log|\det(\mathbf{J}(\mathbf{s}))|$ 的期望的最大值。我们的目标是找到一个自适应算法来进行这样的计算，因此可以考虑瞬时目标函数：

$$\boldsymbol{\Phi} = \log|\det(\mathbf{J})| \tag{10.123}$$

将式(10.121)代入式(10.123)得到：

$$\boldsymbol{\Phi} = \log|\det(\mathbf{A})| + \log|\det(\mathbf{W})| + \sum_{i=1}^{m}\log\left(\frac{\partial z_i}{\partial y_i}\right) \tag{10.124}$$

所以对分离器的权值矩阵 $\mathbf{W}$ 求 $\boldsymbol{\Phi}$ 的微分得到（见习题 10.20）：

$$\frac{\partial \boldsymbol{\Phi}}{\partial \mathbf{W}} = \mathbf{W}^{-T} + \sum_{i=1}^{m}\frac{\partial}{\partial \mathbf{W}}\log\left(\frac{\partial z_i}{\partial y_i}\right) \tag{10.125}$$

为了进一步处理这个公式，必须说明由分离器输出馈入的非线性。这里可以使用的非线性的简单形式为 logistic 函数：

$$z_i = g(y_i) = \frac{1}{1+e^{-y_i}}, i = 1,2,\cdots,m \tag{10.126}$$

图 10.17 画出该函数和其反函数的图像。这个图像表明 logistic 函数满足盲源分离的单调性和可逆性的基本要求。将式(10.126)代入式(10.125)得到：

$$\frac{\partial \boldsymbol{\Phi}}{\partial \mathbf{W}} = \mathbf{W}^{-T} + (\mathbf{1} - 2\mathbf{z})\mathbf{x}^T$$

其中 $\mathbf{x}$ 是接收信号向量，$\mathbf{z}$ 是分离器的输出向量经非线性变化后的输出。$\mathbf{1}$ 是分量都为 1 的向量。

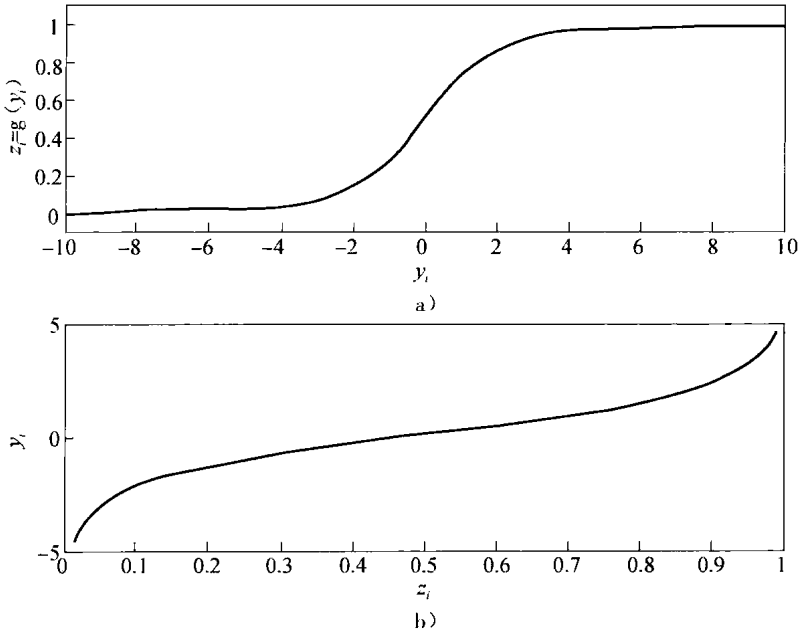


图 10.17　a) logistic 函数：$z_i = g(y_i) = \dfrac{1}{1+e^{-y_i}}$；b) logistic 函数的逆：$y_i = g^{-1}(z_i)$

学习算法的目的就是最大化微分熵 $h(\mathbf{Z})$。因此采用最速下降法，应用于权值矩阵 $\mathbf{W}$ 的变

化可表示为[17]：

$$\Delta \mathbf{W} = \eta \frac{\partial \boldsymbol{\Phi}}{\partial \mathbf{W}} = \eta(\mathbf{W}^{-T} + (\mathbf{1} - 2\mathbf{z})\mathbf{x}^T) \tag{10.127}$$

其中 η 是学习率参数。与 10.14 节描述的 ICA 自然梯度学习算法相类似，可以利用自然梯度消除对转置权值矩阵 $\mathbf{W}^T$ 求逆的要求，这等价于对式(10.127)乘以矩阵积 $\mathbf{W}^T\mathbf{W}$。这个最优调整产生权值变化所希望的公式为：

$$\begin{aligned}\Delta \mathbf{W} &= \eta(\mathbf{W}^{-T} + (\mathbf{1} - 2\mathbf{z})\mathbf{x}^T)\mathbf{W}^T\mathbf{W} = \eta(\mathbf{I} + (\mathbf{1} - 2\mathbf{z})(\mathbf{W}\mathbf{x})^T)\mathbf{W} \\ &= \eta(\mathbf{I} + (\mathbf{1} - 2\mathbf{z})\mathbf{y}^T)\mathbf{W}\end{aligned} \tag{10.128}$$

其中 $\mathbf{I}$ 是单位矩阵，$\mathbf{y}$ 是分离器的输出。所以计算权值矩阵 $\mathbf{W}$ 的学习算法可以表示为：

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \eta(\mathbf{I} + (\mathbf{1} - 2\mathbf{z}(n))\mathbf{y}^T(n))\mathbf{W}(n) \tag{10.129}$$

算法的初值 $\mathbf{W}(0)$选取一组均匀分布的小数值。参照图 10.16 的方框图，我们可以看到在第 n 时间步输出 $\mathbf{y}(n)$由输入 $\mathbf{x}(n)$通过矩阵积 $\mathbf{W}(n)\mathbf{x}(n)$来定义。因此，在每一次分离矩阵 $\mathbf{W}(n)$ 的更新，我们可以相应地计算分离器输出 $\mathbf{y}(n)$的更新值。

10.17　独立分量分析的负熵最大化

在 10.14 节至 10.16 节讨论了 ICA 这样或那样的形式，这些 ICA 算法基本上是根植于统计独立分量原则的，而这一原则自身是基于 10.14 节讨论过的相对熵的。本节我们将背离这一原则并描述另一个不同地根植于信息论的 ICA 算法。这一算法称为 FastICA 算法，由 Hyvärinen and Oja(1997) 提出。

更具体来说，FastICA 算法开发了非高斯性的概念，而非高斯性在前面的 10.12 节中讨论过，它是独立分量分析的需要。对于随机变量的非高斯性的重要测量是负熵，它是基于微分熵的。因此我们通过描述这一新概念来开始对 FastICA 算法的讨论。

负熵

在例 2 中，我们证明了高斯随机变量和其他随机变量的不同在于其具有最大的可能微分熵。具体来说，高斯随机变量的信息内容是约束于二阶统计的，由此能够计算所有高阶统计。为了评估一个随机变量的非高斯性，需要假定一个满足两个性质的测量：

1. 这个测量是非负的，假设对于高斯随机变量其极限值为零。
2. 对于所有其他随机变量，这个测量大于零。

负熵的概念满足这两个性质。

考虑一个已知是非高斯的随机变量 $\mathbf{X}$。$\mathbf{X}$ 的负熵定义为

$$N(\mathbf{X}) = H(\mathbf{X}_{\text{高斯}}) - H(\mathbf{X}) \tag{10.130}$$

其中 $H(\mathbf{X})$是 $\mathbf{X}$ 的微分熵，且 $H(\mathbf{X}_{\text{高斯}})$ 是协方差矩阵等于 $\mathbf{X}$ 的高斯随机向量的微分熵。

按信息论的术语，负熵是关于非高斯性的良好的测量。但这需要大量的计算时间，这限制了其实际应用。要克服这一计算困难，我们必须寻找对负熵的简单逼近。下面考虑 0 -均值单位方差的非高斯随机变量 V。Hyvarinen and Oja(2000) 提案了逼近：

$$N(V) = \mathbb{E}[\Phi(V)] - \mathbb{E}[\Phi(U)]^2 \tag{10.131}$$

其中 U 也是 0 -均值单位方差的高斯随机变量（即它是标准化的）。对所有实际目的，$\Phi(\cdot)$是非二次函数；令人满意的是，这一函数不能快速增长，因而使得估计过程鲁棒。根据 Hyvärinen and Oja(2000)，下面给出的两个选择证明了其有效性。

$$1.\ \Phi(v) = \log(\cosh(v)) \tag{10.132}$$

$$2.\ \Phi(v) = \exp\left(-\frac{v^2}{2}\right) \tag{10.133}$$

其中 v 是随机变量 V 的样本值。因而可以将式(10.131)作为独立分量分析目的的最大化的“反差函数”。除了伸缩因子，函数 $\Phi(v)$可以看作概率密度函数。注意在式(10.132)和式(10.133)中使用的 $\Phi(\cdot)$不能与式(10.123)中使用的矩阵 Φ 相混淆。

FastICA 算法的基本学习规则

为了给 FastICA 的开发铺路，我们首先考虑这个算法的一个单一单元（single-unit）版本。术语“unit”表示一个具有可调权值向量 $\mathbf{w}$ 的神经元。这个神经元将被设计来供我们推出 FastICA 算法的基本学习规则。

令 $\mathbf{x}$ 为预白噪化的 0 -均值随机向量 $\mathbf{X}$ 的样本值，其被应用于神经元的输入。我们通过如下的做法来开始推出基本学习规则。

最大化可调权值向量 $\mathbf{w}$ 对随机向量 $\mathbf{X}$ 的投影的负熵，在 $\|\mathbf{w}\|=1$ 的约束之下。

投影是通过内积 $\mathbf{w}^T\mathbf{X}$ 来定义的。有了预白噪化的随机向量 $\mathbf{X}$，约束 $\|\mathbf{w}\|=1$ 等价于约束投影具有单位方差，如下所示：

$$\text{var}[\mathbf{w}^T\mathbf{X}]=\mathbb{E}[(\mathbf{w}^T\mathbf{X})^2]=\mathbb{E}[\mathbf{w}^T\mathbf{X}\mathbf{X}^T\mathbf{w}]=\mathbf{w}^T\mathbb{E}[\mathbf{X}\mathbf{X}^T]\mathbf{w}=\mathbf{w}^T\mathbf{w}=\|\mathbf{w}\|^2=1 \tag{10.134}$$

在式(10.134)的第一行，利用了加于 $\mathbf{X}$ 的 0 -均值假设，在第三行，利用了加于 $\mathbf{X}$ 的预白噪化假设。

为了使基本的学习规则是计算有效的，我们寻找式(10.131)的逼近来作为计算负熵 $N(V)$ 的公式，这里 $V=\mathbf{w}^T\mathbf{X}$。由于 U 是 0 -均值单位方差的标准高斯随机变量，因而独立于 $\mathbf{w}$，这就有对于 $\mathbf{w}$ 最大化 $N(V)$ 等价于最大化非二次函数 $\Phi(V)=\Phi(\mathbf{w}^T\mathbf{X})$。因此可以重新构造感兴趣的优化问题如下：

最大化期望 $\mathbb{E}[\Phi(\mathbf{w}^T\mathbf{x})]$，在 $\|\mathbf{w}\|=1$ 的约束之下。

根据优化理论的 Karush-Kuhn-Tucker 条件（在第 6 章讨论过），对这一有约束最大化问题的解可以在下面的方程中找到：

$$\frac{\partial}{\partial\mathbf{w}}\mathbb{E}[\Phi(\mathbf{w}^T\mathbf{x})]-\lambda\mathbf{w}=\mathbf{0} \tag{10.135}$$

其中 $\mathbf{x}$ 是随机向量 $\mathbf{X}$ 的样本值。期望 $\mathbb{E}[\Phi(\mathbf{w}^T\mathbf{x})]$对于权值向量 $\mathbf{w}$ 的梯度向量为：

$$\frac{\partial}{\partial\mathbf{w}}\mathbb{E}[\Phi(\mathbf{w}^T\mathbf{x})]=\mathbb{E}\left[\frac{\partial}{\partial\mathbf{w}}\Phi(\mathbf{w}^T\mathbf{x})\right]=\mathbb{E}\left[\frac{\partial(\mathbf{w}^T\mathbf{x})}{\partial\mathbf{w}}\frac{\partial}{\partial\mathbf{w}^T\mathbf{x}}\Phi(\mathbf{w}^T\mathbf{x})\right]=\mathbb{E}[\mathbf{x}\varphi(\mathbf{w}^T\mathbf{x})] \tag{10.136}$$

其中 $\varphi(\cdot)$是非二次函数 $\Phi(\cdot)$对其自变量的一阶导数，即

$$\varphi(v)=\frac{d\Phi(v)}{dv}$$

例如，对于式(10.132)定义的函数 $\Phi(\cdot)$，有

$$\varphi(v)=\frac{\mathrm{d}}{\mathrm{d}v}\log(\cosh(v))=\tanh(v)$$

对于式(10.133) 定的函数 $\Phi(v)$，有

$$\varphi(v)=\frac{\mathrm{d}}{\mathrm{d}v}\left(-\exp\left(-\frac{v^2}{2}\right)\right)=v\exp\left(-\frac{v^2}{2}\right)$$

因此，可以重写式(10.135)为等价形式：

$$\mathbb{E}[\mathbf{x}\varphi(\mathbf{w}^T\mathbf{x})]-\lambda\mathbf{w}=\mathbf{0} \tag{10.137}$$

我们感兴趣的是找到执行基本学习规则的计算有效率的迭代过程，此时最优权值向量 $\mathbf{w}$ 指向独立分量的方向。为此，我们提出将牛顿法应用于式(10.137)的左边。

用向量值函数来记这一表示式如下：

$$\mathbf{f}(\mathbf{w}) = \mathbb{E}[\mathbf{x}\varphi(\mathbf{w}^T\mathbf{x})] - \lambda\mathbf{w} \tag{10.138}$$

在第 3 章和第 4 章讨论过牛顿法。要应用该方法，我们需要函数 $\mathbf{f}(\mathbf{w})$ 的 Jacobi 矩阵，由下式定义：

$$\begin{aligned}\boldsymbol{J}(\mathbf{w}) &= \frac{\partial}{\partial \mathbf{w}}\mathbf{f}(\mathbf{w}) = \frac{\partial}{\partial \mathbf{w}}\{\mathbb{E}[\mathbf{x}\varphi(\mathbf{w}^T\mathbf{x})] - \lambda\mathbf{w}\} = \frac{\partial}{\partial \mathbf{w}}\mathbb{E}[\mathbf{x}\varphi(\mathbf{w}^T\mathbf{x})] - \frac{\partial}{\partial \mathbf{w}}(\lambda\mathbf{w}) \\ &= \mathbb{E}\left[\frac{\partial}{\partial \mathbf{w}}\mathbf{x}\varphi(\mathbf{w}^T\mathbf{x})\right] - \lambda\mathbf{I} = \mathbb{E}[\mathbf{x}\mathbf{x}^T\varphi'(\mathbf{w}^T\mathbf{x})] - \lambda\mathbf{I}\end{aligned} \tag{10.139}$$

其中 $\mathbf{I}$ 是单位矩阵。$\varphi'(\cdot)$ 是函数 $\varphi(\cdot)$ 对其自变量的微分。换句话说 $\varphi'(\cdot)$ 是初始函数 $\Phi(\cdot)$ 对其自变量的二阶导数。现在我们可以看到，为什么稍早时候我们声明 $\varphi(\cdot)$ 必须是非二次函数；否则，在式(10.139)中 $\varphi'(\cdot)$ 将等于一个常量，而这是不可接受的。

然而，在继续进行之前，我们希望进一步简化基本学习规则的推出。由于输入向量 $\mathbf{x}$ 被预白噪化，因此可以假设外积 $\mathbf{x}\mathbf{x}^T$ 和式(10.139)中的项 $\varphi'(\mathbf{w}^T\mathbf{x})$ 是统计独立的。在这一假设下，可以继续写

$$\mathbb{E}[\mathbf{x}\mathbf{x}^T\varphi'(\mathbf{w}^T\mathbf{x})] \approx \mathbb{E}[\mathbf{x}\mathbf{x}^T]\mathbb{E}[\varphi'(\mathbf{w}^T\mathbf{x})] = \mathbb{E}[\varphi'(\mathbf{w}^T\mathbf{x})]\mathbf{I} \tag{10.140}$$

其中，在最后一行，我们利用了输入 $\mathbf{x}$ 的白化性质：即 $\mathbb{E}[\mathbf{x}\mathbf{x}^T] = \mathbf{I}$。相应地，我们现在发现在式(10.139)的 Jacobi 矩阵 $J(w)$ 的整个表达式具有标量乘以单位矩阵 I 的形式，如下所示：

$$\boldsymbol{J}(\mathbf{w}) \approx (\mathbb{E}[\varphi'(\mathbf{w}^T\mathbf{x})] - \lambda)\mathbf{I} \tag{10.141}$$

这是可逆的。有了目前的逼近，我们可以将牛顿迭代步表示为：

$$\mathbf{w}^+ = \mathbf{w} - \boldsymbol{J}^{-1}(\mathbf{w})\mathbf{f}(\mathbf{w}) \tag{10.142}$$

其中 $\mathbf{w}$ 是权值向量的老的值，$\mathbf{w}^+$ 是更新值。也注意到我们在迭代步中使用了负号，因为我们是在寻找函数 $\mathbf{f}(\mathbf{w})$ 的最大值。因此，将式(10.141)代入到式 (10.142)，得到：

$$\mathbf{w}^+ = \mathbf{w} - (\mathbb{E}[\varphi'(\mathbf{w}^T\mathbf{x})] - \lambda)^{-1}(\mathbb{E}[\mathbf{x}\varphi(\mathbf{w}^T\mathbf{x})] - \lambda\mathbf{w})$$

可以通过在等式的两边乘以标量 $(\mathbb{E}[\varphi'(\mathbf{w}^T\mathbf{x})] - \lambda)$ 来简化迭代步，这产生：

$$\mathbf{w}^+ = (\mathbb{E}[\varphi'(\mathbf{w}^T\mathbf{x})] - \lambda)\mathbf{w} - (\mathbb{E}[\mathbf{x}\varphi(\mathbf{w}^T\mathbf{x})] - \lambda\mathbf{w}) = \mathbb{E}[\varphi'(\mathbf{w}^T\mathbf{x})]\mathbf{w} - \mathbb{E}[\mathbf{x}\varphi(\mathbf{w}^T\mathbf{x})] \tag{10.143}$$

其中，在左边，在新的值 $\mathbf{w}^+$ 中我们吸收了伸缩因子 $(\mathbb{E}[\varphi'(\mathbf{w}^T\mathbf{x})] - \lambda)$。并注意到我们不需要知道拉格朗日乘子 λ 的值，因为它在式(10.143)的迭代步中被代数消去了。

式(10.143)是我们所探索的基本学习规则的核心。实际上，根据这一式子，我们现在可以模型化单一神经元，这一公式围绕这个神经元建立，如图 10.18 所示。根据这个图，将非线性函数 $\varphi(\cdot)$ 看作神经元的激活函数。

有了式(10.143)的迭代步，我们最终可以总结 FastICA 算法的基于牛顿法的学习规则如下：

1. 选择权值向量 $\mathbf{w}$ 的初始值，利用随机数产生器在 $\mathbf{w}$ 的欧几里得范数为单位 1 的约束下来选择。

2. 利用权值向量 $\mathbf{w}$ 的老的值来计算更新值：

$$\mathbf{w}^+ = \mathbb{E}[\varphi'(\mathbf{w}^T\mathbf{x})]\mathbf{w} - \mathbb{E}[\mathbf{x}\varphi(\mathbf{w}^T\mathbf{x})]$$

3. 归一化更新后的权值向量 $\mathbf{w}^+$ 使得其欧几里得范数为 1，如下所示：

$$\mathbf{w} = \frac{\mathbf{w}^+}{\|\mathbf{w}^+\|}$$

4. 如果算法还没有收敛，转回到第 2 步并重复这一计算。

为了计算学习规则第 2 步的期望，我们可以调用遍历性（ergodicity）并且用基于输入向量 $\mathbf{x}$ 的独立样本（实现）序列的时间平均来代替期望。

我们说学习规则收敛（即规则达到了一个均衡点）当更新的权值向量 $\mathbf{w}^+$ 和老的权值向量

$\mathbf{w}$ 指向相同的方向时。即，内积 $\mathbf{w}^T\mathbf{w}^+$ 的绝对值接近于单位 1。然而，由于仅在乘积伸缩因子内 ICA 算法能够检测独立分量，因此不需要寻找权值向量 $\mathbf{w}^+$ 和 $\mathbf{w}$ 指向完全相同的方向的均衡点，$\mathbf{w}^+$ 是 $\mathbf{w}$ 的负也是可接受的。

作为最后的批注：算法的推导以及应用是基于混合器输出已经被预白化的前提下；而预白化问题在第 10.12 节中讨论过。

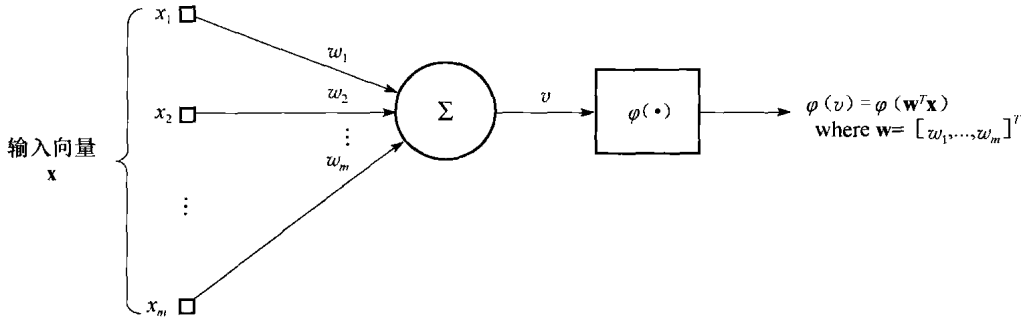


图 10.18　表示 FastICA 算法的基本学习规则特征的神经元模型

FastICA 算法的多单元版本

自然地，在单一神经元上建立的基于牛顿法的学习规则，仅能估计生成观测向量 $\mathbf{x}$ 的 m 个独立分量（源）中的一个。为了将这一规则扩展到估计所有的 m 个独立分量，我们明显需要一个具有 m 个神经元的网络或者其等价物。

为了探索这一网络所需要满足的条件，令 $\mathbf{w}_1,\mathbf{w}_2,\cdots,\mathbf{w}_m$ 记由网络的 m 个神经元产生的权值向量。为了这个向量集能够表示盲源分离（BSS）问题的正确解，需要两个条件：

1. 正交性。假设随机观测向量 $\mathbf{X}$ 被同时作用于 m 个神经元，产生输出集：

$$\{V_i\}_{i=1}^m, \text{其中 } V_i = \mathbf{w}_i^T\mathbf{X}$$

为了阻止所有 m 个权值向量收敛到相同的独立分量，我们需要神经元输出之间是彼此不相关的，即：

$$\mathbb{E}[V_iV_j] = 0, \quad \text{当 } j \neq i \tag{10.144}$$

因此，有了 $V_i=\mathbf{w}_i^T\mathbf{X}$ 和 $V_j=\mathbf{w}_j^T\mathbf{X}=\mathbf{X}^T\mathbf{w}_j$，我们有

$$\mathbb{E}[V_iV_j] = \mathbb{E}[\mathbf{w}_i^T\mathbf{X}\mathbf{X}^T\mathbf{w}_j] = \mathbf{w}_i^T\,\mathbb{E}[\mathbf{X}\mathbf{X}^T]\mathbf{w}_j = \mathbf{w}_i^T\mathbf{w}_j, \quad \text{当 } j \neq i$$

其中，在最后一行，我们利用了观测向量 $\mathbf{X}$ 的白化性质。因此，随后为了满足式(10.144)的去相关性质，权值向量 $\mathbf{w}_1,\mathbf{w}_2,\cdots,\mathbf{w}_m$ 必须形成正交集，如下所示：

$$\mathbf{w}_i^T\mathbf{w}_j = 0, \quad \text{当 } j \neq i \tag{10.145}$$

2. 归一性。为了和基于牛顿法的学习规则相一致，我们需要将每一个权值向量归一化使其欧几里得范数等于单位 1，如下所示：

$$\|\mathbf{w}_i\| = 1, \quad \text{对于所有 } i \tag{10.146}$$

将条件 1 和 2 放在一起，总结如下：

为了使权值向量 $\mathbf{w}_1,\mathbf{w}_2,\cdots,\mathbf{w}_m$ 能提供生成观测向量 $\mathbf{x}$ 的 m 个独立分量（源）的估计，它们必须构成一个正交集，如下所示：

$$\mathbf{w}_i^T\mathbf{w}_j = \begin{cases} 1, & \text{当 } j = i \\ 0, & \text{其他} \end{cases} \tag{10.147}$$

Gram-Schmidt 正交化过程

式(10.147)所加于权值向量的两个必要条件使我们想起一个简单的降阶方法（deflational

method)，它基于 Gram-Schmidt 正交化过程[18]，用于逐一估计所有的 m 个独立分量。该方法最初是由 Hyvarinen and Oja(1997，2000) 提出的。具体来说，假设我们首先在观测向量 $\mathbf{x}$ 的 N 个独立实现（样本）上运行单一神经元的基于牛顿法的学习规则，获得 m 个独立分量中一个权值向量 $\mathbf{w}_1$ 的估计。当在 $\mathbf{x}$ 的下一个 N 独立实现集上运行这一规则时，假设结果权值向量记为 α_2。对于第二个权值向量采用不同记号的理由是向量 α_2 不一定和 $\mathbf{w}_1$ 是正交的。为了矫正正交性这一必要条件的偏移，我们应用 Gram-Schmidt 正交化过程，获得：

$$\boldsymbol{\theta}_2 = \boldsymbol{\alpha}_2 - (\boldsymbol{\alpha}_2^T \mathbf{w}_1)\mathbf{w}_1$$

其中从 α_2 中减去“投影”$(\boldsymbol{\alpha}_2^T \mathbf{w}_1)\ \mathbf{w}_1$。认识到 $\| \mathbf{w}_1 \| = 1$，直接可证 $\boldsymbol{\theta}_2$ 实际上是正交于 $\mathbf{w}_1$ 的，即 $\boldsymbol{\theta}_2^T \mathbf{w}_1 = 0$。剩下要做的是通过下式归一化 $\boldsymbol{\theta}_2$：

$$\mathbf{w}_2 = \frac{\boldsymbol{\theta}_2}{\| \boldsymbol{\theta}_2 \|}$$

按这一种方式进行下去，假设在观测向量 $\mathbf{x}$ 的下一个 N 样本集上，基于牛顿法的学习规则产生权向量 $\boldsymbol{\alpha}_3$，再一次 $\boldsymbol{\alpha}_3$ 和 $\mathbf{w}_2$ 以及 $\mathbf{w}_1$ 不一定正交。为了校正这些偏差，我们再一次应用 Gram-Schmidt 正交化过程，得到：

$$\boldsymbol{\theta}_3 = \boldsymbol{\alpha}_3 - (\boldsymbol{\alpha}_3^T \mathbf{w}_1)\mathbf{w}_1 - (\boldsymbol{\alpha}_3^T \mathbf{w}_2)\mathbf{w}_2$$

这里从 $\boldsymbol{\alpha}_3$ 中减去了投影 $(\boldsymbol{\alpha}_3^T \mathbf{w}_j)\ \mathbf{w}_j$，$j=1$，2。认识到 $\| \mathbf{w}_1 \| = \| \mathbf{w}_2 \| = 1$ 且 $\mathbf{w}_1^T \mathbf{w}_2 = 0$，直接可证 $\boldsymbol{\theta}_3$ 和 $\mathbf{w}_1$ 以及 $\mathbf{w}_2$ 都正交。因此，剩下要做的是归一化 $\boldsymbol{\theta}_3$：

$$\mathbf{w}_3 = \frac{\boldsymbol{\theta}_3}{\| \boldsymbol{\theta}_3 \|}$$

我们可以继续这一方式直到求出所有 m 个独立分量。

下面总结利用 Gram-Schmidt 正交化过程计算期望的 m 个权值向量：

1. 给定 $\mathbf{w}_1$ 为由单一神经元基于牛顿法学习规则在其完全迭代下产生的归一化权值向量，给定 $\alpha_2, \cdots, \alpha_{i+1}$ 为规则在下 i 次完全迭代产生的权值向量，计算

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\alpha}_{i+1} - \sum_{j=1}^{i} (\boldsymbol{\theta}_{i+1}^T \mathbf{w}_j)\mathbf{w}_j, i = 1, 2, \cdots, m-1$$

其中“投影”$(\boldsymbol{\theta}_{i+1}^T \mathbf{w}_j)\mathbf{w}_j$ 被从 $\boldsymbol{\alpha}_{i+1}$ 中减去了，$j = 1, 2, \cdots, i$。

2. 归一化 $\boldsymbol{\theta}_{i+1}$

$$\mathbf{w}_{i+1} = \frac{\boldsymbol{\theta}_{i+1}}{\| \boldsymbol{\theta}_{i+1} \|}, i = 1, 2, \cdots, m-1$$

基于这一过程的 FastICA 算法表示了这个算法的单一单元降阶版本[19]。

FastICA 算法的性质

和其他的 ICA 算法相比，FastICA 算法具有一些希望的性质（Hyvärinen and Oja，2000；Tichavsky 等，2006）：

1. 在无噪、线性生成模型的假设下，FastICA 算法相对来说是快速的——这个算法因此而得名。在 10.14、10.15、10.16 节中讨论过的基于梯度的 ICA 算法趋向于线性方式收敛，而 FastICA 的收敛是三次的（或至少两次）。

2. 和基于梯度的 ICA 算法不同，FastICA 不需要利用学习率参数，使得其设计更简单。

3. FastICA 算法具有利用任意非二次型的非线性 $\varphi(v)$ 找到实际上任意非高斯分布的独立分量的内在能力。与算法的多功能性相比较，基于梯度的 ICA 算法的适用限制于亚高斯或者超高斯分布，而且必须对非线性的选择特别小心。

4. 通过对非二次函数 $\varphi(\cdot)$ 的适当选择，以式(10.132)和式(10.133)为例，FastICA 算法的鲁棒性可以得到保证，甚至在大的数据集以及在某种噪声条件下。

5. 由 FastICA 算法系统化地计算一个一个的独立分量。算法的这一特征使其对于探测数据分析（exploratory data analysis）成为一个有用的工具，其中独立分量极限数的估计可能是感兴趣的应用所需要的。这一分析的计算负载因而得到削减。

6. FastICA 算法具有几个通常和神经网络相关联的特性：并行性、分布式计算、简单性、小的存储容量需求。另一方面，基于随机梯度的 ICA 算法（以 10.14 节讨论的自然梯度算法为例）对于包含不稳定环境的盲源分离问题是更好的选择，此时对于快速自适应有着明确的需要。

10.18　相关独立分量分析

通过回顾本章前面已经介绍过的关于信息论在学习模型的建立方面的素材，我们发现最大化互信息原则（或者简称 Informax 原则）是突出的。Infomax 原则不仅在我们理解冗余删减、感知器的模型化、独立分量的提取时扮演着重要的角色，而且其相关的 Imax 原则自身扮演着提取空间相关特征的角色。实际上，Infomax 和 Imax 原则是互补的角色：

Infomax 处理穿过网络的信息流，而 Imax 处理穿过一对网络输出的空间相关性。

图 10.19 描述了这两个原则包含在一起的情景。具体地，我们有两个分离的但是维数相同的神经网络：神经网络 a 由权值矩阵 $\mathbf{W}_a$ 刻画，网络 b 由权值矩阵 $\mathbf{W}_b$ 刻画。这两个网络都假设为无噪的，目标是将 Infomax 和 Imax 原则组合起来使得前面提到的性质综合起来成为一个混合学习原则，该性质根据 Infomax 原则的每个网络的信息流以及根据 Imax 原则通过视为一对一对（pair-by-pair）基的两个网络的神经输出的空间相关性。

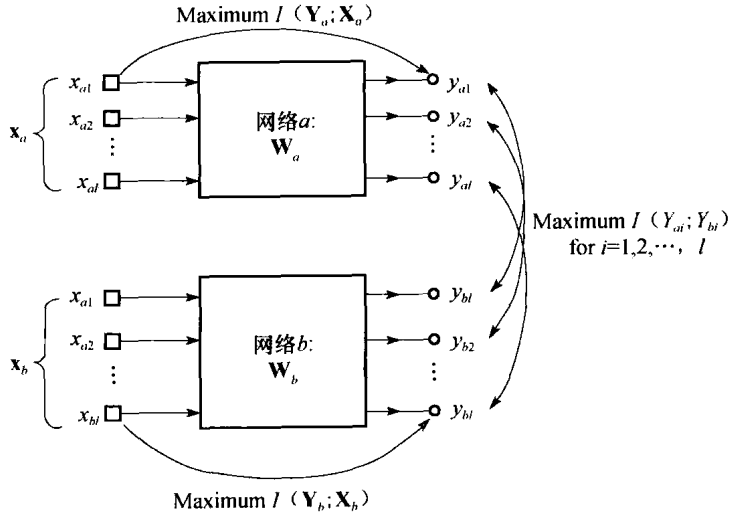


图 10.19　相关 ICA 的一对网络布局

Infomax 原则的部分

首先考虑作用于穿过图 10.19 所示的每个网络输入-输出的 Infomax 原则。然后，由式(10.60)，其属于无噪的例 7，由权值矩阵 $\mathbf{W}_a$ 刻画的网络可通过互信息描述：

$$I(\mathbf{Y}_a;\mathbf{X}_a) = -\mathbb{E}[\log p_{\mathbf{Y}_a}(\mathbf{y}_a)]$$

其中，为了简化表达，我们忽略了和权值矩阵 $\mathbf{W}_a$ 无关的附加常数；而且，我们使用了包含随机向量的熵的式(10.60)。由于构成输出随机向量 $\mathbf{Y}_a$ 的元素是“独立”的，我们可以将 Y_a 的概率密度函数表示为：

$$p_{\mathbf{Y}_a(\mathbf{y}_a)} = \prod_{i=1}^{l} p_{\mathbf{Y}_{a,i}}(y_{a,i})$$

其中 1 是输出端口数。因而可以继续写

$$I(\mathbf{Y}_a;\mathbf{X}_a) = -\mathbb{E}\left[\log \prod_{i=1}^{l} p_{Y_{a,i}}(y_{a,i})\right] = -\mathbb{E}\left[\sum_{i=1}^{l} \log\, p_{Y_{a,i}}(y_{a,i})\right], i = 1,2,\cdots,l \quad (10.148)$$

相似地，对于第二个网络，由权值矩阵 $\mathbf{W}_b$ 刻画，可以写

$$I(\mathbf{Y}_b;\mathbf{X}_b) = -\mathbb{E}\left[\sum_{i=1}^{l} \log p_{Y_{b,i}}(y_{b,i})\right], i = 1,2,\cdots,l \quad (10.149)$$

Imax 原则的部分

下面考虑 Imax 原则，该原则应用于这两个网络的输出，按逐对的原则对待。根据式(10.50)的第二行，在输出 $Y_{a,i}$ 和 $Y_{b,i}$ 之间可以用系词来表示互信息如下：

$$I(\mathbf{Y}_{a,i};\mathbf{Y}_{a,b}) = \mathbb{E}\left[\log c_{Y_{a,i};Y_{b,i}}(\mathbf{y}_{a,i};\mathbf{y}_{b,i})\right], \quad \text{当 } i = 1,2,\cdots,l$$

此外，由于图 10.19 中每个网络的 l 个输出是独立的，这些各个互信息部分是加性的，生成和：

$$\sum_{i=1}^{l} I(Y_{a,i};Y_{b,i}) = \mathbb{E}\left[\sum_{i=1}^{l} \log c_{Y_{a,i};Y_{b,i}}(y_{a,i};y_{b,i})\right] \quad (10.150)$$

总体代价函数

令 $J(\mathbf{W}_a, \mathbf{W}_b)$ 记总体平均目标函数，解释了 Infomax 和 Imax 原则的联合功能。然后，结合式(10.148)到式(10.150)的互信息部分，写出：

$$\begin{aligned} J(\mathbf{W}_a,\mathbf{W}_b) &= -\mathbb{E}\left[\sum_{i=1}^{l} \log p_{Y_{a,i}}(y_{a,i})\right] - \mathbb{E}\left[\sum_{i=1}^{l} \log p_{Y_{b,i}}(y_{b,i}) - \mathbb{E}\left[\sum_{i=1}^{l} \log c_{Y_{a,i}Y_{b,i}}(y_{a,i},y_{b,i})\right]\right] \\ &= -\mathbb{E}\left[\sum_{i=1}^{l} \log(p_{Y_{a,i}}(y_{a,i}) p_{Y_{b,i}}(y_{b,i}) c_{Y_{a,i},Y_{b,i}}(y_{a,i},y_{b,i}))\right] \\ &= -\mathbb{E}\left[\sum_{i=1}^{l} \log p_{Y_{a,i}Y_{b,i}}(y_{a,i},y_{b,i})\right] \end{aligned} \quad (10.151)$$

其中，在最后一行，使用式(10.49)表示输出随机变量 $Y_{a,i}$ 和 $Y_{b,i}$ 的联合概率密度函数。目标函数 $J(\mathbf{W}_a, \mathbf{W}_b)$ 定义了这两个网络输出集 $\{Y_{a,i}\}_{i=1}^{l}$ 和 $\{Y_{b,i}\}_{i=1}^{l}$ 的联合熵的和，而这两个集合被视为有序的一对一对基；这些输出相应地依赖于权值矩阵 $\mathbf{W}_a$ 和 $\mathbf{W}_b$。实际上，更严谨地，在结合系词部分时我们在式(10.151)的第一行引入了负号。这样做，期望的两个网络输出集之间的有序统计相关得到了加强，因此我们可以作出下面的陈述：

相关 ICA 原则最大化网络输出的两个集合 $\{y_{a,i}\}_{i=1}^{l}$ 和 $\{y_{b,i}\}_{i=1}^{l}$ 的联合熵的总体和，这两个集合视为有序一对一对基。最大化是对两个成分网络的权值矩阵 $\mathbf{W}_a$ 和 $\mathbf{W}_b$ 求得的。

为了进一步的处理过程，我们给出两个合理的假设：

1. 图 10.19 的两个神经网络都是线性的，如下所示：

$$\mathbf{y}_i = \begin{bmatrix} y_{a,i} \\ y_{b,i} \end{bmatrix} = \begin{bmatrix} \mathbf{W}_{a,i}^T \mathbf{x}_{a,i} \\ \mathbf{W}_{b,i}^T \mathbf{x}_{b,i} \end{bmatrix}, i = 1,2,\cdots,l \quad (10.152)$$

其中 $\mathbf{w}_{a,i}^T$ 和 $\mathbf{w}_{b,i}^T$ 是权值矩阵 $\mathbf{W}_a$ 和 $\mathbf{W}_b$ 相对应的第 i 行向量。

2. 如第 10.13 节讨论的那样，在自然场景中取得的数据通常是稀疏的，混合输出向量 y_i 的分布可以通过 0-均值广义高斯双变量分布来描述，其 2×2 协方差矩阵是 Σ，如下所示：

$$p_{Y_i}(\mathbf{y}_i)=\frac{1}{2\pi\det^{1/2}(\boldsymbol{\Sigma})}\exp\left(-\frac{1}{2}(\mathbf{y}_i^T\boldsymbol{\Sigma}^{-1}\mathbf{y}_i)^{\alpha/2}\right),i=1,2,\cdots,l \tag{10.153}$$

其中参数 α 控制系词的形状和稀疏。协方差矩阵 $\boldsymbol{\Sigma}$ 定义为：

$$\boldsymbol{\Sigma}=\begin{bmatrix}1 & \rho\\ \rho & 1\end{bmatrix} \tag{10.154}$$

这是定义在式(10.67)的 Imax 的协方差矩阵的方差归一形式。相关系数 ρ 控制一对网络输出 $y_{a,i}$ 和 $y_{b,i}$ 之间的相关程度（对所有 i）。增加 ρ 不影响系词的形状或倾斜度；它通过促成穿过两个网络的学习的更大相关来影响 Imax 在 Infomax 上的相关重要性。

对于 $\alpha=2$，式(10.153)的分布衰减为高斯双变量分布。对于小于 2 的 α，式(10.153)开始呈现超高斯分布的形式，如图 10.20 对三个不同的 α 值所说明的那样。特别对于 $\alpha=1.3$，式(10.153)假设了一个更像语音信号的拉普拉斯分布的形式。

向量 $\mathbf{y}_i$ 包含两个元素 $y_{a,i}$ 和 $y_{b,i}$。因此，将式(10.153)代入式（10.151）并忽略常数项 $2\pi\det^{1/2}(\boldsymbol{\Sigma})$，我们有：

$$\boldsymbol{J}(\mathbf{W}_a,\mathbf{W}_b)=\frac{1}{2}\mathbb{E}\left[\sum_{i=1}^{l}(\mathbf{y}_i^T\boldsymbol{\Sigma}^{-1}\mathbf{y}_i)^{\alpha/2}\right] \tag{10.155}$$

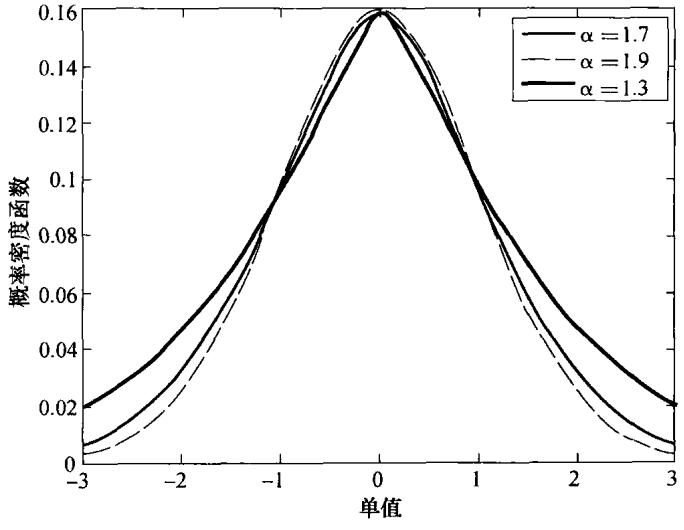


图 10.20　对应于参数 α 的变化值的广义高斯分布

其中总体平均是对 $\mathbf{y}_i$ 来完成的。为了简化计算复杂度，利用二次形式 $\mathbf{y}_i^T\boldsymbol{\Sigma}^{-1}\mathbf{y}_i$，对所有 i 的瞬时值来忽视总体平均的需要。因而，利用式(10.154)对协方差矩阵 $\boldsymbol{\Sigma}$ 的定义，得到：

$$\begin{aligned}\hat{\boldsymbol{J}}(\mathbf{W}_a,\mathbf{W}_b)&=\frac{1}{2}\left[\sum_{i=1}^{l}(\mathbf{y}_i^T\boldsymbol{\Sigma}^{-1}\mathbf{y}_i)^{\alpha/2}\right]\\&=\frac{1}{2(1-\rho^2)}\sum_{i=1}^{l}(y_{i,a}^2-2\rho y_{i,a}y_{i,b}+y_{i,b}^2)^{\alpha/2}\end{aligned} \tag{10.156}$$

其中在 $\hat{J}(\mathbf{W}_a,\mathbf{W}_b)$ 上的小帽将它和总体平均的对应物区分开。

两个网络学习规则的形成

为了形成对权值向量 $\mathbf{W}_{a,i}$ 的自适应规则，通过对 $w_{a,i}$ 微分 $\hat{J}(\mathbf{W}_a,\mathbf{W}_b)$ 来开始。利用微积分学的链式规则，书写为：

$$\frac{\partial\hat{\boldsymbol{J}}(\mathbf{W}_a,\mathbf{W}_b)}{\partial\mathbf{w}_{a,i}}=\frac{\partial\hat{\boldsymbol{J}}(\mathbf{W}_a,\mathbf{W}_b)}{\partial y_{a,i}}\frac{\partial y_{a,i}}{\partial\mathbf{w}_{a,i}} \tag{10.157}$$

对 $y_{i,a}$ 微分式(10.156)产生：

$$\frac{\partial \hat{J}(\mathbf{W}_a,\mathbf{W}_b)}{\partial y_{a,i}} = \frac{\alpha}{(1-\rho^2)}(y_{a,i}-\rho y_{b,i})(y_{a,i}^2-2\rho y_{a,i}y_{b,i}+y_{b,i}^2)^{(\alpha/2)-1} \tag{10.158}$$

利用式(10.152)，我们发现对 $\mathbf{w}_{a,i}$ 微分 $y_{a,i}=\mathbf{w}_{a,i}^T\mathbf{x}_a$ 产生

$$\frac{\partial y_{a,i}}{\partial \mathbf{w}_{a,i}} = \mathbf{x}_a \tag{10.159}$$

因此，在式(10.157)中利用式(10.158)和式(10.159)，获得梯度向量

$$\frac{\partial \hat{J}(\mathbf{W}_a,\mathbf{W}_b)}{\partial \mathbf{w}_{a,i}} = \frac{\alpha}{(1-\rho^2)}(y_{a,i}-\rho y_{b,i})(y_{a,i}^2-2\rho y_{a,i}y_{b,i}+y_{b,i}^2)^{(\alpha/2)-1}\mathbf{x}_a \tag{10.160}$$

目的是最大化瞬时目标函数 $\hat{J}(\mathbf{W}_a,\mathbf{W}_b)$，这意味着我们对迭代计算利用梯度上升（gradient ascent）。相应地，作用于 $\mathbf{w}_{a,i}$ 的改变量定义为：

$$\Delta\mathbf{w}_{a,i} = \frac{\alpha\eta}{(1-\rho^2)}(y_{a,i}-\rho y_{b,i})(y_{a,i}^2-2\rho y_{a,i}y_{b,i}+y_{b,i}^2)^{(\alpha/2)-1}\mathbf{x}_a \tag{10.161}$$

相似地，作用于权值向量 $w_{b,i}$ 的改变量定义为：

$$\Delta\mathbf{w}_{b,i} = \frac{\alpha\eta}{(1-\rho^2)}(y_{b,i}-\rho y_{a,i})(y_{a,i}^2-2\rho y_{a,i}y_{b,i}+y_{b,i}^2)^{(\alpha/2)-1}\mathbf{x}_b \tag{10.162}$$

其中假设网络 b 和网络 a 采用同一个学习率参数 η。

对于网络 a 和 b 的权值修正分别由下式表示：

$$\mathbf{w}_{a,i}^+ = \mathbf{w}_{a,i}+\Delta\mathbf{w}_{a,i} \tag{10.163}$$

$$\mathbf{w}_{b,i}^+ = \mathbf{w}_{b,i}+\Delta\mathbf{w}_{b,i} \tag{10.164}$$

其中 $i=1,2,\cdots,l$。

式(10.163)和式(10.164)这两个修正规则，建立在式(10.161)和式(10.162)的权值改变 $\Delta\mathbf{W}_{a,i}$ 和 $\Delta\mathbf{W}_{b,i}$ 之上，构成了相关 ICA 算法。

式(10.161)和式(10.162)的解释

检查式(10.161)和式(10.162)的学习规则的代数结构是有意义的。首先来看式(10.161)，我们看到作用于权值矩阵 $\mathbf{W}_a$ 的第 i 个列向量的改变 $\Delta\mathbf{W}_{a,i}$，属于图 10.19 所示的网络 a，由下面三个基本因子组成：

1. 伸缩因子 $\alpha\eta/(1-\rho^2)$，这可以简单地看成修正的学习率参数，它对于所有的 i 计算 $\Delta\mathbf{w}_{a,i}$ 和 $\Delta\mathbf{w}_{b,i}$ 而言是共通的。对于参数 α 的修改仅仅影响算法的自适应率。
2. 因子 $(y_{a,i}-\rho y_{b,i})\mathbf{x}_a$ 可以表示为两个二次形式的差，如下所示：

$$(y_{a,i}-\rho y_{b,i})\mathbf{x}_a = (\mathbf{x}_a^T\mathbf{w}_{a,i}\mathbf{x}_a)-\rho(\mathbf{x}_b^T\mathbf{w}_{b,i}\mathbf{x}_a)$$

第一个二次形 $(\mathbf{x}_a^T\mathbf{w}_{a,i}\mathbf{x}_a)$ 仅仅包含网络 a，而第二个二次形 $(\mathbf{x}_b^T\mathbf{w}_{b,i}\mathbf{x}_a)$ 包含了网络 a 和 b。这里需要指出的重点是这样的事实：第二个因子 $(y_{a,i}-\rho y_{b,i})\mathbf{x}_a$ 是独立于参数 α 的；换句话说，这个因子是完全不受输出向量 y_i 是否脱离高斯性的影响的。

3. 第三也是最后一个因子 $(y_{a,i}^2-2\rho y_{a,i}y_{b,i}+y_{b,i}^2)$ 也可以用二次形来表示，如下所示：

$$(y_{a,i}^2-2\rho y_{a,i}y_{b,i}+y_{b,i}^2) = (\mathbf{w}_{a,i}^T\mathbf{x}_a\mathbf{x}_a^T\mathbf{w}_{a,i}-2\rho\mathbf{w}_{a,i}^T\mathbf{x}_a\mathbf{x}_b^T\mathbf{w}_{b,i}+\mathbf{w}_{b,i}^T\mathbf{x}_b\mathbf{x}_b^T\mathbf{w}_{b,i})$$

在这个因子中参数 α 以最显著的方式影响着算法的运行。特别地，当 $\alpha=2$ 时，这个因子的幂变成了 0，因而排除了这个因子对算法的影响。当 $\alpha<2$ 时，这在处理超高斯分布时产生，相关 ICA 算法完成其出色的信号处理角色。

除了下标 a 和 b 相互交换之外，对作用于式(10.162)的学习规则具有相似的解释。

实际考虑

在完成相关 ICA 学习的过程中，假设了图 10.19 中网络输入 $\mathbf{x}_a$ 和 $\mathbf{x}_b$ 是预白化的，这在

ICA 相关的工作中是通常的做法。而且，在学习过程的每一次迭代中，权值是归一化的，如下所示：

$$\mathbf{w}_{a,i} = \frac{\mathbf{w}_{a,i}^{+}}{\| \mathbf{w}_{a,i}^{+} \|} \tag{10.165}$$

以及

$$\mathbf{w}_{b,i} = \frac{\mathbf{w}_{b,i}^{+}}{\| \mathbf{w}_{b,i}^{+} \|} \tag{10.166}$$

这些归一值在算法的下一次迭代中使用。

对于包含数据模型的应用，此时有由空间变换数据构成的两个数据流，如图 10.19 所示。在两个数据流之间加强权值共享约束是有用的，此时设：

$$\mathbf{w}_{a,i} = \mathbf{w}_{b,i}, \quad \text{对于所有 } i \tag{10.167}$$

满足这一约束的明智的方法是利用由式(10.165)和式(10.166)计算的 $\mathbf{w}_{a,i}$ 和 $\mathbf{w}_{b,i}$ 的平均值。因此，通过对网络 a 和 b 分配相同的初始权值矩阵来开始相关 ICA 的权值自适应规则，在自适应规则的每一步权值共享性都得到保持。

为了描述相关 ICA 原则的重要实际应用，我们现在讨论相关 ICA 原则是如何对自然声音的听觉编码中学习滤波器提供一个计算工具的。

听觉编码：相关 ICA 作用于自然声音

在听觉系统的一些结构和函数专业限定中，时间是显而易见的。在听觉刺激的多个时间尺度上，我们发现区别一个听觉刺激波形的两个特定分量是有益的（Joris 等，2004）：

1. 载体，由波形的好的结构来表示，它以“调幅”方式增大和变小。

2. 包络，它是调幅波形的轮廓。

从调幅理论可知，信息承受信号（即调制信号）包含在受调信号的包络（envelope）中。从生理学的观点，对调幅的兴趣是由想要知道包络处理是否实际嵌入在听觉系统而激发的。

事实上，穿过多层听觉系统，存在与进来的调幅语音信号相应的神经元。特别地，听觉系统的连续层通过对不同局限范围的调幅率的响应区分开来：较底层通常响应于进来的听觉刺激能量中的快速变化，渐渐地较慢的变化在较高层中发生。根据这一事实，在声音感知中调幅被认为是一个重要的听觉提示就不奇怪了。

以听觉处理作为感兴趣的问题，我们将要讲述的问题如下所示：

1. 给定调幅语音信号的加性混合，我们如何分离独立分量的包络而忽略相关联的载体？

相关的问题如下所述：

2. 在自组织的方式下，我们如何学习在听觉系统中不同处理层响应于调幅刺激的过程的？

对这一基本问题的实验的答案可以从相关 ICA 中找到（Haykin and Kan，2007）。

在相关 ICA 中，目标是提取包含在通过分离源的保持了“相关”的信号，同时，和源相关联的通过网络的信息流被最大化。因为在调幅中，包络和载体相比缓慢变化，我们可以将调幅看成所考虑的包络范围内的时间相干性：即分别通过两个时间步 Δt 秒，假设 Δt 足够小，可以设 $x(t+\Delta t) \approx x(t)$。

在 Kan(2007)和 Haykin and Kan(2007)中，相关 ICA 算法被应用于英语演讲者们的语音样本集，这个集是从 TIMIT 数据库中取得[20]。这个实验说明了利用相关 ICA 学习的两层听觉处理的语音数据的滤波器集是平滑的且局限于时间的。更重要的是，实验的结果表明了两个重要的特征：

1. 两层中滤波器的带宽仅包含调制谱频率，忽略载体频率。

2. 第一层处理计算的基带（即基于调制的）滤波器具有 10 倍于第二层处理计算的基带滤波器的切断频率。换句话说，实验模型（基于相关 ICA）的第一层更多地响应于输入听觉信号的快速变化，反之，模型的第二层响应于输入的较慢变化。

简单地说，通过相关 ICA 学习的滤波器，当作用于自然声音的时候，基带滤波器显示为展示出相似于耳蜗核和下丘的生物神经元的性质。

10.19　速率失真理论和信息瓶颈

到目前为止，我们集中于信息论的两个基本概念（熵和互信息）作为学习信息论学习的两个支柱。在这一节中，我们转向信息论学习中另一个富有启发性的速率失真理论。在我们的思想中这一方法被称之为*信息瓶颈方法*，在 Tishby 等（1999）中首次提出。

速率失真理论，作为香农信息论（香农，1948）的固有部分，处理具有可能失真数据的压缩，其有目的的应用导致了总数可测量的数据失真。压缩数据的动机在于产生数据的新的流使得从平均上比原始的数据流需要更少的字位数目来表示或者传输。

为了给介绍信息瓶颈方法铺平道路，我们从速率失真理论开始讨论。

速率失真理论

给定一个信息源产生的数据流，速率失真理论的目的在于寻找在具体的信息流速率下可达到的失真的最小期望值，或者等价于，对于预定的失真层寻找可达到的信息流的最小速率。

若要从分析术语上来说明这一理论，令 $\mathbf{X}$ 记概率密度函数 $p_{\mathbf{X}}(\mathbf{x})$ 的一个随机向量，由一个信息源产生。相应地，令概率密度函数 $q_{\mathbf{T}}(\mathbf{t})$ 的随机向量 $\mathbf{T}$ 表示 $\mathbf{X}$ 的一个压缩版本。（注意我们对分布 $\mathbf{X}$ 和 $\mathbf{T}$ 采用不同的记号。）根据式(10.28)的最后一行，$\mathbf{X}$ 和 $\mathbf{T}$ 之间的互信息表示为

$$I(\mathbf{X};\mathbf{T}) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \underbrace{p_{\mathbf{X}}(\mathbf{x})q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})}_{\text{联合pdf}} \log\left(\frac{q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})}{q_{\mathbf{T}}(\mathbf{t})}\right) d\mathbf{x}d\mathbf{t}$$

其中 $q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})$ 是给定 $\mathbf{X}$ 后 $\mathbf{T}$ 的条件概率密度函数。关于向量 $\mathbf{X}$ 和 $\mathbf{T}$ 之间的距离测量，使用记号 $d(\mathbf{x},\mathbf{t})$，这里 $\mathbf{x}$ 和 $\mathbf{t}$ 分别记 $\mathbf{X}$ 和 $\mathbf{T}$ 的样本值。*期望失真定义为*：

$$\mathbb{E}[d(\mathbf{x},\mathbf{t})] = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \underbrace{p_{\mathbf{X}}(\mathbf{x})q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})}_{\text{联合pdf}} d(\mathbf{x},\mathbf{t})d\mathbf{x}d\mathbf{t} \tag{10.168}$$

速率失真理论自身通过一个称为*速率失真函数*的函数来刻画，记为 $R(D)$。

有了目前的记号背景，我们现在可以正式地表示速率失真理论如下（Cover and Thomas，2006）：

寻找速率失真函数

$$R(D) = \min_{q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})} I(\mathbf{X};\mathbf{T})$$

其失真约束为

$$\mathbb{E}[d(\mathbf{x},\mathbf{t})] \leqslant D$$

从这一陈述，很明显地计算速率失真函数 $R(D)$ 包含解下述约束优化问题：

在源及其表示之间最小化互信息，服从预定的失真约束。

这一优化问题能通过 Blahut-Arimoto 算法（Cover and Thomas 2006）来解，这是通过在两个未知分布的凸集之间的交互投影来做，如 10.21 节所讨论的那样。

速率失真理论的最大成果在于，证明速率失真函数是一个给定期望失真数据的任意描述的

速率（编码长度）的渐进可达到的下界。

信息瓶颈方法

信息瓶颈方法建立在速率失真理论基础上，通过“相关变量”信息[21]来代替失真项。在多个应用中，“真实失真”度量是未知的或者未定义的，但是关于其的另一个变量我们可以保留一些给定信息。语音识别问题是一个好的例子。在这一问题中，众所周知的困难是形成能正确捕捉人类声音感知的失真函数；给出一些对话字的样例以及它们的语音录制更容易。在这样的例子中，我们寻找高熵语音信号的压缩来尽可能多地保留低熵语音序列的信息。关于这一类协同出现数据的重要例子是那些其失真函数不能直接提供的例子：单词和主题，图像和物体，基因表达和组织样本，以及刺激和神经响应。信息瓶颈方法已经被成功应用于这类数据（Slonim 等，2006）。

信息瓶颈理论是通过引入记为 Y 的辅助（相关）随机向量来实现的。这一新的随机向量是（随机地）依赖于原始的，通常是高熵的随机向量 $\mathbf{X}$ 的。因此，互信息 $I(\mathbf{X};\mathbf{Y})$ 是非零的。

$\mathbf{X}$ 是将要压缩的随机向量，$\mathbf{Y}$ 是我们将预测的（或者是关于其我们将保持尽可能多的信息）随机向量。通过引入瓶颈随机向量 $\mathbf{T}$ 作为原始随机向量 $\mathbf{X}$ 的压缩表示，实际上，我们已经构造了两个信息量之间的权衡或瓶颈：一个是关于 X 包含在 $\mathbf{T}$ 中；另一个是关于 $\mathbf{Y}$ 包含在 $\mathbf{T}$ 中。

特别地，我们将通过满足组合方式下的两个目标来解信息瓶颈：

1. 将原始（高熵）随机向量 $\mathbf{X}$ 的样本值按照这样的方法分解：关于相关随机向量 $\mathbf{Y}$ 保存尽可能多的互信息。

2. 关于原始随机向量 $\mathbf{X}$ 损失尽可能多的信息来获得最小分解的最简单形式。

因此，在 $\mathbf{X}$ 的所有特征表示特性中，问题在于决定仅有的那些和 $\mathbf{Y}$ 的预测最相关的特性。

基本上，信息瓶颈理论被设计为寻找最优相关数据表示。问题如下：

给定随机向量 $\mathbf{X}$ 和相关随机向量 Y 的联合概率密度函数，在 $\mathbf{X}$ 的关于 $\mathbf{Y}$ 提供信息的样本值中提取最小充分分解，通过未知分布 $q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})$ 来最小化如下信息瓶颈函数来寻找瓶颈随机向量 $\mathbf{T}$。

$$J(q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})) = I(\mathbf{X};\mathbf{T}) - \beta I(\mathbf{T};\mathbf{Y}) \tag{10.169}$$

服从 $\mathbf{T}$ 依赖于 $\mathbf{X}$ 且 $\mathbf{Y}$ 依赖于 $\mathbf{T}$ 的需求，且服从归一化约束。

正的拉格朗日乘子 β 是压缩（最小表达）和可预测性（信息保存）之间的权衡参数。通过在 0 和无穷大之间变化这一参数，可获得一个凹的信息曲线，类似于速率失真函数，它提供了压缩和预测之间的最优可达权衡。

例 11　高斯信息瓶颈

对于信息瓶颈方法的一个分析处理，对于对数函数的导数，我们可以考虑特征向量问题的耦合对：

$$\frac{\partial}{\partial \mathbf{t}}\log p_{\mathbf{X}|\mathbf{T}}(\mathbf{x}|\mathbf{t}) \text{ 和 } \frac{\partial}{\partial \mathbf{t}}\log p_{\mathbf{Y}|\mathbf{T}}(\mathbf{y}|\mathbf{t})$$

通常因为解决这类问题较困难，我们转向分析上易处理的例子，此时，原始随机向量 $\mathbf{X}$ 和其压缩版本 $\mathbf{Y}$ 通过联合多变量高斯分布来描述，如在 Chechik 等（2004）中那样。在高斯框架下解特征向量问题的耦合对有助于典范相关分析（CCA），正如 10.10 节所述，这是 Imax 原则的一个特例。我们因而发现要解决的这一问题是寻找对子空间的线性投影，其维数是由权衡参数 β 决定。特别地，随着参数 β 增长，附加维数（即特征值）被添加到投影（瓶颈）向量 $\mathbf{T}$ 中；通过一系列临界点或结构相变这一附加表明了其自身，同时每个基向量的相关欧几里得范数被重

定比例。继续这一维数扩展过程直到关于压缩向量 $\mathbf{Y}$ 的相关信息被捕捉到瓶颈向量 $\mathbf{T}$ 中。这一过程的网络结果是在信息论项中对于变化 β 而言信息瓶颈方法是如何提供一个连续模型复杂度测量的洞察性说明。

对于在 Chechik 等（2004）中研究的高斯框架，图 10.21 画出了对于变化的 β 而言互信息 $I(\mathbf{T};\mathbf{Y})$ 和互信息 $I(\mathbf{T};\mathbf{X})$ 之间的图形。在图 10.21 中连续平滑曲线表示信息曲线，是从 4 个特征值 $\lambda_i=0.1, 0.5, 0.7, 0.9$ 中获得的。相应地，在图中用小圆圈来表示临界点。信息曲线（通过这些临界点）从几个分段中构造，实现了随着互信息 $I(\mathbf{T};\mathbf{X})$ 的增长，附加特征向量被用于投影。为了比较，图 10.21 也给出了每个 β 用小数目的特征向量计算的信息曲线。

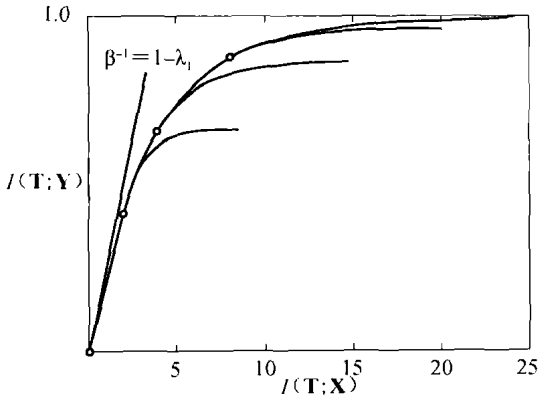


图 10.21　多变量高斯变量的信息曲线。包络是最优压缩-预测权衡，通过从 0 到无穷大改变拉格朗日乘子 β 来捕捉。在每一点曲线的斜率由 $1/\beta$ 给定。总是存在 β 的临界低值决定了原点的斜率，低于其仅有平凡解。次优曲线是在 $\mathbf{T}$ 的维数限制在固定低值时获得（这一图的复制得到了 Naftali Tishby 博士的允许）

由图 10.21 中的结果可知，高斯信息瓶颈方法信息曲线是处处凹的。在互信息 $I(\mathbf{T};\mathbf{X})$ 上的每一个值，信息曲线被切线所界，其斜率由函数 $\beta^{-1}(I(\mathbf{T};\mathbf{X}))$ 所定义。在原点，$I(\mathbf{T};\mathbf{X})=0$，斜率 $\beta^{-1}(0)=1-\lambda_1$，其中 λ_1 是原始随机向量 $\mathbf{X}$ 及其压缩版本 $\mathbf{Y}$ 的典范相关分析的第一个特征值。注意信息曲线的渐进斜率是 0，即 $\beta\to\infty$。这一逼近行为简单地反映了报酬渐减律的实现：在原始随机向量 $\mathbf{X}$ 的描述中增加更多的字位信息对于瓶颈向量 $\mathbf{T}$ 不提供增加的精确度。 ■

信息瓶颈方程

信息瓶颈最优问题的解是通过下列描述向量 T 的瓶颈方程来给出的：

$$q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x}) = \frac{q_{\mathbf{T}}(\mathbf{t})}{Z(\mathbf{x},\beta)}\exp(-D_{p\|q}) \tag{10.170}$$

$$q_{\mathbf{T}}(\mathbf{t}) = \sum_{\mathbf{x}} q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})\,p_{\mathbf{X}}(\mathbf{x}) \tag{10.171}$$

$$q_{\mathbf{Y}|\mathbf{T}}(\mathbf{y}|\mathbf{t}) = \sum_{\mathbf{x}} q_{\mathbf{Y}|\mathbf{T}}(\mathbf{y}|\mathbf{t})\,q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})\left(\frac{p_{\mathbf{X}}(\mathbf{x})}{q_{\mathbf{T}}(\mathbf{t})}\right) \tag{10.172}$$

在式(10.170)中，$D_{p\|q}$ 记两个条件概率密度函数 $p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x})$ 和 $q_{\mathbf{Y}|\mathbf{T}}(\mathbf{y}|\mathbf{t})$ 之间的相对熵，$Z(\mathbf{x},\boldsymbol{\beta})$ 是归一化（分解）函数。图 10.22 描述了在这三个方程的启发下的信息瓶颈思想。

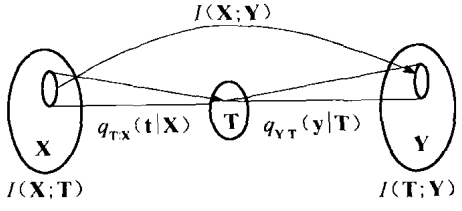


图 10.22　信息瓶颈方法的一种描述。瓶颈 $\mathbf{T}$ 捕捉原始随机向量 $\mathbf{X}$ 关于相关变量 $\mathbf{Y}$ 的相关部分，通过最小化信息 $I(\mathbf{X};\mathbf{T})$ 的同时保持 $I(\mathbf{T};\mathbf{Y})$ 尽可能高。瓶颈 $\mathbf{T}$ 由三个分布 $q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})$，$q_{\mathbf{T}}(\mathbf{t})$ 和 $q_{\mathbf{Y}|\mathbf{T}}(\mathbf{Y}|\mathbf{t})$ 决定，这表示了瓶颈方程(10.170)到(10.172)的解

从式(10.170)到式(10.172)的系统，我们必须对于三个未知分布 $q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})$，$q_{\mathbf{T}}(\mathbf{t})$ 和 $q_{\mathbf{Y}|\mathbf{T}}(\mathbf{y}|\mathbf{t})$ 分别独立地求解。Tishby 等（1999）证明了通过从一个随机分布开始以和速率失真理论的 Blahut-Arimoto 迭代相似的方式迭代这些方程，方程收敛到参数 β 的任意值的最优解。

信息瓶颈问题能用于解决获得相关连续流形（维数削减），如 Chechik 等（2004）对于高斯变量所示，或者如下一节根据 Chigirev and Bi-

alek(2004) 所讨论的那样。

10.20　数据的最优流形表达

在第 7 章中，我们从正则化的角度使用光谱图理论讨论了数据的非监督流形表达。在这一节中，我们回顾同样的问题，但这一次我们从信息论的角度来达到这一点。更具体地说，这里采用的方法来自于 Chigirev and Bialek(2004)，它建立在如下的理解上：

将维数削减视为数据压缩问题可以获得分析上的利益。

数据表达的 Chigirev-Bialek 方法实际上是前一节讨论的信息瓶颈方法的明智的应用。

视为数据压缩的维数削减：基本公式

从第 7 章的讨论我们回顾到，从直观上，流形是指一个嵌入在 m -维欧几里得空间中的 k -维连续区域（例如，一个曲线或一个曲面），其中 k 是小于 m 的。在数据的流形表达中，我们想象流形维数低于数据维数。尤其这个流形“几乎完美”地描述了数据，因为不可避免地存在加性噪声和其他形式的数据退化。

令 $\mathcal{M}$ 记一个维数为 k 的流形，$q_{\mathcal{M}}(\mathbf{u})$ 记流形上点的概率密度函数；$\mathbf{u}$ 表示这样的一个点。令 $\mathbf{X}$ 记一个 m 维的随机数据向量，m 大于 k，这实际上暗示了由 $\mathbf{X}$ 表示的数据集合 $\mathcal{X}$ 是稀疏的。而且，正由于数据集合的稀疏性使得其非监督表示成为一个具挑战性的任务。令 $q_{\mathcal{M}|\mathbf{X}}(\mathbf{u}|\mathbf{x})$ 记给定数据集 X 时流形上点的条件概率密度函数。因此，随机映射

$$P_{\mathcal{M}} : \mathbf{x} \rightarrow q_{\mathcal{M}|\mathbf{X}}(\boldsymbol{\mu} \mid \mathbf{x}) \tag{10.173}$$

描述了从 $\mathbf{x}$ 到 $\mathbf{u}$ 的映射。

流形是由 $\{\mathcal{M}, P_{\mathcal{M}}\}$ 表示的，这隐含了数据集 $\mathcal{X}$ 的“小于可靠的表达”，证实了上面所做的相似备注。从另一个途径，可以说记流形 $\mathcal{M}$ 的一个点的向量 $\mathbf{u}$ 是数据点 $\mathbf{x}$ 的失真版本——因此需要一个记为 $d(\mathbf{x},\boldsymbol{\mu})$ 的距离测量。为了简化问题，我们对这一测量采用欧几里得距离函数，如下所示：

$$d(\mathbf{x},\boldsymbol{\mu}) = \| \mathbf{x} - \boldsymbol{\mu} \|^2 \tag{10.174}$$

这是通常所使用的距离。因此期望失真被定义为双重多维积分：

$$\mathbb{E}[d(\mathbf{x},\boldsymbol{\mu})] = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} p_{\mathcal{X}}(\mathbf{x}) q_{\mathcal{M}|\mathbf{X}}(\boldsymbol{\mu}|\mathbf{x}) \| \mathbf{x} - \boldsymbol{\mu} \|^2 \mathrm{d}\mathbf{x}\mathrm{d}\boldsymbol{\mu} \tag{10.175}$$

其中 $p_{\mathcal{X}}(\mathbf{x})$ 是数据集 $\mathcal{X}$ 的概率密度函数，其样本值由数据点 $\mathbf{x}$ 来表示。

式(10.175)是数据压缩问题的一个重要方面。第二个重要方面是流形 $\mathcal{M}$ 和数据集 $\mathcal{X}$ 之间的互信息，定义为：

$$I(\mathcal{X};\mathcal{M}) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \underbrace{p_{\mathcal{X}}(\mathbf{x}) q_{\mathcal{M}|\mathbf{X}}(\boldsymbol{\mu}|\mathbf{x})}_{\text{联合pdf}} \log\left(\frac{q_{\mathcal{M}|\mathbf{X}}(\boldsymbol{\mu}|\mathbf{x})}{q_{\mathcal{M}}(\boldsymbol{\mu})}\right) \mathrm{d}\mathbf{x}\mathrm{d}\boldsymbol{\mu} \tag{10.176}$$

当对数是以 2 为基数时，这一互信息定义了将数据点 $\mathbf{x}$ 编码到流形 $\mathcal{M}$ 上点 $\mathbf{u}$ 所需要的字位的个数。此外，通过将维数削减视为数据压缩问题，$I(\mathcal{M}; M)$ 定义了给定数据向量 $\mathbf{x}$ 作为输入时需要传输压缩数据 $\mathbf{u}$ 的频道“容量”。

当放在一起看时，式(10.175)和式(10.176)呈现出包含两个基本问题的权衡：

1. 关于数据的一个“可靠的”流形表达，需要最小化式(10.175)的期望失真。

2. 另一方面，对于一个“好的”将数据压缩到流形上的点的压缩而言，需要最大化由式(10.176)定义的互信息。

为了解这一权衡，我们引入最优流形的概念 (Chigirev and Bialek, 2004)：

给定数据集$\mathscr{X}$和频道容量$I(\mathscr{X};M)$，如果下面的两个条件得到满足则流形M被称为数据集X的最优表达：

(i) 期望失真$\mathbb{E}[d(\mathbf{x},\boldsymbol{\mu})]$被最小化。

(ii) 仅由频道容量$I(\mathscr{X};M)$定义的字位数需要用于表示数据点$\mathbf{x}$。

定义最优流形的另一个途径，如下所示：

流形$\mathcal{M}$是最优的，如果频道容量$I(\mathscr{X};\mathcal{M})$在期望失真固定在某个预先指定的值时最大化。

不管哪一种途径，我们都面对速率失真理论中的问题。根据10.19节的讨论，由于这一问题是约束优化问题，我们引入拉格朗日乘子λ来说明期望失真和频道容量之间的权衡，如下所示：

$$F(\mathcal{M},P_{\mathcal{M}})=\mathbb{E}[d(\mathbf{x},\boldsymbol{\mu})]+\lambda I(\mathscr{X};\mathcal{M}) \tag{10.177}$$

为了找到最优流形，必须最小化这一函数。

要从分析术语上来实现最小化，我们需要参数化流形。根据10.19节的信息瓶颈方法，引入瓶颈向量$\mathbf{T}$，它的一个样本值记为$t\in\mathbb{R}^l$，这里新的维数l小于或等于数据向量x的维数m。我们也引入一个新的向量值函数：

$$\boldsymbol{\gamma}(\mathbf{t}):\mathbf{t}\to\mathcal{M} \tag{10.178}$$

这将由瓶颈向量$\mathbf{T}$张成的参数空间的点$\mathbf{t}$映射到流形M。因而向量值函数$\mathbf{r}(\mathbf{t})$是流形M的一个“描述符”。假设$\mathbf{r}(\mathbf{t})$的维数和数据点$\mathbf{x}$的维数相同，因此可以用平方欧几里得距离$\|\mathbf{x}-\boldsymbol{\mu}(\mathbf{t})\|^2$作为使用流形$M$表达数据集$\mathscr{X}$时产生的失真的新的测量。

根据刚刚讨论过的流形参数化，我们重新将两个基本公式(10.175)和式(10.176)分别表示为新的形式：

$$\mathbb{E}[d(\mathbf{x},\boldsymbol{\gamma}(\mathbf{t}))]=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}p_{\mathbf{X}}(\mathbf{x})q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})\|\mathbf{x}-\boldsymbol{\gamma}(\mathbf{t})\|^2\mathrm{d}\mathbf{x}\mathrm{d}\mathbf{t} \tag{10.179}$$

$$I(\mathbf{X};\mathbf{T})=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}p_{\mathbf{X}}(\mathbf{x})q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})\log\left(\frac{q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})}{q_{\mathbf{T}}(\mathbf{t})}\right)\mathrm{d}\mathbf{x}\mathrm{d}\mathbf{t} \tag{10.180}$$

相应地，式(10.177)的函数F重新写为新的形式：

$$F(\boldsymbol{\gamma}(\mathbf{t}),q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x}))=\mathbb{E}[(\mathbf{t}|\mathbf{x},\boldsymbol{\gamma}(\mathbf{t}))]+\lambda I(\mathbf{X};\mathbf{T}) \tag{10.181}$$

在后一公式中期望失真和频道容量都是由$\{\mathcal{M},P_{\mathcal{M}}\}$描述的流形的固有性质，且这些性质在再参数化时是不变的。

通过式(10.179)和式(10.181)，现在可以寻找最优流形。通过应用下面两个优化条件来实现：

$$\frac{\partial F}{\partial\boldsymbol{\gamma}(\mathbf{t})}=\mathbf{0}\quad\text{，对于固定的 }q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x}) \tag{10.182}$$

$$\frac{\partial F}{q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})}=0\quad\text{，对于固定的 }\boldsymbol{\gamma}(\mathbf{t}) \tag{10.183}$$

因此，应用条件1，获得：

$$\int_{-\infty}^{\infty}p_{\mathbf{X}}(\mathbf{x})q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})(-2\mathbf{x}+2\boldsymbol{\gamma}(\mathbf{t}))\mathrm{d}\mathbf{x}=\mathbf{0}$$

这导致下面的方程对，从概率术语上讲它们是相容的：

$$\boldsymbol{\gamma}(\mathbf{t})=\frac{1}{q_{\mathbf{T}}(\mathbf{t})}\int_{-\infty}^{\infty}\mathbf{x}p_{\mathbf{X}}(\mathbf{x})q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})\mathrm{d}\mathbf{x} \tag{10.184}$$

$$q_{\mathbf{T}}(\mathbf{t})=\int_{-\infty}^{\infty}p_{\mathbf{X}}(\mathbf{x})q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})\mathrm{d}\mathbf{x} \tag{10.185}$$

方程对的推导仅仅从函数F的期望-失真分量而来，因为仅有这一分量依赖于$\mathbf{r}(\mathbf{t})$——因此缺

少了拉格朗日乘子 λ。

然而，当继续应用式(10.183)定义的第二个最优条件时，我们必须认识到这一最优化包含了条件 $q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})$ 在下述约束下的所有可能值：

$$\int_{-\infty}^{\infty} q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})\mathrm{d}\mathbf{t} = 1, \quad \text{对于所有 } \mathbf{x}$$

该约束仅仅是需要在曲线 $q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})$ 下的区域是单位 1，这是每一个概率密度函数的基本性质。为了满足这一附加约束，我们对所有 $\mathbf{x}$ 引入新的拉格朗日乘子 $\beta(\mathbf{x})$ 并因此扩展函数 F 的定义来获得：

$$F(\boldsymbol{\gamma}(t), q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\Big\{ p_{\mathbf{X}}(\mathbf{x})q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x}) \| \mathbf{x}-\boldsymbol{\gamma}(\mathbf{t}) \|^2 + \lambda p_{\mathbf{X}}(\mathbf{x})q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})\log\Big(\frac{q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})}{q_{\mathbf{T}}(\mathbf{t})}\Big) + \beta(\mathbf{x})q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})\Big\}\mathrm{d}\mathbf{t}\mathrm{d}\mathbf{x} \tag{10.186}$$

其中 $q_{\mathbf{T}}(\mathbf{t})$ 如式(10.185)中定义的那样。

因此，引用式(10.183)的第二个最优条件到函数 F 的这一新的形式中并且通过式(10.185)来简化项，可得：

$$\frac{1}{\lambda}\| \mathbf{x}-\boldsymbol{\gamma}(\mathbf{t}) \|^2 + \log\Big(\frac{q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})}{q_{\mathbf{T}}(\mathbf{t})}\Big) + \frac{\beta(\mathbf{x})}{\lambda p_{\mathbf{X}}(\mathbf{x})} = 0$$

现在，令

$$\frac{\beta(\mathbf{x})}{\lambda p_{\mathbf{X}}(\mathbf{x})} = \log Z(\mathbf{x},\lambda) \tag{10.187}$$

并且对于期望条件 $q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})$ 解结果方程，得到第二个公式对，它们在概率术语上也是相容的：

$$q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x}) = \frac{q_{\mathbf{T}}(\mathbf{t})}{Z(\mathbf{x},\lambda)}\exp\Big(-\frac{1}{\lambda}\| \mathbf{x}-\boldsymbol{\gamma}(\mathbf{t}) \|^2\Big) \tag{10.188}$$

和

$$Z(\mathbf{x},\lambda) = \int_{-\infty}^{\infty} q_{\mathbf{T}}(\mathbf{t})\exp\Big(-\frac{1}{\lambda}\| \mathbf{x}-\boldsymbol{\gamma}(\mathbf{t}) \|^2\Big)\mathrm{d}\mathbf{t} \tag{10.189}$$

函数 $Z(\mathbf{x}, \lambda)$ 扮演了*归一（分解）函数*的角色，式(10.188)中包含了该项保证了加于 $q_{\mathbf{T}}(\mathbf{t})$ 的约束得到满足。

式(10.184)、式(10.185)、式(10.188)和式(10.189)在非监督方式下描述了*数据表达的最优流形*。该描述自然需要连续概率密度函数 $p_{\mathbf{X}}(\mathbf{x})$ 的知识。

离散过程

然而，在实际上，我们仅有记为 $\{x_i\}_{i=1}^{N}$ 的训练样本$\mathscr{X}$，其中 $\mathbf{N}$ 是样本大小。根据这一实际情况，我们引入*离散逼近*：

$$p_{\mathbf{X}}(\mathbf{x}) \approx \frac{1}{N}\sum_{i=1}^{N}\delta(\mathbf{x}-\mathbf{x}_i) \tag{10.190}$$

其中 $\delta(\cdot)$ 记 Dirac delta *函数*。相应地，通过下面的离散集来模型化流形$\mathcal{M}$：

$$\mathcal{J} = \{\mathbf{t}_j\}_{j=1}^{L} \tag{10.191}$$

然后，注意到瓶颈向量 $\mathbf{T}$ 的样本值 $\mathbf{t}$ 仅仅显示为函数 $\mathbf{r}(\mathbf{t})$、条件 $q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})$ 和边缘 $q_{\mathbf{T}}(\mathbf{t})$ 的自变量，我们可以用这三个连续函数的各自的离散部分 γ_j，$q_i(\mathbf{x}_i)$，q_j 来代替它们，其中下标 i 和 j 用于强调离散过程。为了完成离散过程，引入 α 来记欧几里得空间 $\mathbb{R}^m$ 的坐标索引。

现在有了流形的离散模型，目标是在迭代方式下开发用于计算模型的算法。为了该目的，首先注意式(10.188)和式(10.189)分别定义了 $q_{\mathbf{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})$ 和 $Z(x, \lambda)$，对他们各自的变量 $\mathbf{t}$ 和 $\mathbf{x}$ 都是凸函数；拉格朗日乘子 λ 是预先定义的参数。从计算上讲，这两个公式是流形的离散模型

的困难部分。

为了更进一步说明如何能够降低这一计算困难，考虑如图 10.23 所示的两个凸集$\mathcal{A}$和$\mathcal{B}$。将最小化它们之间的欧几里得“距离”；这一距离定义为 $d(x, y)$，其中 x 和 y 分别是集合$\mathcal{A}$和$\mathcal{B}$中任意的两个点。最小化欧几里得距离的直观方法如下所述（Csiszat and Tusnady，1984）：

固定集合 A 中的点 $\mathbf{x}$，寻找集合 B 中最靠近它的点 $\mathbf{y}$。然后固定新发现的点 $\mathbf{y}$，在集合 A 中寻找最靠近它的点 $\mathbf{x}$。

如果用往返于集合$\mathcal{A}$和$\mathcal{B}$之间的方式来延续这一过程，正如图 10.23 所示，那么距离 $d(\mathbf{x}, \mathbf{y})$将随着每次迭代而逐渐变小。这正是在最小化速率失真函数的 Blahut-Arimoto 算法（Blahut，1972；Arimoto，1972）中所做的那样。式(10.188)和式(10.189)具有和速率失真函数的特征中发现的那些具有相同的数学形式(Cover and Thomas，2006)。而且，在 Csiszar and Tusnady(1984) 中证明了在两个凸集$\mathcal{A}$和$\mathcal{B}$之间的交替过程将收敛，如果这两者都是概率分布集合且距离测量采用两个分布之间的相对熵。

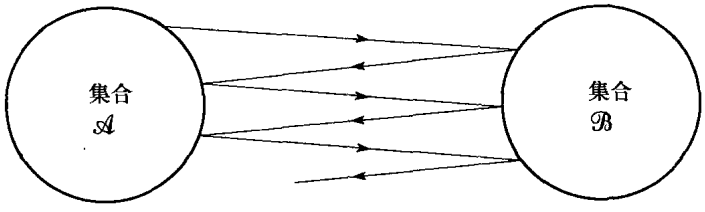


图 10.23　计算两个凸集$\mathcal{A}$和$\mathcal{B}$之间距离的交替过程图示

计算数据的最优流形表示的迭代算法

根据这些可靠的结果，我们可以继续构成计算流形$\mathcal{M}$的离散模型的迭代算法。令 n 记迭代算法的时间步。然后，利用式(10.184)、式(10.185)、式(10.188)和式(10.189)的离散版本并采用 L -点离散集合 $\{\mathbf{t}_1, \mathbf{t}_2, \cdots, \mathbf{t}_L\}$ 来模型化由连续变量 t 表达的流形，我们现在就构成了基于下面 4 个公式集的期望算法，其中时间步 $n = 0,1,2,\cdots$，且索引 $j = 1,2,\cdots,L$(Chigirev and Bialek,2004)：

$$p_j(n) = \frac{1}{N}\sum_{i=1}^{N} p_j(\mathbf{x}_i, n) \tag{10.192}$$

$$\boldsymbol{\gamma}_{j,\alpha}(n) = \frac{1}{p_j(n)} \cdot \frac{1}{N}\sum_{i=1}^{N} x_{i,\alpha} p_j(\mathbf{x}_i, n), \alpha = 1,2,\cdots,m \tag{10.193}$$

$$Z(\mathbf{x}_i, \lambda, n) = \sum_{j=1}^{L} p_j(n)\exp\left(-\frac{1}{\lambda}\| \mathbf{x}_i - \boldsymbol{\gamma}_j(n) \|^2\right) \tag{10.194}$$

$$p_j(\mathbf{x}_i, n+1) = \frac{p_j(n)}{Z(\mathbf{x}_i, \lambda, n)}\exp\left(-\frac{1}{\lambda}\| \mathbf{x}_i - \boldsymbol{\gamma}_j(n) \|^2\right) \tag{10.195}$$

其中 $x_{i,\alpha}$为数据向量 $\mathbf{x}_i$ 的第 α 个元素。

为了*初始化*算法，我们从数据集$\mathcal{X}$中随机选取 L 个点且令：

$$\left.\begin{aligned} \boldsymbol{\gamma}_j &= x_{i,j} \\ p_j(0) &= \frac{1}{L} \end{aligned}\right\} \quad j = 1,2,\cdots,L \tag{10.196}$$

为了*终止*计算，令 ϵ 记流形点将要位于的精确度。在时间步长为 n 时，一旦满足下述条件，算法就得终止

$$\max_j |\boldsymbol{\gamma}_j(n) - \boldsymbol{\gamma}_j(n-1)| < \epsilon$$

余下需要设置的参数是拉格朗日乘子 λ，它决定了包含在函数 F 中的期望失真和频道容量之间的权衡。参数处于设计者的控制下，依赖于这样的权衡是如何实现的。

实际考虑

式(10.192)到式(10.195)的计算数据的最优流形表达的算法，是设计于约束流形点和原始

数据空间点之间的互信息。这一约束是关于这两个空间中所有的可逆坐标变换不变的——可能在某种隐含意义上增强流形的平滑性（Chigirev and Bialek，2004）。从理论框架上来看，利用信息论方法的平滑流形的证明可能不如根植于正则理论的方法。虽然如此，数据的最优流形表达从实际上工作满意。

更重要的是，不像其他维数削减方法（例如，第 7 章讨论的基于正则化光谱图理论的 Belkin-Niyogi 方法），本节中讲述的信息论算法的收敛时间对样本大小 N 是线性的。这一算法的高度期望特征属于描述流形的公式的固有凸性，使得其应用更具吸引力，尤其当我们处理实际中大型数据集的维数削减的困难任务时更是如此。

算法的另一高度期望特征包括下面两点：

- 所考虑的流形的维数知识是不需要的。
- 这一算法很好适用于处理稀疏数据的维数削减，这一点是重要的，因为在高维空间中所有的数据集都是典型稀疏的。

10.21　计算机实验：模式分类

该计算机实验利用了两个算法的组合：首先是用于非监督聚类的输入数据的最优流形表达，其次是采用在第 3 章讲述过的监督分类的最小均方（LMS）算法。通过不同的应用，这两个算法分享了两个有用的性质：有效性能和计算高效。

为了研究组合“最优流形-LMS”算法的性能，我们再次从图 1.8 的双月结构中随机提取数据，其双月之间的垂直分隔固定为 $d=-6$。图 10.24 给出了实验结果，通过双月之间近乎相等共享的 20 个中心来计算。在用 300 个数据点进行监督训练下算法构造的决策边界将从双月中提取的数据“几乎无瑕”的方式分隔开。更精确地，在 2 000 个测试数据点中有 6 个分类错误，说明了误分类错误率为 0.3%。对双月配置的相同设置而言，这一性能接近于支持向量机（SVM）的无误性能，这在 6.7 节中已经介绍过。从这一比较中得到的重点是在部分 SVM 的计算复杂度的基础上，最优流形-LMS 算法达到了和 SVM 接近的性能。

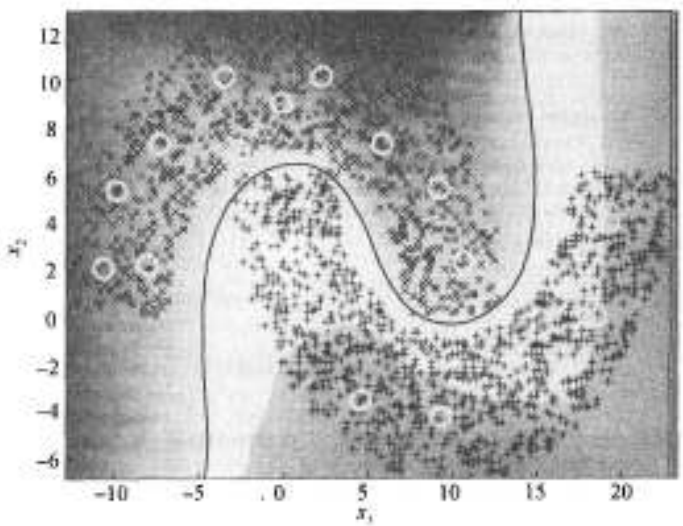


图 10.24　图 1.8 中双月构造的模式分类，使用最优流形+LMS 算法，距离 $d=-6$，有 20 个中心

10.22　小结和讨论

在篇幅较长的本章中，我们将香农信息论作为研究自组织或者非监督学习的多个重要方面

的基本统计工具来建立——这是真正值得注意的成果。

作为自组织目标函数的互信息

在输入和输出随机过程之间的香农互信息，具有一些唯一的性质，这些性质使其可作为自组织学习的目标函数，从而被优化。事实上，一些重要的自组织原则在本章的讨论中已经出现过：

1. Infomax 原则，这包含了最大化神经网络的多维输入和输出向量之间的互信息。这一原则制定了自组织模型和特征映射的开发框架。

2. 最小冗余原则，这基本上是另一种最大化网络的输入和输出之间的互信息导致冗余最小化的说法。

3. Imax 原则，这是最大化一对神经网络的单一输出之间的互信息，这对神经网络是由两个空间位移多维输入向量所驱动的。该原则非常适合于图像处理，目标是发现带噪声传感的输入在空间和时间上表现的相干性。

4. Imin 原则，这是最小化一对神经网络的单一输出之间的互信息，这对神经网络是由两个空间位移多维输入向量所驱动的。该原则在图像处理中的应用目标在于最小化同一环境中两幅相关图像之间的空间时间相干，图像是由具有正交性质的一对传感器获得的。

独立分量分析的两个基本路径

本章中讨论的另一个重要的课题是独立分量分析（ICA），它为使得一个随机向量的分量尽可能地统计独立提供了数学基础。这一原则的应用在于解盲源分离（BSS）问题，其必要条件如下所示：

- 信号的统计独立源。
- 非高斯源信号，除非被允许是高斯分布的。
- 平方混合矩阵，这意味着源信号和观测在数字上是相同的。
- 无噪混合模型。

基本上，推导 ICA 算法有两种途径：

1. 独立分量分析原则（Comon,1994）。建立在相对熵基础上；这一原则导致依赖于如下两个分布的期望代价函数的建立：

- 分离器输出的参数概率密度函数。
- 相应的阶乘分布。

独立分量分析原则的应用在两个著名算法中得以表明：

(i) ICA 的自然梯度算法，这是根据 Amari 等（1996）。

(ii) ICA 算法的 Infomax 原则，这是根据 Bell and Sejnowski (1995)。

这两个算法的主要优点是它们能够适应环境的统计变化。如果使用了正确类型的激活函数，它们也能够具有鲁棒性，这依赖于原始信号是超高斯分布的或是亚高斯分布的。

2. 最大负熵原则（Comon,1994）。负熵的记号提供了随机变量的非高斯性的测量。一个分量集的统计独立是通过负熵最小化来实现的。第二个原则的应用导致 FastICA 算法的建立，这是根据 Hyvärinen and Oja(1997)。FastICA 算法的有吸引力的特征包括：

- 收敛的快速速率。
- 无需学习率参数。
- 鲁棒性，无需源是否超高斯或者亚高斯分布的信息。
- 实现简单性。

然而，由于缺少学习率参数，FastICA 算法不能跟踪时间变化混合。

在三个不同的 ICA 算法中存在的一个问题，如下描述：

在一个大的 ICA 框架下，互信息、熵和非高斯性之间具有什么联系？在没有采取某种去相关约束的情况下。

为了处理 ICA 理论的这一基本问题，Cardoso(2003) 提供了一个了不起的数学理解，在此范围内统计相关、相关性、高斯性等问题得到了考虑。下面是 Cardoso 的论文中报告的主要结果：

当放松了预白化的需要后，相对熵能够在线性变换下被分解为两个项的和：一项表示分量的去相关，另一项表示其非高斯性。

通过限制到线性变换，ICA 实际上允许非高斯分量仅在边缘分布上表示。

关于 ICA 和 BSS 的更多的评论是，这两个概念彼此如此相近以至于使用其中一种时实际上意味着另一种。更重要的是，ICA 和 BSS 构成了一个已经扩大的领域，在理论上和实际应用上都是如此。这一声明已经通过一些给人印象深刻的主题得到了证明，这些主题中的每一个都有其自身的实际的与众不同的方向。(参看注释和参考文献中的注释 22。)

相关 ICA

本章中讨论过的另一个 ICA 相关的原则是相关 ICA(Kan，2007；Haykin and Kan，2007)。该新原则将 Infomax 和 Imax 原则组合起来最大化通过一对具有相同维数的多输入多输出 (MIMO) 网络的输出的时空相干，当这一网络是由空间位移数据流驱动时。利用现实数据，在自然声音的听觉编码中发现两个重要结果：

(i) 相关 ICA 能够展示调幅调节，因此支持包含听觉系统的包络处理概念。

(ii) 相关 ICA 能够学习响应于模拟分层听觉系统方式的声音刺激的滤波器的两个接连处理层的变化速率。

信息瓶颈

在一种或另一种形式下，这里所总结的自组织的信息论原则都是建立在熵和互信息概念上的，它们是香农经典信息论的基础。在本章的后面部分，我们利用速率失真理论（香农信息论的另一个基本概念）来构成本章的最后一个原则：信息瓶颈方法（Tishby 等，1999；Slonim 等，2006)。要强调的这一方法的两个重要方面如下所示：

1. 信息瓶颈方法不是统计模型算法；相反，它是寻找能够解释内在结构和给定变量集之间的统计相关的复杂数据的相关表达的方法。

2. 尽管该方法假设在输入向量 X 和输出向量 Y 之间的联合概率分布 $p_{X,Y}(\mathbf{x}, \mathbf{y})$，在实际中它被应用到基于有限样本的经验分布上。这一插入方法在 Shamir 等 (2008) 中得到了证明，其中提出了关于学习、泛化和一致性的定理。

有了信息瓶颈方法，我们利用其推导数据的最优流形表达 (Chigirev and Bialek，2004)。实现这一表达的该算法具有一些有用的性质：

- 算法的计算复杂度是线性的，它是关于训练样本大小的。
- 算法不需要流形维数的知识。
- 算法非常适合于处理高维数据，这些高维数据往往是稀疏的。

作为结束评论：在本节中总结的内容的宽度和深度是关于香农信息论的值得注意的影响的证明，香农信息论一开始是用于通信系统的，现在已经对非监督学习模型和其应用具有重要影响。

注释和参考文献

1. 香农信息论

想进一步了解信息论，请参考 Cover and Thomas(2006) 相关内容；如果想参考信息论发展的论文集 (包括

1948 年香农的经典论文），可参考 Slepian(1973)。香农的论文经过一些小的改动被重版在 Shannon and Weaver(1949)和 Sloane and Wyner(1993) 的书中。

想对在神经处理中的信息论原则作一个简短的回顾，可参考 Atick(1992)。想从生物的角度来理解信息论方法，可参考 Yockey(1992)。

2. 信息论与感知之间关系的文献综述可以参考 Linsker(1990b) 和 Atick(1992)。

3. 熵

信息论中的术语“熵”的名字是从热力学中的熵衍生来的；热力学中的熵由

$$H = -k_B \sum_{\alpha} p_\alpha \log p_\alpha$$

定义，其中 k_B 是 Boltzmann 常数，p_α 是系统处于状态 α 的概率（见第 11 章）。除了系数 k_B 之外热力学中的熵 H 的公式与式(10.8)给出的熵的定义在数学形式上是一致的。

4. 最大熵原则

Shore and Johnson(1980)中证明在如下意义下最大熵原则是正确的：

以约束形式给出先验知识，在满足这些约束的分布中根据“相容性公理”(consistency axioms) 能够选择唯一的分布；这个唯一的分布由最大化熵定义。

相容性公理包含四个部分：

Ⅰ. 唯一性：结果必须是唯一的。

Ⅱ. 不变性：坐标的选择应当不影响结果。

Ⅲ. 系统独立性：无论用不同密度或用联合密度来解释独立系统的独立信息都应该是无关紧要的。

Ⅳ. 子集独立性：无论用分离的条件密度或用完整的系统密度来处理独立的系统状态子集都应该是无关紧要的。

Shore and Johnson(1980) 证明相对熵或 Kullback-Leibler 散度同样满足相容性公理。

5. Pythagorean 分解

证明式(10.43)的分解，可以进行如下操作。由定义有

$$\begin{aligned} D_{p_{\mathbf{X}} \| p_{\mathbf{U}}} &= \int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x}) \log\left(\frac{p_{\mathbf{X}}(\mathbf{x})}{p_{\mathbf{U}}(\mathbf{x})}\right) \mathrm{d}\mathbf{x} = \int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x}) \log\left(\frac{p_{\mathbf{X}}(\mathbf{x})}{\tilde{p}_{\mathbf{X}}(\mathbf{x})}\right) \cdot \left(\frac{\tilde{p}_{\mathbf{X}}(\mathbf{x})}{p_{\mathbf{U}}(\mathbf{x})}\right) \mathrm{d}\mathbf{x} \\ &= \int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x}) \log\left(\frac{p_{\mathbf{X}}(\mathbf{x})}{\tilde{p}_{\mathbf{X}}(\mathbf{x})}\right) \mathrm{d}\mathbf{x} + \int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x}) \log\left(\frac{\tilde{p}_{\mathbf{X}}(\mathbf{x})}{p_{\mathbf{U}}(\mathbf{x})}\right) \mathrm{d}\mathbf{x} \\ &= D_{p_{\mathbf{X}} \| \tilde{p}_{\mathbf{X}}} + \int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x}) \log\left(\frac{\tilde{p}_{\mathbf{X}}(\mathbf{x})}{p_{\mathbf{U}}(\mathbf{x})}\right) \mathrm{d}\mathbf{x} \end{aligned} \tag{A}$$

从 $\tilde{p}_{\mathbf{X}}(\mathbf{x})$ 和 $p_{\mathbf{U}}(\mathbf{x})$ 的定义得到

$$\log\left(\frac{\tilde{p}_{\mathbf{X}}(\mathbf{x})}{p_{\mathbf{U}}(\mathbf{x})}\right) = \log\left(\frac{\prod_{i=1}^{m} \tilde{p}_{X_i}(x_i)}{\prod_{i=1}^{m} p_{U_i}(x_i)}\right) = \sum_{i=1}^{m} \log\left(\frac{\tilde{p}_{X_i}(x_i)}{p_{U_i}(x_i)}\right)$$

令 I 记式(A)最后一行中的积分，可以写成

$$I = \int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x}) \log\left(\frac{\tilde{p}_{\mathbf{X}}(\mathbf{x})}{p_{\mathbf{U}}(\mathbf{x})}\right) \mathrm{d}\mathbf{x} = \int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x}) \log\left(\frac{\prod_{i=1}^{m} \tilde{p}_{X_i}(x_i)}{\prod_{i=1}^{m} p_{U_i}(x_i)}\right) \mathrm{d}\mathbf{x}$$

$$= \sum_{i=1}^{m} \int_{-\infty}^{\infty} \left(\log\left(\frac{\tilde{p}_{X_i}(x_i)}{p_{U_i}(x_i)}\right) \int_{-\infty}^{\infty} p_{\mathbf{X}}(\mathbf{x}) \mathrm{d}\mathbf{x}^{(i)}\right) \mathrm{d}x_i = \sum_{i=1}^{m} \int_{-\infty}^{\infty} \log\left(\frac{\tilde{p}_{X_i}(x_i)}{p_{U_i}(x_i)}\right) \tilde{p}_{X_i}(x_i) \mathrm{d}x_i \tag{B}$$

上式最后一行使用了式(10.39)的定义。式 (B) 的积分是 Kullback-Leibler 散度，$D_{\tilde{p}_{X_i} \| p_{U_i}}$ $i = 1, 2, \cdots, m$。

为了把式 (B) 写成最终的形式，注意函数 $\tilde{f}_{X_j}(X_j)$ 下面的面积是 1，因此可写为

$$I = \sum_{i=1}^{m} \int_{-\infty}^{\infty} \prod_{j=1}^{m} \tilde{p}_{X_j}(x_j) \left(\log\left(\frac{\tilde{p}_{X_i}(x_i)}{p_{U_i}(x_i)}\right) dx_i\right) \mathrm{d}\mathbf{x}^{(i)} = \int_{-\infty}^{\infty} \tilde{p}_{\mathbf{X}}(\mathbf{x}) \log\left(\frac{\prod_{i=1}^{m} \tilde{p}_{X_i}(x_i)}{\prod_{i=1}^{m} p_{U_i}(x_i)}\right) \mathrm{d}\mathbf{x} = D_{\tilde{p}_{\mathbf{X}} \| p_{\mathbf{U}}} \tag{C}$$

其中在第一个等式中利用了定义 $d\mathbf{x}=dx_i d\mathbf{x}^{(i)}$，如同在 10.5 节描述的一样。因此，将(C)代入(A)，我们得到期望的分解：

$$D_{p_{\mathbf{X}}\|p_{\mathbf{U}}} = D_{p_{\mathbf{X}}\|p_{\mathbf{X}}} + D_{p_{\mathbf{X}}\|p_{\mathbf{U}}}$$

6. *系词*

单词系词是拉丁语的“连接”或“键”的意思；在语法和逻辑上经常用来表示连接主题和判定的命题的部分（Nelsen，2006）。在数学文献中，这一术语由 Sklar(1959) 在以他的名字命名的定理中首次运用：Sklar 定理通过“联合”一维分布函数描述了多变量分布函数的形成。Nelsen 的书提供了关于系词的有趣的历史观点且描述了其基本性质，提供了构造系词的方法以及在模型化和统计相关学习中系词的规则。关于系词的详细文献和相关问题在 Nelsen 的书的最后给出。

7. Nadal and Parga(1994，1997)还讨论了最大互信息和冗余减少之间的关系，得到同样的结果：神经系统的输入向量和输出向量之间的互信息的最大化也就导致数据减少。Haft and van Hemmen(1998)讨论视网膜的最大互信息滤波器的实现情况。结果表明，像视网膜这样的感觉系统所产生的内部环境表示，冗余性对获得噪声鲁棒性是最根本的。

8. *典型相关分析*

典型相关分析理论由 Hotelling(1935，1936)首先提出。为了讲述这一理论，我们遵循 Anderson(1984)的处理方式。

考虑由 m 个分量组成的 0 -均值随机向量 $\mathbf{X}$，其 $m\times m$ 的协方差矩阵为 $\mathbf{\Sigma}$。令 $\mathbf{X}$ 分解为两个子向量 $\mathbf{X}_a$ 和 $\mathbf{X}_b$，其分量个数分别为 $\mathbf{m}_a$ 和 $\mathbf{m}_b$。相应地，协方差矩阵 $\mathbf{\Sigma}$ 被分解为

$$\mathbf{\Sigma}=\mathbb{E}[\mathbf{X}\mathbf{X}^T]=\mathbb{E}\left[\begin{pmatrix}\mathbf{X}_a\\ \mathbf{X}_b\end{pmatrix}(\mathbf{X}_a,\mathbf{X}_b)^T\right]=\begin{bmatrix}\mathbb{E}[\mathbf{X}_a\mathbf{X}_a^T] & \mathbb{E}[\mathbf{X}_a\mathbf{X}_b^T]\\ \mathbb{E}[\mathbf{X}_b\mathbf{X}_a^T] & \mathbb{E}[\mathbf{X}_b\mathbf{X}_b^T]\end{bmatrix}=\begin{bmatrix}\mathbf{\Sigma}_{aa} & \mathbf{\Sigma}_{ab}\\ \mathbf{\Sigma}_{ba} & \mathbf{\Sigma}_{bb}\end{bmatrix}$$

其中

$$\mathbf{\Sigma}_{ba}=\mathbf{\Sigma}_{ab}^T$$

典型相关分析（CCA)的目标是构成子向量 $\mathbf{X}_a$ 和 $\mathbf{X}_b$ 的线性变换使之清楚地以最大的方式展现变换后的随机变量之间的交互相关。为此，考虑线性变换

$$Y_a=\mathbf{w}_a^T\mathbf{X}_a$$

和

$$Y_b=\mathbf{w}_b^T\mathbf{X}_b$$

其中 Y_a 和 Y_b 都是 0 -均值随机变量，且 $m_a\times 1$ 向量 $\mathbf{w}_a$ 和 $m_b\times 1$ 向量 $\mathbf{w}_b$ 是要决定的基向量。由于 Y_a 的倍数和 Y_b 的倍数的互相关函数与 Y_a、Y_b 自身的互相关函数是相同的，因此可以要求权向量 $\mathbf{W}_a$ 和 $\mathbf{W}_b$ 这样选择使得 $\mathbf{Y}_a$ 和 $\mathbf{Y}_b$ 具有单位方差。这一要求导致下面的两个条件：

$$1=\mathbb{E}[Y_a^2]=\mathbb{E}[\mathbf{w}_a^T X_a\mathbf{X}_a^T\mathbf{w}_a]=\mathbf{w}_a^T\mathbf{\Sigma}_{aa}\mathbf{w}_a \tag{A}$$

和

$$1=\mathbb{E}[Y_b^2]=\mathbb{E}[\mathbf{w}_b^T X_b\mathbf{X}_b^T\mathbf{w}_b]=\mathbf{w}_b^T\mathbf{\Sigma}_{bb}\mathbf{w}_b \tag{B}$$

有了上述的引导性素材，我们现在可以说明手头的问题：

寻找权向量 $\mathbf{W}_a$ 和 $\mathbf{W}_b$ 以最大化互相关函数

$$\mathbb{E}[Y_aY_b]=\mathbb{E}[\mathbf{w}_a^T\mathbf{X}_a\mathbf{X}_a^T\mathbf{w}_b]=\mathbf{w}_a^T\mathbf{\Sigma}_{ab}\mathbf{w}_b$$

服从式（A）和（B）所表示的两个条件。

为了解决约束优化问题，我们利用拉格朗日乘子法，因此写出如下拉格朗日算子：

$$J(\mathbf{w}_a,\mathbf{w}_b)=\mathbf{w}_a^T\mathbf{\Sigma}_{ab}\mathbf{w}_b-\frac{1}{2}\mu_a(\mathbf{w}_a^T\mathbf{\Sigma}_{aa}\mathbf{w}_a-1)-\frac{1}{2}\mu_b(\mathbf{w}_b^T\mathbf{\Sigma}_{bb}\mathbf{w}_b-1)$$

其中 μ_a 和 μ_b 是拉格朗日乘子，引入因子 1/2 是为了简化表达。对 $\mathbf{W}_a$ 和 $\mathbf{W}_b$ 微分拉格朗日算子 $J(\mathbf{W}_a,\mathbf{W}_b)$ 并将其结果设为 0，得到如下一对方程：

$$\mathbf{\Sigma}_{ab}\mathbf{w}_b-\mu_a\mathbf{\Sigma}_{aa}\mathbf{w}_a=\mathbf{0} \tag{C}$$

和

$$\mathbf{\Sigma}_{ba}\mathbf{w}_a-\mu_b\mathbf{\Sigma}_{bb}\mathbf{w}_b=\mathbf{0} \tag{D}$$

式（C）和（D）的左边分别乘以 $\mathbf{W}_a^T$ 和 $\mathbf{W}_b^T$，我们有

$$\mathbf{w}_a^T\mathbf{\Sigma}_{ab}\mathbf{w}_b-\mu_a\mathbf{w}_a^T\mathbf{\Sigma}_{aa}\mathbf{w}_a=\mathbf{0} \tag{E}$$

和

$$\mathbf{w}_b^T \boldsymbol{\Sigma}_{ba} \mathbf{w}_a - \mu_b \mathbf{w}_b^T \boldsymbol{\Sigma}_{bb} \mathbf{w}_b = \mathbf{0} \tag{F}$$

然后，在式（E）和（F）中分别调用式（A）和（B）的条件，证明

$$\mu_a = \mu_b = \mathbf{w}_a^T \boldsymbol{\Sigma}_{ab} \mathbf{w}_b \tag{G}$$

其中我们已经用了关系 $\boldsymbol{\Sigma}_{ba} = \boldsymbol{\Sigma}_{ab}^T$。因此，假设在拉格朗日算子 $J(\mathbf{W}_a, \mathbf{W}_b)$ 中两个拉格朗日乘子具有共同值，以后记为 μ。

而且，认识到 Y_a 和 Y_b 的方差都被归一化为单位 1，由式（G）知拉格朗日乘子 μ 是这两个随机变量之间的典型相关。

现在的关键问题是：如何决定基向量 $\mathbf{w}_a$ 和 $\mathbf{w}_b$？利用式(C)和式(D)，可以证明基向量 $\mathbf{W}_a$ 和 $\mathbf{W}_b$ 分别由一对特征方程定义。

$$\underbrace{\boldsymbol{\Sigma}_{aa}^{-1} \boldsymbol{\Sigma}_{ab} \boldsymbol{\Sigma}_{bb}^{-1} \boldsymbol{\Sigma}_{ba}}_{\mathbf{C}_a} \mathbf{w}_a = \lambda \mathbf{w}_a \tag{H}$$

和

$$\underbrace{\boldsymbol{\Sigma}_{bb}^{-1} \boldsymbol{\Sigma}_{ba} \boldsymbol{\Sigma}_{aa}^{-1} \boldsymbol{\Sigma}_{ab}}_{\mathbf{C}_b} \mathbf{w}_b = \lambda \mathbf{w}_b \tag{I}$$

其中

$$\lambda = \mu^2 \tag{J}$$

因此可以陈述如下：

1. 矩阵 $\mathbf{C}_a$ 的特征值 λ 等于典型相关的平方值，相应的特征向量定义了基向量 $\mathbf{w}_a$。
2. 第二个矩阵 $\mathbf{C}_b$ 的特征值 λ 也等于典型相关的平方植，相应的特征向量定义了第二个基向量 $\mathbf{w}_b$。

然而，注意特征方程（G）、（H）和（I）的有意义解的数目受到维数 m_a 或 m_b 的限制，无论哪一个都是较小的。最大特征值 λ_1 产生最强的典型相关；下一个特征值 λ_2 产生第二强的典型相关，以此类推。

这里所描述的典型相关分析（CCA）可用于揭示两个相关但不同的数据集之间的二阶统计相关。甚至，尽管 CCA 不包括高阶统计，但实际中它经常工作良好。

从式（H）和（I），很显然典型相关分析包含了主分量作为特例，当矩阵 $\mathbf{C}_a$ 和 $\mathbf{C}_b$ 被分配给一个共同值时发生，即当子向量 $\mathbf{X}_a$ 和 $\mathbf{X}_b$ 是一个相同的向量时。

另一个有趣的是，在 Fyfe(2005)中，介绍了关于典型相关分析的两个不同神经执行，得到了人工和实际数据模拟的支持。

9. Uttley 的 Informon

在 Uttley(1970) 中考虑负信息通路，通过最优化通路中输入信号与输出信号之间的互信息的负值。结果表明，这样的系统在调整期间适宜变成输入信号集中更常发生的模式的判别器。这种模型称作 informon，它与 Imin 原则有松散关系。

10. 模糊 Imin 处理器

在 Ukrainec and Haykin(1996) 中描述的系统包括一个后探测处理器，它利用反射器沿水道的水陆边界位置的先验知识。模糊处理器结合初始探测性能和基于视觉的边缘检测器的输出以便有效地去除错误警报，从而使系统性能进一步提高。

11. 历史注记

关于盲源分离和独立分量分析的两篇文章在文献中广为人知：

- 关于盲源分离问题（BSS）的 Herault 等(1985) 的文章利用了 Hebb 学习。
- Comon(1994) 关于独立分量分析（ICA）的文章首次提出了这一术语。

关于 BSS 和 ICA 的详细历史记录，包括一些其他的早期贡献，参看 Jutten and Taleb(2000)。

12. 自然梯度

使用$\nabla^* D = (\nabla D)\mathbf{W}^T\mathbf{W}$来代替通常梯度$\nabla D$解决盲源分离问题的思想在 Cardoso and Laheld(1996) 中有详细的介绍。这里$\nabla^* D$称为相对梯度，这个梯度与自然梯度是相同的。自然梯度是从信息几何的观点来定义的（Amari，1998；Amari 等，1996）。

13. 黎曼空间

例如，在 n 维黎曼空间中，向量 $\mathbf{a}$ 的平方范数定义为

$$\| \mathbf{a} \|^2 = \sum_{i=1}^{m} \sum_{j=1}^{m} a_i g_{ij} a_j$$

其中 g_{ij} 是黎曼空间坐标 $x_1, x_2, \cdots, x_n$ 的函数，$g_{ij} = g_{ji}$，表达式右边总是正的。该表达式是欧几里得平方

范数公式

$$\| \mathbf{a} \|^2 = \sum_{i=1}^{m} a_i^2$$

的推广。关于黎曼空间结构的讨论，参考 Amari(1987)，Murray and Rice(1993) 和 Rosenberg(1997)。

14. 超高斯分布和亚高斯分布

考虑随机变量 X，其概率密度函数由 $p_X(x)$定义，其中 x 是 X 的样本值。令 $p_X(x)$由可用形式 $\exp(-g(x))$ 来表示，这里 $g(x)$是 x 的偶函数，对于 x 可能除原点外是可微的，$g(x)$对 x 的导数记为 $g'(x)$。

如果当 $0<x<\infty$，$g'(x)/x$ 是严格递减的，则随机变量 X 称为是超高斯的。例如可能取 $g(x)=|x|^{\beta}$，$\beta<2$。

另一方面，如果随机变量是一致分布的，或者 $g(x)$ 和 $g'(x)/x$ 对于 $0<x<\infty$是严格递增的，则随机变量 X 被称为亚高斯的，例如，可以取 $g(x)=|x|^{\beta}$，$\beta>2$。

有时（也许有些滥用的方式）使用随机变量的峭度（kurtosis）符号作为亚高斯或超高斯的指标。随机变量 X 的峭度定义为：

$$K_4 = \frac{\mathbb{E}[X^4]}{(\mathbb{E}[X^2])^2} - 3$$

在此基础上，根据峭度 K_4 为负或为正，随机变量 X 分别称为亚高斯或超高斯的。

15. 另一个历史注记

从历史上看，Cardoso(1997) 第一个从理论上证明：在自然梯度算法中利用正确类型的非线性激活函数解盲源分离对其达到收敛是充分的。

16. 最大似然估计

最大似然估计具有一些期望的性质。在相当普遍的条件下，可以证明下列的渐进性质（Kmenta，1971）：

(i) 最大似然估计是一致的。令 $L(\boldsymbol{\theta})$记 log-似然函数，$\boldsymbol{\theta}_i$ 记参数向量 $\boldsymbol{\theta}$ 的一个元素。偏导数 $\partial L/\partial \boldsymbol{\theta}_i$ 称为得分（score）。我们说最大似然估计是一致的，是在这样的意义下：$\boldsymbol{\theta}_i$ 的值，对之的得分 $\partial L/\partial \boldsymbol{\theta}_i$ 是恒为 0 的，随着估计中样本大小趋于无穷从概率上收敛于 $\boldsymbol{\theta}_i$ 的真值。

(ii) 最大似然估计是渐进有效的。即

$$\lim_{N\to\infty}\left\{\frac{\mathrm{var}[\theta_i - \hat{\theta}_i]}{I_{ii}}\right\} = 1, \quad \text{对于所有 } i$$

其中 N 是样本大小，$\hat{\boldsymbol{\theta}}_i$ 是 $\boldsymbol{\theta}_i$ 的最大似然估计，I_{ii} 是逆 Fisher 信息矩阵的第 i 个对角元素。Fisher 信息矩阵定义为

$$\mathbf{J} = -\begin{bmatrix} \mathbb{E}\left[\frac{\partial^2 L}{\partial \theta_1^2}\right] & \mathbb{E}\left[\frac{\partial^2 L}{\partial \theta_1 \partial \theta_2}\right] & \cdots & \mathbb{E}\left[\frac{\partial^2 L}{\partial \theta_1 \partial \theta_m}\right] \\ \mathbb{E}\left[\frac{\partial^2 L}{\partial \theta_2 \partial \theta_1}\right] & \mathbb{E}\left[\frac{\partial^2 L}{\partial \theta_2^2}\right] & \cdots & \mathbb{E}\left[\frac{\partial^2 L}{\partial \theta_2 \partial \theta_m}\right] \\ \mathbb{E}\left[\frac{\partial^2 L}{\partial \theta_m \partial \theta_1}\right] & \mathbb{E}\left[\frac{\partial^2 L}{\partial \theta_m \partial \theta_2}\right] & \cdots & \mathbb{E}\left[\frac{\partial^2 L}{\partial \theta_m^2}\right] \end{bmatrix}$$

其中 m 是参数向量 $\boldsymbol{\theta}$ 的维数。

(iii) 最大似然估计是渐进高斯的。即随着样本大小趋于无穷，最大似然估计$\hat{\boldsymbol{\theta}}$的每个元素假设为高斯分布。

实际上，我们发现最大似然估计的大样本（即渐进的）性质在样本大小 $N\geqslant 50$ 时保持得很好。

17. ICA 的 Infomax 的原始版本

式(10.127)讲述了 ICA 算法的 Infomax 的原始版本是由 Bell and Sejnowski (1995) 导出的。这一原始算法收敛非常慢，这是由于记录了转置分解矩阵 $\mathbf{W}$ 的逆的 $\mathbf{W}^{-T}$项的存在。后来发现，通过利用自然梯度来代替通常的（欧几里得）梯度，正如式(10.128)所述，算法的收敛明显加速。

18. Gram-Schmidt 正交化过程在 Golub and Van Loan(1996) 中讲述。

19. 对称 FastICA

作为 10.17 节中讲述的快速 ICA 算法的单一单元压缩版本的补充，存在这一算法的另一个版本，称为对称 FastICA 算法。后一个版本以并行的方式估计盲源分离问题的分量。具体地，对每一分量，这一算法包含了单一单元的更新的并行计算，接着在每次迭代后对估计的分离矩阵进行对称正交。在 Tichavsky et al. (2006) 中，在“局部”意义下推导了算法的两个版本的分析闭式表示刻画的分离性。

20. TIMIT 数据库

TIMIT(Texas Instruments (TI) and Massachusetts Institute of Technology (MIT)) 数据库是语音识别的一个标准数据库。它是在安静环境下录制的 8-kHz 带宽朗读（不是对话）语音组成。这个数据库包括了 630 个发言者（438 位男性和 192 位女性），每位发言者有 10 个发言，平均每个发言是 3 秒钟。

21. 信息瓶颈的另一个观点

关于信息瓶颈的另一个考虑方法是将之看成“最小充分统计量”的经典概念的泛化。在样本概率密度函数 $p_{\mathbf{x}|A}(\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_n|\mathbf{a})$ 下的参数向量 $\mathbf{a}$ 的充分统计量是一个样本的向量函数 $\mathbf{S}(\mathbf{X})$，它保留了关于参数 $\mathbf{a}$ 的样本的所有互信息；即 $I(\mathbf{X};\mathbf{a})=I(\mathbf{S}(\mathbf{X});\mathbf{a})$。最小充分估计量是最简单的可能充分统计量，或者是另一个充分统计量的函数，$\mathbf{T}(\mathbf{X})=f(\mathbf{S}(\mathbf{X}))$。从称为数据处理不等（Cover and Thomas 2006）的互信息的基本性质，对于任意充分统计量 $\mathbf{S}(\mathbf{X})$ 如果 $I(\mathbf{T};\mathbf{X})\leqslant I(\mathbf{S};\mathbf{X})$ 时 $\mathbf{T}(\mathbf{X})$ 是最小的。最小充分统计量捕捉了“样本 $\mathbf{X}$ 关于参数向量 $\mathbf{a}$ 的相关部分”的概念。不幸的是，精确的（固定维数）充分统计量仅仅对指数形式的分布存在。这一重要概念的一个有吸引力的泛化是通过信息瓶颈方法来达到的，它显式寻找 $\mathbf{X}$ 的具有关于 $\mathbf{X}$ 最小互信息和关于相关变量 $\mathbf{Y}$（或者在参数统计意义下的 $\mathbf{a}$）具有最大信息的函数。

22. 在经典 ICA 理论之外

本章的前面重点讨论了经典 ICA 理论。在独立分量分析和盲源分离的研究中已经在多个前沿有了显著的扩展，包括如下这些内容：

- 分离卷积混合，这里的注意力在于实际观测的信号混合中卷积扮演着重要角色的事实。
- 非线性盲源分离，这里非线性是混合过程的固有特性。
- 非独立源的盲源分离，这里我们认识到一个或多个源信号可能不是统计独立的。
- 有噪独立分量分析，这里放松了对经典 ICA 理论的无噪的要求，因此迫使我们面对有噪源信号的实际现实。
- 欠定方案，这里盲源信号大于混合过程输出端的观测数，这可能在现实中发生。
- 多个独立子空间，这里 ICA 理论被扩展来完成这样的情形：源产生的信号占据了不同的子空间，这些子空间是彼此独立的，在每个子空间中有关的源信号依然是相关的。
- 不稳定下的盲源分离技术，这里盲源信号假设为不稳定的，挑战在于建立不稳定的概念。
- 盲源分离技术，其数学基础依赖于源信号的时频表达。
- 稀疏分量分析，这里源信号（如自然图像）的稀疏性的概念在其分离中扮演着关键角色。
- 基于时间相关的盲源分离技术，这里甚至可以分离在特定条件下的独立高斯源。

我们这里所列出的是一系列课题，它们不仅和源信号的实际实现有关，也高度概括了在 ICA 和 BSS 理论及其应用中的理论挑战。对于这些课题的详细讨论，有兴趣的读者可以参考 Hyvärinen 等（2001）、Roberts and Everson（2001）、Cichocki and Amari（2002）的书，以及 Cardoso（2001）和 Choi 等（2005）的综述论文。

习题

最大熵原则

10.1 随机变量 X 的支撑集（也就是取非零的值域）定义为 $[a, b]$，没有别的限制加在 X 上。该随机变量的最大熵分布是什么？证明你的结论。

互信息

10.2 (a) 利用微分熵 $h(X)$ 和条件微分熵 $h(X|Y)$ 的定义从式(10.28)的第一行开始到该式的第二行的积分公式，定义一对连续随机变量 X 和 Y 之间的互信息 $I(X;Y)$。

(b) 利用对互信息 $I(X;Y)$ 推导的积分公式来证明式(10.30)到式(10.32)描述的性质。

(c) 证明式(10.35)的第二行，将相对熵 $D_{p\|q}$ 表示为期望形式。

10.3 假设输入随机向量 $\mathbf{X}$ 由初始分量 $\mathbf{X}_1$ 和背景分量 $\mathbf{X}_2$ 组成，定义

$$Y_i = \mathbf{a}_i^T\mathbf{X}_1$$
$$Z_i = \mathbf{b}_i^T\mathbf{X}_2$$

试问 Y_i 和 Z_i 之间的互信息，以及 X_1 和 X_2 之间的互信息有何关系？假设向量 X 的概率模型是多元高斯分布：

$$p_{\mathbf{X}}(\mathbf{x}) = \frac{1}{(2\pi)^{m/2}(\det\boldsymbol{\Sigma})^{1/2}}\exp((\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}))$$

其中 $\boldsymbol{\mu}$ 是 $\mathbf{X}$ 的均值，$\boldsymbol{\Sigma}$ 是它的协方差矩阵。

10.4 在这个习题中我们探索用相对熵或 Kullback-Leibler 散度来推导多层感知机（Hopfield，1987；Baum and Wilczek，1998）的监督学习算法。更确切地说，考虑一个由一个输入层、一个隐藏层和一个输出层构成的多层感知机，假设实例或样本 α 呈现给输入，输出层神经元 k 的输出解释为概率：

$$y_{k|\alpha} = p_{k|\alpha}$$

相应地，令 $q_{k|\alpha}$ 表示当输入是 α 时，假设 k 为真的条件概率的实际值，该多层感知机的相对熵定义为

$$D_{p\|q} = \sum_{\alpha} p_{\alpha} \sum_{k} \left(q_{k|\alpha} \log\left(\frac{q_{k|\alpha}}{p_{k|\alpha}}\right) + (1-q_{k|\alpha}) \log\left(\frac{1-q_{k|\alpha}}{1-p_{k|\alpha}}\right)\right)$$

其中 p_{α} 是出现 α 情况的一个先验概率。

以 $D_{p\|q}$ 为最优化的代价函数，推导一个多层感知机的学习算法。

系词

10.5 说明在 10.6 节中性质 1 下列出的系词 $C_{UV}(u, v)$ 的三个有限值。

10.6 系词的一个有趣的应用是生成新的分布（Genest and Mackay，1989）。本习题的（a）和（b）讲述这一应用。

（a）积系词

一对统计独立的随机变量 X 和 Y 的每个成员都是均匀分布的，正如下式所示：

$$p_X(x) = \begin{cases} \frac{1}{2}, & -1 \leqslant x \leqslant +1 \\ 0, & \text{否则} \end{cases}$$

$$p_Y(y) = \begin{cases} \frac{1}{2}, & -1 \leqslant y \leqslant 1 \\ 0, & \text{否则} \end{cases}$$

画出系词 $C_{U,V}(u, v)$。

（b）高斯系词

考虑具有 0-均值和单位方差的一对相关高斯分布，为下面的两个相关系数值画出相应的系词：

（i）$\rho = 0.9$

（ii）$\rho = -0.9$

10.7 考虑一对随机变量 X 和 Y，其互信息记为 $I(X; Y)$。比对式(10.28)和基于系词的作为统计相关测量的式(10.49)的 $I(X; Y)$的公式。

10.8 为了推导式(10.50)的互信息和系词熵之间的关系，我们采用了直接方式。根据和推导式(10.49)相似的方法重新推导式(10.50)。

Infomax 原则

10.9 假设有两个通道。它们的输出分别用随机变量 X 和 Y 表示，要求使 X，Y 之间的互信息达到最大。证明只要满足以下条件则就可以达到要求：

（a）出现 X 的概率和出现 Y 的概率分别是 0.5。

（b）X，Y 的联合概率密度函数集中在概率空间的一个小区域内。

10.10 考虑图 P10.10 中的噪声模型，两个神经网络的输入端都为 m 个源节点。输入由 $X_1, X_2, \cdots, X_m$ 表示，相应的输出结果用 Y_1，Y_2 表示。可以假设：

- 网络输出端的加性噪声分量 N_1，N_2 是高斯分布，具有零均值和共同方差 σ_N^2，并且互不相关。
- 每个噪声源与输入信号无关。

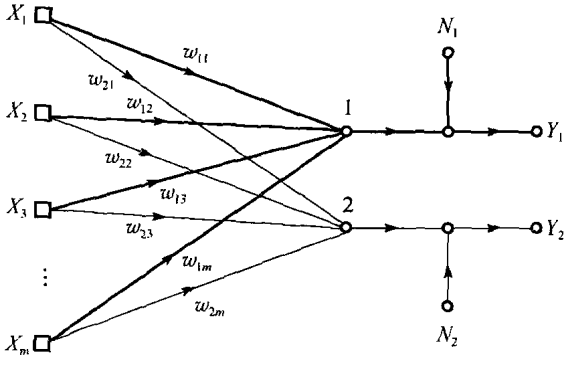


图　P10.10

- 输出信号 Y_1，Y_2 都是 0 -均值的高斯分布。

  (a) 求输出向量 $\mathbf{Y}=[Y_1,Y_2]^T$ 与输入向量 $\mathbf{X}=[X_1,X_2,\cdots,X_m]^T$ 之间的互信息。

  (b) 利用 (a) 中导出的结果，检测在以下情况下冗余/相异性是如何折中的 (Linsher，1998a)：

  (i) 噪声的方差很大，表示为 σ_N^2 相对于 Y_1，Y_2 很大。

  (ii) 噪声的方差很小，表示为 σ_N^2 相对于 Y_1，Y_2 很小。

10.11 在 10.10 节中所描述的 Imax 原则中，目标是根据噪声神经系统的输入向量 X_a 和 X_b 求输出 Y_a，Y_b 之间的互信息 $I(Y_a;Y_b)$ 的最大值。在另一种方法中，一个不同的目标是求输出 Y_a 和 Y_b 的平均值与它们固有的共同信号分量 $\mathbf{S}$ 之间的互信息 $I\left(\frac{Y_a+Y_b}{2};S\right)$ 的最大值。

利用例 8 中描述的噪声模型，完成下列任务：

(a) 证明

$$I\left(\frac{Y_a+Y_b}{2};S\right)=\log\left(\frac{\mathrm{var}[Y_a+Y_b]}{\mathrm{var}[N_a+N_b]}\right)$$

其中 N_a，N_b 是 Y_a，Y_b 相应的噪声分量。

(b) 用信号加噪声与噪声的比来解释此互信息。

独立分量分析

10.12 给出主分量分析（在第 8 章讨论过）与独立分量分析（在第 10.12 节讨论过）的详细比较。

10.13 独立分量分析可以用作检测和分类之前近似数据分析的预处理步骤 (Comon，1994)。讨论能在这种应用中加以利用的独立分量分析的性质。

10.14 Darmois 定理陈述只有当各个独立变量是高斯分布的，其和才是高斯分布的 (Darmois，1953)。用独立分量分析证明这个定理。

10.15 在实际的应用中，一个独立分量分析算法实现只能得到“尽可能统计独立”。比较用该算法解盲源分离问题得到的解与利用去相关方法得到的解的差异。假设观察向量的协方差矩阵为非奇异的。

ICA 的自然梯度学习算法

10.16 参考图 10.12 描述的系统，证明分离器的输出 $\mathbf{Y}$ 的任何两个分量的互信息最小化与参数化的概率密度函数 $p_Y(\mathbf{y},\mathbf{W})$ 和相应的析因分布 $\tilde{p}_Y(\mathbf{y},\mathbf{W})$ 之间的 Kullback-Leibler 散度（相对熵）的最小化等价。

10.17 在式(10.100)中描述的盲源分离问题的自适应算法有两个重要的性质：(1) 等变化性；(2) 权值矩阵 $\mathbf{W}$ 保持非奇异。性质 (1) 在 10.14 节后面部分有详细的介绍。在本习题中考查第二个性质。

假设用于开始式(10.100)算法的初始值 $\mathbf{W}(0)$ 满足条件

$$|\det(\mathbf{W}(n))|\neq 0 \quad \text{对于所有 } n$$

证明这是保证 $\mathbf{W}(n)$ 对所有的 n 是非奇异的充分必要条件。

10.18 本习题讨论式(10.100)所描述的盲源分离算法的批量公式。具体写成：

$$\Delta\mathbf{W}=\eta\left(\mathbf{I}-\frac{1}{N}\boldsymbol{\Phi}(\mathbf{Y})\mathbf{Y}^T\right)\mathbf{W}$$

其中

$$\mathbf{Y}=\begin{bmatrix} y_1(1) & y_1(2) & \cdots & y_1(N)\\ y_2(1) & y_2(2) & \cdots & y_2(N)\\ \vdots & \vdots & \vdots & \vdots\\ y_m(1) & y_m(2) & \cdots & y_m(N)\end{bmatrix}$$

且

$$\boldsymbol{\Phi}(\mathbf{Y})=\begin{bmatrix} \varphi(y_1(1)) & \varphi(y_1(2)) & \cdots & \varphi(y_1(N))\\ \varphi(y_2(1)) & \varphi(y_2(2)) & \cdots & \varphi(y_2(N))\\ \vdots & \vdots & \vdots & \vdots\\ \varphi(y_m(1)) & \varphi(y_m(2)) & \cdots & \varphi(y_m(N))\end{bmatrix}$$

其中 N 是可用数据点的数目。证明上式描述的权值矩阵 $\mathbf{W}$ 的调整 $\Delta\mathbf{W}$ 的公式成立。

ICA 算法的 Infomax

10.19 考虑图 10.16，得到（利用随机向量符号）：

$$\mathbf{Y} = \mathbf{WX}$$

其中

$$\mathbf{Y} = [Y_1, Y_2, \cdots, Y_m]^T$$
$$\mathbf{X} = [X_1, X_2, \cdots, X_m]^T$$

$\mathbf{W}$ 是一个 $m \times m$ 的权值矩阵。令

$$\mathbf{Z} = [Z_1, Z_2, \cdots, Z_m]^T$$

其中

$$Z_k = \varphi(Y_k), k = 1, 2, \cdots, m$$

(a) 证明 $\mathbf{Z}$ 的联合熵与 Kullback-Leibler 散度 $D_{p \| \breve{p}}$ 之间的关系为：

$$h(\mathbf{Z}) = -D_{p \| \breve{p}} - D_{\breve{p} \| q}$$

其中 $D_{\breve{p} \| q}$ 是下面两个量的 Kullback-Leibler 散度：(a) 统计独立的（即析因式的）输出向量组 $\mathbf{Y}$ 的概率密度函数；(b) 由 $\prod_{i=1}^{m} q(y_i)$ 定义的概率密度函数。

(b) 对于所有的 i，当 $q(y_i)$ 与初始源输出 S_i 的概率密度函数相等时，$h(\mathbf{Z})$的公式该如何修改？

10.20 (a) 从式(10.124)开始，推导式(10.125)的结果。

(b) 用式(10.126)中的 logistic 函数，证明使用式(10.125)将产生由式(10.127)给出的公式。

(c) 为建立在式(10.129)的学习算法上的盲源分离 Infomax 算法构造信号流图。

FastICA 算法

10.21 给定由式(10.132)和式(10.133)定义的函数 $\Phi(v)$，即

1. $\Phi(v) = \log(\cosh(v))$

2. $\Phi(v) = \exp\left(-\frac{v^2}{2}\right)$

为下列公式推导相应的表达式：

$$\varphi(v) = \frac{\mathrm{d}\Phi(v)}{dv}$$

和

$$\varphi'(v) = \frac{\mathrm{d}\varphi(v)}{\partial v}$$

在点 1 和点 2 的例子中 $\Phi(v)$，$\varphi(v)$和 $\varphi'(v)$中哪一个函数适合于神经激活函数？证明你的回答。

10.22 FastICA 算法声称比其他 ICA 算法（如自然梯度算法和 Infomax 的 ICA 算法）快得多。验证 FastICA 算法中对于这一重要性质的特征。

相关 ICA

10.23 在组合 Infomax 和 Imax 到目标函数 $J(W_a, W_b)$ 时，我们省略了在 Infomax 和 Imax 之间提供折中的正则性。这样做是为了简化 ICA 算法的公式。如何才能够修改目标函数使其保留网络 a 和 b 的输出之间的统计相关的同时仍然在目标函数中包括正则性？这一延伸有什么意义？

10.24 从计算项上，算法相关 ICA 和 FastICA 共享了两个相似的特征。这些特征是什么？给出详细说明。

10.25 对比相关 ICA 和其他 ICA 有什么不同特征。

信息瓶颈方法

10.26 考虑通过画出如图 10.21 所示的 $I(\mathbf{T}; \mathbf{Y})$对 $I(\mathbf{X}; \mathbf{T})$的信息曲线。证明：对于最优信息瓶颈解，这一曲线是递增凸曲线，在每一点的斜率是 $1/\beta$。

10.27 图 10.22 的关于信息瓶颈方法的直观描画和图 4.19a 的重复符网络（恒等映射）彼此间具有强烈的相似性。详细说明这一陈述及其相关的含义。

10.28 式(10.184) 是由式(10.182) 而来。

(a) 证明式(10.184)。

(b) 证明伴随公式(10.185)。

10.29 在应用式(10.183)的最优条件到式(10.186)的拉格朗日算子的过程中，我们跳过了一些严格步骤。

(a) 从式(10.183)开始，推出达到如下结果的所有步骤

$$\frac{1}{\lambda}\|\mathbf{x}-\boldsymbol{\gamma}(\mathbf{t})\|^2+\log\left(\frac{q_{\mathrm{T}|\mathbf{X}}(\mathbf{t}|\mathbf{x})}{q_{\mathrm{T}}(\mathbf{t})}\right)+\frac{\beta(\mathbf{x})}{\lambda p_{\mathbf{X}}(\mathbf{x})}=0$$

(b) 由此，推导式(10.188)和式(10.189)中的相容公式对。

计算机实验

10.30 考虑在图 10.9 中描述的系统包含如下三个独立源：

$s_1(n)=0.1\sin(400n)\cos(30n)$

$s_2(n)=0.01\mathrm{sgn}(\sin(500n+9\cos(40n)))$

$s_3(n)=$噪声，在范围 [−1，1] 上均匀分布

混合矩阵 **A** 是：

$$\mathbf{A}=\begin{bmatrix}0.56 & 0.79 & -0.37\\ -0.75 & 0.65 & 0.86\\ 0.17 & 0.32 & -0.48\end{bmatrix}$$

(a) 画出三个源信号 $s_1(n)$，$s_2(n)$ 和 $s_3(n)$ 的波形。

(b) 利用 10.14、10.16、10.17 节中讲述的三个 ICA 算法来解盲源分离问题，包含源 $s_1(n)$，$s_2(n)$，$s_3(n)$ 和混合矩阵 **A**。画出分离器输出产生的波形，并和 (a) 部分画出的相比较。

(c) 决定分离矩阵 **W**。

10.31 在 10.21 节中讲述的计算机实验中，我们利用了最优流形（对数据的非监督表达）和最小均方算法（LMS）来完成模式分类。用于分类的数据基于特定的图 1.8 所示的双月结构。

(a) 重复 10.21 节所示的计算机实验，这一次利用递归最小二乘（RLS）算法来代替 LMS 算法。

(b) 从性能收敛和计算复杂度的角度比较你的实验结果和 10.21 节的结果。

第11章
Neural Networks and Learning Machines, 3E

植根于统计力学的随机方法

本章组织

本章的研究主题是研究通过建立在根植于统计力学上的思想的随机算法，用于模拟、优化和学习的随机方法。

本章组织如下：

11.1 节是引言，主要列举对研究该主题的动机的描述。

11.2 节对统计力学进行了介绍性描述，重点是以动力学观点来看待自由能量和熵的概念。

11.3 节主要是讨论一种特殊随机过程名为马尔可夫链（Markov chains），其应用经常能出现在统计力学的研究中。

11.4 节至 11.6 节主要研究下列三个随机模拟/优化的方法：

- Metropolis 算法
- 模拟退火
- Gibbs 采样

Metropolis 算法和 Gibbs 采样分别对于静态过程和非静态过程进行了模拟，而模拟退火方法是面向优化的。

11.7 节至 11.9 节介绍根植于统计力学的随机机器：

- Boltzmann 机器
- losgistic 信度网络
- 深度信度网络

其中深度信度网络具有独特的性质，它克服了古典 Boltzmann 机器和 logistic 信度网络实用的限制。

11.10 节主要描述确定退火方法，它是对模拟退火方法的近似；不论它的名字，确定退火是一种随机算法。11.11 节介绍最大期望算法，同时一并讨论一种确定退火方法。

11.12 节对本章进行小结和讨论。

11.1 引言

作为无监督（自组织）学习系统的最后一种类别，我们以统计力学作为我们思想的出发点。统计力学的主题围绕对大系统宏观平衡态性质的形式化研究，而系统的每个基本元素遵循力学的微观定律。统计力学的主要目标是从微观元素（如原子和电子的运动）推导出宏观物体的热力学性质（Landau and Lifshitz，1980；Parisi，1988）。这里面对的自由度数量是巨大的，这样不得不用概率的方法进行研究。正如香农的信息论一样，在统计力学的研究中熵的概念起着关键的作用：

系统越有序或者它的概率分布越集中，则熵越小。

同理，我们可以说系统越无序或它的概率分布越均匀，则熵越大。在 1975 年，Jaynes 证明了熵不仅可以像前一章所述的那样作为构造统计推理的出发点，而且可以作为产生统计力学研究基础的 Gibbs 分布的出发点。

利用统计力学作为研究神经网络基础的兴趣可以追溯到 Cragg and Temperley(1954) 以及 Cowan(1968) 的早期工作。Boltzmann 机（Hinton & Sejnowski，1983，1986；Ackley 等

1985）也许是第一个由统计力学导出的多层学习机。机器的命名认可了神经网络自身的动力学行为和 Boltzmann 原始关于统计热力学工作的形式上的等价性。基本上说，Boltzmann 机可以对给定数据集的固有概率分布进行建模，这样在诸如模式完备和模式分类等任务中所使用的条件分布就可以导出来了。令人遗憾的是 Boltzmann 机的学习过程是令人难以忍受地慢，这一缺点导致对 Boltzmann 机的修改和产生了新的随机机器。以上这些问题构成了本章的大部分题材。

11.2　统计力学

考虑具有许多自由度的物理系统，它可以驻留在大量可能状态中的任何一个。例如，用 p_i 表示一个随机系统中状态 i 发生的概率，具有如下性质：

$$p_i \geqslant 0, \quad \text{对于所有 } i \tag{11.1}$$

且

$$\sum_i p_i = 1 \tag{11.2}$$

用 E_i 表示系统在状态 i 时的能量，统计热力学基本结论告诉我们，当系统和它周围的环境处于热平衡时，一个基本的结果是状态 i 发生的概率如下：

$$p_i = \frac{1}{Z}\exp\left(-\frac{E_i}{k_B T}\right) \tag{11.3}$$

其中 T 为开尔文绝对温度，k_B 为 Boltzmann 常数，Z 为与状态无关的常数。1 开尔文度相当于 −273 摄氏度，$k_B = 1.38 \times 10^{-23}$ 焦耳/开。

式(11.2)定义概率规范化的条件。将这个条件添加到式(11.3)得到

$$Z = \sum_i \exp\left(-\frac{E_i}{k_B T}\right) \tag{11.4}$$

规范化量 Z 称为状态和或者剖分函数（通常用符号 Z 是因为这项的德文名字为 Zustadsumme）。式(11.3)的概率分布称为典型分布或 Gibbs 分布[1]；指数因子 $(-E_i/k_B T)$ 称为 Boltzmann 因子。

对 Gibbs 分布以下两点值得注意：

1. 能量低的状态比能量高的状态发生的概率高。
2. 随着温度 T 降低，概率集中在低能状态的一个更小的子集上。

温度 T 可以被视为一种伪温度，它控制表示神经元“突触噪声”的热波动。它的精确标度因而无关紧要。相应地，我们可以置常数 k_B 为单位 1 而重新度量之，因此重新定义概率 p_i 和剖分函数（partition 函数）Z 如下：

$$p_i = \frac{1}{Z}\exp\left(-\frac{E_i}{T}\right) \tag{11.5}$$

和

$$Z = \sum_i \exp\left(-\frac{E_i}{T}\right) \tag{11.6}$$

今后我们处理统计力学就在这两个定义基础上进行，其中 T 简单称为系统温度。从式(11.5)我们注意到 $-\log p_i$ 可以被看作在单位温度下“能量”的一种度量。

自由能量和熵

物理系统的 Helmholtz 自由能量记为 F，由剖分函数 Z 定义如下：

$$F = -T\log Z \tag{11.7}$$

系统的平均能量定义为：

$$\langle E\rangle = \sum_i p_i E_i \tag{11.8}$$

$\langle\cdot\rangle$表示总体平均运算。因此，利用式(11.5) 至式(11.8)，可以看出平均能量和自由能量之差为

$$\langle E\rangle - F = -T\sum_i p_i \log p_i \tag{11.9}$$

式(11.9)右边的量忽略温度 T，称为系统的熵，表示为：

$$H = -\sum_i p_i \log p_i \tag{11.10}$$

(这个定义与第 10 章的信息论模型是一致的)

因此可以重写式(11.9) 为：

$$\langle E\rangle - F = TH$$

的形式或等价于：

$$F = \langle E\rangle - TH \tag{11.11}$$

考虑两个系统 A 和 A'彼此热接触。假设系统 A 比系统 A'更小，这样 A'可以看作具有恒温 T 的热储藏器。两个系统的总熵趋向于依照关系式：

$$\Delta H + \Delta H' \geqslant 0$$

增加，其中 ΔH 和 $\Delta H'$分别表示系统 A 和 A'熵的改变量 (Reif, 1965)。根据式(11.11)，这个关系的含义是指系统 F 的自由能量逐渐降低至平衡态时变为最小。由统计力学我们发现此时它的概率分布为 Gibbs 分布。因而我们有一个重要的原则称为最小自由能量原则，它可以陈述如下 (Landau and Lifshitz, 1980; Parisi, 1988)：

随机系统变元的自由能量的最小值可在热平衡时达到，此时系统服从 Gibbs 分布。自然偏爱具有最小自由能量的物理系统。

11.3　马尔可夫链

考虑由多个随机变量组成的一个系统，其演化可由一个随机过程 $\{X_n, n = 1,2,\cdots\}$ 描述。随机变量 X_n 在时刻 n 取值 x_n 称为系统在 n 时刻的状态。随机变量所有可能的值构成的空间称为系统的状态空间。如果随机过程 $\{X_n, n = 1,2,\cdots\}$ 的构造使得 X_{n+1}的条件概率分布仅依靠于 X_n 的值而与其他以前的值无关，称这个过程为马尔可夫链 (Feller, 1950; Ash, 1965)。更准确地说，我们有

$$P(X_{n+1} = x_{n+1} \mid X_n = x_n, \cdots, X_1 = x_1) = P(X_{n+1} = x_{n+1} \mid X_n = x_n) \tag{11.12}$$

这称之为马尔可夫特性。换句话说：

如果系统在 $n+1$ 时刻出现状态 x_{n+1}的概率仅依赖于系统在 n 时刻出现状态 x_n 的概率，则随机变量序列 $X_1, X_2, \cdots, X_n, X_{n+1}$ 成为马尔可夫链。

因此我们可以将马尔可夫链看作产生模型，它由一些可能的状态 (成对的基础上) 转移链接而成。每时刻访问一个特定的状态，模型输出一个该状态相关的符号。

转移概率

在马尔可夫链中，从一个状态到另一个状态的转移是随机的，但输出符号却是确定的。令

$$p_{ij} = P(X_{n+1} = j \mid X_n = i) \tag{11.13}$$

表示在 n 时刻状态 i 转移到 $n+1$ 时刻状态 j 的转移概率。既然 p_{ij}为条件概率，所有的转移概率必须满足两个条件：

$$p_{ij} \geqslant 0, \quad \text{对于所有的 } i,j \tag{11.14}$$

$$\sum_j p_{ij} = 1, \quad \text{对于所有的 } i \tag{11.15}$$

将假定转移概率是固定的，不随时间改变；也就是说，式(11.13)对所有时间 n 成立。在这种情况下，马尔可夫链称为关于时间是齐次的。

如果系统具有有限数目的可能状态，例如 K 个状态，则转移概率构成一个 $K \times K$ 的矩阵

$$\mathbf{P} = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1K} \\ p_{21} & p_{22} & \cdots & p_{2K} \\ \vdots & \vdots & & \vdots \\ p_{K1} & p_{K2} & \cdots & p_{KK} \end{bmatrix} \tag{11.16}$$

它的元素满足式(11.14)和式(11.15)所述的条件，而后一条件就是 $\mathbf{P}$ 的每行的和为 1。这种类型的矩阵称为随机矩阵。任何随机矩阵可以作为转移概率矩阵。

由式(11.13)定义的一步转移概率可以推广到经过固定的步数从一个状态转移到另一个状态。令 $p_{ij}^{(m)}$ 表示从状态 i 到状态 j 的 m 步转移概率：

$$p_{ij}^{(m)} = P(X_{n+m} = x_j \,|\, X_n = x_i), m = 1,2,\cdots \tag{11.17}$$

我们可以把 $p_{ij}^{(m)}$ 看作系统从状态 i 转移到状态 j 经历的所有中间状态 k 的和。特别地，$p_{ij}^{(m+1)}$ 可由 $p_{ij}^{(m)}$ 递推而得：

$$p_{ij}^{(m+1)} = \sum_k p_{ik}^{(m)} p_{kj}, m = 1,2,\cdots \tag{11.18}$$

而

$$p_{ik}^{(1)} = p_{ik}$$

式(11.18)可以推广如下：

$$p_{ij}^{(m+n)} = \sum_k p_{ik}^{(m)} p_{kj}^{(n)}, m = 1,2,\cdots \tag{11.19}$$

这是 Chapman-Kolmogorov 恒等式的特殊情形（Feller，1950）。

马尔可夫链的详细说明

有了状态和转移概率的概念，我们现在可以将马尔可夫链具体总结如下：

(i) 一个由如下项目定义的随机模型：

- 有限 K 可能状态，表示为 $S = \{1,2,\cdots K\}$。
- 一些列相应的概率 $\{p_{ij}\}$，其中 p_{ij} 为从状态 i 到 j 的状态转移概率，并且满足

$$p_{ij} \geqslant 0$$

和

$$\sum_j p_{ij} = 1 \text{ 对所有的 } i$$

(ii) 给定已描述的随机模型，马尔可夫链是由下列一系列的随机变量 X_0，X_1，X_2，…所给定，其中它们的值根据相应的马尔可夫特征取自于状态 S：

$$P(X_{n+1} = j \,|\, X_n = i, X_{n-1} = i_{n-1}, \cdots, X_0 = i_0) = P(X_{n+1} = j \,|\, X_n = i)$$

其中对所有的时间 n 和所有的状态 i，$j \in S$ 都成立，同时所有的可能序列 $i_0, \cdots, i_{n-1}$ 涉及之前的状态。

常返性

假设一个马尔可夫链从状态 i 开始，它以概率 1 返回状态 i，则称状态 i 为常返的；也就是说

$$p_i = P(\text{状态 } i \text{ 的每一个返回}) = 1$$

若概率 $p_i < 1$，则称状态 i 为瞬态（Leon-Garcia，1994）。

如果马尔可夫链从一个常返态开始，则该状态在时间上将无穷次重现。如果从一个瞬态开始，它将只能有限次重现。这可以作如下解释：我们可以把状态 i 重新发生看作一个成功概率为 p_i 的 Bernoulli 试验[2]。它返回的次数为具有均值 $(1-p_i^{-1})$ 的几何随机变量。若 $p_i<1$，这意味着有无穷次成功的次数为零。因此一个瞬态确实在有限次返回后不再发生。

如果一个马尔可夫链有某些瞬态和常返状态，则该过程最终只会在常返态之间移动。

周期性

图 11.1 显示一个具有常返态的马尔可夫链。此链经过一系列的子态，经过三倍次移动之后以相同子态结束。图示说明这个常返的马尔可夫链具有周期性。

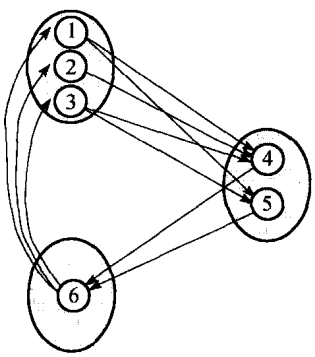


图 11.1　一个 $d=3$ 周期性常返的马尔可夫链

就图 11.1 而言，一个常返的马尔可夫链如果是具有周期性的是指所有状态能被编入 d 个各不相交的子集 $S_1,S_2,\cdots,S_d$，其中 $d>1$，而且所有的从一个子集到另一个子集的转移都有这种方式，在此图中，$d=3$。更精确地，一个周期性常返的马尔可夫链是指满足以下条件 (Bertsekas and Tsitsiklis, 2002)：

$$\text{如果 } i\in S_k \text{ 并且 } p_{ij}>0,\text{则}\begin{cases} j\in S_{k+1},\text{当 } k=1,\cdots,d-1 \\ j\in S_1,\text{当 } k=d \end{cases}$$

一个常返的马尔可夫链是不定期的是指它不具有周期性。

不可约马尔可夫链

一个马尔可夫链上的状态 j 称为从状态 i 可达的，如果从状态 i 到 j 存在有限步具有正概率的转移。如果状态 i 和状态 j 之间互为可达的，则该马尔可夫链的状态 i 和状态 j 称为彼此相通的。这种相通可写作 $i\leftrightarrow j$。很明显，如果状态 i 与状态 j 相通，且状态 j 与状态 k 相通，即 $i\leftrightarrow j$ 和 $j\leftrightarrow k$，则状态 i 和状态 k 相通（即 $i\leftrightarrow k$）。

如果马尔可夫链的两个状态相通，则其属于同一类。一般情况下，一个马尔可夫链的状态组成一个或多个不相通的类。但是，如果所有状态组成一个类，则称该马尔可夫链为不可分的或者不可约的。换句话说，一个不可约的马尔可夫链从任一个状态开始，可以以正的概率达到任何别的状态。可约链在大多数的应用领域无实际价值。相应地我们限制我们的注意仅在不可约的链。

考虑一个不可约的马尔可夫链，在时刻 $n=0$ 时开始于常返态 i。令 $T_i(k)$ 表示第 $k-1$ 次和第 k 次返回状态 i 之间的时间间隔。状态 i 的平均常返时间定义为 $T_i(k)$ 关于 k 的期望值。状态 i 的稳态概率，记为 π_i，等于平均常返时间$\mathbb{E}[T_i(k)]$ 的倒数，即由下式表示：

$$\pi_i=\frac{1}{\mathbb{E}[T_i(k)]}$$

如果$\mathbb{E}[T_i(k)]<\infty$，也就是 $\pi_i>0$，状态 i 称为一个正常返（持久的）态。若$\mathbb{E}[T_i(k)]=\infty$，也就是 $\pi_i=0$，状态 i 称为一个零常返（持久的）态。$\pi_i=0$ 意味着马尔可夫链最终达到的状态再返回状态 i 是不可能的。正常返和零常返是不同类的性质，这意味着同时具有正常返和零常返的马尔可夫链是可约的。

遍历马尔可夫链

大体上说，遍历性意味着我们可以用时间的平均替代总体平均。对一个马尔可夫链来说，遍历性意味着链处于状态 i 的时间长度和稳态概率 π_i 相对应，这可以说明如下：k 次返回后花费在状态 i 的时间，用 $v_i(k)$ 表示，定义为

$$v_i(k) = \frac{k}{\sum_{l=1}^{k} T_i(l)}$$

返回时间 $T_i(l)$构成一列独立的和同分布（iid）的随机变量，因为由定义每次返回的时间都是和以前返回的时间统计独立的。更进一步，对常返态 i，链返回状态 i 无穷次。因此当返回次数 k 逼近无穷大时，大数定律表明，花费在状态 i 的时间比例趋近稳态概率，表示为

$$\lim_{k\to\infty} v_i(k) = \pi_i, \quad \text{当 } i = 1,2,\cdots,K \tag{11.20}$$

其中 K 是状态的个数。

马尔可夫链为遍历的一个充分但不必要的条件是：它为不可约的且非周期的。

收敛于平衡分布

考虑一个遍历的马尔可夫链，相应的转移矩阵为 $\mathbf{P}$。令行向量 π^{n-1}表示链在 $n-1$ 时刻的状态分布向量；π^{n-1}的第 j 个分量为在时刻 $n-1$ 时链处于状态 x_j 的概率。在 n 时刻状态分布向量可以定义为：

$$\boldsymbol{\pi}^{(n)} = \boldsymbol{\pi}^{(n-1)}\mathbf{P} \tag{11.21}$$

由式(11.21)迭代得到：

$$\boldsymbol{\pi}^{(n)} = \boldsymbol{\pi}^{(n-1)}\mathbf{P} = \boldsymbol{\pi}^{(n-2)}\mathbf{P}^2 = \boldsymbol{\pi}^{(n-3)}\mathbf{P}^3 = \cdots$$

并且最后可以写成：

$$\boldsymbol{\pi}^{(n)} = \boldsymbol{\pi}^{(0)}\mathbf{P}^n \tag{11.22}$$

其中 $\boldsymbol{\pi}^{(0)}$是状态分布向量的初始值。也就是说：

马尔可夫链在时刻 n 状态分布向量为初始状态分布向量 $\boldsymbol{\pi}^{(0)}$ 和随机矩阵 $\mathbf{P}$ 的 n 次方的乘积。

令 $p_{ij}^{(n)}$表示 p^n 的第 ij 个元素。假设随时间 n 趋向无穷大时，$p_{ij}^{(n)}$趋于与 i 无关的 $\boldsymbol{\pi}_j$，其中 $\boldsymbol{\pi}_j$ 为状态 j 的稳态概率。相应地，对于大的 n，矩阵 $\mathbf{P}^n$ 逼近于有相等行的方阵形式，可表示为：

$$\lim_{n\to\infty}\mathbf{P}^n = \begin{bmatrix} \pi_1 & \pi_2 & \cdots & \pi_K \\ \pi_1 & \pi_2 & \cdots & \pi_K \\ \vdots & \vdots & & \vdots \\ \pi_1 & \pi_2 & \cdots & \pi_K \end{bmatrix} = \begin{bmatrix} \boldsymbol{\pi} \\ \boldsymbol{\pi} \\ \vdots \\ \boldsymbol{\pi} \end{bmatrix} \tag{11.23}$$

其中 $\boldsymbol{\pi}$ 是行向量由 π_1，π_2，π_k 构成。从而由式(11.22)发现（经过一系列调整）：

$$\left[\sum_{j=1}^{K}\pi_j^{(0)} - 1\right]\boldsymbol{\pi} = \mathbf{0}$$

因为由定义 $\sum_{j=1}^{K}\pi_j^{(0)} = 1$，初始分布的独立向量 $\boldsymbol{\pi}$ 满足这个条件。

现在我们可以叙述关于马尔可夫链的遍历定理如下（Feller，1950；Ash，1965）：

设一个遍历且不可约的马尔可夫链具有状态 x_1，x_2，…，x_K 和随机矩阵 $\mathbf{P}=\{p_{ij}\}$。那么，该链有唯一的平稳分布，可以由任一初始态收敛到它；也就是说，存在唯一一组数 $\{\pi_j\}_{j=1}^{K}$使得

1. $\lim_{n\to\infty} p_{ij}^{(n)} = \pi_j$，　对于所有 i (11.24)
2. $\pi_j > 0$，　对于所有 j (11.25)
3. $\sum_{j=1}^{K}\pi_j = 1$ (11.26)

$$4.\quad \pi_j = \sum_{i=1}^{K} \pi_i p_{ij}, \quad \text{对 } j = 1,2,\cdots,K \tag{11.27}$$

相反，假定一个马尔可夫链为非周期不可约的，存在 $\{\pi_j\}_{j=1}^{K}$ 满足式(11.25)至式(11.27)，那么该链是遍历的，π_j 由式(11.24)给出，状态 j 的平均常返时间为 $1/\pi_j$。

概率分布函数 $\{\pi_j\}_{j=1}^{K}$ 称为不变分布或平稳分布。这样命名是因为它一旦建立，将永远保持。根据遍历定理，我们可以断言：

1. 从任意初始分布开始，一个马尔可夫链的转移概率将收敛于一个平稳分布，只要这个平稳分布存在。

2. 遍历的马尔可夫链的平稳分布独立于它的初始分布。

例 1　一个可遍历的马尔可夫链

考虑一个马尔可夫链，其状态转移图由图 11.2 描绘，它有两个状态 x_1 和 x_2。链的随机矩阵为：

$$\mathbf{P} = \begin{bmatrix} \frac{1}{4} & \frac{3}{4} \\ \frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

它满足式(11.4)和式(11.5)的条件。假设初始条件是

$$\boldsymbol{\pi}^{(0)} = \begin{bmatrix} \frac{1}{6} & \frac{5}{6} \end{bmatrix}$$

由式(11.21)我们发现在时刻 $n=1$ 状态分布向量为

$$\boldsymbol{\pi}^{(1)} = \boldsymbol{\pi}^{(0)}\mathbf{P} = \begin{bmatrix} \frac{1}{6} & \frac{5}{6} \end{bmatrix} \begin{bmatrix} \frac{1}{4} & \frac{3}{4} \\ \frac{1}{2} & \frac{1}{2} \end{bmatrix} = \begin{bmatrix} \frac{11}{24} & \frac{13}{24} \end{bmatrix}$$

升高随机矩阵 $\mathbf{P}$ 的幂次为 $n=2$，3，4，得到

$$\mathbf{P}^2 = \begin{bmatrix} 0.4375 & 0.5625 \\ 0.3750 & 0.6250 \end{bmatrix} \quad \mathbf{P}^3 = \begin{bmatrix} 0.4001 & 0.5999 \\ 0.3999 & 0.6001 \end{bmatrix} \quad \mathbf{P}^4 = \begin{bmatrix} 0.4000 & 0.6000 \\ 0.4000 & 0.6000 \end{bmatrix}$$

因此 $\pi_1 = 0.4000$ 和 $\pi_2 = 0.6000$。在这个例子中，平稳分布的收敛基本上在 $n=4$ 次迭代就完成了。由于 π_1 和 π_2 都大于零，两个状态都是正常返的，并且链为不可约的。同时注意它是非周期的，这是因为使 $(\mathbf{P}^n)_{jj} > 0$ 的所有正整数 $n \geqslant 1$ 的最大公因数是 1。因此得出结论图 11.2 的马尔可夫链是遍历的。 ■

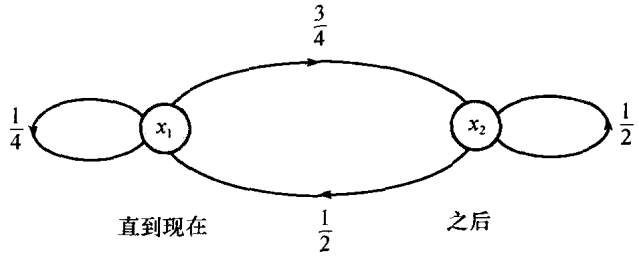


图 11.2　例 1 的马尔可夫链的状态转移图：x_1 和 x_2 分别以直到现在和之后标明

例 2　一个具有平稳分布的遍历马尔可夫链

考虑随机矩阵具有某些零元素的马尔可夫链，如

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 1 \\ \frac{1}{3} & \frac{1}{6} & \frac{1}{2} \\ \frac{3}{4} & \frac{1}{4} & 0 \end{bmatrix}$$

该链的状态转移图由图 11.3 描绘。

应用式(11.27)得到下列联立方程组：

$$\pi_1 = \frac{1}{3}\pi_2 + \frac{3}{4}\pi_3$$

$$\pi_2 = \frac{1}{6}\pi_2 + \frac{1}{4}\pi_3$$

$$\pi_3 = \pi_1 + \frac{1}{2}\pi_2$$

解关于 π_1，π_2 和 π_3 的方程组，得到

$$\pi_1 = 0.395\,3$$

$$\pi_2 = 0.139\,5$$

$$\pi_3 = 0.465\,2$$

这个给定的马尔可夫链是遍历的，它的平稳分布由 π_1、π_2 和 π_3 定义。 ■

状态分类

在所述材料的基础上，我们可以对状态所属的类进行小结，如图 11.4 所示（Feller，1950；Leon-Garcia，1994）。这个图还包括状态相关的长期行为。

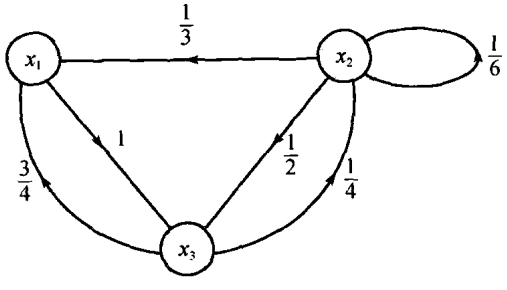


图 11.3　例 2 的马尔可夫状态转移图

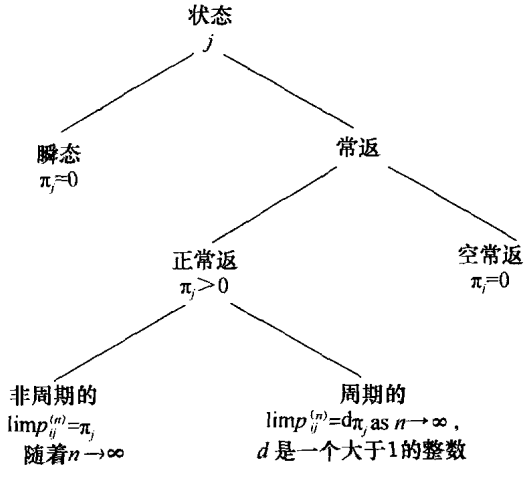


图 11.4　马尔可夫链的状态分类和它们相应的长期行为

细节平衡原则

这一原则通常在统计力学中使用。细节平衡原则表明：

在热平衡中任何转移的发生率等于对应的逆转移的发生率，可表达为：

$$\pi_i p_{ij} = \pi_j p_{ji} \tag{11.28}$$

一个马尔可夫链满足细节平衡原则称为可逆的。

为了说明原则的应用，我们将用它来导出式(11.27) 的关系，它是平稳分布的定义，我们可以对等式的左边进行求和如下：

$$\sum_{i=1}^{K}\pi_i p_{ij} = \sum_{i=1}^{K}\left(\frac{\pi_i}{\pi_j}p_{ij}\right)\pi_j = \sum_{i=1}^{K}(p_{ji})\pi_j = \pi_j$$

在等式的第二行中我们应用了细节平衡原则，在最后一行利用了一个马尔可夫链的转移概率满足的条件（参看式(11.15)，其中交换了 i 和 j 的作用）：

$$\sum_{i=1}^{K}p_{ji} = 1, \quad \text{对于所有 } j$$

从上述讨论，因而断定细节平衡原则意味着 $\{\pi_j\}$ 是一个平稳分布。就平稳分布的范围而言，细节平衡原则比式(11.27)更强，在这个意义上它对平稳分布是充分的，不是必要的。

11.4　Metropolis 算法

至此我们弄清了马尔可夫链的构成，我们将应用它构成一个模拟物理系统演化到热平衡的随机算法。这个算法称为 Metropolis 算法（Metropolis 等，1953）。它是 Monte Carlo 方法的一种修改，在早期的科学计算中 Monte Carlo 方法是对大量原子在给定温度下的平衡态的随机模拟。

由于它是 Monte Carlo 方法的修改，所以 Metropolis 算法也通常被称为 Markov chain Monte Carlo(MCMC) 方法。在上下文中，我们可以正式地陈述以下定义（Robert and Casella，1999）：

对于模拟一个未知分布的 Markov Chain Monte Carlo 方法是指产生一个遍历的马尔可夫链而它的平稳分布是未知的。

Metropolis 算法非常完美地符合这个定义，同样对它的推广形式 Metropolis-Hastings 算法[3] 也是如此。

Metropolis 算法的统计分析

假设随机变量 X_n 表示任一马尔可夫链在时刻 n 的状态为 x_i。我们随机生成新的状态 x_j，它表示另一个随机变量 Y_n 的一次实现。假设生成这个新状态满足对称条件：

$$P(Y_n = x_j \mid X_n = x_i) = P(Y_n = x_i \mid X_n = x_j)$$

令 ΔE 表示系统从状态 $X_n = x_i$ 到状态 $Y_n = x_i$ 所产生的能量差。我们进行如下处理：

1. 如果能量差 ΔE 为负，则这次转移导致一个较低能量状态且这次转移被接受。这个新状态也就接受作为算法下步的起点，即我们令 $X_{n+1} = Y_n$。

2. 反之如果能量差 ΔE 为正，这时算法以概率方式进行处理。首先，我们选择一个在单位区间 [0，1] 上均匀分布的随机数 ξ。如果 $\xi < \exp(-\Delta E/T)$，其中 T 为操作温度，转移被接受且置 $X_{n+1} = Y_n$。否则，转移被拒绝，置 $X_{n+1} = X_n$；即旧的配置被算法的下一步重新利用。

转移概率的选择

对任意马尔可夫链，设它有先验转移概率，记为 τ_{ij}，它满足三个条件：

1. 非负性：　$\tau_{ij} \geqslant 0$，　对于所有 i,j

2. 归一化：　$\sum_j \tau_{ij} = 1$，　对于所有 i

3. 对称性：　$\tau_{ji} = \tau_{ji}$，　对于所有 i,j

令 π_i 表示马尔可夫链在状态 $x_i (i = 1,2,\cdots,K)$ 的平稳态概率。因而我们可以利用已定义的对称的 τ_{ij} 和概率分布比 π_j/π_i 来构成期望的转移概率（Beckerman，1997）：

$$p_{ij} = \begin{cases} \tau_{ij}\left(\dfrac{\pi_j}{\pi_i}\right), & \text{当} \dfrac{\pi_j}{\pi_i} < 1 \\ \tau_{ij}, & \text{当} \dfrac{\pi_j}{\pi_i} \geqslant 1 \end{cases} \tag{11.29}$$

为了确保转移概率归一化为单位 1，我们引入无转移概率的附加定义：

$$p_{ii} = \tau_{ii} + \sum_{j \neq 1} \tau_{ij} \left(1 - \frac{\pi_j}{\pi_i}\right) = 1 - \sum_{j \neq i} \alpha_{ij} \tau_{ij} \tag{11.30}$$

其中 α_{ij} 是移动概率，定义为

$$\alpha_{ij} = \min\left(1, \frac{\pi_j}{\pi_i}\right) \tag{11.31}$$

唯一尚需解决的要求是怎样选择比值 π_j/π_i。为满足这个要求，我们选择概率分布使得所得的马尔可夫链收敛到一个 Gibbs 分布，表示为：

$$\pi_j = \frac{1}{Z} \exp\left(-\frac{E_j}{T}\right)$$

这时概率分布比 π_j/π_i 的简单形式为：

$$\frac{\pi_j}{\pi_i} = \exp\left(-\frac{\Delta E}{T}\right) \tag{11.32}$$

其中

$$\Delta E = E_j - E_i \tag{11.33}$$

利用概率分布比可以排除对剖分函数 Z 的依赖。

根据构造，转移概率是非负的且归整化为单位 1，如式(11.14)和式(11.15)的要求。进一步，它们满足由式(11.28)所定义的细节平衡原则。这个定律对热平衡是一个充分条件。为了说明满足细节平衡原则，我们给出下列的考虑：

情况 1：$\Delta E < 0$。假设从状态 π_i 转移到状态 π_j，能量变化 ΔE 为负。从式(11.32)我们发现 $\pi_j/\pi_i > 1$，所以利用式(11.29)得到

$$\pi_i p_{ij} = \pi_i \tau_{ij} = \pi_i \tau_{ji}$$

和

$$\pi_j p_{ji} = \pi_j \left(\frac{\pi_i}{\pi_j} \tau_{ji}\right) = \pi_i \tau_{ji}$$

因此当 $\Delta E < 0$ 时满足细节平衡原则。

情况 2：$\Delta E > 0$。假设从状态 x_i 到状态 x_j 的能量变化 ΔE 为正，这时我们发现$(\pi_j/\pi_i) < 1$，利用式(11.29)得到

$$\pi_i p_{ij} = \pi_i \left(\frac{\pi_j}{\pi_i} \tau_{ij}\right) = \pi_j \tau_{ij} = \pi_j \tau_{ji}$$

和

$$\pi_j p_{ji} = \pi_i p_{ij}$$

这里细节平衡原则得到满足。

为了完整起见，我们需要指出由 τ_{ij} 表示的先验转移概率的使用。这些转移概率事实上是 Metropolis 算法中的随机步的概率模型。由前面的算法描述，我们回忆随机步后面是随机决策。因此可以得出结论，利用由先验转移概率 τ_{ij} 在式(11.29)和式(11.30)定义的转移概率 p_{ij} 和平稳概率分布 π_j 对 Metropolis 算法来说确实是正确的选择。

我们可以得出由 Metropolis 算法产生一个马尔可夫链[4]。它的转移概率确实收敛到一个独一平稳的 Gibbs 分布(Beckerman，1997)。

11.5　模拟退火

考虑寻找一个低能量系统的问题，其状态由一个马尔可夫链排序。由式(11.11)观察到当温度 T 趋近于零，系统的自由能量 F 趋近平均能量 $\langle E \rangle$。由 $F \to \langle E \rangle$，我们观察到由自由能量

最小化原则，该马尔可夫链的平稳分布（即 Gibbs 分布），当 $T\to 0$ 时坍塌到平均能量 $\langle E\rangle$ 的全局极小点。换句话说，序列中的低能状态在低温时受到更强的支持。这些观察促使我们提出问题：为什么不简单地应用 Metropolis 算法产生大量的代表该随机系统在很低温度下的构形（Configuration）？我们不提倡使用这种策略是因为在很低温度下马尔可夫链到热平衡的收敛速度特别慢。而提高计算效率更好的方法是在较高温度运行随机系统，这时达到平衡态的收敛相当快，接着随温度的精细下降保持系统的平衡态。也就是，我们使用两个相关成分的组合：

1. 一个决定温度下降速度的调度表。

2. 一个算法（如 Metropolis 算法）迭代求解每个调度表给出的新的温度下的平衡分布，这时利用前面温度时的最终状态作为新温度时的起始点。

我们刚才提到的两步格式是被广泛使用的以*模拟退火*[5] 著称的随机松弛技术的精华（Kirkpatrick 等，1983）。这个技术的名字是类比物理/化学中的退火过程得到的，在物理/化学的退火过程中，我们从高温度开始退火过程，接着慢慢降低温度同时保持热平衡。

模拟退火最初的目标是寻找刻画复杂大系统的代价函数的全局极小点。正是因为如此，它提供一个求解非凸最优化问题的有力工具，这由下面的简单想法所导致：

当优化一个非常复杂的大系统（即具有许多自由度的系统）时不要求总是下降而是试图要求大部分时间在下降。

模拟退火在两方面与传统的迭代优化算法不同：

1. 算法不会陷入局部最小，因为当系统在非零温度上运行时脱离局部最小总是可能的。

2. 模拟退火是*自适应的*，在高温时看见系统的终态的大致轮廓，而它的具体细节在低温度时才呈现出来。

退火进度表

如前面提到的，模拟退火过程的基础是 Metropolis 算法，其间温度 T 慢慢下降。也就是说，温度 T 起*调节参数*的作用。假定温度下降没有对数快，则模拟退火过程将收敛于一个具有最小能量的构形。遗憾的是这种退火进度太慢了——慢得不切实际。实际上，我们必须求诸于算法的渐进收敛的*有限时间逼近*。这种逼近所付出的代价是算法不再以概率 1 保证找到全局最小点。然而算法的逼近结果在许多实际应用上能产生近似最优解。

为了实现模拟退火算法的有限时间逼近，我们必须设定一系列控制算法收敛的参数，这些参数组合成所谓的*退火进度表*或*冷却进度表*。退火进度表设定一个温度的有限序列值，以及每一温度值下有限的转移尝试的次数。Kirkpatrick 等（1983）给出的退火进度表的感兴值的参数设定如下[6]：

1. *温度的初始值*。温度的初始值 T_0 选得足够高使得所有提出的转移实际都能被模拟退火算法所接受。

2. *温度的下降*。一般地说，冷却是*按指数形式*完成的，并且温度值的改变量都很小。特别地，*下降函数*定义为

$$T_k = \alpha T_{k-1}, k = 1,2,\cdots \tag{11.34}$$

其中 α 小于但接近于 1。α 的典型值介于 0.8 和 0.99 之间。对每一温度，有足够的转移的尝试，使得平均每次实验有 10 次转移被接受。

3. *温度的最后值*。如果在三次相连的温度下没有得到预期的接收次数，则系统被冻结且退火停止。

后一个标准可以改进，要求*接受率*小于一预定值，而接受率定义为转移接受的次数除以提

出转移的次数（Johnson 等，1989）。

模拟退火用于组合优化

模拟退火特别适用于解组合优化问题。组合优化的目标是针对有很多可能解的有限离散系统，最小化它的代价函数。本质上讲模拟退火利用 Metropolis 算法通过多粒子物理系统和组合优化问题间的类比生成一系列解。

在模拟退火中，我们把式(11.5)的 Gibbs 分布中的能量 E_i 解释成为数值的代价，而温度 T 解释为控制参数。在组合优化问题中对每一构形赋予一数值的代价以描述这个特殊的构形和解的差异。模拟退火程序中下一个需要考虑的问题是如何确认构形和从已有构形以局部方式产生新的构形。这就是 Metropolis 算法发挥作用之处。因此我们概括统计物理的术语和组合优化术语之间的关系如表 11.1 所示(Beckerman，1997)。

表 11.1　统计物理与组合优化之间的对应

统计物理	组合优化
样本	问题实例
状态（构形）	构形
能量	代价函数
温度	控制参数
基态能量	最小代价
基态构形	最优构形

11.6　Gibbs 抽样

类似于 Metropolis 算法，Gibbs *抽样器*[7] 生成一个马尔可夫链，它以 Gibbs 分布作为平衡分布。但是 Gibbs 抽样器的转移概率是非平稳的（Geman and Geman，1984）。在最后的分析里，关于 Gibbs 抽样和 Metropolis 算法的选择取决于具体问题的技术细节。

为了继续描述这个抽样格式，考虑一个 K 维的随机向量 $\mathbf{X}$，由分量 $X_1, X_2, \cdots, X_K$ 构成。假定在给定 X 的其他分量时我们知道 X_k 的条件分布，$k = 1,2,\cdots,K$。我们想问的问题是：对任何 k，怎样获得随机变量 X_k 的边缘密度的数值估计。对随机向量 $\mathbf{X}$ 的每个分量，在已知 $\mathbf{X}$ 的其他分量值的条件下，Gibbs 抽样器对它的条件分布产生一个值。特别地，从任意构形 $\{x_1(0), x_2(0), \cdots, x_K(0)\}$ 开始，我们在 Gibbs 抽样的第一次迭代时做下列采样：

$x_1(1)$ 是在已知 $x_2(0), x_3(0), \cdots, x_K(0)$ 时由 X_1 的分布产生的采样。

$x_2(1)$ 是在已知 $x_1(1), x_3(0), \cdots, x_K(0)$ 时由 X_2 的分布产生的采样。

…

$x_k(1)$ 是在已知 $x_1(1), \cdots, x_{k-1}(1), x_{k+1}(0), \cdots, x_K(0)$ 时由 X_k 的分布产生的采样。

$x_K(1)$ 是在已知 $x_1(1), x_2(1), \cdots, x_{K-1}(1)$ 时由 X_K 的分布产生的采样。

在第二次迭代和其他的每次抽样迭代中我们用这种方式进行处理。以下两点需要特别注意：

1. 随机向量 $\mathbf{X}$ 的每个分量是以自然序列“访问”的，每次迭代产生总共 K 个新的变量值。

2. 对于 $k = 2,3,\cdots,K$，在对 X_k 采样新值时直接利用分量 X_{K-1} 的新的值。

由这个讨论我们看到 Gibbs 采样是迭代的自适应格式。利用它进行 n 次迭代后，我们得到 K 个变化量：$X_1(n), X_2(n), \cdots, X_K(n)$。在相当温和的条件下，以下三个定理对 Gibbs 抽样成立（Geman and Geman,1984;Gelfand and Smith,1990）：

1. *收敛定理*。当 $k = 1,2,\cdots,K, n$ 趋于无穷大时，随机变量 $X_k(n)$ 依分布收敛于 X_k 的真实概率分布；也就是说，

$$\lim_{n\to\infty} P(X_k^{(n)} \leqslant x \mid x_k(0)) = P_{X_k}(x), \quad \text{当 } k = 1,2,\cdots,K \tag{11.35}$$

其中 $P_{X_k}(x)$ 为 X_k 的边缘概率分布函数。

事实上，在 Geman and Geman(1984) 中证明了更强的结果。特别地，不要求随机向量 $\mathbf{X}$ 的每个分量以自然顺序被重复访问，任意的访问方式只要不依赖于变量的值且 $\mathbf{X}$ 的每个分量被

"无限地经常"访问，则 Gibbs 抽样收敛性仍成立。

2. *收敛速度定理*。随机变量 $X_1(n), X_2(n), \cdots, X_K(n)$ 的联合概率分布以 n 的几何级数速度收敛于 $X_1, X_2, \cdots, X_K$ 的联合分布函数。

这个定理假设 $\mathbf{X}$ 的分量以自然顺序访问。但是当任意的但无限地经常访问时，收敛速度需要较小的调整。

3. *遍历定理*。对任何（例如对于随机变量 $X_1, X_2, \cdots, X_K$）的可测函数 g，它的期望存在，有

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} g(X_1(i), X_2(i), \cdots, X_K(i)) \to \mathbb{E}[g(X_1, X_2, \cdots, X_K)] \tag{11.36}$$

以概率 1（即几乎肯定）实现。

遍历定理告诉我们怎样利用 Gibbs 采样的输出获得所期望的边缘密度的数值估计。

在 Boltzmann 机中使用 Gibbs 采样对有关隐藏神经元的分布进行采样；这种随机机器将在下一节讨论。对于使用二值单元的随机机器（即 Boltzmann 机）来说，值得注意的是 Gibbs 采样正好和 Metropolis 算法的一个变体完全一样。在 Metropolis 算法的标准形式中我们以概率 1 下山，相反地在 Metropolis 算法的另一个形式中，我们以 1 或能量差的指数（即上山规则的补充）的概率下山。换句话说，如果一个变化降低了能量 E 或 E 没有变化时，则这个变化被接受；如果变化升高了能量，它是以 $\exp(-\Delta E)$ 的概率被接受，否则被拒绝，而以旧的状态重复（Neal，1993）。

11.7　Boltzmann 机

Boltzmann 机是由随机神经元组成的二值随机机器，随机神经元以概率方式取两个可能状态之一。这两个状态可以指定为 $+1$，表示"开"状态，指定为 -1 表示"关"状态，或分别用 1 和 0 表示。我们将采用前面的记号。Boltzmann 机另一个突出的特征就是它的神经元间使用对称的突触连接，这种形式的突触连接也有统计物理方面的考虑。

Boltzmann 机的随机神经元分成两部分功能组，如图 11.5 所示为可见部分和隐藏部分。可见神经元[8]提供网络和它运行环境之间的一个界面。在网络的训练阶段，所有可见神经元都被钳制在环境所决定的特定状态。另一方面，隐藏神经元总是自由运行的，它们用来解释环境输入向量包含的固有约束。隐藏神经元通过捕获钳制向量中的高阶统计相关来完成这项任务。这里所叙述的网络代表 Boltzmann 机的一种特殊情况。它可以看成是对某确定概率分布建模的无监督学习程序，该确定概率分布决定于在可见神经元上以合适的概率钳制模式。这样做，网络能起到模式完形（pattern completion）的作用。特别地，当一部分携带信息的向量钳制在可见神经元的子集上，如果网络已经恰当地学会了训练分布，这时网络能够对剩下的可见神经元网络给出它们的恰当的值，起到模式完形的作用。

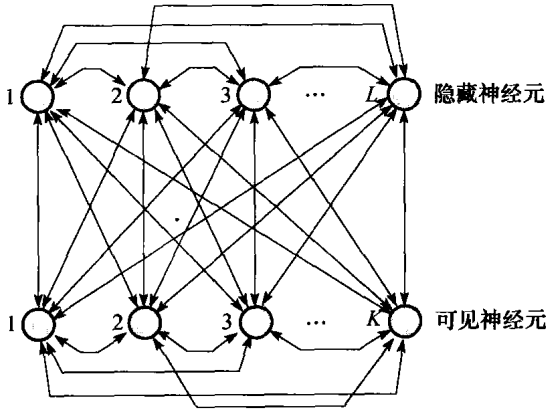


图 11.5　Boltzmann 机体系结构图；K 为可见神经元数目，L 为隐藏神经元数目。Boltzmann 的优点是：1. 可见神经元和隐藏神经元的连接是对称的。2. 对称连接延伸到可见神经元和隐藏神经元

Boltzmann 机学习的主要目的是产生一个神经网络，根据 Boltzmann 分布对输入模式进行正确的建模。在这种学习的应用中，假设两种情况：

1. 每个环境输入向量（模式）持续足够长的时间，允许网络达到热平衡。

2. 环境向量钳制在网络可见单元上的次序是没有任何结构的。

一组特定的突触权值当它导出的可见单元状态的概率分布（当网络自由运行时）和可见单元被环境输入向量所钳制时的状态概率分布完全一样，我们说它构造了环境结构的一个完整模型。一般情况下，除非隐藏单元数目是可见单元数目的指数，否则不可能得到完整模型。但是，如果环境有规则的结构，网络利用隐藏单元捕获这些规则，这时利用较小能处理的隐藏神经元数目可以对环境取得一个好的匹配。

Boltzmann 机的 Gibbs 抽样和模拟退火

令 $\mathbf{x}$ 表示 Boltzmann 机的状态向量，它的分量 x_i 表示神经元 i 的状态。状态 $\mathbf{x}$ 代表随机向量 $\mathbf{X}$ 的一次实现。从神经元 i 到神经元 j 的突触连接记为 w_{ij}，满足：

$$w_{ji} = w_{ij}, \quad \text{对于所有 } i,j \tag{11.37}$$

和

$$w_{ii} = 0, \quad \text{对于所有 } i \tag{11.38}$$

式(11.37)描述对称性，而式(11.38)强调无自反馈。偏置可以利用一个输出恒为+1 的虚节点到神经元 j(对所有 j) 的连接权值 w_{j0} 表示。

类似于热动力学，Boltzmann 机的能量可定义为[9]：

$$E(\mathbf{x}) = -\frac{1}{2}\sum_{i}\sum_{j\neq i} w_{ji}x_i x_j \tag{11.39}$$

利用式(11.5)的 Gibbs 分布，我们可以定义网络（假定处在温度 T 的平衡态）在状态 $\mathbf{x}$ 的概率如下：

$$P(\mathbf{X}=\mathbf{x}) = \frac{1}{Z}\exp\left(-\frac{E(\mathbf{x})}{T}\right) \tag{11.40}$$

其中 Z 为剖分函数。

为了简化表示，定义单个事件 A 及联合事件 B 和 C 如下：

$$A: X_j = x_j$$

$$B: \{X_i = x_i\}_{i=1}^{K} \text{ with } i \neq j$$

$$C: \{X_i = x_i\}_{i=1}^{K}$$

实际上，联合事件 B 排斥 A，而联合事件 C 包括 A 和 B。B 的概率是 C 关于 A 的边缘概率。因此，利用式(11.39)和式(11.40)，我们可写作：

$$P(C) = P(A,B) = \frac{1}{Z}\exp\left(\frac{1}{2T}\sum_{i}\sum_{j\neq i} w_{ji}x_i x_j\right) \tag{11.41}$$

和

$$P(B) = \sum_{A} P(A,B) = \frac{1}{Z}\sum_{x_j}\exp\left(\frac{1}{2T}\sum_{i}\sum_{j\neq i} w_{ji}x_i x_j\right) \tag{11.42}$$

在式(11.41)和式(11.42)中的指数可以表示成两项之和，一项与 x_j 有关，而另一项与 x_j 无关。包含 x_j 的项为：

$$\frac{x_j}{2T}\sum_{\substack{i\\ i\neq j}} w_{ji}x_i$$

相应地，给定 B，置 $x_j = x_i = \pm 1$，我们可以给出 A 的条件概率

$$P(A|B) = \frac{P(A,B)}{P(B)} = \frac{1}{1+\exp\left(-\frac{x_j}{T}\sum_{\substack{i\\ i\neq j}} w_{ji}x_i\right)}$$

也就是可写成：

$$P(X_j = x \mid \{X_i = x_i\}_{i=1, i\neq j}^{K}) = \varphi\left(\frac{x}{T}\sum_{\substack{i\\ i\neq j}}^{K} w_{ji}x_i\right) \tag{11.43}$$

其中 $\varphi(\cdot)$为它变元的 logistic 函数，表示为

$$\varphi(v) = \frac{1}{1+\exp(-v)} \tag{11.44}$$

注意，x 虽然在 -1 和 $+1$ 间变化，但当 v 充分大时，整个变量 $v=\frac{x}{T}\sum_{j\neq j} w_{ji}x_i$ 可在 $-\infty$和 $+\infty$之间变化，如图 11.6 所描画的。同时注意，在推导式(11.43)时，不需剖分函数 Z，这是高度期望的，因为对于非常复杂的网络直接计算 Z 是不现实的。

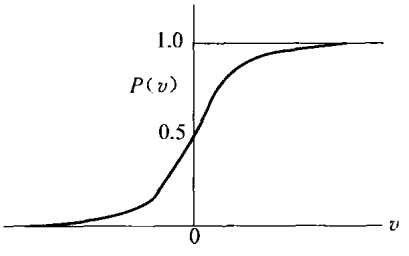


图 11.6　Sigmoid 形状函数 $P(v)$

利用 Gibbs 抽样表示联合分布 $P(A, B)$。基本上，如 11.6 节所解释的那样，这个随机模拟开始时给网络赋予任一状态，神经元以它们的自然顺序依次重复访问，每次访问，选择一个神经元，根据其他神经元的值确定该神经元状态新值的选择概率。假定这个随机模拟进行足够长的时间，则网络将达到在温度 T 下的平衡。

遗憾的是到达热平衡的时间可能非常长。为了克服这个困难，如同在 11.5 节所解释的那样，对有限温度序列 $T_0, T_1, \cdots, T_{\text{final}}$，使用模拟退火。特别地，温度被初始化为一个高的值 T_0，因此可迅速到达热平衡。然后，温度 T 逐渐降低至最后值 T_{final}，这时神经元状态将（有希望）达到它们的边缘分布。

Boltzmann 学习规则

因为 Boltzmann 机是一种随机机器，它自然依赖于用概率论评价其性能。这种标准之一是*似然函数*[10]。在此基础上，根据*最大似然原则*，Boltzmann 学习的目标是最大化似然函数或等价的对数似然函数，这个原则在第 10 章中讨论过。

令$\mathcal{T}$表示感兴趣的概率分布抽样所组成的训练样本。假设它们都是二值的。训练样本允许重复，但必须和它们发生的概率成比例。令状态向量 $\mathbf{x}$ 的子集 $\mathbf{x}_\alpha$ 表示可见神经元状态。向量 x 的剩余部分 $\mathbf{x}_\beta$ 表示隐藏神经元的状态。状态向量 $\mathbf{x}$，$\mathbf{x}_\alpha$ 和 $\mathbf{x}_\beta$ 分别表示随机向量 $\mathbf{X}$，$\mathbf{X}_\alpha$ 和 $\mathbf{X}_\beta$ 的实现。Boltzmann 机的运行分成两个阶段：

1. *正向阶段*。此时网络在钳制环境下（即在训练集$\mathcal{T}$的直接影响下）运行。

2. *负向阶段*。在第二阶段，网络允许自由运行，因此没有环境输入。

对整个网络给定突触间权值 $\mathbf{w}$，可见神经元状态为 $\mathbf{x}_\alpha$ 的概率是 $P(\mathbf{X}_\alpha=\mathbf{x}_\alpha)$。训练集$\mathcal{T}$中包含许多可能值 $\mathbf{x}_\alpha$，假定它们是统计独立的，总体的概率分布是析因分布 $\prod_{\mathbf{x}_\alpha\in\mathcal{T}} P(\mathbf{X}_\alpha=\mathbf{x}_\alpha)$。为了写出对数似然函数 $L(\mathbf{w})$，对析因分布取对数且将 $\mathbf{w}$ 看作未知的参数向量。因此可以写成

$$L(\mathbf{w}) = \log\prod_{\mathbf{x}_\alpha\in\mathcal{T}} P(\mathbf{X}_\alpha = \mathbf{x}_\alpha) = \sum_{\mathbf{x}_\alpha\in\mathcal{T}} \log P(\mathbf{X}_\alpha = \mathbf{x}_\alpha) \tag{11.45}$$

为了通过能量函数形成边缘概率 $P(\mathbf{X}_\alpha=\mathbf{x}_\alpha)$ 的表达式 $E(\mathbf{x})$，利用以下两点：

1. 由式 (11.40)，概率 $P(\mathbf{X}=\mathbf{x})$等于$\frac{1}{Z}\exp(-E(\mathbf{x})/T)$。

2. 由定义，状态向量 $\mathbf{x}$ 是属于可见神经元的状态 $\mathbf{x}_\alpha$ 和属于隐藏神经元的状态 $\mathbf{x}_\beta$ 的联立组合。因此可见神经元处于状态 $\mathbf{x}_\alpha$ 与任何 $\mathbf{x}_\beta$ 的概率为：

$$P(\mathbf{X}_\alpha = \mathbf{x}_\alpha) = \frac{1}{Z}\sum_{\mathbf{x}_\beta} \exp\left(-\frac{E(\mathbf{x})}{T}\right) \tag{11.46}$$

其中随机向量 $\mathbf{X}_\alpha$ 是 $\mathbf{X}$ 的子集，剖分函数 Z 定义为：

$$Z = \sum_{\mathbf{x}} \exp\left(-\frac{E(\mathbf{x})}{T}\right) \tag{11.47}$$

因而将式(11.46)和式(11.47)代入式(11.45)，得出对数似然函数所期望的表达式：

$$L(\mathbf{w}) = \sum_{\mathbf{x}_\alpha \in \mathcal{T}} \left(\log \sum_{\mathbf{x}_\beta} \exp\left(-\frac{E(\mathbf{x})}{T}\right) - \log \sum_{\mathbf{x}} \exp\left(-\frac{E(\mathbf{x})}{T}\right)\right) \tag{11.48}$$

对 $\mathbf{w}$ 的依赖包含在能量函数 $E(\mathbf{x})$ 中，如式(11.39) 所示。

依据式(11.39)，求 $L(\mathbf{w})$ 对 w_{ji} 的微分，经过一些运算后我们得到下列结果（参看习题 11.9)：

$$\frac{\partial L(\mathbf{w})}{\partial w_{ji}} = \frac{1}{T}\sum_{\mathbf{x}_\alpha \in \mathcal{T}} \left(\sum_{\mathbf{x}_\beta} P(\mathbf{X}_\beta = \mathbf{x}_\beta \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha) x_j x_i - \sum_{\mathbf{x}} P(\mathbf{X} = \mathbf{x}) x_j x_i\right) \tag{11.49}$$

为了简单起见，我们引入两个定义：

1. $$\rho_{ji}^{+} = \langle x_j x_i \rangle^{+} = \sum_{\mathbf{x}_\alpha \in \mathcal{T}} \sum_{\mathbf{x}_\beta} P(\mathbf{X}_\beta = \mathbf{x}_\beta \mid \mathbf{X}_\alpha = \mathbf{x}_\alpha) x_j x_i \tag{11.50}$$

2. $$\rho_{ji}^{-} = \langle x_j x_i \rangle^{-} = \sum_{\mathbf{x}_\alpha \in \mathcal{T}} \sum_{\mathbf{x}} P(\mathbf{X} = \mathbf{x}) x_j x_i \tag{11.51}$$

从宽松意义上我们可以将第一项平均值 ρ_{ji}^{+} 看成点火率的平均，或神经元 i 和 j 的状态之间的相关性，此时网络在钳制下运行或者说处于正向阶段。类似地，第二项均值 ρ_{ji}^{-} 可看成神经元 i 和 j 的状态间的相关性，此时网络自由运行或者说是处于负向阶段。利用这些定义，可以简化式(11.49)如下：

$$\frac{\partial L(\mathbf{w})}{\partial w_{ji}} = \frac{1}{T}(\rho_{ji}^{+} - \rho_{ji}^{-}) \tag{11.52}$$

Boltzmann 机学习的目的是最大化对数似然函数 $L(\mathbf{w})$，我们可以利用梯度下降法达到这一点，写成

$$\Delta w_{ji} = \epsilon \frac{\partial L(\mathbf{w})}{\partial w_{ji}} = \eta(\rho_{ji}^{+} - \rho_{ji}^{-}) \tag{11.53}$$

其中 η 是学习率参数；它通过 ϵ 和运行温度 T 定义为

$$\eta = \frac{\epsilon}{T} \tag{11.54}$$

式(11.53)的梯度下降规则称为 Boltzmann 学习规则。这里所叙述的学习是集中完成的；即突触权值的改变是在整个训练样本集都给出的情况下进行的。

总结

式(11.53)描述的 Boltzmann 机学习规则的简易性归因于这样的事实，即在神经元的两种不同操作条件使用*局部可观测量*，这两个不同条件为：一部分钳制运行，另外的自由运行。规则另一个有趣的特征是神经元 i 和 j 之间的突触权值的调整规则是独立于神经元的可见与否的，不管它们可见或都不可见，这一点可能令人吃惊。Boltzmann 学习的所有这些有益的特征归功于 Hinton and Sejnowski(1983，1986) 的关键性见解，它们将 Boltzmann 机的抽象数学模型和神经元网络在以下两点上联系起来：

- 描述一个神经元的随机性的 Gibbs 分布。
- 定义 Gibbs 分布的基于统计物理学的能量函数式(11.39)。

但是从实际观点看，典型地，我们发现 Boltzman 机中学习过程是很慢的，特别当机器中使用的隐藏神经元个数多的时候．这个令人不快的特征的原因是因为机器需要很长一段时间来

达到平衡分布，这通常在可见单元不被钳制的时候经常发生。

虽然如此，过去的这些年里，对随机机器的研究持续关注，这些关注分享古典 Boltzman 机对二进制向量学习概率分布的能力，但也能够实现以下两个功能：

1. 忽略 Boltzman 机负向学习，负向学习为时间的增加而负责。同时找到一些用于运用控制学习过程的其他方法。

2. 在密连接网络中的有效操作。

在下面的两节，我们介绍两个通过不同方式来解决这两个实际问题的方法。

11.8　logistic 信度网络

第一代 logistic 信度网络由 Neal 在 1992 年所发展，Boltzmann 机中对称连接被有向连接取代，从而形成无环图，这也使 Neal 的 logistic 信度网络称为*有向信度网络*（directed belief net）；今后这两个术语可替换地使用。特别地，一个 logistic 信度网络由多层结构组成，如图 11.7 所示。机器具有无环的性质使得概率计算简单。类似于 Boltzmann 机，网络利用式(11.43)的 logistic 函数计算一个神经元受到它自己的诱导局部域刺激时的条件概率。

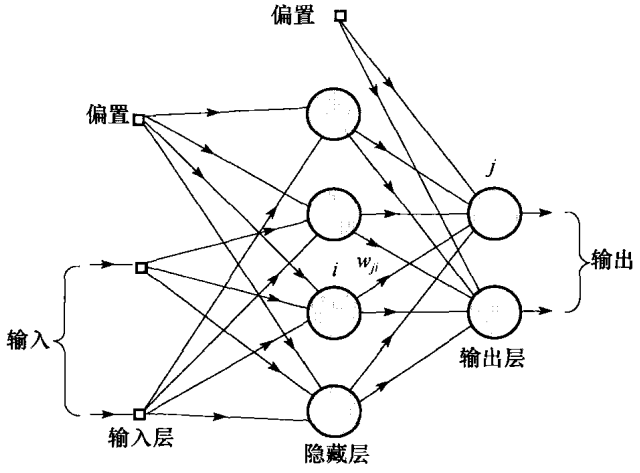


图 11.7　logistic 信度网络

令向量 $\mathbf{X}$ 由二值随机变量 $X_1, X_2, \cdots, X_N$ 组成，它定义由 N 个随机神经元构成的一个 logistic 信度网络。在 $\mathbf{X}$ 中的元素 X_j 的*双亲*（图 11.7 节点 j 的双亲）记为：

$$pa(X_j) \subseteq \{X_1, X_2, \cdots, X_{j-1}\} \tag{11.55}$$

也就是说，其中随机向量 $\mathbf{X}$ 最小的子集 $\{x_1, x_2, \cdots, x_j\}$，它的条件概率

$$P(X_j = x_j \mid X_1 = x_1, \cdots, X_{j-1} = x_{j-1}) = P(X_j = x_j \mid pa(X_j)) \tag{11.56}$$

参考图 11.7，例如，节点 i 是节点 j 的双亲节点，因为节点 i 到节点 j 是有向连接。logistic 信度网络的一个重要优点就是它能清楚揭示输入数据的固有概率模型的条件依赖性。特别地，第 j 个神经元被激发的概率由 logistic 函数定义，其中 w_{ji} 是从神经元 i 到神经元 j 的突触权值，条件概率仅依赖于 $pa(X_j)$ 的输入加权和。因此，式(11.56)提供信度在网络中传播的基础。

在两种非空（null）条件下进行计算 logistic 信度网络的条件概率：

1. $w_{ji}=0$，对所有不属于 $pa(X_j)$ 的 X_i，这一点由双亲的定义可得。

2. $w_{ji}=0$，对所有 $i \geqslant j$，这点由 logistic 信度网络是有向无环图这个事实可得。

正如 Boltzmann 机一样，我们导出 logistic 信度网络所期望的学习规则时仍然最大化对数似然函数，对于样本集合 $\mathcal{T}$ 最大化式(11.45)中对数似然函数式 $L(\mathbf{w})$。同时最大化通过定义如下突触权值 w_{ji} 的变化伴随着在概率空间中使用梯度下降算法：

$$\Delta w_{ji} = \eta \frac{\partial}{\partial w_{ji}} L(\mathbf{w})$$

其中 η 是学习率参数，而权值向量 $\mathbf{w}$ 表示整个网络。

但是，logistic 信度网络学习过程的一个严重缺陷是当它运用到密连接网络中的时候，隐藏神经元的后验概率的计算很棘手，除非在一些简单的应用中，例如带加性高斯噪声的线性模

型。和 Boltzmann 机一样，Gibbs 抽样同样可以用于近似后验概率，但是在 logistic 信度网络中使用 Gibbs 抽样被认为更加复杂。

11.9　深度信度网络

为了克服 logistic(directed) 信度网络中推理应用的困难的缺点，Hinton 等(2006) 发展了一种新的 logistic 信度网络，而这种网络中推理很容易完成。这个模型与 logistic 信度网络中一样，模型可以通过同样的方式学习得到，除了在最顶层的不同之外，它（以这种新方式）形成了无向联想记忆。事实上，正是这种特点使这种新的网络被称为深度信度网络。

在 Smolensky(1986) 中首先描述深度信度网络建立在一个神经网络结构上；同时这个结构被称为“小风琴 (harmonium)。”这种“小风琴”的特别之处在于在可见神经元和隐藏神经元之间没有连接；否则，它将和 Boltzmann 机一样在可见神经元和隐藏神经元之间使用对称连接。由于上述不同，这个“小风琴”也在 Hinton 等(2006) 中被命名为受限 Boltzmann 机 (restricted Boltzmann machine，RBM)。就第一眼所见，可能令人惊讶地发现：一个对称连接模型（如受限 Boltzmann 机）可以如同 logistic 信度网络一样学习一个有向产生模型。

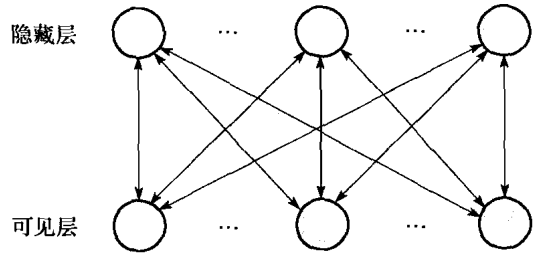


图 11.8　RBM 的神经结构。与图 11.5 比较，我们可以看到与 Boltzmann 机不同，在 RBM 中可见神经元之间和隐藏神经元之间没有连接

由于在 RBM 中隐藏神经元之间没有连接，也因为在可见神经元和隐藏神经元间的连接是无向的（详见图 11.8），则给定可见状态，隐藏神经元的状态相互之间是条件独立的。所以给定一个向量钳制在可见神经元之后，RBM 能够抽取后验分布中无偏见的样本。RBM 的这个特点使得其对相应的有向信度网络具有很大优势（Hinton，2007）。

一个感兴趣的地方就是如图 11.9 所示的权值固定的无限的 logistic 信度网络和图 11.8 所示的单 RBM 是等价的。

受限 Boltzmann 机中最大似然学习

由式(11.44)中的 logistics 函数来定义 RBM 隐藏神经元被激活的概率。令 $\mathbf{x}_a^{(0)}$ 表示一个数据向量被钳制在可见层零时刻的值。然后学习在下面两个操作之间来回交替进行。

- 给定可见状态，并行更新所有隐藏状态。
- 以相反方式做同样的事时：给定隐藏状态，并行更新所有可见状态。

令 $\mathbf{w}$ 是整个网络的权值向量。相应地，我们发现最大似然函数 $L(\mathbf{w})$ 对应的权值 w_{ji} 的梯度，w_{ji} 是连接可见单元 i 和隐藏单元 j 的对称权值，如下：

$$\frac{\partial L(\mathbf{w})}{\partial w_{ji}} = \rho_{ji}^{(0)} - \rho_{ji}^{(\infty)} \qquad (11.57)$$

其中 $\rho_{ji}^{(0)}$ 和 $\rho_{ji}^{(\infty)}$ 分别是神经元 i 和 j 在零时刻和无穷远时间的平均相关性（Hinton 等，2006；Hinton，2007）。除了不重要的术语变化，

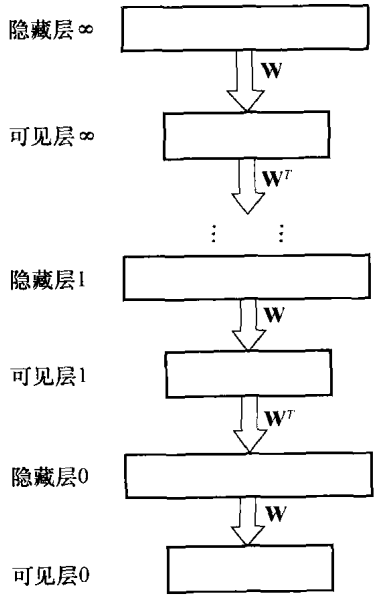


图 11.9　使用无限深度的 logistic 信度网络自顶向下学习

式(11.57)与式(11.52)中 Boltzmann 机的数学形式相同。但是因为我们不在 RBM 中做类比，式(11.57)没有使用温度作为参数。

深度信度网络的训练

深度信度网络的训练在逐层的基础上进行，如下（Hinton 等，2006；Hinton，2007）：

1. 受限 Boltzmann 机是直接在输入数据上训练的，所以使 RBM 的隐藏层随机神经元很有可能获得刻画输入数据的重要特征。所以我们称隐藏层为深度信度网络的第一隐藏层。

2. 经过训练的特征的激活然后被作为“输入数据”，它被用于第二个 RBM 的训练。事实上，刚描述的过程可以视为从特征中学习特征的过程之一。这个观点也许最早可以追溯到 Selfridge(1958) 的一篇早期的文章，它提出了一个称之为“pandeminium”模式识别系统。

3. 这个过程一直持续到深度信度网络中一些规定的个数的隐藏层得到训练。

这里需要注意的重要特性就是：每次一个新的特征层加入到深度信度网络中的时候，原始训练数据的对数概率的可变下界就得到改善（Hinton 等，2006）。

产生模型

如图 11.10 所示训练一个具有三个隐藏层的深度信度网络。向上的箭头指示了从特征中学习到的特征计算所得的权值。这些权值的功能是推理在深度信度网络中当一个数据钳制在可见神经元时隐藏层中的二进制特征值。

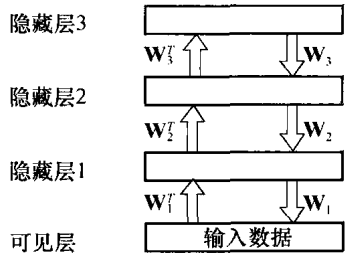


图 11.10　一个混合产生模型，其中最顶的二层是一个受限 Boltzmann 机，底下两层为有向模型。灰色箭头不属于产生模型；它们用来对给定的数据推理特征数据，但是它们不是用来产生数据的

产生模型是由图 11.10 中的无阴影的箭头标识。注意产生模型不包括由向上箭头代表的自底向上的连接；但更重要的，它确实包括在顶层 RBM(如层 2 和层 3) 的自底向上的连接，这些连接起着双边联想记忆的双重作用。当自底向上学习时，顶层 RBM 从隐藏层学习。当自上而下学习时，顶层 RBM 作为产生模型的起始器。

如图 11.10 所示，数据产生过程如下：

1. 通过使用如图 11.11 所示的方式多次交替的 Gibbs 取样后，可以从顶层 RBM 获得一个平衡样本，取样过程可以进行足够长的时间直到平衡。

2. 从可见顶层 RBM“可见”单元开始自顶向下的一次扫描用来随机挑取网络中所有另外隐藏神经层的状态。

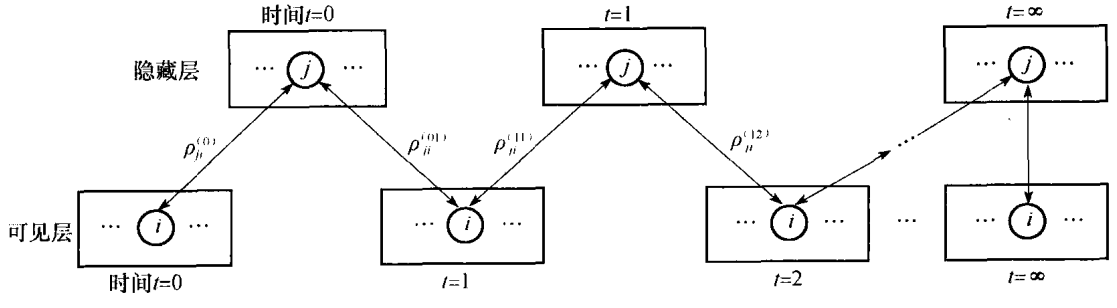


图 11.11　一个 RBM 中交替 Gibbs 取样过程的图例。在足够多次后，当前模型参数定义的静态分布抽取可见神经元向量和隐藏神经元向量

数据产生是很慢的，因为，首先所有顶层 RBM 必须达到平衡分布。幸运的是，产生不是供感知推理或者学习之用。

混合学习过程

深度信度网络中的每个 RMB 将模型化自身“可见”数据的任务分成两个子任务，如图 11.12(Hinton,2007) 所示：

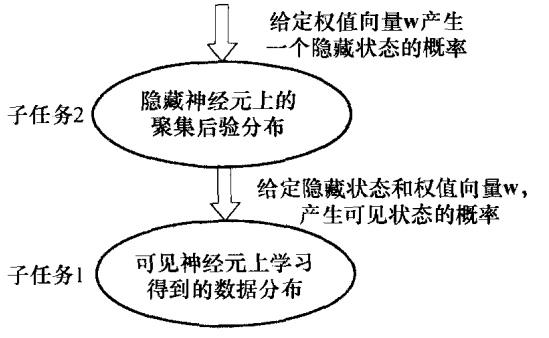


图 11.12　将感知数据模型化的任务分成 2 个子任务

- **子任务 1**　机器学习产生权值 **w**，把具体在隐藏神经元上的后验分布转化到在可见神经元上的对数据的近似分布。
- **子任务 2**　同样的权值集合，以 **w** 表示，同样定义了隐藏数据向量上的先验分布。对这个先验分布的采样需要使用大量的 Gibbs 取样（如图 11.11 所示）。但是这恰好是此复杂的先验概率的出现方式，它负责使 RBM 中的推理变得如此简单。在子任务 2 下，当下一个 RBM 学习之后，这个特殊的 RBM 用一个新的先验概率取代了复杂的先验概率（用 **w** 表示），新的先验概率更好地近似了低层 RBM 中的隐藏神经元的聚集的后验分布。

结束语

1. 除了最顶的两层，深度信度网络是一个多层的 logistic 信度网络，其中网络一层和向后另外一层有方向性连接。

2. 学习过程无监督地逐层自底而上。由于学习过程以这种方式进行，感知推理在深度信度网络中很容易：简单地说，推理过程包括一个自底而上的传递。

3. 深度信度网络为设计者提供很大自由空间。对设计者来说如何创造性地使用这个自由是个挑战。

11.10　确定性退火

现在进入本章最后一个论题——确定性退火。在 11.5 节我们讨论模拟退火，这个随机松弛技巧提供解决非凸优化问题的一个强有力方法。但是必须仔细选择退火进度表。特别地，只有当退火温度的下降率不比对数更快时，全局最小才能得到保证。这种要求使得在许多应用中用模拟退火变得不现实。模拟退火的运行是在能量曲面（地形）上进行随机移动。相反，*在确定性退火时*，随机性以某种形式结合到能量或代价函数中，因此在一系列下降温度情况下进行确定性最优化（Rose 等，1990；Rose，1998）。

下面我们在无监督学习任务（即聚类[11]）的背景下，叙述确定性退火的思想。

通过确定性退火聚类

在第 5 章讨论过聚类的思想。那里，聚类就是对于给定的数据分成子组，而每块尽量相同或者相似。聚类是典型的非凸优化问题，因为实际上用于聚类的畸变函数都是输入数据的非凸函数（第 10 章中描述的最优化流形表示的数据是个例外）。同时畸变函数关于输入的曲线充满局部最小，这使得求全局最小变得更为困难。

在 Rose（1991，1998）中通过*剖分的随机化或等价的编码规则的随机化*，对聚类描绘一个概率框架。这里利用的主要原则就是每个数据点*以概率归为一特定聚类*（子集）。具体地，令随机向量 **X** 表示*源*（输入）向量，令随机向量 **Y** 表示从感兴趣的码本的*最优重构*（输出）向量。这两个向量的单独实现分别记为 **x** 和 **y**。

对聚类我们需要一个*畸变度量*，由 $d(\mathbf{x},\mathbf{y})$表示。假定 $d(\mathbf{x},\mathbf{y})$满足两个希望的性质：

(1) 对任何 $\mathbf{x}$ 它是 $\mathbf{y}$ 的凸函数。

(2) 当变元 $\mathbf{x}$，$\mathbf{y}$ 有限时，它是有限的。

当上述两个温和的条件满足时，例如，在第 5 章和第 10 章使用的欧几里得平方畸变度

$$d(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\|^2 \tag{11.58}$$

满足这种适度的假定。对随机模式的期望畸变定义为

$$D = \sum_{\mathbf{x}}\sum_{\mathbf{y}} P(\mathbf{X}=\mathbf{x},\mathbf{Y}=\mathbf{y})d(\mathbf{x},\mathbf{y}) = \sum_{\mathbf{x}} P(\mathbf{X}=\mathbf{x})\sum_{\mathbf{y}} P(\mathbf{Y}=\mathbf{y}|\mathbf{X}=\mathbf{x})d(\mathbf{x},\mathbf{y}) \tag{11.59}$$

其中 $P(\mathbf{X}=\mathbf{x},\mathbf{Y}=\mathbf{y})$ 是 $\mathbf{X}=\mathbf{x}$ 和 $\mathbf{Y}=\mathbf{y}$ 联合事件的概率。在式(11.59)的第二个等式中，利用联合事件概率公式：

$$P(\mathbf{X}=\mathbf{x},\mathbf{Y}=\mathbf{y}) = P(\mathbf{Y}=\mathbf{y}|\mathbf{X}=\mathbf{x})P(\mathbf{X}=\mathbf{x}) \tag{11.60}$$

条件概率 $P(\mathbf{Y}=\mathbf{y}|\mathbf{X}=\mathbf{x})$ 指联想概率，即码字向量 $\mathbf{y}$ 联想源向量 $\mathbf{x}$ 的概率。

传统上通过对聚类模型的自由参数，即重建向量 $\mathbf{y}$ 和联想概率 $P(\mathbf{Y}=\mathbf{y}|\mathbf{X}=\mathbf{x})$，最小化期望畸变 D。这种形式的最小化产生“硬”聚类解，硬是指源向量 $\mathbf{x}$ 被归入最近的码向量 $\mathbf{y}$。另一方面，在确定性退火中，优化问题被改变成寻找服从特定随机水平概率分布，使得它最小化期望畸变。作为随机水平的一个主要度量，我们使用香农熵，定义为（参看 10.2 节）：

$$H(\mathbf{X},\mathbf{Y}) = -\sum_{\mathbf{x}}\sum_{\mathbf{y}} P(\mathbf{X}=\mathbf{x},\mathbf{Y}=\mathbf{y})\log P(\mathbf{X}=\mathbf{x},\mathbf{Y}=\mathbf{y}) \tag{11.61}$$

期望畸变的约束优化可以表示成拉格朗日函数

$$F = D - TH \tag{11.62}$$

的最小化，其中 T 为拉格朗日乘子。从式(11.62)我们观察到：

- 对大的 T 值，熵 H 被最大化。
- 对小的 T 值，期望畸变 D 被最小化，导致硬（非随机）聚类解。
- 对中间的 T 值，F 的最小值提供在熵 H 增加和期望畸变 D 减少之间的折中。

最重要的是，比较式(11.11)和式(11.62)，我们可以确认表 11.2 所列的约束聚类优化问题和统计力学之间的对应。根据这种类比，我们今后称 T 为温度。

表 11.2　约束聚类和统计物理学之间的对应

约束聚类优化	统计物理学
拉格朗日函数 F	自由能量 F
期望畸变 D	平均能量 $\langle E\rangle$
香农联合熵 H	熵 H
拉格朗日乘子 T	温度 T

为了进一步了解拉格朗日函数 F，根据式(10.26)，我们可以将联合熵 $H(\mathbf{X},\mathbf{Y})$ 分成如下两项：

$$H(\mathbf{X},\mathbf{Y}) = H(\mathbf{X}) + H(\mathbf{Y}|\mathbf{X})$$

其中 $H(\mathbf{X})$ 为信源熵，$H(\mathbf{Y}|\mathbf{X})$ 为在给定源向量 $\mathbf{X}$ 后重建向量 $\mathbf{Y}$ 的条件熵。信源熵 $H(\mathbf{X})$ 是独立于聚类的。因此，我们可以从拉格朗日函数 F 中去掉信源熵 $H(\mathbf{X})$，从而集中在条件熵

$$H(\mathbf{Y}|\mathbf{X}) = -\sum_{\mathbf{x}} P(\mathbf{X}=\mathbf{x})\sum_{\mathbf{y}} P(\mathbf{Y}=\mathbf{y}|\mathbf{X}=\mathbf{x})\log P(\mathbf{Y}=\mathbf{y}|\mathbf{X}=\mathbf{x}) \tag{11.63}$$

这样突出联想概率 $P(\mathbf{Y}=\mathbf{y}|\mathbf{X}=\mathbf{x})$ 的作用。因此，考虑到约束聚类优化问题和统计物理学之间的对应以及 11.2 节描述的最小自由能量原理，我们发现关于联想概率的拉格朗日函数 F 的最小化导致联想概率变为 Gibbs 分布

$$P(\mathbf{Y}=\mathbf{y}|\mathbf{X}=\mathbf{x}) = \frac{1}{Z_{\mathbf{x}}}\exp\left(-\frac{d(\mathbf{x},\mathbf{y})}{T}\right) \tag{11.64}$$

其中 $Z_{\mathbf{x}}$ 为当前问题的剖分函数，定义为：

$$Z_{\mathbf{x}} = \sum_{\mathbf{y}} \exp\left(-\frac{d(\mathbf{x},\mathbf{y})}{T}\right) \tag{11.65}$$

当温度 T 接近无穷时，我们从式(11.64)发现联想概率趋向于均匀分布。这就意味着当温度相当高时，每个输入向量是相等地联想起所有聚类。这种联想可以被视作“极度模糊”。在另一

个极端，当温度 T 趋于零时，联想概率趋近于 delta 函数。因此，当温度较低时，分类是硬的，每个输入样本以概率 1 分给最近的码向量。

为了寻找拉格朗日函数 F 的最小值，我们将式(11.64)的 Gibbs 分布代入式(11.59)和式(11.63)，然后将结果表达式应用到式(11.62)的拉格朗日算子 F 的公式中。这样导致的结果为（参看习题 11.16）：

$$F^* = \min_{P(\mathbf{Y}=\mathbf{y}|\mathbf{X}=\mathbf{x})} F = -T\sum_{\mathbf{x}} P(\mathbf{X}=\mathbf{x})\log Z_{\mathbf{x}} \tag{11.66}$$

对剩下的自由参数即码向量 $\mathbf{y}$，最小化拉格朗日函数，我们置 F^* 关于 $\mathbf{y}$ 的梯度为零。因此，得到条件

$$\sum_{\mathbf{x}} P(\mathbf{X}=\mathbf{x},\mathbf{Y}=\mathbf{y})\frac{\partial}{\partial \mathbf{y}} d(\mathbf{x},\mathbf{y}) = 0, \quad \text{对于所有的 } \mathbf{y}\in\mathcal{Y} \tag{11.67}$$

$\mathcal{Y}$为所有码向量的集合。利用式(11.60)的公式和对 $P(\mathbf{X}=\mathbf{x})$ 规整化，可以重新定义这个最小化条件为：

$$\frac{1}{N}\sum_{\mathbf{x}} P(\mathbf{Y}=\mathbf{y}|\mathbf{X}=\mathbf{x})\frac{\partial}{\partial \mathbf{y}} d(\mathbf{x},\mathbf{y}) = 0, \quad \text{对于所有的 } \mathbf{y}\in\mathcal{Y} \tag{11.68}$$

其中联想概率、$P(\mathbf{Y}=\mathbf{y}|\mathbf{X}=\mathbf{x})$ 由式(11.64)的 Gibbs 分布定义。在式(11.68)中仅为了完整性包括了比例因子 $1/N$，这里 N 为可用样本的数目。

我们现在可以描述聚类的确定性退火算法（Rose,1998）：

算法包括两个部分：开始在温度 T 为很高值时对码向量最小化拉格朗日函数 F^*，然后在降低温度 T 的同时跟踪最小值。

换句话说，确定性退火运行时具有特定的退火进度表，温度依次降低。对温度 T 的每个值，执行算法核心的两步迭代可描述如下：

1. 固定码向量，利用对于给定畸变度量 $d(\mathbf{x},\mathbf{y})$的式(11.64)的 Gibbs 分布计算联想概率。
2. 固定联想，使用式(11.68)对码向量 $\mathbf{y}$ 最优化畸变度量 $d(\mathbf{x},\mathbf{y})$。

这个两步迭代过程对 F^* 单调不升，因此能保证收敛到一个最小点。当温度 T 很高时，拉格朗日算子 F^* 相当光滑，而且在前面对畸变度量 $d(\mathbf{x},\mathbf{y})$的适度假设下，F^* 是 y 的凸函数。在温度较高时可以求得 F^* 的全局极小。随着温度降低，联想概率变“硬”，导致一个“硬”聚类解。

当温度 T 按退火进度表降低，系统经历一系列相变，相变由自然聚类分叉组成，在分叉处聚类模型规模（即聚类的数目）增加（Rose 等，1990;Rose,1991）。这种现象由于以下原因而富有意义：

1. 一系列相变提供控制聚类模型大小的一个有用工具。
2. 正如通常的物理退火一样，相变是确定性退火的关键点，此处需要小心进行退火。
3. 关键点是可计算的，因而提供用于在两个相变之间加速算法的信息。
4. 最优模型大小可以确认，通过耦合一个确认过程检验在不同相位得到的一系列解，这些解是表示模型规模（即聚类的数目）逐渐升高的解。

案例研究：混合高斯分布

图 11.13 和图 11.14 举例说明随温度 T 下降或温度倒数 $B=1/T$ 的上升，确定性退火在不同相位时聚类解的演化（Rose，1991）。产生这些图所使用的数据集由 6 个高斯分布混合而成，它们的中心在图 11.13 中都以“X”标识。计算所得聚类的中心都以“o”标识。由于聚类解在非零温度不是“硬”分类的，这个随机划分在图中由属于该聚类的等概率——如概率为 1/3 的围线所描绘。这个过程开始只有一个自然聚类（见图 11.13a）包括所有训练集。在第一次

相变，它分裂成两个聚类（见图 11.13b），然后经过一系列相变直到它达到 6 个聚类的自然集。当所有聚类都分裂时，下一个相变导致“爆炸”。图 11.14 表示相位图，显示随退火过程的进行平均畸变变量变化的情况，以及在每个相阶段，自然聚类的数目。在这个图中，平均畸变（相对它的最小值规整化）是对温度 T 的倒数即 B（相对于它的最小值规整化 $B_{\min}$）画出的。两个坐标轴都是以它们相关的对数形式标出的。

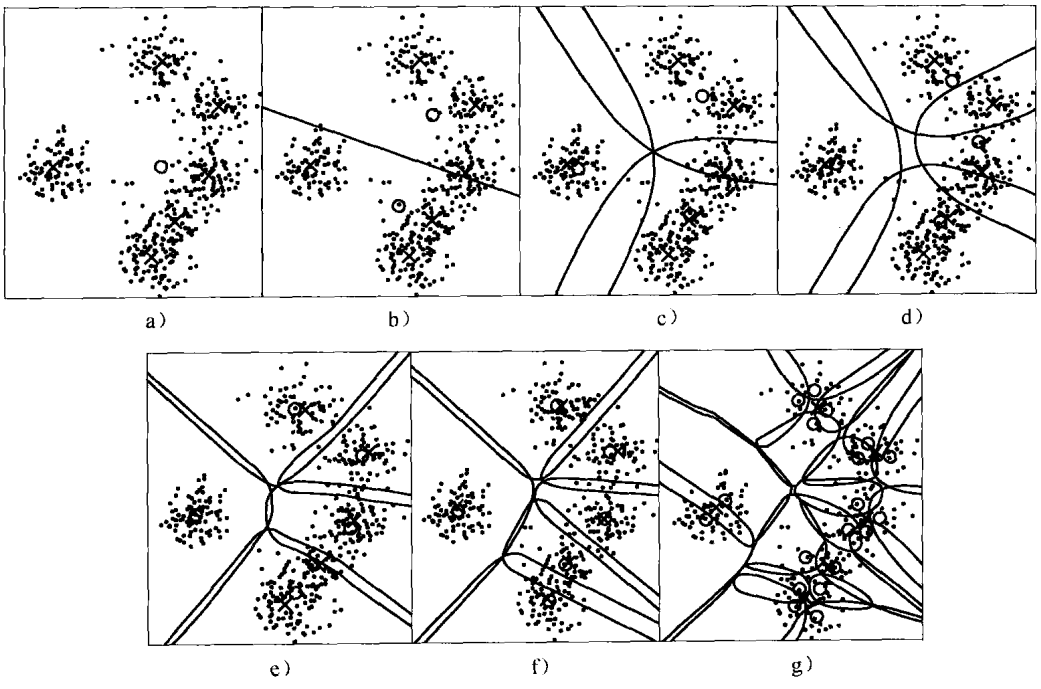


图 11.13　不同相位的聚类。画线是等概率围线，在 b）中 $p=1/2$，其余情况下 $p=1/3$。a）1 个聚类（$B=0$）；b）2 个聚类（$B=0.0049$）；c）3 个聚类（$B=0.0056$）；d）4 个聚类（$B=0.0100$）；e）5 个聚类（$B=0.0156$）；f）6 个聚类（$B=0.0347$）；g）19 个聚类（$B=0.0605$）

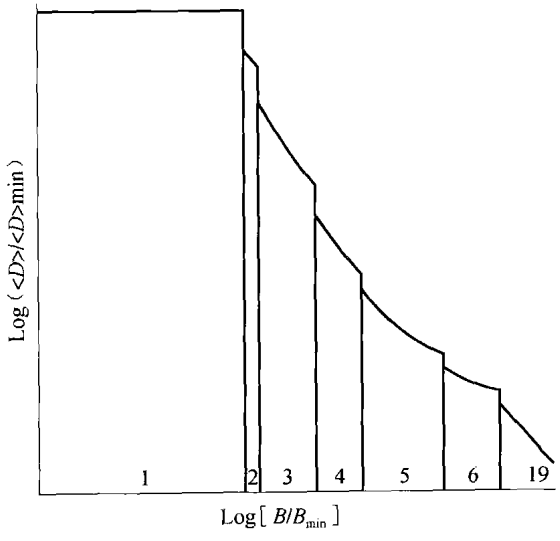


图 11.14　在确定退火中混合高斯分布样本的相位图。对每个相位显示有效聚类的数目

11.11　和 EM 算法的类比

为了说明确定性退火算法的另一个重要方面，假设我们将联想概率 $P(\mathbf{Y}=\mathbf{y}|\mathbf{X}=\mathbf{x})$看成一个二值随机变量 V_{xy}的期望值，其定义为

$$V_{xy}=\begin{cases}1, & \text{如果源向量 } \mathbf{x} \text{ 被分配到向量 } \mathbf{y}\\ 0, & \text{否则}\end{cases} \tag{11.69}$$

从这个观点出发，我们认识到确定性退火算法的两步迭代是期望最大（*EM*）算法的一种形式。为了领会这个关联，我们将先简单地描述 *EM* 算法的基本理论。

EM 算法[12]

让向量 $\mathbf{z}$ 代表缺失的或者未观察的数据。让 $\mathbf{r}$ 代表完整的数据向量，它由一些可观察的数据 d 和缺失的数据向量 $\mathbf{z}$ 组成。因而考虑两个数据空间$\mathcal{R}$和$\mathcal{D}$，他们具有从$\mathcal{R}$到$\mathcal{D}$的多对一映射。我们不能观察到完整数据向量 $\mathbf{r}$，相反实际仅能观察到$\mathcal{D}$中非完整的数据 $d=d(\mathbf{r})$。

令 $p_c(\mathbf{r}|\boldsymbol{\theta})$代表在给定参数向量 $\boldsymbol{\theta}$ 的情况下 $\mathbf{r}$ 的条件概率密度函数（pdf）。那么随机变量 D 在给定 $\boldsymbol{\theta}$ 的情况下的条件概率密度函数可以定义为

$$p_D(d|\boldsymbol{\theta})=\int_{\mathcal{R}(d)} p_c(\mathbf{r}|\boldsymbol{\theta})\mathrm{d}\mathbf{r} \tag{11.70}$$

其中$\mathcal{R}(d)$是由 $d=d(\mathbf{r})$决定的$\mathcal{R}$的子空间。EM 算法的直接目的在于找到 $\boldsymbol{\theta}$ 的一个值使得非完整数据的对数似然函数

$$L(\boldsymbol{\theta})=\log p_D(d|\boldsymbol{\theta})$$

取得最大。但是，这个问题的解决是通过间接地运用完整数据的对数似然函数

$$L_c(\boldsymbol{\theta})=\log p_c(\mathbf{r}|\boldsymbol{\theta}) \tag{11.71}$$

进行迭代来完成的，它是一个随机变量，因为缺失数据向量 $\mathbf{z}$ 是未知的。

更确切地说，让$\hat{\boldsymbol{\theta}}(n)$代表 EM 算法在迭代 n 时参数向量 $\boldsymbol{\theta}$ 的值。在这次迭代的 E 步，我们计算期望

$$Q(\boldsymbol{\theta},\hat{\boldsymbol{\theta}}(n))=\mathbb{E}[L_c(\boldsymbol{\theta})] \tag{11.72}$$

其中期望是对$\hat{\boldsymbol{\theta}}(n)$得到的。在同一的迭代的 M 步，在参数（权值）空间$\mathcal{W}$中对 $\boldsymbol{\theta}$ 最大化 $Q(\boldsymbol{\theta},\hat{\boldsymbol{\theta}}(n))$，这样找到更新参数估计值$\hat{\boldsymbol{\theta}}(n+1)$，表示为：

$$\hat{\boldsymbol{\theta}}(n+1)=\arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta},\hat{\boldsymbol{\theta}}(n)) \tag{11.73}$$

该算法开始时参数向量 $\boldsymbol{\theta}$ 的初始值为$\hat{\boldsymbol{\theta}}(0)$，然后根据式(11.72)和式(11.73)交替进行 E 步和 M 步，直到 $L(\hat{\boldsymbol{\theta}}(n+1))$和 $L(\hat{\boldsymbol{\theta}}(n))$之间的差下降至某一任意小值；此时，整个计算结束。

注意在 EM 算法的一次迭代后，非完整数据对数似然函数不是递减的，表示为：

$$L(\hat{\boldsymbol{\theta}}(n+1))\geqslant L(\hat{\boldsymbol{\theta}}(n)),\quad \text{当 } n=0,1,2,\cdots,$$

等号成立意味着我们处于对数似然函数的稳定点。

关于退火的讨论（续）

回到关于确定性退火和 EM 算法的类比中，我们可以得到两个相关程度很高的观察：

(i) 在确定性退火的第 1 步中计算联想概率，我们有与它等价 EM 算法中的求期望步骤。

(ii) 在确定性退火的第 2 步根据相应的码向量 $\mathbf{y}$ 来优化畸变变量 $d(\mathbf{x},\mathbf{y})$，我们有与它等价的 EM 算法中最大化步骤。

但在进行这种类比时，注意确定性退火比最大似然估计是更一般的。这是因为与最大似然估计不一样，确定性退火不对数据的固有概率分布做任何假定。事实上，联想概率是由最小化拉格朗日函数 F^* 导出的。

11.12　小结和讨论

在本章中我们讨论利用植根于统计力学的思想作为优化技术表示和机器学习的数学基础。

主要讨论了三种模拟算法：

1. Metropolis 算法，它是 Markov chain Monte Carlo（MCMC）针对未知概率分布上的模拟。

2. 模拟退火，它是一个动态的过程，就以下而言，要研究系统的总的特点在较高温度下观察到，同时系统的细节特征出现在较低温度。作为一种优化算法，模拟退火能够避免局部极小值。

3. Gibbs 抽样，它产生一个带 Gibbs 分布作为平衡分布的马尔可夫链。与 Metropolis 算法不同，与 Gibbs 抽样器相关的转移概率不是静态的。

本章主要介绍随机机器学习，主要关注两点：

1. 古典 Boltzmann 机，使用隐藏的和可见的随机二值状态的神经元，它巧妙地利用 Gibbs 分布的良好性质，从而具有一些吸引人的特征：

- 通过训练神经元所显示的概率分布和环境相匹配。
- 网络提供一种推广的方法，可用于搜索、表示和学习的基本问题。
- 如果退火进度表在学习过程中足够慢，则网络保证找到状态能量曲面的全局最小值。

遗憾的是 Boltzmann 机需要很长的时间才能达到平衡分布，以至没有实用价值。

2. 深度信度网络（DBN），它使用受限 Boltzmann 机（restricted Boltzmann machine，RBM）作为基本组成。RBM 一个突出的特点就是隐藏神经单元之间没有连接，否则与古典 Boltzmann 机一样使用可见神经单元和隐藏神经单元之间对称连接。DBN 同样建立在比较旧的思想上——从特征中学习：

- 机器在开始处理阶段，注重未加工的感官数据输入的特性，主要抓住输入数据之间有趣的不规则性。
- 同对待前一层作为“新”的未加工的感官数据输入从而学习另一层。
- 不断地这样学习，逐层之后直到最高层的特性复杂到能够很容易识别原始未加工的感官数据中的感兴趣的部分。

通过聪明地使用对产生模型自顶而下的学习和对推理自底而上的学习，DBN 获得以一个令人印象深刻的精度学习不带标签数字图像的密度模型的能力。

模拟退火的突出点在于在能量曲面上进行随机移动，从而使得退火进度表非常慢，这样使得在许多应用中无法实际使用。相反，确定性退火将随机性耦合到代价函数中，从一个较高温度开始，然后逐渐降低，在每个依次的温度对目标函数进行确定性的优化。但是，注意模拟退火保证到达全局极小，而确定性退火还没有找到这种保证。

注释和参考文献

1. 在式(11.3)中描述的术语“典型分布”是由 J. Willard Gibbs（1902）在《统计力学的基本原理》第一部分 33 页上创造的新名词，他写到：

> “所表示的分布……
>
> $$P = \exp\left(\frac{\psi - \varepsilon}{H}\right)$$
>
> 看来代表了最简单可以想象的情况，因为当系统包括分离能量的部分时，它的分布和分离部分的相位的分布律相同，其中 H 和 ψ 为常数，且 H 为正。分布的这个性质极大地简化了讨论，是和热力学极端重要关系的基础……

当一个整体系统在相位以刚才描述的方式分布，即当概率（P）指标是能量（ε）的线性函数，我们将说整体是典型分布的，称能量的除数 H 为分布的模。”

在物理文献中，式(11.3)通常称为典型分布（Reif，1965）或 Gibbs 分布（Landau and Lifschitz，1980）。在神经网络文献中称为 Gibbs 分布、Boltzmann 分布和 Boltzmann-Gibbs 分布。

2. Bernoulli 实验

考虑一个包含一系列独立同分布的过程的实验——一系列独立的实验。假定每个过程只有两种可能的结果。从而我们可以说这次一系列 Bernoulli 实验。例如，抛硬币始终，结果只有“头”和“尾”。

3. Metropolis-Hastings 算法

为了最优化离散状态空间于 1953 年引入了原始 Metropolis 算法。然后在 1970 年，Hastings 推广了此算法，是为了用于一些非对称转移概率的统计模拟。

$$\tau_{ji} \neq \tau_{ij}$$

相应地，转移概率定义为：

$$\alpha_{ij} = \min\left(1, \frac{\pi_j \tau_{ji}}{\pi_i \tau_{ij}}\right)$$

相应的马尔可夫链仍然满足细节平衡原理。通过这种方式推广得到的 Markov chain Monte Carlo 方法被称为 Metropolis-Hastings 算法（Robert and Casella，2004）。Metropolis 算法是 Metropolis-Hastings 算法中 $\tau_{ji} = \tau_{ij}$ 的特殊情况。

4. 在 Tu 等（2005）中，描述了一种根植于贝叶斯理论的用于图和它的候选部分的解析的算法。这种全息图像解析算法最优化了后验分布，从而产生如同在语音或者自然语言中经过一个句子一样的输出感兴趣部分的表示。

算法的计算模块集成两个流行的方法从而推理：

- 生成（自顶向下）方法，用来形成后验分布。
- 区分（自底向上）方法，使用依下列自底向上的过滤（测试）来计算区分概率。

在 Tu 等设计的算法中，通过生成方法为马尔可夫链来提供目标分布来定义后验概率，同时区分模型用来构造用于导出马尔可夫链的后验分布。换句话说，Markov chain Monte Carlo 方法是全息图像解析算法的核心。

5. 引入温度和模拟退火到组合优化问题的想法是由 Kirkpatrick、Gelatt and Vacchi（1983）和 Cerny（1985）独立提出的。

在物理环境中，退火是自然界的一个精细的过程。Kirkpatrick 等在 1983 的文章中讨论“熔化”一个固体的概念，这涉及升高温度到一个最大值使得固体的所有粒子处于液态时能够随机地运动。接着降低温度，使得所有粒子调整到具有低能基态的相应格点。如果冷却太快，也就是说，在每一温度，固体没有足够时间达到热平衡，这样得到的晶体会有许多缺陷，或物质将形成无晶体序的玻璃体并且仅为局部最优结构的亚稳态。

“熔化”这个概念对于思考玻璃体可能是正确的方法，或许对考虑组合优化问题的计算也有帮助。但是当讨论许多其他应用领域时会失误（Beckerman，1997）。例如，在图像处理中，如果我们升高温度使得所有粒子能够随机地调整自己的位置，就会丢失图像——变成均匀灰度。在相应的冶金学意义上，当退火铁或铜时，我们必须保证退火温度低于熔点；否则将会毁坏样本。

有几个控制冶金退火重要的参数：

- 退火温度，指示金属或合金加热到什么温度。
- 退火时间，指定保持提高温度后的时间长度。
- 退火进度表，指定温度下降的速度。

在描述退火进度表的小节中可以发现，这些参数在模拟退火里能找到和它们相对应的部分。

6. 对更复杂的和理论上的退火进度表，参看图书 Aarts and Korst（1989）和 van Laarhoven and Aarts（1988）。
7. Gibbs 抽样在统计物理中称为 Metropolis 算法的“热浴”形式。自从在 Geman and Geman（1984）及 Gelfand and Smith（1990）的文献中正式出现以后，它被广泛应用于图像处理、神经网络和统计学。后一篇文章还讨论抽样（或 Monte Carlo）的其他方法，这些方法基于对边缘概率估计的数值计算。
8. Boltzmann 机的可见神经元可以被分成输入和输出神经元。在第二种结构中 Boltzmann 机是在教师监督下进行联想，输入神经元从环境接受信息而输出神经元报告计算结果给最终用户。

9. 式(11.39)的表达式适合于 Boltzmann 机的“开”和“关”状态分别用+1 和−1 表示。如果机器利用 1 和 0 分别表示“开”和“关”状态，我们有

$$E(\mathbf{x}) = -\sum_{i}\sum_{j\neq i} w_{ji}x_i x_j$$

10. 传统上，相对熵或 Kullback-Leibler 散度用作 Boltzmann 机的性能指标（Ackley 等，1985；Hinton and Sejnowski，1986）。这个指标在第 10 章讨论过，我们同样展示了 Kullback-Leibler 散度的最小化等于最大化似然估计。
11. 确定性退火已成功应用到许多学习任务：
    - 向量量化（Rose 等，1992；Miller and Rose，1994）
    - 统计分类设计（Miller 等，1996）
12. Newcomb（1886）的文章考虑两个单变元高斯分布的混合参数估计，看起来这是文献报告中最早的一个 EM 类型过程的参考文献。

    “EM 算法”的名字由 Dempster、Laird 和 Rubin 在他们 1977 奠基性的文章中创造的。在那篇文章中第一次给出不同层次下不完整数据中计算最大似然估计的 EM 算法的公式。

    McLachlan and Krishnan（1997）以书的形式第一次统一考虑 *EM* 算法的理论、方法和应用它的历史以及推广。

习题

马尔可夫链

11.1　从状态 i 到状态 j 的 n 步转移概率记为 $p_{ij}^{(n)}$。利用归纳法证明：

$$p_{ij}^{(l+n)} = \sum_{k} p_{ik}\, p_{kj}^{(n)}$$

11.2　图 P11.2 表示随机行走过程的状态转移图，其中转移概率 p 大于零。图中所示的无限长马尔可夫链是不可约吗？说明你回答的理由。

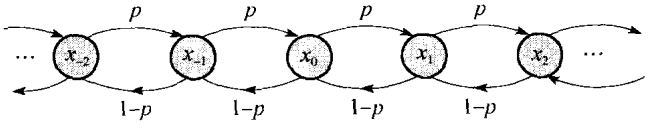


图　P11.2

11.3　考虑图 P11.3 所描绘马尔可夫链，它是可约的。找出包含在这个状态转换图中的各个状态类。

11.4　计算图 P11.4 所示的马尔可夫链的稳定态的概率。

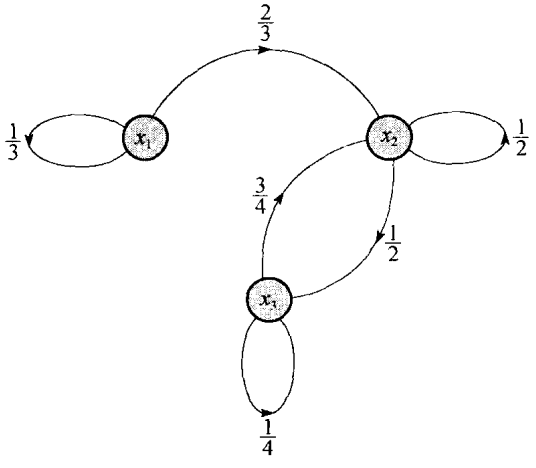


图　P11.3

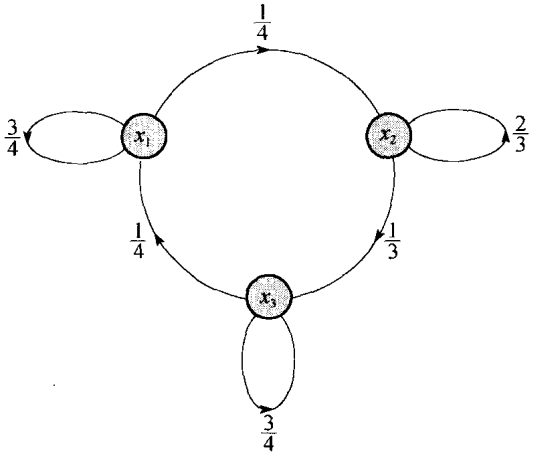


图　P11.4

11.5　考虑图 P11.5 所描绘马尔可夫链，使用这个例子证明 Chapman-Kolmongorov 的正确性。

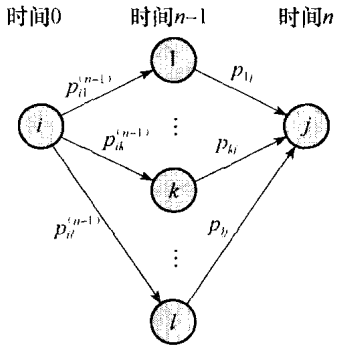


图　P11.5

模拟技术

11.6　Metropolis 算法和 Gibbs 抽样器代表两类不同的模拟大规模问题的技术。讨论它们之间的基本相似点和不同点。

11.7　本题中考虑用模拟退火求解旅行商问题（traveling salesman problem，TSP）。条件如下：

- N 个城市。
- 每两个城市间距离为 d。
- 旅行路线为一个闭合的路径，只访问每个城市一次。

目标是寻找具有最小总长度 L 的旅行路线（即排列城市访问的顺序）。在这个习题中，不同的可能旅行路线称为构形，而需最小化的代价函数为旅行路线的总长度。

(a) 设计出一种产生合法构形的迭代方法。

(b) 旅行路线总长度定义为

$$L_P = \sum_{i=1}^{N} d_{P(i)P(i+1)}$$

其中 P 表示一个置换且 $P(N+1)=P(1)$。因此，剖分函数为

$$Z = \sum_{P} e^{-L_P/T}$$

其中 T 为控制参数。建立用于 TSP 的模拟退火算法。

Boltzmann 机

11.8　考虑一个在温度 T 运行的随机二值神经元 j。它从状态 x_j 翻转到状态 $-x_j$ 的概率为

$$P(x_j \rightarrow -x_j) = \frac{1}{1+\exp(-\Delta E_j/T)}$$

其中 ΔE_j 为翻转所导致的能量改变。Boltzmann 机的总能量定义为

$$E = -\frac{1}{2}\sum_{i}\sum_{i\neq j}\sum_{j} w_{ji}x_i x_j$$

其中 w_{ji} 为从神经元 i 到神经元 j 的突触权值，且 $w_{ji}=w_{ij}$ 和 $w_{jj}=0$。

(a) 证明

$$\Delta E_j = -2x_j v_j$$

其中 v_j 为神经元 j 的诱导局部域。

(b) 因此，证明神经元 j 从初态 $x_j=-1$ 翻转到 $x_j=+1$ 的概率为 $1/(1+\exp(-2v_j/T))$。

(c) 证明当神经元 j 从初态为 $+1$ 翻转到状态 -1 时（b）中的公式仍然正确。

11.9　推导式(11.49)中对数似然函数 $L(\mathbf{w})$ 关于 Boltzmann 机突触权值 w_{ji} 的导数公式。

11.10　Gibbs 分布可以利用自完备的数学方法推导出，而不依赖于统计物理的概念。特别地，一个两步马尔可夫链模型的随机机器可用来导出形成 Boltzmann 机特殊性质的假设（Mazaika，1987）。这一点也不令人惊奇，因为作为 Boltzmann 机运行的模拟退火本身具有马尔可夫性质（van Laarhoven and Aarts，1988）。

考虑在一个随机机器中神经元的状态转移模型由两个随机过程组成：

- 第一个过程决定尝试哪个状态转移。

• 第二个过程决定这次转移是否成功。

(a) 表示状态转移概率 p_{ji} 为两个因子的乘积，即

$$p_{ji} = \tau_{ji} q_{ji}, \quad \text{当 } j \neq i$$

证明

$$p_{ii} = 1 - \sum_{j \neq i} \tau_{ji} q_{ji}$$

(b) 假设尝试率矩阵是对称的，

$$\tau_{ji} = \tau_{ij}$$

并且假设尝试成功的概率满足互补条件转移概率的性质，

$$q_{ji} = 1 - q_{ij}$$

使用这两个假设证明

$$\sum_j \tau_{ji} (q_{ij} \pi_j + q_{ij} \pi_i - \pi_j) = 0$$

(c) 假定 $\tau_{ji} \neq 0$，利用问题 (a) 中的结果证明：

$$q_{ij} = \frac{1}{1 + (\pi_i / \pi_j)}$$

(d) 最后，进行变量变换：

$$E_i = -T \log \pi_i + T^*$$

其中 T 和 T^* 为任意常数。由此推导，其中 $\Delta E = E_j - E_i$：

(i) $\pi_i = \frac{1}{Z} \exp\left(-\frac{E_i}{T}\right)$

(ii) $Z = \sum_j \exp\left(-\frac{E_j}{T}\right)$

(iii) $q_{ji} = \frac{1}{1 + \exp(-\Delta E / T)}$

(e) 你能从这些结果中得出什么结论？

11.11 在 11.7 节我们利用最大似然函数作为推导式(11.53)所描述的 Boltzmann 学习规则的准则。在这个习题中我们利用其他准则重新考虑这个学习规则。由第 10 章的讨论，两个概率 p_α^+ 和 p_α^- 的 Kullback-Leibler 散度定义为：

$$D_{p^+ \| p^-} = \sum_\alpha p_\alpha^+ \log\left(\frac{p_\alpha^+}{p_\alpha^-}\right)$$

其中对所有可能的状态 α 求和。概率 p_α^+ 表示网络在钳制（正向）状态时可见神经元处于状态 α 的概率，概率 p_α^- 表示网络在自由运行（负向）状态时可见神经元处于状态 α 的概率。利用 $D_{p^+ \| p^-}$ 的上述定义重新推导式(11.53)中的 Boltzmann 学习规则。

11.12 考虑 Boltzmann 机的可见神经元分成输入神经元和输出神经元。这些神经元的状态分别表示为 α 和 γ。隐藏神经元状态记为 β。这个机器的 Kullback-Leibler 散度定义为：

$$D_{p^+ \| p^-} = \sum_\alpha p_\alpha^+ \sum_\gamma p_{\gamma|\alpha}^+ \log\left(\frac{p_{\gamma|\alpha}^+}{p_{\gamma|\alpha}^-}\right)$$

其中 p_α^+ 为输入神经元在状态 α 的概率，$p_{\gamma|\alpha}^+$ 为给定输入状态 α 输出神经元被钳制在状态 γ 的条件概率，$p_{\gamma|\alpha}^-$ 为仅输入神经被钳制在状态 α 时处于热平衡中的输出神经元状态为 γ 的条件概率。和前面一样，加号和减号上标分别表示正向（钳制）和负向（自由运行）条件。

(a) 对输入、隐藏和输出神经元的 Boltzmann 机导出公式 $D_{p^+ \| p^-}$。

(b) 对于这种网络配置经过重新解释相关性 $p_{\gamma|\alpha}^+$ 和 $p_{\gamma|\alpha}^-$，证明调整突触权值 w_{ji} 的 Boltzmann 学习规则仍可以被表示成和式(11.53)同样的形式。

深度信度网络

11.13 在学习了深度信度网络和 logistic 信度网络，请问它们之间的主要区别在哪？并解释你的答案。

11.14 请说明如图 11.9 所示的无限的 logistic 信度网络和图 11.8 所示的单个 RBM 是等价的。

确定性退火

11.15 在 11.10 节中我们利用信息论方法讨论确定性退火的思想。确定性退火的思想也可以基于第 10 章讨论的最大熵原理用原理化的方式产生。说明第二种方法的基本原理（Rose，1998）。

11.16 (a) 利用式(11.59)，式(11.64)和式(11.63)，推导式(11.66)所给出的拉格朗日函数 F^* 的结果，该结果是用联想概率的 Gibbs 分布得到的。

(b) 利用本题中 (a) 的结果，导出式(11.68)给出的 F^* 关于码向量 $\mathbf{y}$ 取最小值的条件。

(c) 应用式(11.68)的最小化条件到式(11.58)的平方畸变度量，评论你的结果。

11.17 考虑数据集为混合高斯分布，在这种情况下，怎样才能使得利用确定性退火比利用最大似然估计有优越性？

11.18 在本题中我们探讨基于神经网络的模型分类中确定性退火的应用 (Miller 等，1996)。输出层的神经元 j 的输出记为 $F_j(\mathbf{x})$，其中 $\mathbf{x}$ 为输入向量。分类决策是基于最大判别式 $F_j(\mathbf{x})$。

(a) 对于概率目标函数，考虑

$$F = \frac{1}{N}\sum_{(\mathbf{x},\mathscr{C})\in\mathscr{T}}\sum_{j} P(\mathbf{x}\in\mathscr{R}_j)F_j(\mathbf{x})$$

其中$\mathscr{T}$为带标号向量的训练集，$\mathbf{x}$ 表示输入向量，$\mathscr{C}$为它的类别标识，$P(\mathbf{x}\in\mathscr{R}_j)$为输入向量 $\mathbf{x}$ 和类别区域$\mathscr{R}_i$ 的联想概率。利用第 10 章讨论的最大熵原理，写出 $P(\mathbf{x}\in\mathscr{R}_j)$的 Gibbs 分布。

(b) 令 $\langle P_e\rangle$ 表示错分类代价的均值。写出在联想概率 $P(\mathbf{x}\in\mathscr{R}_j)$的熵为一常值 H 的约束下最小化 $\langle P_e\rangle$的拉格朗日方程。

第12章
Neural Networks and Learning Machines, 3E

动 态 规 划

本章组织

本章有三个目的：（ⅰ）讨论动态规划作为多级动作规划的数学基础的发展，多级动作规划是通过一个智能体（agent）在随机环境中运行来实现的；（ⅱ）给出作为动态规划逼近形式的强化学习的直接推导；（ⅲ）给出处理维数灾逼近动态规划的非直接方法。

本章组织如下：

12.1 节是引言章节，通过 12.2 节中讨论的马尔可夫决策过程，激发了对动态规划的研究。

12.3 节到 12.5 节讨论动态规划的 Bellman 理论以及两个相关的方法：策略迭代和值迭代。

12.6 节讨论动态规划基于直接学习逼近后的理论基础，因而导致了时序差分学习和 Q-学习，它们将分别在 12.7 和 12.8 节中讨论。

12.9 节讲述处理维数灾问题的动态规划的非直接逼近的理论基础，因而导致最小二乘策略评估和逼近值迭代的讨论，这将在 12.10 节和 12.11 节中分别讨论。

最后是 12.12 节的小结和讨论。

12.1 引言

在本节中，我们认识到学习的两种主要范例：有教师学习和无教师学习。无教师学习的范例又可以细分为自组织（无监督）学习和强化（reinforcement）学习。第 1 章到第 6 章讨论了有教师学习或监督学习的不同形式，第 9 章到第 11 章讨论了非监督学习的不同形式。第 7 章中讨论了半监督学习。本章将讨论强化学习。

监督学习是在“教师”教导下进行的“认知”学习问题：它依赖于一组恰当输入-输出样本的可用性，这些样本能够反映运行环境。与此相反，强化学习是一种“行为”学习问题：通过学习系统和环境的交互作用完成任务，尽管存在不确定性，但学习系统仍然希望在环境中达到特定目标（Barto 等，1983；Sutton and Barto，1998）。无教师情况下进行的交互使得强化学习特别适合代价很高或很难（如果不是不可能）找到一组满意的输入-输出样本的动态情况。

有两种途径研究强化学习[1]，概述如下：

1. *传统方法*。通过惩罚和奖励的过程进行学习以期达到高度熟练行为的目标。

2. *现代方法*。它基于称为动态规划的一种数学方法，通过考虑将来可能的但实际并未发生的阶段而决定一系列的行动；这里强调的是规划（planning）。

我们讨论的重点是现代强化学习。

动态规划（dynamic programming）[2]技术处理的是这样一种情况：分阶段做决策，在做下一个决策之前在某种程度上能够预测每个决策的结果。这种情况的一个关键方面是不能孤立地做出决策。相反，现在对低代价的希望必须被将来高代价的失望所抵消。这是一个信任赋值（credit assignment）问题，因为信任或责任必须赋值给一组相互作用的决策中的每一个决策。为了最优的规划，需要在眼前代价和将来代价中取得有效的折中。这种折中确实被动态规划的形式抓住。特别地，动态规划解决下面的一个基本问题：

当可能需要牺牲短期性能的情况下，系统主体或决策者怎样在随机环境中学习而提高其长期性能的？

Bellman 动态规划为这一基础问题提供了一个好的原则方式的最优解。

在数学模型建立时的挑战在于在两个实体之间达到正确的平衡，一个是实际的，另一个是理论上的。这两个实体分别是：

- 给定问题的实际描述
- 作用于这一问题的分析和计算方法的能力

在动态规划中，特别关心的问题是在随机环境中运行的学习主体的决策。为了说明这一问题，我们围绕马尔可夫决策过程来建立模型。给定动态系统的初始状态，马尔可夫决策过程为选择决策序列提供数学基础，这将最大化从 N-阶段决策过程的返回值。我们刚刚讲述的是 Bellman 动态规划的本质。因而从马尔可夫决策过程的讨论来开始动态规划的学习是合适的。

12.2　马尔可夫决策过程

考虑一个学习系统或智能体（agent）或决策者（decision maker）以图 12.1 的方式和环境相互作用。系统依照一个有限的离散时间马尔可夫决策过程运行，这个马尔可夫决策过程有以下特性：

- 环境依概率以一组有限的离散状态来演化。但是注意状态并不包含过去的统计特性，尽管过去的统计特性对学习系统是有用的。
- 对于每一个环境状态，学习系统可以采取一组有限的可能行动。
- 每当学习系统采取一次行动，就会引起一定的代价。
- 观察状态、采取行动和引发代价都是在离散的时间里发生的。

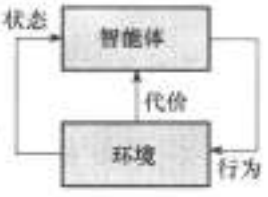


图 12.1　学习系统与环境交互的框图

在当前讨论的背景下，我们引入如下的定义：

环境的状态定义为学习系统从它和环境交互中获得的过去全部经历的总和，它包含学习系统预测环境未来行为所必需的信息。

设表示在时间步 n 的状态的随机变量为 X_n，在时间步 n 的实际状态为 i_n。有限个状态的集合用 X 表示。动态规划令人惊奇的一个特点是它的适用性很少依赖状态的性质。因此可以不对状态空间结构做任何假设而进行。还要注意的是动态规划算法的复杂度是对状态空间的维数二次的并对行为空间的维数是线性的。

例如，对于状态 i，一组可采取的行为（即学习系统作用于环境的输入）设为 $\mathcal{A}_i=\{a_{ik}\}$，这里的学习系统采取的行动 a_{ik} 的第二个下标 k 仅仅说明当环境在状态 i 时，可以有不止一个可能的行动。例如，采取行动 a_{ik} 将环境状态从 i 变化到 j 状态本质上为概率性的。然而，最重要的是，从状态 i 到状态 j 的转移概率完全依赖于当前状态 i 和相应的行动 a_{ik}。这就是第 11 章中讨论的马尔可夫性质。这个性质是很关键的，因为它意味着环境的当前状态为学习系统提供必需的信息以决定采取什么行动。

用一个随机变量 A_n 表示学习系统在时间步 n 时采取的行动，用 $p_{ij}(a)$ 表示在时间步 n 时由于采取行动 a 而导致从 i 状态转移到 j 状态的转移概率，其中 $A_n=a$。由状态动力学的马尔可夫假设有

$$p_{ij}(a)=P(X_{n+1}=j\mid X_n=i,A_n=a) \tag{12.1}$$

由概率论，转移概率 $p_{ij}(a)$ 必须满足以下两个条件：

$$1.\quad p_{ij}(a)\geqslant 0 \qquad \text{对于所有 } i \text{ 和 } j \tag{12.2}$$

$$2.\quad \sum_j p_{ij}(a)=1 \qquad \text{对于所有 } i \tag{12.3}$$

其中 i 和 j 属于状态空间。

对于给定数目的状态和转移概率，学习系统随时间采取行动产生的环境状态序列形成一个马尔可夫链。我们在第 11 章已经讨论过马尔可夫链。

当从一个状态转移到另一个状态时，学习系统招致一个代价。因此在行动 a_{ik} 作用下产生的从状态 i 到状态 j 的第 n 步转移，学习系统招致的代价表示为 $\gamma^n g(i,a_{ik},j)$，这里的 $g(\cdot,\cdot,\cdot)$ 是一个规定的函数，γ 是折扣因子 (discount factor)，$0\leqslant\gamma<1$。通过调节 γ，可以控制学习系统对它自己行动的短期和长期结果考虑的程度。在极端情况下，当 $\gamma=0$ 时系统是短视的 (myopic)，它只考虑它的行动的当前结果。以后将忽略这种极端值，也就是限于讨论 $0<\gamma<1$。当 γ 接近 1 时，未来的代价在采取最优行动时变得更为重要。

我们的兴趣在于形成一种策略 (policy)，这里策略指的是状态到行动的映射。换句话说：

给出环境当前状态的知识，一个策略是学习系统决定做什么所使用的一个规则。

策略表示为

$$\pi = \{\mu_0,\mu_1,\mu_2,\cdots\} \tag{12.4}$$

其中 μ_n 指的是在时间步 $n=0,1,2,\cdots$，状态 $X_n=i$ 到行动 $A_n=a$ 的映射。这个映射满足

$$\mu_n(i)\in\mathscr{A}_i \quad \text{对于所有状态 } i\in\mathscr{X}$$

这里 $\mathscr{A}_i$ 表示在状态 i 时学习系统能够采取的行动集合。这样的策略是允许的。

策略可以是不稳定的或稳定的。不稳定的 (nonstationary) 策略是随时间变化的，正如式 (12.4)所示。但当策略不随时间变化时，即

$$\pi = \{\mu,\mu,\mu,\cdots\}$$

就说策略是稳定的 (stationary)。换句话说，稳定的策略每次遇到一个特定的状态时采取相同的行动。对于稳定的策略，固有的马尔可夫链既可以是不平稳的也可以是平稳的。在不平稳的马尔可夫链上也可使用稳定的策略，但这是不太明智的。如果使用稳定的策略 μ，那么状态序列 $\{X_n,n=0,1,2,\cdots\}$ 形成一马尔可夫链，其转移概率为 $p_{ij}(\mu(i))$，$\mu(i)$表示一个行动。由于这个原因该过程称为马尔可夫决策过程。

基本问题

动态规划问题分为有限范围和无限范围两种。有限范围 (finite-horizon) 问题中在有限的阶段内对代价累积。无限范围 (infinite-horizon) 问题中在无限的阶段内对代价累积。无限范围问题为有限范围但数目非常大的问题提供一个合理的逼近。因为折扣保证对于任何策略所有状态的代价都是有限的，这样无限范围问题有着特殊的应用。

令 $g(X_n,\mu_n(X_n),X_{n+1})$记在策略 $\mu_n(X_n)$的行动下从状态 X_n 转移到 X_{n+1}的结果所发生的观测代价。在无限范围问题中，从初始状态 $X_0=i$ 开始并使用策略 $\pi=\{\mu_n\}$，总的期望代价定义为

$$J^{\pi}(i) = \mathbb{E}\left[\sum_{n=0}^{\infty}\gamma^n g(X_n,\mu_n(X_n),X_{n+1})\,|\,X_0=i\right] \tag{12.5}$$

其中期望值是对马尔可夫链 $\{X_1,X_2,\cdots\}$ 取值，γ 是折扣因子。函数 $J^{\pi}(i)$叫做策略 π 从状态 i 开始的 cost-to-go 函数。它的最优值记为 $J^*(i)$，定义为：

$$J^*(i) = \min_{\pi} J^{\pi}(i) \tag{12.6}$$

当且仅当 π 对 $J^*(i)$是贪婪的 (greedy) 时，策略 π 是最优的。这里术语“贪婪”被用来描述这样的情形：当智能体寻找最小化下一个瞬时代价时不注意这样的行动的话可能废除将来更好的途径。

当策略 π 稳定时，即 $\pi=\{\mu,\mu,\cdots\}$，我们用符号 $J^{\mu}(i)$代替 $J^*(i)$，并当下列条件成立时说 μ 是最佳的：

$$J^{\mu}(i) = J^{*}(i) \quad \text{对于所有初始状态 } i \tag{12.7}$$

动态规划的基本问题可以总结如下：

给定描述学习系统和环境相互作用的稳定马尔可夫决策过程，找到一个稳定的策略 $\pi=\{\mu,\mu,\mu,\cdots\}$ 使对所有的初始状态 i 有最小的 cost-to-go 函数 $J^{\mu}(i)$。

注意，在学习过程中，学习系统的行为可以随时间改变。但是学习系统寻找的最优策略是稳定的。

12.3　Bellman 最优准则

动态规划技术依赖归功于 Bellman（1957）的通称为最优原则（principle of optimality）的非常简单的思想。这个原则可简单陈述为（Bellman and Dreyfus，1962）：

一个最优策略有这样的性质，无论初始状态和初始决策是什么，对于第一个决策所导致的状态，剩余决策必须成为最优策略。

正如这里使用的那样，决策（decision）是在特定时间的一种控制选择，策略（policy）是整个控制序列或控制函数。

为用数学公式表示最优原则，考虑一个有限范围问题，它的 cost-to-go 函数定义为

$$J_0(X_0) = \mathbb{E}\left[g_K(X_K) + \sum_{n=0}^{K-1} g_n(X_n, \mu_n(X_n), X_{n+1})\right] \tag{12.8}$$

其中 K 是规划范围（planning horizon）（即阶段数目），$g_K(X_K)$ 是最终代价。给定 X_0，式(12.8)中的期望值是对剩余状态 $X_1,\cdots,X_{K-1}$ 求出的。现在可以正式陈述最优原则如下（Bertsekas，2005，2007）：

令 $\pi^{*}=\{\mu_0^{*},\mu_1^{*},\cdots,\mu_{k-1}^{*}\}$ 作为基本有限范围问题的最优策略。假设使用最优策略 π^{*} 时，给定的状态 X_n 发生的概率为正。考虑当环境在时刻 n 时状态为 X_n 的子问题，假设我们希望最小化对应的 cost-to-go 函数

$$J_n(X_n) = \mathbb{E}\left[g_K(X_K) + \sum_{k=n}^{K-1} g_k(X_k, \mu_k(X_k), X_{k+1})\right] \tag{12.9}$$

其中 $n=0,1,\cdots,K-1$。这时截断策略 $\{\mu_n^{*},\mu_{n+1}^{*},\cdots,\mu_{K-1}^{*}\}$ 对于子问题是最优的。

通过下面的讨论，我们可以直观地说明最优原则的合理性：如果截断策略 $\{\mu_n^{*},\mu_{n+1}^{*},\cdots,\mu_{K-1}^{*}\}$ 不是如陈述的那样为最优，那么一旦在 n 时刻到达 X_n 状态，通过简单转换到对于子问题最优的策略，我们可以减少 cost-to-go 函数 $J_n(X_n)$。

最优原则基于分而治之（divide and conquer）的工程概念。基本上，一个复杂的多阶段规划或控制问题的最优策略，可通过以下处理构造：

1. 构造一个仅包含系统最后一个阶段的“尾部子问题”（tail subproblem）的最优策略。
2. 扩展最优策略至包含系统最后两个阶段的“尾部子问题”。
3. 以这种方式继续这种过程，直到处理完整个问题。

动态规划算法

在前面描述过程的基础上，我们可以提出动态规划算法，它从时期 $N-1$ 到时期 0 反向处理。令 $\pi=\{\mu_0,\mu_1,\cdots,\mu_{K-1}\}$ 表示允许策略。对每一个 $n=0,1,\cdots,K-1$，令 $\pi^{n}=\{\mu_n,\mu_{n+1},\cdots,\mu_{K-1}\}$，令 $J_n^{*}(X_n)$ 表示从时间 n 的状态 X_n 开始到时间 K 的 $(K-n)$ 阶段问题的最优代价；即

$$J_n^*(X_n) = \min_{\pi^n} \mathop{\mathbb{E}}_{(X_{n+1},\cdots,X_{K-1})}\left[g_K(X_K) + \sum_{k=n}^{K-1} g_k(X_k,\mu_k(X_k),X_{k+1})\right] \tag{12.10}$$

它表示式(12.9)的最优形式。考虑到 $\pi^n=(\mu_n,\pi^{n+1})$和部分展开式(12.10)的右边和，我们可以写成：

$$\begin{aligned} J_n^*(X_n) &= \min_{(\mu_n,\pi^{n+1})} \mathop{\mathbb{E}}_{(X_{n+1},\cdots,X_{K-1})}\left[g_n(X_n,\mu_n(X_n),X_{n+1}) + g_K(X_K) + \sum_{k=n+1}^{K-1} g_k(X_k,\mu_k(X_k),X_{k+1})\right] \\ &= \min_{\mu_n} \mathop{\mathbb{E}}_{X_{n+1}}\left\{g_n(X_n,\mu_n(X_n),X_{n+1}) + \min_{\pi^{n+1}} \mathop{\mathbb{E}}_{(X_{n+1},\cdots,X_{K-1})}\left[g_K(X_K) + \sum_{k=n+1}^{K-1} g_k(X_k,\mu_k(X_k),X_{k+1})\right]\right\} \\ &= \min_{\mu_n} \mathop{\mathbb{E}}_{X_{n+1}}\left[g_n(X_n,\mu_n(X_n),X_{n+1}) + J_{n+1}^*(X_{n+1})\right] \end{aligned} \tag{12.11}$$

在最后一行，使用了式(12.10)的定义，以 $n+1$ 代替 n。相应地，从式(12.11)可以导出：

$$J_n(X_n) = \min_{\mu_n} \mathop{\mathbb{E}}_{X_{n+1}}\left[g_n(X_n,\mu_n(X_n),X_{n+1}) + J_{n+1}(X_{n+1})\right] \tag{12.12}$$

现在可以正式陈述动态规划算法如下（Bertsekas，2005，2007）：

对每一个初始状态 X_0，基本有限范围问题的最优代价 $J^*(X_0)$等于 $J_0(X_0)$，其中函数 J_0 从下面算法的最后一步得到：

$$J_n(X_n) = \min_{\mu_n} \mathop{\mathbb{E}}_{X_{n+1}}\left[g_n(X_n,\mu_n(X_n),X_{n+1}) + J_{n+1}(X_{n+1})\right] \tag{12.13}$$

按时间反向运行，且

$$J_K(X_K) = g_K(X_K) \tag{12.14}$$

另外，若 μ_n^* 使得式(12.13)的右边对于任意 n 和 X_n 为最小，那么策略 $\pi^*=\{\mu_0^*,\mu_1^*,\cdots,\mu_{K-1}^*\}$ 是最优的。

Bellman 最优性方程

以其基本形式，动态规划算法处理有限范围问题。我们感兴趣的是推广这个算法的用途，即处理在稳定策略 $\pi=\{\mu,\mu,\mu,\cdots\}$情况下，式(12.5)的 cost-to-go 函数所描述的无限范围折扣问题。为了达到这一点，我们做下面两件事：

1. 反转算法的时间索引。
2. 定义代价 $g_n(X_n,\mu(X_n),X_{n+1})$如下：

$$g_n(X_n,\mu(X_n),X_{n+1}) = \gamma^n g(X_n,\mu(X_n),X_{n+1}) \tag{12.15}$$

现在可以重新定义动态规划算法如下：

$$J_{n+1}(X_0) = \min_{\mu} \mathop{\mathbb{E}}_{X_1}\left[g(X_0,\mu(X_0),X_1) + \gamma J_n(X_1)\right] \tag{12.16}$$

它从初始条件

$$J_0(X) = 0, \quad \text{对于所有 } X$$

开始，状态 X_0 是初始状态，X_1 是策略 μ 的行动导致的新状态，γ 是折扣因子。

令 $J^*(i)$表示对初始状态 $X_0=i$ 的最优无限范围的代价。我们可以把 $J^*(i)$看作相应的 K 阶段最优代价 $J_K(i)$当 K 趋于无穷大时的极限，即

$$J^*(i) = \lim_{K\to\infty} J_K(i), \quad \text{对于所有 } i \tag{12.17}$$

这个关系联系着有限范围和无限范围之间的折扣问题。在式(12.16)中，置 $n+1=K$，$X_0=i$，并应用式(12.17)，我们得到

$$J^*(i) = \min_{\mu} \mathop{\mathbb{E}}_{X_1}\left[g(i,\mu(i),X_1) + \gamma J^*(X_1)\right] \tag{12.18}$$

为了重写最优无限范围代价 $J^*(i)$的公式，按下面两个阶段进行处理。

1. 计算代价 $g(i,\mu(i),X_1)$对 X_1 的期望值：

$$\mathrm{E}[g(i),\mu(i),X_1] = \sum_{j=1}^{N} p_{ij} g(i,\mu(i),j) \tag{12.19}$$

其中 N 是环境状态的数目，p_{ij} 是初始状态 $X_0=i$ 到新状态 $X_1=j$ 的转移概率。式(12.19)定义的量是在状态 $X_0=i$ 使用策略 μ 建议的行动引起的瞬时期望代价。利用 $c(i,\mu(i))$ 表示这个代价，可以写为：

$$c(i,\mu(i)) = \sum_{j=1}^{N} p_{ij} g(i,\mu(i),j) \tag{12.20}$$

2. 计算 $J^*(X_1)$ 对 X_1 的期望值。注意，如果知道有限状态系统的每一个状态 X_1 的代价 $J^*(X_1)$，则可以根据固有的马尔可夫链的转移概率决定 $J^*(X_1)$ 的期望值如下：

$$\mathrm{E}[J^*(X_1)] = \sum_{j=1}^{N} p_{ij} J^*(j) \tag{12.21}$$

这样，将式(12.19)至式(12.21)代入式(12.18)，得到期望的结果

$$J^*(i) = \min_{\mu}\left(c(i,\mu(i)) + \gamma \sum_{j=1}^{N} p_{ij}(\mu) J^*(j)\right), \quad \text{当 } i = 1,2,\cdots,N \tag{12.22}$$

式(12.22)叫做 Bellman 最优性方程。它不应该被看作算法。相反，它表示 N 个方程组，每个方程对应一个状态。这个方程组的解定义环境 N 个状态的最优 cost-to-go 函数。

有两种计算最优策略基本方法。它们称为策略迭代和值迭代。这两种方法分别在 12.4 节和 12.5 节讨论。

12.4　策略迭代

为了描述策略迭代算法，我们首先介绍 Watkins (1989) 提出的 Q-因子的概念。考虑一个现有的策略 μ，它的所有状态 i 的 cost-to-go 函数 $J^\mu(i)$ 为已知。对每一个状态 $i \in X$ 和行动 $a \in \mathcal{A}_i$，Q-因子定义为瞬时代价加上遵循策略 μ 的所有后继状态的折扣代价之和，表示为

$$Q^\mu(i,a) = c(i,a) + \gamma \sum_{j=1}^{n} p_{ij}(a) J^\mu(j) \tag{12.23}$$

其中活动 $a=\mu(i)$。注意 Q-因子 $Q^\mu(i,a)$ 比 cost-to-go 函数 $J^\mu(i)$ 包含的信息更多。例如，行动可以只依靠 Q-因子来排序，而依靠 cost-to-go 函数排序时还需要状态转移概率和代价的知识。还要注意的是在式(12.22)中的 $J^*(i)$ 是由 $\min_{\mu} Q^\mu(i,a)$ 获得的。

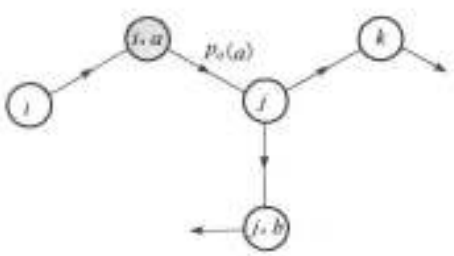


图 12.2　两个可能的转移：从状态 (i,a) 到状态 j 的转移为概率性的，但从状态 i 到状态 (i,a) 的转移为确定性的

通过设想由初始状态 $1,2,\cdots,N$ 和所有状态-行动对 (i,a) 组成其状态的新系统，如图 12.2 所描绘，我们可以深入了解 Q-因子的含义。有两种可能发生的不同概率：

1. 系统在状态 (i,a)，在这种状况下，不采取行动。以概率 $p_{ij}(a)$ 自动转变为状态 j；同时招致代价 $g(i,a,j)$。

2. 系统在状态 i，在这种状况下，采取行动 $a \in \mathcal{A}_i$ 后。下一个确定性状态是 (i,a)。

根据 12.2 节所说，我们说策略 μ 对 cost-to-go 函数 $J^\mu(i)$ 是贪心的，如果对所有的状态，$\mu(i)$ 是满足下列条件的活动：

$$Q^\mu(i,\mu(i)) = \min_{a \in \mathcal{A}_i} Q^\mu(i,a), \quad \text{对于所有 } i \tag{12.24}$$

对式(12.24)的下列两点观察是值得注意的：

1. 对于某一状态，可能存在一个以上的活动，能够最小化 Q 因子集合，在这种情况下，

对于有关的 cost-to-go 函数可以有多于一个的贪心策略。

2. 不同的 cost-to-go 函数可能有一个相同的贪心策略。

另外，下面的事实是所有动态规划方法的基础：

$$Q^{\mu^*}(i,\mu^*(i)) = \min_{a\in\mathcal{A}_i} Q^{\mu^*}(i,a) \tag{12.25}$$

这里 μ^* 是最优策略。

用我们处理 Q-因子和贪心策略的概念，可以描述策略迭代（policy iteration）算法。具体地讲，算法交替在下面两个步骤中运行：

1. 策略评估步骤，在这个步骤里，对所有状态和行动求当前策略的 cost-to-go 函数值和相应的 Q-因子的值。

2. 策略改进步骤，更新当前策略使其成为第一步计算出的 cost-to-go 函数的贪心策略

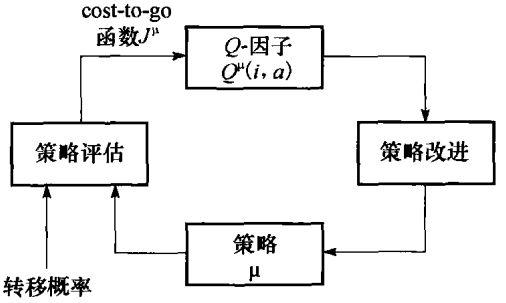


图 12.3　策略迭代算法框图

这两个步骤见图 12.3。具体地讲，我们从某一初始策略 μ_0 开始，然后产生一系列新策略 μ_1，μ_2，…。设当前策略为 μ_n，执行策略求值步骤时，计算 cost-to-go 函数 $J^{\mu_n}(i)$，作为下列线性方程组的解（参看式(12.22)）：

$$J^{\mu_n}(i) = c(i,\mu_n(i)) + \gamma\sum_{j=1}^{N} p_{ij}(\mu_n(i))J^{\mu_n}(j), \quad i = 1,2,\cdots,N \tag{12.26}$$

其中 $J^{\mu_n}(1), J^{\mu_n}(2),\cdots, J^{\mu_n}(N)$ 是未知数。使用这些结果，我们对状态-行动对 (i,a) 计算 Q-因子（参看式(12.23)）

$$Q^{\mu_n}(i,a) = c(i,a) + \gamma\sum_{j=1}^{N} p_{ij}(a)J^{\mu_n}(j), \quad a\in\mathcal{A}_i \text{ 和 } i = 1,2,\cdots,N \tag{12.27}$$

接着，通过计算如下定义的新策略 μ_{n+1} 来完成策略改进（参看式(12.24)）：

$$\mu_{n+1}(i) = \arg\min_{a\in\mathcal{A}_i} Q^{\mu_n}(i,a), \quad i = 1,2,\cdots,N \tag{12.28}$$

利用策略 μ_{n+1} 代替 μ_n，重复刚才描述的两个步骤直到有

$$J^{\mu_{n+1}}(i) = J^{\mu_n}(j), \quad \text{对于所有 } i$$

此时终止算法于策略 μ_n。由于 $J^{\mu_{n+1}} \leqslant J^{\mu_n}$，我们可以说经过有限次迭代后策略迭代算法会结束，因为固有的马尔可夫决策过程仅有有限数目的状态。表 12.1 概括了基于式(12.26)和式(12.28)的策略迭代算法。

表 12.1　策略迭代算法小结

1. 从任意的初始策略 μ_0 开始。
2. 对所有的状态 $i\in\mathcal{X}$ 和行动 $a\in\mathcal{A}_i$，当 $n=0,1,2,\cdots$，计算 $J^{\mu_n}(i)$ 和 $Q^{\mu_n}(i,a)$。
3. 对每一个状态 i，计算
$$\mu_{n+1}(i) = \arg\min_{a\in\mathcal{A}_i} Q^{\mu_n}(i,a)$$
4. 重复第 2，3 步，直到 μ_{n+1} 与 μ_n 无差别，那时的 μ_n 就是所求的策略。

在强化学习文献中，策略迭代算法被看成一种行动-评定结构（actor-critic architecture）(Barto 等，1983)。在这个背景下，策略改进被假设为行动的角色，因为它对应于学习主体行动的方式。根据同样的意义，策略评估被假设为评定的角色，因为它对应于评定主体所采取的行动的角色。

12.5　值迭代

在策略迭代算法中，算法每次迭代过程必须重新计算整个 cost-to-go 函数，这样代价是很高的。即使新策略和旧策略的 cost-to-go 函数很相似，这个计算也没有显著的改进。然而，有另外一种用于寻找最优策略的方法能够在计算 cost-to-go 函数时避免烦琐的重复计算。这个以

逐次逼近为基础的替代方法就是值迭代算法。

值迭代（value iteration）算法涉及对有限范围问题的每个求解序列，求解式(12.22)给出的 Bellman 最优性方程。当算法的迭代数目趋于无穷时，在极限处有限范围问题的 cost-to-go 函数对所有的状态一致收敛于相应的无限范围问题的 cost-to-go 函数（Ross，1983；Bertsekas，2007）。

令 $J_n(i)$表示在值迭代算法中迭代 n 时对状态 i 的 cost-to-go 函数。算法从任意的猜测 $J_0(i)$开始，$i=1,2,\cdots,N$。如果最优 cost-to-go 函数 $J^*(i)$的某一估计可用，那么它应该被用作初始值 $J_0(i)$。一旦选择了 $J_0(i)$，就可以计算 cost-to-go 函数序列 $J_1(i),J_2(i),\cdots$，使用值迭代算法：

$$J_{n+1}(i)=\min_{a\in\mathcal{A}_i}\left\{c(i,a)+\gamma\sum_{j=1}^{N}p_{ij}(a)J_n(j)\right\},\quad i=1,2,\cdots,N \tag{12.29}$$

对于状态 i 应用式(12.29)描述的 cost-to-go 函数的更新，这称为 i 的代价的支持（backing up of i's cost）。这个支持是 Bellman 最优性方程(12.22)的直接实现。注意对状态 $i=1,2,\cdots,N$，式(12.29)中 cost-to-go 函数的值在算法的每一次迭代时同时更新。这个实现方法表示值迭代算法传统的同步形式[3]。这样，从任意的初始值 $J_0(1),J_0(2),\cdots,J_0(N)$开始，当迭代数目 n 趋近无穷时，式(12.29)描述的算法将收敛于相应的最优值 $J^*(1),J^*(2),\cdots,J^*(N)$。换句话说，值迭代需要无限次迭代。

与策略迭代算法不同的是，在值迭代算法中不是直接计算最优策略，而是首先用式(12.29)计算最优值 $J^*(1),J^*(2),\cdots,J^*(N)$，然后获得关于该最优集合的贪心策略作为最优策略。就是说，

$$\mu^*(i)=\arg\min_{a\in\mathcal{A}_i}Q^*(i,a),\quad i=1,2,\cdots,N \tag{12.30}$$

这里

$$Q^*(i,a)=c(i,a)+\gamma\sum_{j=1}^{N}p_{ij}(a)J^*(j),\quad i=1,2,\cdots,N \tag{12.31}$$

表 12.2 给出了基于式(12.29)至式(12.31)的值迭代算法的小结，其中包括式(12.29)的停止准则。

表 12.2　值迭代算法小结

1. 从状态 $i=1,2,\cdots,N$ 的任意初始值 $J_0(i)$开始。
2. 对 $n=0,1,2,\cdots$，计算

$$J_{n+1}(i)=\min_{a\in\mathcal{A}_i}\left\{c(i,a)+\gamma\sum_{j=1}^{N}p_{ij}(a)J_n(j),\right\},\quad \begin{matrix}a\in\mathcal{A}_i\\ i=1,2,\cdots,N\end{matrix}$$

重复这种操作直到

$$\left|J_{n+1}(i)-J_n(i)\right|<\varepsilon,\quad \text{对每个状态 } i$$

这里的 ε 是指定的容许参数。假定 ε 足够小，使 $J_n(i)$充分接近最优 cost-to-go 函数 $J^*(i)$。因此我们可以置

$$J_n(i)=J^*(i)\quad \text{对所有状态 } i$$

3. 计算 Q-因子

$$Q^*(i,a)=c(i,a)+\gamma\sum_{j=1}^{N}p_{ij}(a)J^*(j),\quad \begin{matrix}\text{当 } a\in\mathcal{A}_i \text{ 且}\\ i=1,2,\cdots,N\end{matrix}$$

由此，确定贪心策略作为 $J^*(i)$的最优策略：

$$\mu^*(i)=\arg\min_{a\in\mathcal{A}_i}Q^*(i,a)$$

例 1　值迭代和策略迭代之间的关系

为了理解值迭代和策略迭代之间的关系，考虑图 12.4 所示的例子。图中 a 描写了在策略迭代中计算 Q-因子 $Q^*(i,a)$的候选操作，b 画出了值迭代中计算 Q-因子 $Q^*(i,a)$的相应的候选操

作。图中每一个无阴影的小圈表示一个状态，每一个有阴影的小圈表示一个状态-行动对。假设从状态 j 开始。学习系统可能取三个可能行动中的任意一个，环境能够响应与 6 个可能状态-行动对中的任意一个；(i,a) 是这样的一个状态-行动对，对其的变换代价记为 $g(i,j)$。

检查图 12.4，可以发现策略迭代和值迭代的后备操作是等价的，除了一个基本不同外：值迭代需要在所有可能状态-行动对上取的最大值，如图 12.4b 所示。 ■

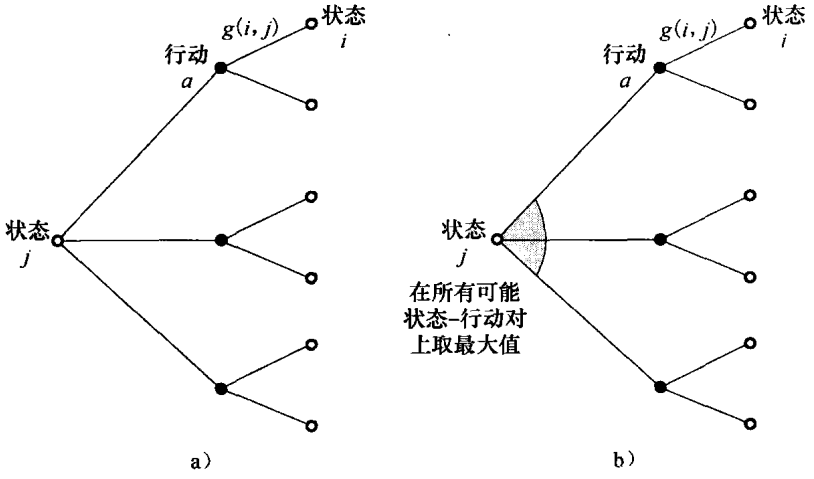


图 12.4　对 a) 策略迭代和 b) 值迭代图示候选方案

例 2　驿车问题

为了说明 Q-因子在动态规划中的作用，考虑驿车问题（stagecoach problem）。在 19 世纪中叶密苏里的一个探索者决定去西部加入在加利福尼亚的淘金潮（Hiller and Lieberman, 1995）。行程需要乘驿车穿过不安全的乡村，沿途会有强盗攻击的危险。行程的起始点（密苏里州）和终点（加利福尼亚州）是固定的。但是有很多可以选择的路径，有可能经过其他 8 个州，如图 12.5 所示。在图中，有以下规定：

- 一共 10 个州，每个州用一个字母表示。
- 行进的方向是从左到右。
- 从开始的状态 A(密苏里州）到终点的状态 J(加利福尼亚州）有 4 个阶段（即，驿车运行路径）。
- 探索者从一个状态到下一个状态行动是向上（Up）、直接向前（Straight）或向下（Down）的。
- 从 A 到 J 一共有 18 条可能路径。

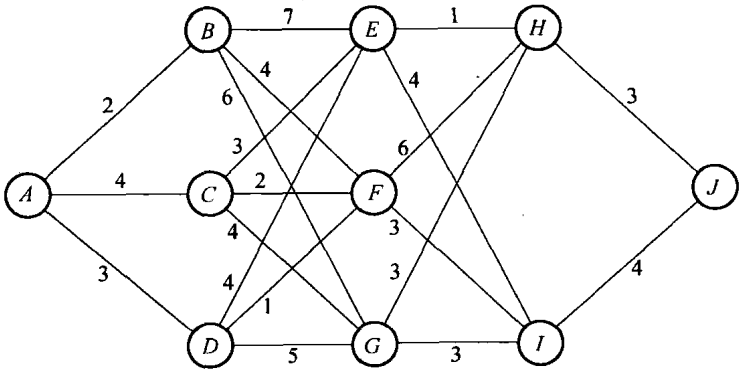


图 12.5　驿车问题的流向图

图 12.5 还包括对每一条路径的人身保险策略的代价，选择每一条路线是基于对该路线的安全代价的仔细评估。问题是从 A 到 J 找到一条人身保险最廉价的路线。

为了找到最优路线，我们从终点 J 开始向后推演，考虑一系列有限范围问题。这符合 12.3 节的 Bellman 最优性原则。

计算终点前的最后一阶段的 Q-因子，从图 12.6a 可以得出终点 Q-值如下：

$$Q(H,\text{down}) = 3$$
$$Q(I,\text{up}) = 4$$

在图 12.6a 中，这些数值分别表示在状态 H 和 I 上。

然后向后再移动一阶段，使用图 12.6a 得出的 Q-值，计算下面的 Q-值：

$$Q(E,\text{straight}) = 1+3=4$$
$$Q(E,\text{down}) = 4+4=8$$
$$Q(F,\text{up}) = 6+3=9$$
$$Q(F,\text{down}) = 3+4=7$$
$$Q(G,\text{up}) = 3+3=6$$
$$Q(G,\text{straight}) = 3+4=7$$

由于需要找到最小保险策略的路径，Q-值表明只有 $E\rightarrow H$，$F\rightarrow I$ 和 $G\rightarrow H$ 路径应保留，而其他路径应删除，如图 12.6b 所示。

再向后移动一阶段，对状态 B，C，D 重复这种 Q-因子计算，保留那些有最低安全评价的路径，就得到图 12.6c。

最后，向后移动到第一阶段，重复上面的计算，就得到图 12.6d。从图中我们看到共有 3 条最优路径如下：

$$A\rightarrow C\rightarrow E\rightarrow H\rightarrow J$$
$$A\rightarrow D\rightarrow E\rightarrow H\rightarrow J$$
$$A\rightarrow D\rightarrow F\rightarrow I\rightarrow J$$

它们产生的总体代价都是 11。也要注意的是通过 B 的所有 3 个最优路径中在前进中的所有 3 个可能选择中从 A 到 B 的瞬时代价是最小的。 ■

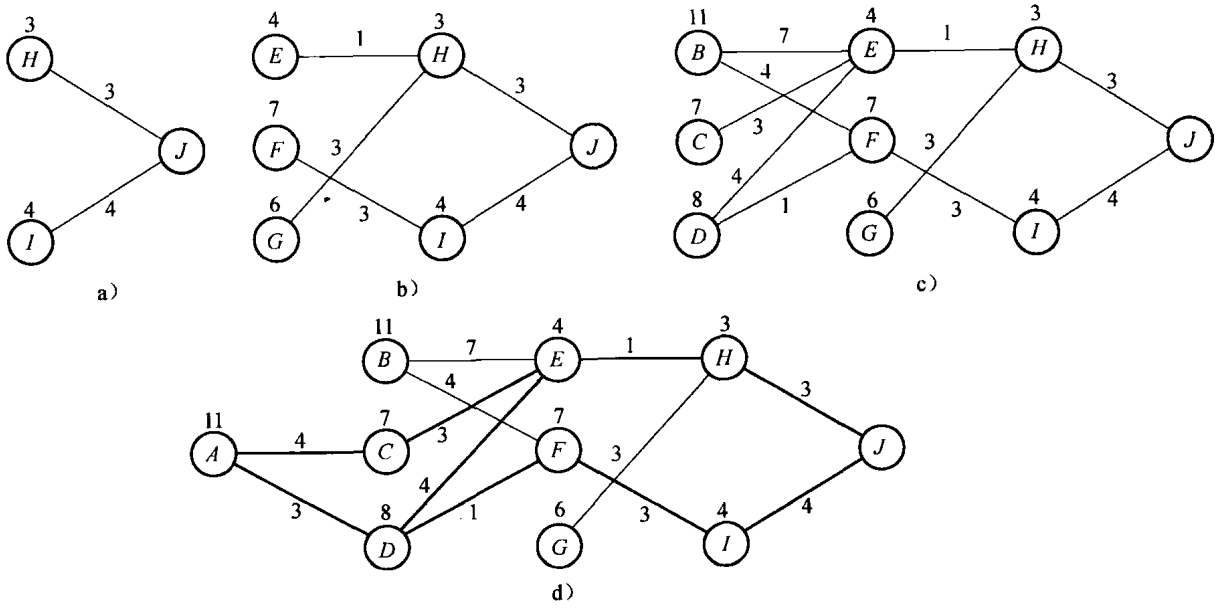


图 12.6　计算驿车问题 Q-因子涉及的步骤

12.6 逼近动态规划：直接法

Bellman 动态规划是完美的。然而，它假设从一个状态到另一个状态之间的转移概率的显式模型是可用的。不幸的是，在多个实际情形下，这样的模型是不能得到的。然而，有了构造良好的动态规划，其状态空间具有易控制的大小，我们可以利用 Monte Carlo 模拟来显式地估计转移概率及相应的转移代价；从其自身的特性，这样的估计是逼近的。我们称这一方法是直接的（direct）逼近动态规划，因为这里讲述的模拟的使用方便了动态规划方法的直接应用。

作为直观的例子，考虑一个多用户信道网络，关于它的感兴趣的问题是动态频道分配。假设分配给频道使用的模式的代价依赖于通过给定频道的通话间的距离。具体来说，在频道分享通话中彼此靠近的用户模式比彼此较远的频道分享通话模式更有利。换句话说，信道网络为在网络中规定方式下操作的用户的服务通话装备有发展良好的代价结构。有了这样的动态系统，就可以利用 Monte Carlo 模拟来允许动态规划直接应用于这样的网络（Nie and Haykin, 1998）。

基本上，动态规划的直接应用的合理性在于利用计算机模拟来产生多系统轨迹（multiple system trajectories），这导致对于每个状态值具有独立入口的查找表（look-up table）的构造；系统轨迹的数目越大，模拟结果将自然地更可信。特别地，每一次状态 i 被模拟系统的轨迹访问时独立变量 $J(i)$ 被保留在存储中。这样做时，我们已经用从状态 i 到状态 j 的概率转移和发生的瞬时转移代价 $g(i,j)$ 模拟了一个动态系统。

因此，该阶段为两个基本动态规划方法直接逼近：值迭代和策略迭代。特别地，

- 在值迭代的情形，我们得到时序差分学习；
- 在策略迭代的情形，我们得到 Q-学习。

这两个算法分别在第 12.7 节和 12.8 节中讨论，它们在强化学习中是广为人知的。我们因此将强化学习看成是动态规划的直接应用。

最后的评论：自然地，查找表的建立是有存储限制的。因而时序差分和 Q-学习的实际应用限制于状态空间是中等大小的状况。

12.7 时序差分学习

时序差分学习的思想最早见于 Sutton（1988）。我们通过考虑这一动态规划逼近形式的最简单版本（称为 TD(0)算法）来开始讨论，TD 是指时序差分（temporal difference）。

TD(0)学习算法

令 μ 为导致马尔可夫决策过程状态演化的策略。状态是通过序列 $\{i_n\}_{n=0}^{N}$ 来描述的；状态转移的最高数目是 N，终止状态 $i_N=0$。令 $g(i_n,i_{n+1})$ 为从状态 i_n 转移到状态 i_{n+1} 时发生的瞬时代价，其中索引 $n=0,1,\cdots,N-1$。然后根据 Bellman 方程，cost-to-go 函数定义为：

$$J^{\mu}(i_n)=\mathbb{E}[g(i_n,i_{n+1})+J^{\mu}(i_{n+1})],\quad n=0,1,\cdots,N-1 \tag{12.32}$$

其中，对每个 n，在所有可能发生的状态 i_{n+1} 上计算总体平均。从实际的角度看，我们需要的是一个迭代算法，它能够避免总体平均的需要。为此，可以调用在第 3 章中讨论过的 Robbins-Monro 随机逼近。

为了这一随机逼近的实质，考虑下面的关系

$$r^{+}=(1-\eta)r+\eta g(r,\bar{v})$$

其中 r 是旧值，η 是小的正的步长参数，它能够从一次迭代到下一次迭代发生改变，新的变量 $\bar{v}$ 是根据分布 $p_{V|R}(\bar{v}|r)$ 产生的随机变量；如前一章所述，在 r^{+} 中的上标加号表示“更新”。

因此，将 Robbins-Monro 随机逼近用于式(12.32)的 Bellman 方程，得到

$$\begin{aligned} J^{+}(i_n) &= (1-\eta)J(i_n) + \eta[g(i_n, i_{n+1}) + J(i_{n+1})] \\ &= J(i_n) + \eta[g(i_n, i_{n+1}) + J(i_{n+1}) - J(i_n)] \end{aligned} \tag{12.33}$$

其中左边的 $J^{+}(i_n)$是更新估计，在每一次状态 i_n 被访问时计算。为了简化问题，我们现在引入时序差分，定义为

$$d_n = g(i_n, i_{n+1}) + J(i_{n+1}) - J(i_n), \quad n = 0,1,\cdots,N-1 \tag{12.34}$$

这表示了两个量之间的差：

- 基于当前状态模拟结果的总体 cost-to-go 函数，即 $g(i_n, i_{n+1}) + J(i_{n+1})$；
- 当前估计 $J(i_n)$。

实际上，时序差分 d_n 为当前估计 $J(i_n)$是增长还是下降提供了信号。利用式(12.34)的定义，可以将式(12.33)的迭代算法重写为简单形式：

$$J^{+}(i_n) = J(i_n) + \eta d_n \tag{12.35}$$

其中 $J(i_n)$是当前估计，$J^{+}(i_n)$是*更新估计*，乘积项 $\eta_n d(n)$是作用于当前估计上为了产生更新项的修正（correction）。

式(12.35)的一步更新规则通常被称为 $TD(0)$算法；这一命名的原理在本节的后面部分将变得很明显。每一次状态 i_n 被访问时更新会发生，时序差分 d_n 也成为可用的。

Monte Carlo 模拟算法

式(12.35)描述了一个特别的迭代算法，由 Bellman 方程推导而得。从另一个观点和不同的算法上看，考虑如下的 cost-to-go 函数

$$J^{\mu}(i_n) = \mathbb{E}\left[\sum_{k=0}^{N-n-1} g(i_{n+k}, i_{n+k+1})\right], \quad n = 0,1,\cdots,N-1 \tag{12.36}$$

其中，这一次，期望算子是作用于属于整个状态转移序列的独立代价的。这里再一次将 Robbins-Monro 随机逼近作用到式(12.36)，得到（在整理了共同项后）：

$$J^{+}(i_n) = J(i_n) + \eta_k\left(\sum_{k=0}^{N-n-1} g(i_{n+k}, i_{n+k+1}) - J(i_n)\right) \tag{12.37}$$

其中 η_k 是随时间变化的步长（学习率）参数。这一更新公式可以表示成等价形式：

$$\begin{aligned} J^{+}(i_n) = J(i_n) + \eta_k[&g(i_n, i_{n+1}) + J(i_{n+1}) - J(i_n) + g(i_{n+1}, i_{n+2}) + J(i_{n+2}) - J(i_{n+1}) \\ &+ g(i_{N-2}, i_{N-1}) + J(i_{N-1}) - J(i_{N-2}) + g(i_{N-1}, i_N) + J(i_N) - J(i_{N-1})] \end{aligned}$$

其中，最后一行利用了终止状态 $i_N = 0$ 的性质，这相应地意味着代价 $J(i_N) = 0$。相应地，引用式(12.34)中引入的时序差分的定义，我们发现式(12.37)的迭代算法可假设为简化形式

$$J^{+}(i_n) = J(i_n) + \sum_{k=0}^{N-n-1} d_{n+k} \tag{12.38}$$

实际上，式(12.38)是轨迹 $\{i_n, i_{n+1}, \cdots, i_N\}$ 的 Monte Carlo 模拟的迭代执行，其中 $i_N = 0$——因此将这一方程称为 Monte Carlo 模拟算法。为了验证这一陈述，我们做两个假设：

1. 差分模拟系统轨迹是统计独立的。
2. 每一个轨迹是根据策略 μ 下的马尔可夫决策过程产生的。

继续这一证明过程，令 $c(i_n)$表示在模拟时间 n 遇到状态 i_n 时序列 $\{i_n, i_{n+1}, \cdots, i_N\}$ 发生的代价总和；即

$$c(i_n) = \sum_{k=0}^{N-n-1} g(i_{n+k}, i_{n+k+1}), \quad n = 0,1,\cdots,N-1 \tag{12.39}$$

然后，可以用

$$J(i_n) = \frac{1}{T}\sum_{n=1}^{T} c(i_n) \tag{12.40}$$

这是在访问了状态 i_n 一共 T 次模拟之后计算的。因此，总体平均 cost-to-go 函数的估计是

$$J^{\mu}(i_n) = \mathbb{E}[c(i_n)], \quad \text{对于所有 } n \tag{12.41}$$

直接可证式(12.40)的样本均值可以通过下面的迭代公式来计算

$$J^{+}(i_n) = J(i_n) + \eta_n(c(i_n) - J(i_n)) \tag{12.42}$$

从如下的初始条件开始

$$J(i_n) = 0$$

并设步长参数为

$$\eta_n = \frac{1}{n}, \quad n = 1,2,\cdots \tag{12.43}$$

我们发现式(12.42)是式(12.38)的迭代算法的简单重写，此时利用了为处理 Monte Carlo 模拟时序差分的观点而引入的新记号。

时序差分的联合观察：TD(λ)

在刚刚讨论过的时序差分学习中，我们推导了迭代算法的两个有限形式：

- 式(12.35)的迭代算法，从 Bellman 方程推导而得，说明从状态 i_n 到 i_{n+1} 的转移瞬时代价。
- 式(12.38)的迭代算法，根植于 Monte Carlo 模拟，说明在整个序列上状态转移招致的累计代价。

显然，在这两个迭代过程中必定存在一个中间范围，这值得考虑。为了得到这一中间范围，我们引入两个修正（Bertsekas and Tsitsiklis，1996）：

1. 扩展 Bellman 方程以考虑对某固定的 l 转移到第一个 $l+1$ 状态招致的独立代价：

$$J^{\mu}(i_n) = \mathbb{E}\left[\sum_{k=0}^{l} g(i_{n+k}, i_{n+k+1}) + J^{\mu}(i_{n+l+1})\right] \tag{12.44}$$

2. 没有先验知识用于促成相对于其他值来说某个希望的 l 值，我们通过在式(12.44)的右端乘以$(1-\lambda)\lambda^l$ 来形成在所有可能多步 Bellman 方程上的加权平均并且对某个固定的$\lambda<1$在 l 上求和：

$$J^{\mu}(i_n) = (1-\lambda)\,\mathbb{E}\left[\sum_{l=0}^{\infty}\lambda^l\left(\sum_{k=0}^{l} g(i_{n+k}, i_{n+k+1}) + J^{\mu}(i_{n+l+1})\right)\right]$$

由于我们正在处理线性方程，因此可以交换和的顺序：

$$J^{\mu}(i_n) = \mathbb{E}\left[(1-\lambda)\sum_{k=0}^{l} g(i_{n+k}, i_{n+k+1})\sum_{l=k}^{\infty}\lambda^l + (1-\lambda)\sum_{l=0}^{\infty}\lambda^l J^{\mu}(i_{n+l+1})\right] \tag{12.45}$$

现在采用下面两个公式的记号：

1.
$$(1-\lambda)\sum_{l=k}^{\infty}\lambda^l = \sum_{l=k}^{\infty}\lambda^l - \sum_{l=k}^{\infty}\lambda^{l+1}$$

2.
$$\begin{aligned}(1-\lambda)\sum_{l=0}^{\infty}\lambda^l J^{\mu}(i_{n+l+1}) &= \sum_{l=0}^{\infty}\lambda^l J^{\mu}(i_{n+l+1}) - \sum_{l=0}^{\infty}\lambda^{l+1} J^{\mu}(i_{n+l+1}) \\ &= \sum_{l=0}^{\infty}\lambda^l J^{\mu}(i_{n+l+1}) - \sum_{l=0}^{\infty}\lambda^l J^{\mu}(i_{n+l}) + J^{\mu}(i_n)\end{aligned}$$

相应地，可以重写式(12.45)为等价形式

$$J^{\mu}(i_n) = \mathbb{E}\left[\sum_{k=0}^{\infty}\lambda^k(g(i_{n+k}, i_{n+k+1}) + \lambda^k J^{\mu}(i_{n+k+1}) - \lambda^k J^{\mu}(i_{n+k}))\right] + J^{\mu}(i_n) \tag{12.46}$$

其中，为了表示的紧凑，我们对右端方括号中的三个项简单地利用了 k 来代替 l。现在可以通过式(12.34)引入的时序差分定义来简化问题了。为了这样做，我们再一次重写式(12.46)为下面的简单形式

$$J^{\mu}(i_n)=\mathbb{E}\left[\sum_{k=0}^{\infty}\lambda^k d_{n+k}\right]+J^{\mu}(i_n)$$

$$=\mathbb{E}\left[\sum_{k=n}^{\infty}\lambda^{k-n}d_k\right]+J^{\mu}(i_n),\quad \text{当 } n=0,1,\cdots,N-1 \tag{12.47}$$

认识到对某固定的值 λ，对依照 Bellman 方程的所有 k，我们有$\mathbb{E}[d_k]=0$，我们几乎不认为式(12.47)有什么奇怪。某种意义上，可以在点 1 和 2 下的修正将分析的网络结果求和，仅仅对于所有的 n 在等式$\mathbb{E}[J^{\mu}(i_n)]=\mathbb{E}[J^{\mu}(i_n)]$的右端加上期望$\mathbb{E}\left[\sum_{k=n}^{\infty}\lambda^{k-n}d_k\right]=0$。无论如何，这一结果都不会对我们继续应用 Robbins-Monro 随机逼近产生显著影响，正如下面要说明的那样。

具体地讲，将这一逼近应用到式(12.47)产生迭代算法：

$$J^{+}(i_n)=(1-\eta)J(i_n)+\eta\left(\sum_{k=n}^{\infty}\lambda^{k-n}d_k+J(i_n)\right)$$

在消去一些项后，简化为

$$J^{+}(i_n)=J(i_n)+\eta\sum_{k=n}^{\infty}\lambda^{k-n}d_k \tag{12.48}$$

式(12.48) 的迭代算法通常称为 TD(λ)；如前所述，TD 意味着“时序差分”。这一算法是 Sutton (1988) 首先提出的。值得注意的是为了推导这一算法，我们利用了 Bellman 动态规划、Monte Carlo 模拟、随机逼近的思想。

而且，TD(λ)包含了式(12.35)和式(12.38)的迭代算法作为两个特例：

1. 如果令 $\lambda=0$ 且利用规定 $0^0=1$，则式(12.48)衰减为：

$$J^{+}(i_n)=J(i_n)+\eta d_n$$

这是由利用动态规划方法推导的式(12.35)的重复。事实上，这是式(12.35)的算法被称为 TD(0)的原因，正如我们前面指出的那样。

2. 对另一个有限情形，如果令 $\lambda=1$，则式(12.48)衰减为

$$J^{+}(i_n)=J(i_n)+\eta\sum_{k=0}^{N-n-1}d_{n+k}$$

除伸缩因子 η 外，上式是利用 Monte Carlo 评估方法推导的式(12.38)的重复。注意对于 n 大于或等于规划范围 N 时时序差分 d_n 是 0.

作为小结，我们可以陈述如下：

式(12.48)讲述的 TD 方法是一种在线预测方法，它学习如何在部分基于其他估计时计算它们的估计。

换句话说，TD 方法是一种引导指令 (bootstrapping) 方法。更重要的是，它们不需要环境模型。

实际考虑

根据 Bertsekas and Tsitsiklis (1996)，对某状态 i_n，由 TD(λ)算法产生的估计值 $J(i_n)$收敛到策略 μ 的总体平均值 $J^{\mu}(\lambda)$，如果下面的两个条件得到满足：

1. 对所有的 n 状态 i_n 被轨迹频繁地访问无数次。
2. 步长参数 η 被允许在适当的速率下减少到 0。

在 Bertsekas and Tsitsiklis (1996) 对这一收敛性的证明中显示，在完成 TD(λ)算法的学习过程中，参数 λ 的改变没有理论上的障碍。那儿的理论考虑为选择合适的 λ 值建议了一个敏感策略，从接近于 1 的大的 λ 值开始 TD(λ)算法的执行 (即初始阶段促进总体平均 cost-to-go 函数的 Monte Carlo 估计)，然后允许 λ 衰减到 0 (即，向根据 Bellman 方程产生的估计移动)。

在广泛的意义上说，λ是某种形式的在时间过程上的退火。

12.8　Q-学习

前一节作为动态规划的随机逼近推导得到的 TD(λ)算法是无模型算法。这一节中，我们描述另一个随机算法，称为 Q-学习，它也不需要显式的关于环境的知识。Q-学习由 Watkins (1989) 首先推导出来的。Q-学习中的字母 Q 并不意味着什么特别的意义；它仅仅是 Watkins 在他最初推导这一算法时采用的记号。

为了激发 Q-学习的讨论，考虑图 12.1 中的强化学习系统。这一系统的行为目标，是在试验各种可能的行动序列和观察引起的代价以及发生的状态转移之后，如何寻找最优（即最小化代价）策略。用于产生行为的策略被称为*行为策略* (behavior policy)。这一策略是与估计策略值为目的的*估计策略* (estimation policy) 不同的。有了这两个彼此不同的策略，Q-学习被称为*用于控制的 off-policy 方法*。从这一分别中得到的好处是估计策略可以是贪心的，而行为策略用于样本化所有可能的行动。Off-policy 方法可以从 on-policy 方法中区别开来，其中策略的值被估计，同时该值被用于控制。

Q-学习算法

为了推导 Q-学习算法，令

$$s_n = (i_n, a_n, j_n, g_n) \tag{12.49}$$

一个四元组样本由下述项组成：在状态 i_n 上的一个试验行动 a_n，以代价

$$g_n = g(i_n, a_n, j_n) \tag{12.50}$$

对 $j_n = i_{n+1}$ 的状态转移。其中 n 表示离散时间。给定了这样的方案之后，我们现在给出如下的基本问题：

是否存在在线方法通过经验学习最优控制策略？经验是仅仅从观察样本的基础上获得的，样本的形式在式(12.49)和式(12.50)中定义。

对于这个基本问题的回答是肯定的，它能在 Q-学习中找到[4]。

Q-学习是一种增量式的动态规划过程，用一步一步的方式来决定最优策略。它非常适合于求解没有明显的转移概率知识的马尔可夫决策问题。但是，和 TD(λ)相似，成功应用 Q-学习的关键在于假设环境状态是完全可观察的，这就意味着环境是完全可观察的马尔可夫链。

回忆 12.4 节，状态-行动对 (i,a) 的 Q-因子 $Q(i,a)$是由式(12.23)定义的，而 Bellman 最优性方程由式(12.22)定义。联合这两个方程并且利用(12.20)给出的瞬时期望代价 $c(i,a)$的定义，我们得到

$$Q^*(i,a) = \sum_{j=1}^{N} p_{ij}(a)\Big(g(i,a,j) + \gamma \min_{b \in \mathcal{A}_j} Q^*(j,b)\Big), \quad \text{对于所有}(i,a) \tag{12.51}$$

这可看作 Bellman 最优性方程的两步形式。式(12.51)的线性方程组的解对所有状态-行动对 (i,a) 唯一地定义最优 Q-因子 $Q^*(i,a)$。

我们可以利用 12.4 节中基于 Q-因子构造的值迭代算法求解这个线性方程组。因此，对于算法的一步迭代我们有

$$Q^*(i,a) = \sum_{j=1}^{N} p_{ij}(a)\Big(g(i,a,j) + \gamma \min_{b \in \mathcal{A}_j} Q(j,b)\Big), \quad \text{对于所有}(i,a)$$

这个迭代的小步长的形式可描述为

$$Q^*(i,a) = (1-\eta)Q(i,a) + \eta \sum_{j=1}^{N} p_{ij}(a)\Big(g(i,a,j) + \gamma \min_{b \in \mathcal{A}_j} Q(j,b)\Big), \quad \text{对于所有}(i,a) \tag{12.52}$$

其中 η 为很小的学习率参数，位于区间 $0<\eta<1$ 内。

从它的形式上看，由式(12.52)描述的值迭代算法的一次迭代要求转移概率的知识。我们可以构造这一方程的随机方式，从而消除对这一先验知识的需求。具体地讲，在式(12.52)的一次迭代中对所有可能状态求平均被单个样本所替代，因而导出下列对 Q-因子的更新公式：

$$Q_{n+1}(i,a)=(1-\eta_n(i,a))Q_n(i,a)+\eta_n(i,a)[g(i,a,j)+\gamma J_n(j)] \quad \text{当}(i,a)=(i_n,a_n) \tag{12.53}$$

其中

$$J_n(j)=\min_{b\in\mathcal{A}} Q_n(j,b) \tag{12.54}$$

且 j 为后继状态，$\eta_n(i,a)$ 为在时间步 n 时状态-行动对 (i,a) 的学习率参数。更新公式(12.53)应用于当前状态-行动对 (i_n,a_n)，根据式(12.49)此时 $j=j_n$。对允许的其余状态-行动对，Q-因子仍保持不变，表示为

$$Q_{n+1}(i,a)=Q_n(i,a) \quad \text{对于所有的}(i,a)\neq(i_n,a_n) \tag{12.55}$$

式(12.53)至式(12.55)组成 Q-学习算法的一次迭代。

收敛定理[5]

假设学习率参数 $\eta_n(i,a)$ 满足条件：

$$\sum_{n=0}^{\infty}\eta_n(i,a)=\infty \quad \text{和} \quad \sum_{n=0}^{\infty}\eta_n^2(i,a)<\infty \quad \text{对于所有的}(i,a) \tag{12.56}$$

当迭代步数 n 趋于无穷大时，假定所有的状态-行动对被无限地经常访问，那么，对所有状态行动对 (i,a) 由 Q-学习算法产生的 Q-因子序列 $\{Q_n(i,a)\}$ 以概率 1 收敛于最优值 $Q^*(i,a)$。

一个保证算法收敛的时变学习率参数的样本为

$$\eta_n=\frac{\alpha}{\beta+n},\quad n=1,2,\cdots \tag{12.57}$$

其中 α 和 β 为正数。

小结和讨论

Q-学习可以看成两个等价方式中的一个：

作为 Robins-Monro 随机逼近算法或作为值迭代和 Monte Carlo 模拟的组合。

在算法的每一步迭代中它支持单个状态-行动对的 Q-因子。最重要的是，无需形成固有的马尔可夫决策过程的明显模型，算法的极限收敛到最优 Q-值。一旦最优 Q-值可用，利用式(12.30)以相当少的计算便可决定一个最优策略。

假设使用查找表来表示状态-行动对(i,a)的 Q-因子 $Q_n(i,a)$，Q-学习算法收敛到最优策略这种表示方法简单且计算效率高。但是它仅在构成联合输入空间的状态-行动对为中等规模时才能有效。

探测

在策略迭代中，状态空间的所有潜在的重要部分都应探测到。在 Q-学习中我们有一个附加要求：所有潜在有用的行动也都应被测试。特别地，对所有允许的状态-行动对应该经常探测足够的次数以满足收敛定理。对于记为 μ 的贪心策略，只有状态-行动对$(i,\mu(i))$被探测。遗憾的是并不能保证测试所有有用的行动，即使探测完所有状态空间亦是如此。

我们需要的策略是提供两个冲突目标之间的折中，以此扩展 Q-学习 (Thrun，1992)：

- 探测，它保证对所有允许的状态-行动对探测足够次数以满足 Q-学习收敛定理。

• 利用，它遵循贪心策略以寻求最小化 cost-to-go 函数。

达到这种折中的一种方法为遵循混合非稳定（mixed nonstationary）策略，这一策略在辅助马尔可夫过程和原始马尔可夫过程之间转换，原始马尔可夫过程，是由 Q-学习确定的稳定贪心策略控制的（Cybenko,1995）。辅助过程有下列解释：可能状态间的转移概率由原始控制过程的转移概率确定，原始过程具有附加成分，其对应的行动是一致随机性的。混合策略从辅助过程的任何状态开始，随之选择行动，然后切换到原始控制过程，以图 12.7 中的方式向前或向后进行。消耗在辅助过程上的操作时间占有固定数目的 L 步，定义为访问辅助过程所有状态的最长期望时间的 2 倍。消耗在原始控制过程的时间随每次切换逐步增加。令 n_k 表示从辅助过程到原始控制过程的切换时间，m_k 表示切换回辅助过程的时间，n_k 和 m_k 分别定义为

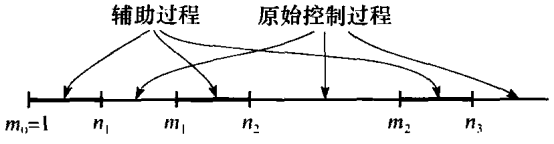


图 12.7　属于辅助过程和原始控制过程的时间段

$$n_k = m_{k-1} + L, \quad k = 1,2,\cdots \text{和} \ m_0 = 1 \tag{12.58}$$

和

$$m_k = n_k + kL, \quad k = 1,2,\cdots$$

构造辅助过程使得当 $k \to \infty$ 时，以概率 1 访问所有状态无穷次，因而保证收敛到最优 Q-因子。进一步，当 $k \to \infty$，混合策略在辅助过程上所消耗的操作时间渐进地为消耗在原始控制过程的操作时间的一小部分，这就意味着混合策略渐进收敛到一个贪心策略。因此，如果 Q-因子收敛到它们的最优值，贪心策略确实必定是最优的，只要策略变为贪心策略时足够地慢。

12.9　逼近动态规划：非直接法

典型地，大规模动态系统具有高维状态空间。所以，当我们处理这样的系统时，会遇到维数灾问题，这是指随着状态空间维数的增加计算复杂度呈指数增长。不幸的是，维数灾不仅在 Bellman 动态规划中出现，而且在其两个直接逼近形式中（时序差分学习和 Q-学习）也是棘手的。为了说明这一重要的实际问题，考虑包含 N 个可能状态和对每个状态有 M 个允许行动的动态规划问题；在这样的系统中，例如值迭代算法的每一次迭代对于稳定策略需要 N^2M 次运算。当 N 很大时运算所需的计算量使得它甚至不可能完成算法的一次迭代。

为了处理包含大量状态的困难的实世界问题，我们可以寻找逼近动态规划的某种逼近形式，这与 12.6 节讨论的直接法是不同的。特别地，不同于我们在 12.6 节所做的对转移概率和相联转移代价的显式估计，我们现在做如下事情：

利用 Monte Carlo 估计来生成一个或多个系统轨迹使其逼近一个给定策略的 cost-to-go 函数甚至最优的 cost-to-go 函数，然后在某种统计意义下最优化这个逼近。

我们将这一逼近动态规划方法称为非直接[6] 的，以便和 12.6 节讨论的直接方法区分开来。不管怎样，假设模拟动态系统的状态空间具有低于原始动态系统的维数。

因此，放弃了最优性的概念，我们可以通过下面简单的陈述来给出非直接法逼近动态规划的目标：

尽可能地做好，而不是更多。

事实上，性能最优性是计算易处理性的折中。这种策略正是人类大脑每天所做的：给定一个复杂的决策问题，大脑提供一个次优解，它从可靠性以及可用资源分配的角度上来说是“最好的”。

有了 Bellman 动态规划理论作为参考的框架，逼近动态规划的目标可以陈述如下：

对于状态 i 寻找逼近最优 cost-to-go 函数 $J^*(i)$ 的函数 $\tilde{J}(i,\mathbf{w})$，使得代价差 $J^*(i)-\tilde{J}(i,\mathbf{w})$ 根据某种统计准则最小化。

有了这两个目标，我们现在有两个基本问题：

问题 1：一开始如何选择逼近函数 $\tilde{\mathbf{J}}(i,\mathbf{w})$？

问题 2：已经选择了逼近函数 $\tilde{\mathbf{J}}(i,\mathbf{w})$ 以后，如何自适应权值向量 $\mathbf{w}$ 来为 Bellman 方程的最优性提供“最好匹配”？

为了回到问题 1，我们有线性和非线性逼近函数的选择，这反过来也决定了问题 2 的答案。下面首先考虑线性方法，然后讨论非线性方法。

逼近动态规划的线性方法

在这一方法中，通常的做法是将逼近函数 $\tilde{\mathbf{J}}(i,\mathbf{w})$ 表示为参数向量 $\mathbf{w}$ 的线性函数；即

$$\tilde{J}(i,\mathbf{w})=\sum_j \varphi_{ij}w_j=\boldsymbol{\phi}_i^T\mathbf{w}\quad \text{对于所有的 } i \tag{12.59}$$

其中 $\boldsymbol{\Phi}_i$ 是预编的基函数或特征，由逼近方案的设计者选择。式(12.59)的逼近在图 12.8 中说明。

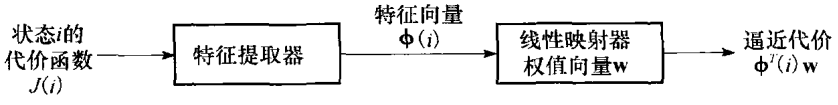


图 12.8　逼近动态规划线性方法的结构布局

逼近动态规划的线性方法有如下优点：

(ⅰ) 从数学上线性函数逼近器是容易形成和分析的；所以，逼近器的内在行为也是一样容易理解的。

(ⅱ) 通常，线性逼近器的数学形式提供了对实际操作中可能发生错误的观察，因而使其易于修正可能发生的错误。

(ⅲ) 在真实 cost-to-go 函数中的非线性性可以通过特别选择的基函数来逼近地获得，这些基函数能通过手头的动态规划问题的直觉来构造。

(ⅳ) 尤其是，线性规划是相对容易执行的。

对于点 (ⅲ)，必须注意的是好的基函数的选择可能在实际中是很困难的。

式(12.59)的选择为线性方法的问题 1 提供了答案。作为问题 2 的回答，通常被用于为 Bellman 方程的最优性提供最佳匹配的是最小二乘法，这在第 2 章讨论过。在第 12.10 节，我们将描述实现这一问题的一个途径[7]。

逼近动态规划的非线性方法

除了其上述吸引点外，逼近动态规划的线性方法被认为是一种实现更高目标的有用的踏脚石法 (stepping-stone)，通过下面考虑的一般情形来考虑：

认识到在实际中遇到的多个动态环境本质上是非线性的，逼近动态规划将不仅其自身是非线性的，也需要以“任意期望精确度”来逼近“任意”非线性动态环境。

换句话说，这里提倡的作为问题 1 的回答的非线性方法是一个通用逼近器 (universal approximator) 的逼近函数。

从前面的关于多层感知器和径向基函数 (RBF) 的讨论我们知道这些网络都是通用逼近

器。而且，第 15 章将要讨论的循环多层感知器也是通用逼近器。给定这样的网络的广泛选择，我们循环多层感知器为非线性逼近动态规划系统的最优设计提供了实际基础，这样说是基于如下两个重要理由：

1. 不同于具有单个非线性隐藏层和线性输出层的浅结构（shallow architecture）（以 RBF 网络为例），循环多层感知器可以被设计为具有两个或更多个隐藏层。通过一层馈给其他层，循环多层感知器具有“从特征学习特征”的性质，由是底层特征被累进地组合到更抽象和更高层的表达中。在 Bengio and LeCun（2007）中，提出深度结构（deep architecture）具有以非局部方式（即在中间邻居之外）泛化的潜力，这样的性质在应用于高度复杂任务的机器学习算法设计的进展中具有决定性意义。

2. 循环多层感知器具有内在多种方式的全局反馈（即包括两个或多个网络层）。这里，我们需要提醒自己大脑系统具有内在的丰富的全局反馈。特别地，在大脑中的不同区域几乎总是存在反馈连接，这些连接至少和前馈连接一样多（Churchland and Sejnowski，1992）。例如，从主视觉皮层回到外侧膝状核（LGN）的循环投影是从 LGN 到主视觉皮层的前向投影的大约 10 倍[8]。因此视觉系统具有强大能力是不奇怪的，而大脑的马达控制、听觉以及其他部分都是如此。鉴于我们所知道的关于大脑系统的知识，我们可以肯定全局反馈是计算智能的服务商——循环神经网络作为逼近动态规划系统模拟的候选神经网络具有实际重要性。

比较循环多层感知器和通常的多层感知器，我们发现在考虑结构深度的范围内它们共享第 1 点。然而，全局反馈的性质 2 使得循环多层感知器要好于通常的多层感知器，问题在于如何以最有效的方式构造网络的前馈和反馈连接。

现在我们已经回答了逼近动态规划非线性方法的问题 1，下面我们处理问题 2，是关于如何自适应逼近函数 $\tilde{J}(i,\mathbf{w})$ 中的权值向量 $\mathbf{w}$ 来为 Bellman 方程的最优性提供最优匹配。现在能够做如下陈述：

循环多层感知器的监督训练能通过利用无导数的非线性序列状态估计算法来最有效地完成。

通过采用这样的监督学习方法，我们不再需要考虑决策系统的非线性是如何发生的。因而，在这种情况下，将在 14 章讨论的无导数非线性序列状态估计算法，变得尤其重要。序列状态估计算法用于循环多层感知器（或者对这一问题的普通多层感知器）的监督训练将在第 15 章中讨论。

12.10 最小二乘策略评估

作为逼近动态规划的第一个非直接法，我们讨论一个称为最小二乘策略评估（least-squares policy evaluation）的算法，或者简记为 LSPE(λ)算法。在 LSPE(λ)中 λ 扮演着和 TD(λ)中的 λ 相似的角色。

LSPE(λ)背后的基本思想可以总结为：

在由一组基函数张成的低维子空间中完成值迭代。

具体来说，令 s 记表示状态 i 的特征向量 $\boldsymbol{\Phi}_i$ 的维数。我们可以定义 $N\times s$ 矩阵

$$\boldsymbol{\Phi}=\begin{bmatrix}\boldsymbol{\phi}_1^T\\ \boldsymbol{\phi}_2^T\\ \vdots\\ \boldsymbol{\phi}_N^T\end{bmatrix} \tag{12.60}$$

令$\mathcal{T}$记对于策略μ代价$\mathbf{J}$作为唯一固定点的映射，且令Π记由矩阵积$\mathbf{\Phi}_{\mathbf{w}}$定义的子空间上的投影（以合适的形式），其中$\mathbf{w}$是具有维数$s$的参数向量。以模拟作为 LSPE($\lambda$)算法的基础，我们可以给出如下的分析性描述（Bertsekas，2007）：

$$\mathbf{\Phi w}_{n+1} = \Pi \mathcal{T}(\mathbf{\Phi w}_n) + (\text{模拟噪声}) \tag{12.61}$$

算法的构成使得随着迭代数n趋于无穷加性模拟噪声收敛到 0。

背景和假设

考虑一个固定状态马尔可夫链，其状态记为$i=1,2,\cdots,N$，由稳定策略μ控制。我们可将式(12.5)重写成这样的形式：

$$J(i) = \mathbb{E}\left[\sum_{n=0}^{\infty} \gamma^n g(i_n, i_{n+1}) \mid i_0 = i\right]$$

其中i_n是时间n时的第i个状态，γ是折扣因子，$g(i_n, i_{n+1})$是从状态i_n到i_{n+1}的转移代价。带着线性结构的思想，代价$\mathbf{J}(i)$如下逼近：

$$J(i) \approx \tilde{J}(i, \mathbf{w}) = \mathbf{\phi}^T(i)\mathbf{w} \tag{12.62}$$

特征向量$\mathbf{\Phi}(i)$假设为s维的，则权值向量$\mathbf{w}$也必然具有相同的维数。感兴趣的问题是在如下子空间中逼近参数化代价$\tilde{J}(i, \mathbf{w})$：

$$\mathcal{S} = \{\mathbf{\Phi w} \mid \mathbf{w} \in \mathbb{R}^s\} \tag{12.63}$$

这一空间是由矩阵$\mathbf{\Phi}$的列张成的。注意矩阵积$\mathbf{\Phi w}$的维数等于可能的状态数N。

我们立刻做两个假设：

1. 马尔可夫链具有正的稳定状态概率；即

$$\lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} P(i_k = j \mid i_0 = i) = \pi_j > 0 \quad \text{对于所有的 } i \tag{12.64}$$

这一假设的意义是马尔可夫链具有单一循环类而没有瞬变状态。

2. 矩阵$\mathbf{\Phi}$的秩是s。

第二个假设的含义是特征矩阵$\mathbf{\Phi}$的列以及因此通过$\mathbf{\Phi w}$表示的基函数是线性独立的。

策略评估的投影值迭代

带着值迭代的思想，我们可以用式(12.20)和式(12.29)来写

$$\mathcal{T} J(i) = \sum_{j=1}^{N} p_{ij}(g(i,j) + \gamma J(i)), \quad i = 1,2,\cdots,N \tag{12.65}$$

其中$\mathcal{T}$记一个映射。现在，令

$$\mathbf{g} = \begin{bmatrix} \sum_j p_{1j} g(1,j) \\ \sum_j p_{2j} g(2,j) \\ \vdots \\ \sum_j p_{Nj} g(N,j) \end{bmatrix} \tag{12.66}$$

$$\mathbf{P} = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1N} \\ p_{21} & p_{22} & \cdots & p_{2N} \\ \vdots & \vdots & & \vdots \\ p_{N1} & p_{N2} & \cdots & p_{NN} \end{bmatrix} \tag{12.67}$$

以及

$$\mathbf{J}=\begin{bmatrix} J(1) \\ J(2) \\ \vdots \\ J(N) \end{bmatrix} \approx \mathbf{\Phi w} \tag{12.68}$$

其中利用了式(12.62)的逼近公式。我们就可以用向量 $\mathbf{g}$，$\mathbf{J}$ 和随机矩阵 $\mathbf{P}$ 重写式(12.65)为紧凑形式

$$\mathcal{T}\,\mathbf{J}=\mathbf{g}+\gamma\mathbf{P}\mathbf{J} \tag{12.69}$$

我们对值迭代的逼近形式感兴趣

$$\mathbf{J}_{n+1}=\mathcal{T}\,\mathbf{J}_n$$

它被限制在子空间$\mathcal{S}$中且包含了值迭代到$\mathcal{S}$的投影。具体地讲，根据式(12.68)，可以写

$$\mathbf{\Phi w}_{n+1}=\Pi\mathcal{T}(\mathbf{\Phi w}_n),\quad n=0,1,2,\cdots \tag{12.70}$$

其中，如前所述，Π记到子空间$\mathcal{S}$上的投影。式(12.70)被称为投影值迭代（PVI）方法，其本质可以陈述如下：

在迭代步 n，当前迭代 $\mathbf{\Phi w}_n$ 被施以映射$\mathcal{T}$且新的向量$\mathcal{T}(\mathbf{\Phi w}_n)$被投影到子空间$\mathcal{S}$上，从而产生更新的迭代 $\mathbf{\Phi w}_{n+1}$。

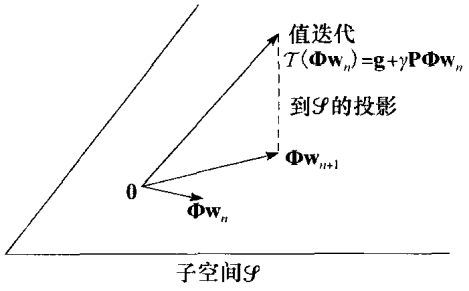


图 12.9　投影值迭代（PVI）方法的图示

图 12.9 图示了 PVI 方法。

PVI 方法可以被看成是解 Bellman 方程的值迭代方法的投影或者逼近形式。在 Bertsekas（2007）中描述了如下的发现：

1. 映射$\mathcal{T}$和$\Pi\mathcal{T}$是对加权欧几里得范数 $\|\cdot\|_\pi$ 的模数的收缩（contraction），其中 $\pi_1,\pi_2,\cdots,\pi_N$（表示马尔可夫链的稳定状态概率）扮演着定义欧几里得范数时的伸缩因子的角色。

2. 矩阵积 $\mathbf{\Phi w}^*$ 是权值向量 $\mathbf{w}^*$ 的映射$\Pi\mathcal{T}$的唯一固定点。（在当前讨论的背景中，当我们说固定点时，我们的意思是一个解，即向量 $\mathbf{w}^*$ 满足条件$\Pi\mathcal{T}\,\mathbf{w}^*=\mathbf{w}^*$。）

因此可以说 PVI 方法是逼近 Bellman 方程的分析方法。

然而，除了好的点之外，PVI 方法有两个严重的缺陷：

1. 如果 $\mathbf{\Phi w}$ 具有维数 N，变换向量$\mathcal{T}(\mathbf{\Phi w}_n)$是一个 N 维向量，因此，对于 N 很大时的大规模应用而言，方法的计算复杂度变得不可控制。

2. 向量$\mathcal{T}(\mathbf{\Phi w}_n)$到子空间$\mathcal{S}$的投影需要稳定状态概率 $\pi_1,\pi_2,\cdots,\pi_N$ 的知识。通常，这些概率是未知的。

幸运的是，这两个缺点可以通过利用 Monte Carlo 模拟法来减轻。

从投影值迭代到最小二乘策略评估

对投影Π利用最小二乘最小化，可以将式(12.70)表示为下面的形式：

$$\mathbf{w}_{n+1}=\arg\min_{\mathbf{w}}\|\mathbf{\Phi w}-\mathcal{T}(\mathbf{\Phi w}_n)\|_\pi^2 \tag{12.71}$$

等价地，可以将 PVI 算法的最小二乘方案表示为如下形式：

$$\mathbf{w}_{n+1}=\arg\min_{\mathbf{w}}\sum_{i=1}^{N}\pi_i\Big(\boldsymbol{\phi}^T(i)\mathbf{w}-\Big(\sum_{j=1}^{N}p_{ij}g(i,j)+\gamma\boldsymbol{\phi}^T(j)\mathbf{w}_n\Big)\Big) \tag{12.72}$$

为了从实际上完成式(12.72)的最优化，我们提出通过利用 Monte Carlo 模拟法来逼近，对状态 i 生成无限长的轨迹 $(i_0,i_1,i_2,\cdots)$，并根据下述公式在每次迭代 (i_n,i_{n+1}) 后更新权值向量 $\mathbf{w}_n$：

$$\mathbf{w}_{n+1} = \arg\min_{\mathbf{w}} \sum_{k=1}^{n} (\boldsymbol{\phi}^T(i_k)\mathbf{w} - g(i_k, i_{k+1}) - \gamma \boldsymbol{\phi}^T(i_{k+1})\mathbf{w}_n)^2 \tag{12.73}$$

由于明显的理由，这一递归称为最小二乘策略评估，或者简写为 LSPE。如图 12.10 所示，LSPE 可以看成是带有说明最小二乘逼近的加性模拟噪声的 PVI。

而且，由于联合映射 $\Pi\mathcal{T}$ 的收缩性质和模拟噪声的渐进减少特性，LSPE 收敛到 PVI 的相同极限，即，满足如下固定点方程的唯一权值向量 $\mathbf{w}^*$：

$$\boldsymbol{\Phi}\mathbf{w}^* = \Pi\mathcal{T}(\boldsymbol{\Phi}\mathbf{w}^*) \tag{12.74}$$

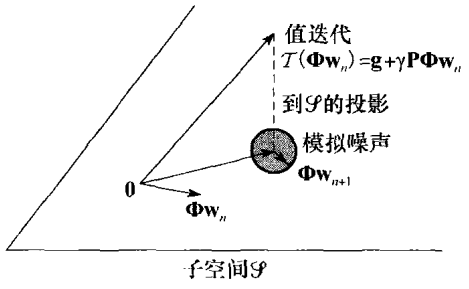


图 12.10　作为投影值迭代（PVI）随机方案的最小二乘策略评估（LSPE）的图示

LSPE(λ)

以与第 12.7 节中介绍 TD(λ)的相似的方式，我们引入时序差分（参看式(12.34)）：

$$d_n(i_k, i_{k+1}) = g(i_k, i_{k+1}) + \gamma \boldsymbol{\phi}^T(i_{k+1})\mathbf{w}_n - \boldsymbol{\phi}^T(i_k)\mathbf{w}_n \tag{12.75}$$

相应地，可以表示基于模拟的 LSPE(λ)算法如下：

$$\mathbf{w}_{n+1} = \arg\min_{\mathbf{w}} \sum_{k=0}^{n} \left(\boldsymbol{\phi}^T(i_k)\mathbf{w} - \boldsymbol{\phi}^T(i_k)\mathbf{w}_n - \sum_{m=k}^{n} (\gamma\lambda)^{m-k} d_n(i_m, i_{m+1})\right)^2 \tag{12.76}$$

其中 $(i_0, i_1, i_2, \cdots)$ 是由 Monte Carlo 模拟法生成的无限长轨迹。用语言表述：

在 LSPE(λ)算法的第 $n+1$ 次迭代，更新权值向量 $\mathbf{w}_{n+1}$ 作为权值向量 $\mathbf{w}$ 的特殊值来计算，它最小化下列两个量之间的最小二乘差：

- *逼近代价函数 $J(i_k)$ 的内积 $\boldsymbol{\phi}^T(i_k)\mathbf{w}$；*
- *时序差分的对应部分*

$$\boldsymbol{\phi}^T(i_k)\mathbf{w}_n + \sum_{m=k}^{n} (\gamma\lambda)^{m-k} d_n(i_m, i_{m+1})$$

这是对 $k=0,1,\cdots,n$ 由单个模拟轨迹中提取得到的。

注意权值向量 $\mathbf{w}_n$ 的当前值在完成式(12.76)的最小二乘最小化的每次迭代时保持为常数。

LSPE(λ)的逼近性质归于两个因子：

1. 估计稳定状态概率 π_i 和转移概率 p_{ij} 时，使用基于模拟的实验频率。
2. 为逼近 PVI 方法在式(12.76)中利用时序差分的有限折扣和。

无论如何，随着迭代数 n 趋于无穷，实验频率收敛于真实概率且有限折扣和收敛到无限折扣。所以，LSPE(λ)算法以渐进的意义收敛到其 PVI 部分。

下面关于 LSPE(λ)算法的收敛行为的具洞察力的备注是尤其值得注意的：

LEPS(λ)算法由快速收敛的确定性分量和慢慢收敛到 0 的随机分量组成，在算法的早期迭代阶段确定性分量支配了随机波动。

这一陈述是通过 Bertsekas 等（2004）的计算机模拟证实的。特别地，那里所示的结果说明 LSPE(λ)算法对 $0 \leqslant \lambda \leqslant 1$ 是真正可靠的算法，它收敛快，性能可靠。一般来说，选择靠近 1 的 λ 提高计算精确度（即，使得矩阵积 $\boldsymbol{\phi}^T(i)\mathbf{w}^*$ 靠近 $J(i)$），但增加了模拟噪声的影响，因而需要更多的样本和更长的轨迹来达到收敛。

12.11　逼近策略迭代

LSPE 算法为逼近动态规划提供了有力的线性方式。在本节中，我们描述利用神经网络作

为逼近动态规划非线性方法的工具。为此，假设有一个动态规划问题，它的可能状态数目和允许的行动数目非常大，使得利用传统处理方法是不现实的。假如我们有该系统的模型，即转移概率 $p_{ij}(a)$和观察代价 $g(i,a,j)$都是已知的。为了处理这种情况，我们基于下面所述的 Monte Carlo 模拟和最小二乘法提出使用策略迭代的近似。

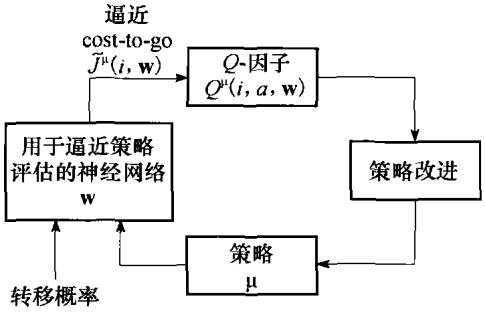


图 12.11　逼近策略迭代算法框图

图 12.11 给出逼近策略迭代算法的简化框图。在图 12.3 中的策略评估步骤由它的一个逼近所替代。因此逼近策略迭代算法交替进行如下的逼近策略评估步骤和策略改进步骤：

1. 逼近策略评估步骤。给定当前策略 μ，对所有状态 i 的实际 cost-to-go 函数 $J^{\mu}(i)$计算它的逼近，即 cost-to-go 函数 $\tilde{J}^{\mu}(i,\mathbf{w})$。向量 $\mathbf{w}$ 是完成逼近的神经网络参数。

2. 策略改进步骤。利用逼近 cost-to-go 函数 $\tilde{J}^{\mu}(i,\mathbf{w})$产生改进的策略 μ。对所有 i，新策略设计对 $\tilde{J}^{\mu}(i,\mathbf{w})$是贪心的。

为了逼近策略迭代算法产生满意解，仔细挑选策略初始化算法非常重要。这可利用启发式思想完成。或者可以从某个权值向量 $\mathbf{w}$ 开始，用它导出一个贪心策略，接着利用该策略为初始策略。

假设除知道转移概率和观察代价之外，我们有如下几项：

- 一个稳定的策略 μ 作为初始策略。
- 一个状态集$\mathscr{X}$代表运行环境。
- 对于每个 $i\in\mathscr{X}$，cost-to-go 函数 $J^{\mu}(i)$的 $M(i)$个样本组成的集合；一个这样的样本记为 $k(i,m)$，其中 $m=1,2,\cdots,M(i)$。

神经网络的参数向量 $\mathbf{w}$ 利用最小二乘法决定，即最小化代价函数：

$$\mathscr{E}(\mathbf{w})=\sum_{i\in\mathscr{X}}\sum_{m=1}^{M(i)}(k(i,m)-\tilde{J}^{\mu}(i,\mathbf{w}))^2 \tag{12.77}$$

在确定最优权值向量 $\mathbf{w}$ 从而有逼近 cost-to-go 函数 $\tilde{J}^{\mu}(i,\mathbf{w})$之后，下面确定逼近 Q-因子。为此，我们利用式(12.20)和式(12.23)来逼近 Q-因子：

$$Q(i,a,\mathbf{w})=\sum_{j\in\mathscr{X}}p_{ij}(a)(g(i,a,j)+\gamma\tilde{J}^{\mu}(j,\mathbf{w})) \tag{12.78}$$

其中 $p_{ij}(a)$为在行动 a(已知)下从状态 i 到状态 j 的转移概率，$g(i,a,j)$是观察代价（也为已知），而 γ 是规定的折扣因子。根据下列公式，通过使用这些逼近 Q-因子确定一种改进策略以完成迭代（参看 (12.28)）：

$$\mu(i)=\arg\min_{a\in\mathscr{U}_i}Q(i,a,\mathbf{w}) \tag{12.79}$$

注意，式(12.76)和式(12.77)仅被模拟器用在由模拟实际访问的状态而不是在所有状态产生行动。正因为如此，这两个公式没有受到维数灾的影响。

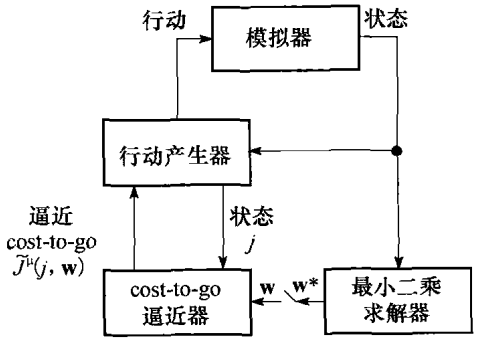


图 12.12　逼近策略迭代算法详细设计

图 12.12 给出一个逼近策略迭代算法的更加详细的框图。这个框图由四个互连的模块组成（Bertsekas and Tsitsiklis，1996）：

1. 模拟器，它利用给定的对状态转移概率和观察到的一步代价构建环境的一个替代模型。

模拟器产生两类东西：(a) 模拟环境的行动进行响应的状态，(b) 对给定策略 μ 的 cost-to-go 函数抽样。

2. 行动发生器，它根据式(12.77)产生一个改进策略（即一系列行动）。

3. cost-to-go 逼近器，它对状态 i 和参数向量 $\mathbf{w}$ 产生在式(12.76)和式(12.77)中使用的逼近 cost-to-go 函数 $\tilde{J}^{\mu}(i,\mathbf{w})$。

4. 最小二乘求解器，它利用由模拟器对策略 μ 和状态 i 提供的 cost-to-go 函数 $J^{\mu}(i)$的样本，计算使式(12.75)的代价函数最小化的参数向量 $\mathbf{w}$。只有充分评估一个策略和确定一个最优参数向量 $\mathbf{w}^*$ 之后，才能启动从最小二乘求解器到 cost-to-go 逼近器的连接。此时，由 $\tilde{J}^{\mu}(i,\mathbf{w}^*)$替代 cost-to-go 逼近 $\tilde{J}^{\mu}(i,\mathbf{w})$。

表 12.3 给出逼近策略迭代算法的小结。

表 12.3　逼近策略迭代算法

已知参数：转移概率 $p_{ij}(a)$和代价函数 $g(i,a,j)$。

计算：

1. 选择一个稳定策略 μ 作为初始策略。

2. 使用由模拟器产生的 cost-to-go 函数 $J^{\mu}(i)$的样本集 $\{k(i,m)\}_{m=1}^{M(i)}$，确定神经网络用作最小二乘求解器的参数向量 $\mathbf{w}$。

$$\mathbf{w}^* = \min_{\mathbf{w}} \mathscr{E}(\mathbf{w}) = \min_{\mathbf{w}} \sum_{i \in \mathscr{U}} \sum_{m=1}^{M(i)} (k(i,m) - \tilde{J}^{\mu}(i,\mathbf{w}))^2$$

3. 根据第 2 步决定的最优向量 $\mathbf{w}^*$，对访问的状态计算逼近 cost-to-go 函数 $\tilde{J}^{\mu}(i,\mathbf{w}^*)$。确定逼近 Q-因子：

$$Q(i,a,\mathbf{w}^*) = \sum_{j \in \mathscr{U}} p_{ij}(a)(g(i,a,j) + \gamma \tilde{J}^{\mu}(j,\mathbf{w}^*))$$

4. 确定改进策略

$$\mu(i) = \arg \min_{a \in \mathscr{A}_i} Q(i,a,\mathbf{w}^*)$$

5. 重复第 2 步至第 4 步。

注：第 3 步和第 4 步仅在实际访问的状态而不是所有状态上应用。

很自然，这个算法的运行会有误差，产生误差是由于模拟器和最小二乘求解器的设计有不可避免的缺点。对期望的 cost-to-go 函数进行最小二乘逼近的神经网络可能缺乏适当的计算能力，因而成为第一个误差源。神经网络逼近器的最优化和由此而来的参数向量 $\mathbf{w}$ 的调整是基于模拟器提供的期望反应，因此成为第二个误差源。假设所有的策略评估和策略改进是分别在 ε 和 δ 一定的误差容许限度内完成的，Bertsekas and Tsitsiklis (1996) 中证明逼近策略迭代算法所产生的策略和最优策略的性能之间差异的因子随 ε 和 δ 降低而趋于零。换句话说，逼近策略算法具有最小性能（差异）的可靠保证。根据 Bertsekas and Tsitsiklis (1996)，逼近策略迭代算法初始阶段能够取得迅速而且十分单调的进展，但在极限情况下一个随机性的持续的策略振荡可能发生。这种振荡行为出现在逼近 cost-to-go 函数 $\tilde{J}$ 到达最优值 J^* 的区域 $O((\delta+2\gamma\varepsilon)/(1-\gamma)^2)$内之后，其中 γ 为折扣参数。对所有逼近策略迭代的变体，它们都明显地有导致振荡行为的根本结构。

12.12　小结和讨论

本章前面部分介绍了用于多阶段决策动态规划的 Bellman 理论的详细讨论。作为建立在马尔可夫决策过程上的稳定策略，这一理论依赖于环境显式模型的有效性，该模型包含了转移概率和相连代价。我们还讨论了用于求解 Bellman 方程最优性的策略迭代和值迭代这两种方法。

逼近动态规划：直接方法

动态规划是强化学习的核心。本章中通过利用动态规划来推导两个在强化学习文献中广为

人知的无模型的在线学习算法：

- 时序差分（TD）学习，由 Sutton（1988）提出。
- Q-学习，由 Watkins（1989）提出。

由于是无模型的，这两个算法都避免了转移概率的需要。然而，存储的局限限制了它们在决策问题上的实际使用，只能用于中等大小的状态空间。

逼近动态规划：非直接方法

在本章的后面部分，我们讨论了有实际重要性的问题：*称为维数灾难的问题*。在解决大规模决策问题时遇到的这一问题使得 Bellman 动态规划变得棘手。为了克服这一困难，我们可以诉诸*非直接逼近动态规划*，它建立在 Bellman 理论基础上。非直接逼近动态规划可以通过如下两种途径中的一个来执行：

1. *线性构造方法*，包括两个步骤：
   - 状态 i 的特征提取
   - 代价 $\tilde{J}(i,\mathbf{w})$的最小二乘最小化，其中 $\mathbf{w}$ 是和状态 i 相连的权值向量

   我们通过推导最小二乘策略评估（LSPE）算法说明了这一方法的应用。

2. *非线性构造方法*，这一方法的提出依赖于通用逼近器的使用，它能以期望的任意精确度逼近任意非线性函数。神经网络可以作为通用逼近器来使用。

除了在逼近动态规划上的显著进展外[9]，在建立能够对大规模应用做高层决策的系统方面也有很多需要做的工作，这一系统是可靠的并且计算易处理的。在这一背景下，也许局部可观测性问题成为影响动态规划的所有实际问题的最大挑战。

部分可观测性

Bellman 动态规划理论假设了*完全可观测系统*（fully observable system）。更精确地说，为了最优策略解动态规划问题，假设环境状态服从马尔可夫性质：在时间 $n+1$ 的状态仅依赖于时间 n 的状态和策略，因而独立于时间 n 之前发生的所有一切。在实际中因为不可观测状态的发生是不可避免的，所以经常违背这一严格的假设。于是，作为基于马尔可夫决策过程（MDP）的模型（Bellman 动态规划理论的基石）的替代，如果我们要使逼近动态规划理论更接近实际现实，不得不处理*部分可观测马尔可夫决策过程*（POMDP）。某种意义上，部分可观测性可看成是第二个动态规划“灾”，称为“模型灾”，意味着可观测值包含了关于环境固有动态性的不完全信息。我们因而可以将动态规划描述为“遭受着模型灾和维数灾的全局最优化方法”。

多年以来，POMDP 问题在各种文献中被认为是严重的问题，在包含不确定下的规划（planning under uncertainty）的应用中（如机器人）设置了主要障碍。这一问题是困难的，因为需要学习行动选择策略，而行动选择可以是所有可能不确定类型中的偶然事件。注释和参考文献的注记 10 中，试图给出文献中如何处理 POMDP 问题的研究方向。

动态规划和 Viterbi 算法之间的关系

这一章主要是讲述动态规划。但是如果不讨论它和 Viterbi 算法的关系，动态规划的学习就是不完整的，Viterbi 算法的命名是因为其提出者 Viterbi（1968）。实际上，Bellman 动态规划（Bellman，1957；Bellman and Dreyfus，1962）比 Viterbi 的论文早好多年。这两个算法的等价性在 Omura（1969）中可以找到。

在最优化的背景下，动态规划试图寻找通过加权图的最短路径（如图 12.5 所示的驿车问题图），是通过从目的地开始一阶段一阶段回到起始点的方式来实现的。另一方面，在卷积解码的背景下，Viterbi 算法在权值图自身上工作，称为格子图（trellis diagram）。这个图表达了卷积编码器的图形描述，可看成有限状态机器（Lin and Costello，2004）。在最大似然意义下

Viterbi 算法对于卷积编码的最优性在 Forney（1973）中认识到。

注释和参考文献

1. 强化学习的传统处理方法植根于心理学，可追溯到 Thorndike（1911）关于动物学习早期的工作和 Pavlov（1927）关于条件反射的研究。Widrow 等（1973）的工作也对传统强化学习方法做出了贡献；在那篇文章中，引入了评价（critic）的概念。Hampson（1990）一书讨论了传统的强化学习。

   对现代强化学习的主要贡献包括 Samuel（1959）有关他的著名的棋子游戏程序的工作，Barto 等（1983）关于自适应评价系统的工作，Sutton（1988）关于时序差分（temporal difference）方法的工作和 Watkins（1989）关于 Q-学习的工作。在 Sutton and Barto（1998）的书中给出了强化学习的细节。

   在神经生物学背景下，报酬信号由称为多巴胺神经元的中脑神经元处理。为了详细地说明，在 Schultz（1998）中报告了一系列实验操作性条件反射被用于训练猴子对刺激（如，光和声音）的反应。为了得到以食物或饮料形式的报酬，猴子必须释放一个键，然后按另一个键。多巴胺的活动性结果在每次试验的 20 次实现下平均。Schultz 获得的结果揭示多巴胺神经元确实在刺激发生和报酬交付后激发。有了 Schultz 的值得关注的发现，我们如何对其建模？将多巴胺神经元看成“报酬系统的视网膜”，可以考虑将多巴胺神经元产生的响应作为 Pavlovian 条件反射和 TD-学习的教师信号（Schultz，2007；Iszhikevich，2007b）；然而需要注意的是 TD-学习的有关形式是 TD(λ)而不是 TD(0)，这两者都在 12.7 节中讨论过了。

   作为最后的备注：在强化学习文献中考虑 TD-学习时，报酬是最大化的。相反，在动态规划中考虑同样算法时，cost-to-go 函数是最小化的。

2. 本书在随机环境的一般背景下讨论了动态规划。因而重新给本章取个“随机动态规划”的题目是有吸引力的。然而，没有那样做，因为“动态规划”为工作于这一领域的研究者描述了合适的领域。

3. 策略迭代和值迭代是动态规划的两个主要方法。另外还有两个值得注意的方法：高斯-Seidel 方法和异步动态规划（Barto 等，1995；Bertsekas，1995b）。在高斯-Seidel 方法中，串行扫描所有状态，每个状态根据其他状态的最新代价进行竞争，在一个时刻只更新一个状态的 cost-to-go 函数。异步动态规划和高斯-seidel 的区别在于它没有组织成系统化的依次扫描状态集。

4. Watkin（1989）在他的博士论文的第 96 页，对 Q-学习做如下评论：

   “附录 1 给出这个学习方法对有限马尔可夫决策过程工作的证明。证明也表明该学习方法会很快收敛到最优行动-值函数。虽然这是非常简单的思想，但据我所知，以前从未被明显提出。但是必须指出，有限马尔可夫决策过程和随机动态规划用于若干不同领域已经被广泛研究 30 多年了，它不像 Monte-Carlo 方法那样以前无人考虑过。”

   在对这些评论的一个脚注中，Barto 等（1995）指出，虽然对状态-行动对赋值的思想被 Denardo（1967）所采用，构成动态规划方法的基础，但他们没有看见比 Watkins 的 1989 论文更早的像 Q-学习这样用于估计这些值的算法。

5. Watkins（1989）给出 Q-学习收敛定理证明的概要，后来在 Watkins and Dayan（1992）中对其进行了改进。Tsitsiklis（1994）给出了 Q-学习收敛的更一般的结果，也可参考 Bertsekas and Tsitsiklis（1996）。

6. 逼近动态规划的早期发展可追溯到 Werbos 在 1977 年的论文，其中第一次描述了避免维数灾的启发式动态规划思想。根据 Howard（1960），启发式动态规划的思想是逼近迭代过程的简单方法，是通过可调整权值的网络的有监督训练来实现的。

   现在，“逼近动态规划”通常被用于称呼用逼近来克服 Bellman 动态规划局限的方法。Bertsekas（2007）的书的第二卷有一章关于逼近动态规划，确定了逼近的直接和非直接方法。

7. 最小二乘时序差分（LSTD）算法

   根据 Bradtke and Barto（1996）的 LSTD 算法，为动态规划的非直接逼近提供了另一个线性结构方法。LSTD 算法的发展过程如下：

   - 基函数被用于表达每一个状态，Bellman 方程首先被通过这样的方式逼近：输入和输出观测作为噪声变量显示。
   - 然后，聪明地使用第 2 章讨论过的“媒介变量方法”，使之避免由“变量误差”问题引入的渐进偏置；这一阶段应用最小二乘方法。
   - 用一种和第 5 章讨论过的递归最小二乘（RLS）算法相似的过程，推导 LSTD 算法的相似的递归执行。

   LSTD 算法的原始方案是对 $\lambda=0$ 来推导的。建立在 Bradtke and Barto 工作基础上，Boyan（2002）扩展

LSTD 算法到 $\lambda>0$。LSTD 算法也在 Lagoudakis and Parr（2003）中在逼近策略迭代的背景下讨论。

8. 视觉皮层的反馈

主视觉皮层（视觉区域 1，通常简写为 V1）具有清晰的解剖层，每一个都有其自身的特性函数。V1 和更细节分析感知的高阶视觉区域相邻或相连（Kandel 等，1991）。

外侧膝状核（LGN）是大脑中处理视觉的部分（Kandel 等，1991）。

9. 逼近动态规划的书

Bertsekas and Tsitsiklis（1996）的经典书《神经动态规划》是关于逼近动态规划的第一本书。Si 等（2004）的编辑版中给出了在学习和逼近动态规划（ADP）、ADP 中的技术进步及其应用下这一课题的广泛讨论。

10. 部分可观测性

在部分可观测环境下规划的问题是非常困难的。下面文献的简短列举试图为这一高度挑战性领域的研究者提供有趣的方向：

（1）*分层方法*　在部分可观测环境下的规划可以简化为将一个困难任务分解为多层简单规划问题，这样的技术可以看成是工程上广为人知的“分步解决”范例的应用。Charlin 等（2007）研究了这一问题，通过将分层策略的最优化作为容易处理的一般非线性求解器的非凸最优问题来自动揭示分级结构。

Guestrin and Gordon（2002）中描述了协作多智能体动态系统 POMDP 的分层分解的另一种方法。在规划和执行阶段，计算在智能体中分布，每个智能体只需要模型化和规划系统的很小一部分。子系统通过分级结构联系在一起，这个结构通过消息传递算法在智能体间处理配位和通信；这样就能得到全局一致规划。另一个消息传递算法允许结果策略的执行。

（2）*POMDP 值迭代*　POMDP 的最优策略可以通过记为 J(b)的 cost-to-go 函数来表示。这个函数将信度状态（belief state）b（表示在可能真的但不可观测的世界构型上的后验分布）映射到最优策略能得到的总返回值的估计，假设 b 是正确的信度状态。尽管不可能精确地计算 cost-to-go 函数（Sondik，1971），但很多作者提出了逼近它的算法。特别地，称为基于点（point-based）的算法表明了潜在的保证（Smith，2007）。这些算法在信度的离散样本上估计 J(b)的值和梯度，通过利用 J(b)的凸性泛化到任意的信度。信度样本可以通过模拟 POMDP 得到可达信度的树来获得，也可以通过利用在随机选取的或在网格上放置的样本填充可能信度的单通道获得。

（3）*信度压缩*　在实际的 POMDP 问题中，大多数“信度”状态是不太可能的。更重要的是，在高维信度空间中包含着貌似真实的信度的结构化低维流形。Roy and Gordon（2003）介绍了一个新的称为“信度压缩”的方法来解大规模 POMDP 问题，它利用了信度空间的稀疏性。特别地，信度空间的维数可以通过利用指数族主分量分析（Collins 等 2002）来删减。（在第 10 章中讨论了可微流形。）

（4）*自然策略梯度*　在大规模 MDP 逼近规划直接策略梯度方法中，动机是通过未来返回值的梯度在策略的有限类中找到好的策略 μ。Kakade（2002）讲述了基于参数空间固有结构表示最速下降方向的自然梯度方法。和策略迭代的联系是通过证明自然梯度朝向选择贪心策略行动的移动来建立的。（Amari 自然梯度在第 10 章中讨论过了。）

习题

Bellman 最优准则

12.1　当折扣因子 γ 接近于 1 时，(12.22) 中 cost-to-go 函数的计算变长。为什么？说明你的回答的理由。

12.2　在本题中我们给出由 Ross（1983）得到的关于 Bellman 最优性方程（12.22）的另一个证明。

(a) 令 π 为任意策略，假设 π 在时间步 0 选择行动 a 的概率为 p_a，$a\in A_i$。那么

$$J^{\pi}(i)=\sum_{a\in\mathcal{A}_i}p_a\Big(c(i,a)+\sum_{j=1}^{N}p_{ij}(a)W^{\pi}(j)\Big)$$

其中 $W^{\pi}(j)$代表从时间步 1 以前的 cost-to-go 函数的期望，这里假设在时间步 1 状态为 j 且使用策略 π。由此证明

$$J^{\pi}(i)\geqslant\min_{a\in\mathcal{A}_i}\Big(c(i,a)+\gamma\sum_{j=1}^{N}p_{ij}(a)J(j)\Big)$$

其中

$$W^{\pi}(j)\geqslant\gamma J(j)$$

(b) 令 π 是在时间步 0 选择行动 a_0 的策略，如果下一个状态为 j，可看作过程以状态 j 开始，遵循策略 π_j 使得

$$J^{\pi_j}(j) \leqslant J(j) + \varepsilon$$

其中 ε 是一个很小的正数。由此证明

$$J(i) \geqslant \min_{a \in \mathcal{A}_i} \left(c(i,a) + \gamma \sum_{j=1}^{N} p_{ij}(a) J(j) \right) + \gamma\varepsilon$$

(c) 用 (a) 和 (b) 导出的结果证明式(12.22)。

12.3 式(12.22)表示 N 个方程的线性方程组，每个状态用一个方程。令

$$\mathbf{J}^{\mu} = [J^{\mu}(1), J^{\mu}(2), \cdots, J^{\mu}(N)]^T$$

$$\mathbf{c}(\mu) = [c(1,\mu), c(2,\mu), \cdots, c(N,\mu)]^T$$

$$\mathbf{P}(\mu) = \begin{bmatrix} p_{11}(\mu) & p_{12}(\mu) & \cdots & p_{1N}(\mu) \\ p_{21}(\mu) & p_{22}(\mu) & \cdots & p_{2N}(\mu) \\ \vdots & \vdots & & \vdots \\ p_{N1}(\mu) & p_{N2}(\mu) & \cdots & p_{NN}(\mu) \end{bmatrix}$$

证明式(12.22)可以重新写成等价的矩阵形式：

$$(\mathbf{I} - \gamma \mathbf{P}(\mu))\mathbf{J}^{\mu} = \mathbf{c}(\mu)$$

其中 $\mathbf{I}$ 为单位矩阵。讨论表示 N 个状态的 cost-to-go 函数的向量 $\mathbf{J}^{\mu}$ 的唯一性。

12.4 12.3 节中推导了用于有限范围问题的动态规划算法。在本题中对一个折扣问题重新推导这个算法，其中 cost-to-go 函数由下式定义：

$$J^{\mu}(X_0) = \lim_{K \to \infty} \left[\sum_{n=0}^{K-1} \gamma^n g(X_n, \mu(X_n), X_{n+1}) \right]$$

特别地，证明

$$J_K(X_0) = \min_{\mu} E_{X_1}[g(X_0, \mu(X_0), X_1) + \gamma J_{K-1}(X_1)]$$

策略迭代

12.5 在 12.4 节中我们说 cost-to-go 函数满足

$$J^{\mu_{n+1}}(i) \leqslant J^{\mu_n}(i), \quad \text{对于所有的 } i$$

证明这个论断。

12.6 讨论式(12.25)描述的论断的重要性。

12.7 利用控制器评价系统 (controller-critic system)，说明策略迭代算法中策略更新和策略求值之间的相互作用。

值迭代

12.8 一个动态规划问题共涉及 N 个允许状态 M 个允许行动。假定使用一个稳定策略，证明值迭代算法的一次迭代需要阶为 N^2M 的操作。

12.9 表 12.2 给出依据对状态 $i \in \mathcal{X}$ 的 cost-to-go 函数 $J^{\mu}(i)$ 构造的值迭代算法公式的小结。依据 Q-因子 $Q(i,a)$ 重新构造这个算法公式。

12.10 策略迭代总是在有限步后终止，但是值迭代可能要无限次迭代。讨论这两个动态规划方法之间的其他差异。

时序差分学习

12.11 (a) 构造在式(12.34)和式(12.35)中描述的 TD(0)算法的信号流图表示。

(b) TD(0)算法具有和第 3 章描述的 LMS 算法相似的数学组成。讨论这两个算法之间的异同点。

12.12 证明式(12.40)的样本均值可以通过式(12.42)的迭代公式来计算。

12.13 (a) 证明等式 1 和 2 是从式(12.45)和(12.46)而来。

(b) 构造式(12.48)的信号流图表示，描述 TD(λ)算法。

Q-学习

12.14 证明

$$J^*(i) = \min_{a \in \mathcal{A}_i} Q(i,a)$$

12.15 Q-学习算法有时称作值迭代策略的自适应形式。证明这种描述的正确性。

12.16　构造由表 P12.16 小结的逼近 Q-学习算法的信号流图。

12.17　表 P12.16 小结的逼近 Q-学习算法假定缺乏状态转移概率的知识。假定可以用这些概率，重构这个算法。

逼近动态规划：非直接方法

12.18　式(12.70)是投影值迭代（PVI）算法的最小二乘方案。为了实际执行这一算法，我们提议利用 Monte Carlo 模拟法来逼近它，这里运用在式(12.71)中描述的最小二乘策略评估（LSPE）算法。

(a) 通过设式(12.70)的代价函数的梯度为 0，推导 $\mathbf{w}_{n+1}$ 的闭逻辑式。

(b) 对式(12.71)同样地做。寻找状态 i 的实验频率和转移 (i,j)（即估计稳定状态概率 π_i 和转移概率 p_{ij}）来说明 PVI 和 LSPE 算法一致渐进。

12.19　LSPE(λ)算法比 TD(λ)算法具有更快的收敛速率。证明这一陈述。

12.20　图 P12.20 显示了逼近目标 Q-因子的基于神经网络的方案，目标 Q-因子记为 $Q^{\text{target}}(i,a,\mathbf{w})$，其中 i 记网络的状态，a 记要采取的行动，$\mathbf{w}$ 记在逼近中使用的神经网络的权值向量。相应地，表 P12.16 给出了逼近 Q-学习算法的小结。解释图 P12.20 的逼近动态规划的运行以证明表 P12.16 中的小结。

表 P12.16　逼近 Q-学习算法小结

1. 从初始权值向量 $\mathbf{w}_0$ 开始，得到 Q-因子 $Q(i_0,a_0,\mathbf{w}_0)$；权值向量 $\mathbf{w}_0$ 借助所用的神经网络完成逼近。

2. 对迭代 $n=1,2,\cdots$，做下面几步：

(a) 对于神经网络设定的 $\mathbf{w}$，确定最优行动

$$a_n = \min_{a\in\mathcal{A}_{j_n}} Q_n(i_n,a,\mathbf{w})$$

(b) 确定目标 Q-因子

$$Q_n^{\text{target}}(i_n,a_n,\mathbf{w}) = g(i_n,a_n,j_n) + \gamma \min_{b\in\mathcal{A}_{j_n}} Q_n(j_n,b,\mathbf{w})$$

(c) 更新 Q-因子

$$Q_{n+1}(i_n,a_n,\mathbf{w}) = Q_n(i_n,a_n,\mathbf{w}) + \Delta Q_n(i_n,a_n,\mathbf{w})$$

其中

$$\Delta Q_n(i_n,a_n,\mathbf{w}) = \begin{cases} \eta_n(i_n,a_n)(Q_n^{\text{target}}(i_n,a_n,\mathbf{w}) - Q_n(i_n,a_n,\mathbf{w})) & (i,a) = (i_n,a_n) \\ 0 & \text{其他} \end{cases}$$

(d) 应用 (i_n, a_n) 作为神经网络的输入，产生输出 $\hat{Q}_n(i_n,a_n,\mathbf{w})$ 作为目标 Q-因子 $Q_n^{\text{target}}(i_n,a_n,\mathbf{w})$ 的逼近。轻微地改变权值向量使得 $\hat{Q}_n(i_n,a_n,\mathbf{w})$ 更靠近目标值 $Q_n^{\text{target}}(i_n,a_n,\mathbf{w})$

(e) 回到步骤 (a)，重复计算。

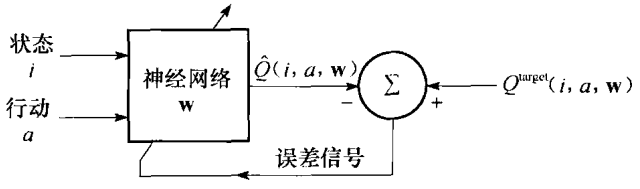


图　P12.20

神经动力学

本章组织

本章研究递归神经网络，重点放在用 Lyapunov 直接方法来解决稳定性问题上。

本章组织如下

13.1 节提出确定神经动力学系统的稳定性的研究动机，指出这个问题的历史观点。

13.2 节到 13.6 节提供背景材料。特别地，13.2 节介绍一些动态系统中的基本概念，随后在 13.3 节中讨论平衡点稳定性。13.4 节中描绘在动态系统研究中浮现出的各种类型的吸引子。在 13.5 节再次讨论神经元的加性模型。13.6 节讨论作为神经网络范例的吸引子的运作。

13.7 节到 13.9 节是本章的第二部分，处理联想记忆。13.7 节详细讨论 Hopfield 模型和作为按内容寻址记忆使用的离散 Hopfield 模型的细节问题。13.8 节中对非线性系统给出了它们的 Cohen－Grossberg 定理，系统包含 Hopfield 网络和其他联想记忆作为其特例。13.9 节描述另一个被称为盒中脑状态模型的神经动力学模型，该模型非常适用于聚类。

最后部分是 13.10 节到 13.11 节，处理混沌的相关论题。13.10 节讨论混沌过程的不变特征，随后在 13.11 节讨论混沌过程动力学重建这一紧密相关题目。

最后是 13.12 节的评论。

13.1 引言

以这种或那种形式，时间在学习中扮演着重要的角色，本书前面的章节中多数材料例示了这点。基本上说，时间以两种方式显示了它在学习过程中的作用：

1. 静态神经网络（如第 4 章中的多层感知器）将它通过一个或短或长的记忆结构作为动态映射器运行。
2. 把时间以隐含的方式嵌入神经网络的运行之中的重要途径是通过使用反馈。

把反馈应用于神经网络有两种基本途径：

1. *局部反馈*，应用于网络的单一神经元层次上。
2. *全局反馈*，它包括一个或多个隐藏神经元或更好的整个网络。

局部反馈处理起来相对简单，但全局反馈有更深的含义。在关于神经网络的文献中，带有一个或者更多反馈回路的神经网络被称为递归网络。

基本上，递归神经网络有两个功能：

1. 联想记忆
2. 输入-输出映射网络

本章讨论把递归神经网络作为联想记忆，作为映射器的使用推迟到第 15 章讲述。这两个功能中的任何一个都是感兴趣的应用，其中一个特别重要的主题是稳定性，在本章中也将讨论。

反馈就像一柄双刃剑，如果你不能恰当地使用它，它就会产生负面效果。特别地，反馈的应用能导致本来是稳定的系统变得不稳定。在这一章中，我们的主要兴趣在于递归网络的稳定性。

神经网络视为非线性动力系统，并特别强调其稳定性问题，称为*神经动力学*（neurodynamics）。非线性动力系统的稳定性（或不稳定性）的一个重要特征就在于它是整个系统的特

性。作为一个推论：

稳定性的存在总是意味着在系统的各个独立部分之间某种形式的协调。

对神经动力学的研究开始于 1938 年 Nicholas Rashevsky 的工作，在他富于想象力的思维中动力学第一次应用于生物学。

非线性动态系统的稳定性是一个处理起来很棘手的问题。当谈到稳定性问题的时候，拥有工程背景的人经常会想到*有界输入和有界输出*（BIBO）的*稳定性准则*。依照这一准则，稳定性意味着如果有界的输入，初始条件或不必要干扰，那么系统的输出就必定不会无界地增长。BIBO 稳定性准则非常适合于线性动态系统。但是，由于嵌入神经元结构之中的饱和非线性使得所有的这样一些非线性动态系统都是 BIBO 稳定的，所以把 BIBO 稳定性准则应用到神经网络上是无用的。

当在非线性动态系统背景下谈到稳定性时，我们通常都意味着 Lyapunov *意义的稳定性*。在 1892 年一个值得庆贺的日子里，Lyapunov（一位俄罗斯数学家和工程师）提出了众所周知的稳定性理论的基本概念——Lyapunov *直接方法*。这一方法被广泛用于线性和非线性系统中的稳定性分析，包括时不变和时变两种情况。因此，它可以直接用于神经网络中的稳定性分析。事实上，本章中提到的很多材料都涉及 Lyapunov 直接方法。但是，它的应用不是一个轻松的任务。

对神经动力学的研究可能会遵从两种途径之一，这取决于实际的应用：

- *确定性神经动力学*：此时神经网络模型带有确定的行为。数学上用一组非线性微分方程来描述，微分方程定义作为时间函数的模型的精确进化（Grossberg，1967；Cohen and Grossberg，1983；Hopfield，1984）。
- *统计性神经动力学*：此时神经网络受到存在噪声的干扰。在这种情况下，我们将不得不处理随机非线性微分方程组，因而用概率术语表示解（Amari 等，1972；Peretto，1984；Amari，1990）。随机性和非线性的组合使得这个主题非常难于处理。

在本章中，我们将限制在确定性神经动力学之内。

13.2　动态系统

为了进行神经动力学的研究，我们需要用一个*数学模型*描述非线性系统的动力学。自然最适合这一用途的模型就是*状态空间模型*。根据这个模型，我们考虑一组*状态变量*，假设这些变量的值（在任意特定时刻）都包含充分的信息，可以预测系统的可能演化。令 $x_1(t)$，$x_2(t),\cdots,x_N(t)$表示非线性动态系统的状态变量，其中连续时间 t 是*独立变量*且 N 为系统的*阶*。为了简化符号，把这些状态变量收集在一个叫做*系统状态向量*，或简称为*状态*的 $N\times 1$ 的向量 $x(t)$里。那么非线性动态系统的一大类的动力学特性就可以用一阶微分方程组的形式给出

$$\frac{\mathrm{d}}{\mathrm{d}t}x_j(t) = F_j(x_j(t)),\quad j = 1,2,\cdots,N \tag{13.1}$$

其中的函数 $F_j(\cdot)$是它的自变量的非线性函数。我们可以用向量符号把这个方程组写成紧凑形式：

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}(t) = \mathbf{F}(\mathbf{x}(t)) \tag{13.2}$$

其中非线性函数 $\mathbf{F}$ 是向量值的，它的每一个元素作用于下述状态向量中的一个对应元素：

$$\mathbf{x}(t) = [x_1(t),x_2(t),\cdots,x_N(t)]^T \tag{13.3}$$

如在式(13.2)中那样，若向量函数 $\mathbf{F}(\mathbf{x}(t))$不显式地依赖于时间 t，则这样的非线性动态系统

被称为自治的（autonomous）；否则称为非自治的（nonautonomous）[1]。

不管非线性函数 $\mathbf{F}(\cdot)$的精确形式是什么，状态向量 $\mathbf{x}(t)$必须随时间改变；否则，$\mathbf{x}(t)$就是常量而系统也不再是动态的。因此我们可以正式定义一个动态系统如下：

动态系统是状态随时间变化的系统。

此外，我们可以把 $d\mathbf{x}/dt$ 作为“速度”来考虑，不是在物理意义上而是在抽象意义上的。那么，根据式(13.2)，可以将向量函数 $\mathbf{F}(\mathbf{x})$称为速度向量场或者简单地称为向量场（vector field）。

状态空间

将状态空间方程（13.2）看做描述 N 维状态空间中一个点的运动是有益的。状态空间可能是欧几里得空间或者是它的一个子集。也可能是非欧几里得空间，就像圆、球、环或者其他一些*微分流形*。但是，我们的兴趣只限于欧几里得空间（第 7 章中已讨论过微分流形）。

状态空间很重要，因为它给我们提供可视的且概念化的工具，用来分析由式(13.2)描述的非线性系统的动力学。它是通过把我们的注意力集中于运动的*全局特性*而不是方程的解析解或数值解的细节方面来实现的。

在某一特定时刻 t，用 N 维状态空间中的一个点表示系统被观察状态（即状态向量 $\mathbf{x}(t)$）。用状态空间中的一条曲线表示系统状态随时间 t 的变化，曲线上的每一点都（显式地或隐含地）带有记录观察时间的标记。这条曲线叫做系统的轨线或轨道。图 13.1 描绘了一个二维系统的轨线。轨线的瞬时速度（即速度向量 $d\mathbf{x}(t)/dt$)用*切向量*表示，如图 13.1 中 $t=t_0$ 时刻用虚线的表示。因此我们可以得出轨线上每一点的速度向量。

由不同初始条件产生的不同轨线的集合称为系统的*状态相图*（state portrait）。状态相图包含状态空间中所有那些定义向量场 $\mathbf{F}(\mathbf{x})$的点。注意对于自治系统来说，每种初始状态将只有一条轨线穿过。从状态相图产生的一个有用概念是动态系统的*流*（flow），被定义为状态空间在系统内部的运动。换句话说，可以想象一下状态空间在自身内部流动，就像一种流体，每一个点（状态）沿着一条特定轨线的流动。这里描述的流的思想在图 13.2 的状态相图中有生动的说明。

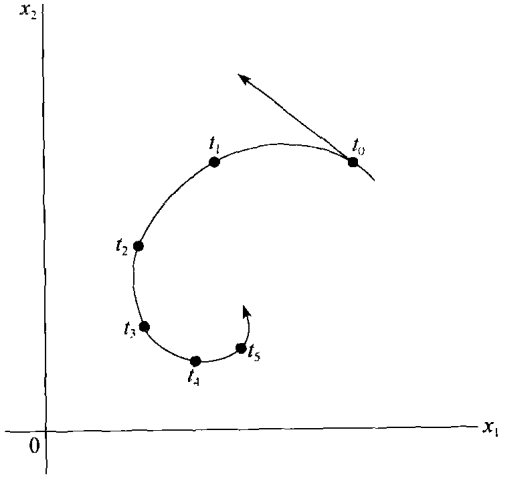


图 13.1　二维动态系统的轨线（轨道）

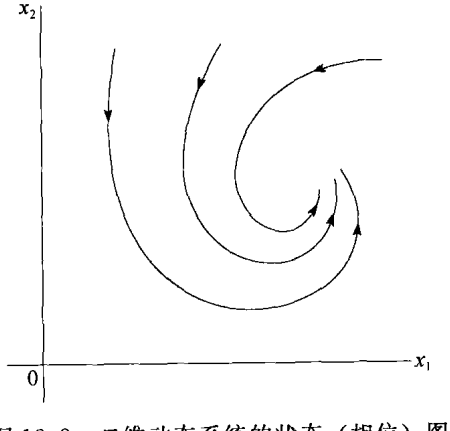


图 13.2　二维动态系统的状态（相位）图

给定一个动态系统的状态相图，可以构造一个对应于状态空间中每一个点的速度（切线）

向量场。这样得到的图也提供了系统中向量场的描绘。图 13.3 中显示许多速度向量，展现完全的场看起来像什么。向量场的用处在于事实上它通过在状态空间中每一个特定点以惯性速度移动，给我们提出一种对动态系统固有运动倾向的可视描述。

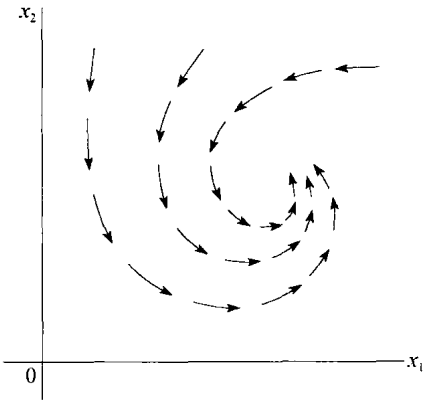


图 13.3　二维动力系统向量场

Lipschitz 条件

为了状态空间方程式(13.2)有解且是唯一解，必须在向量函数 $\mathbf{F}(\mathbf{x})$ 上施加一定的限制。为了便于表示，我们已经舍弃了状态向量 $\mathbf{x}$ 对时间 t 的依赖，而这是我们一次又一次遵从的惯例。存在解的充分条件为 $\mathbf{F}(\mathbf{x})$ 对它的所有自变量是连续函数。然而，它这一限制本身不足以保证解的唯一性。为了做到这一点，我们必须施加被称为 Lipschitz 条件的额外限制。令 $\|\mathbf{x}\|$ 表示向量 $\mathbf{x}$ 的范数或者欧几里得长度。令 $\mathbf{x}$ 和 $\mathbf{u}$ 作为赋范向量（状态）空间上某一开集 $\mathcal{M}$ 上的一个向量对。然后，根据 Lipschitz 条件，存在一个常量 K 使得下式对 $\mathcal{M}$ 中所有的 $\mathbf{x}$ 和 $\mathbf{u}$ 都成立（Hirsch and Smale，1974；Jackson，1989）：

$$\|\mathbf{F}(\mathbf{x})-\mathbf{F}(\mathbf{u})\| \leqslant K\|\mathbf{x}-\mathbf{u}\| \tag{13.4}$$

满足式(13.4)的向量值函数 $\mathbf{F}(\mathbf{x})$ 被称为满足 Lipschitz 条件，K 叫做 $\mathbf{F}(\mathbf{x})$ 的 Lipschitz 常数。式(13.4)也意味着函数 $\mathbf{F}(\mathbf{x})$ 关于 $\mathbf{x}$ 的连续性。因此，对自治系统来说，Lipschitz 条件是状态空间方程(13.2)存在且只存在唯一解的充分条件。特别地，如果所有偏导数 $\partial F_i/\partial x_j$ 处处有限，则函数 $\mathbf{F}(\mathbf{x})$ 满足 Lipschitz 条件。

散度定理

考虑自治系统状态空间中某个容积 V 和曲面 S 的区域，并且设想由区域的点组成的“流”。从以前的讨论，我们认识到速度向量 $\mathrm{d}\mathbf{x}/\mathrm{d}t$ 和向量场 $\mathbf{F}(\mathbf{x})$ 是相等的。倘若容积 V 内的向量场 $\mathbf{F}(\mathbf{x})$ 是相当光滑，则可以从向量微积分学的角度应用散度定理（Jackson，1975）。令 $\mathbf{n}$ 表示曲面 S 上某小块 $\mathrm{d}S$ 处指向所包含容积外部的单位法向量。然后，根据散度定理，关系式

$$\int_S (\mathbf{F}(\mathbf{x})\cdot\mathbf{n})\mathrm{d}S = \int_V (\nabla\cdot\mathbf{F}(\mathbf{x}))\mathrm{d}V \tag{13.5}$$

在 $\mathbf{F}(\mathbf{x})$ 散度的容积积分和 $\mathbf{F}(\mathbf{x})$ 向外法线分量的曲面积分之间成立。式(13.5)左端的值被认为是从曲面 S 所包围的区域中流向外部的净流量。如果该值为零，则说系统是保守的（conservative）；若为负，则说系统是耗散的（dissipative）。根据式(13.5)，同样可以说：

如果散度 $\nabla\cdot\mathbf{F}(\mathbf{x})$（一个标量）为零则系统是保守的，若为负则系统是耗散的。

13.3　平衡状态的稳定性

考虑由状态空间方程(13.2)描述的自治动态系统。一个常向量 $\bar{\mathbf{x}}\in\mathcal{M}$ 称为系统的平衡（稳定）状态，如果条件

$$\mathbf{F}(\bar{\mathbf{x}}) = 0 \tag{13.6}$$

满足，其中的 $\mathbf{0}$ 为零向量。速度向量 $\mathrm{d}\mathbf{x}/\mathrm{d}t$ 在平衡状态 $\bar{\mathbf{x}}$ 处消失，因此常量方程 $\mathbf{x}(t)=\bar{\mathbf{x}}$ 是方程(13.2)的解。此外，由于解的唯一性，没有其他的解曲线能够穿过平衡状态 $\bar{\mathbf{x}}$。平衡状态也称为奇异点，表示在平衡点这种情况下，轨线将会退化到这个点本身。

为了加深对平衡条件的理解，假设非线性函数 $\mathbf{F}(\mathbf{x})$ 对于状态空间方程(13.2)来说足够光滑，使得在 $\bar{\mathbf{x}}$ 的邻域可以作为线性函数处理。特别是，令

$$\mathbf{x}(t) = \bar{\mathbf{x}} + \Delta\mathbf{x}(t) \tag{13.7}$$

其中的 $\Delta\mathbf{x}(t)$是 $\bar{\mathbf{x}}$ 的微小偏差。然后，保留 $\mathbf{F}(\mathbf{x})$的 Taylor 级数展开中的前两项，将其近似为$\mathbf{F}(\mathbf{x})$

$$\mathbf{F}(\mathbf{x}) \approx \bar{\mathbf{x}} + \mathbf{A}\Delta\mathbf{x}(t) \tag{13.8}$$

矩阵 $\mathbf{A}$ 是非线性方程 $\mathbf{F}(\mathbf{x})$的 Jacobi 矩阵，在 $\mathbf{x}=\bar{\mathbf{x}}$ 点处计值，表示为

$$\mathbf{A} = \frac{\partial}{\partial \mathbf{x}}\mathbf{F}(\mathbf{x})\Big|_{\mathbf{x}=\bar{\mathbf{x}}} \tag{13.9}$$

将式(13.7)和式(13.8)代入式(13.2)，然后使用平衡状态的定义，得到

$$\frac{\mathrm{d}}{\mathrm{d}t}\Delta\mathbf{x}(t) \approx \mathbf{A}\Delta\mathbf{x}(t) \tag{13.10}$$

倘若 Jacobi 矩阵 $\mathbf{A}$ 是非奇异的，即逆矩阵 $\mathbf{A}^{-1}$存在，则式(13.10)描述的近似值足以确定系统轨线在平衡状态邻域 $\bar{\mathbf{x}}$ 的局部性质。如果 $\mathbf{A}$ 是非奇异的，则平衡状态的性质主要取决于 $\mathbf{A}$ 的特征值，因此可以根据它的相应方式进行分类。特别地，当 Jacobi 矩阵 $\mathbf{A}$ 的特征值有 m 个带有正实数部分时，我们可以说 $\bar{\mathbf{x}}$ 平衡状态具有类型(type)m。

对于二阶系统这种特殊情况而言，平衡状态的分类可归结为表 13.1 所列的情况，相应相图表示在图 13.4 中（Cook，1986；Arrowsmith and Place，1990）。不失泛化性，假设平衡状态位于状态空间的原点，也就是 $\mathbf{x}=0$ 的地方。注意对于图 13.4e 中的鞍点，通向鞍点的轨线是稳定的，而从鞍点离开的轨线则是不稳定的。

表 13.1　二阶系统平衡状态的分类

平衡状态 $\bar{\mathbf{x}}$ 的类型	Jacobi 矩阵 $\mathbf{A}$ 的特征值
稳定结点	负实数
稳定焦点	实部为负的共轭复数
不稳定结点	正实数
不稳定焦点	实部为正的共轭复数
鞍点	不同号的实数
中心	共轭纯虚数

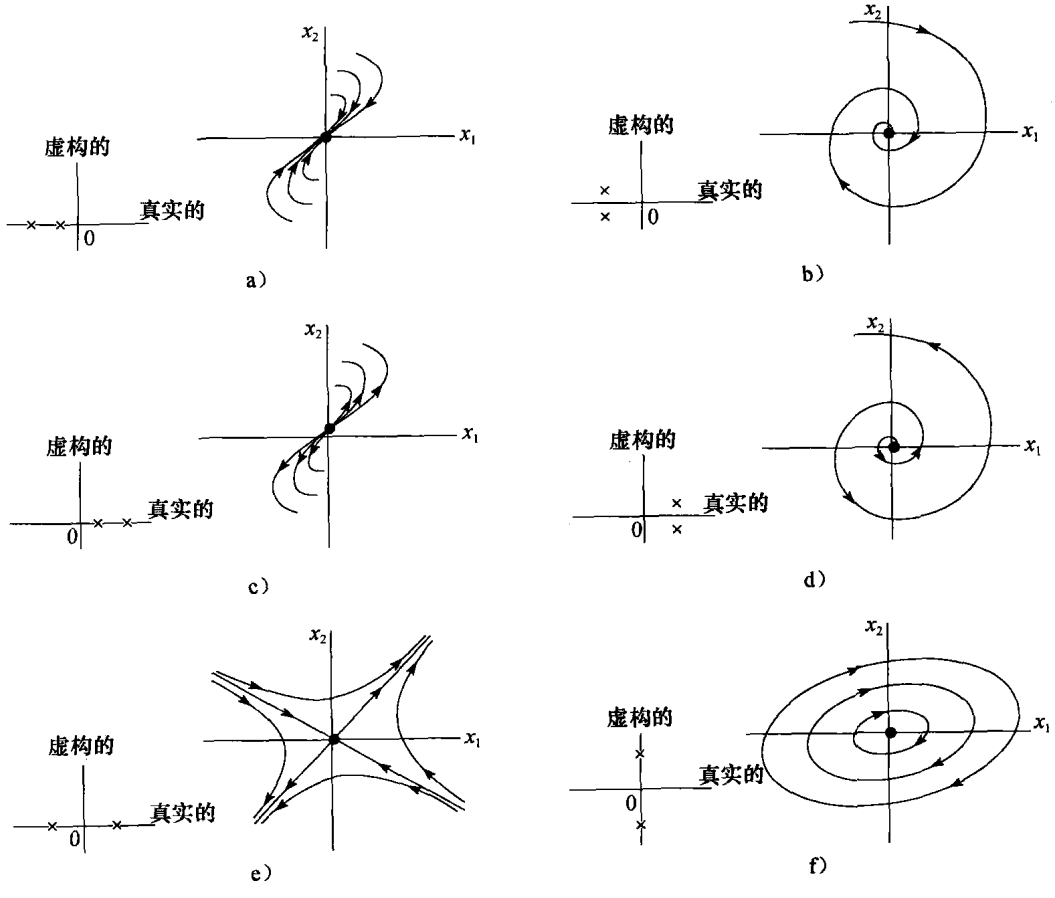


图 13.4　a）稳定结点；b）稳定焦点；c）不稳定结点；d）不稳定焦点；e）鞍点；f）中心

稳定性定义

如前所述，状态空间方程的线性化可以提供关于一个平衡状态的局部稳定特性的有用信息。但是，为了能以一种更加细节化的方式研究非线性动态系统的稳定性，我们需要关于平衡状态的稳定性和收敛性的精确定义。

在与带有平衡状态 $\bar{\mathbf{x}}$ 的自治非线性动态系统相关的环境中，稳定性和收敛性的定义如下(Khalil，1992)：

定义 1　若对于任意给定的正数 ε，存在一正数 $\delta=\delta(\varepsilon)$，使得当满足条件

$$\|\mathbf{x}(0)-\bar{\mathbf{x}}\|<\delta$$

时，对于所有 $t>0$ 恒有

$$\|\mathbf{x}(t)-\bar{\mathbf{x}}\|<\varepsilon$$

则称平衡状态 $\bar{\mathbf{x}}$ 为一致稳定的。

这一定义表明如果初始状态 $\mathbf{x}(0)$ 很接近 $\bar{\mathbf{x}}$，则系统的一条轨线可能会停留在平衡状态 $\bar{\mathbf{x}}$ 很小的一个邻域内，否则系统将不平衡。

定义 2　如果存在一个正数 δ 使得当条件

$$\|\mathbf{x}(0)-\bar{\mathbf{x}}\|<\delta$$

时，对于

$$\mathbf{x}(t)\to\bar{\mathbf{x}},\quad t\to\infty$$

则称平衡状态 $\bar{\mathbf{x}}$ 为收敛的。

第二个定义的含义是，如果一条轨线的初始状态 $\mathbf{x}(0)$ 足够接近于平衡状态 $\bar{\mathbf{x}}$，则在时间 t 接近无穷的时候，由状态向量 $\mathbf{x}(t)$ 所描述的轨线将收敛于 $\bar{\mathbf{x}}$。

定义 3　若平衡状态是稳定的并且是收敛的，则称平衡状态 $\bar{\mathbf{x}}$ 为渐近稳定的。

这里要注意稳定性和收敛性是互相独立的性质。只有两者都具备才有渐近稳定性。

定义 4　如果平衡状态是稳定的，并且所有的系统轨线在时间 t 接近无穷的时候都收敛于 $\bar{\mathbf{x}}$，则称平衡状态 $\bar{\mathbf{x}}$ 为全局渐近稳定的。

这一定义意味着系统不可能有其他的平衡状态，而且它要求系统中的每一条轨线对所有的时间 $t>0$ 都保持有界。换句话说，全局渐近稳定性意味着对于任意初始条件系统都将最终稳定在一个稳态上。

例 1　一致稳定性

令式(13.2)表示的非线性动态系统的解 $\mathbf{u}(t)$ 随时间变化，就像图 13.5 中显示的那样。如图 13.5 所示，为了解 $\mathbf{u}(t)$ 是一致稳定的，我们需要 $\mathbf{u}(t)$ 和任何其他解 $\mathbf{v}(t)$ 在同样的 t 值（即时间“滴答”）时保持互相接近。这种行为被称为两个解 $\mathbf{u}(t)$ 和 $\mathbf{v}(t)$ 的同步对应（isochronous correspondence）。设解 $\mathbf{u}(t)$ 是收敛的，假定对于每一个其他的解 $\mathbf{v}(t)$，在 $t=0$ 处 $\|\mathbf{v}(0)-\mathbf{u}(0)\|\leqslant\delta(\varepsilon)$ 成立，则解 $\mathbf{v}(t)$ 和 $\mathbf{u}(t)$ 当 t 趋于无穷时收敛于平衡状态。

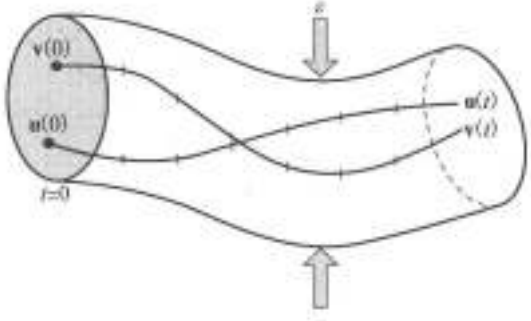


图 13.5　状态向量一致稳定的概念图示

■

Lyapunov 定理

定义了动态系统的稳定性和渐近稳定性之后，下一个要考虑的问题就是确定稳定性。显而易见我们可以通过实际地找到系统状态空间方程的所有可能解来做到；但是这种方法（即使有可能）也是非常困难的。一个更好的方法可以在现代稳定性理论中找到，该理论由 Lyapunov（1892）创立。

具体地讲，我们可以通过应用 Lyapunov 直接方法来研究稳定性问题，这个方法使用叫做 Lyapunov 函数的状态向量的连续标量函数。

由方程(13.2)描述的具有状态向量 $\mathbf{x}(t)$和平衡状态 $\bar{\mathbf{x}}$ 的自治非线性动态系统，关于它的状态空间的稳定性和渐近稳定性的 Lyapunov 定理可以陈述如下（Khalil，1992）：

定理 1　如果在 $\bar{\mathbf{x}}$ 的小邻域内存在一个正定函数 $V(\mathbf{x})$，其对时间的导数在该区域内是半负定的，则平衡状态 $\bar{\mathbf{x}}$ 是稳定的。

定理 2　如果在 $\bar{\mathbf{x}}$ 的小邻域内存在一个正定函数 $V(\mathbf{x})$，其对时间的导数在该区域内是负定的，则平衡状态 $\bar{\mathbf{x}}$ 是渐近稳定的。

满足以上要求的标量函数 $V(\mathbf{x})$叫做平衡状态的 $\bar{\mathbf{x}}$ 的 Lyapunov 函数。

这两个定理要求 Lyapunov 函数是正定函数。这样的函数定义如下：

1. 函数 $V(\mathbf{x})$对状态向量 $\mathbf{x}$ 中所有元素有连续偏导数

2. $V(\bar{\mathbf{x}})=0$

3. 如果 $\mathbf{x}\in\mathcal{U}-\bar{\mathbf{x}}$，则 $V(\mathbf{x})>0$

给出这样的 Lyapunov 函数 $V(\mathbf{x})$，根据定理 1，若

$$\frac{\mathrm{d}}{\mathrm{d}t}V(\mathbf{x})\leqslant 0,\quad \text{对于 } \mathbf{x}\in\mathcal{U}-\bar{\mathbf{x}} \tag{13.11}$$

成立[2]，则平衡状态 $\bar{\mathbf{x}}$ 是稳定的。此外，根据定理 2，若

$$\frac{\mathrm{d}}{\mathrm{d}t}V(\mathbf{x})<0,\quad \text{对于 } \mathbf{x}\in\mathcal{U}-\bar{\mathbf{x}} \tag{13.12}$$

成立，则平衡状态 $\bar{\mathbf{x}}$ 是渐近稳定的。

这一讨论的重要之处在于可以不求解系统的状态空间方程而直接应用 Lyapunov 定理。不幸的是，定理并没有给出如何找到 Lyapunov 函数的提示；在每种情况下，它是一件创造性的、试验性的和易错的事情。对于感兴趣的很多问题，能量函数可以起到 Lyapunov 函数的作用。但是，无法找到适用的 Lyapunov 函数并不能证明系统的不稳定性。因为 Lyapunov 函数的存在是系统稳定的充分条件，而不是必要条件。

Lyapunov 函数 $V(\mathbf{x})$为对由式(13.2)描述的非线性动态系统进行稳定性分析提供了数学基础。另一方面，基于 Jacobi 矩阵 $\mathbf{A}$，使用式(13.10)为进行系统局部稳定性分析提供基础。简单地说，Lyapunov 稳定性分析的结论比局部分析更有力。

Lyapunov 平面

为了直观地理解两个 Lyapunov 定理，我们引入 Lyapunov 平面的概念，正式定义如下

$$V(\mathbf{x})=c,\quad \text{对于一些正常数 } c>0$$

在定理 1 下，条件

$$\frac{\mathrm{d}}{\mathrm{d}t}V(\mathbf{x})\leqslant 0$$

意味着一旦轨迹对于某一正常数 c 经过 Lyapunov 平面，轨迹将移入一些点定义的集合

$$\mathbf{x}\in\mathbb{R}^N,\quad \text{给定 } V(\mathbf{x})\leqslant 0$$

并且不会再跑出 Lyapunov 平面。在这个意义上我们说在定理 1 下系统是稳定的。

另一方面，在定理 2 下，条件

$$\frac{\mathrm{d}}{\mathrm{d}t}V(\mathbf{x})<0$$

意味着轨迹将从一个 Lyapunov 平面移入一个具有更小常数 c 的内部的 Lyapunov 平面，如图

13.6 所示，特别地，伴随着常数 c 的值减少，Lyapunov 平面以相应的方式向平衡状态 $\bar{\mathbf{x}}$ 靠近，这点暗示着随着 t 的前进轨迹接近平衡状态 $\bar{\mathbf{x}}$。但是我们不能肯定随着 $t\to\infty$，轨迹将真正收敛到 $\bar{\mathbf{x}}$。虽然如此，我们能得出结论：在此严格意义上平衡状态 $\bar{\mathbf{x}}$ 是稳定的，即轨迹被包含在任何具有某一小半径 ε 的球体中，要求初始条件 $\mathbf{x}(0)$ 位于包含在一个球中的 Lyapunov 平面内（Khalil，1992）。另外，这个条件是我们在 8.5 节中提到的有关最大特征过滤渐进稳定性的条件。

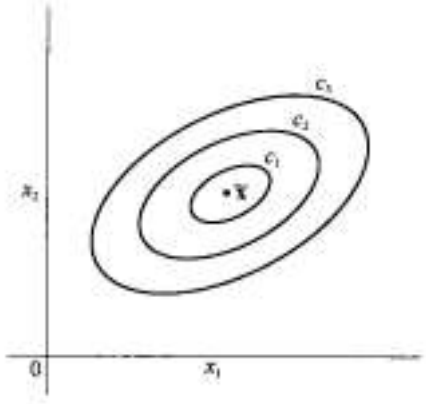


图 13.6　随着常数 c 减少的 Lyapunov 平面，$c_1<c_2<c_3$，平衡状态用点 $\bar{\mathbf{x}}$ 表示

13.4　吸引子

耗散系统一般可以用存在吸引集或者比状态空间维数低的流形来表征。流形的概念在第 7 章讨论过。简单地说，“流形”是指嵌入在 N 维状态空间中的一个 k 维曲面，它由方程组

$$M_j(x_1,x_2,\cdots,x_N)=0,\quad \begin{cases} j=1,2,\cdots,k \\ k<N \end{cases} \tag{13.13}$$

定义，其中 $x_1,x_2,\cdots,x_N$ 是系统 N 维状态向量的元素，M_j 是这些元素的一个函数。这些流形称为吸引子[3]，这是因为吸引子为有界子集，初始条件为非零状态空间体积的区域随时间增加而收敛到它们。

流形可以是状态空间中的一个点，这种情况叫做点吸引子。另外，它也可以是周期性轨道，这种情况叫做稳定的极限环，稳定意味着附近的轨线渐近地趋近它。图 13.7 描绘了这两种类型的吸引子。吸引子只代表动态系统中的平衡状态，它们可以通过用实验方法观察到。但是，注意在吸引子的情况下，平衡状态（equilibrium）既不意味着一个静态平衡（static equilibrium），也不意味一个定常状态（steady state）。例如，一个极限环代表一个吸引子的稳定状态（stable state），但是它随时间连续变化。

在图 13.7 中，我们注意到每个吸引子由它自己独有的区域包围。这样的区域叫做吸引盆（域）（basin（domain）of attraction），同时注意系统的每个初始状态都在某一吸引子的盆中。分隔不同吸引盆的边界叫做分界线（separatrix）。图 13.7 中盆的边界由轨线 T_1、鞍点 Q 和轨线 T_2 的并表示。

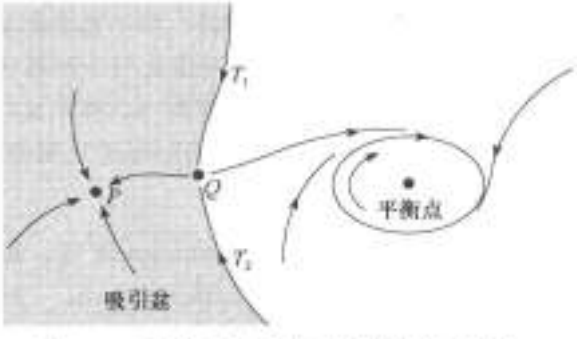


图 13.7　吸引盆概念和分界线思想的说明图

极限环组成非线性系统的平衡点变得不稳定时出现的振荡行为的典型形式。因此，它可能出现在任意阶的系统中。虽然如此，极限环是二阶系统特殊的特征。

双曲吸引子

考虑一个点吸引子，它的非线性动态方程在平衡状态 $\bar{\mathbf{x}}$ 附近被线性化，如 13.2 节中描述的方式。令 $\mathbf{A}$ 表示系统在 $\mathbf{x}=\bar{\mathbf{x}}$ 处计算出的 Jacobi 矩阵。如果 A 所有特征值的绝对值都小于 1，则吸引子是双曲吸引子（hyperbolic attractor）（Ott，1993）。例如，二阶双曲吸引子的流可以为图 13.4a 或者 13.4b 中所显示的形式；两种情况下 Jacobi 矩阵 $\mathbf{A}$ 的特征值都有负实数部分。双曲吸引子在称为“消除梯度问题”的研究中受到特别的关注，这种问题出现在动态驱动的递

归网络中；这一问题将在第 15 章讨论。

13.5　神经动态模型

对非线性动态系统的性能有所了解之后，准备在本节和下一节探讨一下神经动力学所包含的一些重要问题。我们要强调的是，对于神经动力学还没有一个被普遍认可的定义。我们也不是要给出这样一个定义，而是将定义本章中所考虑的神经动力学最普遍的属性。特别地，讨论将局限于状态变量是连续的并且运动方程由微分方程或差分方程描述的神经动态系统。受关注的系统具有四个普遍特性（Peretto and Niez，1986；Pineda，1988a）：

1. *大量自由度*。大脑皮层是高度并行的分布式系统，据估计约有 100 亿个神经元，每个神经元用一个或更多状态变量来描述。据信这样一个神经动力学系统的计算能力和容错能力是系统的集体动力学的结果。系统可以表征为大量的由每个突触连接的强度（效能（effficacy））表示的耦合常量。

2. *非线性性*。神经动力学系统是非线性的。事实上，非线性是建立通用计算机器的基础。

3. *耗散性*。神经动力学系统是耗散的。因此，它由随时间状态空间的收敛性所表征，这个空间在维数较低的空间上。

4. *噪声*。最后，噪声是神经动态系统内在特征。在实际神经元中，膜噪声在突触连接处产生（Katz，1966）。

噪声的存在需要对神经元行为利用概率处理，这给分析神经动力学系统增加了另一层次上的复杂性。对随机神经动力学的详细处理超出本书的范围。因此，以后的材料中均忽略噪声的影响。

加性模型

考虑图 13.8 所示的神经元的无噪声动态模型，其数学基础已在 13 章讨论过了。使用物理术语来说，突触权值 $w_{j1},w_{j2},\cdots,w_{jN}$ 表示传导系数，各自的输入 $x_1(t),x_2(t),\cdots,x_N(t)$ 表示电压，N 是输入数量。这些输入被用于有如下特点的电流求和连接上：

- 低输入阻抗
- 单位电流增益
- 高输出阻抗

因此对输入电流来说，它扮演求和节点的角色。图 13.8 中非线性元素（激活函数）流向输入节点的总电流流量为：

$$\sum_{i=1}^{N} w_{ji}x_i(t) + I_j$$

其中第一项（求和项）是由于刺激 $x_1(t),x_2(t),\cdots,x_N(t)$ 分别作用在突触权值（传导系数）$w_{j1},w_{j2},\cdots,w_{jN}$ 上，第二项是由于电流源 I_j 代表额外施加的偏置。令 $v_j(t)$ 表示非线性激活函数 $\varphi(\cdot)$ 输入处的诱导局部域。因此可以表示从非线性元素的输入节点流出的总电流量为两项的和：

$$\frac{v_j(t)}{R_j} + C_j\frac{\mathrm{d}v_j(t)}{\mathrm{d}t}$$

其中第一项是由于漏泄阻抗 R_j，第二项是由于漏泄电容 C_j。根据 Kirchoff 电流定律，我们知道电路中流向任何节点的总电流流量为零。通过应用 Krichoff 电流定律于图 13.8 中的非线性输入节点，得到

$$C_j\frac{\mathrm{d}v_j(t)}{\mathrm{d}t} + \frac{v_j(t)}{R_j} = \sum_{i=1}^{N} w_{ji}x_i(t) + I_j \tag{13.14}$$

式(13.14)左端的电容项 $C_j \mathrm{d}v_j(t)/\mathrm{d}t$ 是在神经元模型上添加动力学（记忆）的最简单的途径。给定诱导局部域 $v_j(t)$，可以通过使用非线性关系：

$$x_j(t) = \varphi(v_j(t)) \tag{13.15}$$

来确定神经元 j 的输出。由式(13.14)描述的 RC 模型通常称为*加性模型*；这一术语用于区别本模型 w_{ji} 和依赖于 x_i 的乘法（或并联）模型。

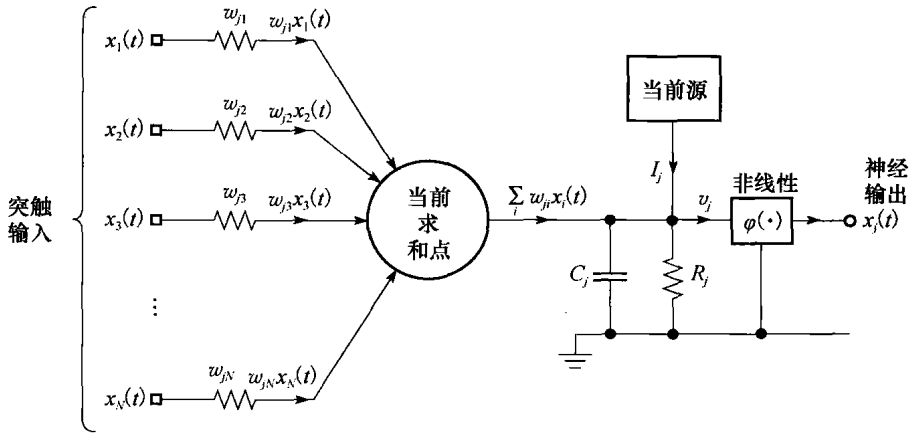


图 13.8　神经元的加性模型

由式(13.14)描述的加性模型的一个显著特性就是相邻神经元 i 施加在神经元 j 上的信号 $x_i(t)$是随时间 t 缓慢改变的。因此描述的模型组成*传统神经动力学的基础*[4]。

继续考虑一个包含 N 个互相连接的神经元的*递归网络*，假设其中每一个神经元都有与式(13.14)和式(13.15)同样的数学模型。那么，忽略神经元内部时间传播的延迟，我们可以用*联立的一阶微分方程组的系统*

$$C_j \frac{\mathrm{d}v_j(t)}{\mathrm{d}t} = -\frac{v_j(t)}{R_j} + \sum_{i=1}^{N} w_{ji}x_i(t) + I_j, \quad j = 1,2,\cdots,N \tag{13.16}$$

定义网络的动力学，它和状态方程(13.1)有同样的数学形式，并且是式(13.14)中各项的简单再排列。假设与神经元 j 的输出 $x_j(t)$相关的激活函数 $\varphi(\cdot)$对它的诱导局部域 $v_j(t)$来说是连续和对时间 t 是可微的函数。普遍使用的激活函数是 logistic 函数

$$\varphi(v_j) = \frac{1}{1+\exp(-v_j)}, \quad j = 1,2,\cdots,N \tag{13.17}$$

13.6 节至 13.11 节中描述的学习算法存在的必要条件在于由式(13.15)和式(13.16)描述的递归网络具有固定点（即点吸引子）。

相关模型

为了简化说明，我们假设式(13.16)中神经元 j 的时间常数 $\tau_j = R_jC_j$ 对所有的 j 都相同。那么，通过关于这一时间常数的公共值归一化时间 t，关于 R_j 归一化 w_{ji} 和 I_j，我们可以重新构造式 (13.16) 的模型以如下简单形式：

$$\frac{\mathrm{d}v_j(t)}{\mathrm{d}t} = -v_j(t) + \sum_i w_{ji}\varphi(v_i(t)) + I_j, \quad j = 1,2,\cdots,N \tag{13.18}$$

其中我们也并入了式(13.15)。联立一阶非线性微分方程组(13.18)的吸引子结构和以下描述的紧密相关模型的吸引子结构基本上相同（Pineda，1987）：

$$\frac{\mathrm{d}x_j(t)}{\mathrm{d}t} = -x_j(t) + \varphi\Big(\sum_i w_{ji}x_i(t)\Big) + K_j, \quad j = 1,2,\cdots,N \tag{13.19}$$

由式(13.18)描述的加性模型中，独立神经元的诱导局部域 $v_1(t), v_2(t), \cdots, v_N(t)$构成状态向

量。另一方面，在由式(13.19)描述的相关模型中，神经元的输出 $x_1(t), x_2(t), \cdots, x_N(t)$ 构成状态向量。

这两种神经动力学模型事实上通过线性的可逆变换是相关的。具体地讲，通过在式(13.19)两侧同乘以 w_{kj}，对 j 求和，然后用变换

$$v_k(t) = \sum_j w_{kj} x_j(t)$$

进行替换，得到一个由式(13.18)所描述的类型的模型，并且由此发现两个模型的偏置项由

$$I_k = \sum_j w_{kj} K_j$$

相关联。这里的重要之处是注意与式(13.18)的加性模型的稳定性相关的结果也适用于与式(13.19)相关的模型。

对于式(13.18)和式(13.19)的神经动力学模型的框图工具的说明，可以参考习题 13.2。

13.6　作为递归网络范例的吸引子操作

当神经元数量 N 非常大的时候，除去噪声的影响，式（13.16）描述的神经动力学模型具有 13.5 节中概述的普遍特性：大量的自由度、非线性性和耗散性。因而，这样一个神经动力学模型可能拥有复杂的吸引子结构，并因此展示出有用的计算能力。

确认具有计算对象（如联想记忆、输入-输出映射器）的吸引子是神经网络范例的一个基础。为了实现这一思想，我们必须训练控制吸引子在系统状态空间中的位置。于是为了以希望的形式编码信息或者学习感兴趣的时间结构，学习算法采用了非线性动力学方程的形式来操纵吸引子在状态空间的位置。通过这一途径，在机器的物理性能和计算的算法之间建立紧密的联系是可能的。

利用神经网络的集体属性实现计算任务的一种途径就是经由能量最小化的概念。在 13.7 节和 13.9 节中将分别考虑的 Hopfield 网络和盒中脑状态模型是这种方法著名的例子。这两种模型都是能量最小化网络；它们的不同之处在于应用领域不同。Hopfield 网络作为按内容寻址存储或者用于解决组合类型最优化问题的模拟计算机是有用的。另一方面，盒中脑状态模型对于聚类类型的应用是有用的。本章后面几节将对这些应用进行说明。

Hopfield 网络和盒中脑状态模型是不含隐藏神经元的联想记忆的实例：联想记忆是智能行为的一个重要来源。另一个神经动力学模型是输入输出映射器类型的，它的运行依赖于隐藏神经元的可用性。在这后一种情况中，最速下降方法经常被用于最小化根据网络参数定义的代价函数，并因此改变吸引子的位置。第 15 章中讨论的动态驱动的递归网络可以作为这后一种神经动力学模型的应用的例证。

13.7　Hopfield 模型

如图 13.9 中描绘的那样，Hopfield 网络（模型）包含一组神经元和一组相应的单位延迟，构成一个多回路反馈系统。反馈回路的数量等于神经元数量。基本上，每个神经元的输出都通过一个单位延迟元素被反馈到网络中另外的每一个神经元。换句话说，网络中没有自反馈；避免使用自反馈的原因将在后面解释。

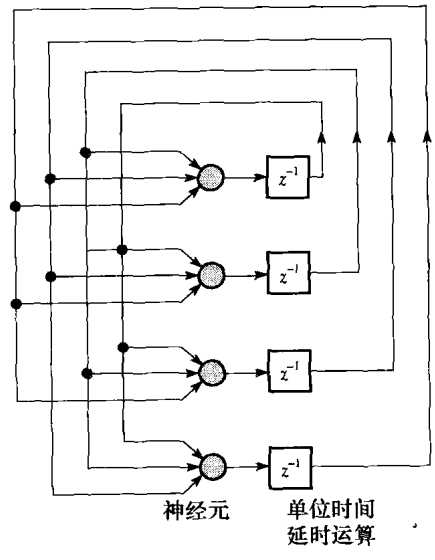


图 13.9　有 4 个神经元的 Hopfield 网络结构图

为了研究 Hopfield 网络的动力学，我们使用式(13.16)

描述的基于神经元加性模型的神经动力学模型。

认识到 $x_i(t)=\varphi_i(v_i(t))$之后，我们可以把式(13.16)改写成以下形式：

$$C_j\frac{\mathrm{d}}{\mathrm{d}t}v_j(t)=-\frac{v_j(t)}{R_j}+\sum_{i=1}^{N}w_{ji}\varphi_i(v_i(t))+I_j,\quad j=1,\cdots,N \tag{13.20}$$

为了继续讨论，我们作出以下假定：

1. 突触权值矩阵是对称的，表示为：

$$w_{ji}=w_{ij},\quad \text{对于所有的 } i \text{ 和 } j \tag{13.21}$$

2. 每个神经元有它自己的非线性激活函数——因此在式(13.20)中使用 $\varphi_i(\cdot)$。

3. 非线性激活函数可逆，因此可以写成：

$$v=\varphi_i^{-1}(x) \tag{13.22}$$

令 sigmoid 函数 $\varphi_i(v)$由双曲线正切函数定义：

$$x=\varphi_i(v)=\tanh\left(\frac{a_iv}{2}\right)=\frac{1-\exp(-a_iv)}{1+\exp(-a_iv)} \tag{13.23}$$

在原点处斜率为 $a_i/2$，表示为

$$\frac{a_i}{2}=\frac{\mathrm{d}\varphi_i}{\mathrm{d}v}\bigg|_{v=0} \tag{13.24}$$

此后我们将把 a_i 称为神经元 i 的增益。

在式(13.23)的 sigmoid 函数的基础上，式(13.22)的逆输出-输入关系可以写成：

$$v=\varphi_i^{-1}(x)=-\frac{1}{a_i}\log\left(\frac{1-x}{1+x}\right) \tag{13.25}$$

一个单位增益神经元的逆输出-输入关系的标准形式定义为：

$$\varphi^{-1}(x)=-\log\left(\frac{1-x}{1+x}\right) \tag{13.26}$$

按照这一标准关系可以把式(13.25)改写为：

$$\varphi_i^{-1}(x)=\frac{1}{a_i}\varphi^{-1}(x) \tag{13.27}$$

图 13.10a 显示标准 sigmoid 的非线性函数 $\varphi(v)$的曲线，图 13.10b 显示相应的非线性反函数 $\varphi^{-1}(x)$的曲线。

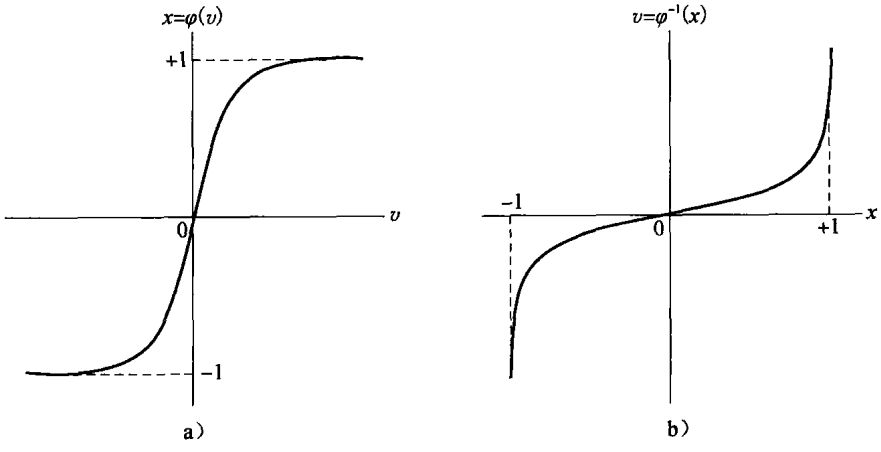


图 13.10　a) 标准的 sigmoid 非线性图；b) 它的逆定义

图 13.9 中的 Hopfield 网络的能量（Lyapunov）函数定义为：

$$E=-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}w_{ji}x_ix_j+\sum_{j=1}^{N}\frac{1}{R_j}\int_0^{x_j}\varphi_j^{-1}(x)\mathrm{d}x-\sum_{j=1}^{N}I_jx_j \tag{13.28}$$

由式(13.28)定义的能量函数 E 为可能具有很多极小点的复杂图像。网络的动力学由寻找那些极小点的机制描述。

有了最小化在心中，求 E 对时间 t 的微分，得到：

$$\frac{\mathrm{d}E}{\mathrm{d}t}=-\sum_{j=1}^{N}\left(\sum_{i=1}^{N}w_{ji}x_i-\frac{v_j}{R_j}+I_j\right)\frac{\mathrm{d}x_j}{\mathrm{d}t} \tag{13.29}$$

由于式(13.20)，式(13.29)右端圆括号内的值被认为是 $C_j\mathrm{d}v_j(t)/\mathrm{d}t$。于是可以把式(13.29)简化为：

$$\frac{\mathrm{d}E}{\mathrm{d}t}=-\sum_{j=1}^{N}C_j\left(\frac{\mathrm{d}v_j}{\mathrm{d}t}\right)\frac{\mathrm{d}x_j}{\mathrm{d}t} \tag{13.30}$$

现在考虑由 x_j 定义的 v_j 的逆关系。将式(13.22)代入式(13.30)，得到：

$$\frac{\mathrm{d}E}{\mathrm{d}t}=-\sum_{j=1}^{N}C_j\left[\frac{\mathrm{d}}{\mathrm{d}t}\varphi_j^{-1}(x_j)\right]\frac{\mathrm{d}x_j}{\mathrm{d}t}=-\sum_{j=1}^{N}C_j\left(\frac{\mathrm{d}x_j}{\mathrm{d}t}\right)^2\left[\frac{\mathrm{d}}{\mathrm{d}x_j}\varphi_j^{-1}(x_j)\right] \tag{13.31}$$

从图 13.10b 中可以看出逆输出输入关系 $\varphi_j^{-1}(x_j)$ 对输出 x_j 是单调增函数。因此它遵守

$$\frac{\mathrm{d}}{\mathrm{d}x_j}\varphi_j^{-1}(x_j)\geqslant 0,\quad \text{对于所有的 } x_j \tag{13.32}$$

我们也注意到：

$$\left(\frac{\mathrm{d}x_j}{\mathrm{d}t}\right)^2\geqslant 0,\quad \text{对于所有的 } x_j \tag{13.33}$$

因而，所有在式(13.31)右端求和的因子都是非负的。换句话说，对式(13.28)定义的能量函数 E 来说，我们有

$$\frac{\mathrm{d}E}{\mathrm{d}t}\leqslant 0,\quad \text{对于所有的 } t$$

由式(13.28)的定义可以看出函数 E 是有界的。因此，我们可以做出以下两个陈述：

1. 能量函数 E 是连续 Hopfield 模型的 Lyapunov 函数。
2. 根据 Lyapunov 定理 1 模型是稳定的。

换句话说，由非线性一阶微分方程组(13.20)的系统描述的连续 Hopfield 模型的时间演化代表状态空间中的一条轨线，该轨线找出能量（Lyapunov）函数 E 的极小值并在这样的固定点上终止。从式(13.31)也要注意，仅当

$$\frac{\mathrm{d}}{\mathrm{d}t}x_j(t)=0,\quad \text{对于所有的 } j$$

导数 $\mathrm{d}E/\mathrm{d}t$ 变为零。因此可以进一步写出

$$\frac{\mathrm{d}E}{\mathrm{d}t}<0,\quad \text{固定点除外} \tag{13.34}$$

式(13.34)给出了下述定理的基础：

Hopfield 网络的（Lyapunov）能量函数 E 是时间的单调减函数。

因此，Hopfield 网络在 Lyapunov 意义上说是全局渐近稳定的；吸引子固定点是能量函数的极小值，反之亦然。

离散和连续 Hopfield 模型的稳定状态之间的关系

Hopfield 网络可以用连续方式或离散方式运行，依赖于描述神经元所采用的模型。连续模型的运行基于前面描述的加性模型。另一方面，离散模型的运行基于 McCulloch-Pitts 模型。

通过重新定义神经元的输入-输出关系，很容易在连续 Hopfield 模型稳定状态和相应的离散 Hopfield 模型的稳定状态之间建立联系，使得这样的关系满足下面两个简化特性：

1. 神经元的输出有渐近值

$$x_j = \begin{cases} +1 & \text{当 } v_j = \infty \\ -1 & \text{当 } v_j = -\infty \end{cases} \tag{13.35}$$

2. 神经元激活函数的中点在原点处，表示为

$$\varphi_j(0) = 0 \tag{13.36}$$

相应地，可以对所有的 j 设置偏置 I_j 为零。

为了表示连续 Hopfield 模型的能量函数 E，允许神经元有自反回路。另一方面，离散 Hopfield 模型不需要自反回路。因此，可以通过在两种模型中对所有的 j 都设置 $w_{jj}=0$ 来简化讨论。

根据这些观察，可以用如下形式重新定义式(13.28)给出的连续 Hopfield 模型的能量函数：

$$E = -\frac{1}{2}\sum_{\substack{i=1 \\ i\neq j}}^{N}\sum_{j=1}^{N} w_{ji}x_i x_j + \sum_{j=1}^{N}\frac{1}{R_j}\int_0^{x_j}\varphi_j^{-1}(x)\mathrm{d}x \tag{13.37}$$

由式(13.27) 定义反函数 $\varphi_j^{-1}(x)$。于是可以重写式(13.37)的能量函数如下：

$$E = -\frac{1}{2}\sum_{\substack{i=1 \\ i\neq j}}^{N}\sum_{j=1}^{N} w_{ji}x_i x_j + \sum_{j=1}^{N}\frac{1}{a_j R_j}\int_0^{x_j}\varphi^{-1}(x)\mathrm{d}x \tag{13.38}$$

积分

$$\int_0^{x_j}\varphi^{-1}(x)\mathrm{d}x$$

有图 13.11 中显示的标准形式。在 $x_j=0$ 积分值为零，其他情况其值为正。假设在 x_j 接近 ± 1 时其值非常大。但是，如果神经元 j 增益 a_j 变为无穷大（例如 sigmoid 函数的非线性趋于理想的硬限制形式），式(13.38)中的第二项就小得可以忽略不计了。在限制情况下，对所有的 j，当 $a_j=\infty$ 时连续 Hopfield 模型的极大、极小值变成和离散 Hopfield 模型中的对应值相等。在后一情况下，能量（Lyapunov）函数的定义简化为：

$$E = -\frac{1}{2}\sum_{\substack{i=1 \\ i\neq j}}^{N}\sum_{j=1}^{N} w_{ji}x_i x_j \tag{13.39}$$

其中第 j 个神经元状态为 $x_j=\pm 1$。因此，我们得出结论：高增益的、连续的和确定的 Hopfield 模型仅有的稳定点对应于离散随机 Hopfield 模型的稳定点。

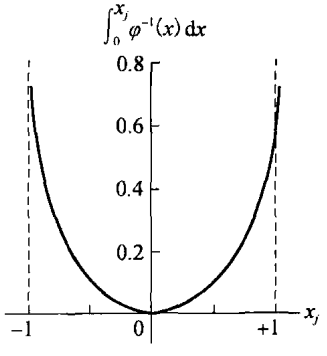


图 13.11　积分 $\int_0^{x_j}\varphi^{-1}(x)\mathrm{d}x$ 的图形

然而，当每一个神经元 j 有很大但是有限的增益 a_j 时，我们发现式(13.38)右端第二项对连续模型的能量函数有明显的贡献。特别地，这一贡献在靠近定义模型状态空间的超立方体的所有面、边和角点处都很大并且为正。而另一方面，该贡献在远离曲面的点处又小得可以忽略。因此，这种模型能量函数的最大值在角点处，但最小值却略微向超立方体的内部偏移。

图 13.12 画出了两个神经元的连续 Hopfield 模型的能量等值线图或能量图。两个神经元的输出定义图中的两个坐标轴。图 13.12 中左下角和右上角代表无穷增益限制情况下的稳定最小值；有限增益情况下的最小值将向内部偏移。流向固定点（即稳定最小值）的流可以解释为式(13.28)定义的能量函数 E 的最小化的解。

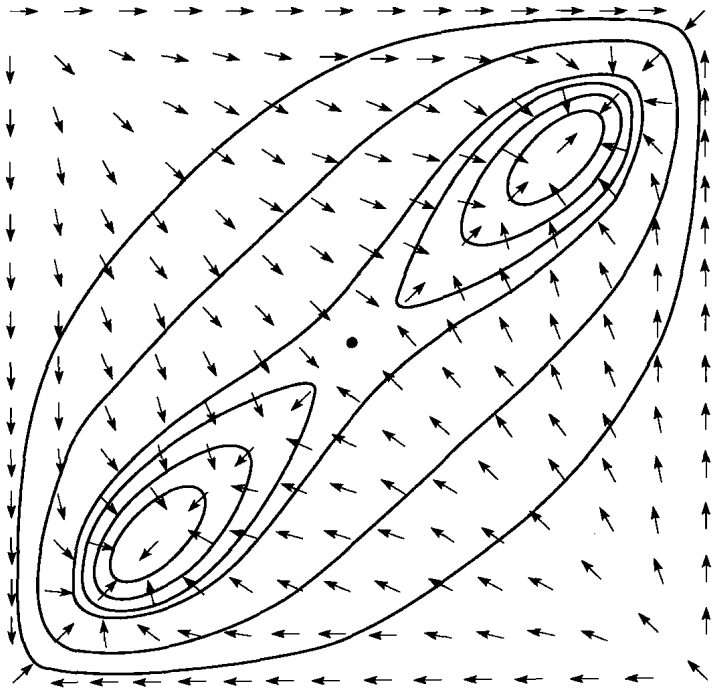


图 13.12　两个神经元的双稳定态系统的能量等值线图。纵轴和横轴为两个神经元的输出。稳定状态位于左下角和右上角，不稳定的极点位于另外两个角。箭头表示状态的移动。移动一般不垂直于能量的等值线图。（经美国国家科学院允许，摘自 J. J. Hopfield，1984）

把离散 Hopfield 网络作为按内容寻址存储器

Hopfield 网络应用于按内容寻址存储器，我们预先知道网络的固定点，它们对应被存储模式。但是，产生期望中固定点的网络突触权值是未知的，因而问题在于如何确定它们。按内容寻址存储器的主要功能是根据模式不完整或有噪声的表示获取存储在存储器中相应模式（项）。为了以简洁方式说明这一陈述的含义，最好的方法就是引用 Hopfield1982 年的论文：

假定存储在存储器中的项是“H. A Kramers & G. H Wannier Physi Rev 60，252 (1941)”，一个普通的按内容寻址存储器，根据足够的部分信息能检索这个完整的存储项。输入“& Wannier，(1941)”可能就足够了。理想的存储器能处理错误并且甚至只输入“Wannier，(1941)”就能检索这一参考文献。

因此，按内容寻址存储器的一个重要属性就是，在给出存储模式的信息内容的一个合理子集的情况下检索该模式的能力。此外，根据提供的线索能够覆盖不一致的信息，在这种意义下按内容寻址存储是可以纠错的。

按内容寻址存储器（CAM）的本质是映射基本存储 ξ_μ 到动态系统的固定点（稳定点）x_μ 上，就像图 13.13 描绘的那样。在数学可以把这个映射表示为

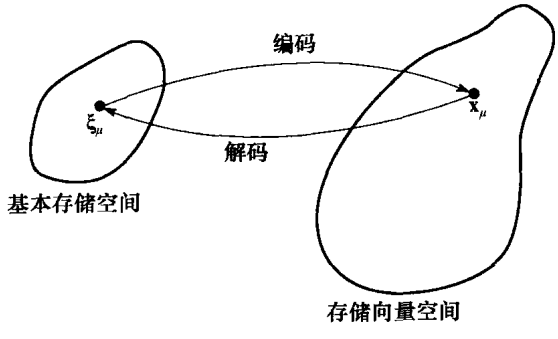


图 13.13　递归网络实现的编码-解码示意图

$$\xi_\mu \rightleftharpoons \mathbf{x}_\mu$$

的形式。从左向右的箭头代表编码操作，而从右向左的箭头代表解码操作。网络状态空间的吸引子固定点为网络的基本记忆或原型状态。假设现在呈现给网络一个模式，这个模式包含基本记忆的部分，但信息是足够的。那么我们可以将该特定模式表示为状态空间中的起点。原则上，倘若该起点靠近表示待检索记忆的固定点（即它位于固定点的吸引盆内部），则系统应该随时间演化并最终收敛于记忆状态本身。在该点上全部的记忆由网络生成。结果 Hopfield 网络有再现（emergent）的性质，该性质帮助它检索信息和处理错误。

在使用 McCulloch and Pitts（1943）的正规神经元作为基本处理单元的 Hopfield 模型中，每一个这样的神经元具有由作用其上的诱导局部域所决定的两个状态。神经元 i 的“开”或“点火”状态用输出值 $x_i=+1$ 表示，而“关”或“静止”状态用 $x_i=-1$ 表示。因此对由 N 个神经元构成的网络来说，网络状态由向量

$$\mathbf{x}=[x_1,x_2,\cdots,x_N]^T$$

定义。由于 $x_i=\pm 1$，神经元 i 的状态表示 1 比特信息，而 $N\times 1$ 的向量 x 表示 N 比特信息的二进制字。

神经元 j 的诱导局部域 v_j 定义为

$$v_j=\sum_{i=1}^{N} w_{ji}x_i+b_j \tag{13.40}$$

其中 b_j 是额外施加在神经元 j 上的固定偏置。因此，神经元 j 根据确定性规则

$$x_j=\begin{cases}+1, & \text{如果 } v_j>0\\ -1, & \text{如果 } v_j<0\end{cases}$$

修改它的状态 x_j。这一关系可以改写为紧凑形式

$$x_j=\operatorname{sgn}(v_j)$$

其中 sgn 是符号函数。如果 v_j 恰好是零会出现什么情况？可采取任意的行动。例如，如果 $v_j=0$，可以设置 $x_j=\pm 1$。然而，我们将使用如下约定：如果 v_j 是 0，神经元 j 保持它原有状态，不管它是开还是关。就像将在后面说明的那样，这一假定的显著意义在于作为结果的流图表是对称的。

把离散 Hopfield 网络作为按内容寻址存储器的操作有两个阶段，即存储阶段和检索阶段，如下所述：

1. *存储阶段*。假设我们希望存储一组表示为 $\{\xi_\mu \mid \mu=1,2,\cdots,M\}$ 的 N 维向量（二进制字）集合。我们称这 M 个向量为基本记忆，表示被网络存储的模式。令 $\xi_{\mu,i}$ 表示基本记忆 ξ_μ 的第 i 个元素，其中类 $\mu=1,2,\cdots,M$。根据存储的外积规则，也就是 Hebb 学习的基本原则的推广，从神经元 i 到神经元 j 的突触权值定义为

$$w_{ji}=\frac{1}{N}\sum_{\mu=1}^{M}\xi_{\mu,j}\xi_{\mu,i} \tag{13.41}$$

使用 $1/N$ 作为比例常数的原因是为了简化信息检索的数学表述。也要注意式(13.41)的学习规则是“单射”（one shot）计算。在 Hopfield 网络正常运行中，设置

$$w_{ii}=0, \quad \text{对于所有 } i \tag{13.42}$$

这意味着神经元没有自反馈。令 W 表示网络 $N\times N$ 的突触权值矩阵，用 w_{ji} 作为它的第 ji 个元素。从而可以把式(13.41)和式(13.42)用矩阵形式组合为如下的等式：

$$\mathbf{W}=\frac{1}{N}\sum_{\mu=1}^{M}\xi_\mu\xi_\mu^T-M\mathbf{I} \tag{13.43}$$

其中 $\xi_\mu\xi_\mu^T$ 表示向量 ξ_μ 和它自身的外积，而 **I** 表示单位矩阵。从这一突触权值和权值矩阵的定义式我们可以重新确认如下事实：

- 网络中每一神经元的输出都反馈到所有的其他神经元上。
- 网络中没有自反馈（即 $w_{ii}=0$）。
- 网络权值矩阵是对称的，表示为（参照式(13.21)）

$$\mathbf{W}^T = \mathbf{W} \tag{13.44}$$

2. *检索阶段*。在检索阶段，一个称为探针（probe）的 N 维向量 ξ_{probe} 被强加于 Hopfield 网络作为它的状态。探针向量的元素为 ± 1。它典型地表征网络中基本记忆的不完整或噪声形式。然后信息检索依照*动态规则*进行，在该规则中网络的每一神经元 j *随机地*但按某一固定比率检测作用在其上的诱导局部域 v_j（包含任意非零偏置 b_j）。如果在某一时刻 v_j 大于零，则神经元 j 将切换它的状态到 $+1$，或者保持在该状态，如果已经是 $+1$。类似地，如果 v_j 小于零，则神经元 j 将切换它的状态到 -1，或者保持在该状态，如果已经是 -1。如果 v_j 恰好为零，则不管是开还是关，神经元 j 都将保持原有状态。因此，从一个迭代到另一个迭代的状态更新是确定的，但是选择进行更新操作的神经元则是随机的。这里描述的异步（串行）更新过程继续下去直到没有任何进一步的变化可以报告为止。那就是说，用探针向量 ξ_{probe} 开始，最终网络生成一个不随时间改变的状态向量 y，它的每个元素都满足*稳定性条件*

$$y_j = \operatorname{sgn}\Big(\sum_{i=1}^{N} w_{ji}y_i + b_j\Big), \quad j = 1,2,\cdots,N \tag{13.45}$$

或者其矩阵形式

$$\mathbf{y} = \operatorname{sgn}(\mathbf{W}\mathbf{y} + \mathbf{b}) \tag{13.46}$$

其中 **W** 是网络突触权值矩阵，**b** 是外部施加的偏置向量。这里描述的稳定性条件也称为对齐（alignment）条件。满足条件的状态向量 **y** 称为系统状态空间的*稳定状态*或*固定点*。因此我们可以作这样的陈述，当检索操作异步进行时，Hopfield 网络将肯定收敛于一稳定状态[5]。

表 13.2 提出对 Hopfield 网络操作包括存储阶段和检索阶段的步骤的一个小结。

表 13.2　Hopfield 模型小结

1. *学习*。令 $\xi_1,\xi_2,\cdots,\xi_M$ 表示已知 N 维基本记忆的集合。使用外积规则（即 Hebb 学习的基本原则）计算网络的突触权值：

$$w_{ji} = \begin{cases} \dfrac{1}{N}\displaystyle\sum_{\mu=1}^{M}\xi_{\mu,j}\xi_{\mu,i}, & j \neq i \\ 0, & j = i \end{cases}$$

其中 w_{ji} 为从神经元 i 到神经元 j 的突触权值。向量 ξ_μ 的元素等于 ± 1。一旦它们被计算出，则突触权值保持不变。

2. *初始化*。令 ξ_{probe} 表示出现在网络中的一未知 N 维输入向量（探针）。通过设置

$$x_j(0) = \xi_{j,\text{probe}}, \quad j = 1,\cdots,N$$

初始化算法，其中 $x_j(0)$ 是神经元 j 在时间 $n=0$ 时的状态，$\xi_{j,\text{probe}}$ 是探针向量 ξ_{probe} 的第 j 个元素

3. *迭代直到收敛*。根据如下规则异步地（即随机并且每次一个）更新状态向量 $\mathbf{x}(n)$ 中的元素：

$$x_j(n+1) = \operatorname{sgn}\Big(\sum_{i=1}^{N} w_{ji}x_i(n)\Big), \quad j = 1,2,\cdots,N$$

重复这一迭代直到状态向量 **x** 保持不变。

4. *输出*。令 $\mathbf{x}_{\text{fixed}}$ 表示第 3 步计算出的固定点（稳定状态）。作为结果的网络输出向量 **y** 为

$$\mathbf{y} = \mathbf{x}_{\text{fixed}}$$

第 1 步是存储阶段，第 2 步到第 4 步构成检索阶段。

例 2　三个神经元的 Hopfield 模型的再现行为

为了说明 Hopfield 模型的再现行为，考虑图 13.14a 所示的三个神经元的网络。网络权值矩阵为

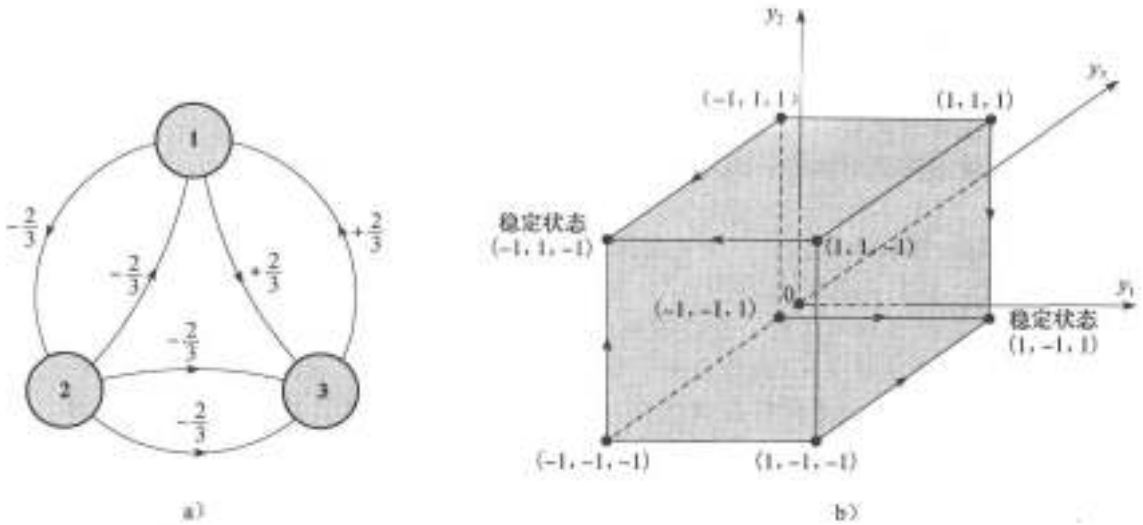


图 13.14　a) $N=3$ 个神经元的 Hopfield 网络结构图；b) 描绘两个稳定态和网络流的图

$$\mathbf{W}=\frac{1}{3}\begin{bmatrix}0 & -2 & +2\\ -2 & 0 & -2\\ +2 & -2 & 0\end{bmatrix}$$

因为它满足式(13.42)和式(13.44)的条件，所以是合法的。假定施加在每个神经元上的偏置为零。由于网络中有三个神经元，所以要考虑的可能状态有 $2^3=8$ 种。这 8 种状态中，只有 $(1,-1,1)$ 和 $(-1,1,-1)$ 这两种状态是稳定的；其余的 6 种状态都是不稳定的。我们说这两种特殊状态是稳定的是因为它们都满足式(13.46)的对齐条件。对状态向量 $(1,-1,1)$，我们有

$$\mathbf{W}\mathbf{y}=\frac{1}{3}\begin{bmatrix}0 & -2 & +2\\ -2 & 0 & -2\\ +2 & -2 & 0\end{bmatrix}\begin{bmatrix}+1\\ -1\\ +1\end{bmatrix}=\frac{1}{3}\begin{bmatrix}+4\\ -4\\ +4\end{bmatrix}$$

硬限制这一结果得到

$$\operatorname{sgn}(\mathbf{W}\mathbf{y})\begin{bmatrix}+1\\ -1\\ +1\end{bmatrix}=\mathbf{y}$$

类似地，对状态向量 $(-1,1,-1)$，我们有

$$\mathbf{W}\mathbf{y}=\frac{1}{3}\begin{bmatrix}0 & -2 & +2\\ -2 & 0 & -2\\ +2 & -2 & 0\end{bmatrix}\begin{bmatrix}-1\\ +1\\ -1\end{bmatrix}=\frac{1}{3}\begin{bmatrix}-4\\ +4\\ -4\end{bmatrix}$$

硬限制这一结果之后，得到

$$\operatorname{sgn}(\mathbf{W}\mathbf{y})\begin{bmatrix}-1\\ +1\\ -1\end{bmatrix}=\mathbf{y}$$

因此，这两种状态向量都满足对齐条件。注意到这两个状态互为相反。

此外，遵从表 13.2 小结的异步更新过程，我们得到图 13.14b 所描绘的流。这个流图展示关于网络中直观上满足条件的两个稳定状态之间的对称性。这种对称性是令作用于其上的诱导局部域恰好为零的神经元保留在原有状态的结果。

图 13.14b 也显示出如果图 13.14a 的网络初始状态是（1,1,1）、（−1,−1,1）或（1,−1,−1），那么在一次迭代之后它将收敛于稳定状态（1,−1,1）。如果初始状态是（−1,−1,−1）、（−1,1,1）或（1,1,−1），则它将收敛于第二个稳定状态（−1,1,−1）。

因此，网络有两个基本记忆（1,−1,1）和（−1,1,−1）表征这两个稳定状态。式(13.43)的应用产生突触权值矩阵

$$\mathbf{W}=\frac{1}{3}\begin{bmatrix}+1\\-1\\+1\end{bmatrix}[+1,-1,+1]+\frac{1}{3}\begin{bmatrix}-1\\+1\\-1\end{bmatrix}[-1,+1,-1]-\frac{2}{3}\begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix}=\frac{1}{3}\begin{bmatrix}0&-2&+2\\-2&0&-2\\+2&-2&0\end{bmatrix}$$

它和图 13.14a 所示的突触权值符合。

通过检验图 13.14b 的流图，Hopfield 网络的纠错能力是显而易见的：

1. 如果作用在网络上的探针向量 ξ_{probe} 等于（−1,−1,1）、（1,1,1）或（1,−1,−1），则作为结果的输出是基本记忆（1,−1,1）。每个这样的探针的值表示一个和存储模式相比的单一错误。

2. 如果探针向量 ξ_{probe} 等于（1,1,−1）、（−1,−1,−1）或（−1,1,1），则作为结果的输出是基本记忆（−1,1,−1）。这里再次表明，每个这样的探针表示一个和存储模式相比的单一错误。

伪状态

就像式(13.44)指出的那样，离散 Hopfield 网络的权值矩阵 **W** 是对称的。因此 **W** 的特征值都是实数。然而，当 M 很大的时候特征值通常是退化的（degenerate），这意味着有几个特征向量有同样的特征值。通过退化特征值联系的几个特征向量构成了一个子空间。此外，权值矩阵 **W** 退化特征值有等于零的，这种情况下的子空间叫做零空间。零空间的存在是由于基本记忆的数量 **M** 小于网络中神经元数量 N 的事实。零空间的出现是 Hopfield 网络的内在特性。

权值矩阵 **W** 的特征分析，使得我们对把离散 Hopfield 网络作为按内容寻址存储器支持下列观点（Aiyer 等，1990）：

1. 离散 Hopfield 网络将探针向量投影到被基本记忆向量扩张成的子空间$\mathcal{U}$上，从这种意义上说，它起到向量投影器的作用。

2. 网络固有的动力学把结果投影向量驱动到单位超立方体的能量函数最小的一个角点处。

单位超立方体是 N 维的。扩张成子空间的 M 个基本记忆向量组成由单位超立方体确定的角点表示的固定点（稳定状态）的集合。单位超立方体的其他位于子空间内部或附近的角点是潜在伪状态（spurious states）的所在位置，也称为伪吸引子。伪状态表示 Hopfield 网络中不同于网络基本记忆的其他稳定状态。

因此，在设计作为按内容寻址存储器的 Hopfield 网络过程中，我们面临着对两个矛盾需求的权衡：

- 需要在状态空间中保持基本记忆向量作为固定点。
- 希望有少量的伪状态。

不幸的是，Hopfield 网络的基本记忆不总是稳定的。而且，可能出现由伪状态表征的不同于基本记忆的其他稳定状态。这两种现象倾向于降低作为按内容寻址存储器 Hopfield 网络的效率。

13.8　Cohen-Grossberg 定理

在 Cohen and Grossberg（1983），给出评价一类神经网络的稳定性的一般原则：由如下联

立非线性微分方程组描述

$$\frac{\mathrm{d}}{\mathrm{d}t}u_j = a_j(u_j)\Big[b_j(u_j) - \sum_{i=1}^{N} c_{ji}\varphi_i(u_i)\Big], \quad j = 1,\cdots,N \tag{13.47}$$

这类神经网络允许定义一 Lyapunov 函数

$$E = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} c_{ji}\varphi_i(u_i)\varphi_j(u_j) - \sum_{j=1}^{N}\int_0^{u_j} b_j(\lambda)\varphi_j'(\lambda)\mathrm{d}\lambda \tag{13.48}$$

其中 $\varphi_j'(\lambda)$是 $\varphi_j(\lambda)$相应于 λ 的导数。为了使式(13.48)的定义有效，需要下面三个条件成立：

1. 网络的突触权值对称：

$$c_{ij} = c_{ji} \tag{13.49}$$

2. $a_j(u_j)$满足非负性条件：

$$a_j(u_j) \geqslant 0 \tag{13.50}$$

3. 非线性输入-输出函数 $\varphi_j(u_j)$满足单调性条件：

$$\varphi_j'(u_j) = \frac{\mathrm{d}}{\mathrm{d}u_j}\varphi_j(u_j) \geqslant 0 \tag{13.51}$$

有了这些基础，我们可以正式地陈述 Cohen-Grossberg 定理：

如果非线性微分方程组 (13.47) 满足对称性、非负性和单调性，则由式(13.48)描述的 Lyapunov 函数 E 满足条件

$$\frac{\mathrm{d}E}{\mathrm{d}t} \leqslant 0$$

一旦 Lyapunov 函数 E 的基本属性具备，系统的全局稳定性从 Lyapunov 定理 1 推出。

Hopfield 模型作为 Cohen-Grossberg 定理的特例

对一个连续的 Hopfield 模型，通过比较式(13.47)和式(13.20)，我们可以得到 Hopfield 模型和 Cohen-Grossberg 定理之间的对应关系，这种关系如表 13.3 所示。在式(13.48)中运用此表，就可以得到连续的 Hopfield 模型的 Lyapunov 函数：

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ji}\varphi_i(v_i)\varphi_j(v_j) + \sum_{j=1}^{N}\int_0^{v_j}\Big(\frac{v_j}{R_j} - I_j\Big)\varphi_j'(v)\mathrm{d}v \tag{13.52}$$

其中非线性激活函数 $\varphi_j(\cdot)$ 由式(13.23)定义。

表 13.3　Cohen-Grossberg 定理和 Hopfield 模型的对应关系

Cohen-Grossberg 定理	Hopfield 模型	Cohen-Grossberg 定理	Hopfield 模型
u_j	$C_j v_j$	$a_j(u_j)$	1
$b_j(u_j)$	$-(v_j/R_j)+I_j$	c_{ji}	$-w_{ji}$
$\varphi_i(u_i)$	$\varphi_i(v_i)$		

接下来，我们得到如下的观察结果：

1. $\varphi_i(v_i) = x_i$

2. $\int_0^{v_j}\varphi_j'(v)\mathrm{d}v = \int_0^{x_j}\mathrm{d}x = x_j$

3. $\int_0^{v_j} v\varphi_j'(v)\mathrm{d}v = \int_0^{x_j} v\mathrm{d}x = \int_0^{x_j}\varphi_j^{-1}(x)\mathrm{d}x$

基本地，关系式 2 和 3 通过应用 $x=\varphi_i(v)$得到。这样，在式(13.52)的 Lyapunov 函数中运用这些观察就可以得到和我们早先描述的相同的结果；参看式(13.28)。然而，尽管 $\varphi_i(v)$必须是输入 v 的非减函数，但为使式(13.52)描述的通用 Lyapunov 函数成立，并不需要具有逆。

Cohen-Grossberg 定理是有广泛应用的神经动力学的一个基本原理 (Grossberg，1990)。

在下一节我们考虑这个重要定理的另一个应用。

13.9　盒中脑状态模型

在这一节中，我们通过学习盒中脑状态（brain state in a box，BSB）模型来继续联想记忆的神经动力学的分析。该模型首先由 Anderson 等（1977）描述。BSB 模型基本上是一个带幅度限制的正反馈系统，该模型是由一组反馈回自身的高度互连的神经元组成。模型用内置的正反馈来放大输入模式，直到模型中的所有神经元饱和。这样，BSB 模型可以看作一个分类器，在该分类器中，给定一个模拟输入模式，产生一个由模型稳定状态描述的数字表示。

用 $\mathbf{W}$ 表示对称权值矩阵，该矩阵的最大特征值为正实数。用 $\mathbf{x}(0)$ 表示模型的初始状态向量，代表输入激活模式。假定模型中有 N 个神经元。模型的状态向量是 N 维的，$\mathbf{W}$ 是 $N\times N$ 矩阵。BSB 算法由下面两个方程完全定义：

$$\mathbf{y}(n)=\mathbf{x}(n)+\beta\mathbf{W}\mathbf{x}(n) \tag{13.53}$$

$$\mathbf{x}(n+1)=\varphi(\mathbf{y}(n)) \tag{13.54}$$

其中 β 是一个称为反馈因子的正的小常数，$\mathbf{x}(n)$ 是模型在时刻 n 的状态向量。图 13.15a 显示式(13.53)和式(13.54)的框图的组合。方框 $\mathbf{W}$ 代表一个单层线性神经网络，如图 13.15b 所示。激活函数 φ 是一个作用在 $y_j(n)$ 上的分段线性函数，$y_j(n)$ 是向量 $\mathbf{y}(n)$ 的第 j 个分量，如下所示（参见图 13.16）：

$$x_j(n+1)=\varphi(y_j(n))=\begin{cases}+1, & \text{如果 } y_j(n)>+1\\ y_j(n), & \text{如果 } -1\leqslant y_j(n)\leqslant+1\\ -1, & \text{如果 } y_j(n)<-1\end{cases} \tag{13.55}$$

式(13.55)限制 BSB 模型的状态向量处于中心在原点的一个 N 维单位立方体中。

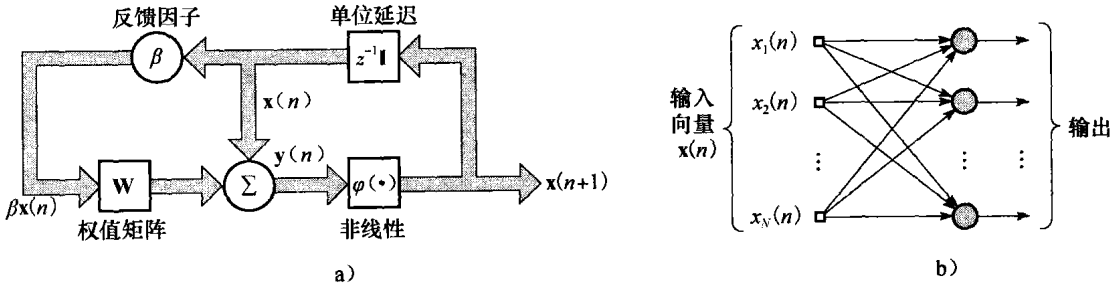


图 13.15　a）盒中脑状态（BSB）模型框图的组合；b）权值矩阵 $\mathbf{W}$ 表示的线性联想器的信号流图

算法如下进行：一个激活模式 $\mathbf{x}(0)$ 作为一个初始状态向量输入 BSB 模型，式(13.53)用来计算向量 $\mathbf{y}(0)$，式(13.54)用来截断 $\mathbf{y}(0)$，获得更新状态向量 $\mathbf{x}(1)$。接着，$\mathbf{x}(1)$ 通过式(13.53)和式(13.54)循环得到 $\mathbf{x}(2)$。这个过程一直重复直到 BSB 模型达到一个稳定状态，该状态代表超立方体的一个角点。直觉上，BSB 模型的正反馈引起初始状态向量 $\mathbf{x}(0)$ 的欧几里得长度（范数）随迭代次数的增加而增加，直到它撞到盒子（单位超立方体）的壁上，然后顺着壁滑行，最终停在盒子的一个稳定角点上，在这里它继续“推进”却不能脱离盒子（Kawamoto and Anderson 1985），这就是该模型名字的由来。

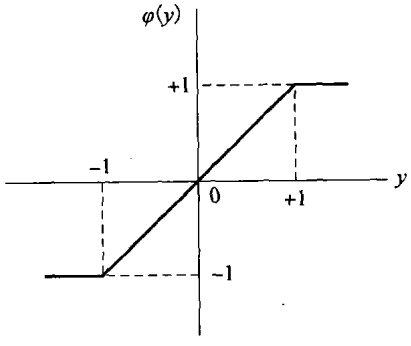


图 13.16　BSB 模型使用的分段线性函数

BSB 模型的 Lyapunov 函数

重新定义 BSB 模型可以作为由式(13.16)描述的神经动力学模型的一个特例（Grossberg，1990）。为了看到这一点，首先以下述形式重写由式(13.53)和式(13.54)描述的 BSB 算法的第 j 个组成部分：

$$x_j(n+1) = \varphi\Big(\sum_{i=1}^{N} c_{ji}x_i(n)\Big), \quad j = 1,2,\cdots,N \tag{13.56}$$

系数 c_{ji} 由

$$c_{ji} = \delta_{ji} + \beta_{w_{ji}} \tag{13.57}$$

定义，其中 δ_{ji} 为 Kronecker delta 函数，仅当 $j=i$ 时为 1，其余情况为 0；w_{ji} 是权矩阵 $\mathbf{W}$ 的第 ji 个元素。式(13.56)是离散的时间形式。为了进一步处理，重新用连续时间形式写出它的公式

$$\frac{\mathrm{d}}{\mathrm{d}t}x_j(t) = -x_j(t) + \varphi\Big(\sum_{i=1}^{N} c_{ji}x_i(t)\Big), \quad j = 1,2,\cdots,N \tag{13.58}$$

其中偏置 I_j 对所有的 j 都为 0。然而，为了应用 Cohen-Grossberg 定理，必须进一步把式(13.58)转换成加性模型的形式。我们可以通过引入一组新变量

$$v_j(t) = \sum_{i=1}^{N} c_{ji}x_i(t) \tag{13.59}$$

来做到这点。然后，通过式(13.57)中 c_{ji} 的定义，发现

$$x_j(t) = \sum_{i=1}^{N} c_{ji}v_i(t) \tag{13.60}$$

相应地，重置式(13.58)的模型为等价形式

$$\frac{\mathrm{d}}{\mathrm{d}t}v_j(t) = -v_j(t) + \sum_{i=1}^{N} c_{ji}\varphi(v_i(t)), \quad j = 1,2,\cdots,N \tag{13.61}$$

现在，我们准备把 Cohen-Grossberg 定理应用到 BSB 模型上。通过比较式(13.61)和式(13.47)，得到如表 13.4 所示的 BSB 模型和 Cohen-Grossberg 定理的对应关系。因此，把表 13.4 的结果用于式(13.48)，就得到 BSB 模型的 Lyapunov 函数：

$$E = -\frac{1}{2}\sum_{j=1}^{N}\sum_{i=1}^{N} c_{ji}\varphi(v_j)\varphi(v_i) + \sum_{j=1}^{N}\int_{0}^{v_j} v\varphi'(v)\mathrm{d}v \tag{13.62}$$

其中 $\varphi'(v)$ 是 sigmoid 函数 $\varphi_i(v)$ 对它的参数的一阶导数。最后，将式(13.55)、式(13.57)和式(13.59)的定义代入式(13.62)，就能用原始状态向量定义 BSB 模型的 Lyapunov（能量）函数如下（Grossberg，1990）：

$$E = -\frac{\beta}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ji}x_jx_i = -\frac{\beta}{2}\mathbf{x}^T\mathbf{W}\mathbf{x} \tag{13.63}$$

表 13.4　Cohen-Grossberg 定理和 BSB 模型的对应关系

Cohen-Grossberg 定理	BSB 模型	Cohen-Grossberg 定理	BSB 模型
u_j	v_j	$a_j(u_j)$	1
$b_j(u_j)$	$-v_j$	c_{ji}	$-c_{ji}$
$\varphi_j(u_j)$	$\varphi_j(v_j)$		

在 13.7 节中对 Hopfield 网络 Lyapunov 函数的估计，假定模型的非线性 sigmoid 函数的逆的导数存在，此条件是通过用一个双曲线正切函数来满足的。相反，在 BSB 模型中，当第 j 个神经元的状态变量是+1 或−1 时，这个条件并不满足。尽管困难重重，BSB 模型的 Lyapunov 函数能通过 Cohen-Grossberg 定理来估计，从而清楚地表明这个重要定理可以普遍应用。

BSB 模型动力学

在由 Golden（1986）进行的直接分析中，说明 BSB 模型实际是一个梯度下降算法，使得由式(13.63)所定义的能量函数 E 达到最小。然而 BSB 模型的这个重要性质要假设权值矩阵 W 满足下面两个条件：

- 权值矩阵 **W** 是对称的，即

$$\mathbf{W} = \mathbf{W}^T$$

- 权值矩阵 **W** 是半正定的；也就是说，关于 **W** 的特征值，我们有

$$\lambda_{\min} \geqslant 0$$

其中 $\lambda_{\min}$ 是 **W** 的最小特征值。

这样，当在时间 $n+1$ 时的状态向量 $\mathbf{x}(n+1)$ 与在时间 n 的状态向量 $\mathbf{x}(n)$ 不同时，BSB 模型的能量函数 E 随 n（迭代次数）的增加而减小。更进一步，能量函数 E 的最小点定义 BSB 模型的平衡状态，模型由

$$\mathbf{x}(n+1) = \mathbf{x}(n)$$

表征。换句话说，像 Hopfield 模型一样，BSB 模型是一个能量最小化网络。

BSB 模型的平衡状态由单位超立方体的特定的角点和它的原点定义。在后一种情况（在原点），状态向量的任何波动，无论是多么小，都被模型中的正反馈放大，因此引起模型从原点向稳定状态漂移；换句话说，原点是一个鞍点。对超立方体来说，要使它的每个角点作为 BSB 模型的平衡状态，权值矩阵 **W** 必须满足第三个条件（Greenberg，1988）：

- 权矩阵 **W** 是对角优势的（dominant），其含义是

$$w_{jj} \geqslant \sum_{i \neq j} |w_{ij}|, \quad \text{当 } j = 1,2,\cdots,N \tag{13.64}$$

其中 w_{ij} 是 **W** 的第 ij 个元素。

为了使平衡状态 **x** 稳定，也就是为了使单位超立方体的一个特定角是一个固定点吸引子（attractor），在单位立方体中必须有一个吸引盆 $\mathcal{N}(\mathbf{x})$，使得对 $\mathcal{N}(\mathbf{x})$ 中的所有初始状态向量 $\mathbf{x}(0)$，BSB 模型都收敛于 **x**。为了使单位超立方体的每一个角点是一个可能的点吸引子，权值矩阵必须满足第四个条件（Greenberg，1988）：

- 权矩阵 **W** 是强对角优势的，表示为

$$w_{jj} \geqslant \sum_{i \neq j} |w_{ij}| + \alpha, \quad \text{当 } j = 1,2,\cdots,N \tag{13.65}$$

其中 α 是一个正的常数。

这里讨论的重点是：如果 BSB 模型的权值矩阵 **W** 只是对称的和正半定的，单位立方体中只有一些（不是所有）角点是点吸引子。为了使单位立方体中的所有角点是潜在的点吸引子，权矩阵 **W** 也必须满足式(13.65)，式(13.65)当然蕴含式(13.64)。

聚类

BSB 模型的一个自然应用是聚类（Anderson，1995）。这是因为单位超立方体的稳定角点作为有吸引盆的点吸引子，会把状态空间划分为相应的明确定义的区域。因此，BSB 模型可以用作一种无监督的聚类算法，其中单位超立方体的每一个稳定角点代表相关数据的一个“聚类”。由正反馈所提供的自放大（符合在第 8 章描述的自组织规则 1）是聚类性质的一个重要成分。

例 3　自联想

对于一个包含两个神经元的 BSB 模型。2×2 权值矩阵 **W** 定义为

$$\mathbf{W} = \begin{bmatrix} 0.035 & -0.005 \\ -0.005 & 0.035 \end{bmatrix}$$

此权矩阵是对称正定的，并满足式(13.65)。

图 13.17 的四个不同部分分别对应初始状态 $\mathbf{x}(0)$的四种不同的赋值，如下所示：

(a) $\mathbf{x}(0) = [0.1, 0.2]^T$

(b) $\mathbf{x}(0) = [-0.2, 0.3]^T$

(c) $\mathbf{x}(0) = [-0.8, -0.4]^T$

(d) $\mathbf{x}(0) = [0.6, 0.1]^T$

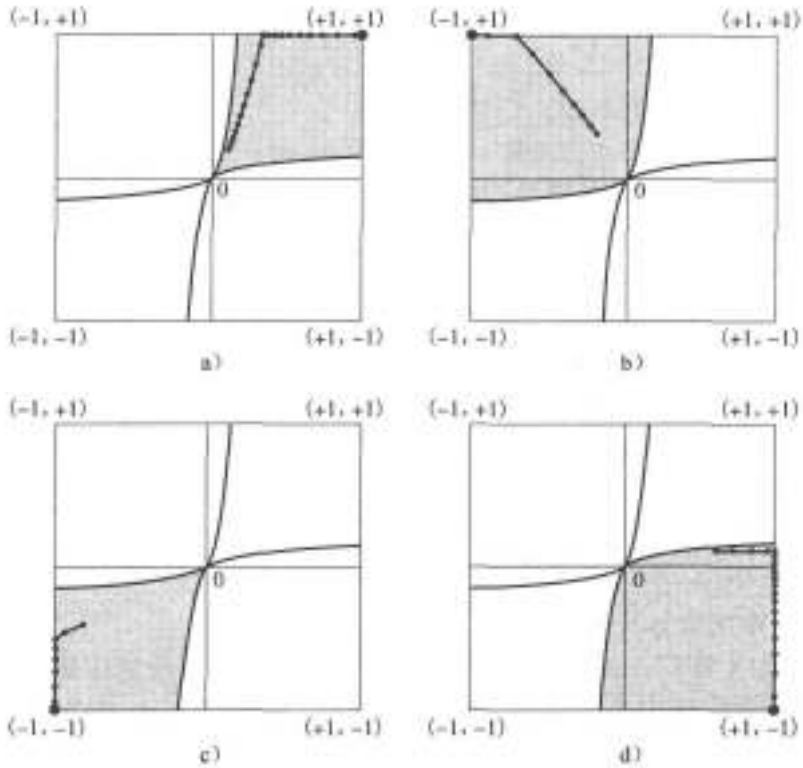


图 13.17　BSB 模型计算机实验的轨线，四种不同初始条件下的操作：

- 四个阴影代表模型的吸引盆
- 相应的模型的轨迹用红线表示
- 四个角落，也就是轨迹终结的地方，用黑色表示

图中阴影区域是标记模型的四个吸引盆。该图清晰地阐明当模型的初始状态在一个特定的吸引盆时，模型固有动力学驱使权值矩阵 $\mathbf{W}(n)$随着迭代次数 n 的增加而增加，直到网络状态 $\mathbf{x}(n)$终止在一个固定点吸引子（即一个 2×2 正方形的角点），此吸引子属于那个吸引盆。特别有趣的是图 d 中的轨迹：初始条件 $\mathbf{x}(0)$在第一象限，然而轨迹在第四象限终止于角点（+1，−1），因为那就是合适的吸引盆中点吸引子所在的地方。

在这个例子中，具有二神经元的 BSB 模型的方块状态空间被完全地分为四个不同的吸引盆；每个盆包括方块的一个角，代表具有最小能量的稳定状态。因此，BSB 模型可以视为自联想网络的例子，是从这个意义上说的，即所有的点都位于其中一个吸引盆，而它们每个都与一个最小能量稳定状态点相关。

13.10　奇异吸引子和混沌

到目前为止，在我们讨论的神经动力学中，集中于由固定点吸引子所刻画的非线性动力学系统的行为。在这一节考虑一种称为奇异吸引子的另一类吸引子，它们刻画阶数高于 2 的某种

非线性动力学系统。

一个奇异吸引子表现出高度复杂的混乱行为。使研究奇异吸引子和混沌特别有趣的是：因为系统运行是由固定规则所支配的，所以系统是确定的。然而这样一个只有少数几个自由度的系统却有如此复杂的行为以至于它看起来是随机的。确实，随机性在以下意义上是基本的：一个混沌（chaos）时间序列的二阶统计性似乎显示它是随机的。然而，不像一个真正的随机现象，一个混沌系统所展示的随机性并不随着收集信息的增加而减少。原则上，一个混沌系统未来的行为完全由它的过去所决定。但实际上，初始条件选择的任何不确定性，无论是多么小，随着时间量指数增加。这样即使一个混沌系统的动态行为在短期内可以预测，却不可能预测系统的长期行为。因此，一个混沌时间序列表现这样一种矛盾：它的产生是由一个确定动态系统支配的，然而它看起来却是随机的。混沌现象的这种属性最初是 Lorenz 在发现一种吸引子时所强调的，并以他的名字命名（Lorenz，1963）。

在一个非线性动态系统中，当吸引子中具有相近初始条件的不同轨迹随着时间增加而逐渐分离时，我们就说系统具有一个奇异吸引子（strange attractor），并且说系统本身是混沌的（chaotic）。换句话说，使得一个吸引子奇异的本质属性是对初始条件的敏感性依赖。这里，敏感性意味着如果两个相同的非线性系统开始于稍有差别的初始条件，即分别为 $\mathbf{x}$ 和 $\mathbf{x}+\boldsymbol{\varepsilon}$，这里 $\boldsymbol{\varepsilon}$ 是一个非常小的量，它们的动态状态在状态空间中会相互散开，并且它们的间隔平均而言将按指数增加。

混沌动力学的不变特征

两个主要特征分数维（fractal dimensions）和 Lyapunov 指数，已经成为一种混沌过程的分类器。分数维刻画一个奇异吸引子的几何结构。术语“分形”（fractal）是由 Mandelbrot（1982）提出的。不像整数维（如二维平面、三维空间），分数维并不是整数。对于 Lyapunov 指数，它们描述吸引子的轨道如何随动态系统的演化而运动。这两个混沌动态系统的不变特征将在下面讨论。术语“不变”表明：一个混沌过程的分数维和 Lyapunov 指数在该过程坐标系统的光滑非线性变换下保持不变。

分数维

考虑一个奇异吸引子，它在 d 维状态空间的动力学由

$$\mathbf{x}(n+1)=\mathbf{F}(\mathbf{x}(n)),\quad n=0,1,2,\cdots \tag{13.66}$$

描述，它是式(13.2)的离散时间形式。通过设置 $t=n\Delta t$，这很容易看出，其中 Δt 是采样周期。假定 Δt 足够小，我们可以相应地设置

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}(t)=\frac{1}{\Delta t}[\mathbf{x}(n\Delta t+\Delta t)-\mathbf{x}(n\Delta t)]$$

这样，我们可以得到式(13.2)的离散时间形式如下：

$$\frac{1}{\Delta t}[\mathbf{x}(n\Delta t+\Delta t)-\mathbf{x}(n\Delta t)]=\mathbf{F}(\mathbf{x}(n\Delta t)),\quad \text{对很小的 } \Delta t$$

为了表示方便，令 $\Delta t=1$ 并对项进行重新排列，得到

$$\mathbf{x}(n+1)=\mathbf{x}(n)+\mathbf{F}(\mathbf{x}(n))$$

它能写成式(13.66)的形式，只要简单地重新定义向量值函数 $\mathbf{F}(\cdot)$ 吸收 $\mathbf{x}(n)$。

回到式(13.66)，假定我们在吸引子的轨道上或附近的一个位置 y 处构造半径为 r 的小球。那么，我们对吸引子可以定义点的自然分布（natural distribution）如下：

$$\rho(\mathbf{y})=\lim_{N\to\infty}\frac{1}{N}\sum_{n=1}^{N}\delta(\mathbf{y}-\mathbf{x}(n)) \tag{13.67}$$

其中 $\delta(\cdot)$ 是 d 维 delta 函数，N 是数据点的个数。注意 N 在用法上的变化。自然分布 $\rho(\mathbf{y})$ 对

一个奇异吸引子扮演的角色就像一个概率密度函数对一个随机变量那样。相应地，我们可以随动态系统演化定义函数 $f(\mathbf{y})$的不变量为多重积分 $\overline{f}$

$$\overline{f}=\int_{-\infty}^{\infty}f(\mathbf{y})\rho(\mathbf{y})\mathrm{d}\mathbf{y} \tag{13.68}$$

一个感兴趣的函数 $f(\mathbf{y})$是使我们能衡量当小球半径 r 趋向于 0 时，小球内的点的数目是如何变化的。注意 d 维球所占的空间体积正比于 r^d，因此，通过观察在状态空间中吸引子上的点的密度在小距离范围内如何变化，我们可以了解吸引子的维数。

球的中心 $\mathbf{y}$ 和在时刻 n 时的点 $\mathbf{x}(n)$之间的欧几里得距离是 $\|\mathbf{y}-\mathbf{x}(n)\|$。因此，如果

$$\|\mathbf{y}-\mathbf{x}(n)\|<r$$

或等价地

$$r-\|\mathbf{y}-\mathbf{x}(n)\|>0$$

则点 $\mathbf{x}(n)$在半径为 r 的球内。因此，在所描述情况下的函数 $f(\mathbf{x})$可以写成一般形式

$$f(\mathbf{x})=\Big(\frac{1}{N-1}\sum_{\substack{k=1\\k\neq n}}^{N}\theta(r-\|\mathbf{y}-\mathbf{x}(k)\|)\Big)^{q-1} \tag{13.69}$$

其中 q 是一个整数，$\theta(\cdot)$ 是由

$$\theta(z)=\begin{cases}1, & \text{当 } z>0\\ 0, & \text{当 } z<0\end{cases}$$

定义的 Heaviside 函数。将式(13.67)和式(13.69)代入式(13.68)，得到一个新的依赖于 q 和 r 的函数 $C(q,r)$，如下所示：

$$C(q,r)=\int_{-\infty}^{\infty}\Big(\frac{1}{N-1}\sum_{\substack{k=1\\k\neq n}}^{N}\theta(r-\|\mathbf{y}-\mathbf{x}(k)\|)\Big)^{q-1}\Big(\frac{1}{N}\sum_{n=1}^{N}\delta(\mathbf{y}-\mathbf{x}(n))\Big)\mathrm{d}\mathbf{y} \tag{13.70}$$

因此，利用 delta 函数的筛选（sifting）性质，也就是对某些函数 $g(\cdot)$ 的关系

$$\int_{-\infty}^{\infty}g(\mathbf{y})\delta(\mathbf{y}-\mathbf{x}(n))\mathrm{d}\mathbf{y}=g(\mathbf{x}(n))$$

并交换求和顺序，可以重新定义函数 $C(q,r)$如下：

$$C(q,r)=\frac{1}{N}\sum_{n=1}^{N}\Big(\frac{1}{N-1}\sum_{\substack{k=1\\k\neq n}}^{N}\theta(r-\|\mathbf{x}(n)-\mathbf{x}(k)\|)\Big)^{q-1} \tag{13.71}$$

函数 $C(q,r)$被称为*相关函数*（correlation function）[6]，用文字的方式定义如下：

吸引子的相关函数用 $C(q,r)$表示，是用来度量吸引子上两点 $\mathbf{x}(n)$和 $\mathbf{x}(k)$对于某一整数 q 以距离 r 隔开的概率。

在式(13.71)的定义中数据点的总数 N 假定很大。

相关函数 $C(q,r)$是吸引子本身的不变量。虽然如此，在实际中我们集中在 r 很小时 $C(q,r)$的行为。这个极限行为由

$$C(q,r)\approx r^{(q-1)D_q} \tag{13.72}$$

描述，其中 D_q 称为吸引子的*分数维*，假定它是存在的。在式(13.72)两边取对数，得到 D_q 的正式定义

$$D_q=\lim_{r\to 0}\frac{\log C(q,r)}{(q-1)\log r} \tag{13.73}$$

然而，由于通常仅有有限的数据点，半径 r 必须恰好足够小，使得有足够的点落在球内。对一个给定的 q，可以根据 $C(q,r)$作为 $\log r$ 的线性函数的斜率确定分数维 D_q。

对 $q=2$，分数维 D_q 的定义具有一个适宜于可靠计算的简单形式。所得维数 D_2 被称为吸

引子的相关维数（correlation dimension）（Grassberger and Procaccia，1983）。相关维数反映固有动态系统的复杂性，并且限定描述该系统所需的自由度。

Lyapunov 指数

Lyapunov 指数是描述吸引子未来状态不确定性的统计量。更具体地，它们量化在移向吸引子时邻近轨道相互分离的指数速度。假定 $\mathbf{x}(0)$是初始条件，$\{\mathbf{x}(n), n=0,1,2\cdots\}$ 是相应的轨道。考虑从初始条件 $\mathbf{x}(0)$向和轨道相切的向量 $\mathbf{y}(0)$方向上的一个无穷小偏移，该向量的演化确定被扰动轨道 $\{\mathbf{y}(n), n=0,1,2,\cdots\}$ 从未受扰动轨道 $\{\mathbf{x}(n), n=0,1,2\cdots\}$ 的无穷小偏移的演化。特别地，比值 $\mathbf{y}(n)/\|\mathbf{y}(n)\|$ 定义轨道从 $\mathbf{x}(n)$的无穷小偏移。当 $\|\mathbf{y}(n)\| > \|\mathbf{y}(0)\|$ 时，比值 $\mathbf{y}(n)/\|\mathbf{y}(0)\|$ 为无穷小偏移的增长因子；当 $\|\mathbf{y}(n)\| < \|\mathbf{y}(0)\|$ 时，它为无穷小偏移的缩减因子。对初始条件 $\mathbf{x}(0)$和初始偏移 $\boldsymbol{\alpha}_0 = \mathbf{y}(0)/\|\mathbf{y}(0)\|$，Lyapunov 指数被定义为：

$$\lambda(\mathbf{x}(0), \boldsymbol{\alpha}) = \lim_{n\to\infty} \frac{1}{n} \log\left(\frac{\|\mathbf{y}(n)\|}{\|\mathbf{y}(0)\|}\right) \tag{13.74}$$

一个 d 维混沌过程共有 d 个 Lyapunov 指数，可为正、负或 0。正的 Lyapunov 指数说明状态空间中一轨道的不稳定性。这种情况也可以表述为，

- *正的 Lyapunov 指数导致混沌过程对初始条件的敏感性。*
- *负的 Lyapunov 指数控制轨道中瞬态的衰减。*
- *一个为 0 的 Lyapunov 指数表明用以产生混沌的固有的动态系统可用一个联立的非线性微分方程组描述，即该混沌过程是一个流。*

在 d 维状态空间中体积依 $\exp(L(\lambda_1+\lambda_2+\cdots+\lambda_d))$变化，这里 L 是未来的时间步数。因此对一个耗散过程，所有 Lyapunov 指数之和必须是负数。这是状态空间的体积要随时间增加而缩减所必须满足的条件，它是物理实现的一个要求。

Lyapunov 维数

给定 Lyapunov 谱 $\lambda_1, \lambda_2, \cdots, \lambda_d$，Kaplan and Yorke（1979）提出了一个奇异吸引子的 Lyapunov 维数定义如下：

$$D_L = K + \frac{\sum_{i=1}^{K} \lambda_i}{|\lambda_{K+1}|} \tag{13.75}$$

其中 K 是满足下列两个条件的整数：

$$\sum_{i=1}^{K} \lambda_i > 0 \quad \text{和} \quad \sum_{i=1}^{K+1} \lambda_i < 0$$

通常，Lyapunov 维数 D_L 和相关维数 D_2 的大小大体相同。这是混沌过程的一个重要属性。也就是说，虽然 Lyapunov 维数和相关维数是用完全不同的方式定义，但对一个奇异吸引子，它们的值是非常接近的。

混沌过程的定义

在整章中我们说到了混沌过程，但没有正式定义它。根据我们对 Lyapunov 指数的了解，可以给出如下定义：

一个混沌过程是由一个非线性确定系统产生的，它至少有一个正的 Lyapunov 指数。

至少有一个正的 Lyapunov 指数是对初始条件敏感性成立的必要条件，对初始条件敏感是一个奇异吸引子的特点。

最大的 Lyapunov 指数也定义一个混沌过程的可预测范围。特别地，一个混沌过程的短期可预测性近似等于最大 Lyapunov 指数的倒数（Abarbanel，1996）。

13.11　混沌过程的动态重构

*动态重构*可以定义为映射的辨识，该映射对未知的 m 维动态系统提供模型。这里，我们的兴趣是对一个已知为混沌的物理系统产生的时间序列进行动态建模。换句话说，给定一时间序列 $\{y(n)\}_{n=1}^{N}$，我们希望建造一个模型来捕获产生可观察 $y(n)$ 的潜在动力学。如我们在前面一节开头指出的那样，N 代表样本大小。动态重构的主要动机是从这样一个时间序列中得到实际意义，从而绕开对潜在动力学的详细数学知识的需要。感兴趣的系统一般太复杂以至于不能用数学方式刻画它。我们仅有的可用信息包含在对系统的一个可观测量进行测量所得到的时间序列内。

动态重构理论[7] 最基本的结果是一个称为*延迟嵌入*（delay embedding）定理的几何定理，该定理是由 Takens（1981）提出的。Takens 考虑一个无噪声系统，集中于*延迟坐标映射*（delay coordinate map）或*预测*（predictive）*模型*，映射或模型是由表示动态系统的一个可观测量所表示的时间序列构造的。特别地，Takens 证明：如果动态系统和可观测量是一般的（generic），那么从一个 d 维光滑紧流形到 $\mathbb{R}^{2d+1}$ 的延迟坐标映射在该流形上是*微分同胚*（diffeomorphism），这里 d 是动态系统状态空间的维数（微分同胚已在第 7 章节讨论）。

为了用信号处理术语对 Takens 定理作解释，首先考虑一个未知的动态系统，该系统在离散时间的演化由非线性差分方程

$$\mathbf{x}(n+1) = \mathbf{F}(\mathbf{x}(n)) \tag{13.76}$$

描述，其中 $\mathbf{x}(n)$ 是系统在时刻 n 的 d 维状态向量，$\mathbf{F}(\cdot)$ 是一个向量值函数。这里假定采样周期为 1。系统输出的时间序列 $\{y(n)\}$ 用状态向量 $\mathbf{x}(n)$ 定义如下：

$$y(n) = g(\mathbf{x}(n)) + v(n) \tag{13.77}$$

其中 $g(\cdot)$ 是标量值函数，$v(n)$ 表示加性噪声。噪声 $v(n)$ 解释为在观测 $y(n)$ 中的不完全和不精确的综合效果。式(13.76)和式(13.77)描述动态系统的状态空间行为。根据 Takens 定理，当 $v(n)=0$ 时多变量动态系统的几何结构可以从新向量

$$\mathbf{y}_R(n) = [y(n), y(n-\tau), \cdots, y(n-(D-1)\tau)]^T \tag{13.78}$$

构成的 D 维空间中观察的 $y(n)$ 展现，其中 τ 是一个称为*归一化嵌入延迟*的正整数。也就是说，对不同的离散时间 n，给定观察值 $y(n)$，它和未知动态系统的一个可观察值（分量）有关，假定 $D \geqslant 2d+1$，使用 D 维向量 $\mathbf{y}_R(n)$ 动态重构是可能的，其中 d 是系统状态空间的维数。以后我们就称这个陈述为*嵌入-延迟定理*。对动态重构来说，条件 $D \geqslant 2d+1$ 是*充分的但不是必要的*。寻找合适 D 的过程称为*嵌入*。能够实现动态重构的最小的整数 D 称为*嵌入维数*，用 D_E 表示。

嵌入-延迟定理具有很强的意义：重建空间中点 $\mathbf{y}_R(n) \rightarrow \mathbf{y}_R(n+1)$ 的演化服从原始状态空间中未知动态系统 $\mathbf{x}(n) \rightarrow \mathbf{x}(n+1)$ 的演化。也就是说，不能观察的状态向量 $\mathbf{x}(n)$ 的许多重要属性可以在由 $\mathbf{y}_R(n)$ 定义的重建空间中毫无疑义地得到。然而，为了获得这个重要结果，我们需要嵌入维数 D_E 和归一化嵌入延迟 τ 的可靠估计，如下综述：

1. 充分条件 $D \geqslant 2d+1$ 使得解除吸引子一个轨道的自相交成为可能，这是出现在轨道投影到低维数时出现的问题。嵌入维数 D_E 可以小于 $2d+1$。推荐的过程就是从可观测数据直接估计 D_E。估计 D_E 的可靠方法在 Abarbanel（1996）中描述的*假最近邻方法*。在此方法中，系统地考察数据点和它们的近邻，先在维数 $d=1$，然后 $d=2, \cdots$，如此等等。我们借以确立明显近邻停止时的条件，是当添加更多元素到重构向量 $\mathbf{y}_R(n)$ 时“不被投影”，这样就获得对嵌入维

数 D_E 的估计。

2. 很不幸，延迟-嵌入定理并未提及归一化嵌入延迟 τ 的选择问题。事实上，只要可用时间序列无限长，它允许用任何的 τ。然而，实际上我们只能在有限长度 N 的观察数据上工作。选择 τ 的正确方法是认识到归一化嵌入延迟 τ 对 $y(n)$和 $y(n-\tau)$应足够大，使它们基本上独立，这样才能作为重建空间的坐标；但也不能使它们完全独立，以致没有任何联系。满足这个要求的最好办法就是选择特定的 τ 使得 $y(n)$和 $y(n-\tau)$之间的互信息获得它们第一个最小值(Fraser，1989)。(互信息在第 10 章讨论。)

递归预测

从前面讨论中知道，动态重构问题可以解释为恰当地表示信号动力学（嵌入步骤）和建造一个预测映射（识别步骤）。因此，实际上我们用下面的网络拓扑结构来进行动态建模。

- 短期记忆（例如延迟线记忆）结构实现嵌入，由此根据可观察的 $y(n)$和它的延迟形式来定义重建向量 $\mathbf{y}_R(n)$，参见式(13.78)。
- 训练作为单步预测器（如神经网络）的多输入单输出（MISO）自适应非线性系统，用它识别未知映射 f：$\mathbb{R}^D \rightarrow \mathbb{R}^1$，定义如下：

$$\hat{y}(n+1) = f(\mathbf{y}_R(n)) \tag{13.79}$$

式(13.79)描述的预测映射是动态建模的中心问题：一旦确定，演化 $\mathbf{y}_R(n) \rightarrow \mathbf{y}_R(n+1)$变成已知，由此确定未知演化 $\mathbf{x}(n) \rightarrow \mathbf{x}(n+1)$。

现在，假设有一个严格的理论来帮助我们决定非线性预测器是否已成功地识别这个未知映射 f。在线性预测中，最小化预测误差的均方值可以得到一个精确的模型。然而，混沌时间序列不同。同一个吸引子的两个轨道在每次采样基础上都有很大的不同，所以最小化预测误差的均方值对一个成功的映射仅是必要条件而不是充分条件。

动态不变量（即相关维数和 Lyapunov 指数），度量吸引子的全局属性，所以它们应该可以判断动态建模的成功与否。因此，检验动态建模的一个实际方法是在奇异吸引子上挑选一点，然后反馈输出到其输入成为一个自治系统，如 13.18 图所示。这样一个操作称为迭代预测或递归预测。一旦初始化完成，该自治系统的输出就是动态重构过程的一个实现。这当然要假定预测器开始时已被正确地设计。

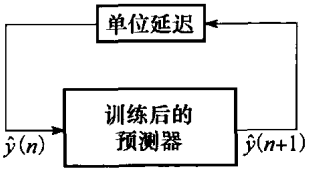


图 13.18　在混沌过程动态重构中用于迭代预测的单步预测器

对于可靠动态重构，我们可以把重建向量 $\mathbf{y}_R(n)$定义为一个完全的 m 维向量

$$\mathbf{y}_R(n) = [y(n), y(n-1), \cdots, y(n-m+1)]^T \tag{13.80}$$

其中 m 是一个整数，定义为

$$m \geqslant D_E\tau \tag{13.81}$$

这种重建向量 $\mathbf{y}_R(n)$的形式比式(13.78)提供的形式对可预测模型提供更多的信息，因此可能产生一个更精确的动态重构。然而，这两种形式有一个共同的特点：它们的组成都由嵌入维数 D_E 的知识唯一定义。在任何情况下，明智的方法是用最小允许的值 D，也就是 D_E，来最小化加性噪声 $v(n)$对动态重构质量的影响。

动态重构是一个不适定的过滤问题

现实中，动态重构是一个不适定的逆问题。之所以这样说是因为以下情况极有可能发生，即破坏对于逆问题适定的 Hadamard 三个条件中的一个或者多个，这点在第 7 章明确地表达过：

1. 由于一些未知的原因存在条件可能被破坏。

2. 在可观察时间序列上，可能没有充分的信息足以唯一地重建非线性动态系统。

3. 不可避免地出现加性噪声和观察时间序列的某种不精确都会增加动态重构的不确定性。特别地，如果噪声水平太高，连续性标准也可能被破坏。

那么怎么使动态重构问题适定呢？答案在于把包含关于输入-输出映射的先验知识的某种形式作为主要要求。换句话说，在预测模型的设计中，为了解决动态重构问题需要引入某种形式的限制（例如输入-输出映射的光滑性）。满足这个要求的有效方法是用 Tikhonov 的正则化理论，这也在第 7 章讨论。

另一个需要考虑的问题是预测模型以足够精度解决逆问题的能力。在这个背景下，用神经网络建造预测模型是合适的。特别地，多层感知器或径向基函数网络的通用逼近特性意味着我们利用具有适当规模的这种或那种神经网络可以注意重建精度的问题。另外，由于刚才说明的理由我们需要正则化的解决方法。理论上，多层感知器和径向基函数网络都适宜正则化的使用；但如第 7 章所解释，我们发现在径向基函数网络中包括正则化理论作为它们设计的整体部分，在数学上易于处理。

案例研究：Lorenz 吸引子的动态重构

为了阐明动态重构的思想，我们考虑有三个联立常微分方程组的系统。该系统由 Lorenz (1963) 从低压大气热对流的偏微分方程组的 Galerkin 近似抽象而来，它成为测试非线性动态系统思想的一个主要方程组。Lorenz 吸引子的方程组为

$$\begin{aligned}\frac{dx(t)}{dt} &= -\sigma x(t) + \sigma y(t)\\ \frac{dy(t)}{dt} &= -x(t)z(t) + rx(t) - y(t)\\ \frac{dz(t)}{dt} &= x(t)y(t) - bz(t)\end{aligned} \tag{13.82}$$

其中 σ，r 和 b 是无量纲参数。这些参数的典型值是 $\sigma=10$，$b=8/3$，$r=28$。

图 13.19 显示在两个具有 400 个中心的 RBF 网络上，使用基于 Lorenz 吸引子的 $x(t)$分量的带噪声时间序列实施迭代预测的结果。信噪比是 +25 分贝。为了设计正则化的 RBF 网络，我们使用以下参数：

输入层的大小 $m=20$

正则化参数 $\lambda=10^{-2}$

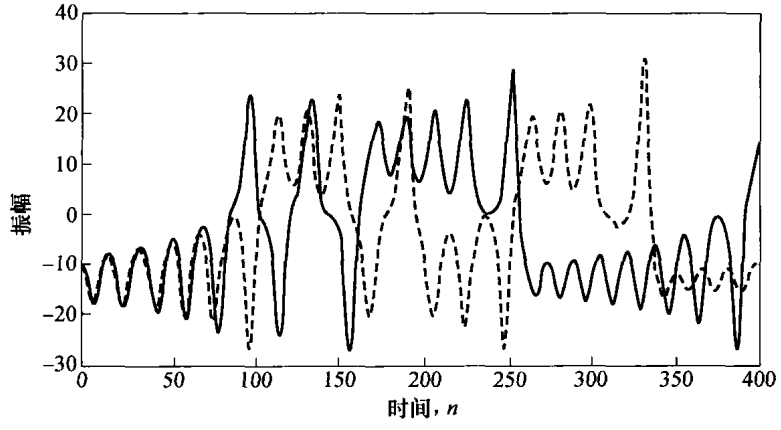


图 13.19　在 Lorenz 数据上正则化迭代预测（$N=400$，$m=20$），SNR=+25 分贝；实曲线为实际的混沌信号，红色曲线为重构信号

用式(13.81)决定输入层的大小；使用第 7 章描述的广义交叉验证过程决定正则化参数。

如图 13.9 所示，使用一个正则 RBF 网络，动态重构的解已经学习了这个动力学系统，是在下列意义上说：在迭代预测下网络的输出十分近似 Lorenz 吸引子在短时上的实际轨迹。这个结果由表 13.5 中的事实为根据，我们总结了两种情况下的 Lorenz 数据：

(a) 信噪比 SNR＝25 分贝的 Lorenz 系统。

(b) 用表 13.5 的带噪声 Lorenz 时间序列的重建数据。

表 13.5　用 Lorenz 系统的动态重构试验的参数小结

(a) 有噪声 Lorenz 系统：25 分贝 SNR	(b) 用图 13.19 的有噪声 Lorenz 数据重构的系统
使用样本数：35 000	产生样本数（递归地）：35 000
1. 归一化嵌入延迟，$\tau=4$	1. 归一化嵌入延迟，$\tau=4$
2. 嵌入维数，$D_E=5$	2. 嵌入维数，$D_E=3$
3. Lyapunov 指数：	3. Lyapunov 指数：
$\lambda_1=13.2689$	$\lambda_1=2.5655$
$\lambda_2=5.8562$	$\lambda_2=-0.6275$
$\lambda_3=-3.1447$	$\lambda_3=-15.0342$
$\lambda_4=-18.0082$	4. 可预测范围约为 61 个样本
$\lambda_5=-47.0572$	
4. 可预测范围约 100 个样本	

注：所有的 Lyapunov 指数的单位为奈特/秒。如第 10 章讨论的那样，奈特是测量信息的一个自然单位。同样，在情形 (b) 中，动态重构只用一个正的 Lyapunov 阶把 Lyapunov 谱还原到正确的个数 3（等于方程的个数）。

用带噪声数据的重建数据的不变量和用无噪声 Lorenz 数据的重建数据不变量相近。偏差的绝对值是由于嵌入重建吸引子的噪声的残留影响以及估计程序的不精确。图 13.19 清楚地显示动态建模比预测有更多内容。这幅图以及很多不包括在这里的其他图像都显示已正则化 RBF 关于吸引子上的点的解的“鲁棒性”，这些用于初始化迭代预测过程。

从图 13.19 使用正则化得出下面两点观察，是值得特别注意的：

1. 图 13.19 的重建时间序列的短期可预测性是大约 60 个样本。从无噪声 Lorenz 吸引子的 Lyapunov 谱计算的理论可预测值是 100 个样本。试验和无噪声 Lorenz 吸引子的预测范围的偏差仅仅显示用来实施动态重构的实际数据里面存在噪声。从重建数据计算的理论可预测值范围是 61（见表 13.5），这非常接近短期可预测性的试验观察值。

2. 一旦超出短期可预测性的期限，用图 13.19 中的重建时间序列开始偏离真正 Lorenz 吸引子的无噪声实现。这基本上是混沌动力学的一个现象，也就是对初始条件的敏感性。像前面提到的那样，对初始条件的敏感性是混沌的一个标志。

13.12　小结和讨论

递归网络中稳定性问题

在本章中，我们介绍了确定神经动力学系统的数学基础，用式(13.2)表示，为了便于表示再写一遍：

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}(t)=\mathbf{F}(\mathbf{x}(t))$$

其中 t 是连续时间，$\mathbf{x}(t)$是系统的状态；而 $\mathbf{F}(\cdot)$ 是一个向量值的函数，它的每个元素作用于状态 $\mathbf{x}(t)$的相应的每个元素。

本章开头的讨论主要集中在系统稳定性这个问题上。特别描述了 Lyapunov *直接方法*，它为就有关 $\mathbf{x}(t)$连续标量函数而言的稳定性问题研究提供了强大有力的数学工具，称之为 Lyapunov *方程*。这个方法包括两个定理能够帮助我们确定一个给定的自治非线性动力学系统是否

稳定或者渐近稳定。这里有适当的提示语：这个方法没有教我们如何找到 Lyapunov 方程，反而，这个任务需要研究人员的精巧设计以找到它们。但是在感兴趣的很多实际问题上，能量函数能作为 Lyapunov 方程。

联想记忆的模型

在本部分，我们讨论了两个联想记忆的模型：Hopfield 模型和 BSB 模型，这两个模型有下面一些共同特点：

- 都使用相应于 Hebb 学习规则的正反馈。
- 它们都有能量（Lyapunov）函数，固有的动力学以迭代方式使能量函数最小化。
- 它们都能利用吸引子动力学进行计算。

很自然，它们各自的应用领域是不同的。BSB 模型具有固有的聚类能力。另一方面，Hopfield 模型能够按内容访问存储器自我操作；但是，在数字通信领域中它的误差-修正码没有已确立的误差-修正编码优秀[8]。Hopfield 网络的模拟版本同样也作为解决旅行商问题的一个模型[9]。

进一步讨论 Hopfield 模型

Hopfield 在 1982 年的论文对神经网络界有重大影响。事实上，它是复苏 20 世纪 80 年代持续的神经网络研究兴趣的催化剂之一。

更重要的是，在这篇经典论文中进行以下操作：

- 考虑递归网络，人工配置使其具有对称突触权值，来满足式(13.21)中对称条件。
- 明确地表达能量函数 E，如式(13.28)中定义。
- 证明能量函数 E 是一个 Lyapunov 方程。
- 通过迭代最小化能量函数来证明网络能够以几个稳定点展示再现行为。

同时在一篇相对短的文章中做到所有这些，这也就使得 Hopfield 在 1982 年的论文更加优秀和令人印象深刻。事实上，它也是过去十年在物理学家和数学家中产生许多令人兴奋的事的缘由。

简言之，Hopfield 向我们展示了一个简单的，结构性的行为产生一个复杂的，时变非线性动力学系统是可能的。这种动态行为的可能性之前被其他研究者研究过，但是 Hopfield 的论文第一次把递归网络的再现行为的内在以一种可见可信的方式融汇在一起。

适当的提醒是：以下想法是天真的，即认为 Hopfield 网络连同其他神经网络界的联想记忆模型能够适用于人类记忆（Anderson，1995）。

作为理解哺乳动物大脑的帮助者的大规模计算机模型

模型化大脑的部分功能，或者更加雄心勃勃的，模型化整个大脑本身，这是一项具有挑战性的任务。激励人心的是 Izhikevich and Edelman 在哺乳动物大脑结构和动态复杂性上面的先驱性的工作。在他们 2008 的论文中描述了哺乳动物皮层系统的大规模计算机模型。众所周知，丘脑-皮层系统对意识从以下意义上是重要的，即失去丘脑或皮层将丢失意识；另一方面，例如，丢失海马体或者小脑将损害大脑的部分功能，但保留意识。对丘脑-皮层系统的关注使得 Izhikevich-Edelman 模型变得更加有趣。

整个模型的主要特点包括：

1. 一百万多个多区划元的模拟。为了模拟，神经元经校准来再生有名的小鼠的体外反应。在模拟中 Izhikevich（2007a）之前关于神经元峰值动力学的工作是突出的。

2. 大约五亿个突触，这个大规模突触模型自动展示三个高度相关的神经活动：

1）*神经动力学*。模拟的峰值动力学中每个神经元和每个树状突的区划可以用下面两个微分方程描述。

$$C\frac{dv}{dt} = k(v - v_r)(v - v_{thr}) - u + I \tag{13.83}$$

$$\frac{du}{dt} = a[b(v - v_r) - u] \tag{13.84}$$

其中　C=细胞膜电容

v=细胞膜电位

v_r=静息电位

v_{th}=瞬时阈电位

u=定义所有向内和向外电压门控电流差的回复变量

I=树突的突触的电流

a 和 b 是常数。假定细胞膜电位大于峰值的极大值的时候，神经模型启动尖脉冲（动作电位），并且模型中的所有变量重设。

2）短时突触可塑性。在模型中，每个突触的传导率（长度）可以升高也可以降低，在短时间规模内分别代表抑制和促进。

3）长时峰值定时相关可塑性。这个模型的第二个可塑性特点，每个突触增强或者抑制，依赖前突触神经元点火的顺序和相应的后突触神经元的树状分隔。

3. 泛化性能。这个模型具有展示未建立在该模型中的正常脑组织动作的行为制度。

赋予了这些神经生物学特性的大规模计算模型说明了我们正逐渐地向建立哺乳动物大脑的大规模计算模型接近，这样的模型能实现实时操作。

注释和参考文献

1. 一个非自治（nonautonomous）系统由状态方程

$$\frac{d}{dt}\mathbf{x}(t) = \mathbf{F}(\mathbf{x}(t), t)$$

定义，初始条件为 $\mathbf{x}(t_0)=\mathbf{x}_0$。对一个非自治系统，向量域 $\mathbf{F}(\mathbf{x}(t), t)$依赖于时间 t。因此，不像自治系统那样，我们一般不置初始时间为 0（Parker and Chua，1989）。

2. 一般来说，除式(13.11)外一个非线性动态系统的全局稳定性还需要径向无界条件

$$V(\mathbf{x}) \to \infty, \quad \|\mathbf{x}\| \to \infty$$

成立（Slotine and Li，1991）。由具有 sigmoid 激活函数的神经网络构造的 Lyapunov 函数通常满足该条件。

3. 我们给出一个吸引子的严格定义如下（Lanford，1981；Lichtenberg and Lieberman，1992）：

状态空间的一个子集（流形）$\mathcal{M}$被称为一个吸引子，如果：

- $\mathcal{M}$根据流保持不变
- 在流中，$\mathcal{M}$周围有一个（开）邻域收缩到$\mathcal{M}$
- $\mathcal{M}$的所有部分都不是瞬态的
- $\mathcal{M}$不能被分成两个互不相交的不变片（piece）

4. 集中点火神经元

式(13.14)的加性模型并没完全抓住一个生物神经元的精髓。特别地，它忽略了动作电位中编码的时序信息；动作电位在介绍章节中给出简要的定性描述。Hopfield（1994）描述一个动态模型，通过考虑一个集中点火（Integrate and Fire）神经元捕捉动作电位。这样一个神经元的运行由一阶微分方程

$$C\frac{d}{dt}u(t) = -\frac{1}{R}(u(t) - u_0) + i(t) \tag{A}$$

描述，其中　$u(t)$=神经元内部电位

C=神经元周围细胞膜的电容

R=细胞膜的漏阻（leakage resistance）

$i(t)$=由另一神经元注入当前神经元的电流

u_0=当 $i(t)$消失时神经元减少的电位

在每次内部电位 $u(t)$达到阈值时产生一个动作电位。

动作电位被看作是 Dirac delta（冲击）函数，表示为

$$g_k(t) = \sum_n \delta(t - t_{k,n}) \tag{B}$$

其中 $t_{k,n}$，$n=1,2,3,\cdots$，代表神经元 k 的激活动作电位的次数，这些次数由式(A)所定义。流入神经元 k 的总电流 $i_k(t)$的行为 $i_k(t)$模型化为

$$\frac{\mathrm{d}}{\mathrm{d}t} i_k(t) = -\frac{1}{\tau} i_k(t) + \sum_j w_{kj} g_j(t) \tag{C}$$

其中 w_{kj} 为神经元 j 到神经元 k 的突触权值，τ 是神经元 k 的特征时间常数，函数 $g_j(t)$由式(2)定义。

式(13.14)的加性模型可看作是(C)的一个特例。具体地，忽略 $g_j(t)$尖峰（spiky）性质，而代之以 $g_j(t)$和一个光滑函数的卷积。这是因为高度连接在一个合理的时间间隔内式(C)右边的总和会有许多项，并且我们只关心神经元 k 点火率的短期行为。

5. Little 模型（Little，1974；Little and Shaw，1978）和 Hopfield 模型一样使用同样的权值。然而，它们不同之处在于 Hopfield 模型用异步（串行）动力学，而 Little 模型用同步（并行）动力学。相应地，它们展示不同的收敛性（Bruck，1990；Goles and Martinez，1990）。Hopfield 网络总是会收敛到一个稳定状态，而 Little 模型总是会收敛到一个稳定状态或长度至多为 2 的极限环。所谓“极限环”是指网络状态空间的长度小于或等于 2 的环。
6. 式（13.71)定义的相关函数 $C(q,r)$的思想在统计上已知是从 Rényi（1970）的工作得来的。然而用它去刻画一个奇异吸引子是在 Grassberger and Procaccia（1983）中提出的。他们最初是讨论相关维数 $q=2$ 时 $C(q,r)$的应用。
7. 从一个时间序列里用独立坐标来构建动态系统首先由 Packard 等（1980）提出。然而，这篇论文并没有给出证明，用的是“导数”嵌入而不是时间-延迟嵌入。时间-延迟嵌入或延迟坐标嵌入归功于 Ruelle 和 Takens。特别地，1981 年 Takens 发表了一篇在数学上很深刻的时间-延迟嵌入方面的文章，它应用于吸引子为曲面或类似环面；也可以参看 Mañé(1981)在同一杂志上发表的同一主题的论文。Takens 的论文对非数学家来说很难懂，Mañé 的更难懂。延迟坐标映射的思想在 Sauer 等（1991）中得到提炼。在这篇论文中采用的方法是对 Whitney(1936)和 Takens(1981)的早期结果的综合和扩展。
8. 伪状态干扰 Hopfield 模型的检索阶段是因为它们趋于把存储的模型混合起来。相应地，Hopfield 模型的误差-修正能力由于伪状态的产生而降低。网络的结果作为一个误差-修正系统，Hopfield 模型不是那么好。这种情况特别是在数字通信领域中当 Hopfield 模型对比已确立的误差-修正编码时（Lin and Costello，2004)。后者编码让人印象深刻在于（聪明地按照制定编码方案通过插入奇偶校验）它们能够接近所谓的香农极限，而这个挑战自从香农的 1948 关于信息论的经典论文就引起编码理论学家的注意。
9. 组合最优化问题在数学中以几乎最难而出名。这类最优化问题包括旅行商问题（TSP)，它被视为一个经典。给定具体数目的城市的地点，假定在一个平面上，找到起点和终点相同的最短旅行路径。TSP 问题很容易陈述，但是难于精确求解，因为没有方法来找到最优化旅途，达不到计算所有可能的路径的长度，然后挑取最短的。它被称为 NP 完全（Hopcroft and Ullman，1979)。

Hopfield 和 Tank 在 1985 年的一篇论文中基于式(13.20)中 N 对一阶微分方程的系统提出了使用模拟网络的应用，代表了 TSP 问题的一个解法。具体地，网络的权值由旅途中访问过的城市之间的距离决定，问题的最优化解被作为式(13.20)方程中的固定点。此处困难在于将组合最优化问题“映射”到连续（模拟）Hopfield 模型。模型遵守最小化能量（Lyapunov）函数，起到限制一些硬约束的目标函数的作用。如果违背约束中任一个，解将视为无效。在 Gee 等（1993)中，证明了 Hopfield 模型的成功对这样的方式极为敏感，即为联立的方程组系统的 Lyapunov 方程的构造方式。

习题

动力系统

13.1　对于状态向量 $\mathbf{x}(0)$作为一个动态系统的平衡状态，重述 Lyapunov 定理。

13.2　验证图 P13.2a 和 b 的框图分别对应神经动力学方程(13.18)和(13.19)。使用这两个等式说明图 P13.2 中的两个框图的有效性。

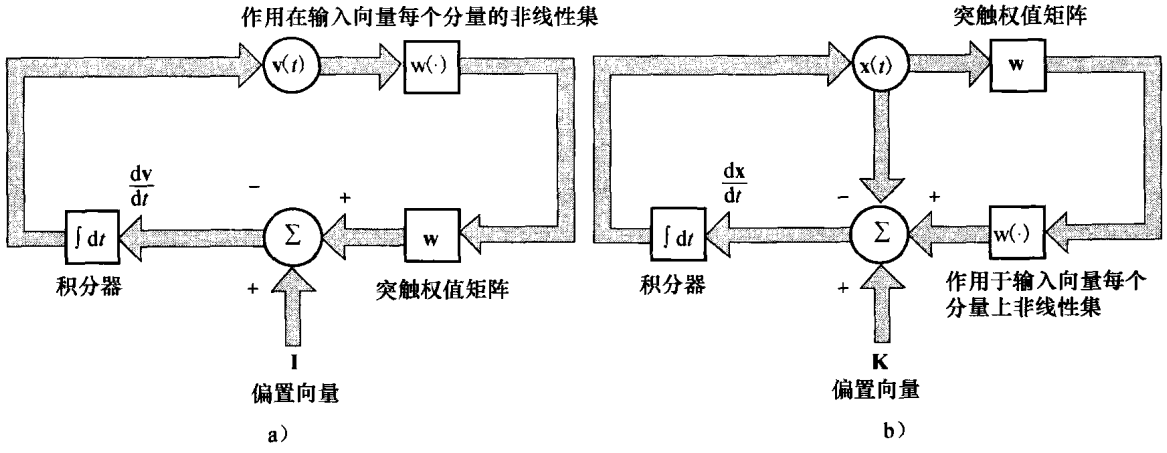


图　P13.2

13.3　考虑一般的神经动力学系统，它依赖于未指定的内部状态参数、外部动态刺激和状态变量。系统由状态方程

$$\frac{dx_j}{dt} = \varphi_j(\mathbf{W},\mathbf{u},\mathbf{x}), \quad j = 1,2,\cdots,N$$

定义，其中矩阵 $\mathbf{W}$ 代表系统的内部动态参数，向量 $\mathbf{u}$ 代表外部动态刺激，$\mathbf{x}$ 是状态向量，它的第 j 个元素用 x_j 表示。对于 $\mathbf{W}$，$\mathbf{u}$ 的值和在状态空间的某些运行区域 $\mathbf{x}(0)$ 的值，假定系统的轨迹收敛到点吸引子（Pineda，1988b）。讨论所描述的系统如何能用于如下应用：

(a) 连续映射器，$\mathbf{u}$ 是输入，$\mathbf{x}(\infty)$ 是输出。

(b) 自联想记忆，$\mathbf{x}(0)$ 是输入，$\mathbf{x}(\infty)$ 是输出。

Hopfield 模型

13.4　考虑由 5 个神经元组成的 Hopfield 网络，它需要存储以下三个基本记忆：

$$\xi_1 = [+1,+1,+1,+1,+1]^T \quad \xi_2 = [+1,-1,-1,+1,-1]^T \quad \xi_3 = [-1,+1,-1,+1,+1]^T$$

(a) 计算网络的 5×5 突触权值矩阵。

(b) 用异步更新演示所有三个基本记忆 ξ_1，ξ_2，ξ_3 满足对齐条件。

(c) 若 ξ_1 是有噪声的，它的第二个元素极性反转，研究网络的检索性能。

13.5　研究同步更新习题 13.4 所描述 Hopfield 网络的检索性能。

13.6　(a) 证明

$$\xi_1 = [-1,-1,-1,-1,-1]^T \quad \xi_2 = [-1,+1,+1,-1,+1]^T \quad \xi_3 = [+1,-1,+1,-1,-1]^T$$

也是习题 13.4 所描述的 Hopfield 网络的基本记忆。这些基本记忆和习题 13.4 中的基本记忆之间有什么关系？

(b) 假定习题 13.4 中基本记忆 ξ_3 的第一个元素被掩模（即减少为 0）。确定 Hopfield 网络所产生的结果模式。比较这个结果和 ξ_3 的原始形式。

13.7　考虑由两个神经元构成的简单 Hopfield 网络，网络的突触权值矩阵为

$$\mathbf{W} = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix}$$

每个神经元的偏置为 0，网络的四个可能状态是

$$\mathbf{x}_1 = [+1,+1]^T \quad \mathbf{x}_2 = [-1,+1]^T \quad \mathbf{x}_3 = [-1,-1]^T \quad \mathbf{x}_4 = [+1,-1]^T$$

(a) 说明状态 $\mathbf{x}_2$ 和 $\mathbf{x}_4$ 是稳定的，而状态 $\mathbf{x}_1$ 和 $\mathbf{x}_3$ 成为极限环。用下面两个工具来说明：

1. 对齐（稳定性）条件

2. 能量函数

(b) 刻画状态 $\mathbf{x}_1$ 和 $\mathbf{x}_3$ 的极限环的长度是多少？

13.8　Hopfield 网络的能量函数可表达为：

$$E = -\frac{N}{2}\sum_{v=1}^{M} m_v^2$$

其中 m_v 代表由

$$m_v = \frac{1}{N}\sum_{j=1}^{N} x_j \xi_{v,j}, \quad v = 1,2,\cdots,M$$

定义的重叠，其中 x_j 是状态向量 $\mathbf{x}$ 的第 j 个元素，$\xi_{v,j}$ 是基本记忆 ξ_v 的第 j 个元素，M 是基本记忆个数。

13.9 可以证明 Hopfield 网络相对于干扰是鲁棒的，如突触噪声。用一个说明性的例子来证明这个说法的有效性。

13.10 第 11 章中的 Boltzmann 机可以视为 Hopfield 网络的扩展。请列出两个非监督学习系统的异同点。

Cohen-Grossberg 定理

13.11 考虑式(13.48)定义的 Lyapunov 函数 E。如果式(13.49)至式(13.51)的条件满足，证明

$$\frac{dE}{dt} \leqslant 0$$

13.12 在 13.9 节，我们通过应用 Cohen-Grossberg 定理导出了 BSB 模型的 Lyapunov 函数。在推导式(13.63)时，省略了一些细节。请写出这些细节。

13.13 图 13.13 显示非单调激活函数的一个图形，该函数由 Morita (1993) 提出，这在注释 6 中讨论过。这个函数在构造 Hopfield 网络时用于代替双曲线正切函数。Cohen-Grossberg 定理适用于这样构造的联想存储器吗？请说明你的理由。

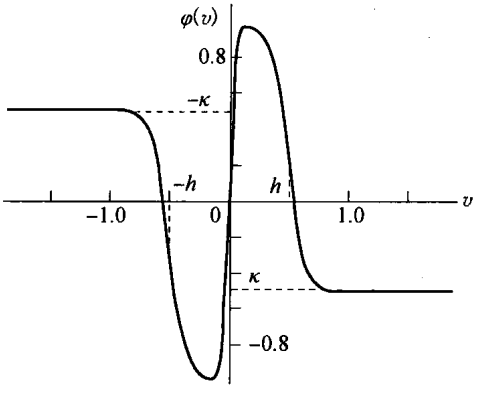


图　P13.13

数据表达

13.14 根据 Chigirev and Bialek (2005)，在第 10 章中我们使用了优化流形的思想描述了一种数据表达的算法。给定一些不带标签的数据作为算法输入，算法可以产生下列两种结果：

- 一些列的流形点，在其周围是已聚类的数据。
- 一个随机图，它把输入数据映射到这个流形上。

用在 13.10 节中描述的 Grassberger-Procacia 相关维的思想，为验证 Chigirev-Bialek 算法作为流形维复杂度的概率估计的有效性而概述一个实验。

动态系统状态估计的贝叶斯滤波

本章组织

本章围绕着一个基本而重要的主题展开：给定一组观测值，估计动态系统中的隐藏状态。

本章的内容结构如下：

14.1 节为概述，引起读者对逐次状态估计的学习兴趣。

14.2 节讨论状态空间的概念和状态空间的各种建模方法。

14.3 节介绍著名的卡尔曼滤波器，14.4 节将讨论保证滤波算法数值稳定性的平方根方法。14.5 节阐述利用扩展的卡尔曼滤波器处理轻度的非线性问题。

14.6 节讨论贝叶斯滤波。贝叶斯滤波算法至少在概念上为动态系统状态估计提供了统一的框架。而卡尔曼滤波器正是这一滤波模型的一个特例。

14.7 节对贝叶斯滤波器直接的数值近似问题提出了数值积分法则。在此基础上介绍了一种新的滤波器——数值积分卡尔曼滤波器，它的思想源于卡尔曼滤波器理论。

14.8 节对贝叶斯滤波近似问题提出了另一个算法。这一算法源于 Monte Carlo 模拟。特别地，提出了对粒子滤波器的详细处理。14.9 节通过计算机实验，比较了扩展的卡尔曼滤波器和粒子滤波器的性能。

14.10 节讨论卡尔曼滤波在对大脑各部分的建模中扮演的角色。

14.11 节总结并讨论了整章内容。

14.1 引言

在第 13 章介绍的神经动力学系统中，我们关注的主要问题是稳定性。在本章中，我们将考虑另一个重要的问题，即给定一组基于某一类型的状态观测值，如何估计动态系统的状态。观测发生在离散的时间点上，这并非是为了数学上的方便，而是因为观测值就是产生在离散时间点上的。此外，状态不仅是未知的，且对于观测者而言是隐藏的。因此，我们可以将状态估计问题视为逆向问题。

举一个说明性的例子，考虑一个动态驱动的多层感知器，该网络的每一层都有向前一层的反馈回路（例如从隐藏层到输入层）。网络的状态可被看做一个向量，该向量是由网络所有权重，按某一排序方式排列构成的。我们要做的是给定一个训练样本，利用逐次状态估计理论对网络权重向量进行有监督的调整。这一应用将在下一章详细讨论。然而，对此应用我们需要一个状态估计的连续过程，相关的基本原理也将在下一章阐述。

逐次状态估计理论的首次严格论述，出现在 1960 年卡尔曼发表的论文中。为了便于数学处理，卡尔曼的论述基于以下两个简单假设：

1. 动态系统完全是线性的。

2. 噪声对动态系统状态有扰动的作用且观测变量是加性的、服从高斯分布的。

基于上述假设，卡尔曼提出了对系统中未知状态进行最优估计的递归算法。在其适用领域中，卡尔曼滤波器毫无疑问经受住了时间的考验。

迄今为止，逐次状态估计理论仍是当下热门的研究领域。大多数该领域的研究工作集中于解决非线性及非高斯空间下的实际问题。在以上一种或两种情况下，通常无法得到最优估计结果。因此，我们需要解决近似估计算法的实现问题。所面临的挑战是如何使得算法既有理论依据又具备较高的计算效率。

14.2　状态空间模型

动态系统有一个共同的基本特征：系统的状态。该特征的严格定义如下：

一个随机动态系统的状态被定义为最少量的信息，这些信息包含过去作用于该系统的输入的影响，并足以完全描述系统将来的行为。

通常情况下，状态不是直接可测量的。而是用间接的方式测量一组观测值来反应状态对外部世界的影响。这样，未知动态系统的特征可以由状态空间模型描述，它包含了以下两个公式：

1. 系统（状态）模型，用公式表示为一阶马尔可夫链，用关于时间的函数描述状态的演变。公式如下：

$$\mathbf{x}_{n+1} = \mathbf{a}_n(\mathbf{x}_n, \boldsymbol{\omega}_n) \tag{14.1}$$

其中，n 表示离散时间，向量 $\mathbf{x}_n$ 表示当前状态的值，向量 $\mathbf{x}_{n+1}$ 表示下一状态的值，向量 $\boldsymbol{\omega}_n$ 表示动态噪声或过程噪声，$\mathbf{a}_n(\cdot,\cdot)$ 是关于两个参数的向量函数。

2. 测量（观测）模型，用公式表达如下：

$$\mathbf{y}_n = \mathbf{b}_n(\mathbf{x}_n, \boldsymbol{v}_n) \tag{14.2}$$

向量 $\mathbf{y}_n$ 表示一组观测值，向量 $\boldsymbol{v}_n$ 表示噪声的测量值，$\mathbf{b}_n(\cdot,\cdot)$ 表示另一个向量函数。

$\mathbf{a}_n$ 和 $\mathbf{b}_n$ 的下标 n 用于包括所有的状态，这两个函数是随时间改变的。为了让状态空间模型更具实用的价值，在研究中必须严密地描述系统的底层物理特征。

图 14.1 是单信号流示意图，它描述了由式(14.1)和式(14.2)定义的状态空间模型。图 14.2将状态随时间的演变描述为一个马尔可夫链。在两幅图中，模型的时间域表示具有以下特点：

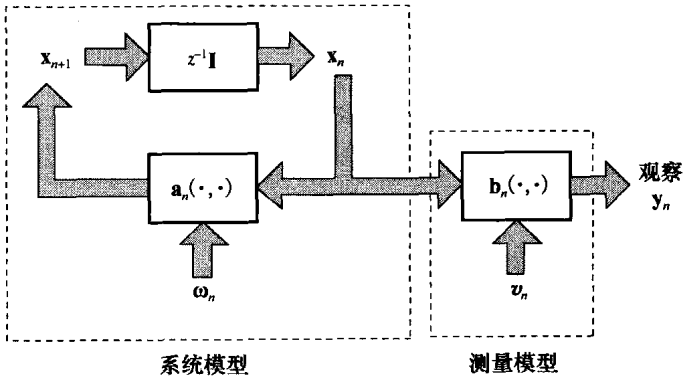


图 14.1　随时间变化的非线性动态系统的一般状态空间模型，其中 $z^{-1}\mathbf{I}$ 表示一组单位时间延时

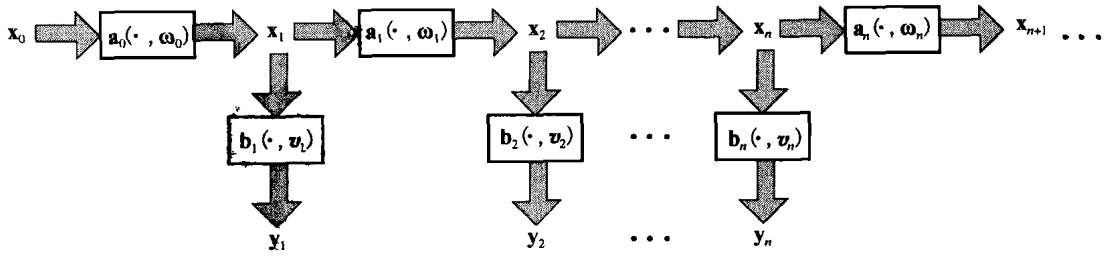


图 14.2　状态随时间的演变，看作一阶马尔可夫链

- 数学上和记法上的便利。
- 模型与物理现实的紧密联系。
- 解释系统统计行为的有意义的基础。

有理由做出如下假设：

1. 对任意的 n，初始状态 $\mathbf{x}_0$ 与动态噪声 $\boldsymbol{\omega}_n$ 是无关的。

2. 两种噪声源 $\boldsymbol{\omega}_n$ 和 $\boldsymbol{v}_n$ 是统计独立的，也就是说

$$\mathbb{E}[\boldsymbol{\omega}_n \boldsymbol{v}_k^T] = 0 \quad \text{，对于所有的 } n \text{ 和 } k \tag{14.3}$$

当 $\boldsymbol{\omega}_n$ 和 $\boldsymbol{v}_n$ 高斯相关时，上述等式是 $\boldsymbol{\omega}_n$ 和 $\boldsymbol{v}_n$ 相互独立的充分条件。

值得注意的是，图 14.2 中的马尔可夫模型，从根本上不同于第 12 章当中涉及动态规划的马尔可夫模型。因为在动态规划中，状态对于观测者而言是直接可以获得的，而逐次状态估计中的状态对于观测者而言是隐藏的。

逐次状态估计问题的描述

给定由 $\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n$ 组成的全部观测值的记录，计算在统计意义上最优的隐藏状态 $\mathbf{x}_k$ 的估计值，将这些估计值用逐次的方式表示出来。

这样，这一描述包含了两个系统：

- 未知的动态系统，其观测量 $\mathbf{y}_n$ 是关于隐藏状态的函数。
- 逐次状态估计器或滤波器，用于开发观测值中包含的状态信息。

从广义上说，我们可以将此视为“编码-解码”问题。观测值可视为被编码的状态，而由滤波器实现的状态估计过程则可视为对观测值的解码。

总之，当 $k>n$ 时状态估计称为预测，当 $k=n$ 时称为滤波，当 $k<n$ 时称为平滑。通常情况下，因为平滑器使用更多的观测量，所以在统计上比预测器、滤波器更为精确。另一方面，预测器和滤波器可用于实时应用，而平滑器不能。

状态空间模型的分类体系

解决状态估计问题，在数学上的困难主要取决于状态空间模型的实际描述。因而产生了状态空间模型的分类体系：

1. 线性、高斯模型。该模型是最简单的状态空间模型。式(14.1)和式(14.2)可分别变换为

$$\mathbf{x}_{n+1} = \mathbf{A}_{n+1,n}\mathbf{x}_n + \boldsymbol{\omega}_n \tag{14.4}$$

和

$$\mathbf{y}_n = \mathbf{B}_n\mathbf{x}_n + \boldsymbol{v}_n \tag{14.5}$$

其中，$\mathbf{A}_{n+1,n}$是从状态 $\mathbf{x}_n$ 到状态 $\mathbf{x}_{n+1}$的过渡矩阵，$\mathbf{B}_n$ 是测量矩阵。动态噪声 $\boldsymbol{\omega}_n$ 和测量噪声 $\boldsymbol{v}_n$ 均是加性的，并假设为统计独立的均值为 0 的高斯过程，其协方差矩阵分别用 $\mathbf{Q}_{\omega,n}$和 $\mathbf{Q}_{v,n}$表示。用式(14.4)及式(14.5)定义的状态空间模型即为卡尔曼提出的递归滤波器所使用的模型。它在数学上是完美的，并回避了任何可能的近似问题。卡尔曼滤波器相关的内容将在 14.3 节中介绍。

2. 线性、非高斯模型。在此模型中，我们仍然使用式(14.4) 以及式(14.5)，但动态噪声 $\boldsymbol{\omega}_n$ 和测量噪声 $\boldsymbol{v}_n$ 都被假设为加性的、统计独立的非高斯过程。这两个过程的非高斯性是导致了数学上的困难的唯一来源。在这样的情况下，我们可以使用高斯求和近似扩展卡尔曼滤波器的应用范围，总结如下：

任何描述多维非高斯向量的概率密度函数 $p(\mathbf{x})$，用样本值 $\mathbf{x}$ 表示，能够用高斯求和公式尽可

能地逼近

$$p(\mathbf{x}) = \sum_{i=1}^{N} c_i \mathcal{N}(\bar{\mathbf{x}}_i, \sum_i) \tag{14.6}$$

对整数 N 和正计数器 c_i，$\sum_{i=1}^{N} c_i = 1$。对 $i = 1, 2, \cdots, N$，项 $\mathcal{N}(\bar{\mathbf{x}}_i, \sum_i)$ 表示均值为 $\bar{\mathbf{x}}_i$，协方差矩阵为 $\sum_i$ 的高斯（正态）密度函数。

式(14.6)等号右边的高斯和，随着项数 N 的增加，一致收敛到给定的概率密度函数 $p_x(\mathbf{x})$，且对所有的 i，协方差矩阵 $\sum_i$ 趋于 0（Anderson and Moore，1971）。对给定的概率密度函数 $p(\mathbf{x})$，计算式(14.6)的高斯求和近似，例如可以使用基于期望最大化（EM）算法的程序；该算法的内容在第 11 章介绍过。已经计算得到近似值，可以使用一组卡尔曼滤波器，解决用线性、非高斯的模型描述的逐次状态估计问题（Alspach and Sorenson，1972）。然而，注意到高斯和模型的项，随着时间按指数级增大，需要使用修剪算法。

3. 非线性、高斯模型。在复杂性增加的状态空间模型的分类体系中，第三个模型用公式表示如下：

$$\mathbf{x}_{n+1} = \mathbf{a}_n(\mathbf{x}_n) + \boldsymbol{\omega}_n \tag{14.7}$$

和

$$\mathbf{y}_n = \mathbf{b}_n(\mathbf{x}_n) + \boldsymbol{v}_n \tag{14.8}$$

假设动态噪声 $\boldsymbol{\omega}_n$ 和测量噪声 $\boldsymbol{v}_n$ 都是加性和服从高斯分布的。这里就是解决逐次状态估计问题的数学上困难的开始。计算该问题的近似解，主要有两个完全不同的方法：

1. 局部近似。在非线性滤波的第一种方法中，式(14.7)的系统模型的非线性函数 $\mathbf{a}_n(\cdot)$ 和式(14.8)的测量模型中的非线性函数 $\mathbf{b}_n(\cdot)$，靠这两个线性等式，近似于状态的局部估计值。接着应用卡尔曼滤波器计算近似解。14.5 节介绍的扩展的卡尔曼滤波器，是对非线性滤波器的局部近似方法的例子。

2. 全局近似。在非线性滤波的第二种方法中，解用贝叶斯估计结构的公式表示，通过这种方法，使得问题固有困难的解释在数学上易于处理。

3. 非线性、非高斯模型。式(14.1) 和式(14.2)描述了状态空间模型的最后一种类型，系统模型和测量模型都是非线性的，动态噪声 $\boldsymbol{\omega}_n$ 和测量噪声 $\boldsymbol{v}_n$ 不仅是非高斯的，而且可能是非加性的。在这种情况下，粒子滤波器是当前选择的方法，但不是解决逐次状态估计问题的唯一选择。

14.3　卡尔曼滤波器

式(14.4)、式(14.5)定义了卡尔曼滤波器的状态空间模型。此线性高斯模型中涉及的参数如下：

- 状态转移矩阵 $\mathbf{A}_{n+1,n}$，它是可逆的。
- 测量矩阵 $\mathbf{B}_n$，通常情况下它是长方形矩阵。
- 高斯动态噪声 $\boldsymbol{\omega}_n$，假设它具有零均值且有协方差矩阵 $\mathbf{Q}_{\omega,n}$。
- 高斯测量噪声 $\boldsymbol{v}_n$，假设它具有零均值且有协方差矩阵 $\mathbf{Q}_{v,n}$。

假设上述所有参数均已知。并给定一组观测值 $\{\mathbf{y}_k\}_{k=1}^{n}$。现要求最小均方误差意义下状态 $\mathbf{x}_k$ 的最优估计值。我们将滤波的讨论限定在 $k=n$，单步预测 $k=n+1$ 的情况。

新息过程

处理此类优化估计问题的一个有效办法，是利用关于观测量 $\mathbf{y}_n$ 的所谓的新息过程。其定

义如下：

$$\boldsymbol{\alpha}_n = \mathbf{y}_n - \hat{\mathbf{y}}_{n|n-1} \tag{14.9}$$

其中$\hat{\mathbf{y}}_{n|n-1}$是在给定至 $n-1$ 时刻（包括 $n-1$ 时刻）所有观测值的情况下，对 $\mathbf{y}_n$ 的最小均方差的估计。实际上，我们可以说：

新息过程 $\boldsymbol{\alpha}_n$ 是包含在测量值 $\mathbf{y}_n$ 但不在$\hat{\mathbf{y}}_{n|n-1}$的预测部分的新信息的测量，因为 $\mathbf{y}_n$ 可以预测的部分(记为$\hat{\mathbf{y}}_{n|n-1}$)是完全由序列 $\{\mathbf{y}_k\}_{k=1}^{n-1}$决定的。

新息过程有如下重要的性质：

性质 1　与观测值 $\mathbf{y}_n$ 有关的新息过程 $\boldsymbol{\alpha}_n$ 与之前的所有观测值 $\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_{n-1}$ 正交，表示为：

$$\mathbb{E}[\boldsymbol{\alpha}_n \mathbf{y}_k^T] = \mathbf{0}, \quad 1 \leqslant k \leqslant n-1 \tag{14.10}$$

性质 2　新息过程由一系列相互正交的随机向量构成，表示为：

$$\mathbb{E}[\boldsymbol{\alpha}_n \boldsymbol{\alpha}_k^T] = \mathbf{0}, \quad 1 \leqslant k \leqslant n-1 \tag{14.11}$$

性质 3　代表观测数据的随机向量序列 $\{\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_{n-1}\}$，与表示更新过程的序列$\{\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \cdots, \boldsymbol{\alpha}_n\}$一一对应。因此，通过能够保证线性稳定并且不丢失任何信息的操作，可以从一个序列得到另一个序列。因此可写作：

$$\{\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n\} \Longleftrightarrow \{\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \cdots, \boldsymbol{\alpha}_n\} \tag{14.12}$$

鉴于上述特性，就能理解为什么使用更新过程比使用观测值本身要简单：总的来说，观测量是相关的，而与之对应的更新过程中的部分是无关的。

新息过程的协方差矩阵

从初始状态 $\mathbf{x}_0$ 开始，我们可以用式(14.4)所描述的系统模型表示 k 时刻的系统状态：

$$\mathbf{x}_k = \mathbf{A}_{k,0}\mathbf{x}_0 + \sum_{i=1}^{k-1} A_{k,i}\boldsymbol{\omega}_i \tag{14.13}$$

式(14.13)表明状态 $\mathbf{x}_k$ 是 $\mathbf{x}_0$ 以及 $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \cdots, \boldsymbol{\omega}_n$ 的线性组合。

根据假设，测量噪声 $\boldsymbol{v}_n$ 与初始状态 $\mathbf{x}_0$ 以及动态噪声 $\boldsymbol{\omega}_i$ 无关。因此，在式(14.13)两边同乘以 $\boldsymbol{v}_n^T$ 后得到：

$$\mathbb{E}[\mathbf{x}_k \boldsymbol{v}_n^T] = \mathbf{0}, \quad k, n \geqslant 0 \tag{14.14}$$

同理，我们可以从测量公式(14.5)得到：

$$\mathbb{E}[\mathbf{y}_k \boldsymbol{v}_k^T] = \mathbf{0}, \quad 0 \leqslant k \leqslant n-1 \tag{14.15}$$

和

$$\mathbb{E}[\mathbf{y}_k \boldsymbol{\omega}_n^T] = \mathbf{0}, \quad 0 \leqslant k \leqslant n \tag{14.16}$$

给定先前的观测值 $\mathbf{y}_1, \cdots, \mathbf{y}_{n-1}$，我们可以从测量公式(14.5)中得出当前观测值 $\mathbf{y}_n$ 的最小均方估计为：

$$\hat{\mathbf{y}}_{n|n-1} = \mathbf{B}_n \hat{\mathbf{x}}_{n|n-1} + \hat{\boldsymbol{v}}_{n|n-1} \tag{14.17}$$

其中$\hat{\boldsymbol{v}}_{n|n-1}$是给定先前的观测值 $\mathbf{y}_1, \cdots, \mathbf{y}_{n-1}$ 后所对应的测量噪声估计。因为根据式(14.15)，$\boldsymbol{v}_n$ 与先前的观测值是正交的，因此估计值$\hat{\boldsymbol{v}}_{n|n-1}$为零。于是化简式(14.17)得到：

$$\hat{\mathbf{y}}_{n|n-1} = \mathbf{B}_n \hat{\mathbf{x}}_{n|n-1} \tag{14.18}$$

将式(14.5) 和式(14.18)代入式(14.9)，将项合并得：

$$\boldsymbol{\alpha}_n = \mathbf{B}_n \boldsymbol{\varepsilon}_{n|n-1} + \boldsymbol{v}_n \tag{14.19}$$

其中，新引入的项 $\boldsymbol{\varepsilon}_{n|n-1}$是状态预测误差向量。其定义为：

$$\boldsymbol{\varepsilon}_{n,n-1} = \mathbf{x}_n - \hat{\mathbf{x}}_{n|n-1} \tag{14.20}$$

在习题 14.1 中，$\boldsymbol{\varepsilon}_{n|n-1}$与动态噪声 $\boldsymbol{\omega}_n$ 以及测量噪声 $\boldsymbol{v}_n$ 均是正交的。由此定义零均值新息过程 $\boldsymbol{\alpha}_n$ 的协方差矩阵为：

$$\mathbf{R}_n = \mathbb{E}[\boldsymbol{\alpha}_n \boldsymbol{\alpha}_n^T] \tag{14.21}$$

利用式(14.19)，我们容易得到：

$$\mathbf{R}_n = \mathbf{B}_n \mathbf{P}_{n|n-1} \mathbf{B}_n^T + \mathbf{Q}_{v,n} \tag{14.22}$$

其中 $\mathbf{Q}_{v,n}$是测量噪声 $\boldsymbol{v}_n$ 的协方差矩阵，新引入的项

$$\mathbf{P}_{n|n-1} = \mathbb{E}[\boldsymbol{\varepsilon}_{n|n-1} \boldsymbol{\varepsilon}_{n|n-1}^T] \tag{14.23}$$

为预测误差协方差矩阵。式(14.22)是我们理解卡尔曼滤波算法的第一步。

利用新息过程进行滤波状态估计：预测-修正公式

下一步的任务是利用新息过程实现任意时刻 i 系统状态 $\mathbf{x}_i$ 的最小均方误差估计。为此，给定新息序列 $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \cdots, \boldsymbol{\alpha}_m$，我们首先线性展开的形式表示对状态 $\mathbf{x}_i$ 的估计：

$$\hat{\mathbf{x}}_{i|n} = \sum_{k=1}^{n} \mathbf{C}_{i,k} \boldsymbol{\alpha}_k \tag{14.24}$$

其中 $\{\mathbf{C}_{i,k}\}_{k=1}^n$是 i 时刻的展开式系数矩阵的集合。状态预测误差与新息过程满足下述正交条件(参见习题 14.3)：

$$\mathbb{E}[\boldsymbol{\varepsilon}_{i|n} \boldsymbol{\alpha}_k^T] = \mathbf{0} \quad \text{当 } k = 1,2,\cdots,n \text{ 且 } i \leqslant n \tag{14.25}$$

因此，将式(14.24)代入式(14.25)并使用式(14.11)所描述的新息过程的正交性，可得：

$$\mathbb{E}[\mathbf{x}_i \boldsymbol{\alpha}_k^T] = \mathbf{C}_{i,k} \mathbf{R}_k$$

其中，根据先前定义，$\mathbf{R}_n$ 是新息过程的协方差矩阵。解此方程的系数矩阵 $\mathbf{C}_{i,k}$，得到：

$$\mathbf{C}_{i,k} = \mathbb{E}[\mathbf{x}_i \boldsymbol{\alpha}_k^T] \mathbf{R}_k^{-1}$$

再利用式(14.24)中的表示方法得：

$$\hat{\mathbf{x}}_{i|n} = \sum_{k=1}^{n} \mathbb{E}[\mathbf{x}_i \boldsymbol{\alpha}_k^T] \mathbf{R}_k^{-1} \boldsymbol{\alpha}_k \tag{14.26}$$

当 $i=n$ 时，为滤波过程，因此可用式(14.26)描述该状态的滤波估计为：

$$\hat{\mathbf{x}}_{n|n} = \sum_{k=1}^{n} \mathbb{E}[\mathbf{x}_n \boldsymbol{\alpha}_k^T] \mathbf{R}_k^{-1} \boldsymbol{\alpha}_k = \sum_{k=1}^{n-1} \mathbb{E}[\mathbf{x}_n \boldsymbol{\alpha}_k^T] \mathbf{R}_k^{-1} \boldsymbol{\alpha}_k + \mathbb{E}[\mathbf{x}_n \boldsymbol{\alpha}_n^T] \mathbf{R}_n^{-1} \boldsymbol{\alpha}_n \tag{14.27}$$

在等式的第二行，$k=n$ 的项从求和中分离了出来。为了将式(14.27)转换为更理解的形式，我们首先用式(14.26)

$$\hat{\mathbf{x}}_{n|n-1} = \sum_{k=1}^{n-1} \mathbb{E}[\mathbf{x}_n \boldsymbol{\alpha}_k^T] \mathbf{R}_k^{-1} \boldsymbol{\alpha}_k \tag{14.28}$$

为了简化式(14.27)的第二项，我们引入下述定义：

$$\mathbf{G}_n = \mathbb{E}[\mathbf{x}_n \boldsymbol{\alpha}_n^T] \mathbf{R}_n^{-1} \tag{14.29}$$

由此，我们可以将状态滤波估计表示为下述递归的形式：

$$\hat{\mathbf{x}}_{n|n} = \hat{\mathbf{x}}_{n|n-1} + \mathbf{G}_n \boldsymbol{\alpha}_n \tag{14.30}$$

式(14.30)等号右边的两项意义如下：

1. $\hat{\mathbf{x}}_{n|n-1}$表示单步预测，其表示在给定 $n-1$ 时刻前（包括 $n-1$ 时刻）所有观测值的基础上对状态 $\mathbf{x}_n$ 的预测估计。

2. $\mathbf{G}_n \boldsymbol{\alpha}_n$ 表示修正项，新息过程 $\boldsymbol{\alpha}_n$ 表示由观测值 $\mathbf{y}_n$ 引入滤波过程的新信息，乘以“增益因子” $\mathbf{G}_n$。因此，$\mathbf{G}_n$ 通常被称为卡尔曼增益，以纪念卡尔曼在 1960 年发表的文章中所做出的突出贡献。

根据上述两点，式(14.30)在卡尔曼滤波器理论中被称为预测-修正公式。

卡尔曼增益的计算

式(14.30)是我们拥有的第二个用于卡尔曼滤波器递归计算的公式。然而，为了让这一公式具备使用价值，我们需要计算卡尔曼增益的公式。该公式能够用于状态估计中的递归计算。

有了这一目标，我们可以应用式(14.19)得：

$$\mathbb{E}[\mathbf{x}_n\boldsymbol{\alpha}_n^T] = \mathbb{E}[\mathbf{x}_n(\mathbf{B}_n\boldsymbol{\varepsilon}_{n|n-1} + \boldsymbol{v}_n)^T] = \mathbb{E}[\mathbf{x}_n\boldsymbol{\varepsilon}_{n|n-1}^T]\mathbf{B}_n^T$$

在上式的第二行，我们利用了状态 $\mathbf{x}_n$ 与测量噪声 $\mathbf{v}_n$ 无关性。注意到，根据正交原理，状态预测误差向量 $\boldsymbol{\varepsilon}_{n|n-1}$ 与状态估计 $\hat{\mathbf{x}}_{n|n-1}$ 是正交的。因此，$\boldsymbol{\varepsilon}_{n|n-1}$ 与 $\hat{\mathbf{x}}_{n|n-1}$ 外积的期望为零，进而我们用 $\boldsymbol{\varepsilon}_{n|n-1}$ 代替 $\mathbf{x}_n$ 不影响期望值 $\mathbb{E}[\mathbf{x}_n\boldsymbol{\alpha}_n^T]$。由此可得：

$$\mathbb{E}[\mathbf{x}_n\boldsymbol{\alpha}_n^T] = \mathbb{E}[\boldsymbol{\varepsilon}_{n|n-1}\boldsymbol{\varepsilon}_{n|n-1}^T]\mathbf{B}_n^T = \mathbf{P}_{n|n-1}\mathbf{B}_n^T$$

所以，对式(14.29)中 $\mathbb{E}[\mathbf{x}_n\boldsymbol{\alpha}_n^T]$ 一项使用这一公式，我们可以用预测误差协方差矩阵 $\mathbf{P}_{n|n-1}$ 将卡尔曼增益 $\mathbf{G}_n$ 表示为：

$$\mathbf{G}_n = \mathbf{P}_{n|n-1}\mathbf{B}_n^T\mathbf{R}_n^{-1} \tag{14.31}$$

这就是卡尔曼滤波器递归算法所需的第三个等式。

用于更新预测误差协方差矩阵的黎卡堤（Riccati）差分方程

为了完成卡尔曼滤波器的递归计算过程，我们需要一个迭代公式，从一个迭代到下一次迭代中更新预测误差协方差矩阵。

为了解决这一状态估计过程中的最后一步，我们在式(14.20)中用 $n+1$ 替代 n 得到：

$$\boldsymbol{\varepsilon}_{n+1|n} = \mathbf{x}_{n+1} - \hat{\mathbf{x}}_{n+1|n}$$

随后我们发现用含有滤波估计的项表示状态的预测估计是有益的。故而将式(14.28)中的 n 替换为 $n+1$ 并应用式(14.4)，可得：

$$\begin{aligned}\hat{\mathbf{x}}_{n+1|n} &= \sum_{k=1}^{n}\mathbb{E}[\mathbf{x}_{n+1}\boldsymbol{\alpha}_k^T]\mathbf{R}_k^{-1}\boldsymbol{\alpha}_k = \sum_{k=1}^{n}\mathbb{E}[(\mathbf{A}_{n+1,n}\mathbf{x}_n + \boldsymbol{\omega}_n)\boldsymbol{\alpha}_k^T]\mathbf{R}_k^{-1}\boldsymbol{\alpha}_k \\ &= \mathbf{A}_{n+1,n}\sum_{k=1}^{n}\mathbb{E}[\mathbf{x}_n\boldsymbol{\alpha}_n^T]\mathbf{R}_k^{-1}\boldsymbol{\alpha}_k = \mathbf{A}_{n+1,n}\hat{\mathbf{x}}_{n|n}\end{aligned} \tag{14.32}$$

式(14.32)的第一行，因为动态噪声 $\boldsymbol{\omega}_n$ 与观测值是相互独立的，故期望 $\mathbb{E}[\boldsymbol{\omega}_n\boldsymbol{\alpha}_k^T]$ 为零。对滤波估计 $\hat{\mathbf{x}}_{n|n}$，应用式(14.27)的定义公式的第一行，以及式(14.32)和对状态 $\mathbf{x}_n$ 的预测滤波估计的关系，利用 $\boldsymbol{\varepsilon}_{n+1|n}$ 的公式得到：

$$\boldsymbol{\varepsilon}_{n+1|n} = \underbrace{(\mathbf{A}_{n+1,n}\mathbf{x}_n + \boldsymbol{\omega}_n)}_{\text{状态}\mathbf{x}_{n+1}} - \underbrace{\mathbf{A}_{n+1,n}\hat{\mathbf{x}}_{n|n}}_{\substack{\text{预测估计}\\ \hat{\mathbf{x}}_{n+1|n}}} = \mathbf{A}_{n+1,n}(\mathbf{x}_n - \hat{\mathbf{x}}_{n|n}) + \boldsymbol{\omega}_n = \mathbf{A}_{n+1,n}\boldsymbol{\varepsilon}_{n|n} + \boldsymbol{\omega}_n \tag{14.33}$$

其中，滤波误差向量的定义为：

$$\boldsymbol{\varepsilon}_{n|n} = \mathbf{x}_n - \hat{\mathbf{x}}_{n|n} \tag{14.34}$$

因为滤波误差向量 $\boldsymbol{\varepsilon}_{n|n}$ 与动态噪声 $\boldsymbol{\omega}_n$ 是无关的，我们可以将预测误差协方差矩阵表示为：

$$\mathbf{P}_{n+1,n} = \mathbb{E}[\boldsymbol{\varepsilon}_{n+1|n}\boldsymbol{\varepsilon}_{n+1|n}^T] = \mathbf{A}_{n+1,n}\mathbf{P}_{n|n}\mathbf{A}_{n+1,n}^T + \mathbf{Q}_{\omega,n} \tag{14.35}$$

其中 $\mathbf{Q}_{\omega,n}$ 为动态噪声 $\boldsymbol{\omega}_n$ 的误差协方差矩阵。在式(14.35)中我们引入了最后一个参数，称为*滤波误差协方差矩阵*，其定义为：

$$\mathbf{P}_{n|n} = \mathbb{E}[\boldsymbol{\varepsilon}_{n|n}\boldsymbol{\varepsilon}_{n|n}^T] \tag{14.36}$$

为了完成卡尔曼滤波算法的递归循环，我们需要用于计算滤波误差协方差矩阵 $\mathbf{P}_{n|n}$ 的式子。因此我们首先将式(14.30)代入式(13.34)得：

$$\boldsymbol{\varepsilon}_{n|n} = \mathbf{x}_n - \hat{\mathbf{x}}_{n|n-1} - \mathbf{G}_n\boldsymbol{\alpha}_n = \boldsymbol{\varepsilon}_{n|n-1} - \mathbf{G}_n\boldsymbol{\alpha}_n$$

然后应用式(14.36)，得到：

$$\begin{aligned}\mathbf{P}_{n|n} &= \mathbb{E}[(\boldsymbol{\varepsilon}_{n|n-1} - \mathbf{G}_n\boldsymbol{\alpha}_n)(\boldsymbol{\varepsilon}_{n|n-1} - \mathbf{G}_n\boldsymbol{\alpha}_n)^T] \\ &= \mathbb{E}[\boldsymbol{\varepsilon}_{n|n-1}\boldsymbol{\varepsilon}_{n|n-1}^T] - \mathbf{G}_n\,\mathbb{E}[\boldsymbol{\alpha}_n\boldsymbol{\varepsilon}_{n|n-1}^T] - \mathbb{E}[\boldsymbol{\varepsilon}_{n|n-1}\boldsymbol{\alpha}_n^T]\mathbf{G}_n^T + \mathbf{G}_n\,\mathbb{E}[\boldsymbol{\alpha}_n\boldsymbol{\alpha}_n^T]\mathbf{G}_n^T \\ &= \mathbf{P}_{n|n-1} - \mathbf{G}_n\,\mathbb{E}[\boldsymbol{\alpha}_n\boldsymbol{\varepsilon}_{n|n-1}^T] - \mathbb{E}[\boldsymbol{\varepsilon}_{n|n-1}\boldsymbol{\alpha}_n^T]\mathbf{G}_n^T + \mathbf{G}_n\mathbf{R}_n\mathbf{G}_n^T\end{aligned} \tag{14.37}$$

接着，我们注意到因为$\hat{\mathbf{x}}_{n|n-1}$与新息过程$\boldsymbol{\alpha}_n$正交，于是可得：

$$\mathbb{E}[\boldsymbol{\varepsilon}_{n|n-1}\boldsymbol{\alpha}_n^T] = \mathbb{E}[(\mathbf{x}_n - \hat{\mathbf{x}}_{n|n-1})\boldsymbol{\alpha}_n^T] = \mathbb{E}[\mathbf{x}_n\boldsymbol{\alpha}_n^T]$$

同理：

$$\mathbb{E}[\boldsymbol{\alpha}_n\boldsymbol{\varepsilon}_{n|n-1}^T] = \mathbb{E}[\boldsymbol{\alpha}_n\mathbf{x}_n^T]$$

利用这一对关系以及式(14.29)中对卡尔曼增益的定义，易得：

$$\mathbf{G}_n\,\mathbb{E}[\boldsymbol{\alpha}_n\boldsymbol{\varepsilon}_{n|n-1}^T] = \mathbb{E}[\boldsymbol{\varepsilon}_{n|n-1}\boldsymbol{\alpha}_n^T]\mathbf{G}_n^T = \mathbf{G}_n\mathbf{R}_n\mathbf{G}_n^T$$

根据式(14.37)化简得：

$$\mathbf{P}_{n|n} = \mathbf{P}_{n|n-1} - \mathbf{G}_n\mathbf{R}_n\mathbf{G}_n^T$$

最后我们应用卡尔曼增益的式(14.31)以及协方差矩阵$\mathbf{R}_n$和$\mathbf{P}_{n|n-1}$的对称性得到：

$$\mathbf{P}_{n|n} = \mathbf{P}_{n|n-1} - \mathbf{G}_n\mathbf{B}_n\mathbf{P}_{n|n-1} \tag{14.38}$$

至此，我们得到了式(14.38)和式(14.35)这一对更新预测误差协方差矩阵的重要公式。特别是式(14.38)，其通常被看做控制论中著名的黎卡堤方程的离散形式。

这一对等公式连同式(14.32)完成了卡尔曼滤波算法的公式化。

卡尔曼滤波器总结

表 14.1 列出了解决卡尔曼滤波问题所涉及的所有变量和参数。滤波器的输入是一系列的观测值$\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n$，输出是滤波估计$\hat{\mathbf{x}}_{n|n}$。其计算过程是递归的，详见表 14.2。为递归计算所需的初始条件也一并被列出。需要注意的是表 14.2 中，新息过程$\boldsymbol{\alpha}_n$的计算公式是根据式(14.9)和式(14.18)得出的。

表 14.1　卡尔曼变量和参数总结

变量	定义	维数
$\mathbf{x}_n$	n 时刻的状态	$M\times 1$
$\mathbf{y}_n$	n 时刻的观测值	$L\times 1$
$\mathbf{A}_{n+1,n}$	n 时刻的状态转移到 $n+1$ 时刻的状态的可逆过渡矩阵	$M\times M$
$\mathbf{B}_n$	n 时刻的测量矩阵	$L\times M$
$\mathbf{Q}_{\omega,n}$	动态噪声 $\boldsymbol{\omega}_n$ 的协方差矩阵	$M\times M$
$\mathbf{Q}_{v,n}$	测量噪声 $\mathbf{v}_n$ 的协方差矩阵	$L\times L$
$\hat{\mathbf{x}}_{n\|n-1}$	n 时刻给定观测值 $\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_{n-1}$，状态的预测估计	$M\times 1$
$\hat{\mathbf{x}}_{n\|n}$	n 时刻给定观测值 $\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n$，状态的预测估计	$M\times 1$
$\mathbf{G}_n$	n 时刻的卡尔曼增益	$M\times L$
$\boldsymbol{\alpha}_n$	n 时刻的新息过程	$L\times 1$
$\mathbf{R}_n$	新息过程 $\boldsymbol{\alpha}_n$ 的协方差矩阵	$L\times L$
$\mathbf{P}_{n\|n-1}$	预测误差协方差矩阵	$M\times M$
$\mathbf{P}_{n\|n}$	滤波误差协方差矩阵	$M\times M$

表 14.2 总结出的卡尔曼滤波器通常被称做协方差（卡尔曼）滤波算法。这一术语来源于该算法需在一次完整的递归计算循环中传播表示预测的协方差矩阵$\mathbf{P}_{n|n-1}$。

表 14.2　基于滤波状态估计的卡尔曼滤波器总结

观测值 $=\{\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n\}$

已知的参数

过渡矩阵 $=\mathbf{A}_{n+1,n}$

测量矩阵 $=\mathbf{B}_n$

动态噪声的协方差矩阵 $=\mathbf{Q}_{\omega,n}$

测量噪声的协方差矩阵 $=\mathbf{Q}_{v,n}$

计算：$n=1, 2, 3, \cdots$

$$\mathbf{G}_n=\mathbf{P}_{n|n-1}\mathbf{B}_n^T[\mathbf{B}_n\mathbf{P}_{n|n-1}\mathbf{B}_n^T+\mathbf{Q}_{v,n}]^{-1}$$

$$\boldsymbol{\alpha}_n=\mathbf{y}_n-\mathbf{B}_n\hat{\mathbf{x}}_{n|n-1}$$

$$\hat{\mathbf{x}}_{n|n}=\hat{\mathbf{x}}_{n|n-1}+\mathbf{G}_n\boldsymbol{\alpha}_n$$

$$\hat{\mathbf{x}}_{n+1|n}=\mathbf{A}_{n+1,n}\hat{\mathbf{x}}_{n|n}$$

$$\mathbf{P}_{n|n}=\mathbf{P}_{n|n-1}-\mathbf{G}_n\mathbf{B}_n\mathbf{P}_{n|n-1}$$

$$\mathbf{P}_{n+1|n}=\mathbf{A}_{n+1,n}\mathbf{P}_{n|n}\mathbf{A}_{n+1,n}^T+\mathbf{Q}_{\omega,n}$$

初始条件：

$$\hat{\mathbf{x}}_{1|0}=\mathbb{E}[\mathbf{x}_1]$$

$$\mathbf{P}_{1,0}=\mathbb{E}[(\mathbf{x}_1-\mathbb{E}[\mathbf{x}_1])(\mathbf{x}_1-\mathbb{E}[\mathbf{x}_1])^T]=\Pi_0$$

矩阵 Π_0 是对角阵，对角线上的元素均为 δ^{-1}，δ 是一个很小的数。

图 14.3 是卡尔曼滤波器的信号流程图，其中 $z^{-1}\mathbf{I}$ 表示一组单位延时。从这幅图可以清楚地看出卡尔曼滤波器是一个双回路反馈系统。其中一个反馈回路包括了系统（状态）模型的状态转移矩阵 $\mathbf{A}_{n,n-1}$，起预测作用。第二个反馈回路包括了测量模型中的矩阵 $\mathbf{B}_n$，起修正作用。这两个反馈回路一起作用产生对 $\mathbf{x}_n$ 的滤波状态估计，即输出与观测值 $\mathbf{y}_n$ 对应的 $\hat{\mathbf{x}}_{n|n}$。除此之外，正如图 14.3 描绘的那样，卡尔曼滤波器还是一个可以用于实时应用的系统。实际上，我们也有包括上述两个反馈回路的全局的反馈回路。

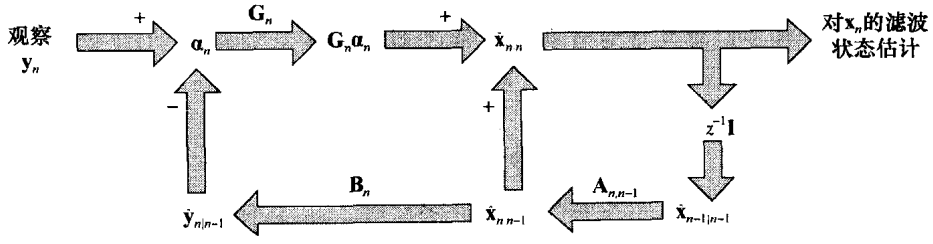


图 14.3　卡尔曼滤波器的信号流程图，用一个双回路反馈系统来描述

由于卡尔曼滤波器的关键部分卡尔曼增益 $\mathbf{G}_n$ 会随时间 n 的改变而变化，因此我们说卡尔曼滤波器是随时间变化的滤波器。即使在原始动态系统的状态空间模型具备时间不变特性的情况下，这一性质依然存在。

14.4　发散现象及平方根滤波

表 14.2 所总结的协方差滤波算法容易遇到数值困难，在一些文献中已经充分说明（Kaminski 等，1971；Bierman 和 Thornton，1977）。

在实际应用中，有两种基本的途径能导致数值困难。一个是*数值不精确*。具体来说，如式(14.38)所示，矩阵 $\mathbf{P}_{n|n}$ 是两个非负定矩阵的差值。因此，除非算法中的每一次循环都能保证足够高的数值精度，才有可能使得计算结果的矩阵满足对称性和非负定性。而根据式(14.36)，$\mathbf{P}_{n|n}$ 是协方差矩阵，其必须满足非负定性。因此实际应用与理论间产生了矛盾，计算过程中数值误差的存在将导致卡尔曼滤波器行为的不稳定。卡尔曼滤波器的这种非稳定行为通常称为*发散现象*。

在实际应用中还有另一途径可能导致发散现象。卡尔曼滤波器的导出过程基于式(14.4)、式(14.5)所描述的线性高斯状态空间模型。而这一模型源于尚在研究中的动态系统底层理论，此亦可能导致该算法的不稳定。虽然算法是由现实的观测值序列驱动的，但是算法的数学导出是基于假设的状态空间模型。因此，实际应用与理论再次产生矛盾，并由此可能导致前述的算法的发散。

考虑到这些实际关系，我们或许会提出这样的疑问：

在实际应用中，如何克服发散现象以确保卡尔曼滤波器的操作是稳定的呢？

下面将讨论这一重要问题的实际解决办法。

平方根滤波

一个数学上优美且计算上可行的，解决发散问题的方法就是利用平方根滤波。其思想是对卡尔曼滤波器进行修正，在算法的每一次循环中使用数值稳定的正交变换。具体而言，应用乔里斯基分解可以将 $\mathbf{P}_{n|n}$ 转换为其平方根的形式，由此可得：

$$\mathbf{P}_{n|n} = \mathbf{P}_{n|n}^{1/2}\mathbf{P}_{n|n}^{T/2} \tag{14.39}$$

其中，$\mathbf{P}_{n|n}^{1/2}$ 是一个下三角矩阵，$\mathbf{P}_{n|n}^{T/2}$ 是其转置。在线性代数中，通常将乔里斯基因子 $\mathbf{P}_{n|n}^{1/2}$ 认为是矩阵 $\mathbf{P}_{n|n}$ 的平方根。需要特别注意的是矩阵积 $\mathbf{P}_{n|n}^{1/2}\mathbf{P}_{n|n}^{T/2}$ 可能是不定的，因为任意矩阵和其转置矩阵的乘积始终是非负定的。正因为如此，即便存在数值误差，通常乔里斯基系数 $\mathbf{P}_{n|n}^{1/2}$ 仍然优于 $\mathbf{P}_{n|n}$ 本身。

卡尔曼滤波器的平方根实现

线性代数中的矩阵分解引理是平方根滤波算法的关键。设有任意两个 $L\times M$ 维的矩阵 $\mathbf{X}$ 和 $\mathbf{Y}$，其中 $L\leqslant M$，则矩阵分解引理的表述如下（Stewart，1973；Golub and Van Loan，1996）：

等式 $\mathbf{X}\mathbf{X}^T=\mathbf{Y}\mathbf{Y}^T$ 成立，当且仅当存在正交矩阵 $\boldsymbol{\Theta}$，使得

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\Theta} \tag{14.40}$$

为了证明这一引理，我们可以将矩阵积 $\mathbf{Y}\mathbf{Y}^T$ 表示为

$$\mathbf{Y}\mathbf{Y}^T = \mathbf{X}\boldsymbol{\Theta}(\mathbf{X}\boldsymbol{\Theta})^T = \mathbf{X}\boldsymbol{\Theta}\boldsymbol{\Theta}^T\mathbf{X}^T = \mathbf{X}\mathbf{X}^T$$

在上式的最后一行，我们使用了正交矩阵 $\boldsymbol{\Theta}$ 的性质，即

正交矩阵与其转置的积是单位矩阵

由这一性质可推出：

$$\boldsymbol{\Theta}^{-1} = \boldsymbol{\Theta}^T \tag{14.41}$$

即正交矩阵的逆矩阵等于它的转置矩阵。

有了上述引理，我们可以开始讨论卡尔曼滤波器的平方根协方差实现问题。首先，我们应用式(14.31)及式(14.38)中对增益矩阵 $\mathbf{G}_n$ 的定义，可得：

$$\mathbf{P}_{n|n} = \mathbf{P}_{n|n-1} - \mathbf{P}_{n|n-1}\mathbf{B}_n^T\mathbf{R}_n^{-1}\mathbf{B}_n\mathbf{P}_{n|n-1} \tag{14.42}$$

其中 $\mathbf{R}_n$ 的定义由式(14.22)给出。为了表示的方便将上式重写为：

$$\mathbf{R}_n = \mathbf{B}_n\mathbf{P}_{n|n-1}\mathbf{B}_n^T + \mathbf{Q}_{v,n}$$

观察式(14.42)重新用公式表示出的黎卡提微分方程，我们可以发现等号的右端包含了 3 个不同的矩阵项：

$M\times L$ 维的矩阵：预测状态 $\mathbf{P}_{n|n-1}$ 的协方差矩阵

$L\times M$ 维的矩阵：乘上 $\mathbf{P}_{n|n-1}$ 的测量矩阵 $\mathbf{B}_n$

$L\times L$ 维的矩阵：新息过程的协方差矩阵 $\mathbf{R}_n$

在牢记这三个矩阵项不同维数的同时，用一致的形式将其整合在一个 $N\times N$ 的分块矩阵中

$$\mathbf{H}_n=\left[\begin{array}{c|c}\mathbf{R}_n & \mathbf{B}_n\mathbf{P}_{n|n-1}\\ \hline \mathbf{P}_{n|n-1}\mathbf{B}_n^T & \mathbf{P}_{n|n-1}\end{array}\right]=\left[\begin{array}{c|c}\mathbf{Q}_{v,n}+\mathbf{B}_n\mathbf{P}_{n|n-1}\mathbf{B}_n^T & \mathbf{B}_n\mathbf{P}_{n|n-1}\\ \hline \mathbf{P}_{n|n-1}\mathbf{B}_n^T & \mathbf{P}_{n|n-1}\end{array}\right] \tag{14.43}$$

其中，在第二个等号后，加入了关于 $\mathbf{R}_n$ 的式子。式(14.33)中的矩阵为 $N\times N$ 维的方阵，$N=L+M$。根据定义，新的分块矩阵 $\mathbf{H}_n$ 是非负定的。因此我们对其进行乔里斯基分解，可得：

$$\mathbf{H}_n=\left[\begin{array}{c|c}\mathbf{Q}_{v,n}^{1/2} & \mathbf{B}_n\mathbf{P}_{n|n-1}^{1/2}\\ \hline \mathbf{O} & \mathbf{P}_{n|n-1}^{1/2}\end{array}\right]\left[\begin{array}{c|c}\mathbf{Q}_{v,n}^{1/2} & \mathbf{O}^T\\ \hline \mathbf{P}_{n|n-1}^{1/2}\mathbf{B}_n^T & \mathbf{P}_{n|n-1}^{1/2}\end{array}\right] \tag{14.44}$$

其中，$\mathbf{P}_{n|n-1}^{1/2}$是协方差矩阵 $\mathbf{P}_{n|n-1}$的平方根，$\mathbf{O}$ 是零矩阵。

式(14.44)等号右边的矩阵乘积可理解成矩阵 $\mathbf{X}_n$ 及其转置 $\mathbf{X}_n^T$ 的乘积。由此可知其满足应用矩阵分解引理的条件，根据该引理利用式(14.40)得：

$$\underbrace{\left[\begin{array}{c|c}\mathbf{Q}_{v,n}^{1/2} & \mathbf{B}_n\mathbf{P}_{n|n-1}^{1/2}\\ \hline \mathbf{O} & \mathbf{P}_{n|n-1}^{1/2}\end{array}\right]}_{\mathbf{X}_n}\boldsymbol{\Theta}_n=\underbrace{\left[\begin{array}{c|c}\mathbf{Y}_{11,n} & \mathbf{O}^T\\ \hline \mathbf{Y}_{21,n} & \mathbf{Y}_{22,n}\end{array}\right]}_{\mathbf{Y}_n} \tag{14.45}$$

式中矩阵 $\boldsymbol{\Theta}_n$ 为正交矩阵，更确切地说，$\boldsymbol{\Theta}_n$ 与 $\mathbf{X}_n$ 的乘积 $\mathbf{Y}_n$ 为下三角矩阵，即位于 $\mathbf{Y}_n$ 主对角线上方的元素均为零。因此，$\boldsymbol{\Theta}_n$ 通常称作正交旋转。利用 $\boldsymbol{\Theta}_n$ 正交的特性，可将式(14.45)展开为：

$$\underbrace{\left[\begin{array}{c|c}\mathbf{Q}_{v,n}^{1/2} & \mathbf{B}_n\mathbf{P}_{n|n-1}^{1/2}\\ \hline \mathbf{O} & \mathbf{P}_{n|n-1}^{1/2}\end{array}\right]}_{\mathbf{X}_n}\underbrace{\left[\begin{array}{c|c}\mathbf{Q}_{v,n}^{1/2} & \mathbf{O}^T\\ \hline \mathbf{P}_{n|n-1}^{1/2}\mathbf{B}_n^T & \mathbf{P}_{n|n-1}^{T/2}\end{array}\right]}_{\mathbf{X}_n^T}=\underbrace{\left[\begin{array}{c|c}\mathbf{Y}_{11,n} & \mathbf{O}^T\\ \hline \mathbf{Y}_{21,n} & \mathbf{Y}_{22,n}\end{array}\right]}_{\mathbf{Y}_n}\underbrace{\left[\begin{array}{c|c}\mathbf{Y}_{11,n}^T & \mathbf{O}_{21,n}^T\\ \hline \mathbf{O}^T & \mathbf{Y}_{22,n}^T\end{array}\right]}_{\mathbf{Y}_n^T} \tag{14.46}$$

展开矩阵的乘积 $\mathbf{X}_n\mathbf{X}_n^T$ 和 $\mathbf{Y}_n\mathbf{Y}_n^T$，建立等式两边相对应矩阵块之间的相等关系，得到三个式子：

$$\mathbf{Q}_{v,n}+\mathbf{B}_n\mathbf{P}_{n|n-1}\mathbf{B}_n^T=\mathbf{Y}_{11,n}\mathbf{Y}_{11,n}^T \tag{14.47}$$

$$\mathbf{B}_n\mathbf{P}_{n|n-1}=\mathbf{Y}_{11,n}\mathbf{Y}_{21,n}^T \tag{14.48}$$

$$\mathbf{P}_{n|n-1}=\mathbf{Y}_{21,n}\mathbf{Y}_{21,n}^T+\mathbf{Y}_{22,n}\mathbf{Y}_{22,n}^T \tag{14.49}$$

式(14.47)等号左边的项可视为协方差矩阵 $\mathbf{R}_n$，其可被分解为 $\mathbf{R}_n^{1/2}\mathbf{R}_n^{T/2}$。因此，式(14.47)中的第一个未知项满足：

$$\mathbf{Y}_{11,n}=\mathbf{R}_n^{1/2} \tag{14.50}$$

接着，将 $\mathbf{Y}_{11,n}$的值代入式(14.48)，解出 $\mathbf{Y}_{21,n}$，由此我们得到了第二个未知项的表达式：

$$\mathbf{Y}_{21,n}=\mathbf{P}_{n|n-1}\mathbf{B}_n^T\mathbf{R}_n^{-T/2} \tag{14.51}$$

根据前面卡尔曼增益 $\mathbf{G}_n$ 的定义和式(14.31)，也可以将 $\mathbf{Y}_{21,n}$表示为：

$$\mathbf{Y}_{21,n}=\mathbf{G}_n\mathbf{R}_n^{1/2} \tag{14.52}$$

再者，将式(14.51)中 $\mathbf{Y}_{21,n}$的值代入式(14.49)，计算矩阵积 $\mathbf{Y}_{22,n}\mathbf{Y}_{22,n}^T$，然后应用式(14.42)，我们可以得到：

$$\mathbf{Y}_{22,n}\mathbf{Y}_{22,n}^T=\mathbf{P}_{n|n-1}-\mathbf{P}_{n|n-1}\mathbf{B}_n^T\mathbf{R}_n^{-1}\mathbf{B}_n\mathbf{P}_{n|n-1}=\mathbf{P}_{n|n}$$

将协方差矩阵 $\mathbf{P}_{n,n}$分解为 $\mathbf{P}_{n|n}^{1/2}\mathbf{P}_{n|n}^{T/2}$，可以得到第三个未知项：

$$\mathbf{Y}_{22,n}=\mathbf{P}_{n|n}^{1/2} \tag{14.53}$$

在确定了 $\mathbf{Y}_n$ 的三个非零子矩阵后，我们可替换式(14.45)中的未知子矩阵，得到：

$$\left[\begin{array}{c|c}\mathbf{Q}_{v,n}^{1/2} & \mathbf{B}_n\mathbf{P}_{n,n-1}^{1/2}\\ \hline \mathbf{O} & \mathbf{P}_{n|n-1}^{1/2}\end{array}\right]\boldsymbol{\Theta}_n=\left[\begin{array}{c|c}\mathbf{R}_n^{1/2} & \mathbf{Q}^T\\ \hline \mathbf{G}_n\mathbf{R}_n^{1/2} & \mathbf{P}_{n|n}^{1/2}\end{array}\right] \tag{14.54}$$

在最终得到的式(14.54)中，仔细观察我们可以区分两个定义清楚的数值矩阵：

1. 前矩阵。此矩阵是位于式(14.54)等号左侧的数值矩阵。它与 $\boldsymbol{\Theta}_n$ 相乘的目的是逐个元

素的消去整个子矩阵 $\mathbf{B}_n\mathbf{P}_{n|n-1}^{1/2}$。测量矩阵 $\mathbf{B}_n$ 和测量噪声的协方差矩阵 $\mathbf{Q}_{v,n}$ 均是已知量。平方根 $\mathbf{P}_{n|n-1}^{1/2}$ 经数值更新后也是已知的。因此，在 n 时刻，组成前矩阵的所有子矩阵均是已知的。

2. *后矩阵*。此矩阵是位于式(14.54)等号右侧的数值矩阵。它是由前矩阵经正交旋转消去 $\mathbf{B}_n\mathbf{P}_{n|n-1}^{1/2}$ 后得到的下三角矩阵。特别地，在前矩阵中所包含的平方根 $\mathbf{Q}_{v,n}^{1/2}$ 产生了两个有用的矩阵：

- 矩阵 $\mathbf{R}_n^{1/2}$，表示了新息过程 $\boldsymbol{\alpha}_n$ 的协方差矩阵的平方根。
- 矩阵的乘积 $\mathbf{G}_n\mathbf{R}_n^{1/2}$，用于计算卡尔曼增益。

另一个由计算后矩阵而得到的重要的矩阵是滤波误差协方差矩阵的平方根 $\mathbf{P}_{n|n}^{1/2}$。

有了从后矩阵提取出的信息，我们可以对平方根协方差滤波算法中涉及的计算过程加以总结。其已在表 14.3 中列出。该算法一个完整的递归循环包括了前矩阵到后矩阵的变换以及各参数的更新计算。关于参数的更新已经在表中 3、4 两项分别列出。从表中可以很清楚地看出，该算法确实是在传播预测误差协方差矩阵的平方根 $\mathbf{P}_{n|n-1}^{1/2}$。

表 14.3　平方根滤波算法的计算总结

1. *已知的参数*：
   过渡矩阵：$\mathbf{A}_{n+1,n}$
   测量矩阵：$\mathbf{B}_n$
   测量噪声的协方差矩阵：$\mathbf{Q}_{v,n}$
   动态噪声的协方差矩阵：$\mathbf{Q}_{\omega,n}$
2. *待更新的参数值*：
   状态的预测估计：$\hat{\mathbf{x}}_{n|n-1}$
   预测误差协方差矩阵的平方根：$\mathbf{P}_{n|n-1}^{1/2}$
3. *将前矩阵变换为后矩阵的正交旋转*：

$$\begin{bmatrix} \mathbf{Q}_{v,n}^{1/2} & \mathbf{B}_n\mathbf{P}_{n,n-1}^{1/2} \\ \mathbf{0} & \mathbf{P}_{n|n-1}^{1/2} \end{bmatrix}\boldsymbol{\Theta}_n = \begin{bmatrix} \mathbf{R}_n^{1/2} & \mathbf{O}^T \\ \mathbf{G}_n\mathbf{R}_n^{1/2} & \mathbf{P}_{n|n}^{1/2} \end{bmatrix}$$

4. *已更新的参数*：

$$\mathbf{G}_n = [\mathbf{G}_n\mathbf{R}_n^{1/2}]\,[\mathbf{R}_n^{1/2}]^{-1}$$

$$\boldsymbol{\alpha}_n = \mathbf{y}_n - \mathbf{B}_n\hat{\mathbf{x}}_{n|n-1}$$

$$\hat{\mathbf{x}}_{n|n} = \hat{\mathbf{x}}_{n|n-1} + \mathbf{G}_n\boldsymbol{\alpha}_n$$

$$\hat{\mathbf{x}}_{n+1|n} = \mathbf{A}_{n+1|n}\hat{\mathbf{x}}_{n|n}$$

$$\mathbf{P}_{n|n} = \mathbf{P}_{n|n}^{1/2}[\mathbf{P}_{n|n}^{1/2}]^T$$

$$\mathbf{P}_{n+1|n} = [\mathbf{A}_{n+1|n}\mathbf{P}_{n|n}^{1/2} \,\vdots\, \mathbf{Q}_{\omega,n}^{1/2}] \begin{bmatrix} \mathbf{P}_{n|n}^{T/2}\mathbf{A}_{n+1|n}^T \\ \hline \mathbf{Q}_{\omega,n}^{T/2} \end{bmatrix}$$

说明：

1. 第 4 点中，方括号中所有的矩阵都是从后矩阵中提取出的，并且是已知的。
2. 书写已更新参数时，使用了表 14.2 的相关计算公式。

吉文斯旋转

到目前为止，在用公式表示平方根协方差滤波算法的过程中，我们更多地关注通过消去过程将前矩阵转换为下三角后矩阵，而忽略了如何确定正交矩阵 $\boldsymbol{\Theta}_n$。解决这一问题的巧妙方法就是利用吉文斯旋转的方法，具体实现是多步的（Golub 和 Van Loan，1996）。

在这一方法中，正交矩阵 $\boldsymbol{\Theta}_n$ 被表示为 N 个正交旋转的积，用下式表示：

$$\boldsymbol{\Theta} = \prod_{k=1}^{N}\boldsymbol{\Theta}_k$$

这里，我们不考虑离散时间 n 以简化其表述。每个正交旋转的特点如下：

1. $\boldsymbol{\Theta}_k$ 对角线上除四个关键元素外的其他元素均为 1，非对角线上的元素均为 0。

2. $\boldsymbol{\Theta}_k$ 的下标 k 称为关键点，围绕关键点定位 $\boldsymbol{\Theta}_k$ 的四个策略元素。由上条特性可知，关键点总是位于前矩阵的主对角线上。

3. 策略元素中的两个为余弦参数，另外两个为正弦参数。为了更为详细的阐述这些正、余弦参数的数学意义，现假设欲消去前矩阵的第 kl 个元素，其中 k 为行数 l 为列数。因此，对应的余弦参数（位于主对角线上）θ_{kk} 和 θ_{ll} 具有相同的值，而正弦参数（位于主对角线外）中的一个必须为负值，如下 2×2 矩阵所示：

$$\begin{bmatrix}\theta_{kk} & \theta_{kl}\\ \theta_{lk} & \theta_{ll}\end{bmatrix}=\begin{bmatrix}c_k & -s_k\\ s_k & c_k\end{bmatrix} \tag{14.55}$$

所有的四个参数均为实数值，并需满足以下约束：

$$c_k^2+s_k^2=1 \quad ,对于所有\ k \tag{14.56}$$

下面的例子演示了将前矩阵转换为下三角后矩阵的具体步骤。

例 1　3×3 前矩阵的吉文斯旋转

假设欲将 3×3 的前矩阵 **X** 转换为 3×3 的下三角后矩阵 **Y**，需经过三个步骤。

步骤一：在第一步中，计算

$$\underbrace{\left[\begin{array}{c|cc}x_{11} & x_{12} & x_{13}\\ \hline 0 & x_{22} & x_{23}\\ 0 & x_{32} & x_{33}\end{array}\right]}_{\text{第一步的前矩阵}}\quad\underbrace{\left[\begin{array}{c|cc}c_1 & -s_1 & 0\\ \hline s_1 & c_1 & 0\\ 0 & 0 & 1\end{array}\right]}_{\text{第一次吉文斯旋转}}=\underbrace{\left[\begin{array}{c|cc}u_{11} & u_{12} & u_{13}\\ \hline u_{21} & u_{22} & u_{23}\\ u_{31} & u_{32} & u_{33}\end{array}\right]}_{\text{第一步的后矩阵}} \tag{14.57}$$

前矩阵中的两个零元素来源于式(14.54)，且

$$u_{12}=-x_{11}s_1+x_{12}c_1$$

由于需要将 u_{12} 变换为 0，因此需满足以下条件：

$$s_1=\frac{x_{12}}{x_{11}}c_1$$

利用 $c_1^2+s_1^2=1$ 解出 c_1 和 s_1，我们定义式(14.57)中的第一个正交旋转：

$$c_1=\frac{x_{11}}{\sqrt{x_{11}^2+x_{12}^2}}\quad s_1=\frac{x_{12}}{\sqrt{x_{11}^2+x_{12}^2}} \tag{14.58}$$

步骤二：在第二步中，计算

$$\underbrace{\begin{bmatrix}u_{11} & 0 & u_{13}\\ u_{21} & u_{22} & u_{23}\\ u_{31} & u_{32} & u_{33}\end{bmatrix}}_{\text{第二步的前矩阵}}\quad\underbrace{\begin{bmatrix}c_2 & 0 & -s_2\\ 0 & 1 & 0\\ s_2 & 0 & c_2\end{bmatrix}}_{\text{第二次吉文斯旋转}}=\underbrace{\begin{bmatrix}v_{11} & 0 & v_{13}\\ v_{21} & v_{22} & v_{23}\\ v_{31} & v_{32} & v_{33}\end{bmatrix}}_{\text{第二步的后矩阵}} \tag{14.59}$$

其中

$$v_{13}=-u_{11}s_2+u_{13}c_2$$

由于希望将 v_{13} 变换为 0，因此需满足以下条件：

$$s_2=\frac{u_{13}}{u_{11}}c_2$$

利用 $s_2^2+c_2^2=1$ 解出 c_2 和 s_2，我们定义式(14.57)中的第二个正交旋转：

$$c_2=\frac{u_{11}}{\sqrt{u_{11}^2+u_{13}^2}}\quad s_2=\frac{u_{13}}{\sqrt{u_{11}^2+u_{13}^2}} \tag{14.60}$$

步骤三：对第三步也是最后一步，计算

$$\underbrace{\begin{bmatrix}v_{11} & 0 & 0\\ v_{21} & v_{22} & v_{23}\\ v_{31} & v_{32} & v_{33}\end{bmatrix}}_{\text{第三步的前矩阵}}\quad\underbrace{\begin{bmatrix}1 & 0 & 0\\ 0 & c_3 & -s_3\\ 0 & s_3 & c_3\end{bmatrix}}_{\text{第三次吉文斯旋转}}=\underbrace{\begin{bmatrix}y_{11} & 0 & 0\\ y_{21} & y_{22} & y_{23}\\ y_{31} & y_{32} & y_{33}\end{bmatrix}}_{\text{第三步的后矩阵}} \tag{14.61}$$

其中

$$y_{23} = -v_{22}s_3 + v_{23}c_3$$

由于希望将 y_{23} 变换为 0，因此需满足以下条件：

$$s_3 = \frac{v_{23}}{v_{22}}c_3$$

利用 $s_3^2 + c_3^2 = 1$ 解出 c_3 和 s_3，我们定义式(14.57)中的第三个正交旋转：

$$c_3 = \frac{v_{22}}{\sqrt{v_{22}^2 + v_{23}^2}} \quad s_3 = \frac{v_{23}}{\sqrt{v_{22}^2 + v_{23}^2}} \tag{14.62}$$

由上述三步转换的最后乘积是一个下三角后矩阵：

$$\mathbf{Y} = \begin{bmatrix} y_{11} & 0 & 0 \\ y_{21} & y_{22} & 0 \\ y_{31} & y_{32} & y_{33} \end{bmatrix}$$

这是我们需要的结果。 ■

14.5　扩展的卡尔曼滤波器

14.3 节中所讨论的卡尔曼滤波器问题，提出了由式(14.4)、式(13.5)的线性状态空间模型描述的对动态系统进行状态估计的问题。但是，如果动态系统是如式(14.7)和式(14.8)定义的那样非线性服从高斯分布的，我们可以通过线性化系统的非线性空间状态模型的方法，扩展卡尔曼滤波器的应用范围。这一扩展的状态估计器即为扩展的卡尔曼滤波器。这一扩展是可行的，因为卡尔曼滤波器是在离散时间系统的情况下，用差分方程的形式来定义的。

为了确定扩展的卡尔曼滤波器的实现策略，我们首先需要对定义卡尔曼滤波器的式子进行一些细微的变化，以使得其更利于现在的讨论。

卡尔曼滤波器定义式的变形

首先我们应用式(14.9)和式(14.18)重写新息过程的定义式：

$$\boldsymbol{\alpha}_n = \mathbf{y}_n - \mathbf{b}_n(\hat{\mathbf{x}}_{n|n-1}) \tag{14.63}$$

紧接着，做如下的观察：我们假设实现卡尔曼滤波器除了使用式(14.4)和式(14.5)的状态等式，还有如下状态空间模型的替换形式：

$$\mathbf{x}_{n+1} = \mathbf{A}_{n+1,n}\mathbf{x}_n + \boldsymbol{\omega}_n + \boldsymbol{\xi}_n \tag{14.64}$$

和

$$\mathbf{y}_n = \mathbf{B}_n\mathbf{x}_n + \boldsymbol{\nu}_n \tag{14.65}$$

式(14.65)中给出的测量模型和式(14.5)所给出的模型是完全相同的。然而，式(14.64)和式(14.4)所定义的状态空间模型主要不同点在于引入了新的参数 $\boldsymbol{\xi}_n$，其被假设为已知的（如非随机的）向量。由此，在不考虑式(14.32)对其作出的修改，卡尔曼滤波器可被表述为如下形式：

$$\hat{\mathbf{x}}_{n+1|n} = \mathbf{A}_{n+1,n}\hat{\mathbf{x}}_{n|n} + \boldsymbol{\xi}_n \tag{14.66}$$

这一修改是为了接下来将讨论的扩展的卡尔曼滤波器的实现。

实现扩展的卡尔曼滤波器的预备步骤

如前文所述，扩展的卡尔曼滤波器（EKF）是一个近似解，使得我们可以将卡尔曼滤波的思想扩展到非线性状态空间模型（Jazwinski，1970；May beck，1982）。这里考虑的非线性状态空间模型是式(14.7)和式(14.8)所描述的形式，再次列出只是为了表述的方便：

$$\mathbf{x}_{n+1} = \mathbf{a}_n(\mathbf{x}_n) + \boldsymbol{\omega}_n \tag{14.67}$$

和

$$\mathbf{y}_n = \mathbf{b}_n(\mathbf{x}_n) + \boldsymbol{\nu}_n \tag{14.68}$$

如前文所述，动态噪声 $\boldsymbol{\omega}_n$ 和测量噪声 $\boldsymbol{v}_n$ 是无关的均值为零的高斯噪声过程。其协方差矩阵分别为 $\mathbf{Q}_{\omega,n}$ 和 $\mathbf{Q}_{v,n}$。此外，非线性模型可能随时间而改变，因此用向量函数 $\mathbf{a}_n(\cdot)$ 和 $\mathbf{b}_n(\cdot)$ 的下标 n 表述这种变化。

扩展的卡尔曼滤波器（EMF）的基本思想是在每个时间点，围绕最近状态估计结果对式(14.67)和式(14.68)中定义的状态空间模型线性化。此估计可能是滤波估计也可能是预测估计，其取决于线性化过程中究竟谁起作用。一旦得到了线性化模型，我们就可以使用卡尔曼滤波器的相关公式了。

这一近似过程分为如下两阶段：

阶段 1　新矩阵的构建

通过求偏微分，构建下述两个矩阵：

$$\mathbf{A}_{n+1,n} = \left.\frac{\partial \mathbf{a}_n(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{n|n}} \tag{14.69}$$

和

$$\mathbf{B}_n = \left.\frac{\partial \mathbf{b}_n(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{n|n-1}} \tag{14.70}$$

具体来说，转移矩阵 $\mathbf{A}_{n+1,n}$ 的第 ij 个元素等于向量函数 $\mathbf{a}_n(\mathbf{x})$ 的第 i 个分量对向量 $\mathbf{x}$ 的第 j 个分量的偏微分。同样的，测量矩阵 $\mathbf{B}_n$ 的第 ij 个元素等于向量函数 $\mathbf{b}_n(\mathbf{x})$ 的第 i 个分量对向量 $\mathbf{x}$ 的第 j 个分量的偏微分。前者在滤波状态为 $\hat{\mathbf{x}}_{n|n}$ 时估计，后者在预测估计 $\hat{\mathbf{x}}_{n|n-1}$ 时估计。当 $\hat{\mathbf{x}}_{n|n}$ 和 $\hat{\mathbf{x}}_{n|n-1}$ 已知时，$\mathbf{A}_{n+1,n}$ 和 $\mathbf{B}_n$ 均可计算。

例 2　二维非线性模型

设一由下述二维非线性状态空间模型描述的动态系统：

$$\begin{bmatrix} x_{1,n+1} \\ x_{2,n+1} \end{bmatrix} = \begin{bmatrix} x_{1,n} + x_{2,n}^2 \\ nx_{1,n} - x_{1,n}x_{2,n} \end{bmatrix} + \begin{bmatrix} \omega_{1,n} \\ \omega_{2,n} \end{bmatrix}$$

$$y_n = x_{1,n}x_{2,n}^2 + v_n$$

此例中，有

$$\mathbf{a}_n(\mathbf{x}_n) = \begin{bmatrix} x_{1,n} + x_{2,n}^2 \\ nx_{1,n} - x_{1,n}x_{2,n} \end{bmatrix}$$

和

$$\mathbf{b}_n(\mathbf{x}_n) = x_{1,n}x_{2,n}^2$$

应用式(14.69)和式(14.70)可得：

$$\mathbf{A}_{n+1,n} = \begin{bmatrix} 1 & 2\hat{x}_{2,n|n} \\ n - \hat{x}_{2,n|n} & -\hat{x}_{1,n|n} \end{bmatrix}$$

和

$$\mathbf{B}_n = \begin{bmatrix} \hat{x}_{2,n|n-1}^2 & 2\hat{x}_{1,n|n-1}\hat{x}_{2,n|n-1} \end{bmatrix}$$

阶段 2　空间模型线性化

一旦构建了转移矩阵 $\mathbf{A}_{n+1,n}$ 和测量矩阵 $\mathbf{B}_n$，它们可被用于对非线性函数 $\mathbf{a}_n(\mathbf{x}_n)$ 及 $\mathbf{b}_n(\mathbf{x}_n)$ 围绕状态估计 $\hat{\mathbf{x}}_{n+1,n}$ 和 $\hat{\mathbf{x}}_{n|n}$ 分别进行的一阶泰勒近似中。具体来说：

$$\mathbf{a}_n(\mathbf{x}_n) \approx \mathbf{a}_n(\hat{\mathbf{x}}_{n|n}) + \mathbf{A}_{n+1,n}[\mathbf{x}_n - \hat{\mathbf{x}}_{n|n}] \tag{14.71}$$

和

$$\mathbf{b}_n(\mathbf{x}_n) \approx \mathbf{b}_n(\hat{\mathbf{x}}_{n|n-1}) + \mathbf{B}_n[\mathbf{x}_n - \hat{\mathbf{x}}_{n|n-1}] \tag{14.72}$$

有了上述近似表示，我们可以近似表示式(14.64)和式(14.65)的非线性状态等式。近似结果分别为：

$$\mathbf{x}_{n+1} \approx \mathbf{A}_{n+1,n}\mathbf{x}_n + \boldsymbol{\omega}_n + \boldsymbol{\xi}_n \tag{14.73}$$

和

$$\bar{\mathbf{y}}_n \approx \mathbf{B}_n \mathbf{x}_n + \boldsymbol{v}_n \tag{14.74}$$

这里，我们引入了两个新的量：系统模型中的 $\boldsymbol{\xi}_n$ 和测量模型中的 $\bar{\mathbf{y}}_n$。两者的定义如下：

$$\boldsymbol{\xi}_n = \mathbf{a}_n(\hat{\mathbf{x}}_{n|n}) - \mathbf{A}_{n+1,n}\,\hat{\mathbf{x}}_{n|n} \tag{14.75}$$

和

$$\bar{\mathbf{y}}_n = \mathbf{y}_n - [\mathbf{b}_n(\hat{\mathbf{x}}_{n|n-1}) - \mathbf{B}_n\,\hat{\mathbf{x}}_{n|n-1}] \tag{14.76}$$

其中，$\mathbf{a}_n(\hat{\mathbf{x}}_{n|n})$和 $\mathbf{b}_n(\hat{\mathbf{x}}_{n|n-1})$分别是给定的非线性函数 $\mathbf{a}_n(\mathbf{x}_n)$和 $\mathbf{b}_n(\mathbf{x}_n)$在 $\mathbf{x}_n=\hat{\mathbf{x}}_{n|n}$和 $\mathbf{x}_n=\hat{\mathbf{x}}_{n|n-1}$时的估计值。如果联系式(14.69)中给出的 $\mathbf{A}_{n+1,n}$是已知的，那么新加入的项 $\boldsymbol{\xi}_n$ 对任意时刻 n 均是已知的。其论证了我们先前观察的有效性。同理，根据式(14.70)$\mathbf{B}_n$ 是已知的，所以第二个新引入的项 $\bar{\mathbf{y}}_n$ 对任意时刻 n 均是已知的。因此，我们可以将 $\bar{\mathbf{y}}_n$ 视为线性化模型在 n 时刻有效的观测向量。

扩展的卡尔曼滤波器的实现

式(14.73)和式(14.74)所描述的近似状态空间模型，是与式(14.64)和式(14.65)所描述的有相似的数学表达形式的线性模型。两者的唯一细微差别在于，为了模型的线性化，式(14.65)中的观测值 $\mathbf{y}(n)$由新的观测值 $\bar{\mathbf{y}}_n$ 代替。基于这一目的，我们已经预先将式(14.64)和式(14.65)的状态空间模型公式化了。

因此，扩展的卡尔曼滤波器（EMF)的定义公式和表 14.2 中卡尔曼滤波器的定义公式类似，只是用表 14.4 描述的方式，对卡尔曼滤波器的表 14.2 中的第二个和第四个公式进行了相应的修改。

表 14.4　扩展的卡尔曼滤波器的总结

输入过程：
Observations$=[\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n]$
已知参数：
非线性状态向量函数$=\mathbf{a}_n(\mathbf{x}_n)$
非线性测量向量函数$=\mathbf{b}_n(\mathbf{x}_n)$
过程噪声向量的协方差矩阵$=\mathbf{Q}_{\omega,n}$
测量噪声向量的协方差矩阵$=\mathbf{Q}_{v,n}$
计算：$n=1, 2, 3, \cdots$
$\mathbf{G}_n=\mathbf{P}_{n,n-1}\mathbf{B}_n^T[\mathbf{B}_n\mathbf{P}_{n,n-1}\mathbf{B}_n^T+\mathbf{Q}_{v,n}]^{-1}$
$\boldsymbol{\alpha}_n=\mathbf{y}_n-\mathbf{b}_n(\hat{\mathbf{x}}_{n\mid n-1})$
$\hat{\mathbf{x}}_{n\mid n}=\hat{\mathbf{x}}_{n\mid n-1}+\mathbf{G}_n\boldsymbol{\alpha}_n$
$\hat{\mathbf{x}}_{n+1\mid n}=\mathbf{a}_n(\hat{\mathbf{x}}_{n\mid n})$
$\mathbf{P}_{n\mid n}=\mathbf{P}_{n\mid n-1}-\mathbf{G}_n\mathbf{B}_n\mathbf{P}_{n\mid n-1}$
$\mathbf{P}_{n+1\mid n}=\mathbf{A}_{n+1,n}\mathbf{P}_{n\mid n}\mathbf{A}_{n+1,n}^T+\mathbf{Q}_{\omega,n}$

说明：

1. 线性化的矩阵 $\mathbf{A}_{n+1,n}$和 $\mathbf{B}_n$ 是从它们相应的非线性函数 $\mathbf{a}_n(\mathbf{x}_n)$、$\mathbf{b}_n(\mathbf{x}_n)$，分别用式(14.69)和式(14.70)计算得到的。
2. $\mathbf{a}_n(\hat{\mathbf{x}}_{n\mid n})$ 和 $\mathbf{b}_n(\hat{\mathbf{x}}_{n\mid n-1})$的值是通过将非线性向量函数 $\mathbf{a}_n(\mathbf{x}_n)$、$\mathbf{b}_n(\mathbf{x}_n)$中的状态 $\mathbf{x}_n$ 分别替换为滤波状态估计 $\hat{\mathbf{x}}_{n\mid n}$和预测状态估计$\hat{\mathbf{x}}_{n\mid n-1}$得到的。
3. 检查表 14.4 的迭代顺序，现在知道用式(14.69)、式(14.70)描述的方式为 $\mathbf{A}_{n+1,n}$和 $\mathbf{B}_n$ 赋值的原因。

初始条件：

$\hat{\mathbf{x}}_{1\mid 0}=\mathbb{E}[\mathbf{x}_1]$

$\mathbf{P}_{1,0}=\mathbb{E}[(\mathbf{x}_1-\mathbb{E}[\mathbf{x}_1])(\mathbf{x}_1-\mathbb{E}[\mathbf{x}_1])^T]=\Pi_0$

其中，$\Pi_0=\delta^{-1}\mathbf{I}$，$\delta$ 是一个小的正常数，$\mathbf{I}$ 是单位矩阵。

对扩展的卡尔曼滤波器的评价

扩展的卡尔曼滤波器在非线性状态估计领域得到关注的原因主要有两个：

1. 扩展的卡尔曼滤波器建立在卡尔曼滤波器理论的框架之上，有较强的理论依据。
2. 扩展的卡尔曼滤波器相对易于理解，因此被直接用于实践，并已有相当长的应用历史。

然而，扩展的卡尔曼滤波器有两个主要缺点，限制了它的应用范围：

1. 为使扩展的卡尔曼滤波器能令人满意地运行，状态空间模型的非线性必须是轻度的，以满足应用一阶泰勒展开式的条件。这是扩展的卡尔曼滤波器的理论基础。

2. 扩展的卡尔曼滤波器的实现，需要非线性动态系统的状态空间的一阶偏微分（如函数行列式）的相关知识，这一内容尚处于研究阶段。然而，在许多实际应用中，函数行列式的计算结果难以令人满意或根本无法计算。

为了指出扩展的卡尔曼滤波器的局限性，描述状态估计的贝叶斯方法是有意义的，我们将在下一节做详细的讨论。

14.6　贝叶斯滤波器

采用贝叶斯滤波器解决动态系统的状态估计问题，从线性到非线性，是由于至少它在概念上为动态系统状态估计提供了统一的框架，因此把它作为这一节的标题。

自然地，概率原理是解决状态估计问题的贝叶斯方法的核心。为了易于表示，下面我们用“分布”一词表示概率密度函数。此外，参照式(14.1)的系统（状态）模型和式(14.2)的测量模型，使用以下标记：

$\mathbf{Y}_n$	=观测值序列，表示 $\{\mathbf{y}_i\}_{i=1}^n$。
$p(\mathbf{x}_n \mid \mathbf{Y}_{n-1})$	=在当前时刻 n，给定整个观测序列直到并包括 $\mathbf{y}_{n-1}$ 时，状态 $\mathbf{x}_n$ 的先验分布。
$p(\mathbf{x}_n \mid \mathbf{Y}_n)$	=给定整个观测序列直到并包括当前时刻 n 时，当前状态 $\mathbf{x}_n$ 的后验分布；这一分布一般简单地称为“后验”。
$p(\mathbf{x}_n \mid \mathbf{x}_{n-1})$	=给定最近的过去态 $\mathbf{x}_{n-1}$，当前状态 $\mathbf{x}_n$ 的过渡态分布；这一分布一般称为“过渡先验”或者“先验”。
$l(\mathbf{y}_n \mid \mathbf{x}_n)$	=给定当前状态 $\mathbf{x}_n$，当前观测值 $\mathbf{y}_n$ 的似然函数。

贝叶斯滤波器的实现，唯一的假设是状态的变化是服从马尔可夫过程的；这一假设也隐含在卡尔曼滤波器的公式和公式的变体中，这些在本章之前的部分讨论过。基本上，该假设包含了以下两个条件的结合：

1. 给定状态序列 $\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_{n-1}, \mathbf{x}_n$，当前状态 $\mathbf{x}_n$ 仅取决于最近的过去态 $\mathbf{x}_{n-1}$，通过状态过渡分布 $p(\mathbf{x}_n \mid \mathbf{x}_{n-1})$。初始态 $\mathbf{x}_0$ 是分布式的，根据

$$p(\mathbf{x}_0 \mid \mathbf{y}_0) = p(\mathbf{x}_0)$$

2. 观测值 $\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n$ 仅条件依赖于相应的状态 $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n$；这一假设意味着观测值的条件联合似然函数（例如，所有观测值的联合分布与直到且包括 n 时刻的状态有关）如下

$$l(\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n \mid \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n) = \prod_{i=1}^{n} l(\mathbf{y}_i \mid \mathbf{x}_i) \tag{14.77}$$

后验分布 $p(\mathbf{x}_n \mid \mathbf{Y}_n)$在贝叶斯分析中至关重要，它包含了在 n 时刻，已经接收整个观测序列 $\mathbf{Y}_n$ 的条件下，关于状态 $\mathbf{x}_n$ 的全部知识。因此，$p(\mathbf{x}_n \mid \mathbf{Y}_n)$包含了所有状态估计的必要信息。假设，例如希望决定状态 $\mathbf{x}_n$ 满足最小均方误差（MMSE）时最优的滤波估计，根据贝叶斯估计量，需要的解是

$$\hat{\mathbf{x}}_{n|n} = \mathbb{E}_p[\mathbf{x}_n \mid \mathbf{Y}_n] = \int \mathbf{x}_n p(\mathbf{x}_n \mid \mathbf{Y}_n) \mathrm{d}\mathbf{x}_n \tag{14.78}$$

相应地，为了滤波估计$\hat{\mathbf{x}}_{n|n-1}$精度的评估，计算协方差矩阵

$$\mathbf{P}_{n|n} = \mathbb{E}_p[(\mathbf{x}_n - \hat{\mathbf{x}}_{n|n})(\mathbf{x}_n - \hat{\mathbf{x}}_{n|n})^T] = \int (\mathbf{x}_n - \hat{\mathbf{x}}_{n|n})(\mathbf{x}_n - \hat{\mathbf{x}}_{n|n})^T p(\mathbf{x}_n \mid \mathbf{Y}_n) \mathrm{d}\mathbf{x}_n \tag{14.79}$$

计算效率已经成为令人关注的实际要素，因此，用递归的方式计算滤波估计$\hat{\mathbf{x}}_{n|n-1}$和相关

的参数是非常必要的。假设我们有 $n-1$ 时刻状态 $\mathbf{x}_{n-1}$ 的后验分布 $p(\mathbf{x}_{n-1}|\mathbf{Y}_{n-1})$。那么 n 时刻的状态的后验分布的更新值，通过以下两个基本的时间步骤产生：

1. 时间更新，包括给定观测序列 $\mathbf{Y}_{n-1}$，计算 $\mathbf{x}_n$ 的预测分布，如下所示：

$$\underbrace{p(\mathbf{x}_n|\mathbf{Y}_{n-1})}_{\text{预测分布}} = \int \underbrace{p(\mathbf{x}_n|\mathbf{x}_{n-1})}_{\text{先验分布}} \underbrace{p(\mathbf{x}_{n-1}|\mathbf{Y}_{n-1})}_{\text{旧的后验分布}} d\mathbf{x}_{n-1} \tag{14.80}$$

这一公式用概率原理的基本定律证明如下：旧的后验分布 $p(\mathbf{x}_{n-1}|\mathbf{Y}_{n-1})$ 和先验分布 $p(\mathbf{x}_n|\mathbf{x}_{n-1})$ 的乘积，得到了旧状态 $\mathbf{x}_{n-1}$ 和当前状态 $\mathbf{x}_n$ 在 $\mathbf{Y}_{n-1}$ 的条件下的联合分布。这一联合分布对 $\mathbf{x}_{n-1}$ 积分，得到了预测分布 $p(\mathbf{x}_n|\mathbf{Y}_{n-1})$。

2. 测量更新，利用当前状态 $\mathbf{x}_n$ 的包含在新观测值 $\mathbf{y}_n$ 中的信息，计算更新的后验分布 $p(\mathbf{x}_n|\mathbf{Y}_n)$。特别地，对预测分布 $p(\mathbf{x}_n|\mathbf{Y}_{n-1})$ 运用著名的贝叶斯定理得到

$$\underbrace{p(\mathbf{x}_n|\mathbf{Y}_n)}_{\text{更新的后验分布}} = \frac{1}{Z_n} \underbrace{p(\mathbf{x}_n|\mathbf{Y}_{n-1})}_{\text{预测分布}} \underbrace{l(\mathbf{y}_n|\mathbf{x}_n)}_{\text{似然函数}} \tag{14.81}$$

其中

$$Z_n = p(\mathbf{y}_n|\mathbf{Y}_{n-1}) = \int l(\mathbf{y}_n|\mathbf{x}_n) p(\mathbf{x}_n|\mathbf{Y}_{n-1}) d\mathbf{x}_n \tag{14.82}$$

是标准化常数（也称作分析函数）；它保证了后验分布 $p(\mathbf{x}_n|\mathbf{Y}_n)$ 的多维曲线下的全部体积是正如所要求的那样，是单位的。标准化常数序列 $\{Z_i\}_{i=1}^n$，产生了相应观测序列 $\{\mathbf{Y}_i\}_{i=1}^N$ 的联合对数似然函数，如下所示

$$\log(p(\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n)) = \sum_{i=1}^{n} \log(Z_i) \tag{14.83}$$

式(14.80)和式(14.83)都是前面描述的马尔可夫假设的推论。

在每个时间步骤，都通过贝叶斯模型的计算来执行时间更新和测量更新。事实上，它们构成了一个计算的递归或者循环，如图 14.4 描述的那样；为了表示的方便，省略了 Z_n。

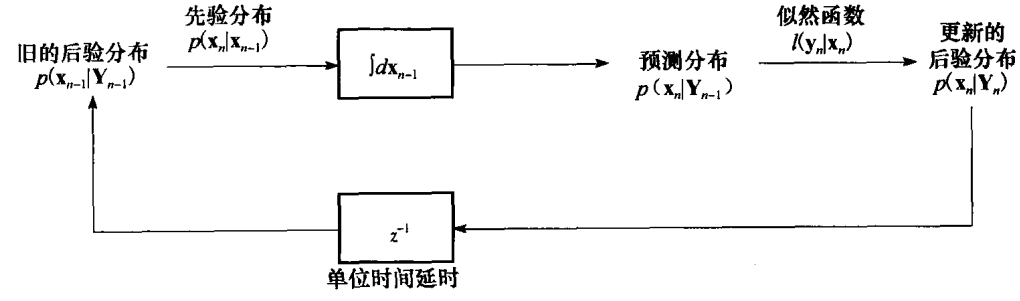


图 14.4　贝叶斯滤波器的框图，将更新的后验分布 $p(\mathbf{x}_n|\mathbf{Y}_n)$ 作为所关心的输出

近似的贝叶斯滤波

图 14.4 的贝叶斯滤波器是概念上最优的，有以下两个有趣的性质：

1. 模型以递归的方式运行，传播后验概率 $p(\mathbf{x}_n|\mathbf{Y}_n)$。

2. 提取自全部观测过程 $\mathbf{Y}_n$ 的关于状态 $\mathbf{x}_n$ 的模型知识，完全包含在后验分布 $p(\mathbf{x}_n|\mathbf{Y}_n)$ 中。

随着这一分布称为关注的焦点，现在列出滤波目标的基础。详细地说，考虑状态 $\mathbf{x}_n$ 的任意的函数，记为 $h(\mathbf{x}_n)$。在实际的滤波应用中，我们感兴趣的是在线估计函数 $h(\mathbf{x}_n)$ 的信号特征。这些特征包含在贝叶斯估计量中，用函数 $h(\mathbf{x}_n)$ 的总体平均值定义，称为

$$\overline{h}_n = \mathbb{E}_p[h(\mathbf{x}_n)] = \int \underbrace{h(\mathbf{x}_n)}_{\text{任意函数}} \underbrace{p(\mathbf{x}_n \mid \mathbf{Y}_n)}_{\text{后验分布}} \mathrm{d}\mathbf{x}_n \tag{14.84}$$

其中$\mathbb{E}_p$是对后验分布 $p(\mathbf{x}_n \mid \mathbf{Y}_n)$的期望，该后验分布是关于线性或者非线性动态系统的。式(14.84)包括两个特例：关于状态的滤波估计的式(14.78)和关于估计的协方差矩阵的式(14.79)，说明了贝叶斯模型的一般的统一框架。对式(14.78)，有 $h(\mathbf{x}_n) = \mathbf{x}_n$，对式(14.79)，有

$$h(\mathbf{x}_n) = (\mathbf{x}_n - \hat{\mathbf{x}}_{n|n})(\mathbf{x}_n - \hat{\mathbf{x}}_{n|n})^{\mathrm{T}}$$

其中的 h 是一个向量函数的形式。

对于由式(14.4)和式(14.5)的线性高斯模型描述的动态系统的特例，式(14.84)的递归解恰好是通过卡尔曼滤波器实现的，见习题 14.10。然而，当动态系统是非线性或非高斯的，或者既非线性也非高斯，那么构成式(14.84)的被积函数的生成分布不再是服从高斯分布的，这造成了最优贝叶斯估计量$\overline{h}_n$的计算困难。对于后一种情况，我们别无选择只能放弃贝叶斯最优，寻找一个易计算的近似估计量。

为了这一实际的实现，现在正式确定非线性滤波目标：

在 n 时刻，给定关于式(14.7)和式(14.8)的非线性状态空间模型的全部观测序列 $\mathbf{Y}_n$，推导出式(14.84)定义的贝叶斯估计量$\overline{h}_n$的近似实现，满足两个实际要求：

1. *计算的可信性。*
2. *递归的可实现性。*

通过近似的贝叶斯滤波器获得的非线性滤波问题的局部最优解，可能通过两个途径中的一个得到，取决于求近似的方法：

1. *后验分布的直接数值近似。*这一非线性滤波的直接方法的基本原理总结如下：

一般地，用局部的观点看，相对于求表示滤波器系统（状态）模型特征的非线性函数的近似，直接求后验分布 $p(\mathbf{x}_n \mid \mathbf{Y}_n)$的近似更容易。

详细地说，给定直到并包含 n 时刻的全部观测值，在点 $\mathbf{x}_n = \hat{\mathbf{x}}_{n|n}$附近求后验分布$p(\mathbf{x}_n \mid \mathbf{Y}_n)$的局部近似，其中$\hat{\mathbf{x}}_{n|n}$是状态 $\mathbf{x}_n$ 的滤波估计；对局部的强调使得滤波器的设计在计算上简单且执行速度快。近似的目的是促进卡尔曼滤波器理论的随后应用。事实上，通过直接使用数值方法，广泛使用的扩展的卡尔曼滤波器成为近似贝叶斯滤波的例子。最重要的是，在14.7 节介绍了一个新的贝叶斯滤波器，称为数值积分卡尔曼滤波器，它比扩展的卡尔曼滤波器更强大。

2. *后验分布的间接数值近似。*非线性滤波的第二种方法的基本原理总结如下：

从全局的观点看，通过使用 Monte Carlo 模拟，求后验分布 $p(\mathbf{x}_n \mid \mathbf{Y}_n)$的间接近似，使得非线性滤波的贝叶斯框架在计算上易处理。

将在 14.8 节讨论的粒子滤波器，是非线性滤波的第二种方法的一个普及的例子。更确切地说，粒子滤波器依赖于一个称为逐次 Monte Carlo 方法的技术，该方法使用一系列随机抽取带关联权值的样本，来近似后验分布 $p(\mathbf{x}_n \mid \mathbf{Y}_n)$。随着模拟过程中使用的样本数的增大，后验分布的 Monte Carlo 计算越来越精确，这是我们想要的目标。然而，样本数量的增大使得使用 SMC 方法的计算更加复杂。换句话说，以计算上的代价换取了滤波精度。

通过简短的讨论，显然局部的直接方法来近似贝叶斯滤波是建立在卡尔曼滤波器理论的基础上的，而全局的间接方法脱离了这一理论，另辟蹊径。一般来说，非线性滤波的全局的间接方法比局部的直接方法在计算上要求更高。

14.7　数值积分卡尔曼滤波器：基于卡尔曼滤波器

到目前为止，我们已经知道，当假设所有的条件分布都是高斯分布时，贝叶斯滤波器是易于计算的。在这个特例中，贝叶斯滤波器的近似值归结为计算一个特殊形式的多维积分，表示为

非线性函数 × 高斯函数

具体地说，给定一个关于向量 $\mathbf{x}\in\mathbb{R}^M$ 的任意非线性函数 $\mathbf{f}(\mathbf{x})$，使用高斯函数，考虑如下形式的积分：

$$h(\mathbf{f})=\int_{\mathbb{R}^M}\underbrace{\mathbf{f}(\mathbf{x})}_{\substack{\text{任意}\\\text{函数}}}\underbrace{\exp(-\mathbf{x}^T\mathbf{x})}_{\substack{\text{高斯}\\\text{函数}}}\mathrm{d}\mathbf{x} \tag{14.85}$$

这是定义在笛卡尔坐标系下的。对非线性函数 $h(\mathbf{f})$ 的数值近似，我们打算使用*三阶球面径向数值积分法则*（Stroud，1971；Cools，1997）。数值积分法则是通过迫使数值积分点服从某种对称的形式建立起来的。这样，为了求出一些权值和数值积分点，而求解一系列非线性方程的复杂度显著降低。在详细介绍数值积分法则之前，先引入一些记法和定义：

- 用$\mathcal{D}$来表示积分区域，如果满足以下两个条件，我们就说定义在$\mathcal{D}$上的加权函数 $w(\mathbf{x})$ 是完全对称的：
  1）$\mathbf{x}\in\mathcal{D}$说明 $\mathbf{y}\in\mathcal{D}$，其中 $\mathbf{y}$ 是从 $\mathbf{x}$ 获得的任意一点，通过交换和改变 $\mathbf{x}$ 坐标的记号得到。
  2）在$\mathcal{D}$上 $w(\mathbf{x})=w(\mathbf{y})$。
- 在完全对称的区域中，我们称点 $\mathbf{u}$ 是一个发生器，如果 $\mathbf{u}=(u_1,u_2,\cdots,u_r,0,\cdots,0)\in\mathbb{R}^M$，其中 $u_i\geqslant u_{i+1}>0$，对 $i=1,2,\cdots,(r-1)$。
- 我们用记号 $[u_1,u_2,\cdots,u_r]$ 来表示整个点集，可以通过交换和改变发生器 $\mathbf{u}$ 的记号的一切方式得到。为了简洁，我们在记数中抑制$(n-r)$个零结点。比如$[1]=\mathbb{R}^2$ 表示以下点集：

$$\left\{\begin{pmatrix}1\\0\end{pmatrix},\begin{pmatrix}0\\1\end{pmatrix},\begin{pmatrix}-1\\0\end{pmatrix},\begin{pmatrix}0\\-1\end{pmatrix}\right\}$$

- 我们用记号$[u_1,u_2,\cdots,u_r]_i$ 来表示发生器 $\mathbf{u}$ 的第 i 个点。

转换为球面径向积分

这个转换过程中的关键步骤是变量转换，即将 Cartesian 向量 $\mathbf{x}\in\mathbb{R}^M$ 转换为由半径 r 和方向向量 $\mathbf{z}$ 定义的球面径向向量，概括为：

令 $\mathbf{x}=r\mathbf{z}$，$\mathbf{z}^T\mathbf{z}=1$，使得对 $r\in[0,\infty)$，有 $\mathbf{x}^T\mathbf{x}=r^2$

然后式(14.85)的积分可以改写为球面径向坐标系下的二重积分，如下所示：

$$h(\mathbf{f})=\int_0^\infty\int_{\mathcal{U}M}\mathbf{f}(r\mathbf{z})r^{M-1}\exp(-r^2)\mathrm{d}\sigma(\mathbf{z})\mathrm{d}r \tag{14.86}$$

$\mathcal{U}_M$ 是由$\mathcal{U}_M=\{\mathbf{z};\mathbf{z}^T\mathbf{z}=1\}$定义的区域，在对$\mathcal{U}_M$ 的积分中$\sigma(\cdot)$是球表面积。

$$S(r)=\int_{\mathcal{U}_M}\mathbf{f}(r\mathbf{z})\mathrm{d}\sigma(\mathbf{z}) \tag{14.87}$$

式(14.87)的积分是根据球面法则计算的。已经计算出 $S(r)$，我们发现对半径的积分

$$h=\int_0^\infty S(r)r^{M-1}\exp(-r^2)\mathrm{d}r \tag{14.88}$$

可以通过运用高斯求积计算得出。计算出 h，式(14.85)的计算就完成了。这两种法则将在下

文中依次介绍。

球面法则

首先来看一个具有如下形式的三阶球面法则

$$\int_{\mathcal{U}_M} \mathbf{f}(\mathbf{z})\mathrm{d}\sigma(\mathbf{z}) \approx w\sum_{i=1}^{2M}\mathbf{f}[u]_i \tag{14.89}$$

式(14.89)的法则需要发生器$[u]$中总共 $2M$ 个数值积分点；这些数值积分点位于一个 M 维的球体和它的轴线的交集里。为了确定未知的参数 u 和 w，由于是完全对称发生器，考虑单项式$\mathbf{f}(\mathbf{z})=1$ 和 $\mathbf{f}(\mathbf{z})=z_1^2$ 就足够了，给出

$$\mathbf{f}(\mathbf{z}) = 1: \quad 2Mw = \int_{\mathcal{U}_M} \mathrm{d}\sigma(\mathbf{z}) = A_M \tag{14.90}$$

$$\mathbf{f}(\mathbf{z}) = z_1^2: \quad 2wu^2 = \int_{\mathcal{U}_M} z_1^2\mathrm{d}\sigma(\mathbf{z}) = \frac{A_M}{M} \tag{14.91}$$

其中 M 是向量 $\mathbf{x}$ 的维数，单位球体的表面积定义为：

$$A_M = \frac{2\sqrt{\pi^M}}{\Gamma(M/2)}$$

其中

$$\Gamma(M) = \int_0^\infty x^{M-1}\exp(-x)\mathrm{d}x$$

是伽玛函数。给定刚定义的 A_M，通过式(14.90)和式(14.91)解出 w 和 u 得到

$$w = \frac{A_M}{2M} \quad 且 \quad u^2 = 1$$

径向法则

对于径向法则来说，我们打算使用高斯求积，它被认为是一维空间中就算积分的最有效的数值方法。一个 m 点高斯求积精确到$(2M-1)$次多项式，如下：

$$\int_{\mathcal{D}} f(x)w(x)\mathrm{d}x \approx \sum_{i=1}^{m} w_i\, f(x_i) \tag{14.92}$$

其中 $w(x)$表示一个加权函数（Press 等，1988）。x_i 和 w_i 分别是待确定的正交点和关联权值。比较式(14.88)和式(14.92)的积分，得出加权函数为 $w(x)=x^{M-1}\exp(-x^2)$，积分区域是 $[0,\infty)$。因此用 $t=x^2$ 做最后的变量替换，得到想要的半径积分

$$\int_0^\infty f(x)x^{M-1}\exp(-x^2)\mathrm{d}x = \frac{1}{2}\int_0^\infty \tilde{f}(t)t^{(M/2)-1}\exp(-t)\mathrm{d}t \tag{14.93}$$

其中$\tilde{f}(t)=f(\sqrt{t})$。式(14.93)等号右边的积分，现在的形式是著名的广义高斯拉盖尔公式(Stroud，1966；Press and Teukolsky，1990)。

一阶的高斯拉盖尔法则对 $\tilde{f}(t)=1$，t 是精确的。相应地，法则对 $f(x)=1$，x^2 是精确的；对奇数次次多项式，它不是精确的，例如对 $f(x)=x$，x^3。幸运的是，当径向法则与球面法则结合之后计算式(14.85)的积分，由此得到的球面径向法则消去了所有的奇数次多项式。得到这个好的结果是由于对称性的优点，使得球面法则消去了任意奇数次多项式，见式(14.86)。因此，计算式(14.85)的球面径向法则对所有一次多项式是精确的。根据这个论证，球面径向法则对所有 $\mathbf{x}\in\mathbb{R}^M$ 中的三次多项式是精确的，考虑一阶广义高斯拉盖尔法则，它使用单一点和单一权值。因此可以写成

$$\int_0^\infty f(x)x^{M-1}\exp(-x^2)\mathrm{d}x \approx w_1 f(x_1)$$

其中

$$w_1 = \frac{1}{2}\Gamma\left(\frac{M}{2}\right) \text{ 和 } x_1 = \sqrt{M/2}$$

球面径向法则

在最后一节里，我们阐述两个有用的结论，它们被用于结合球面和径向法则和对高斯加权积分，扩展球面径向法则。各自的结果表达为以下的两个定理（Arasaratnam and Haykin, 2009）：

定理 1　用 mr 点高斯求积法则数值计算半径积分：

$$\int_0^\infty f(r)r^{M-1}\exp(-r^2)\mathrm{d}r = \sum_{i=1}^{m_r} a_i f(r_i)$$

用 ms 点球面法则数值计算球状积分：

$$\int_{\mathcal{U}_M} \mathbf{f}(r\mathbf{s})\mathrm{d}\sigma(\mathbf{s}) = \sum_{j=1}^{m_s} b_j \mathbf{f}(r\mathbf{s}_j)$$

然后，一个 $(m_s \times m_r)$ 点的球面径向数值积分法则通过双求和近似

$$\int_{\mathbb{R}^M} \mathbf{f}(\mathbf{x})\exp(-\mathbf{x}^T\mathbf{x})\mathrm{d}\mathbf{x} \approx \sum_{j=1}^{m_s}\sum_{i=1}^{m_r} a_i b_j \mathbf{f}(r_i\mathbf{s}_j)$$

定理 2　将两个加权函数表示为 $w_1(\mathbf{x}) = \exp(-\mathbf{x}^T\mathbf{x})$ 和 $w_2(\mathbf{x}) = \mathcal{N}(\mathbf{x};\boldsymbol{\mu},\boldsymbol{\Sigma})$，其中，对给定的向量 $\mathbf{x}$，项 $\mathcal{N}(\mathbf{x};\boldsymbol{\mu},\boldsymbol{\Sigma})$ 表示一个均值为 $\boldsymbol{\mu}$ 方差矩阵为 $\boldsymbol{\Sigma}$ 的高斯分布。然后，对于每个平方根矩阵 $\boldsymbol{\Sigma}^{1/2}$ 使得 $\boldsymbol{\Sigma}^{1/2}\boldsymbol{\Sigma}^{T/2} = \boldsymbol{\Sigma}$，有

$$\int_{\mathbb{R}^M} \mathbf{f}(\mathbf{x})w_2(\mathbf{x})\mathrm{d}\mathbf{x} = \frac{1}{\sqrt{\pi^M}}\int_{\mathbb{R}^M} f(\sqrt{2\boldsymbol{\Sigma}}\mathbf{x}+\boldsymbol{\mu})w_1(\mathbf{x})\mathrm{d}\mathbf{x}$$

对于三阶球面径向法则，$m_r = 1$ 和 $m_s = 2M$。相应地，我们只需要总共为 $2M$ 个数值积分点。而且，这个法则对以下被积函数是精确的，该被积函数能写作不超过三次多项式和所有奇数次多项式的线性组合的形式。调用定理 1 和定理 2，我们现在可以扩展三阶球面径向法则进行标准高斯加权积分的数值计算。

$$h_N(\mathbf{f}) = \int_{\mathbb{R}^M} \mathbf{f}(\mathbf{x})\,\mathcal{N}(\mathbf{x};0,\mathbf{I})\mathrm{d}\mathbf{x} \approx \sum_{i=1}^{m} w_i \mathbf{f}(\boldsymbol{\xi}_i) \tag{14.94}$$

其中

$$\xi_i = \sqrt{\frac{m}{2}}[1]_i \text{ 和 } w_i = \frac{1}{m}, i = 1,2,\cdots,m = 2M$$

实际上，$\boldsymbol{\xi}_i$ 是 M 维向量 $\mathbf{x}$ 的数值积分点的表示。

数值积分卡尔曼滤波器的推导过程

式(14.94)是数值积分法则，我们寻找式(14.85)的积分的数值近似。实际上，数值积分法则是计算非线性滤波的贝叶斯框架中包含的所有积分的核心。对扩展的卡尔曼滤波器，我们假设动态噪声 $\boldsymbol{\omega}_n$ 和测量噪声 $\boldsymbol{\nu}_n$ 是联合服从高斯分布的。这个假设可用以下内容证明：

1. 从数学的角度来看，高斯进程是简单的，数学上是容易解决的。

2. 在很多现实问题中出现的噪声过程，可以建模为高斯过程，根据概率理论的中心极限定理。

在高斯假设条件下，我们现在可以通过以下的数值积分法则来近似贝叶斯滤波器：

1. 时间更新。假设先验分布 $p(\mathbf{x}_{n-1}|\mathbf{Y}_{n-1})$ 是用一个高斯分布来近似，该高斯分布的均值为 $\hat{\mathbf{x}}_{n-1|n-1}$，协方差矩阵等于滤波误差协方差矩阵 $\mathbf{P}_{n-1|n-1}$。然后，对贝叶斯估计量使用公式，我们可以将状态的预测估计表示为：

$$\hat{\mathbf{x}}_{n|n-1} = \mathbb{E}[\mathbf{x}_n \mid \mathbf{Y}_{n-1}] = \int_{\mathbb{R}^M} \underbrace{\mathbf{a}(\mathbf{x}_{n-1})}_{\text{非线性状态转换函数}} \underbrace{\mathcal{N}(\mathbf{x}_{n-1}; \hat{\mathbf{x}}_{n-1|n-1}, \mathbf{P}_{n-1|n-1})}_{\text{高斯分布}} \mathrm{d}\mathbf{x}_{n-1} \tag{14.95}$$

这里我们运用了式(14.7)系统模型的知识，以及动态噪声 $\boldsymbol{\omega}_{n-1}$ 与观测序列 $\mathbf{Y}_{n-1}$ 无关的事实。类似地，我们获得预测－错误协方差矩阵

$$\mathbf{P}_{n|n-1} = \int_{\mathbb{R}^M} \mathbf{a}(\mathbf{x}_{n-1})\mathbf{a}^T(\mathbf{x}_{n-1})\, \mathcal{N}(\mathbf{x}_{n-1}; \hat{\mathbf{x}}_{n-1|n-1}, \mathbf{P}_{n-1,n-1})\mathrm{d}\mathbf{x}_{n-1} - \hat{\mathbf{x}}_{n|n-1}\hat{\mathbf{x}}_{n|n-1}^T + \mathbf{Q}_{w,n} \tag{14.96}$$

2. 测量更新。式(14.95)是时间更新的一个近似公式。下面找寻一个测量更新的公式。以序列 $\mathbf{Y}_{n-1}$ 为条件，状态 $\mathbf{x}_n$ 和测量值 $\mathbf{y}_n$ 的联合分布也是服从高斯分布的，表示为：

$$\mathcal{N} = \underbrace{\left(\begin{bmatrix}\mathbf{x}_n \\ \mathbf{y}_n\end{bmatrix}\right)}_{\text{联合变量}}; \underbrace{\begin{bmatrix}\hat{\mathbf{x}}_{n|n-1} \\ \hat{\mathbf{y}}_{n|n-1}\end{bmatrix}}_{\text{联合均值}}, \underbrace{\begin{bmatrix}\mathbf{P}_{n|n-1} & \mathbf{P}_{xy,n|n-1} \\ \mathbf{P}_{yx,n|n-1} & \mathbf{P}_{yy,n|n-1}\end{bmatrix}}_{\text{联合协方差矩阵}} \tag{14.97}$$

其中，$\hat{\mathbf{x}}_{n|n-1}$ 定义于公式(14.95)中，给定序列 $\mathbf{Y}_{n-1}$，$\hat{\mathbf{y}}_{n|n-1}$ 是观测值 $\mathbf{y}_n$ 的预测估计量，表示为

$$\hat{\mathbf{y}}_{n|n-1} = \int_{\mathbb{R}^M} \underbrace{\mathbf{b}(\mathbf{x}_n)}_{\text{非线性测量函数}} \underbrace{\mathcal{N}(\mathbf{x}_n; \hat{\mathbf{x}}_{n|n-1}, \mathbf{P}_{n|n-1})\mathrm{d}\mathbf{x}_n}_{\text{高斯分布}} \tag{14.98}$$

新息协方差矩阵定义为：

$$\mathbf{P}_{yy,n|n-1} = \int_{\mathbb{R}^M} \underbrace{\mathbf{b}(\mathbf{x}_n)\mathbf{b}^T(\mathbf{x}_n)}_{\text{非线性测量函数与自身的外积}} \underbrace{\mathcal{N}(\mathbf{x}_n; \hat{\mathbf{x}}_{n|n-1}, \mathbf{P}_{n|n-1})}_{\text{高斯分布}}\mathrm{d}\mathbf{x}_n - \underbrace{\hat{\mathbf{y}}_{n|n-1}\hat{\mathbf{y}}_{n|n-1}^T}_{\text{估计值}\hat{\mathbf{y}}_{n|n-1}\text{与自身的外积}} + \underbrace{\mathbf{Q}_{v,n}}_{\text{测量噪声的协方差矩阵}} \tag{14.99}$$

最后，给出状态 $\mathbf{x}_n$ 和测量值 $\mathbf{y}_n$ 的互协方差阵矩阵

$$\mathbf{P}_{xy,n|n-1} = \mathbf{P}_{yx,n|n-1}^T = \int_{\mathbb{R}^M} \underbrace{\mathbf{x}_n\mathbf{b}^T(\mathbf{x}_n)}_{\mathbf{x}_n\text{与}\mathbf{b}(\mathbf{x}_n)\text{的外积}} \underbrace{\mathcal{N}(\mathbf{x}_n; \hat{\mathbf{x}}_{n|n-1}, \mathbf{P}_{n|n-1})}_{\text{高斯分布}}\mathrm{d}\mathbf{x}_n - \underbrace{\hat{\mathbf{x}}_{n|n-1}\hat{\mathbf{y}}_{n|n-1}^T}_{\text{估计值}\hat{\mathbf{x}}_{n|n-1}\text{与}\hat{\mathbf{y}}_{n|n-1}\text{的外积}} \tag{14.100}$$

式(14.95)、式(14.96)、式(14.98)到式(14.100)，这五个积分公式针对着贝叶斯滤波器近似的不同方面。然而，这些公式都不相同，它们的被积函数有一个共同形式：非线性函数和相应的已知均值、协方差矩阵的高斯函数的乘积。所以，这五个积分使用数值积分法提供近似。

最重要的是，状态的滤波估计的递归计算是建立在线性卡尔曼滤波器理论上的，遵循以下几点：

- 卡尔曼增益按以下公式计算

$$\mathbf{G}_n = \mathbf{P}_{xy,n|n-1}\mathbf{P}_{yy,n|n-1}^{-1} \tag{14.101}$$

- 收到新的观测值 $\mathbf{y}_n$ 的基础上，状态 $\mathbf{x}_n$ 的滤波估计预测值按预测-修正公式计算

$$\underbrace{\hat{\mathbf{x}}_{n|n}}_{\text{更新估计}} = \underbrace{\hat{\mathbf{x}}_{n|n-1}}_{\text{旧估计}} + \underbrace{\mathbf{G}_n}_{\text{卡尔曼增益}} \underbrace{(\mathbf{y}_n - \hat{\mathbf{y}}_{n|n-1})}_{\text{新息过程}} \tag{14.102}$$

- 相应地，滤波估计误差的协方差矩阵按下式计算

$$\mathbf{P}_{n|n} = \mathbf{P}_{n|n-1} - \mathbf{G}_n\mathbf{P}_{yy,n|n-1}\mathbf{G}_n^T \tag{14.103}$$

分别注意以下公式的一致性：新的非线性滤波器的式(14.101)、式(14.102)、式(14.103)和卡尔曼滤器的式(14.31)、式(14.30)、先前未编号的式(14.38)。在任何情况下，后验分布最终能按如下定义的高斯分布来计算：

$$p(\mathbf{x}_n \mid \mathbf{Y}_n) = \mathcal{N}(\mathbf{x}_n; \hat{\mathbf{x}}_{n|n}, \mathbf{P}_{n|n}) \tag{14.104}$$

其中，均值$\hat{\mathbf{x}}_{n|n}$定义于式(14.102)中，协方差矩阵 $\mathbf{P}_{n|n}$由式(14.103)定义。

因此，已经开始在时间更新阶段计算先验分布 $p(\mathbf{x}_{n-1} \mid \mathbf{Y}_{n-1})$，通过测量更新阶段，递归循环按步骤进行，最后计算后验分布 $p(\mathbf{x}_n \mid \mathbf{Y}_n)$；接下来循环按需要重复进行。

显而易见，这个新的非线性滤波器称为数值积分卡尔曼滤波器（Arasaratnam and Haykin，2009）。这个新的非线性滤波器的重要性质总结如下：

1. 数值积分卡尔曼滤波器（CKF）是无导数在线逐次状态估计。

2. 在函数数量的评估中，通过使用数值积分法则计算的矩量积分的近似值都是线性的。并且，在数值积分法则中的点和相关权值是独立于式(14.84)的非线性函数 $\mathbf{f}(\mathbf{x})$的；因此，它们能够被非在线的计算和存储以提高滤波过程的速度。

3. 与 EKF 一起，CKF 的计算复杂度用 flops 来度量，以 M^3 增长，其中 M 是状态空间的维数。

4. 从原理的角度来看，CKF 建立在卡尔曼滤波器理论上，为了达到和提高数字的精确度，使用了平方根滤波的方法；这个合成的滤波器称为平方根数值积分卡尔曼滤波器（SCKF），它传播了预测和后验误差协方差矩阵的平方根（Arasaratnam and Haykin，2009）。

5. 最重要的是，在先验分布中的二阶矩是在后验分布中完全保留的。由于我们知道的关于状态的信息实际上是包含在观测值中的，我们可以说 CKF 完全保留了关于状态的二阶信息，该信息包含在观测序列中，因此 EKF 在精确度和可信度上有很好的效果。

6. CKF 是最新的对贝叶斯滤波器的直接近似，它最大程度上缓解了维数灾难的问题，但是，仅靠 CKF 不能解决这个问题。

在这些性质的结合下，数值积分卡尔曼滤波器成为周期性多层感知器的有监督训练的受关注的方法，这将在第 15 章中讨论。在第 15 章中，我们也提出了一个计算机实验，它清楚地证明了这个新的强大工具的实用性。

14.8　粒子滤波器

在这一节，我们将通过贝叶斯滤波器的间接的全局近似，继续讨论非线性滤波问题。非线性滤波的第二种方法包含的基础理论（其中的大部分，并非全部），来源于 Monte Carlo 统计算法（Robert and Casella，2004）。粒子滤波器是这一新类型的非线性滤波器中的最好的例子。最重要的是，粒子滤波器已经成为一个解决非线性滤波问题的重要工具，因为它能应用于很多领域，例如信号处理、雷达和声音媒体的目标跟踪、计算机视觉、神经计算，这里只列出一部分。

在详细阐述粒子滤波器之前，先引入一些新记法和定义。令 $\mathbf{X}_n$ 表示所有的目标状态序列 $\{\mathbf{x}_i\}_{i=1}^n$。与前文一致，$\mathbf{Y}_n$ 表示所有观测序列$\{\mathbf{y}_i\}_{i=1}^n$。相应地，我们可以表示给定观测序列 $\mathbf{Y}_n$ 的条件下，所有状态 $\mathbf{X}_n$ 的联合后验分布为 $p(\mathbf{X}_n \mid \mathbf{Y}_n)$。由于 $\mathbf{X}_n$ 表示的状态序列对观测者是隐藏的，为计算式(14.84)的积分，获得直接从后验分布 $p(\mathbf{X}_n \mid \mathbf{Y}_n)$中的随机取样通常是不可行的。为了绕开这个实际困难，我们从另外一个分布中取样，这一分布称为工具（instrumental）分布，或者重要分布。今后，这个新的分布用 $q(\mathbf{X}_n \mid \mathbf{Y}_n)$来表示。顺其自然地，为了使重要分布能够有效地代替后验分布，$q(\mathbf{X}_n \mid \mathbf{Y}_n)$必须有一个足够广的支集，以完全包括 $p(\mathbf{X}_n \mid \mathbf{Y}_n)$的支集。

Monte Carlo 积分

按照所谓的重要性抽样方法，我们从重要性分布 $q(\mathbf{X}_n \mid \mathbf{Y}_n)$中随机地抽取 N 个统计独立且

同分布的（idd）样本构成一个集合。令 n 时刻随机取出的样本集记为 $\mathbf{x}_n^{(i)}, i = 1,2,\cdots,N$。从零时刻开始直到 n 时刻，一步一步地，在状态空间中根据重要性分布 $q(\mathbf{X}_n \mid \mathbf{Y}_n)$，$N$ 个样本追踪自己的“轨迹”。它们的轨迹记为 $\mathbf{X}_n^{(i)}$，其中 $i = 1,2,\cdots,N$，称为粒子，因此命名为“粒子滤波器”。

下面，我们定义重要性函数为

$$r(\mathbf{X}_n \mid \mathbf{Y}_n) = \frac{p(\mathbf{X}_n \mid \mathbf{Y}_n)}{q(\mathbf{X}_n \mid \mathbf{Y}_n)} \tag{14.105}$$

然后，利用式(14.84)的定义，我们可以改写贝叶斯估计量的公式

$$\bar{h}_n = \int h(\mathbf{X}_n)\left(\frac{p(\mathbf{X}_n \mid \mathbf{Y}_n)}{q(\mathbf{X}_n \mid \mathbf{Y}_n)}\right) q(\mathbf{X}_n \mid \mathbf{Y}_n)\mathrm{d}\mathbf{x}_n = \int h(\mathbf{X}_n) r(\mathbf{X}_n \mid \mathbf{Y}_n) q(\mathbf{X}_n \mid \mathbf{Y}_n)\mathrm{d}\mathbf{x}_n \tag{14.106}$$

其中，我们使用了 $h(\mathbf{X}_n)$作为任意函数，为了使它和粒子滤波的术语保持一致性。

在式(14.106)的贝叶斯估计量上运用重要性取样方法，我们得到相应的 Monte Carlo 估计量

$$\hat{h}_n(N) \approx \frac{1}{N}\sum_{i=1}^{N} \tilde{w}_n^{(i)} h(\mathbf{X}_n^{(i)}) \tag{14.107}$$

其中，$\tilde{w}_n^{(i)}$是重要性权重，定义为

$$\tilde{w}_n^{(i)} = r(\mathbf{X}_n^{(i)} \mid \mathbf{Y}_n) = \frac{p(\mathbf{X}_n^{(i)} \mid \mathbf{Y}_n)}{q(\mathbf{X}_n^{(i)} \mid \mathbf{Y}_n)}, \quad i = 1,2,\cdots,N \tag{14.108}$$

为了确保 Monte Carlo 估计量$\hat{h}_n(N)$不需要知道分布 $p(\mathbf{X}_n^{(i)} \mid \mathbf{Y}_n)$的正规化常数，这可能导致很多麻烦或者无法计算，所以通常情况下我们需要标准化重要性权值，使得它们的和为单位一。最后，我们改写式(14.107)估计量的公式

$$\hat{h}_n(N) \approx \sum_{i=1}^{N} w_n^{(i)} h(\mathbf{X}_n^{(i)}) \tag{14.109}$$

其中

$$w_n^{(i)} = \frac{\tilde{w}_n^{(i)}}{\sum_{j=1}^{N} \tilde{w}_n^{(j)}}, \qquad i = 1,2,\cdots,N \tag{14.110}$$

对有限数量的粒子，N 个，估计量$\hat{h}_n(N)$是“有偏的”。但是，在渐进的意义上，我们发现了下面的关系（Doucet 等，2001）：

$$\lim_{N\to\infty} \hat{h}_n(N) \to \bar{h}_n \tag{14.111}$$

为了改进重要性取样方法，我们可以按照它进行重采样的第二阶段，像在 Rubin（1998）的 sampling-importance-resampling（SIR）方法中那样。在 SIR 方法的第一个阶段，在第 n 次循环用通常的方法，随机地从重要性分布 $q(\mathbf{X}_n \mid \mathbf{Y}_n)$取样，得到一个 iid 样本集合$\{\mathbf{X}_n^{(i)}\}_{i=1}^{N}$，接着根据式(14.110)计算出相应的标准化重要性权值集合$\{w_n^{(i)}\}_{i=1}^{N}$。在 SIR 方法的第二个阶段，第二个样本集合表示为$\{\tilde{\mathbf{X}}_n^{(i)}\}_{i=1}^{M}$，从中间集合$\{\mathbf{X}_n^{(i)}\}_{i=1}^{N}$中提取得到，考虑到标准化重要性权值$w_n^{(i)}$的相关强度；实际上，每一个权重可以看做一个相关样本出现的概率。取样的第二个阶段背后的基本原理可以归纳为：

重取样的第二阶段取出的样本$\tilde{\mathbf{X}}_n^{(i)}$，它的标准化重要性权值 $w_n^{(i)}$很大，很有可能服从联合后验分布 $p(\mathbf{X}_n \mid \mathbf{Y}_n)$；比起标准化重要性权值小的样本，应该以更大的概率选择这样的样本。

实现 SIR 的方式有几种。Cappé 等（2005）介绍了一种方法，在每一次循环中我们进行以下操作：

1. 采样。随机地从重要性分布 $q(\mathbf{X}|\mathbf{Y})$中抽取一个 N 个样品$\{\mathbf{X}^{(i)}\}_{i=1}^{N}$的 iid 集合。

2. 加权。利用式(14.110)，计算相关的标准化权值$\{w^{(i)}\}_{i=1}^{N}$的集合。

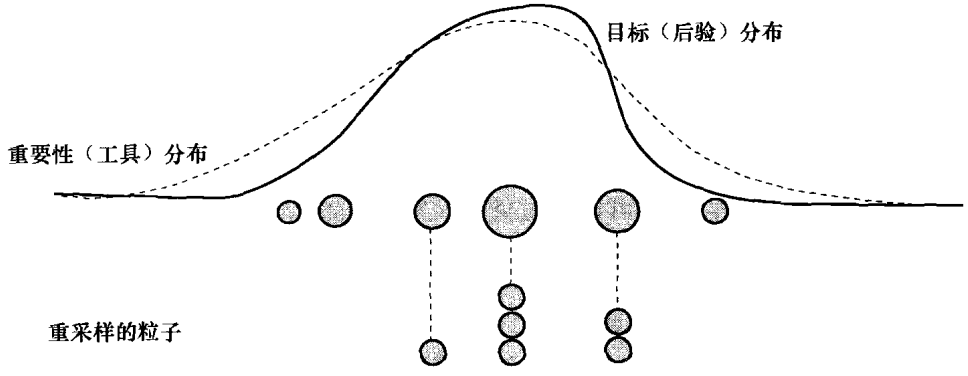


图 14.5　对样本数的样本进行重采样过程的说明，重采样 6 个样本

3. 重采样。

(i) 给定中间样本 $\mathbf{X}^{(1)},\mathbf{X}^{(2)},\cdots,\mathbf{X}^{(N)}$，条件独立地抽取含 L 个离散随机变量 $\{I^{(1)},I^{(2)},\cdots,I^{(L)}\}$ 的集合，从集合 $\{1,2,\cdots,N\}$ 中依概率 $(w^{(1)},w^{(2)},\cdots,w^{(N)})$ 取值，如以下的例子所示：

$$P(I^{(1)}=j)=w^{(j)},\quad j=1,2,\cdots,N$$

对 $I^{(2)},\cdots,I^{(L)}$ 等，典型地，有 $L\leqslant N$。

(ii) 设$\tilde{X}^{(i)}=X^{(I_i)}$，其中 $i=1,2,\cdots,L$。

集合$\{I^{(1)},\ I^{(2)},\ \cdots,\ I^{(L)}\}$被认为是多项式实验过程。因此，根据 SIR 方法被描述为一种多项式类型，可以从图 14.5 中的例子 $L=N=6$ 中看到。

在本节的后续内容里，我们将会讨论重采样在克服重要性权值的退化问题上的作用。然而，使用重采样引入了一些它自身的实际限制：

1. 重采样限制了粒子滤波器的并行执行的范围，这是由这一过程的本质决定的。

2. 在重采样期间，与大的重要性权值相关的粒子多次被选择，这导致了粒子多样性的损失；这一现象称为采样枯竭或者权值退化。例如，当空间状态模型的动态噪声相对小时，在几次循环后，所有的粒子可能会最终崩溃断裂成一个粒子，这显然是我们不希望看到的。

3. 始终不变的是，重采样增加了 Monte Carlo 估计量的方差。

顺序重要性采样

在式(14.109)中提及的 Monte Carlo 估计量$\hat{h}_n(N)$，由重要性采样方法得到，对任意函数 $h(\mathbf{X}_n)$的贝叶斯估计量$\hat{h}_n$ 的近似，提供了一个计算上可行的解，因此，满足我们非线性滤波器目标的第一个实际要求，这点在前面给出了详细的说明。然而，仍然需要满足第二个要求：Monte Carlo 估计量的递归实现。

不幸地，重要性采样方法的简单形式不满足递归计算的需要。这是因为在我们对后验分布 $p(\mathbf{X}_n|\mathbf{Y}_n)$做估计之前，需要完整的观测序列，用 $\mathbf{Y}_n$ 表示。特别地，每得到一个新的观测值 $\mathbf{y}_n$，需要对整个状态序列 $\mathbf{X}_n$ 计算重要性权值$\{\tilde{w}_n^{(i)}\}_{i=1}^{N}$。为了满足这个需求，重要性采样过程的计算复杂度将会随着时间 n 继续增加，这显然是不切实际的。为了解决这个计算上的困难，我们采样重要性采样的一个顺序实现，通常称为序贯重要性采样 (SIS)。

为了描述 SIS 程序的基本原理，首先我们用式(14.80)的时间更新和式(14.81)的测量更新去消除预测分布，这里我们用 $p(\mathbf{X}_n|\mathbf{Y}_{n-1})$、$p(\mathbf{X}_{n-1}|\mathbf{Y}_{n-1})$分别代替 $p(\mathbf{x}_n|\mathbf{Y}_{n-1})$和 $p(\mathbf{x}_{n-1}|\mathbf{Y}_{n-1})$，以便和粒子滤波器的术语一致。因此我们得到

$$\underbrace{p(\mathbf{X}_n \mid \mathbf{Y}_n)}_{\text{更新后验}} = \int \frac{1}{Z_n} p(\mathbf{x}_n \mid \mathbf{x}_{n-1}) l(\mathbf{y}_n \mid \mathbf{x}_n) p(\mathbf{X}_{n-1} \mid \mathbf{Y}_{n-1}) \mathrm{d}\mathbf{x}_{n-1}$$

$$= \int \frac{1}{Z_n} \underbrace{p(\mathbf{x}_n \mid \mathbf{x}_{n-1})}_{\text{先验}} \underbrace{l(\mathbf{y}_n \mid \mathbf{x}_n)}_{\text{似然函数}} \frac{p(\mathbf{X}_{n-1} \mid \mathbf{Y}_{n-1})}{q(\mathbf{X}_n \mid \mathbf{Y}_n)} \underbrace{q(\mathbf{X}_n \mid \mathbf{Y}_n) \mathrm{d}\mathbf{x}_{n-1}}_{\text{重要性分布}} \tag{14.112}$$

在等式的第一行中，我们将似然函数 $l(\mathbf{y}_n \mid \mathbf{x}_n)$ 移到了积分内，在马尔可夫假设下，它独立于先前的状态值 $\mathbf{x}_{n-1}$；在等式的第二行中，引入了重要性分布 $q(\mathbf{X}_n \mid \mathbf{Y}_n)$。在重要性采样的框架下，多个项的乘积

$$\frac{1}{Z_n} p(\mathbf{x}_n \mid \mathbf{x}_{n-1}) l(\mathbf{y}_n \mid \mathbf{x}_n) \frac{p(\mathbf{X}_{n-1} \mid \mathbf{Y}_{n-1})}{q(\mathbf{X}_n \mid \mathbf{Y}_n)}$$

是在 n 时刻关于重要性分布的重要性权值。特别地，由于 Z_n 是一个常数，可以写为

$$w_n^{(i)} \propto \frac{p(\mathbf{x}_n^{(i)} \mid \mathbf{X}_{n-1}^{(i)}) l(\mathbf{y}_n \mid \mathbf{x}_n^{(i)}) p(\mathbf{X}_{n-1}^{(i)} \mid \mathbf{Y}_{n-1})}{q(\mathbf{X}_n^{(i)} \mid \mathbf{Y}_n)} \tag{14.113}$$

这里∝表示成比例。

假设现在按以下的方式选择重要性分布，在式(14.113)中的分母中，因式分解

$$q(\mathbf{X}_n^{(i)} \mid \mathbf{Y}_n) = q(\mathbf{X}_{n-1}^{(i)} \mid \mathbf{Y}_{n-1}) q(\mathbf{x}_n^{(i)} \mid \mathbf{X}_{n-1}^{(i)}, \mathbf{y}_n) \tag{14.114}$$

对所有的 i 成立。然后，来自重要性分布 $q(\mathbf{X}_n^{(i)} \mid \mathbf{Y}_n)$ 的更新后的样品序列，简单地通过以下方式获得，得到一个新的观测值 $\mathbf{y}_n$，用新重要性分布 $q(\mathbf{x}_n^{(i)} \mid \mathbf{X}_{n-1}^{(i)}, \mathbf{y}_n)$ 的样本序列，来增大从重要性分布 $q(\mathbf{X}_{n-1}^{(i)} \mid \mathbf{Y}_{n-1})$ 中提取的旧样本序列。因此，式(14.114)可以看做序贯重要性采样的“把戏”。在任何情况下，在式(14.113)中使用式(14.114)的分解，我们得到

$$\tilde{w}_n^{(i)} \propto \frac{p(\mathbf{X}_{n-1}^{(i)} \mid \mathbf{Y}_{n-1})}{q(\mathbf{X}_{n-1}^{(i)} \mid \mathbf{Y}_{n-1})} \times \frac{p(\mathbf{x}_n^{(i)} \mid \mathbf{X}_{n-1}^{(i)}) l(\mathbf{y}_n \mid \mathbf{x}_n^{(i)})}{q(\mathbf{x}_n^{(i)} \mid \mathbf{X}_{n-1}^{(i)}, \mathbf{y}_n)} \tag{14.115}$$

一个实际而有趣的情况是，在每个时间步骤 n 中，只有一个后验分布 $p(\mathbf{X}_n \mid \mathbf{Y}_n)$ 的滤波估计。在这种情况下，我们可以设

$$q(\mathbf{x}_n^{(i)} \mid \mathbf{X}_{n-1}^{(i)}, \mathbf{y}_n) = q(\mathbf{x}_n^{(i)} \mid \mathbf{x}_{n-1}^{(i)}, \mathbf{y}_n) \text{ 对于所有的 } i$$

和 $p(\mathbf{x}_n^{(i)} \mid \mathbf{X}_{n-1}^{(i)})$。在这种情况下，我们只需要保存当前状态 $\mathbf{x}_n^{(i)}$，因此丢弃旧的轨迹 $\mathbf{X}_{n-1}^{(i)}$ 和观测值 $\mathbf{Y}_{n-1}$ 的相关历史记录。相应地，更新重要性权值的式(14.115)化简为

$$\underbrace{w_n^{(i)}}_{\substack{\text{更新}\\ \text{重要性}\\ \text{权值}}} \propto \underbrace{w_{n-1}^{(i)}}_{\substack{\text{旧的}\\ \text{重要性}\\ \text{权值}}} \times \underbrace{\frac{p(\mathbf{x}_n^{(i)} \mid \mathbf{x}_{n-1}^{(i)}) l(\mathbf{y}_n \mid \mathbf{x}_n^{(i)})}{q(\mathbf{x}_n^{(i)} \mid \mathbf{x}_{n-1}^{(i)}, \mathbf{y}_n)}}_{\text{增量修正因子}}, \text{对于所有的 } i \tag{14.116}$$

其中，∝表示成比例。式(14.116)是在时间上递归的估计标准化重要性权值的一个必要公式；它满足非线性滤波目标的第二个要求，粒子滤波器的递归实现。特别地，SIS 程序在每一个时间步骤中，每当获得一个新的观测值就传播重要性权值。式(14.116)等号右边的乘法因子，允许“旧的”重要性权值在时间步骤 n 中，当获得新的观测值 $\mathbf{y}_n$ 时被更新，这个因子称为*增量修正因子*。

显然，序贯重要性采样应用于后验分布 $p(\mathbf{x}_n \mid \mathbf{Y}_n)$ 的 Monte Carlo 估计同样有好的效果；根据式(14.112)和式(14.116)，可以写作

$$p(\mathbf{x}_n \mid \mathbf{Y}_n) \approx \sum_{i=1}^{N} w_n^{(i)} \delta(\mathbf{x}_n - \mathbf{x}_n^{(i)}) \tag{14.117}$$

这里 $\delta(\mathbf{x}_n - \mathbf{x}_n^{(i)})$ 是*迪拉克德耳塔函数*，它是位于 $\mathbf{x}_n = \mathbf{x}_n^{(i)}$ 对 $i = 1, 2, \cdots, N$，并且对滤波情况，根据式(14.116)更新权值。随着粒子的数量 N，趋近于无穷，式(14.117)的估计值接近于真实的后验分布 $p(\mathbf{x}_n \mid \mathbf{Y}_n)$。

权值退化问题

重要性分布 $q(\mathbf{X}_n \mid \mathbf{Y}_n)$ 扮演着粒子滤波器设计方面的关键角色。由于它与后验分布 $p(\mathbf{X}_n \mid \mathbf{Y}_n)$ 总是不同，我们发现式(14.108)中定义的重要性权值的方差，能够仅仅随着时间的增加而增大。这个现象，在使用序贯重要性采样时遇到过，从而导致了之前提及的权重值退化问题。

直觉上的权值退化问题的解释，在时间步骤 n 中，考虑一个具有标准化重要性权值 $w_n^{(i)}$ 的粒子 $\mathbf{X}_n^{(i)}$。根据定义，一个小的权值意味着粒子 $w_n^{(i)}$ 已经从重要性分布 $q(\mathbf{X}_n \mid \mathbf{Y}_n)$ 中取样得到，以一个合适的距离远离后验分布 $p(\mathbf{X}_n \mid \mathbf{Y}_n)$ 的主体，因此意味着这个特别粒子的分布对式(14.109)里的 Monte Carlo 估计量 $\hat{h}_n(N)$ 不起作用。当退化问题变得严重时，有大量的不起作用的粒子，结果导致 Monte Carlo 估计量 $\hat{h}_n(N)$ 在统计上和计算上一样是没有效率。在这种情况下，少数量的粒子承担起计算的责任。然而更严重的是，随着时间步骤 n 的增加，我们发现在粒子总体的多样性减少，并且估计量 $\hat{h}_n(N)$ 的方差增大，构成了一个不好的情况。

为了警惕序贯重要性采样中的权值退化问题，我们显然需要一个退化度量。有了这个度量的概念，Liu(1996) 定义了一个有效的样本大小为

$$N_{\text{eff}} = \left[\sum_{i=1}^{N} (w_n^{(i)})^2\right]^{-1} \tag{14.118}$$

其中 $w_n^{(i)}$ 是式(14.110)中的标准化重要性权值。应用这个简单的公式时，需要考虑两个极端的情况：

1. 当 N 权值的分布都是均衡的，对所有的 i，$w_n^{(i)} = 1/N$，这时 $N_{\text{eff}} = N$。

2. 除了一个权值是单位元的，所有的 N 个权值都为零，在这种情况下，$N_{\text{eff}} = 1$。

继续遵循以上原则，因此，N_{eff} 的取值范围是[1，N]。特别地，一个小的 N_{eff} 值意味着权值退化的一个严重情况，反之亦然。

因此关键问题为：

意识到在序贯重要性采样中的权值退化问题是规则而不是例外，我们怎么能解决它呢？

这个基础问题的回答包含在本节之前讨论的重采样的使用中。例如，粒子滤波器算法的算法的公式化可以包含一个规定的阈值，记为 N_{thr}。当有效的样品大小 N_{eff} 低于阈值 N_{thr} 时，SIS 程序暂时的停止并且运用重采样步骤，而后 SIS 程序再继续执行；这个过程将重复进行直到滤波器被终止。

采样重要性重采样粒子滤波器

第一次粒子滤波器的粒子实现是 Gordon、Salmond and Smith (1993) 记录的，当时命名为“bootstrap 滤波器”。在 Gordon、Salmond、Smith 的论文发表之前，序贯重要性采样中的权值退化的严重问题，既没有清楚的定义也没有令人满意的解决方法。在 1993 年的论文中，权值退化问题通过一个复原过程被解决，依靠删去相关权值小的粒子，权值大的粒子不仅保留下来而且被复制，这点在很大程度上与传统的非序贯采样过程相同。的确，由于这个原因，现在 bootstrap 滤波器一般被认为是采样重要性重采样 (SIR) 滤波器。这段简要的历史记录中的重要的一点是，SIR 滤波器是第一个成功使用 Monte Carlo 模拟进行非线性滤波的证明。

SIR 滤波器的实现简单，因此经常用于解决非线性滤波问题。这一滤波器有两方面与众不同的特色：

1. *将先验分布视为重要性分布。*检查为式(14.116)更新权值的递归公式，我们看到重要

性分布的定义是靠如何选择等式右边的分母 $q(\mathbf{x}_n^{(i)} \mid \mathbf{x}_{n-1}^{(i)}, \mathbf{y}_n)$ 来确定的。在 SIR 滤波器中，这个选择是依据下面的公式得出

$$q(\mathbf{x}_n \mid \mathbf{x}_{n-1}, \mathbf{y}_n) = p(\mathbf{x}_n \mid \mathbf{x}_{n-1}) \tag{14.119}$$

其中，在的等式的右边，$p(\mathbf{x}_n \mid \mathbf{x}_{n-1})$是先验分布或者状态转移分布。实际上，SIR 滤波器盲目地从先验分布 $p(\mathbf{x}_n \mid \mathbf{x}_{n-1})$中取样，完全忽略了包含在观测值 $\mathbf{y}_n$ 中的关于状态 $\mathbf{x}_n$ 的信息。式(14.119)由马尔可夫假设产生。

2. *采样重要性重采样*。在 SIR 滤波器中，重采样被运用在非线性滤波器过程的每一个时间步骤；因此，由式(14.116)我们得到

$$w_{n-1}^{(i)} = 1/N, \quad \text{当 } i = 1,2,\cdots,N \tag{14.120}$$

因为 $1/N$ 是一个常数，它可以被忽略。因此，需要在式(14.116)的增量修正因子随时间的累积就不再需要了。

因此，在式(14.116)中运用式(14.119)和式(14.120) 得到一个更简单的公式

$$\tilde{w}_n^{(i)} \propto l(\mathbf{y}_n \mid \mathbf{x}_n^{(i)}), \quad \text{当 } i = 1,2,\cdots,N \tag{14.121}$$

这里 $l(\mathbf{y}_n \mid \mathbf{x}_n^{(i)})$是观测值 $\mathbf{y}_n$ 的似然函数，给定粒子 i 的状态 $\mathbf{x}_n^{(i)}$。自然地，重要性权值标准化的计算，用到式(14.121)的概率，SIR 滤波算法的每一个重采样步骤之后执行的。表 14.5 总结了 SIR 滤波器。

表 14.5　粒子滤波的 SIR 算法总结

记法

粒子用 $i = 1,2,\cdots,N$ 来表示，其中 N 是粒子的总数。

初始化

给定状态分布 $p(\mathbf{x})$和 $\mathbf{x}$ 的初始值 $\mathbf{x}_0$，随机取样

$\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0)$

其中记号“$x \sim p$”是“x 是分布 p 的一个观察值”的简写，设置初始权值

$w_0^{(i)} = \frac{1}{N}$

其中 $i = 1,2,\cdots,N$。

循环

对每个时间步骤 $n = 1,2,3,\cdots$，按照下标 $i = 1,2,\cdots,N$，做如下操作：

1. 重要性分布定义为

   $q(\mathbf{x}_n \mid \mathbf{x}_{n-1}^{(i)}, \mathbf{y}_n) = p(\mathbf{x}_n \mid \mathbf{x}_{n-1}^{(i)})$

   其中假设已知先验分布 $p(\mathbf{x}_n \mid \mathbf{x}_{n-1}^{(i)})$，取样

   $\mathbf{x}_n^{(i)} \sim p(\mathbf{x}_n \mid \mathbf{x}_{n-1}^{(i)})$

2. 计算重要性权值

   $\tilde{w}_n^{(i)} = l(\mathbf{y}_n \mid \mathbf{x}_n^{(i)})$

   其中也假设似然函数 $l(\mathbf{y}_n \mid \mathbf{x}_n^{(i)})$已知，因此，计算标准化权值

   $$w_n^{(i)} = \frac{\tilde{w}_n^{(i)}}{\sum_{j=1}^{N} \tilde{w}_n^{(j)}}$$

3. 重采样，一个含 N 个离散随机变量的集合 $\{I^{(1)}, I^{(2)}, \cdots, I^{(N)}\}$，在相关集合 $\{1,2,\cdots,N\}$ 中依照以下概率取值：

   $P(I^{(s)} = i) = w_n^{(i)}$

   因此，集合

   $\tilde{\mathbf{x}}_n^{(i)} = \mathbf{x}_n^{(i)}$

   并且

   $w_n^{(i)} = \frac{1}{N}$

4. 继续计算直到滤波完成。

从以上的讨论中，显然在 SIR 滤波器公式化中的假设是轻度的，总结如下：

1. 式(14.1)的过程模型中的非线性函数 $\mathbf{a}_n(\cdot,\cdot)$，以及式(14.2)的测量模型中的非线性函数 $\mathbf{b}_n(\cdot,\cdot)$，两者必须都是已知的。

2. 确定先验分布 $p(\mathbf{x}_n|\mathbf{x}_{n-1})$需要式(14.1)的动态噪声 $\boldsymbol{\omega}_n$ 的统计学知识；从动态噪声 $\boldsymbol{\omega}_n$ 的基础分布中抽取样本（粒子），也必须是允许的。

3. 包含在式(14.121)中的似然函数 $l(\mathbf{y}_n|\mathbf{x}_n)$，必须是已知的，反过来，这意味着在式(14.2)中的测量噪声 $\boldsymbol{v}_n$ 的统计信息是可得到的。

另外在 SIR 滤波器（就此而言，对任何粒子滤波器）的设计中需要提出的另一个问题是粒子个数 N 的合适值的选择。一方面，N 应该足够的大以满足式(14.111)渐进的结果。另一方面，由于在滤波的每一个时间步骤粒子同时行动，N 应该足够的小以便将计算负担控制在可处理的水平上。(这里，我们假设在重要性采样和重采样操作之后，粒子的个数保持着相同的值 N。) 因此 N 值的选择必须在两个冲突情况下做一个“折中”，这个问题只有在一个问题解决的基础上得到解决。

重要性分布的最佳选择

先验分布 $p(\mathbf{x}_n|\mathbf{x}_{n-1})$为选择重要性分布提供了一个有吸引力的方法，正如在 SIR 滤波器的情况下。然而，一个粒子滤波器设计的选择，可能导致在不利条件下的不良的表现。例如，如果输入数据被异常值所干扰的情况下恶化，我们拥有“无信息”的观测值，并且如果测量噪声的方差小，那么我们就有“非常翔实”的观测值。这时在给定观测值的情况下，有一个潜在的错配存在于状态的预测先验分布和后验分布之间。为了用“最佳”形式缓和这种错配，粒子应该在重要性分布之下，选择移动到状态空间，这被定义为（Doucet 等，2000；Cappé 等，2007）

$$q(\mathbf{x}_n|\mathbf{x}_{n-1},\mathbf{y}_n)_{\text{opt}}=\frac{p(\mathbf{x}_n|\mathbf{x}_{n-1})l(\mathbf{y}_n|\mathbf{x}_n)}{\int p(\mathbf{x}_n|\mathbf{x}_{n-1})l(\mathbf{y}_n|\mathbf{x}_n)\mathrm{d}\mathbf{x}_n} \tag{14.122}$$

这个重要性分布的特殊的选择是最优的，在这个意义上权值的条件方差为零时，给定了粒子先前的历史记录。

用式(14.122)替换式(1.116)的 SIS 公式中，得到权值更新的公式

$$\underbrace{w_n^{(i)}}_{\substack{\text{更新的}\\\text{权值}}} \propto \underbrace{w_{n-1}^{(i)}}_{\substack{\text{旧的}\\\text{权值}}} \int \underbrace{p(\mathbf{x}_n|\mathbf{x}_{n-1}^{(i)})}_{\text{先验}} l \underbrace{(\mathbf{y}_n|\mathbf{x}_n)\mathrm{d}\mathbf{x}_n}_{\text{似然函数}} \tag{14.123}$$

其中，我们看到增量修正因子（如积分项），仅仅取决于被提议的粒子 $\mathbf{x}_{n-1}^{(i)}$的“过去”的位置和当前的观测值 $\mathbf{y}_n$。

式(14.123)的最优公式和式(14.121)的 SIR 公式的一个重要不同点是：在 SIR 滤波器中，在状态空间中允许粒子盲目的移动，然而在式(14.122)的最佳重要性分布下，粒子允许在后验分布有大量的高概率的位置上聚类，这个显然是我们希望看到的情况。

然而，在式(14.122)中定义的最优重要性分布计算法可能并不是直接能进行的，除了在一些特殊的情况下。比如，在一类状态空间模型中，条件分布 $p(\mathbf{x}_n|\mathbf{x}_{n-1}^{(i)},\mathbf{y}_n)$是高斯分布，选择最优重要性分布去设计一个粒子滤波器的确是可行的（Doucet 等，2000）。

14.9　计算机实验：扩展的卡尔曼滤波器和粒子滤波器对比评价

比较评价的实验设置是建立在非线性高斯动态系统的状态空间模型之上的，该模型用以下

两个等式描述：

系统（状态）模型：

$$x_n = 0.5x_{n-1} + \frac{25x_{n-1}}{1+x_{n-1}^2} + 8\cos(1.2(n-1)) + \omega_n$$

测量（观测值）模型：

$$y_n = \frac{1}{20}x_n^2 + v_n$$

在这个系统中，动态噪声 ω_n 服从高斯分布$\mathcal{N}(0, 1)$，测量噪声 v_n 也服从高斯分布$\mathcal{N}(0, 1)$。状态的真实最初值为 $x_0=0.1$。

粒子滤波器的 SIR 版本应用于实验中。以下的实验条件应用在 EKF 和 SIR 滤波器中：

模仿状态轨迹：50 个时间步长

独立的 Monte Carlo 运行的数量：100

滤波估计的最初值：$\hat{x}_{0|0}=\mathcal{N}(x_0, 2)$

SIR 粒子滤波器的说明如下：

- 粒子的数量 N 的值是 100。
- 在滤波过程的每个时间步骤中运用重采样，随后进行重要性权值的标准化。
- 先验（如状态转换）分布应用于重要性分布中。

EFK 滤波器和 SIR 粒子滤波器的实验结果分别在图 14.6 和图 14.7 中给出。在每个图中，实线曲线表示真实的状态，标记为星号的点表示运行 50 次的平均结果。在图 14.6 和图 14.7，较高的和较低的用虚线连成的曲线分别表示用 EKF 和 PF 生成的状态估计的置信区间。

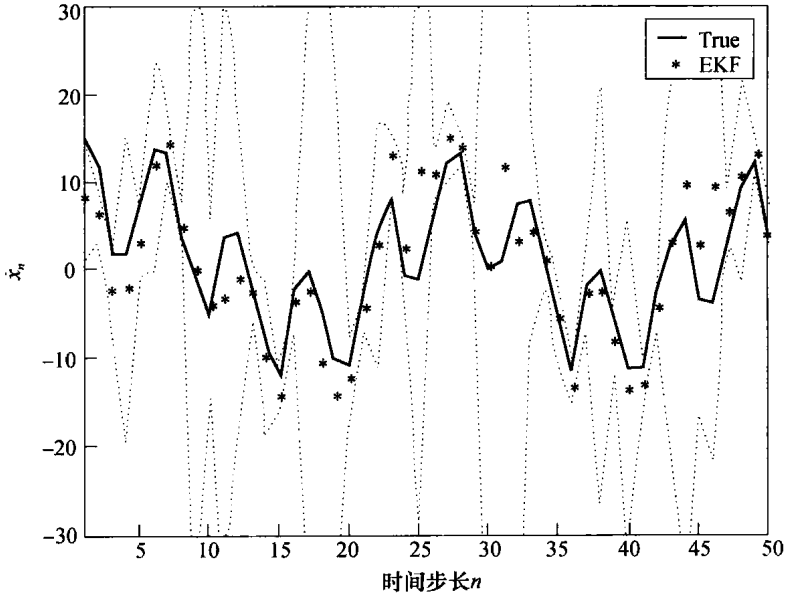


图 14.6　扩展的卡尔曼滤波器（EKF）的总体平均状态的估计$\hat{x}_n$，绘制成曲线，用连续的 * 点标记。较高的和较低的虚线连成的曲线（在估计值的附近），表示由扩展的卡尔曼滤波器生成的状态估计的置信区间。连续的曲线是状态随着时间 n 的真实变化过程

通过检查这两张图，揭示了如下的观察结果：

- 对于 EKF，状态滤波估计的平均轨迹明显的偏离了真实的轨迹。

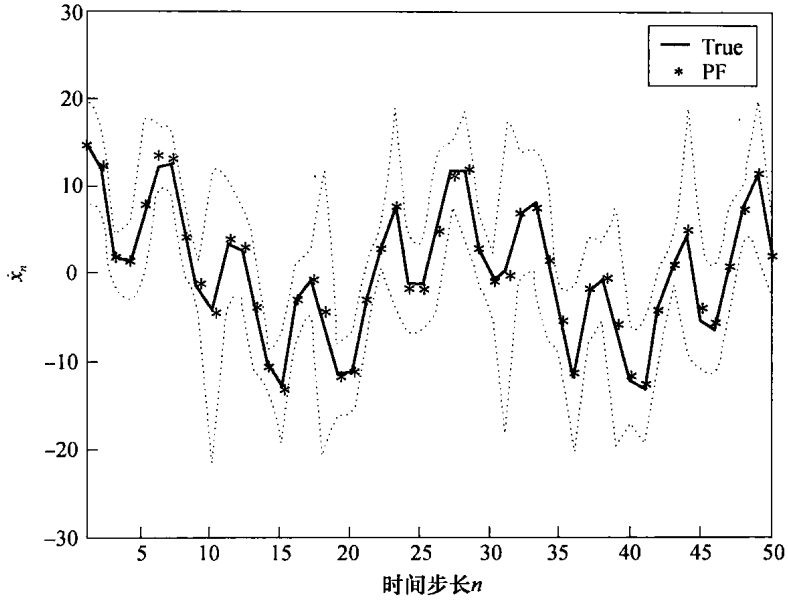


图 14.7　SIR 粒子滤波器的总体平均状态的估计$\hat{x}_n$，绘制成曲线，用连续的 * 点标记。较高的和较低的虚线连成的曲线（在估计值的附近），表示由粒子滤波器（PF）生成的状态估计的置信区间。连续的曲线是状态随着时间 n 的真实变化过程

• 另一方面，从 SIR 粒子滤波器计算出来的对应的平均轨迹，与真实的轨迹非常接近。

另一个实验的结果是关于粒子滤波器的，在图 14.8 中，状态的滤波估计的均方误差的平方根（RMSE），随着 SIR 粒子滤波器中使用的粒子数量的变化曲线被绘制。我们看到 RMSE 最初是很高的，随着粒子数量的增大而逐渐减少，同时粒子的数量在增加。粒子数量超过 $N=100$ 时，RMSE 没有显著的变化；在实验中为 SIR 滤波器选择 $N=100$ 个粒子，因此得到了证明。

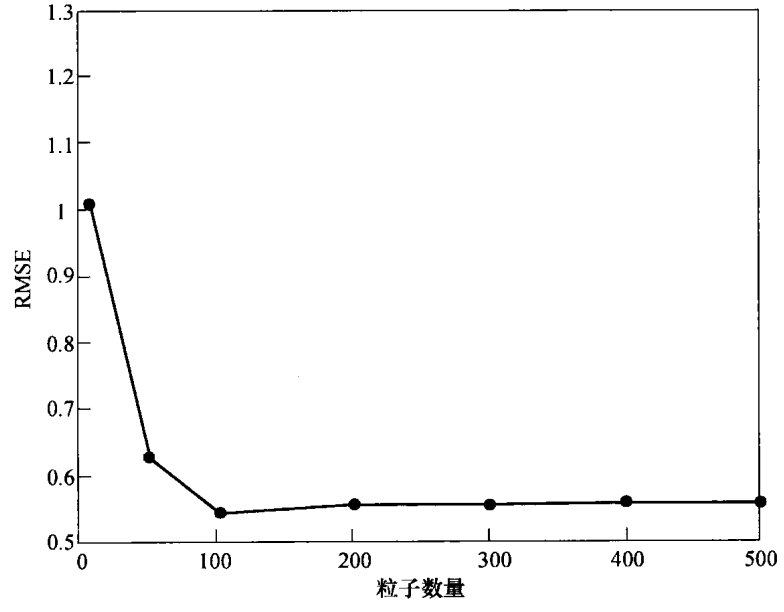


图 14.8　绘制了由 SIR 粒子滤波器生成的均方误差的平方根（RMSE）随粒子数量的变化的变化情况；点 · 是通过实验计算得到的

14.10　大脑功能建模中的卡尔曼滤波

到现在为止本章的讨论集中于卡尔曼滤波器的理论，随后是贝叶斯滤波器和它的近似形式。这样做，我们着重强调了这些滤波器以自己的方式进行逐次状态估计的实际功效。在这一节中，我们将综述类卡尔曼滤波器在不同的大脑功能建模中的应用（Chen 等，2007）。

视觉识别的动态模型

视觉的皮层包括一个层状结构层次（从 V1 到 V5）和大量的连接，这些连接处于皮层内以及皮层和视觉丘脑之间（如外侧膝状体核，或者 LGN）；对于视觉系统中的这一部分的一个简要概括，请参见第 12 章的参考文献。特别地，视觉皮层被赋予了两个重要解剖学的属性（Chen 等，2007）：

- *反馈的大量使用*。视觉皮层的任意两个位置的连接是相互的，因此调节向前以及反馈信号的传输。
- *分层多尺度结构*。视觉皮层范围内的下区细胞的感受域，只跨越视觉区域的一小部分，反之，高区细胞的感受域的大小增大，直到它们跨越了几乎整个视觉区域。正是这个约束网络使得它可以在高维的数据空间，为全连接的视觉皮层，用数量减少了的自由参数进行预测，因此这是一个计算上有效率的方法。

从 1997 年到 2003 年期间的一系列的研究，Rao 和他的合作者开发了这两个性质，来构建一个视觉识别的动态模型，以及了解到这一构想根本上是一个非线性的动态过程。视觉识别的 Rao-Ballard 模型是一个分层组织的神经网络，每一个中级分层接收到两种信息：来自于前一层的自下而上信息水平以及来自于较高层的自上而下的信息。为了它的实现，这个模型用一个多尺度估计算法，可能被看做一个从扩展的卡尔曼滤波器里的分层形式。特别地，通过一个动态环境下的视觉实验，EKF 被同时用作学习模型的前馈、反馈和预测参数。由此产生的适应过程运行在两个不同的时间尺度上：

- 快速动态状态估计过程允许这个动态模型去预计传入的刺激。
- 慢速 Hebb 学习过程，为突触权重模型提供了调整。

特别地，Rao-Ballard 模型可以看做一个 EKF 神经网络的实现，在各个层之间使用自顶向下的前馈，并且能够对静态图像和随时间变化的图像序列，训练视觉感受野。这个模型非常有吸引力，这是由于它简单性和灵活性，但是功能很强大。首先，它允许视觉感知的贝叶斯解释（Knill and Richards，1995；Lee and Mumford，2003）。

声音流分离的动态模型

众所周知，在计算神经科学的文献中，听觉感知与视觉感知有着许多相同的特征（Shamma，2001）。特别地，Elhilali（2004）提出了计算听觉场景分析框架下的声音流分离问题（CASA）。在其中描述的计算模型中，隐藏向量包含了一种声音流的区间（抽象）表示法；观测值由一个特征向量的集合或者从声音的混合体中得到的声学线索（如音高和开始）。由于声音流的暂时的连续性是一个重要的特性，它能够用作于构建系统（状态）模型。测量模型描述了含有皮层模型参数的皮层滤波过程。这个动态声音流分离的基础组成部分包括以下两个方面：第一，在每个时间点，推断声音模式的分布为流的一个集合；第二，给定新的观测值，估计每个聚类的状态。第二个估计问题已经用卡尔曼滤波器的操作解决，第一个聚类问题已经用类 Hebb 竞争性学习的操作解决。

卡尔曼滤波器的动态本质不仅对于声音流分离是很重要的，而且对于声音定位和跟踪也同样重要。所有的这些都被视为有效听觉的关键成分（Haykin and Chen，2006）。

小脑和运动学习的动态模型

小脑在运动的控制和协调中起到了非常重要的作用，通常进行得非常平稳并且几乎毫不费力。在文献中，已经提到小脑扮演着动态状态估计的控制者或者神经模拟的角色。支持动态状态估计假设的关键点包含在以下的叙述中，它的有效性已经被数十年的自动跟踪和指导系统设计的研究所证实：

任何一个生物或者人造的系统，需要预测或控制一个多元动态系统的随机轨迹，可以通过使用或引用的卡尔曼滤波的本质在这种或其他方式下才能有效。

建立在这个关键点之上，Paulin（1997）发表了一些关于支持小脑是一个动态状态估计值的神经模拟的假设的证据。特别是 Paulin 证据的有一行表述了关于前庭眼反射（VOR），它是眼球运动系统的一部分。VOR 的作用是去维持视觉图像的稳定性（如视网膜）通过与脑袋旋转相反的眼睛的转动，这点在之前的前言部分已经讨论过的那样。这个功能调节包括小脑皮层和前庭的核子的神经网络。从 14.3 节的讨论中，我们知道卡尔曼滤波器是一个最佳的线性系统，且伴通过噪声测量，预测一个动态系统状态轨迹的方差最小；给出一个对于潜在的系统动态的假设模型，它通过估计特别的状态轨迹做到上述这些。这个策略的结果是，当动态系统源自于假定模型，卡尔曼滤波器产生一种预测的估计错误，这个错误可能归因于滤波器“相信”假定模型而不是真实的感觉数据。根据 Paulin（1997），此类估计错误在 VOR 行为中被观测到。

总结归纳

总之，卡尔曼滤波器的预测修正的属性使它成为一个对计算神经建模中的预测编码问题的潜在有用的候选方法，这就是在动态环境下自主的大脑功能的一个基础属性。同样需要注意到的重要问题是，在之前提及的例子中，假设神经系统（如小脑或者新大脑皮层）是卡尔曼滤波器的神经模拟，它并不意味着在物质层面上，神经系统类似于卡尔曼滤波器。一般来说，生物系统的确表现出一些状态估计的形式，并且相关的神经算法可能含有卡尔曼滤波器的一般的“特征”。此外，一些貌似合理的状态估计形式广泛分布在中心神经系统的其他部分。

14.11　小结和讨论

本章讨论的主题是：给定一个依赖于状态的观测值序列，估计动态系统中未知的（隐藏的）状态。解决这个问题的基础在于状态空间模型，由两个公式组成：一个等式建模了状态随着时间的演变过程，并且含有来源于这个变化的动态噪声，另外一个等式建模了状态观测值的噪声版本。假设状态空间模型是服从马尔可夫的。

卡尔曼滤波器理论

当动态系统是线性的，并且服从高斯分布，状态的最优估计值是著名的卡尔曼滤波器。当动态系统是非线性的，并且服从高斯分布，我们可使用状态空间模型下的一阶泰勒展开近似得到的扩展的卡尔曼滤波器。假如是轻度的非线性，这个对于非线性滤波的近似方法得到可以接受的结果。

贝叶斯滤波器

从理论上说，贝叶斯滤波器是最为一般的非线性滤波器，卡尔曼滤波器则被视为它的一个特例。然而，在实际应用中实现贝叶斯滤波器，必须采用近似。这里的近似可以是以下两种方式中的一种：

1. *后验分布的直接数值近似。*第一个方法背后的思想总结如下：

通过线性卡尔曼滤波器原理用数值法使非线性动态系统状态的估计值近似变得容易。

使用该方法进行非线性滤波的例子包括扩展的卡尔曼滤波器，无气味的卡尔曼滤波器(Julier 等，2000)，正交卡尔曼滤波器（Ito and Xing，2000；Arasaratnam 等，2007）和数值积分卡尔曼滤波器（Arasaranam and Haykin，2009)。在这些非线性滤波器当中，扩展的卡尔曼滤波器是最简单的，数值积分卡尔曼滤波器是最强大的。简单地说，是用计算复杂度的增加换取可靠度的增加。

2. *后验分布的间接数值近似*。非线性滤波的第二个方法中，最突出并广泛使用的例子是粒子滤波器。由于贝叶斯滤波器的后验分布很难接近，我们凭借随机取样的方法，从必须支持后验分布的重要性，或者工具的分布里抽取样本。粒子滤波器的递归实现通过序贯重要性采样(SIS) 过程来完成的。为了避免滤波器的进入权值退化的情况，常用的方法是采用带重采样的重要性采样，依靠这一方法相对较弱的正规化的权值被删除，剩余的正规化权值根据它们出现的可能性被复制。

一方面，尽管有卡尔曼滤波器和它的变体以及它的近似扩展，而另一方面，粒子滤波器在它们的分析推论和实际实现上是根本不同的，虽然大家都分享同一个重要的性质：*预测-修正性质*。

计算上的考虑

(i) *卡尔曼滤波器*。无论何时，当我们开发一个滤波器的算法，通常要检查算法的收敛性。特别是，算法的使用者想知道能使算法收敛的条件，以及如何确定收敛问题。例如，众所周知卡尔曼滤波器会有收敛现象，以下两个因素是产生这一现象的原因：

- 状态空间模型间（卡尔曼滤波器的起源以此为基础）的模型错配，实际的动态环境的底层物理学负责观测值的产生；
- 卡尔曼滤波器的实际实现使用的不够精确的算术精度。

发散现象的根本可能涉及矩阵 $\mathbf{P}_{n|n}$违反了协方差矩阵的正定的性质。平方根滤波器提供了缓和发散现象的方法。

(ii) *粒子滤波器*。接下来开始考虑粒子滤波器的计算部分。给定粒子滤波的 Monte Carlo 根，这个观察结果其实并不令人感到奇怪。在任何情况下，我们总结了一些在文献中的重要结论：

1. 对于指定的粒子的数量 N，式(14.84)的积分的 Monte Carlo 估计引发的误差是 $O(N^{-1/2})$级的，它与状态向量的维数无关（Ristic 等，2004)。这个结果建立在两个假设之上：

- 在式(14.84)积分中的后验分布 $p(\mathbf{x}_n|\mathbf{Y}_n)$是明确已知的。
- 粒子（如样本）是统计独立的。

然而，粒子滤波中的这两个假设都违反了：精确知道 $p(\mathbf{x}_n|\mathbf{Y}_n)$是不可能的，并且在粒子滤波器中使用重采样，粒子轨迹变得有实际依赖性。

2. Crisan and Doucet（2002）提出的，粒子滤波器产生的估计的方差的上界表示为：$O(N^{-1/2})$乘以一个常量比例系数 c。

不幸的是，这个结果导致了错误的结论，粒子滤波器产生的估计误差与状态向量的维数无关，并因此免于维数灾难。Daum and Huang（2003）提出，争论乘数因子不是一个常量；而是随着时间 n 按指数级增长，因此记为 c_n。它非常依赖状态向量的维数，这意味着粒子滤波器确实经历了维数灾难。

3. 在 Bengtesson 等（2008）的独立性研究中，证明了用粒子滤波器的“brute-force-only”实现来描述高维后验分布将失败，这归因于维数灾难。应对这个现象的推荐的方法是在粒子滤

波之前先得到一些维数降低的形式；正如在第 10 章中指出的，高维数据经常是稀少的，因此可以降低维数。

注释和参考文献

1. **相关的动态和测量噪声。**在一个线性高斯状态空间模型中，动态噪声 $\boldsymbol{\omega}_n$ 和测量噪声 $\boldsymbol{v}_n$ 中的相关性有时候是允许的。这个条件被用在经济学中。特别地，我们现在有

$$\mathbb{E}[\boldsymbol{\omega}_n \boldsymbol{v}_k^T] = \begin{cases} \mathbf{C}_n & \text{当 } k = n \text{ 时} \\ \mathbf{0} & \text{当 } k \neq n \text{ 时} \end{cases}$$

其中 $\mathbf{C}_n$ 是已知的矩阵。根据这个等式，这两个噪声过程 $\boldsymbol{\omega}_n$ 和 $\boldsymbol{v}_n$ 是同时相关的，但是它们在非零延迟的情况下保持着不相关性。在这种情况下，卡尔曼滤波器的公式化必须进行修改。对这个问题的第一次讨论是在 Jazwinski (1970)；也可以参见 Harvey (1989)。

2. **信息滤波算法。**协方差滤波算法是实现卡尔曼滤波器的一种方法。在另一种称为信息滤波器算法的形式中，卡尔曼滤波器通过传播协方差矩阵 $\mathbf{P}_{n|n}$ 的逆来实现；这个逆与 Fisher 的信息矩阵是相关的，允许滤波器在信息理论形式的解释。关于信息滤波算法的更多细节，参见第 10 章 Haykin (2002)。

3. **记法。**为了式(14.6)的彻底正确并且与本书前面的记法已知，我们应该用 $p_{\mathbf{x}}(\mathbf{x})$ 代替 $p(\mathbf{x})$，代表随机变量 $\mathbf{X}$，它的样本值用 $\mathbf{x}$ 表示。我们已经在式(14.6)中使用了记号 $p(\mathbf{x})$，并且在本章中其他相似情况有以下两个原因：
   - 为了简化表示，因为本章有大量的随机过程的概率表示。
   - 最重要的是，避免在本章后面部分的混乱，在后面记号 $\mathbf{X}$ 用于表示状态的序列。

4. **贝叶斯估计。**估计理论中的一个经典的问题是随机参数贝叶斯估计。对这个问题有不同的答案，根据贝叶斯估计中的损失函数是如何被公式化的。一个特别而有趣的贝叶斯估计器类型是所谓的条件平均估计。在这种情况下，我们做两件事：
   (1) 从第一个原理获得条件均值估计量的公式。
   (2) 表明这个估计量与最小均方误差估计量是一样的。
   对于这些结果，考虑随机参数 x。给定了一个依赖于 x 的观测值 y，需要做的是估计 x。令 $\hat{x}(y)$ 表示参数 x 的一个估计值，符号 $\hat{x}(y)$ 强调了估计是观测值 y 的一个函数这一事实。令 R 表示损失函数，依赖于 x 和它的估计值。然后，根据贝叶斯估计理论，我们可以定义贝叶斯风险为：

$$R = \mathbb{E}[C(x,\hat{x}(y))] = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} C(x\,\hat{x}(y))\,p(x,y)\,\mathrm{d}x\mathrm{d}y \tag{A}$$

这里 $p(x, y)$ 是 x 和 y 的联合概率密度函数。对于一个具体的损失函数 $C(x, \hat{x}(y))$，贝叶斯估计 $\hat{x}(y)$ 被定义为最小化危险 R 的估计。

一个有特别引起大家兴趣的损失函数（这是很大程度上这本书涵盖内容的精神所在）是均方误差，具体化为估计误差的平方，它本身定义为实际参数值 x 和估计值 $\hat{x}(y)$ 的不同点，即

$$\varepsilon = x - \hat{x}(y)$$

相应地，我们写成

$$C(x,\hat{x}(y)) = C(x - \hat{x}(y))$$

或者，更简单地

$$C(\varepsilon) = \varepsilon^2$$

因此我们把公式(A)重新改写成

$$R_{\mathrm{ms}} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (x - \hat{x}(y))^2\,p(x,y)\,\mathrm{d}x\mathrm{d}y \tag{B}$$

其中风险 R_{ms} 的下标指出使用均方误差作为它的基础。从概率理论，我们得到

$$p(x,y) = p(x|y)p(y) \tag{C}$$

其中 $p(x|y)$ 是给定的 x 和 y 的条件概率密度函数，$p(y)$ 是 y 的（边缘的）概率密度函数。因此，将公式(C)代入到公式(B)里，我们得到

$$R_{\mathrm{ms}} = \int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty} (x - \hat{x}(y))^2\,p(x|y)\,\mathrm{d}x\right]p(y)\,\mathrm{d}y \tag{D}$$

现在我们意识到里面的积分（在方括号里）和公式(D)的 $p(y)$ 都是非负的。因此我们可能简单的通过最小

化里面的积分从而最小化风险 R_{ms}。令这样获得的估计值用$\hat{x}_{ms}(y)$表示。我们发现通过里面的积分对$\hat{x}(y)$求导，然后令结果等于零。

为了简化表述方法，令 I 表示公式(D)的里面的积分。然后 I 对$\hat{x}(y)$求导得到

$$\frac{dI}{d\hat{x}} = -2\int_{-\infty}^{\infty} xp(x|y)dx + 2\hat{x}(y)\int_{-\infty}^{\infty} p(x|y)dx \tag{E}$$

公式(E)的等号右边的第二个积分，表示在概率密度函数下的全部面积，因此值为单位一。因此，设定 $dI/d\hat{x}$等于零，我们得到

$$\hat{x}_{ms}(y) = \int_{-\infty}^{\infty} xp(x|y)dx \tag{F}$$

公式(F)定义的解是唯一的最小值。

公式(F)中定义的估计量$\hat{x}_{ms}(y)$是自然的最小均方误差估计量。对这个估计量的另一种解释，我们认识到给定观测值 y，等式右边的积分仅仅是参数 x 的条件平均。

因此得到结论最小均方误差估计量和条件平均估计量确实是同一个。换句话说，我们有

$$\hat{x}_{ms}(y) = \mathbb{E}[x|y] \tag{G}$$

用公式(G)替换$\hat{x}(y)$带入到公式(D)，我们发现里面的积分刚好是给定 y 的条件下，参数 x 的条件方差。相应地，风险 R_{ms}的最小值是对所有的观测值 y 这个条件方差的平均值。

5. *基于电位序列的贝叶斯滤波器*。在 14.10 节中讨论过大脑功能的动态建模，我们采样一个传统的信号处理框架，并重视对卡尔曼滤波理论的作用。

事实上，皮层神经网络从感官传入收到电位序列观察一个不确定的动态环境，而不是直接从环境观察。电位序列提供了在大脑中神经的主要交流通道；它们用峰电位到达的时间的形式来表示（Koch，1999；Rieke 等，1997）。Bobrowski 等（2007）考虑了动力环境隐藏状态概率分布的最佳估计问题，以电位序列的形式给出噪声观测值。最重要的是，它们描述了一个线性周期性的神经网络模型，这个模型可以切实的实现实时的贝叶斯滤波。这个输入可能是多模态的，由两个不同子集组成：例如，一个是视觉的，另一个听觉的。并且，提出了综合实例来证明系统的操作。

值得注目的是在连续时间内的非线性滤波，在点过程观测的基础上，第一次被Snyder（1972）描述过；也可以参见 Synder 的 1975 年出版的书中关于随机点过程的讨论。

习题

卡尔曼滤波器

14.1　预测状态误差向量被定义为

$$\boldsymbol{\varepsilon}_{n|n-1} = \mathbf{x}_n - \hat{\mathbf{x}}_{n|n-1}$$

这里$\hat{\mathbf{x}}_{n|n-1}$是状态 $\mathbf{x}_n$ 的最小均方估计，给定观测数据序列 $\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_{n-1}$。令 $\boldsymbol{\omega}_n$ 和 $\boldsymbol{v}_n$ 分别表示动态噪声和测量噪声向量。表示 $\boldsymbol{\varepsilon}_{n|n-1}$与 $\boldsymbol{\omega}_n$ 和 $\boldsymbol{v}_n$ 正交，可以写为

$$\mathbb{E}[\boldsymbol{\varepsilon}_{n|n-1}\boldsymbol{\omega}_n^T] = \mathbf{0}$$

和

$$\mathbb{E}[\boldsymbol{\varepsilon}_{n|n-1}\boldsymbol{v}_n^T] = \mathbf{0}$$

14.2　考虑一个均值为零的标量观测值 $\mathbf{y}_n$ 的集合，变换成相应的均值为零、方差为 $\sigma_{a,n}^2$的新息过程 α_n 的集合。给定数据集合，令状态向量 $\mathbf{x}_i$ 的估计值表示为如下形式

$$\hat{\mathbf{x}}_{i|n} = \sum_{k=1}^{n} \mathbf{b}_{i,k}\alpha_k$$

其中 $\mathbf{b}_{i,k}$，$k=1, 2, \cdots, n$ 是待确定向量的集合。需要选择 $\mathbf{b}_{i,k}$使得估计状态误差向量的范数的开平方的期望值最小

$$\boldsymbol{\varepsilon}_{i,n} = \mathbf{x}_i - \hat{\mathbf{x}}_{i|n}$$

这表明，最小化得到结果

$$\hat{\mathbf{x}}_{i|n} = \sum_{k=1}^{n} \mathbb{E}[\mathbf{x}_i\varphi_k]\varphi_k$$

其中

$$\varphi_k = \frac{\alpha_k}{\sigma_{\alpha,k}}$$

是正规化后的新息。这个结果可以看做是式(14.24)和式(14.26)的一个特例。

14.3 证明式(14.25)，这个式子说明了新息过程 α_k 和状态估计误差 $\boldsymbol{\varepsilon}_{i|n}$ 是不相关的，对于 $k = 1,2,\cdots,n$ 并且 $i \leqslant n$。

14.4 在卡尔曼滤波理论中证明，滤波的状态估计误差向量 $\boldsymbol{\varepsilon}_{i|n}$ 是均值为零，服从高斯分布的，并且是一阶马尔可夫过程。

14.5 卡尔曼增益 $\mathbf{G}_n$，由式(14.31)定义，包含逆矩阵 $\mathbf{R}_n^{-1}$。矩阵 $\mathbf{R}_n$ 是在式(14.22)中自定义的。这个矩阵 $\mathbf{R}_n$ 是正定的，但是并不需要是非奇异的。

(a) 为什么 $\mathbf{R}_n$ 是正定的?

(b) 为了保证逆矩阵 $\mathbf{R}_n^{-1}$ 存在，你选择什么样的先验分布作用于矩阵 $\mathbf{Q}_{v,n}$ 上?

14.6 在许多情况下，随着循环次数 n 趋于无穷大，预测误差协方差矩阵 $\mathbf{P}_{n+1|n}$ 收敛到稳定的状态值 $\mathbf{P}$。证明极限值 $\mathbf{P}$ 满足代数 Riccati 等式

$$\mathbf{PB}^T(\mathbf{BPB}^T + \mathbf{Q}_v)^{-1}(\mathbf{BP} - \mathbf{Q}_\omega) = 0$$

其中假设状态转移矩阵等于单位阵矩阵的 $\mathbf{B}$、$\mathbf{Q}_\omega$、$\mathbf{Q}_v$ 分别是 $\mathbf{B}_n$、$\mathbf{Q}_{\omega,n}$、$\mathbf{Q}_{v,n}$ 的极限值。

14.7 可以这样说，原始动态系统的状态空间模型嵌入了卡尔曼滤波器的结构。证明这一叙述。

14.8 在卡尔曼滤波器中预测修正框架的检查揭示了以下两个性质：

(a) 预测状态的 $\hat{\mathbf{x}}_{n+1|n}$ 和预测误差协方差矩阵 $\mathbf{P}_{n+1|n}$ 的计算仅仅依赖于从系统（状态）模型众提取的信息。

(b) 滤波状态 $\hat{\mathbf{x}}_{n|n}$ 和滤波误差协方差矩阵 $\mathbf{P}_{n|n}$ 的计算仅仅依赖于从测量模型众提取的信息。

证明卡尔曼滤波器的这两个性质。

14.9 预测误差协方差矩阵 $\mathbf{P}_{n+1|n}$ 和滤波误差协方差矩阵 $\mathbf{P}_{n|n}$ 不可以假设为同一个值。这是为什么?

14.10 在 14.3 节中卡尔曼滤波器的引出是建立在最小均方差估计的概念上的。在这个问题中，我们研究了另外一个卡尔曼滤波器的推导，以最大化后验概率（MAP）标准为基础。对于这个推导，假设动态噪声 $\boldsymbol{\omega}_n$ 和测量噪声 v_n 都是均值为零的高斯过程，协方差矩阵分别是 $\mathbf{Q}_{\omega,n}$ 和 $\mathbf{Q}_{v,n}$。令 $p(\mathbf{x}|\mathbf{Y}_n)$ 表示 $\mathbf{x}_n$ 的条件概率分布，给定 $\mathbf{Y}_n$ 表示观测值 $\mathbf{y}_1,\mathbf{y}_2,\cdots,\mathbf{y}_n$ 的集合。$\mathbf{x}_n$ 的 MAP 估计表示为 $\hat{\mathbf{x}}_{\text{MAP},n}$，定义为 $\mathbf{x}_n$ 的特殊值，使得 $p(\mathbf{x}_n|\mathbf{Y}_n)$ 最大化，或者等价为 $p(\mathbf{x}_n|\mathbf{Y}_n)$ 的对数。这个评价要求我们求解以下的条件

$$\left.\frac{\partial \log p(\mathbf{x}_n|\mathbf{Y}_n)}{\partial \mathbf{x}_n}\right|_{\mathbf{x}_n=\hat{\mathbf{x}}_{\text{MAP},n}} = 0 \tag{A}$$

表明

$$\left.\frac{\partial^2 \log p(\mathbf{x}_n|\mathbf{Y}_n)}{\partial^2 \mathbf{x}_n}\right|_{\mathbf{x}_n=\hat{\mathbf{x}}_{\text{MAP},n}} < 0 \tag{B}$$

(a) 我们可以将分布 $p(\mathbf{x}_n|\mathbf{Y}_n)$ 表示为

$$p(\mathbf{x}_n|\mathbf{Y}_n) = \frac{p(\mathbf{x}_n,\mathbf{Y}_n)}{p(\mathbf{Y}_n)}$$

鉴于联合分布的定义，也可以表示为如下的形式

$$p(\mathbf{x}_n|\mathbf{Y}_n) = \frac{p(\mathbf{x}_n,\mathbf{y}_n,\mathbf{Y}_{n-1})}{p(\mathbf{y}_n,\mathbf{Y}_{n-1})}$$

因此，表明

$$p(\mathbf{x}_n|\mathbf{Y}_n) = \frac{p(\mathbf{y}_n|\mathbf{x}_n)p(\mathbf{x}_n|\mathbf{Y}_{n-1})}{p(\mathbf{y}_n,\mathbf{Y}_{n-1})}$$

(b) 使用动态噪声 $\boldsymbol{\omega}_n$ 和测量噪声 v_n 的高斯特征，推导表达式 $p(\mathbf{y}_n|\mathbf{x}_n)$ 和 $p(\mathbf{x}_n|\mathbf{Y}_{n-1})$。接着认识到 $p(\mathbf{x}_n|\mathbf{Y}_{n-1})$ 可以作为一个常数，由于它不依赖于状态 $\mathbf{x}_n$，将 $p(\mathbf{x}_n|\mathbf{Y}_n)$ 公式化。

(c) 使用公式(A)中(b)部分的结果，根据矩阵求逆引理（在章节 5 中讨论过），推导出 $\hat{\mathbf{x}}_{\text{MAP},n}$ 的公式，证明它和在 14.3 节的卡尔曼滤波器的推导完全一致。

(d) 最后，证明(c)部分得到的 MAP 的估计 $\hat{\mathbf{x}}_{\text{MAP},n}$，确实满足公式(B)。

14.11 考虑一个无噪声状态空间模型描述的线性动态系统

$$\mathbf{x}_{n+1} = \mathbf{A}\mathbf{x}_n$$

和

$$\mathbf{y}_n = \mathbf{B}\mathbf{x}_n$$

其中 $\mathbf{x}_n$ 表示状态，$\mathbf{y}_n$ 是观测值，$\mathbf{A}$ 是转移矩阵，$\mathbf{B}$ 是测量矩阵。

(a) 证明

$$\hat{\mathbf{x}}_{n|n} = \mathbf{A}(\mathbf{I} - \mathbf{G}_n\mathbf{B})\,\hat{\mathbf{x}}_{n|n-1} + \mathbf{B}\mathbf{G}_n\mathbf{y}_n \quad \boldsymbol{\alpha}_n = \mathbf{y}_n - \mathbf{B}\,\hat{\mathbf{x}}_{n|n-1}$$

其中 $\mathbf{G}_n$ 是卡尔曼增益，$\boldsymbol{\alpha}_n$ 表示新息过程。$\mathbf{G}_n$ 是如何定义的？

(b) 使用(a)部分的结果，证明卡尔曼滤波器是一个白化滤波器，因为它产生了一个对 $\mathbf{y}_n$ 的“白的”估计误差。

14.12 表 14.2 总结了以状态的滤波估计为基础的卡尔曼滤波器。产生了另外一个卡尔曼滤波器的总结，这一次使用了状态的预测估计作为基础，描述了卡尔曼滤波器的相关的信号流图。

平方根卡尔曼滤波器

14.13 从恒等式(14.47)到式(14.49)，以及式(14.46)等号两边相应的相等的项。事实上，需要考虑四个恒等式。找出这些恒等式并且证明它是其中一个已知恒等式的移项。

扩展的卡尔曼滤波器

14.14 从式(14.64)的修正系统（状态）模型开始，证明 $\boldsymbol{\xi}_n$，它是一个已知的（如非随机的）向量，由式(14.75)定义。

14.15 令 $\mathbf{P}_{xy,n}$ 表示状态误差向量 $\mathbf{x}_n - \hat{\mathbf{x}}_{n|n-1}$ 和测量误差向量 $\mathbf{y}_n - \hat{\mathbf{y}}_{n|n-1}$ 的交叉协方差矩阵。令 $\mathbf{P}_{yy,n}$ 表示测量误差向量 $\mathbf{y}_n - \hat{\mathbf{y}}_{n|n-1}$ 的协方差矩阵。证明修正的卡尔曼收益

$$\mathbf{G}_{f,n} = \mathbf{A}_{n+1,n}^{-1}\mathbf{G}_n$$

可以用这两个协方差矩阵的形式表示为

$$\mathbf{G}_{f,n} = \mathbf{P}_{xy,n}\mathbf{P}_{yy,n}^{-1}$$

贝叶斯滤波器

14.16 (a) 证明式(14.77)

(b) 证明式(14.83)。

粒子滤波器

14.17 扩展的卡尔曼滤波器和粒子滤波器，在以下的意义上代表了非线性滤波器两个不同的例子：

- 扩展的卡尔曼滤波器的推导是基于统计分布约束条件下的一个局部方法。
- 在另一方面，粒子滤波器的推导是基于没有统计约束的一个全局方法。

阐述这两个叙述。

14.18 图 14.5 解释说明了当样本的数量和重采样都等于 6 时的重采样过程；也就是说，在重采样之后的粒子数量与取样之前的粒数量相同。解释这张图是如何得到的。

14.19 考虑一个非线性动态系统，它的状态空间模型定义如下

$$\mathbf{x}_{n+1} = \mathbf{a}_n(\mathbf{x}_n) + \boldsymbol{\omega}_n$$

和

$$\mathbf{y}_n = \mathbf{b}_n(\mathbf{x}_n) + \boldsymbol{v}_n$$

其中，动态噪声 $\boldsymbol{\omega}_n$ 和测量噪声 $\boldsymbol{v}_n$ 都是均值为零，白噪声高斯过程、协方差矩阵分别为 $\mathbf{Q}_{\omega,n}$ 和 $\mathbf{Q}_{v,n}$。决定以下的分布：

(a) 先验预测分布 $p(\mathbf{x}_n \mid \mathbf{Y}_{n-1})$。

(b) 似然分布 $p(\mathbf{y}_n \mid \mathbf{x}_n)$。

(c) 后验分布 $p(\mathbf{x}_n \mid \mathbf{Y}_n)$，其中 $\mathbf{Y}_n$ 表示观测值的序列 $\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n$。

14.20 继续 14.9 题，证明最优重要性密度分布 $p(\mathbf{x}_n \mid \mathbf{x}_{n-1}, \mathbf{y}_n)$ 是高斯分布。

计算机实验

14.21 在这个问题中，我们利用了粒子滤波器求解计算机视觉中的非线性跟踪问题。一个物体由 5×5 个像素组成，按以下两个等式定义的轨迹移动：

$$x_n = 200\left|\sin\left(\frac{2\pi n}{N}\right)\right| + 50 \qquad y_n = 100\sin\left(\frac{3.5\pi n}{N}\right) + 150$$

其中 x_n 和 y_n 是第 n 步的图像坐标，N 是帧的总数。300×300 像素的场景通过图 P14.21 可视化。这个白色的背景区域被 4 个等距的高度为 $h=10$ 像素的黑色条分割，它显示区域的前景。物体能够通过它本身的红色被分辨出来。

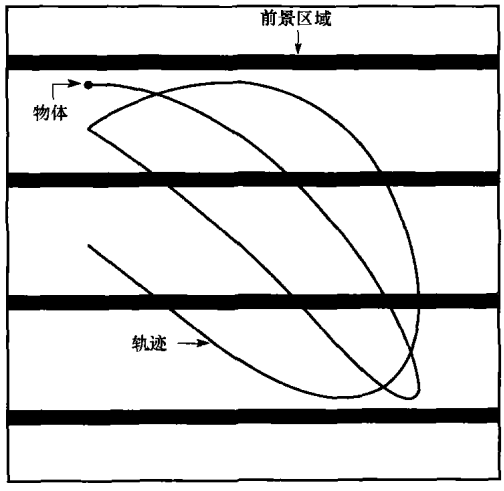


图 P14.21　14.21 题的场景和轨迹

(a) 用浅灰色表示的模拟轨迹，作为一个图像序列使用了 $N=150$ 帧。当物体移动到背景区域时确保物体被显示出来，如果物体被前景遮挡，确保它被隐藏。

(b) 将模拟数据作为输入，实现让粒子滤波器去跟踪这个物体。在物体可见的区域，你可以用颜色信息来获得一个位置的测量值，但是在物体被遮挡的区域，你就必须依靠滤波估计了。当设置状态空间模型的时候，你需要做什么样的假设呢？在场景中可视化真实的和估计的轨迹。

(c) 现在在不同的实验中，逐渐增加前景区域的高度 h。解释为了保持物体的轨迹贯穿整个图像序列所需要的权衡。帧速率和粒子数量对实验有怎样的影响？

(d) 在跟踪过程中收集的信息可以被用来估计场景的前景和背景部分，也就是说，获取物体与它所交互部分的深度。讨论解决这个问题的可能的方法。

动态驱动递归网络

本章组织

本章学习动态递归网络作为输入输出映射器的多个方面。

本章的主要内容组织如下

通过和第 13 和第 14 章的链接，15.1 节的引言给出了动态驱动递归网络学习的动机。

15.2 节讨论了不同的递归网络结构。

15.3 节和 15.4 节讨论递归网络的理论方面的内容，强调了通用逼近定理以及可控性和可观测性。

15.5 节讨论递归网络的计算能力。

15.6 节到 15.8 节介绍学习算法，15.6 节是学习算法的概述，然后介绍两个基于梯度的算法：在 15.7 节中讨论通过时间的反向传播算法，15.8 节中讨论实时递归学习算法。

15.9 节讨论消失梯度问题，它限制了基于梯度递归学习算法的实际应用能力；这里也讨论了如何使用二阶方法来缓和这一问题。

15.10 节描述通过使用序列状态估计器，解决递归神经网络的有监督训练（即估计其突触权值）。在 15.11 节中给出一个计算机实验。

15.12 节讨论自适应行为的受限制形式，这种形式仅仅在完成有监督训练并固定权值后在递归神经网络中被观测到。为增强这一自适应行为，通过包含自适应评估，使得网络的结构得到相应的扩展。

15.13 节强调了一个使用模型参考的神经控制器的实例学习。

15.1 引言

我们用下面这句话来开始本书的最后一章：

全局反馈是计算智能的促进者。

在第 13 章中通过学习作为联想记忆的递归网络已经很好地说明了这句话。在那里，我们论证了在递归网络中使用全局反馈如何完成以下一些有用任务：

- 内容可寻址的存储，以 Hopfield 网络为例。
- 自联想，以 Anderson 的盒中脑状态模型为例。
- 混沌过程的动态重构，使用围绕着正则一步预测器来建立的反馈。

在本章中，我们学习递归网络的另一个重要的应用：输入-输出映射器，它的学习自然地从第 14 章的逐次状态估计中获益。例如，考虑将具有单隐藏层的多层感知器作为递归网络的基本构建块。围绕多层感知器的全局反馈应用，可以有多种不同的形式。可以从多层感知器隐藏层的输出反馈到输入层。另外，也可以从输出层反馈到隐藏层的输入。我们甚至可以更进一步，在单一递归网络结构中，将所有这些可能的反馈结合起来。当然我们也可以考虑其他的神经网络结构作为构造递归神经网络的基本构建块。重要的是递归网络具有非常丰富的结构布局，这使得他们在计算上具有更强大的能力。

根据定义，一个映射网络的输入空间被映射到一个输出空间。对于这方面的应用，递归网络依时序响应外部应用的输入信号。因此，我们可以称这一章里的递归网络为动态驱动递归网络——本章的标题由此而来。而且，反馈的应用，使递归网络能够得到状态表示，这使得它成为适应于不同应用的工具，例如非线性预测和建模，通信信道的自适应平衡，语音处理，设备控制等。

15.2　递归网络体系结构

如前面引言所述，递归网络的结构布局有许多不同形式。本节讨论 4 种特殊结构，每一种着重于全局反馈的一种特殊形式[1]。它们有如下共同的特点：

- 它们都结合一个静态多层感知器或其中某些部分。
- 它们都利用多层感知器的非线性映射能力。

输入-输出递归模型

图 15.1 显示由一个多层感知器的自然推广而得到的通用递归网络模型。模型有一个输入被应用到有 q 个单元的抽头延迟线记忆。模型的单个输出通过另外 q 个单元抽头延迟线记忆反馈到输入。两个抽头延迟线记忆的内容被用于反馈到多层感知器的输入。模型输入的当前值用 u_n 代表，相对应的输出用 y_{n+1} 表示；也就是输出领先输入一个时间单位。因此应用到多层感知器输入层的信号向量的数据窗口由如下分量组成。

- 现在和过去的输入值，即 $u_n, u_{n-1}, \cdots, u_{n-q+1}$，表示来自网络外部的输入。
- 输出的延迟值，即 $y_n, y_{n-1}, \cdots, y_{n-q+1}$，在此基础上模型输出 y_{n+1} 进行回归。

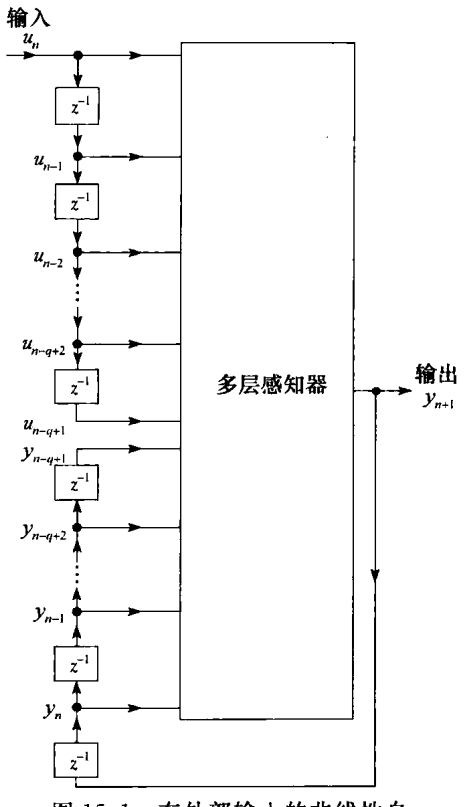


图 15.1　有外部输入的非线性自回归（NARX）模型

图 15.1 的递归网络称为*有外部输入的非线性自回归模型*（nonlinear autoregressive with exogenous inputs model，NARX）[2]。NARX 的动态行为由

$$y_{n+1} = F(y_n, \cdots, y_{n-q+1}; u_n, \cdots, u_{n-q+1}) \tag{15.1}$$

描述，其中 F 是它的自变量的一个非线性函数。注意，在图 15.1 中已经假设两个延迟线记忆有同样大小的 q；它们一般是不同的。

状态空间模型

图 15.2 表示另一种通用的递归网络的框图，称为*状态空间模型*。隐藏神经元定义网络的状态。隐藏层的输出通过一个单位时间模块反馈回输入。输入层为反馈节点和源节点的联合。网络是通过源节点和外部连接的。用于将隐藏层输出反馈回输入层的延迟单元的数目决定了模型的阶数。$m \times 1$ 维的向量 $\mathbf{u}_n$ 代表输入向量，$q \times 1$ 向量 x_n 代表隐藏层在 n 时刻的输出向量。我们可以用下列两个联立方程组描述在图 15.2 中的模型的动态行为：

$$\mathbf{x}_{n+1} = \mathbf{a}(\mathbf{x}_n, \mathbf{u}_n) \tag{15.2}$$

$$\mathbf{y}_n = \mathbf{B}\mathbf{x}_n \tag{15.3}$$

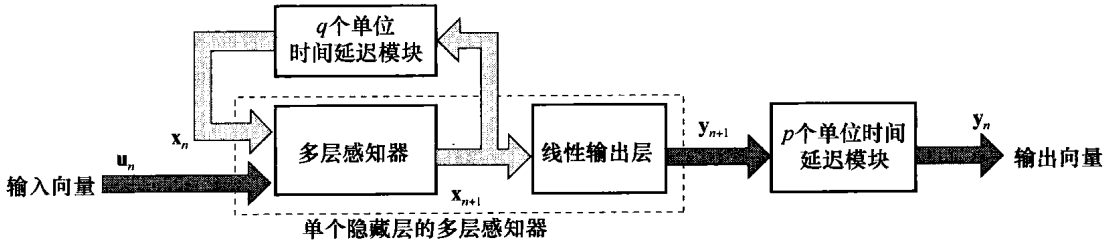


图 15.2　状态空间模型

这里 $\mathbf{a}(\cdot,\cdot)$ 是一个刻画隐藏层特征的非线性函数，$\mathbf{B}$ 是代表输出层特征的突触权值矩阵。隐藏层是非线性的，但输出层是线性的。

图 15.2 的递归网络包括几个特殊的递归结构作为其特例。例如，Elman(1990，1996) 描述过的在图 15.3 所示的简单递归网络（simple recurrent network，SRN)。Elman 网络结构和图 15.2 所示结构有相似之处，除了输出层可以是非线性的和省略了输出的单位时间延迟模块之外。在文献中它通常被称为简单递归网络，其意义是由递归网络计算的误差导数是回到过去的一个时间步的“简单”延迟；然而，这个简单性不阻止网络从很远的过去存储信息。

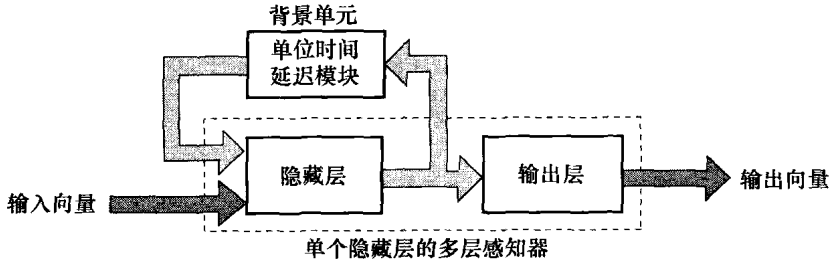


图 15.3　简单递归网络（SRN)

Elman 网络包含从隐藏层神经元到由单位时间延迟组成的背景单元（context unit）层之间的递归连接。这些背景单元存储隐藏神经元对应一个时间步的输出，接着反馈回输入层。因此隐藏神经元具有它们以前激活的记录，这使得网络可以进行通过时间扩展的学习任务。隐藏神经元也反馈给输出神经元，输出神经元给出在外部激励作用下网络的响应。由于隐藏神经元反馈的特性，这些神经元在多时间步内通过网络继续递归信息，从而发现时间的抽象表示，这就是反馈的能力。

递归多层感知器

第三种递归结构是一种*递归多层感知器*（recurrent multilayer perceptron，RMLP)(Puskorius 等，1996)。它有一个或多个隐藏层，基于同样的原因，静态多层感知器比使用单个隐藏层的感知器更有效和节约。RMLP 的每一个计算层对它的邻近层有一个反馈，如图 15.4 所示，此时 RMLP 有两个隐藏层[3]。

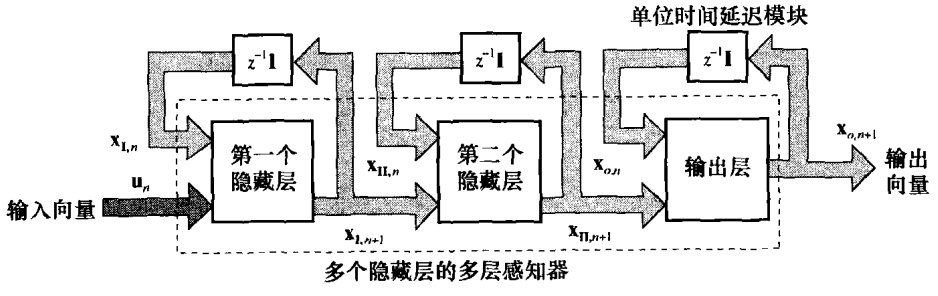


图 15.4　递归多层感知器

向量 $\mathbf{x}_{\mathrm{I},n}$ 代表第一个隐藏层的输出，$\mathbf{x}_{\mathrm{II},n}$ 代表第二个隐藏层的输出，以此类推。向量 $\mathbf{x}_{o,n}$ 代表输出层的输出。那么，RMLP 通常对输入向量 $\mathbf{u}_n$ 的响应的动态行为可用如下联立方程组描述：

$$\begin{aligned}\mathbf{x}_{\mathrm{I},n+1} &= \boldsymbol{\phi}_{\mathrm{I}}(\mathbf{x}_{\mathrm{I},n},\mathbf{u}_n)\\ \mathbf{x}_{\mathrm{II},n+1} &= \boldsymbol{\phi}_{\mathrm{II}}(\mathbf{x}_{\mathrm{II},n},\mathbf{x}_{\mathrm{I},n+1})\\ &\vdots\\ \mathbf{x}_{o,n+1} &= \boldsymbol{\phi}_{o}(\mathbf{x}_{o,n},\mathbf{x}_{K,n+1})\end{aligned} \tag{15.4}$$

其中 $\boldsymbol{\phi}_{\mathrm{I}}(\cdot,\cdot)$，$\boldsymbol{\phi}_{\mathrm{II}}(\cdot,\cdot)$，…，$\boldsymbol{\phi}_{o}(\cdot,\cdot)$ 分别表示代表 RMLP 第一个隐藏层、第二个隐藏层……以及输出层的激活函数；K 表示网络中隐藏层的数目。在图 15.4 中，$K=2$。

这里描述的 RMLP 包括图 15.3 的 Elman 网络和图 15.2 的状态空间模型，因为 RMLP 的输出层或任何隐藏层没有限定其激活函数的具体形式。

二阶网络

在描述图 15.2 的状态空间模型中，我们用“阶”来表示隐藏神经元的数目，其输出通过单位时间延迟模块反馈回输入层。

但是在另外的背景中，术语“阶”有时用来表示如何定义神经元的诱导局部域的方法。例如，一个多层感知器神经元 k 的诱导局部域 v_k 定义为

$$v_k = \sum_j w_{a,kj} x_j + \sum_i w_{b,ki} u_i \tag{15.5}$$

其中 x_j 源于隐藏层神经元 j 的反馈信号，u_i 是输入层应用于节点 i 的源信号；w 表示网络中对应的突触权值。将式(15.5)所描述的神经元称为一阶神经元。但是，有时诱导局部域 v_k 由乘法组成，表示为

$$v_k = \sum_i \sum_j w_{kij} x_i u_j \tag{15.6}$$

我们称这里的神经元为二阶神经元。二阶神经元 k 用了单一的权值 w_{kji}，它和输入节点 i，j 连接起来。

二阶神经元组成基本的二阶递归网络（Giles 等，1990），它的一个例子如图 15.5 所示。网络接受按时间顺序的输入序列，并且按如下两个式子定义的动力学演化：

$$v_{k,n} = b_k + \sum_i \sum_j w_{kij} x_{i,n} u_{j,n} \tag{15.7}$$

$$x_{k,n+1} = \varphi(v_{k,n}) = \frac{1}{1 + \exp(-v_{k,n})} \tag{15.8}$$

其中 $v_{k,n}$ 为隐藏神经元 k 的诱导局部域，b_k 为相关联的偏置，$x_{k,n}$ 为神经元 k 的状态（输出），$u_{j,n}$ 是应用于源节点 j 的输入，w_{kji} 为二阶神经元 k 的权值。

图 15.5 所示的二阶递归网络的一个特点是乘积 $x_{j,n}u_{j,n}$ 代表一对 {状态，输入}，一个正的

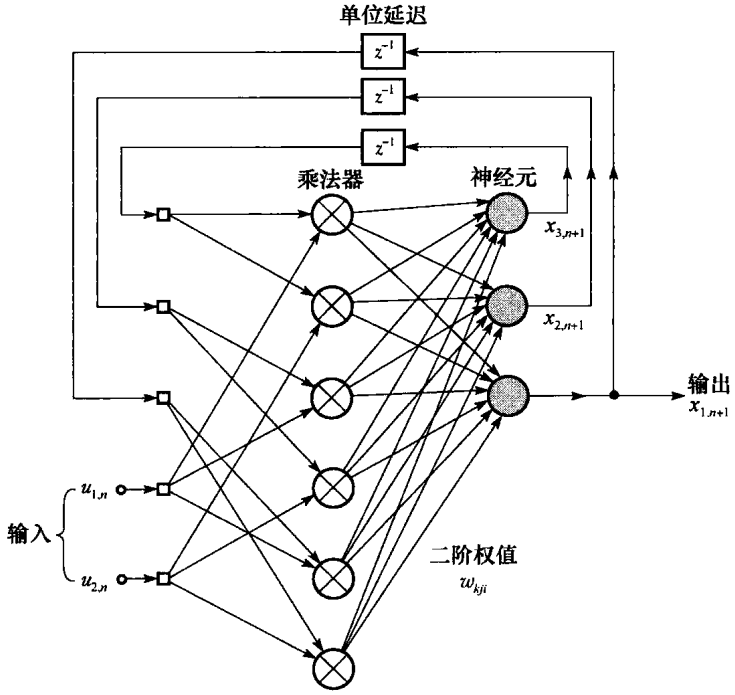


图 15.5　二阶递归网络；为简单起见省略神经元的偏置连接。网络包含 2 个输入和 3 个状态神经元，因此需要 3×2=6 个乘法器

权值 w_{kji} 表示从〈状态，输入〉到〈下一个状态〉的状态转移的出现，而权值为负表示没有转移出现。状态转移描述如下：

$$\delta(x_i, u_j) = x_k \tag{15.9}$$

根据这种关系，二阶网络可以用来表示和学习确定性有限状态自动机[4]（deterministic finite-state automated，DFA），DFA 是一个有确定状态数目的信息处理装置。在 15.5 节将介绍更多关于神经网络和自动机关系的细节。

15.3　通用逼近定理

在动态系统的数学描述上，状态的概念起着重要的作用，正如在第 14 章中解释的细节那样。动态系统的状态形式地定义为一些数量的集合，它概括为了唯一地描述系统将来行为所必需的系统过去行为的全部信息，除了用于输入（激励）产生的外部效果之外。$q \times 1$ 向量 x_n 表示非线性离散时间系统的状态。$m \times 1$ 向量 u_n 表示用于系统的输入，$p \times 1$ 向量 y_n 表示相应的输出。假设无噪声，考虑递归网络的动态行为用非线性方程组

$$\mathbf{x}_{n+1} = \boldsymbol{\phi}(\mathbf{W}_a \mathbf{x}_n + \mathbf{W}_b \mathbf{u}_n) \tag{15.10}$$

$$\mathbf{y}_n = \mathbf{W}_c \mathbf{x}_n \tag{15.11}$$

描述，其中 $\mathbf{W}_a$ 是 $q \times q$ 矩阵，$\mathbf{W}_b$ 是 $q \times m$ 矩阵，$\mathbf{W}_c$ 是 $p \times q$ 矩阵；$\boldsymbol{\phi}: \mathbb{R}^q \to \mathbb{R}^q$ 是对角映射，由

$$\boldsymbol{\phi}: \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_q \end{bmatrix} \to \begin{bmatrix} \varphi(x_1) \\ \varphi(x_2) \\ \vdots \\ \varphi(x_q) \end{bmatrix} \tag{15.12}$$

描述，表示某种无记忆的分量方式的非线性 $\varphi: \mathbb{R} \to \mathbb{R}$。空间 $\mathbb{R}^m$，$\mathbb{R}^q$ 和 $\mathbb{R}^p$ 分别称为输入空间、状态空间和输出空间。状态空间的维数（即 q）是系统的阶。因此图 15.2 的状态空间模型是 m 输入、p 输出的 q 阶递归模型。式(15.10)是模型的系统（状态）方程，式(15.11)是度量方程。系统方程(15.10)是式(15.2)的特殊形式。

建立在使用静态多层感知器和两个延迟线记忆基础上的图 15.2 的递归网络提供一种实现式(15.10)和式(15.12)非线性反馈系统的方法。注意图 15.2，在多层感知器的神经元中，只有那些通过延迟将其输出反馈到输入层的神经元与确定递归网络的状态有关。因此这就把输出层的神经元排除在状态的定义之外。

对于矩阵 $\mathbf{W}_a$，$\mathbf{W}_b$ 和 $\mathbf{W}_c$ 的解释，以及对非线性函数 $\varphi(\cdot)$，陈述如下：

- 矩阵 $\mathbf{W}_a$ 代表隐藏层的 q 个神经元连接到输入层的反馈节点的突触权值。矩阵 $\mathbf{W}_b$ 代表连接到输入层源节点的这些隐藏神经元的突触权值。为了简化式(15.10)的构成，状态模型中排除了偏置的使用。
- 矩阵 $\mathbf{W}_c$ 代表输出层中连接到隐含神经元的 p 个线性神经元的突触权值。这里再一次输出层的偏置被忽视了以简化表达。
- 非线性函数 $\varphi(\cdot)$ 代表隐藏神经元的 sigmoid 激活函数。激活函数通常具有双曲正切的形式：

$$\varphi(x) = \tanh(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}} \tag{15.13}$$

或 logistic 函数的形式：

$$\varphi(x) = \frac{1}{1 + e^{-x}} \tag{15.14}$$

式(15.10)和式(15.11)描述了状态空间模型递归网络的一个重要性质，即它是所有非线性动态系统的通用逼近器。具体可以陈述如下：

如果网络具有充分多的隐藏神经元，任意的非线性动态系统可以由递归神经网络以期望的精确度来逼近，对于状态空间的紧致性没有限制。

确实，关于通用逼近的深刻陈述是递归网络用于信号处理和控制应用的计算能力的证据。

例 1　全连接递归网络

为了表示矩阵 $\mathbf{W}_a$，$\mathbf{W}_b$ 和 $\mathbf{W}_c$ 的组成，考虑图 15.6 所示的完全连接递归网络，其中反馈路径来自隐藏神经元。在这个例子中，$m=2, q=3, p=1$。矩阵 $\mathbf{W}_a$，$\mathbf{W}_b$ 定义如下：

$$\mathbf{W}_a = \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix}$$

和

$$\mathbf{W}_b = \begin{bmatrix} b_1 & w_{14} & w_{15} \\ b_2 & w_{24} & w_{25} \\ b_3 & w_{34} & w_{35} \end{bmatrix}$$

其中矩阵 $\mathbf{W}_b$ 的第一列由 b_1, b_2, b_3 组成，分别代表神经元 1,2,3 的偏置项。矩阵 $\mathbf{W}_c$ 是一个行向量，定义为

$$\mathbf{W}_c = [1,0,0]$$

■

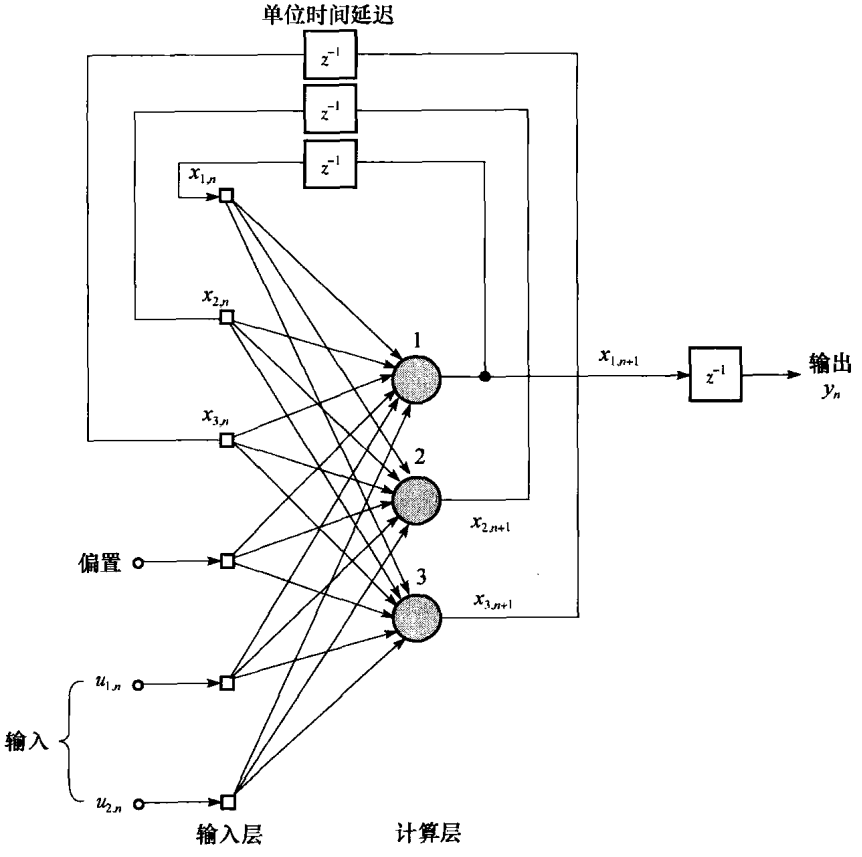


图 15.6　有两个输入、两个隐藏神经元和一个输出神经元的完全连接递归网络

15.4　可控性和可观测性

前面已提到过，许多递归网络能用图 15.2 所示的状态空间模型表示，其中状态定义为通过一系列延迟单元反馈回输入层的隐藏层输出。在此背景下，知道递归网络是否可控和可观测是很重要的。可控性是指我们能否控制递归网络的动态行为。可观测性是指我们能否观测到应用于递归网络的控制结果。

说递归网络是可控的，是指在有限时间步内，初始状态可以控制到任意想达到的状态；输出与这个定义无关。说递归网络是可观测的，是指在有限的输入/输出度量中网络的状态可以确定。在线性系统理论中对可控性和可观测性的概念有完整的论述[5]。在这里学习的递归神经网络中，我们将自己限制在可控性和可观测性的*局部*形式。局部是指将这些概念应用于网络平衡状态邻域的意义下，在第 13 章中讨论了平衡状态的细节。

如果对于输入 $\mathbf{u}$ 和一个待定义的矩阵 $\mathbf{A}_1$，它满足条件

$$\bar{\mathbf{x}} = \mathbf{A}_1\bar{\mathbf{x}} \tag{15.15}$$

就说状态 $\bar{\mathbf{x}}$ 是式(15.10)的一个*平衡状态*。

为了简化阐述，平衡状态由下面条件描述

$$\mathbf{0} = \boldsymbol{\phi}(\mathbf{0}) \quad \text{对 } \mathbf{x} = \mathbf{0},$$

换句话说，原点 $(\mathbf{0},\mathbf{0})$ 代表平衡点。

同样不失一般性，我们可以限制到一个单输入、单输出（single input, single output, SISO）系统来简化我们的论述。可以把式(15.10)和式(15.11)分别改写为

$$\mathbf{x}_{n+1} = \boldsymbol{\phi}(\mathbf{W}_a\mathbf{x}_n + \mathbf{w}_b u_n) \tag{15.16}$$

和

$$y_n = \mathbf{w}_c^T\mathbf{x}_n \tag{15.17}$$

其中 $\mathbf{w}_b$ 和 $\mathbf{W}_c$ 都是 $q\times 1$ 的列向量，u_n 是标量输入，y_n 为标量输出。由于 φ 对应于式(15.13)或式(15.14)的 sigmoid 函数是连续可微的，我们可以通过在平衡点 $\bar{\mathbf{x}}=\mathbf{0}$ 和 $\bar{u}=0$ 的附近把式(15.16)展开成 Taylor 级数而使其线性化，并保留一阶项，得到

$$\delta\mathbf{x}_{n+1} = \boldsymbol{\Phi}(\mathbf{0})\mathbf{W}_a\delta\mathbf{x}_n + \boldsymbol{\Phi}(\mathbf{0})\mathbf{w}_b\delta u_n \tag{15.18}$$

其中 $\delta\mathbf{x}_n$ 和 δu_n 是分别应用到状态和输入的小位移。$q\times q$ 矩阵 $\boldsymbol{\Phi}(\mathbf{0})$ 是 $\boldsymbol{\phi}(\mathbf{v})$ 在 $\mathbf{v}=\mathbf{0}$ 时对变量 $\mathbf{v}$ 的 Jacobi 行列式。我们可以描述线性化的系统如下：

$$\delta\mathbf{x}_{n+1} = \mathbf{A}_1\delta\mathbf{x}_n + \mathbf{a}_2\delta u_n \tag{15.19}$$

和

$$\delta y_n = \mathbf{w}_c^T\delta\mathbf{x}_n \tag{15.20}$$

其中 $q\times q$ 矩阵 $\mathbf{A}_1$ 和 $q\times 1$ 列向量 $\mathbf{a}_2$ 分别定义如下：

$$\mathbf{A}_1 = \boldsymbol{\Phi}(\mathbf{0})\mathbf{W}_a \tag{15.21}$$

和

$$\mathbf{a}_2 = \boldsymbol{\Phi}(\mathbf{0})\mathbf{w}_b \tag{15.22}$$

状态方程(15.19)和(15.20)是标准的线性形式。因此可以利用线性动态系统的可控性和可观测性的众所周知的结果，它们是数学控制论的一个标准部分。

局部可控性

从线性化的方程(15.19)，重复迭代产生下列结果：

$$\delta\mathbf{x}_{n+1} = \mathbf{A}_1\delta\mathbf{x}_n + \mathbf{a}_2\delta u_n$$

$$\delta\mathbf{x}_{n+2} = \mathbf{A}_1^2\delta\mathbf{x}_n + \mathbf{A}_1\mathbf{a}_2\delta u_n + \mathbf{a}_2\delta u_{n+1}$$

$$\vdots$$

$$\delta\mathbf{x}_{n+q} = \mathbf{A}_1^q \delta\mathbf{x}_n + \mathbf{A}_1^{q-1}\mathbf{a}_2\delta u_n + \cdots + \mathbf{A}_1\mathbf{a}_2\delta u_{n+q-2} + \mathbf{a}_2\delta u_{n+q-1}$$

其中 q 是状态空间的维数。相应地，我们可以说（Levin and Narendra，1993）：

方程(15.19)表示的线性化系统是可控的，如果矩阵

$$\mathbf{M}_c = [\mathbf{A}_1^{q-1}\mathbf{a}_2, \cdots, \mathbf{A}_1, \mathbf{a}_2, \mathbf{a}_2] \tag{15.23}$$

有秩 q，即满秩，因为这样线性化的系统（15.23）有唯一的 $\delta\mathbf{x}_{n+q}$ 的用 $u_n, u_{n+1}, \cdots, u_{n+q-1}$ 的表示，给定 $\mathbf{A}_1$，$\mathbf{a}_2$ 和 $\delta\mathbf{x}_n$。

矩阵 $\mathbf{M}_c$ 称为线性系统的可控性矩阵。

设方程(15.16)和(15.17)描述的递归网络由一系列输入 $\mathbf{u}_{q,n}$ 驱动，其定义为

$$\mathbf{u}_{q,n} = [u_n, u_{n+1}, \cdots, u_{n+q-1}]^T \tag{15.24}$$

因此可以考虑映射

$$\mathbf{G}(\mathbf{x}_n, \mathbf{u}_{q,n}) = (\mathbf{x}_n, \mathbf{x}_{n+q}) \tag{15.25}$$

其中 $\mathbf{G}: \mathbb{R}^{2q} \to \mathbb{R}^{2q}$。在习题 15.4 中证明：

- 状态 $\mathbf{x}_{n+q}$ 是其过去值 $\mathbf{x}_n$ 和输入 $u_n, u_{n+1}, \cdots, u_{n+q-1}$ 的嵌套非线性函数。
- $\mathbf{x}_{n+q}$ 关于 $\mathbf{u}_{qn}$ 的 Jacobi 矩阵在原点的值等于式(15.23)的可控性矩阵 $\mathbf{M}_c$。

我们可以把映射 G 关于 $\mathbf{u}_{q,n}$ 和 $\mathbf{x}_n$ 的 Jacobi 矩阵在原点（0，0）的值表示为

$$\mathbf{J}_{(0,0)}^{(c)} = \begin{bmatrix} \left(\frac{\delta\mathbf{x}_n}{\delta\mathbf{x}_n}\right)_{(0,0)} & \left(\frac{\delta\mathbf{x}_{n+q}}{\delta\mathbf{x}_n}\right)_{(0,0)} \\ \left(\frac{\delta\mathbf{x}_n}{\delta\mathbf{u}_{q,n}}\right)_{(0,0)} & \left(\frac{\delta\mathbf{x}_{n+q}}{\delta\mathbf{u}_{q,n}}\right)_{(0,0)} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{X} \\ \mathbf{0} & \mathbf{M}_c \end{bmatrix} \tag{15.26}$$

其中 $\mathbf{I}$ 是单位矩阵，$\mathbf{0}$ 是零矩阵，项 $\mathbf{X}$ 是不感兴趣的部分。因为它的特殊形式，$\mathbf{J}_{(0,0)}^{(c)}$ 的行列式等于单位矩阵 $\mathbf{I}$ 的行列式（等于 1）和可控性矩阵 $\mathbf{M}_c$ 的行列式乘积。如果 $\mathbf{M}_c$ 是满秩矩阵，那么 $\mathbf{J}_{(0,0)}^{(c)}$ 也是满秩的。

为了继续处理，我们需要引用反函数定理，它可以陈述如下（Vidyasagar，1993）：

考虑映射 $\mathbf{f}: \mathbb{R}^q \to \mathbb{R}^q$，假设映射 $\mathbf{f}$ 的每一个分量对于它的变量在平衡点 $\mathbf{x}_0 \in \mathbb{R}^q$ 都是可微的，并令 $\mathbf{y}_0 = \mathbf{f}(\mathbf{x}_0)$。那么存在开集 $\mathcal{U} \subseteq \mathbb{R}^q$ 包含 $\mathbf{x}_0$ 及 $\mathcal{V} \subseteq \mathbb{R}^q$ 包含 $\mathbf{y}_0$，使得 $\mathbf{f}$ 为 $\mathcal{U}$ 到 $\mathcal{V}$ 上的微分同胚。如果 $\mathbf{f}$ 还是光滑的，那么逆映射 $\mathbf{f}^{-1}: \mathbb{R}^q \to \mathbb{R}^q$ 也是光滑的，即 $\mathbf{f}$ 是光滑微分同胚。

映射 $\mathbf{f}: \mathcal{U} \to \mathcal{V}$ 如果满足下列 3 个条件（参见第 7 章），则说它是 $\mathcal{U}$ 到 $\mathcal{V}$ 上的微分同胚：

1. $\mathbf{f}(\mathcal{U}) = \mathcal{V}$。
2. 映射 $\mathbf{f}: \mathcal{U} \to \mathcal{V}$ 是一对一的（即可逆的）。
3. 逆映射 $\mathbf{f}^{-1}: \mathcal{V} \to \mathcal{U}$ 的每个分量关于它的变量是连续可微的。

回到可控性的问题，我们将对式(15.25)定义的映射验证满足反函数定理中的 $\mathbf{f}(\mathcal{U}) = \mathcal{V}$ 条件。应用反函数定理，如果可控性矩阵 $\mathbf{M}_c$ 的秩为 q，可以说局部存在一个反映射，定义为

$$(\mathbf{x}_n, \mathbf{x}_{n+q}) = \mathbf{G}^{-1}(\mathbf{x}_n, \mathbf{u}_{q,n}) \tag{15.27}$$

式(15.27)实际上指出存在一个输入序列能局部驱动网络在 q 个时间步中从状态 $\mathbf{x}_n$ 到 $\mathbf{x}_{n+q}$。所以，我们可以正式陈述局部可控性定理如下（Levin and Narendra，1993）：

假定递归网络由式(15.16)和式(15.17)定义，它在原点（即平衡点）附近的线性化方程由式(15.19)和式(15.20)定义。如果线性化系统是可控的，则递归网络在原点附近是局部可控的。

局部可观测性

重复使用线性化的式(15.19)和式(15.20)，可得

$$\delta y_n = \mathbf{w}_c^T \delta \mathbf{x}_n$$

$$\delta y_{n+1} = \mathbf{w}_c^T \delta \mathbf{x}_{n+1} = \mathbf{w}_c^T \mathbf{A}_1 \delta \mathbf{x}_n + \mathbf{w}_c^T \mathbf{a}_2 \delta u_n$$

$$\vdots$$

$$\delta y_{n+q-1} = \mathbf{w}_c^T \mathbf{A}_1^{q-1} \delta \mathbf{x}_n + \mathbf{w}_c^T \mathbf{A}_1^{q-2} \mathbf{a}_2 \delta u_n + \cdots + \mathbf{w}_c^T \mathbf{A}_1 \mathbf{a}_2 \delta u_{n+q-3} + \mathbf{w}_c^T \mathbf{a}_2 \delta u_{n+q-2}$$

其中 q 是状态空间的维数。所以，我们可以陈述（Levin and Narendra，1993)：

式(15.19)和式(15.20)描述的线性化系统是可观测的，如果矩阵

$$\mathbf{M}_o = [\mathbf{w}_c, \mathbf{w}_c \mathbf{A}_1^T, \cdots, \mathbf{w}_c (\mathbf{A}_1^T)^{q-1}] \tag{15.28}$$

的秩为 q，即满秩。

矩阵 $\mathbf{M}_o$ 称为线性系统的可观测性矩阵。

令用于驱动由式(15.19)和式(15.20)描述的递归网络的一系列输入定义如下：

$$\mathbf{u}_{q-1,n} = [u_n, u_{n+1}, \cdots, u_{n+q-2}]^T \tag{15.29}$$

相应地，令

$$\mathbf{y}_{q,n} = [y_n, y_{n+1}, \cdots, y_{n+q-1}]^T \tag{15.30}$$

代表由初始状态 $\mathbf{x}_n$ 和输入序列 $\mathbf{u}_{q-1,n}$ 产生的输出向量。那么我们可以考虑映射

$$\mathbf{H}(\mathbf{u}_{q-1,n}, \mathbf{x}_n) = (\mathbf{u}_{q-1,n}, \mathbf{y}_{q,n}) \tag{15.31}$$

其中 $\mathbf{H}: \mathbb{R}^{2q-1} \to \mathbb{R}^{2q-1}$。在习题 15.5 中证明 $\mathbf{y}_{q,n}$ 对 $\mathbf{x}_n$ 的 Jacobi 矩阵在原点的值等于式(15.28)的可观测矩阵 $\mathbf{M}_o$。因此 H 关于 $\mathbf{u}_{q-1,n}$ 和 $\mathbf{x}_n$ 的 Jacobi 矩阵在原点 (0,0) 的值可表示为

$$\mathbf{J}_{(0,0)}^{(o)} = \begin{bmatrix} \left(\dfrac{\partial \mathbf{u}_{q-1,n}}{\partial \mathbf{u}_{q-1,n}}\right)_{(0,0)} & \left(\dfrac{\partial \mathbf{y}_{q,n}}{\partial \mathbf{u}_{q-1,n}}\right)_{(0,0)} \\ \left(\dfrac{\partial \mathbf{u}_{q-1,n}}{\delta \mathbf{x}_n}\right)_{(0,0)} & \left(\dfrac{\partial \mathbf{y}_{q,n}}{\delta \mathbf{x}_n}\right)_{(0,0)} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{X} \\ \mathbf{0} & \mathbf{M}_o \end{bmatrix} \tag{15.32}$$

其中 X 同样为不感兴趣的部分。$\mathbf{J}_{(0,0)}^{(o)}$ 的行列式等于单位矩阵 $\mathbf{I}$ 的行列式（等于 1）和矩阵 $\mathbf{M}_o$ 的行列式的乘积。如果 $\mathbf{M}_o$ 是满秩，那么 $\mathbf{J}_{(0,0)}^{(o)}$ 也是。引用反函数定理，可以说如果线性化系统的可观测性矩阵 $\mathbf{M}_o$ 是满秩的，则存在一个逆映射，定义为

$$(\mathbf{u}_{q-1,n}, \mathbf{x}_n) = \mathbf{H}^{-1}(\mathbf{u}_{q-1,n}, \mathbf{y}_{q,n}) \tag{15.33}$$

实际上，这个等式表明在原点的局部邻域，$\mathbf{x}_n$ 是 $\mathbf{u}_{q-1,n}$ 和 $\mathbf{y}_{q,n}$ 的非线性函数，非线性函数是递归网络的观测器。因此局部可观测性定理可正式地陈述如下（Levin and Narendra，1993)：

由式(15.16)和式(15.17)所定义的递归网络，令它在原点（即平衡点）附近线性化的形式由式(15.19)和式(15.20)所定义。如果线性系统是可观测的，则递归网络在原点附近是可观测的。

例 2　简单状态空间模型的可控制性和可观测性

考虑具有矩阵 $\mathbf{A}_1 = a\mathbf{I}$ 的状态空间模型，这里 a 是标量，$\mathbf{I}$ 是单位矩阵。式(15.23)的可控性矩阵 $\mathbf{M}_c$ 简化为

$$\mathbf{M}_c = a[\mathbf{a}_2, \cdots, \mathbf{a}_2, \mathbf{a}_2]$$

矩阵的秩是 1。因此，具有矩阵 $\mathbf{A}_1$ 的值的线性化系统是不可控的。

在式(15.28)中置 $\mathbf{A}_1 = a\mathbf{I}$，得到可观测性矩阵

$$\mathbf{M}_o = a[\mathbf{w}_c, \mathbf{w}_c, \cdots, \mathbf{w}_c]$$

它的秩也为 1。这个线性系统也是不可观测的。 ■

15.5　递归网络的计算能力

如图 15.2 所示的状态空间模型和图 15.1 的 NARX 模型，递归网络具有模拟有限状态自动机的固有能力。自动机表示像计算机一样的信息处理设备的抽象。实际上自动机和神经网络有久远的渊源[6]。Minsky 在他 1967 年的书《Computation：Finite and Infinite Machines》中第 55 页有如下重要的说明：

"每一有限状态机等价于某神经网络，并且可以由它模拟。也就是说，给定一有限状态机 $\mathcal{M}$，可以建立一个神经网络 $\mathcal{N}^{\mathcal{M}}$，若将它看做一个黑箱机器，则其行为酷似 $\mathcal{M}$!"

递归网络的早期工作用硬的阈值逻辑作为神经元的激活函数而不用软的 sigmoid 函数。

也许是 Cleeremans(1989) 第一个报道了展示递归网络能否学会由小型有限状态语法所包含的例外（偶发性）的试验。特别，由语法导出的字符串赋给简单递归网络（见图 15.3），需要它在每一步预测下一字母。预测是上下文相关的，因为每一个在语法中出现两次的字母每次它的后继字母都不同。这表明网络能够在隐藏神经元中发展对应自动机（有限状态机）状态的内部表示。在 Kremer(1995) 中给出正式的证明，表明简单递归网络有和任何有限状态机一样的计算能力。

在一般意义下，递归网络的计算能力体现在两个主要定理。

定理 Ⅰ（Siegelmann and Sontag，1991）

所有图灵机都可由建立在用 sigmoid 激活函数的神经元上的完全连接递归网络模拟。

图灵机是 Turing(1936) 发明的抽象计算工具；其数学模型比有限状态自动机更一般。因此，通过递归网络来模拟图灵机是更具挑战性的命题。它由图 15.7 所示的三个功能块构成：

1. **控制单元**，假设任何可能的有限状态之一。

2. **线性带**，假设在两个方向上是无限的，被划分成分离的方块，每个方块都可以存储一个单一的符号，这些符号是从一个有限的符号集合中取出的。

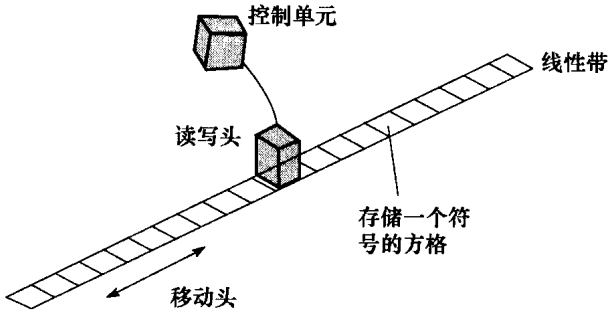


图 15.7　图灵机

3. **读写头**，沿着线性带移动，并从控制单元得到信息和把信息传送到控制单元。

函数 $f(x)$ 称为是可计算的，如果存在图灵机，在给定自变量 x 表示的带时，最终在带表示值 $f(x)$ 时停止。然而，这一思想是有问题的，因为计算思想缺少正式定义。无论如何，Church-Turing 理论认为图灵机能够计算任意可计算函数，这一理论被作为充分条件而广为接受（Russell and Norvig，1995）。

定理 Ⅱ（siegelmann 等，1997）

对于 NARX 网络，若具有一隐藏层单元，其激活函数为有界和单侧饱和的并且有一个线性输出神经元，那么不计线性延迟（linear slowdown），它可以模拟用完全连接的具有有界且单侧饱和的激活函数的递归网络。

线性延迟是指如果一个完全连接的有 N 个神经元的递归网络在时间 T 内计算一个我们感

兴趣的任务，那么等价的 NARX 网络所占用的总时间是 $(N+1)T$。函数 $\varphi(\cdot)$ 如果满足下列条件则说它是有界且单边饱和的（bounded，one-sided saturated，BOSS）函数：

1. **函数 $\varphi(\cdot)$ 值域有界**；即 $a \leqslant \varphi(x) \leqslant b$，$a \neq b$，对于所有 $x \in \mathbb{R}$。
2. **函数 $\varphi(\cdot)$ 是左饱和的**；即存在值 s 和 S，对于所有的 $x \leqslant s$，有 $\varphi(x) = S$。
3. **函数 $\varphi(\cdot)$ 是非常数的**；即存在不相同的两个数 x_1 和 x_2，满足 $\varphi(x_1) \neq \varphi(x_2)$。

阈值（Heaviside）和分段线性函数满足 BOSS 条件。但是在严格意义上 sigmoid 函数不是一个 BOSS 函数，因为它不满足条件 2。但是做一个小的修改，它可以满足 BOSS 条件，即写成（在 logistic 函数的情况下）

$$\varphi(x) = \begin{cases} \dfrac{1}{1+\exp(-x)}, & \text{对于 } x > s \\ 0, & \text{对于 } x \leqslant s \end{cases}$$

其中 $x \in \mathbb{R}$。实际上，在 $x \leqslant s$ 时 logistic 函数是截断的。

作为定理 I 和定理 II 的推论，我们可以得到（Giles，1996）：

有一个隐藏层神经元且激活函数为 BOSS 函数及一个线性输出神经元的 NARX 网络是图灵等价的。

图 15.8 给出定理 I 和定理 II 及这个推论的图解。但是，必须注意当网络体系结构受到限制时，递归网络的计算能力就不再成立，如同在 Sperduti(1997) 描述的一样。在注释 7 中给出受限制的网络体系结构的参考文献。

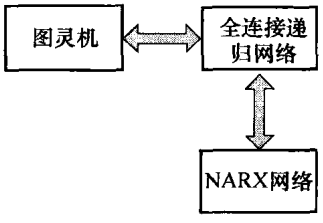


图 15.8　定理 I 和定理 II 及它们的推论的图解

15.6　学习算法

现在来研究递归网络的训练问题。第 4 章讨论过普通（静态）多层感知器的两种方式：批量方式和随机（串行）方式。在批量方式中，网络的敏感度是在调整网络的自由参数前针对整个训练集计算的。在随机方式中，参数的调整是在给出训练集的每一个模式的表示之后进行的。同样，有两个训练递归网络的方式如下（Williams and Zipser，1995）：

1. *分回合（epochwise）的训练*。在给定的回合，递归网络利用输入-目标响应对的时间序列从初始状态出发到达一个新的状态后停止，此时训练亦停止；然后对于下一个回合又重新设置一个新的初始状态。初始状态在每个训练时期并不总是一样的。重要的是对于新的回合的初始状态和网络在此前一个回合到达的状态不一样。例如，考虑用递归网络模拟有限状态机的运行。在这种条件下，有理由使用分回合的训练，因为我们有很大的可能性用递归网络去模拟机器中大量的不同的初始状态和不同的最终状态的集合。在递归网络的分回合训练中，“回合”与一般普通多层感知器中使用的意义不同。尽管在多层感知器的训练的一个回合包含整个输入-目标响应对的训练样本，递归网络训练的回合包含时间串行输入-目标响应对的一个训练模式。

2. *连续训练*。训练的第二种方法适合于没有可用的重置状态或需要在线学习的情况。连续训练的显著特征是网络学习和被网络处理的信号处理同时进行。简单地说，学习过程永不停止。例如，考虑让递归网络去对一个非稳态过程如语音信号建模。在这种情况下，网络的连续运行不能提供方便的时刻以决定何时停止训练而重新开始用网络不同自由参数的值。

记住这两种训练的方式，在下面的两节中我们将描述递归网络的不同的学习算法，可概述如下：

- 在 15.7 节讨论的通过时间的反向传播（back-propagation-through-time，BPTT）算法是在这样的前提下提出的，即递归网络的时序操作可以展开为一个多层感知器。这就为标准反向传播算法的应用铺平了道路。通过时间的反向传播算法可以用分回合的方式、连续方式或两种方式的组合来实现。
- 在 15.8 节讨论的实时递归学习（RTRL）算法是从式(15.10)和式(15.11)描述的状态空间模型导出的。

基本上，BPTT 和 RTRL 包含了导数的传播，一个是反向的另一个是前向的。它们能用于任何需要利用导数的训练过程。BPTT 比 RTRL 需要更少的计算量，但随着串行输入-目标响应对序列长度的增加，BPTT 需要的存储空间也快速增加。一般而言，我们因此说 BPTT 处理离线训练更好，而 RTRL 更适合于在线连续训练。

两种算法有很多共同点。第一，它们都是基于梯度下降的方法，因此代价函数的瞬时值（基于平方误差准则）对网络的突触权值被最小化。第二，它们实现都很简单，但可能收敛很慢。第三，它们是相关的，因为通过时间的反向传播算法的信号流图的表示，能够由实时递归学习算法的确定形式的信号流图的表示经转置而得到（Lefebvre，1991；Beaufays and Wan，1994）。

一些启发

在开始这两种学习算法的描述之前，我们罗列一些对于改进递归网络训练的启发，这些训练涉及梯度下降方法的使用（Giles，1996）：

- 训练样本应该按照字典顺序排序，最短的符号字符串首先提交给网络。
- 训练应该开始于一个小的训练样本集，然后随着训练进行逐步增加样本。
- 只有当正在被网络处理的训练样本的绝对误差大于某一指定的标准时才应该更新网络的突触权值。
- 在训练过程中建议使用权值衰减；权值衰减可作为复杂性正则化（第 4 章讨论过）的一个粗略的形式。

第一个启发有特别重要的意义。如果可以实现的话，它提供减轻在采用梯度下降方法训练递归网络时出现的消失梯度问题。这个问题的细节在 15.9 节讨论。

15.7　通过时间的反向传播

用于训练一个递归网络的*通过时间的反向传播*（BPTT）算法是标准反向传播算法的扩展[8]。它可以通过将网络的时序操作展开成一个分层的前馈网络导出，它的拓扑结构在每个时间步增加一层。

具体地，让$\mathcal{N}$表示需要学习时序任务的递归网络，从时间 n_0 开始一直到时间 n。$\mathcal{N}^*$ 表示对递归网络$\mathcal{N}$的时序操作进行展开所得的前馈网络。展开后的网络$\mathcal{N}^*$ 和初始网络$\mathcal{N}$的关系如下：

1. 对区间 $(n_0, n]$ 内的每一个时间步，网络$\mathcal{N}^*$有一个包含 K 个神经元的层，K 是包含在网络$\mathcal{N}$中的神经元的数量。

2. 在网络$\mathcal{N}^*$ 的每一层有网络$\mathcal{N}$的每一个神经元的拷贝。

3. 对每一个时间步 $l \in [n_0, n]$，从网络$\mathcal{N}^*$ 中 l 层的神经元 i 到 $l+1$ 层的神经元 j 的突触连接，是在网络$\mathcal{N}$中从神经元 i 到神经元 j 的突触连接的拷贝。

这些要点在下面的例子中解释。

例 3　两神经元递归网络的展开

考虑图 15.9a 所示的两个神经元递归网络$\mathcal{N}$。为简化表示，省略单位延迟操作符 z^{-1}。这

个操作符应该插入到图 15.9a 所示突触连接（包括自连接环）的每一步。通过一步一步地展开网络的时序操作，得到图 15.9b 的信号流图，其中起始时间 $n_0=0$。图 15.9b 代表分层的前馈网络$\mathcal{N}^*$，其中在每一步时序操作都有新的层加入。 ■

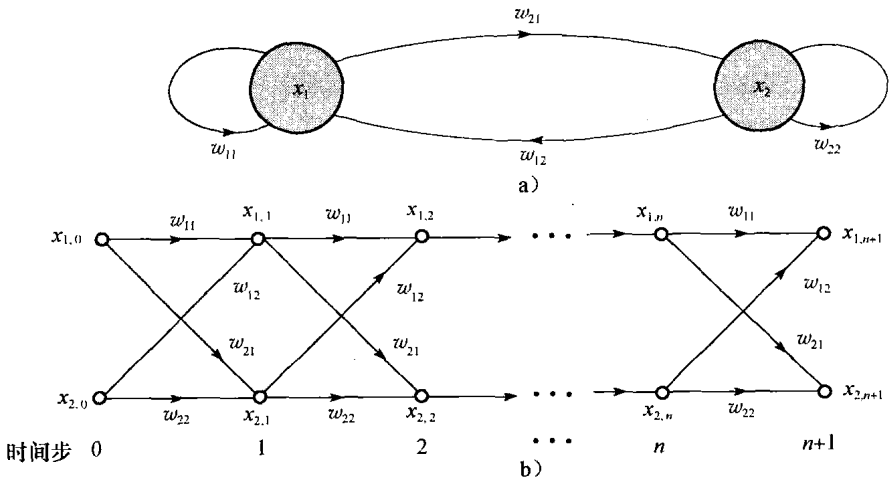


图 15.9　a）两个神经元递归网络$\mathcal{N}$的结构图；b）网络$\mathcal{N}$依时间展开的信号流图

依赖于使用分回合训练或使用连续（实时）训练，展开过程的应用导致通过时间的反向传播两个根本不同的实现。下面依次描述这两种递归学习方法。

分回合的通过时间的反向传播

将用于递归网络训练的数据集分割为独立的回合，每一回合表示一个感兴趣的时序模式。令 n_0 表示一个回合的开始时间，n_1 表示其结束时间。在这个回合里，可以定义代价函数

$$\mathscr{E}_{\text{total}}=\frac{1}{2}\sum_{n=n_0}^{n_1}\sum_{j\in\mathscr{A}}e_{j,n}^2 \tag{15.34}$$

其中$\mathscr{A}$为网络中指定期望响应的那些神经元标号 j 的集合，$e_{j,n}$是该神经元关于期望响应和计算出的实际输出之间的误差信号。我们希望计算网络的敏感度，即计算代价函数对网络突触权值的偏导数。为此，可以使用*通过时间的反向传播*（back-propagation-through-time，BPTT）算法，这个算法建立在第 4 章讨论的标准反向传播学习批量方式的基础上。分回合的 BPTT 算法进行如下（Williams and Peng，1990）：

- 首先，对时间区间 (n_0,n_1) 执行单纯的数据前向传播通过网络。保存完整的输入数据记录、网络状态（即网络的突触权值）以及期望响应。
- 对过去这条记录执行一个单纯的反向传播通过网络，计算局部梯度

$$\delta_{j,n}=-\frac{\partial\mathscr{E}_{\text{total}}}{\partial v_{j,n}} \tag{15.35}$$

的值，对于所有的 $j\in\mathscr{A}$，$n_0<n\leqslant n_1$。这个计算用公式

$$\delta_{j,n}=\begin{cases}\varphi'(v_{j,n})e_{j,n} & ,\text{对于 } n=n_1\\ \varphi'(v_{j,n})\left[e_{j,n}+\sum_{k\in\mathscr{A}}w_{jk}\delta_{k,n+1}\right] & ,\text{对于 } n_0<n<n_1\end{cases} \tag{15.36}$$

进行，其中 $\varphi'(\cdot)$ 是激活函数对其自变量的导数，$v_{j,n}$是神经元 j 的诱导局部域。这里假设网络的所有神经元有同样的激活函数 $\varphi(\cdot)$。重复使用式(15.36)，从时刻 n_1 出发，向后一步一步进行直到时刻 n_0；此处涉及的步数与包含在这个回合内的步数相同。

• 一旦执行反向传播的计算回到 n_0+1 时，对神经元 j 的突触权值 w_{ji} 调整如下：

$$\Delta w_{ji} = -\eta \frac{\partial \mathscr{E}_{\text{total}}}{\partial w_{ji}} = \eta \sum_{n=n_0+1}^{n_1} \delta_{j,n} x_{i,n-1} \tag{15.37}$$

其中 η 是学习率参数，$x_{i,n-1}$ 是在时刻 $n-1$ 时作用于神经元 j 的第 i 个突触的输入。

比较刚才描述的分回合的 BPTT 的过程和标准反向传播学习的集中方式，可以看出它们根本的差别是前者在网络的许多层里指定对神经元的期望响应，因为实际输出层在网络的时序行为展开时被重复很多次。

截断的通过时间的反向传播

为了使用通过时间的反向传播的实时形式，我们用误差平方和的瞬时值，即

$$\mathscr{E}_n = \frac{1}{2} \sum_{j \in \mathscr{A}} e_{j,n}^2$$

作为需要最小化的代价函数。如同标准反向传播学习的串行（随机）模式一样，我们使用代价函数 $\mathscr{E}_n$ 的负梯度去计算对于每个时刻 n 网络突触权值的适当调整量。当网络运行时，调整建立在连续的基础上。但是为了采用计算可行的方式，我们只在一个固定数目的时间步内储存相关的输入数据和网络状态的历史记录，该时间步数目称为*截断深度*（truncation depth）。此后截断深度用 h 表示。任何比 h 时间步早的信息是无关的，因此可以省略。如果不截断计算，由此容许回到开始时间，计算时间和储存要求当网络运行时会随时间线性增长，最终达到某点使得整个学习过程成为不可行的。

算法的第二种形式称为*截断的通过时间的反向传播*（truncated back-propagation-through-time，BPTT(h)）算法（Williams and Peng，1990）。神经元 j 的局部梯度定义为

$$\delta_{j,l} = -\frac{\partial \mathscr{E}_l}{\partial v_{j,l}}, \quad \begin{cases} \text{对于 } j \in \mathscr{A} \\ \text{并且 } n-h < l \leqslant n \end{cases} \tag{15.38}$$

由此导出公式

$$\delta_{j,l} = \begin{cases} \varphi'(v_{j,l}) e_{j,l} & \text{对于 } l = n \\ \varphi'(v_{j,l}) \sum_{k \in \mathscr{A}} w_{jk,l} \delta_{k,l+1} & \text{对于 } n-h < l < n \end{cases} \tag{15.39}$$

一旦执行反向传播的计算到达时刻 $n-h+1$ 时，对神经元 j 的突触权值 w_{ji} 进行如下调整：

$$\Delta w_{ji,n} = \eta \sum_{j=n-h+1}^{n} \delta_{j,l} x_{i,l-1} \tag{15.40}$$

其中 η 和 $x_{i,l-1}$ 如前定义。注意式(15.39)中 $w_{jk,l}$ 的使用需要保留权值的历史记录。只有当学习率参数 η 小到足以确保权值从一个时间步到下一时间步不会有很大改变的时候，在等式中使用 $w_{jk,l}$ 才是合理的。

比较式(15.39)和式(15.36)，可以看出与分回合的 BPTT 算法不同，误差信号只有在当前时间 n 才会进入计算。这就解释为什么不保存过去期望响应记录的原因。实际上，截断的通过时间的反向传播算法对前期时间步的处理，和随机反向传播算法（在第 4 章讨论）对待多层感知器中的隐藏神经元的计算是一样的。

一些实际考虑

在 BPTT(h) 的实际应用中，截断并不像看起来那样是完全人为的。除非递归网络是不稳定的，对于导数 $\partial \mathscr{E}_l / \partial v_{j,l}$ 应该收敛，这是因为时间上非常靠后的计算对应于更高的反馈能力（粗略地等于 sigmoid 斜率乘以权值）进行的。在任何情况下，截断深度 h 应该大到足以产生接近实际值的导数。这就要求值 h 有一个低的下界。例如，把动态驱动递归网络用于引擎慢速

(idle-speed) 控制时，$h=30$ 是一个完成学习任务的相当保守的选择 (Puskorius 等，1996)。

有序导数方法

另一实际问题需要讨论。本节讨论的通过时间的反向传播的展开过程提供一个利用相似层随时间前向处理的级联描绘的有用工具，这样可以帮助我们深入理解过程是如何作用的。然而这个优点也是产生缺点的原因。在由很少神经元组成的相对简单的递归网络中过程运行良好。但是，当展开过程应用到实际中常遇到的更一般的结构时，基本公式，特别是式(15.39)，就变得笨拙。在这种情况下，更好的方法是用 Werbos(1990) 描述的更一般的方法，此时每层的前向传播每一个表示引发一个相应的反向传播表示的集合。这个方法的优点是对前向和递归(反馈) 连接的相似处理。

为描述 BPTT(h) 特殊形式的机理，令 $\mathbf{F}^l_{-x}$ 表示在节点 l 的网络输出对 x 的有序导数 (ordered derivative)。为了导出反向传播方程，以相反的次序考虑前向传播方程。从每个方程根据下列原理推导一个或多个反向传播表达式：

$$\text{如果 } a=\varphi(b,c),\quad \text{那么 } \mathbf{F}^l_{-b}=\frac{\partial\varphi}{\partial b}\mathbf{F}^l_{-a}\quad \text{和}\quad \mathbf{F}^l_{-c}=\frac{\partial\varphi}{\partial c}\mathbf{F}^l_{-a} \tag{15.41}$$

例 4　式(15.41)的说明

为了让有序导数的概念清晰，考虑下列两个方程的非线性系统：

$$x_1=\log u+x_2^3$$
$$y=x_1^2+3x_2$$

变量 x_2 在两个方面影响输出 y：直接通过第二个方程，和间接通过第一个方程。y 对 x_2 的有序导数由包括 x_2 对 y 的直接和间接的作用效果的总因果影响所定义，可表示如下：

$$\mathbf{F}_{-x_2}=\frac{\partial y}{\partial x_2}+\frac{\partial y}{\partial x_1}\times\frac{\partial x_1}{\partial x_2}=3+(2x_1)(3x_2^2)=3+6x_1x_2^2$$ ■

有序导数方法的其他期望特征

在编写程序时，对 BPTT(h) 的有序导数，式(15.41)右侧的每一个有序导数值被加到左侧的原来的值上。在这种方法中，适当的导数从网络中的一个给定的节点分配到所有以前向方式前馈该节点的网络其他节点和突触权值，并且对于每一连接中可能出现的延迟做出适当补偿。

式(15.41)描述的简洁有序导数表达式减少了对诸如时间展开或信号流图的可视化的需要。在 Feldkamp and Puskorius(1998) 以及 Puskorius 等 (1996) 中，利用这个过程产生了实现 BPTT(h) 算法的伪代码。

15.8　实时递归学习

本节我们描述另一种称为*实时递归学习* (real-time recurrent learning，RTRL)[9] 的学习算法，在第 15.6 节简单描述过。算法的名称来自于下面的事实，完全连接网络的突触权值调整是实时的，也就是说，是在网络继续执行它的信号处理功能的时候 (Williams and Zipser，1989)。图 15.10 显示一个递归网络结构布局。它由 q 个神经元和 m 个外部输入组成。网络有两个不同的层：*并置的输入-反馈层和计算节点的处理层*。相应地，网络突触连接也是由前馈和反馈连接构成的。

网络状态空间的描述由式(15.10)和式(15.11)定义。系统方程(15.10)重写成以下扩展形式：

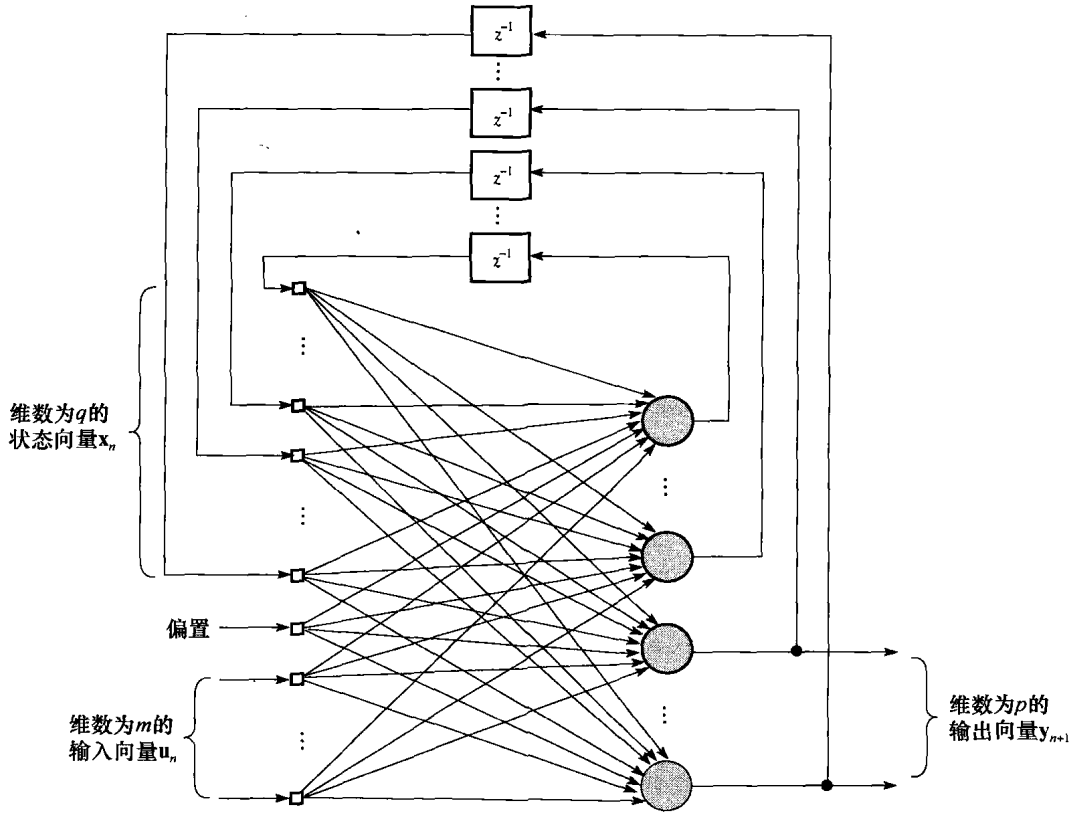


图 15.10　用于描述 RTRL 算法的完全连接递归网络

$$\mathbf{x}_{n+1} = \begin{bmatrix} \varphi(\mathbf{w}_1^T \boldsymbol{\xi}_n) \\ \vdots \\ \varphi(\mathbf{w}_j^T \boldsymbol{\xi}_n) \\ \vdots \\ \varphi(\mathbf{w}_q^T \boldsymbol{\xi}_n) \end{bmatrix} \tag{15.42}$$

其中假设所有的神经元有相同的激活函数 $\varphi(\cdot)$。$(q+m+1)\times 1$ 向量 $\mathbf{w}_j$ 是递归网络的神经元 j 的突触权值向量，即

$$\mathbf{w}_j = \begin{bmatrix} \mathbf{w}_{a,j} \\ \mathbf{w}_{b,j} \end{bmatrix}, \quad j = 1,2,\cdots,q \tag{15.43}$$

其中 $\mathbf{w}_{a,j}$ 和 $\mathbf{w}_{b,j}$ 分别是转置矩阵 $\mathbf{W}_a^T$ 和 $\mathbf{W}_b^T$ 的第 j 列。$(q+m+1)\times 1$ 向量 $\boldsymbol{\xi}_n$ 定义为

$$\boldsymbol{\xi}_n = \begin{bmatrix} \mathbf{x}_n \\ \mathbf{u}_n \end{bmatrix} \tag{15.44}$$

其中 $\mathbf{x}_n$ 是 $q\times 1$ 状态向量，$\mathbf{u}_n$ 是 $(m+1)\times 1$ 输入向量。$\mathbf{u}_n$ 的第一个元素是 $+1$，对应的 $\mathbf{w}_{b,j}$ 的第一个元素等于应用于神经元 j 的偏置 b_j。

为表达简单起见，引入新的矩阵 $\boldsymbol{\Lambda}_{j,n}$，$\mathbf{U}_{j,n}$ 和 $\boldsymbol{\Phi}_n$，分别描述如下：

1. $\boldsymbol{\Lambda}_{i,n}$ 是状态向量 $\mathbf{x}_n$ 关于权值 $\mathbf{w}_j$ 的偏导数所构成的 $q\times(q+m+1)$ 矩阵：

$$\boldsymbol{\Lambda}_{j,n} = \frac{\partial \mathbf{x}_n}{\partial \mathbf{w}_j}, \quad j = 1,2,\cdots,q \tag{15.45}$$

2. $\mathbf{U}_{j,n}$ 是 $q\times(q+m+1)$ 矩阵，除了第 j 行等于向量 $\boldsymbol{\xi}_n$ 外，其他行都为 0：

$$\mathbf{U}_{j,n} = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\xi}_n^T \\ \mathbf{0} \end{bmatrix} \leftarrow \text{第 } j \text{ 行}, \quad j = 1,2,\cdots,q \tag{15.46}$$

3. $\boldsymbol{\Phi}_n$ 是 $q \times q$ 的对角矩阵，它的第 j 个对角元素是激活函数对其自变量的偏导数，可以写

$$\boldsymbol{\Phi}_n = \text{diag}(\varphi'(\mathbf{w}_1^T\boldsymbol{\xi}_n),\cdots,\varphi'(\mathbf{w}_j^T\boldsymbol{\xi}_n),\cdots,\varphi'(\mathbf{w}_n^T\boldsymbol{\xi}_n)) \tag{15.47}$$

有了这些定义，就可以对式(15.42)关于 $\mathbf{w}_j$ 求导。用微积分的链式法则，得到下列递归公式：

$$\boldsymbol{\Lambda}_{j,n+1} = \boldsymbol{\Phi}_n(\mathbf{W}_{a,n}\boldsymbol{\Lambda}_{j,n} + \mathbf{U}_{j,n}), \quad j = 1,2,\cdots,q \tag{15.48}$$

这个递归公式描述实时递归学习过程的非线性状态动力学（即状态演化）。

为了描述这个学习过程，我们需要将矩阵 $\boldsymbol{\Lambda}_{j,n}$ 和误差曲面对 $\mathbf{w}_j$ 的梯度相联系。为此，首先用度量方程(15.11)定义 $p \times 1$ 误差向量：

$$\mathbf{e}_n = \mathbf{d}_n - \mathbf{y}_n = \mathbf{d}_n - \mathbf{W}_c\mathbf{x}_n \tag{15.49}$$

其中 p 是输出向量 $\mathbf{y}_n$ 的维数。根据 $\mathbf{e}_n$ 定义的平方误差瞬间和为

$$\mathscr{E}_n = \frac{1}{2}\mathbf{e}_n^T\mathbf{e}_n \tag{15.50}$$

学习过程的目标是极小化由对所有时间 n 的 $\mathscr{E}_n$ 求和所得到的代价函数，即

$$\mathscr{E}_{\text{total}} = \sum_n \mathscr{E}_n$$

为完成这个目标，使用最陡下降方法，这就需要梯度矩阵的知识，可写为

$$\nabla_{\mathbf{W}}\mathscr{E}_{\text{total}} = \frac{\partial \mathscr{E}_{\text{total}}}{\partial \mathbf{W}} = \sum_n \frac{\partial \mathscr{E}_n}{\partial \mathbf{W}} = \sum_n \nabla_{\mathbf{W}}\mathscr{E}_n$$

其中 $\nabla_{\mathbf{W}}\mathscr{E}_n$ 是 $\mathscr{E}_n$ 对权值矩阵 $\mathbf{W}=\{\mathbf{w}_k\}$ 的梯度。如果需要，可以继续使用这个方程并且得到递归网络的突触权值的更新方程，并且不用近似。但是，为了得到一个实时的训练递归网络使用的学习算法，必须使用一个梯度的瞬时估计值，即 $\nabla_{\mathbf{W}}$，这就导致对最陡下降方法的近似。从某种意义上，我们遵循了第 3 章中最小均方（LMS）算法相似的方法。

回到式(15.50)，以它作为最小化的代价函数，求它对权值向量 $\mathbf{w}_j$ 的微分，得到

$$\frac{\partial \mathscr{E}_n}{\partial \mathbf{w}_j} = \left(\frac{\partial \mathbf{e}_n}{\partial \mathbf{w}_j}\right)\mathbf{e}_n = -\mathbf{W}_c\left(\frac{\partial \mathbf{x}_n}{\partial \mathbf{w}_j}\right)\mathbf{e}_n = -\mathbf{W}_c\boldsymbol{\Lambda}_{j,n}\mathbf{e}_n, \quad j = 1,2,\cdots,q \tag{15.51}$$

因此应用于神经元 j 的突触权值向量 $\mathbf{w}_{j,n}$ 的调整由

$$\Delta\mathbf{w}_{j,n} = -\eta\frac{\partial \mathscr{E}_n}{\partial \mathbf{w}_j} = \eta\mathbf{W}_c\boldsymbol{\Lambda}_{j,n}\mathbf{e}_n, \quad j = 1,2,\cdots,q \tag{15.52}$$

决定，其中 η 是学习率参数，$\boldsymbol{\Lambda}_{j,n}$ 由式(15.48)决定。

现在仅剩下确定开始学习过程的初始条件。为此令

$$\boldsymbol{\Lambda}_{j,0} = \mathbf{0} \quad \text{对于所有 } j \tag{15.53}$$

这意味着递归网络的初始状态停留在一常态。

表 15.1 概括实时递归学习算法。这里所描述的算法公式可应用到任意的对其自变量可微的激活函数 $\varphi(\cdot)$。对于特殊情况，取双曲线切线方程形式的 sigmoid 非线性函数，我们有

$$x_{j,n+1} = \varphi(v_{j,n}) = \tanh(v_{j,n})$$

和

$$\varphi'(v_{j,n}) = \frac{\partial\varphi(v_{j,n})}{\partial v_{j,n}} = \text{sech}^2(v_{j,n}) = 1 - x_{j,n+1}^2 \tag{15.54}$$

其中 $v_{j,n}$ 是神经元 j 的诱导局部域，$x_{j,n+1}$ 是它在 $n+1$ 时刻的状态。

表 15.1　实时递归学习算法小结

参数：

m=输入空间维数

q=状态空间维数

p=输出空间维数

$\mathbf{w}_j$=神经元 j 的突触权值向量，$j=1,2,\cdots,q$。

初始化：

1. 对算法的突触权值赋予从一个均匀分布中选出的较小值。
2. 设状态向量 $\mathbf{x}(0)$ 的初始值为 $\mathbf{x}(0)=\mathbf{0}$。
3. 对 $j=1,2,\cdots,q$，设 $\mathbf{\Lambda}_{j,0}=\mathbf{0}$。

计算：对 $n=0,1,2,\cdots$，计算

$$\mathbf{e}_n=\mathbf{d}_n-\mathbf{W}_c\mathbf{x}_n$$

$$\Delta\mathbf{w}_{j,n}=\eta\mathbf{W}_c\mathbf{\Lambda}_{j,n}\mathbf{e}_n$$

$$\mathbf{\Lambda}_{j,n+1}=\mathbf{\Phi}_n(\mathbf{W}_{a,n}\mathbf{\Lambda}_{j,n}+\mathbf{U}_{j,n}),\quad j=1,2,\cdots,q$$

$\mathbf{x}_n$，$\mathbf{\Lambda}_n$，$\mathbf{U}_{j,n}$ 和 $\mathbf{\Phi}_n$ 的定义分别由式(15.42)、式(15.45)、式(15.46) 和式(15.47)给出。

从真实梯度行为推导

使用瞬时梯度$\nabla_{\mathbf{w}}\mathscr{E}_n$ 意味着实时递归学习算法偏离建立在真正梯度$\nabla_{\mathbf{w}}\mathscr{E}_{\text{total}}$基础上的非实时算法。但是，该偏离和在第 4 章中使用的训练多层感知器的反向传播算法很相似。虽然实时递归算法不保证和总的误差函数$\mathscr{E}_{\text{total}}(\mathbf{W})$ 对权值矩阵 $\mathbf{W}$ 的负梯度精确一致，但实时和非实时的实际差别很小；在算法速率参数 η 减少时它们近似相等。与真正梯度偏离的行为所导致的潜在的最严重的结果，是观测的轨道（由绘制$\mathscr{E}_n$ 对权值矩阵 $\mathbf{W}$ 的元素的图形获得）可能取决于算法产生的权值改变，这也可看作另一个反馈源并从而导致系统不稳定。使参数 η 小到让权值变化的时间尺度远小于网络运行的时间尺度，可以避免生成这个效果。基本上，这和第 3 章中对 LMS 算法提议的算法稳定性是相同的。

例 5　RTRL 算法说明

针对图 15.6 有两个输入和一个输出的完全递归网络，本例我们提出 RTRL 算法的公式。网络有三个神经元，由例 1 的矩阵 $\mathbf{W}_a$，$\mathbf{W}_b$ 和 $\mathbf{W}_c$ 构成。

由于 $m=2,q=3,p=1$，从式(15.44)可得

$$\boldsymbol{\xi}_n=\begin{bmatrix}x_{1,n}\\x_{2,n}\\x_{3,n}\\1\\u_{1,n}\\u_{2,n}\end{bmatrix}$$

设 $\lambda_{j,kl,n}$表示矩阵 $\mathbf{\Lambda}_{j,n}$的第 kl 个元素。利用式(15.48)和式(15.52)分别得到

$$\Delta w_{kl,n}=\eta(d_{1,n}-x_{1,n})\lambda_{1,kl,n}$$

$$\lambda_{j,kl,n+1}=\varphi'(v_{j,n})\Big(\sum_{i=1}^{3}w_{a,ji}\lambda_{i,kl,n}+\delta_{kj}\xi_{l,n}\Big)$$

其中 δ_{kj}是 Kronecker delta，即 $k=j$ 时为 1，其他情况下为 0；$j,k=1,2,3$ 和 $l=1,2,3$ 和 $l=1,2,\cdots,6$。图 15.11 表示一个决定权值调整 $\Delta w_{kl,n}$演化的敏感度图。注意 $\mathbf{W}_a=\{w_{ji}\}$，$(j,i)=1,2,3$ 和 $\mathbf{W}_b=\{w_{jl}\}$，$j=1,2,3$ 和 $l=4,5,6$。并且，不要将 Kronecker delta 和 15.7 节关于 BPTT 的局部梯度相混淆。 ■

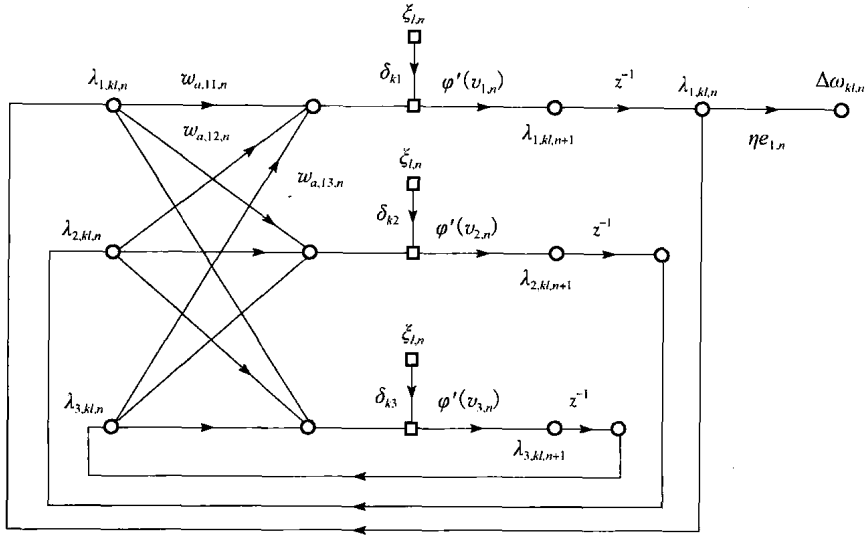


图 15.11　图 15.6 的全连接递归网络敏感度图。注意：标号为 $\xi_{l,n}$ 的三个节点都看作单输入

教师强制

递归网络训练中经常用到的策略是*教师强制*（teacher forcing）(Williams and Zipser, 1989, 1995)；在自适应性滤波中，教师强制称为*方程-误差*（equation-error）方法（Mendel, 1995)。基本上教师强制涉及在网络的训练过程中每当期望响应可用时，在随后网络动态行为的计算中利用期望响应（即目标信号）替代实际神经元的输出。虽然教师强制是在 RTRL 算法下描述的，它的用法可以应用到另外的算法。但是，为了让它是可应用的，问题中的神经元必须将它的输出反馈回网络输入。

教师强制的良好效果包括（Williams and Zipser, 1995)：

- *教师强制可以使网络训练更快*。原因在于使用教师强制等于假设网络已经知道属于那些使用教师强制的神经元的任务的早期部分。
- *教师强制可以作为训练期的校正机制*。例如，网络的突触权值可能有正确的值，但是由于某种原因网络可能运行在状态空间的错误区域。显然在这种情况下，调整突触权值是错误的策略。

基于梯度的学习算法使用教师强制实际上是优化与不用教师强制不同的代价函数。教师强制算法和无强制算法产生不同的解，除非有关的误差信号为 0，这时无需学习。

15.9　递归网络的消失梯度

递归网络的实际应用需要引起注意的一个问题是*消失梯度*（vanishing gradient)，它和依靠很久以前的输入数据用来训练网络使之在当前时刻产生一个期望响应有关。由于组合的非线性，一个时间上隔得远的输入的一个微小变化对网络的训练几乎不会产生影响。即使时间上隔得远的输入的大的变化产生影响，但影响不能被梯度检测到，这时问题同样可能出现。消失梯度问题在一些特定情况下使得基于梯度的训练算法中长期依赖的学习即使不是完全不可能也是变得很困难。

在 Bengio 等（1994）中，对许多实际应用曾经讨论过，在有噪声的情况下需要递归网络能够存储任意时间长度的状态信息。在递归网络状态变量中长期存储的有限位的信息称为*信息锁存*（information latching)。信息锁存必须很鲁棒，不能被与当前学习任务无关的事件删除。

用特殊术语，我们可以陈述如下（Bengio 等，1994）：

如果网络状态包含在一个双曲吸引子的压缩吸引集中，则递归网络的鲁棒性信息锁存就可以实现。

双曲吸引子的概念在 13 章讨论过。一个双曲吸引子的压缩集是在吸引盆的一个点集合，在这些点处 Jacobi 矩阵的所有特征值的绝对值小于 1。这就意味着如果递归网络的状态 $\mathbf{x}_n$ 在一个双曲吸引盆，而不在压缩吸引集中，那么在 $\mathbf{x}_n$ 周围的一个不确定球（ball of uncertainty）的大小会随时间而指数增长，如图 15.12a 所示。所以，对于递归网络输入的小扰动（噪声）能够将轨道推向另一个（可能是错的）吸引盆。但是如果状态 $\mathbf{x}_n$ 继续保持在双曲吸引子的压缩吸引集中，这时在输入 $\mathbf{x}_n$ 能够找到一个有界范围使得 $\mathbf{x}_n$ 停留在吸引子的一定距离之内，如图 15.12b 所示。

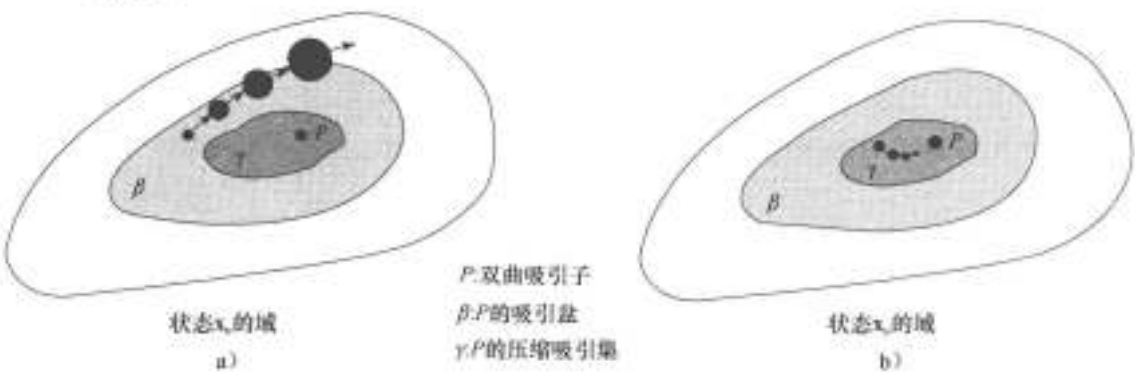


图 15.12　消失梯度问题图示：a）状态 $\mathbf{x}_n$ 在吸引盆 β 内但不在压缩吸引集 γ 内；b）状态 $\mathbf{x}_n$ 在压缩吸引集 γ 内

长期依赖

为了理解梯度基础上学习的鲁棒性信息锁存的作用，我们注意在时刻 n 应用到递归网络的权值向量 $\mathbf{w}$ 由

$$\Delta \mathbf{w}_n = -\eta \frac{\partial \mathscr{E}_{total}}{\partial \mathbf{w}}$$

调整，这里 η 是学习率参数，$\partial \mathscr{E}_{total}/\partial \mathbf{w}$ 是代价函数 $\mathscr{E}_{total}$ 关于 $\mathbf{w}$ 的梯度。代价函数 $\mathscr{E}_{total}$ 通常由

$$\mathscr{E}_{total} = \frac{1}{2}\sum_i \| \mathbf{d}_{i,n} - \mathbf{y}_{i,n} \|^2$$

定义，其中 $\mathbf{d}_{i,n}$ 是期望响应，$\mathbf{y}_{i,n}$ 是网络对第 i 个模式在时间 n 时的实际响应。因此，利用这两个方程，可以写成如下形式：

$$\Delta \mathbf{w}_n = \eta \sum_i \left(\frac{\partial \mathbf{y}_{i,n}}{\partial \mathbf{w}} \right) (\mathbf{d}_{i,n} - \mathbf{y}_{i,n}) = \eta \sum_i \left(\frac{\partial \mathbf{y}_{i,n}}{\partial \mathbf{x}_{i,n}} \times \frac{\partial \mathbf{x}_{i,n}}{\partial \mathbf{w}} \right) (\mathbf{d}_{i,n} - \mathbf{y}_{i,n}) \tag{15.55}$$

其中在第二行使用了微积分的链式法则；状态向量 $\mathbf{x}_{i,n}$ 属于训练样本的第 i 个模式。在应用诸如通过时间的反向传播算法的时候，代价函数的偏微分根据在不同时间标号的独立权值进行计算。可以扩展方程(15.55)的结果如下：

$$\Delta \mathbf{w}_n = \eta \sum_i \left(\frac{\partial \mathbf{y}_{i,n}}{\partial \mathbf{x}_{i,n}} \sum_{k=1}^{n} \frac{\partial \mathbf{x}_{i,n}}{\partial \mathbf{w}_k} \right) (\mathbf{d}_{i,n} - \mathbf{y}_{i,n})$$

第二次应用微积分的链规则得到

$$\Delta \mathbf{w}_n = \eta \sum_i \left(\frac{\partial \mathbf{y}_{i,n}}{\partial \mathbf{x}_{i,n}} \sum_{k=1}^{n} \left(\frac{\partial \mathbf{x}_{i,n}}{\partial \mathbf{x}_{i,k}} \times \frac{\partial \mathbf{x}_{i,k}}{\partial \mathbf{w}_k} \right) \right) (\mathbf{d}_{i,n} - \mathbf{y}_{i,n}) \tag{15.56}$$

根据状态方程(15.2)我们认识到有

$$\mathbf{x}_{i,n} = \boldsymbol{\phi}(\mathbf{x}_{i,k}, \mathbf{u}_n), \quad 1 \leqslant k < n$$

因此我们可以把 $\partial \mathbf{x}_{i,n}/\partial \mathbf{x}_{i,k}$ 解释为非线性函数 $\boldsymbol{\phi}(\cdot,\cdot)$ 扩展到 $n-k$ 个时间步的 Jacobi 矩阵，即

$$\frac{\partial \mathbf{x}_{i,n}}{\partial \mathbf{x}_{i,k}} = \frac{\partial \boldsymbol{\phi}(\mathbf{x}_{i,k}, \mathbf{u}_n)}{\partial \mathbf{x}_{i,k}} = \mathbf{J}_{x,n,k} \tag{15.57}$$

在 Bengio 等（1994）中，证明如果输入 $\mathbf{u}_n$ 使得递归网络在时间 $n=0$ 之后鲁棒地锁存在双曲吸引子内，则 Jacobi 矩阵 $\mathbf{J}_{x,n,k}$ 关于 k 是指数递减的，因此有

$$\det(\mathbf{J}_{x,n,k}) \to 0 \quad \text{当} \quad k \to \infty \quad \text{对于所有的} \ n \tag{15.58}$$

式(15.58)的含义是网络的权值向量 $\mathbf{w}$ 的一个微小变化在最近的过去（即接近当前时间步 n 的 k 的值）有作用。在时间 n 时可能存在权值向量 $\mathbf{w}$ 的调整 $\Delta\mathbf{w}$ 使得 $\mathbf{x}_n$ 移动到一个更好的吸引盆，但代价函数 $\mathscr{E}_{\text{total}}$ 对 $\mathbf{w}$ 的梯度并不携带该信息。

作为结论，假设递归网络的双曲吸引子存储状态信息时使用基于梯度的学习，我们可以发现下列两种情况之一：

- 在输入信号具有噪声时网络不是鲁棒的。
- 网络不能发现长期性依赖（即时间间隔比较长的输入和目标输出之间的关系）。

减缓消失梯度问题的二阶方法

基于梯度的学习算法的运行完全依赖于一阶信息——即 Jacobi 矩阵。因而它们不能充分运用训练数据的信息内容。为了提高在训练数据中包含的信息的使用从而为消失梯度问题提供补救，我们需要向二阶方法寻求帮助。在这一背景下，我们具有两个选择：

1. 我们能利用二阶最优化技术，如在第 2 章和第 4 章讨论过的拟牛顿法、Levenberg-Marquardt法以及共轭梯度法等。尽管这些非线性最优化算法已经证明了其有效性，但它们常常收敛到可怜的局部极小点[10]。

2. 我们能够利用非线性逐次状态估计方法，这在第 14 章中讨论过。在神经网络的训练中，完成了两个功能：
   - 神经网络中权值的演化是以逐次方式进行的。
   - 关于训练数据的二阶信息是以预测-误差协方差矩阵的形式提供的，这也将保持并逐次演化。

在 Puskorius and Feldkamp(2001)、Feldkamp 等（2001）、Prokhorov(2006，2007）报告的多方面工作中说明了形成二阶神经网络训练方法基础的非线性逐次状态估计过程是现实而有效的，可作为面向批量的非线性最优化技术的替代方法。相应地，从此之后我们将注意力集中于利用非线性逐次状态估计过程来训练递归多层感知器。

15.10　利用非线性逐次状态估计的递归网络监督学习框架

为了描述非线性逐次状态估计器是如何在监督方式下训练递归网络的，考虑围绕具有 s 个突触权值和 p 个输出节点的多层感知器建立的递归网络。用 n 来记网络监督训练的每一时间步，令向量 $\mathbf{w}_n$ 记时间步 n 时计算的网络突触权值的全部集合。例如，我们可以这样构造向量 $\mathbf{w}_n$，首先将和第一隐藏层神经元 1 相关联的权值放置在最上面，然后是神经元 2 的权值，继续这一方式直到完成所有第一隐藏层的神经元；然后我们对网络中第二和其他隐藏层做同样的工作直到所有网络权值都以刚刚讨论的顺序体现在向量 $\mathbf{w}_n$ 中。

有了逐次状态估计的思想，训练下的网络的状态空间模型由下面的模型对（参见图

15.13）来定义：

1. 系统（状态）模型，它是由下面的随机行走方程来描述的

$$\mathbf{w}_{n+1} = \mathbf{w}_n + \boldsymbol{\omega}_n \tag{15.59}$$

动态噪声 $\boldsymbol{\omega}_n$ 是高斯白噪，均值为 0，协方差矩阵为 $\mathbf{Q}_\omega$，在系统模型中使用它是为了退火（anneal）在时间上的网络监督训练。在训练的早期阶段，采用大的协方差矩阵 $\mathbf{Q}_\omega$ 来鼓励监督学习算法逃离局部极小点，然后它渐渐地衰减到有限的小值。

2. 测量模型，由下面的方程描述

$$\mathbf{d}_n = \mathbf{b}(\mathbf{w}_n, \mathbf{v}_n, \mathbf{u}_n) + \boldsymbol{v}_n \tag{15.60}$$

其中新的单元定义如下：

- $\mathbf{d}_n$ 是可观测的。
- $\mathbf{v}_n$ 是表示网络中递归节点激活的向量，其元素以和权值向量 $\mathbf{w}_n$ 一致的顺序列出；此后，称 $\mathbf{v}_n$ 为内部状态（internal state）。
- $\mathbf{u}_n$ 是记作用于网络的输入信号的向量；即 $\mathbf{u}_n$ 是作用于网络的驱动力（driving force）。
- $\boldsymbol{v}_n$ 是记破坏向量 $\mathbf{d}_n$ 的测量噪声的向量；它假设为多变量白噪过程，其均值为 0，具有对角协方差矩阵 $\mathbf{R}_n$。噪声源来自于实际获得 $\mathbf{d}_n$ 的途径中。

在式(15.60)中给出的向量值测量函数 $\mathbf{b}(\cdot,\cdot,\cdot)$ 说明了从输入到输出层的多层感知器的总体非线性性；它是递归网络状态空间模型仅有的非线性源。

在所关心的状态的范围内，在网络的监督训练过程中这个概念自然地突出了两个重要的背景：

1. 外部可调整状态，在通过监督训练作用在网络权值的调整上得到体现——因此在式(15.59)和式(15.60)描述的状态空间模型中包含了权值向量 $\mathbf{w}_n$。

2. 内部可调整状态，它由递归节点激活向量 $\mathbf{v}_n$ 来表示；这些激活值是在当前构造的监督训练过程范围之外的，这也是为什么向量 $\mathbf{v}_n$ 仅仅包含在式(15.60)的测量模型中的原因。外部作用驱动力（输入向量）$\mathbf{u}_n$，动态噪声 $\boldsymbol{\omega}_n$ 和围绕多层感知器的全局反馈是时间 n 上 $\mathbf{v}_n$ 演化的原因。

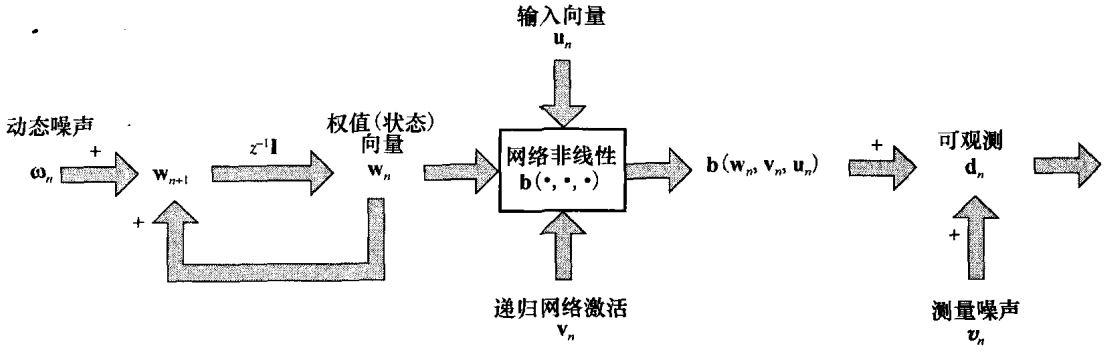


图 15.13　在监督训练下递归网络内在动态的非线性状态空间模型

利用扩展卡尔曼滤波器的监督训练框架描述

给定训练样本 $\{\mathbf{u}_n, \mathbf{d}_n\}_{n=1}^{N}$，感兴趣的问题是如何通过逐次状态估计器的方式来监督训练递归多层感知器（RMLP）。由于式(15.60)，RMLP 是非线性的，逐次状态估计器将不得不对应于非线性。带着这样的要求，我们考虑如何将第 14 章学习过的扩展卡尔曼滤波器（EKT）用于完成这一工作[11]。

从我们目前讨论的目的上看，在表 15.2 中总结的 EKF 算法的有关公式是如下的两个，其中利用了式(15.59)和式(15.60)的状态空间模型的术语。

1. *革新过程* (innovations process)，定义为

$$\boldsymbol{\alpha}_n = \mathbf{d}_n - \mathbf{b}(\hat{\mathbf{w}}_{n|n-1}, \mathbf{v}_n, \mathbf{u}_n) \quad (15.61)$$

其中期望（目标）响应 $\mathbf{d}_n$ 扮演着 EKF “可观测性”的角色。

2. *权值（状态）更新*，定义为

$$\hat{\mathbf{w}}_{n|n} = \hat{\mathbf{w}}_{n|n-1} + \mathbf{G}_n\boldsymbol{\alpha}_n \quad (15.62)$$

其中 $\hat{\mathbf{w}}_{n|n-1}$ 是在时间 n 时 RMLP 权值向量 $\mathbf{w}$ 的预测（老）估计，给定包含时间 $n-1$ 的期望响应，$\hat{\mathbf{w}}_{n|n}$ 是接受到观测值 $\mathbf{d}_n$ 后 $\mathbf{w}$ 的滤波（更新）估计。矩阵 $\mathbf{G}_n$ 是卡尔曼增益，它是 EKF 算法的积分部分。

检查 RMLP 的基本操作，我们发现 $\mathbf{b}(\hat{\mathbf{w}}_{n|n-1}, \mathbf{v}_n, \mathbf{u}_n)$ 是 RMLP 由其“老的”权值向量 $\hat{\mathbf{w}}_{n|n-1}$ 和响应于输入向量 $\mathbf{u}_n$ 的内部状态 $\mathbf{v}_n$ 产生的实际输出向量 $\mathbf{y}_n$。因此可以重写式(15.61)和式(15.62)的组合为单一方程：

$$\hat{\mathbf{w}}_{n|n} = \hat{\mathbf{w}}_{n|n-1} + \mathbf{G}_n(\mathbf{d}_n - \mathbf{y}_n) \quad (15.63)$$

在这一公式的基础上，我们现在可以画出作为两个形成闭递归反馈系统的互相耦合分量的 RMLP 的监督训练，如图 15.14 所示。

表 15.2　RMLP 监督训练的 EKF 算法小结

训练样本：

$\mathcal{T} = \{\mathbf{u}_n, \mathbf{d}_n\}_{n=1}^{N}$

其中 $\mathbf{u}_n$ 是作用于 RMLP 的输入向量，$\mathbf{d}_n$ 是相应的期望响应。

RNLP 和卡尔曼滤波器：参数和变量

$\mathbf{b}(\cdot,\cdot,\cdot)$	：向量值测量函数
$\mathbf{B}$	：线性测量矩阵
$\mathbf{w}_n$	：时间步 n 的权值向量
$\hat{\mathbf{w}}_{n\|n-1}$	：权值向量的预测估计
$\hat{\mathbf{w}}_{n\|n}$	：权值向量的滤波估计
$\mathbf{v}_n$	：RMLP 中递归节点激活向量
$\mathbf{y}_n$	：响应于输入向量 $\mathbf{u}_n$ 而产生的 RMLP 的输出向量
$\mathbf{Q}_\omega$	：动态噪声 $\boldsymbol{\omega}_n$ 的协方差矩阵
$\mathbf{Q}_v$	：测量噪声 $\mathbf{v}_n$ 的协方差矩阵
$\mathbf{G}_n$	：卡尔曼增益
$\mathbf{P}_{n\|n-1}$	：预测误差协方差矩阵
$\mathbf{P}_{n\|n}$	：滤波误差协方差矩阵

计算：

对 $n=1, 2, \cdots$，计算如下：

$\mathbf{G}_n = \mathbf{P}_{n|n-1}\mathbf{B}_n^T[\mathbf{B}_n\mathbf{P}_{n|n-1}\mathbf{B}_n^T + \mathbf{Q}_{v,n}]^{-1}$

$\boldsymbol{\alpha}_n = \mathbf{d}_n - \mathbf{b}_n(\hat{\mathbf{w}}_{n|n-1}, \mathbf{v}_n, \mathbf{u}_n)$

$\hat{\mathbf{w}}_{n|n} = \hat{\mathbf{w}}_{n|n-1} + \mathbf{G}_n\boldsymbol{\alpha}_n$

$\hat{\mathbf{w}}_{n+1|n} = \hat{\mathbf{w}}_{n|n}$

$\mathbf{P}_{n|n} = \mathbf{P}_{n|n-1} - \mathbf{G}_n\mathbf{B}_n\mathbf{P}_{n|n-1}$

$\mathbf{P}_{n+1|n} = \mathbf{P}_{n|n} + \mathbf{Q}_{\omega,n}$

初始化：

$\hat{\mathbf{w}}_{1|0} = \mathbb{E}[\mathbf{w}_1]$

$\mathbf{P}_{1|0} = \boldsymbol{\delta}^{-1}\mathbf{I}$，其中 $\boldsymbol{\delta}$ 是小的正常数，$\mathbf{I}$ 是单位矩阵。

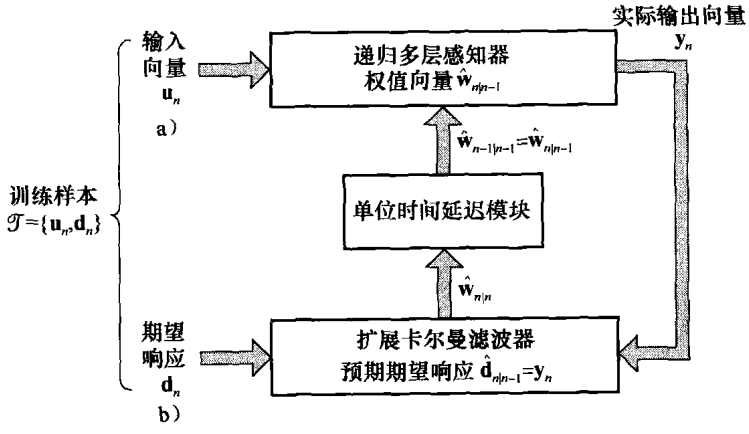


图 15.14　包含 RMLP 和 EKF 的闭递归反馈系统：a) RMLP，权值向量 $\hat{\mathbf{w}}_{n|n-1}$，运行在输入向量 $\mathbf{u}_n$ 上来产生输出向量 $\mathbf{y}_n$；b) EKF，提供了预测 $\hat{\mathbf{d}}_{n|n-1}=\mathbf{y}_n$，运行在期望响应 $\mathbf{d}_n$ 上来产生滤波权值向量 $\hat{\mathbf{w}}_{n|n}=\hat{\mathbf{w}}_{n+1|n}$，从而为下一次迭代准备闭递归反馈系统

1. 图的顶部画出了部分从网络角度看的监督学习过程。权值向量被设为其老的（预测）值 $\hat{\mathbf{w}}_{n|n-1}$，RMLP 计算相应于输入向量 $\mathbf{u}_n$ 的实际输出向量 $\mathbf{y}_n$。因此，RMLP 给 EKF 提供了 $\mathbf{y}_n$ 作为观测值——$\hat{\mathbf{d}}_{n|n-1}$ 的预测估计。

2. 图的底部画出了 EKF 作为训练过程的便捷器（facilitator）的角色。设 $\hat{\mathbf{d}}_{n|n-1}=\mathbf{y}_n$，

EKF 通过在当前期望响应 $\mathbf{d}_n$ 上运行来更新权值向量的老的估计。权值向量的滤波估计（即 $\hat{\mathbf{w}}_{n|n}$）。因而可通过式(15.63)来计算。这样通过 EKF 计算的$\hat{\mathbf{w}}_{n|n}$被提供给 RMLP 作为单位时间延迟模块。

有了等于单位矩阵的转移矩阵，如式(15.59)所证，我们可以为下一次迭代设$\hat{\mathbf{w}}_{n+1|n}$等于 $\hat{\mathbf{w}}_{n|n}$。这一等式允许重复监督训练直到训练终止。

注意在图 15.14 的监督学习框架中，训练样本$\mathcal{T}=\{\mathbf{u}_n,\mathbf{d}_n\}$ 是 RMLP 和 EKF 之间的分割：输入向量 $\mathbf{u}_n$ 作用在 RMLP 上作为激发，期望响应 $\mathbf{d}_n$ 作用在 EKF 上作为观测，它是独立于隐藏权值（状态）向量 $\mathbf{w}_n$ 的。

在第 14 章中，我们强调作为卡尔曼滤波器，其变量和扩展的预测器-修正器性质这一内在特性。根据这一性质，检查图 15.14 的块状图，我们可以做如下陈述：

递归神经网络的训练完成预测器的角色；而扩展卡尔曼滤波器的监督学习完成修正器的角色。

因此，在卡尔曼滤波器对逐次状态估计的传统应用中，预测器和修正器隐藏在卡尔曼滤波器自身中，在监督训练的应用中，这两个角色在递归神经网络和扩展卡尔曼滤波器之间被分割开。这样的监督学习中的责任分割很好地对应了在图 15.14 中训练样本$\mathcal{T}$的输入和期望响应元素的分割。

EKF 算法

为了利用 EKF 算法作为监督学习任务的便捷器，我们需要通过重新训练式(15.60)非线性部分的 Taylor 展开的一阶项来线性化式(15.60)的测量方程。$\mathbf{b}(\mathbf{w}_n,\mathbf{v}_n,\mathbf{u}_n)$ 是唯一的非线性源，我们用下式逼近式(15.60)：

$$\mathbf{d}_n = \mathbf{B}_n\mathbf{w}_n + \boldsymbol{\upsilon}_n \tag{15.64}$$

其中 $\mathbf{B}_n$ 是线性化模型的 $p\times s$ 测量矩阵。线性化过程包括计算 RMLP 的 p 个输出对其 s 个权值的偏导数，得到矩阵

$$\mathbf{B}=\begin{bmatrix}\dfrac{\partial b_1}{\partial w_1} & \dfrac{\partial b_1}{\partial w_2} & \cdots & \dfrac{\partial b_1}{\partial w_s}\\ \dfrac{\partial b_2}{\partial w_1} & \dfrac{\partial b_2}{\partial w_2} & \cdots & \dfrac{\partial b_2}{\partial w_s}\\ \vdots & \vdots & \vdots & \vdots\\ \dfrac{\partial b_p}{\partial w_1} & \dfrac{\partial b_p}{\partial w_2} & \cdots & \dfrac{\partial b_p}{\partial w_s}\end{bmatrix} \tag{15.65}$$

其维数是 $p\times s$。认识到权值向量 $\mathbf{w}$ 的维数是 s，则有矩阵积 $\mathbf{Bw}$ 是 $p\times 1$ 向量，这和观测值 $\mathbf{d}$ 的维数很好地匹配。

在 $\mathbf{b}(\mathbf{w},\mathbf{v}_n,\mathbf{u}_n)$ 中的向量 $\mathbf{v}_n$ 保持相同的常数值；在式(15.65)中时间步 n 被省略了用来简化表达。方程中 $b_i, i=1,2,\cdots,p$，记向量函数 $\mathbf{b}(\mathbf{w}_n,\mathbf{v}_n,\mathbf{u}_n)$ 的第 i 个元素。根据第 14 章的式(14.70)，方程右端项的偏导数在 $\mathbf{w}_n=\hat{\mathbf{w}}_{n|n-1}$上评价，其中$\hat{\mathbf{w}}_{n|n-1}$是权值 $\mathbf{w}_n$ 在时间 n 上的预测，给定包含时间 $n-1$ 的期望响应。

实际上，式(15.65)的偏导数使用通过时间的反向传播（BPTT）或者实时递归学习（RTRL）算法来计算。事实上，EKF 算法建立在这两个算法的一个或其他算法基础上，这两个算法已在 15.7 节和 15.8 节中描述过。这里的意思是 $\mathbf{b}$ 必须是递归节点激活的函数，这在式(15.60)的测量方程中说明了。

式(15.59)的状态演化方程是线性的，因此它不受测量方程线性化的影响。因此，递归网络的线性状态空间模型允许定义在式(15.59)和式(15.64)的 EKF 的应用。

解耦扩展卡尔曼滤波器

在表 15.2 中总结扩展卡尔曼滤波器（EKF）的计算需要，主要是在每个时间步 n 存储和更新滤波误差协方差矩阵 $\mathbf{P}_{n|n}$。对于包含 p 个输出节点和 s 个权值的递归神经网络而言，EKF 的计算复杂度是 $O(ps^2)$，其存储需求是 $O(s^2)$。对大的 s，这些需求可能是高要求。在这种情况下，我们可以通过解耦扩展卡尔曼滤波器（DEKF）作为计算资源合适管理的实际补正(Puskorious and Feldkamp，2001)。

DEKF 的基本思想是忽视递归神经网络确定权值估计之间的交互作用。这样可控制的多个 0 被引入到协方差矩阵 $\mathbf{P}_{n|n}$ 中。更具体地，如果网络的权值以这样的方式解耦，我们创建相互排斥的权值组，则协方差矩阵 $\mathbf{P}_{n|n}$ 构造成如图 15.15 所示的对角块形式。

令 g 记指定的以刚刚描述的方式创建的不相连权值组个数。因此，当 $i=1,2,\cdots,g$，令

$\hat{\mathbf{w}}_{n|n}^{(i)}$ = 第 i 组的滤波权值向量

$\mathbf{P}_{n|n}^{(i)}$ = 第 i 组滤波误差协方差矩阵的子集

$\mathbf{G}_n^{(i)}$ = 第 i 组的卡尔曼增益矩阵

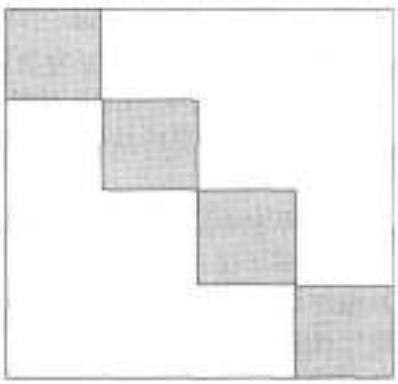


图 15.15　属于解耦卡尔曼滤波器（DEKF）的滤波误差协方差矩阵 $\mathbf{P}_{n|n}^{(i)}$ 的对角块表示。阴影部分表示 $\mathbf{P}_{n|n}^{(i)}$ 的非零值，对图中所示的例子 $i=1,2,3,4$。随着我们使不相连权值组的数目 g 变大，在协方差矩阵 $\mathbf{P}_{n|n}$ 中创建了更多的 0；换句话说，矩阵 $\mathbf{P}_{n|n}$ 变得更稀疏，因而计算负担变少了，但状态估计的数值精确度下降了

对 DEKF 中的其他元素也这样做。滤波器权值向量 $\hat{\mathbf{w}}_{n|n}^{(i)}$ 的连接形成总体滤波权值向量 $\hat{\mathbf{w}}_{n|n}$；对 $\mathbf{P}_{n|n}^{(i)}$ 和 $\mathbf{G}_n^{(i)}$ 以及其他 DEKF 的元素应用相似的记号。根据这些新的记号，将 DEKF 算法重写为如下对第 i 个权值组的式子：

$$\mathbf{G}_n^{(i)} = \mathbf{P}_{n|n-1}^{(i)}(\mathbf{B}_n^{(i)})^T\left[\sum_{j=1}^{g}\mathbf{B}_n^{(j)}\mathbf{P}_{n|n-1}^{(j)}(\mathbf{B}_n^{(j)})^T + \mathbf{Q}_{n,n}^{(i)}\right]^{-1}$$

$$\boldsymbol{\alpha}_n^{(i)} = \mathbf{d}_n^{(i)} - \mathbf{b}_n^{(i)}(\hat{\mathbf{w}}_{n|n-1}^{(i)}, \mathbf{v}_n^{(i)}, \mathbf{u}_n^{(i)})$$

$$\hat{\mathbf{w}}_{n|n}^{(i)} = \hat{\mathbf{w}}_{n|n-1}^{(i)} + \mathbf{G}_n^{(i)}\boldsymbol{\alpha}_n^{(i)}$$

$$\hat{\mathbf{w}}_{n+1|n}^{(i)} = \hat{\mathbf{w}}_{n|n}^{(i)}$$

$$\mathbf{P}_{n|n}^{(i)} = \mathbf{P}_{n|n-1}^{(i)} - \mathbf{G}_n^{(i)}\mathbf{B}_n^{(i)}\mathbf{P}_{n|n-1}^{(i)}$$

$$\mathbf{P}_{n+1|n}^{(i)} = \mathbf{P}_{n|n}^{(i)} + \mathbf{Q}_{n,n}^{(i)}$$

DEKF 算法的初始化以前面在 EKF 算法的表 15.2 描述的方式进行。

DEKF 的计算需要假设为如下的阶：

计算复杂度：$O\left(p^2 s + p\sum_{i=1}^{g} s_i^2\right)$

存储需要：$O\left(\sum_{i=1}^{g} s_i^2\right)$

其中 s_i 是组 i 中状态的大小，s 是总体状态大小；p 是输出节点数。依赖于不相连组个数 g，DEKF 的计算需要可以比 EKF 显著减小。

EKF 的总结批注

利用 EKF 作为递归神经网络监督训练逐次状态估计器的有吸引力的特征是其基本算法结构（因而其执行）相对简单，正如表 15.2 的总结所证。然而，它有如下两个实际局限：

1. EKF 需要线性化递归神经网络向量测量函数 $\mathbf{b}(\mathbf{w}_n, \mathbf{v}_n, \mathbf{u}_n)$。

2. 依赖于权值向量 $\mathbf{w}$ 的大小（即状态空间的维数），我们可能必须利用 DEKF 来减少计算复杂度和存储需要。然而这一实际问题是我们因此牺牲了计算精确度。

我们可以通过利用无导数非线性逐次状态估计器来回避第一个局限，这在下面讨论。

利用无导数逐次状态估计器做神经网络的监督训练

在第 14 章中，我们讨论了*数值积分卡尔曼滤波器*（Arasaratnam 和 Haykin，2009），其形成基于称为*数值积分规则*（Stroud，1971；Cools，1997）的数值方法。类似于 EKF，数值积分卡尔曼滤波器（CKF）是贝叶斯滤波器的逼近实现；然而，在理论背景下，CKF 是逐次状态估计的最优非线性滤波器。CKF 具有一些独有的性质：

1. CKF 是比 EKF 对贝叶斯滤波器更加数值精确的逼近器。它完全保留了状态的包含在观测值中的二阶信息。

2. CKF 是*无导数的*；因此，没有线性化递归神经网络测量矩阵的需要。

3. 最后但不限于这几点，数值积分规则被用于逼近时间更新积分，这包含了后验分布和所有其他高斯环境下运行的贝叶斯滤波器形式的积分公式；作为一个规则，积分比微分好，因为其“平滑”性质。

根据这些性质，可以说 CKF 是递归神经网络监督学习的有高度吸引力的选择。将在 15.11 节中描述的实验包含了混沌吸引子的动态重构，说明了 CKF 比 EKF 以及另一个称为*中心差分卡尔曼滤波器*（CDKF）[12]的无导数逐次状态估计器更好的性能。Nörgaard 等（2000）的 CDKF，通过用基于 Stirling 公式的展开来代替权值向量当前估计附近非线性测量方程的 Taylor 级数展开来推导，在指定区间上插入分析函数。在一维情况下，可通过相应地替换 Taylor 展开的一阶和二阶偏导数为一阶和二阶中心差分来得到 Stirling 公式[13]。然后，一旦测量方程的逼近线性化在多维设置下推导，CDKF 算法遵循卡尔曼滤波器理论。原始的 CDKF 算法在 Nörgaard 等（2000）中描述，采用方根滤波来提高数值精确度；这一过程在第 14 章卡尔曼滤波的上下文中描述过。

15.11　计算机实验：Mackay-Glass 吸引子的动态重构

Mackey-Glass 吸引子是 Mackey and Glass(1977) 在模型化人体血液细胞动态构成时首先形成的。它通过下面单一的连续时间微分方程来描述：

$$\frac{\mathrm{d}}{\mathrm{d}t}x_t = -bx_t + \frac{ax_{t-\Delta t}}{1 + x_{t-\Delta t}^{10}} \tag{15.66}$$

其中 t 记连续时间，系数 $a=0.2$ 和 $b=0.1$，时间延迟 $\Delta t=30$。正式意义上 Mackey-Glass 吸引子具有无限多的自由度，因为我们需要连续时间区间上的函数 $x(t)$ 的初始值。然而，它行为上像是具有有限维数的奇异吸引子。

为了数值上解式(15.66)，我们利用四阶 Runge-Kutta 方法（Press 等，1988），$6s$ 的取样周期，初始条件 $x_n=0.9$，$0 \leqslant n \leqslant \Delta t$，其中如通常一样，$n$ 记离散时间。我们因此获得长度 1000 的时间序列，前一半用于训练，剩下的用于测试。给定混沌吸引子，我们回顾第 13 章，下一个数据样本 $x_{n+\tau}$ 能由恰当选择的时间序列 $\{x_n, x_{n-\tau}, \cdots, x_{n-[d_E-2]\tau}, x_{n-[d_E-1]\tau}\}$ 来预测，其中 d_E 和 τ 分别称为*嵌入维数*（embedding dimension）和*嵌入延迟*（embedding delay）。对于混沌

Mackey-Glass 系统，d_E 和 τ 分别选为 7 和 1。

递归多层感知器（RMLP）被证明了在学习时间相关信号时是数值鲁棒的。对这个实验，我们执行一个具有 7 个输入（表示观测时间序列的嵌入）1 个输出和一个具有 5 个神经元的自循环隐藏层。因此，RMLP 具有总共 71 个突触权值（包含了偏置参数）。输出神经元利用线性激活函数，所有的隐藏神经元利用双曲正切函数：

$$\varphi(v) = \tanh(v)$$

三个算法的方根方案被用来训练 RMLP：扩展卡尔曼滤波器，中心差分卡尔曼滤波器，以及数值积分卡尔曼滤波器。为了展开神经网络的递归循环，我们使用切断深度 $h=1$，对这一实验是充分的。而且，对 EKF 算法，我们使用反向传播算法来计算非线性测量函数 $\mathbf{b}_n$ 的偏导数，使用 15.7 节中描述的过程。

对所有三个算法，每次运行使用 10 个回合来训练 RMLP。每个回合从包含 107 个时间步的长子序列中获得，从随机选择点开始。更精确地说，每个回合由 100 个样本组成，是通过一个长度为 8 的窗口在子序列上滑动而得的。RMLP 的权值被初始化为 0 -均值高斯分布，其对角协方差矩阵是 $10^{-2}\times\mathbf{I}_s$，其中 $\mathbf{I}_s$ 是 $s\times s$ 单位矩阵。

为了以公平方式比较 CKF 训练的 RMLP 和 CDKF、EKF 训练的 RMLP，我们做了 50 次独立训练。为了测量从 500 个时间索引开始的 100 个时间步预测的性能，我们使用总体平均累积绝对误差，由下式定义

$$\mathscr{E}_n = \frac{1}{50}\sum_{r=1}^{50}\sum_{i=1}^{n} |\, d_i^{(r)} - \hat{d}_i^{(r)} \,|; \quad n = 1,2,\cdots,100$$

其中 $d_i^{(r)}$ 是时间 i 对第 r 次运行的期望响应，$\hat{d}_i^{(r)}$ 是在 RMLP 的输出端计算得到的估计。长期累积预测误差是随时间 n 而增长的函数。

如已经指出的那样，在这个实验中使用了贝叶斯滤波器的三个不同逼近：

- 扩展卡尔曼滤波器（EKF）
- 中心差分卡尔曼滤波器（CDKF）
- 数值积分卡尔曼滤波器（CKF）

实验结果在图 15.16 中给出，其中画出了动态重构的总体平均累积绝对误差对动态重构中使用的预测时间步的图形。正如期望的那样，实验结果为 CKF 和 CDKF、EKF 相比具有更好的性能并提高了计算精度提供了清晰的证据。

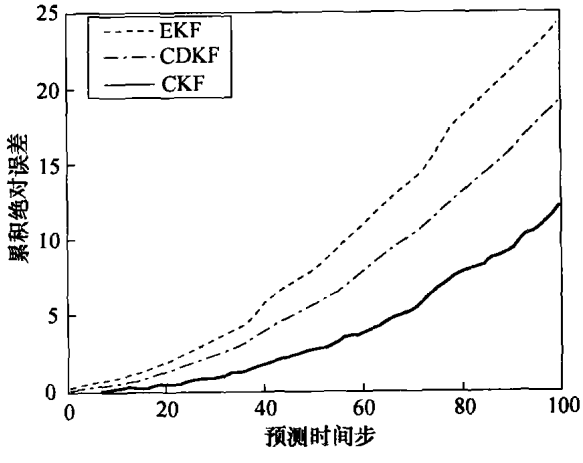


图 15.16　Mackey-Glass 吸引子动态重构自主预测阶段中总体平均累积绝对误差曲线

15.12　自适应考虑

递归神经网络（如 RMLP）的一个有趣的性质是在网络以监督方式训练后观测到的自适应行为的显露[14]。这一现象的出现无视网络突触权值已经固定的事实。这一自适应行为可以追溯到如下的基本定理（Lo and Yu，1995b）：

考虑在具有相对小的统计行为变化的随机环境中的递归神经网络。如果环境的内在概率分布是通过提供给网络的监督训练样本完全表示的，这一网络可能自适应到相对小的环境的统计变化，不需要对网络的突触权值做更多在线修正。

这一基本定理仅对递归网络有效。我们这样说是因为递归网络的动态状态实际上是作为“短时记忆”，包含了网络所在用于自适应的不确定环境的估计或统计。

这一自适应行为在文献中称呼不同。在 Lo（2001）中，它被称为*适应学习*（accommodative learning）。在同一年发表的另一个论文中（Younger 等，2001），它被称为*元-学习*（meta-learning），意味着“学习如何去学习”。从此，我们将称这种自适应行为“元-学习”。

不管这个自适应行为是如何称呼，并不能期望它和真正的自适应神经网络工作得一样有效，此时如果环境表现出大的统计变化将提供自主在线权值调整。这一观测在 Lo(2001) 中通过实验证实，此时在使用元-学习的递归神经网络和具有长时和短时记忆的自适应神经网络之间进行了性能比较；比较评估是在系统辨识的背景下完成的。

无论如何，递归神经网络的元-学习能力可看成是控制和信号处理应用中期望的性质，尤其是在突触权值的在线调整不是实际可行或者完成这一工作代价太高的时候。

自适应评价

对感兴趣的应用来说如果递归神经网络的监督训练不能得到期望响应，已有的非监督训练方法不能足够快地收敛，则强化学习（即逼近动态规划）可能是仅有的可用选择。从第 12 章，我们回顾逼近动态规划，一个智能体（即学习系统）需要从其所在的环境仅对智能体采取的行动有响应。基本上，在智能体和其环境间的实时交流是我们需要构造短时记忆以允许递归神经网络的内部状态自适应到环境的统计变化。

递归神经网络的突触权值固定后，内部状态能够自适应的唯一途径是通过作用于网络内部递归节点激活上的调整，该激活由式(15.60)测量方程中的向量 $\mathbf{v}_n$ 来记。因此，与作用于隐藏权值向量 $\mathbf{w}_n$ 的监督调整不同，对向量 $\mathbf{v}_n$ 的调整是直接作用于式(15.60)的测量方程上的。

图 15.17 的块状图画出了围绕固定权值递归神经网络建立的方案，此时递归节点激活能实时自适应。具体来说，我们具有*自适应评价*（adaptive critic），它接受两个输入，一个是从网络而来，另一个是从响应于网络采取的相关行动（如智能体）的环境而来。作为这两个输入的响应，自适应评价计算网络内部递归节点行为的合适调整。

作为总结，我们可以说通过使用自适应评价，递归神经网络装备有下面两种形式的记忆：

1. 长时记忆，它是网络自身通过监督训练而取得，其结果是固定权值集。

2. 短时记忆，它使得网络能够自适应其内部状态（即递归节点激活）于环境的统计变化，不影响固定权值。

值得注意的是通过和环境的连续交流，短时记忆能发展成*无模型设置*（model-free setting），这在 Prokhorov(2007) 中描述。

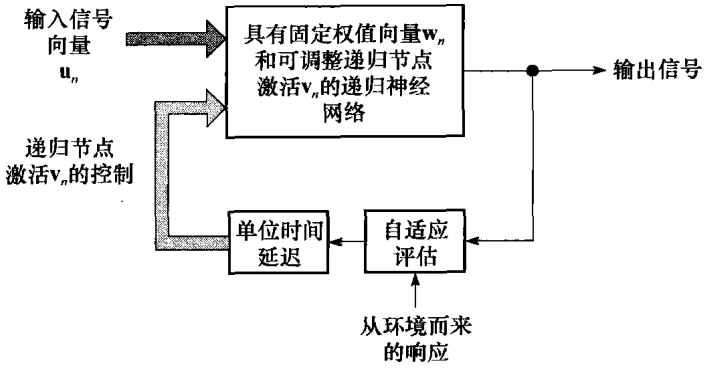


图 15.17　递归神经网络（假设具有单一输出）中使用自适应评价来控制递归节点激活的块状图

15.13　实例学习：应用于神经控制的模型参考

在本章的最后一个主题中，我们讨论一个实例学习，它不仅非常适合本章，而且将本书前面章中讨论过的几个主题放到了一起。

具体而言，我们讨论递归神经网络在反馈控制系统设计中的重要应用，此时设备（plant）的状态和强加的控制非线性耦合。系统的设计因为其他因素而变得更加复杂，如未测量的随机分布的存在、不唯一设备逆的可能性、设备状态不可观测等。

适合使用递归神经网络的控制策略是模型参考控制（model-reference control）(Narendra and Annaswamy，1989；Puskorius and Feldkamp，2001；Prokhorov，2006)。如图 15.18 所示，模型参考控制系统包括五个函数分量：

1. 设备，它被控制以补偿设备动态的改变。作为控制信号和其自身参数向量 $\boldsymbol{\theta}_k$ 的函数的设备输出随时间而演化，其中 $\boldsymbol{\theta}_k$ 中的时间参数 k 远远不如时间索引 n 改变的频率快。例如，$\boldsymbol{\theta}_k$ 可以是分段常数的，k 变化时它从一个常数层转换到另一个。

2. 神经控制器，它以由递归多层感知器为例的递归网络组成。它提供作用在设备输入上的控制信号。这一信号作为参考信号、反馈信号的函数变化，控制器的权值向量记为 $\mathbf{w}$。

3. 模型参考，它被假设为稳定的。模型参考提供响应于参考信号的期望信号作为输入。

4. 比较器，由求和单元表示，它比较设备输出和模型参考的期望响应来产生误差信号。

5. 单位时间延迟模块，表示为 $z^{-1}\mathbf{I}$，它通过配比设备输出向量元素和参考信号元素来关闭围绕设备的反馈循环；事实上，外部递归网络是通过反馈循环来实现的。

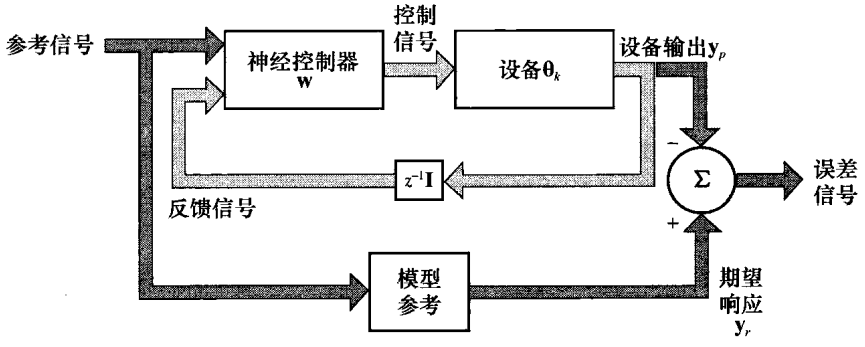


图 15.18　模型-参考自适应系统

由这一描述，很明显设备输出是通过控制信号和设备自身参数向量 $\boldsymbol{\theta}_k$ 的直接函数的神经控制器权值向量 $\mathbf{w}$ 的非直接函数。我们因此可以将设备输出表示为 $\mathbf{y}_{i,p}(n,\mathbf{w},\boldsymbol{\theta}_k)$，其中下标 i 表示设备操作的特别样例。设备输出显式依赖于时间 n 是包含在强调设备非稳定行为上的。相应地，令 $\mathbf{y}_{i,r}(n)$ 记模型参考对同一样例的输出。参考信号对模型参考自适应控制系统的两个前向路径是共同的；我们在设备输出或模型参考输出中不包含对参考信号的依赖来简化问题。

误差信号通过对每个样例 i 的模型参考输出和设备输出之间的差来定义。因此形成均方误差

$$J(\mathbf{w},\boldsymbol{\theta}_k)=\frac{1}{T}\sum_{n=1}^{T}\sum_{i}\|\mathbf{y}_{i,r}(n)-\mathbf{y}_{i,p}(n,\mathbf{w},\boldsymbol{\theta}_k)\|^2 \tag{15.67}$$

其中内部的求和是在训练神经控制器的整个样例集合上进行的，外面的求和是在整个训练过程 $1\leqslant n\leqslant T$ 上取的。为了给出对于参数改变和外部扰动（后者在图 15.18 中给出）鲁棒的神经控制器的设计，通过这样的方式来调整神经控制器的权值向量 $\mathbf{w}$，即均方误差 $J(\mathbf{w},\boldsymbol{\theta}_k)$ 和其最大值在设备的参数向量 $\boldsymbol{\theta}_k$ 的所有可能值上衰减（Prokhorov，2006）。这一最优性使得设备输

出追踪模型参考输出。

在图 15.18 中的模型参考控制系统标为“设备”的块具有双重意义，依赖于从神经控制器的角度是如何看的：

- 一种意思是作为设备被控制的实际系统。
- 另一个意思是那个实际系统的模型。

相应地，我们可以利用*直接控制*（direct control）来补偿设备动态中的不确定性，此时实际设备被用于控制系统，或者采用*间接控制*（indirect control），此时设备模型被用于控制系统（Adetona 等，2000）。

在多种情形下，我们发现设备基于物理的模型（即被控制的实际系统）是适当的；在工业中这样的模型的有效性是常见的，这是对时间以及努力的大量投资的结果上知道的。作为替代，我们可以利用在导言中讨论的系统辨识原则来建立设备的基于神经网络的模型。然而典型地，我们发现下面情况（Prokhorov，2006）：

1. 基于物理的模型比基于神经网络的模型更精确。
2. 基于物理的模型不包括专用可微元素。

Prokhorov(2006) 报告的用于训练神经控制器的方法是方根状态估计算法的修正版本，方根状态估计算法是由 Nörgaard 等（2000）提出。如前所述，这一算法合适称为*中心差分卡尔曼滤波器*（CDKF）。

在 Prokhorov(2006) 中给出的实验结果不仅通过非线性逐次状态估计框架验证神经控制器的训练，也描述了由无导数 CDKF 算法所获得的比依赖于导数的 EKF 算法更好的精度。

15.14　小结和讨论

递归网络模型

本章讨论涉及应用*全局反馈*到静态（无记忆）多层感知器的递归网络。反馈的应用使得神经网络获得状态表示，使得它们成为信号处理和控制中各种应用的合适工具。属于有全局反馈的递归网络类型的四个主要网络结构如下：

- 使用从输出层反馈到输入层的具有外部输入的非线性自回归（NARX）网络。
- 具有从隐藏层到输入层反馈的完全连接递归网络。
- 有多于一个隐藏层的递归多层感知器，其中每个计算层输出反馈到它自己的输入。
- 使用二阶神经元的二阶递归网络。

在所有这些递归网络中，反馈通过抽头延迟线记忆。

前三个递归网络可以使用状态空间框架研究其动态行为。这个根植于现代控制论的方法提供一个研究非线性动态递归网络的有力工具。

递归神经网络的性质

下面是递归神经网络的一些重要性质：

1. 它们是非线性动态系统的通用逼近器，如果它们具有充分多的隐藏神经元的话。
2. 它们是局部可控制和局部可观测的，如果它们的线性方案满足围绕平衡点的一定条件的话。
3. 给定任意的有限状态机器，我们能够建立作为黑盒机器的递归神经网络，其行为像有限状态机器。
4. 递归神经网络表现出元-学习（即学习如何学习）的能力。

事实上，正是这些性质使得递归神经网络适合于计算、控制、信号处理等的应用。

基于梯度的学习算法

在本章中，我们讨论了两种基于监督学习算法的训练递归网络的算法：通过时间的反向传播（BPTT），实时递归学习（RTRL）。这两者是建立在梯度基础上的，这使其执行时计算简单。BPTT 更适合于离线学习，而由定义 RTRL 是设计用于在线学习的。然而，这两个算法的实际局限是消失梯度问题，这是因为它们不能使用训练数据中的二阶信息而导致的。

基于非线性逐次状态估计的监督学习算法

克服消失梯度问题的一个有效的方法是利用非线性逐次状态估计来为递归多层感知器提供监督训练。这里我们具有两个可用的选择：

1. 因为其计算简单性，我们可以使用扩展卡尔曼滤波器（EKF）。然而，我们必须利用 BPTT 或 RTRL 算法来为属于递归神经网络的测量模型提供线性化。

2. 我们可以利用无导数非线性逐次状态估计，以第 14 章描述的数值积分卡尔曼滤波器（CKF）和本章简单介绍的中心差分卡尔曼滤波器（CDKF）为例。这样，我们不仅拓宽了这一新方法对监督学习的应用，也提高了数值精度。然而，要付出的代价是增加计算需求。

在这三个非线性滤波器中，CKF 的突出性不仅体现在其最逼近于贝叶斯滤波器（至少从概念意义上是最优的）也因为其在三者中是最强大的。假设高斯性，CKF 的构造受卡尔曼滤波理论（如革新过程）的影响，如第 14 章所讨论的那样。

不论怎样，监督学习的这一新方法是好的，如图 15.14 的 EKF 块状图所证明。更重要的是，这一过程可以应用于递归神经网络和其他的神经网络（如多层感知器）。而且，因为这一通用应用性，我们可以将这一类监督学习的非线性逐次状态估计算法（包括 EKF，CDKF，CKF）作为*启动技术*（enabling technology），使其能够解决困难的信号处理和控制问题，尤其是大规模学习问题中二阶信息的使用几乎是“必须”的。

从理论上，具有全局反馈的递归网络（例如，用 EKF 算法训练的递归多层感知器）能通过把训练样本中获得的知识存储到权值固定集中学习非稳定环境下的内在动力学。更重要的是，网络能够*追踪*环境的统计变化，如果下面的两个条件得到满足：

- 递归网络不发生欠适应（underfitting）或过适应（overfitting）。
- 训练样本能表示环境的小的统计变化。

多路训练

在图 15.14 中描述的递归网络监督训练方法可能从称为*多路训练*（multistream training）的过程中获益。这一过程应用于这样的情形：通过利用多样本模式的优点*坐标权值更新*（coordinated weight update）是有利的（Puskorius and Feldamp，2001）。

在神经网络的监督训练中，依赖于输入-目标响应对训练序列的性质可能出现两种方案：

1. *同种序列*（homogeneous sequences），此时通过训练数据的一个或多个通过可以很好地产生满意结果。

2. *异种序列*（heterogenous sequences），此时，例如，可能在输入-目标响应对中快速变化区域之后紧随着慢速变化区域。

在后一种方案下，存在着标准训练过程中网络权值为了当前出现的训练数据而不适当更新的倾向，我们称之为*新近效应*（recency effect）。对于前馈网络，有效的解决办法是打乱（shuffle）提供给神经网络的训练数据的顺序，或者利用训练的批量形式；这两种方法都在第 4 章中讨论过。对递归神经网络，打乱数据顺序的直接模型是随机选择子序列；这样做具有仅对子序列最后的输入-目标响应对进行权值更新的效果。例如，在利用 EKF 算法的训练过程的

情形，是完全的批量更新。它通过完整的训练样本运行递归网络，对每一个输入-目标响应对计算必要的偏导数，然后基于整个估计误差集更新网络权值。

多路训练过程通过打乱（即随机选择子序列）和批量更新的组合应用来克服新近效应。特别地，多路训练基于这样的原则：每一次权值更新都代表着联立方式下多个输入-目标响应对的信息内容。

作为最后的备注：多路训练不仅在使用 EKF 算法时是可用的，也在使用无导数非线性逐次状态算法（如 CDKF 和 CKF）时可用。

最终结束备注：大规模学习问题

作为本章的最后一个小节，而本章也是全书的最后一章，本节将讨论大规模学习问题。特别地，在前面的三个章节中也以一定篇幅讨论过这一问题：

- 在关于多层感知器的第 4 章中，学习了大规模学习问题和小规模学习问题的比较。
- 在关于正则理论的第 7 章中，我们利用可微流形来构造能够开发包含在标注训练样本和未标注样本中信息的半监督学习策略。
- 然后再次在动态规划的第 12 章中，维数灾问题在处理大规模动态环境时成为严重关心的问题。

在模式分类和非线性回归监督学习问题的背景下，处理这些问题的过程是容易理解的，这由本书中给出的内容得到验证。另一方面，能够正当地宣称大规模学习问题的研究还处在早期阶段。

事实上，我们可以将大规模学习问题看成是关于*学习的未来*（future of learning）的视窗。这一视窗将我们直接带到实际世界。相应地，我们可以辨别在处理大规模学习问题时的四个具体阶段：

1. *用于训练数据源的详细清单的开发。*这第一阶段是非常重要的，因为毕竟训练数据提供了属于这一问题的实际世界和被研究来解这一问题的学习机之间的联系。这一训练数据源的清单可能包括：
   - 高质量有标签数据。
   - 不是那么高质量的有标签数据。
   - 大量无标签数据。

   给定这样训练数据的混合，挑战在于如何构造值得追求的训练策略的不同方案，在计算资源有限的情况下实现。

2. *相应于生成训练数据的环境的模型化。*在第二个阶段，挑战在于构成网络模型，它具有足够多的自由度并且是正确的。在构造中的目标是捕获相应于数据生成的环境的内在统计物理过程（性质）。这一问题的实质是，除非这一问题被正确解决，否则将不可避免地在数据生成的物理现实和提案的网络模型理论基础之间存在不匹配。如果模型的不匹配很严重的话，此后无论怎么做也不能治愈模型的缺陷。

3. *用于估计网络模型可调整参数的算法选择。*第三个阶段的挑战性在于我们必须选择以计算有效的方式良好适合于估计模型未知参数的算法。更精确地，网络模型必须具有从输入到输出的充分深度来有效地处理问题。

4. *可调整参数的最优估计。*最后的挑战是选择具有可靠地提取训练数据信息内容的内在能力的优化算法。典型地，二阶信息被认为是适合的。最重要的是，优化算法必须是计算效率高的。在这一背景下，有两个潜在的候选者：
   - 非线性逐次估计算法，以数值积分卡尔曼滤波器为例。

- 二阶优化算法，以高斯-牛顿和 Levenberg-Marquardt 算法的在线方案为例，当估计精度被合理地良好保持时，找到了免除精确计算 Hessian 矩阵的方法。

我们以这样的说法来结束本书：在解实际世界大规模学习问题时，认真地对待这里描述的四个阶段我们才能确信实现成功解。

注释和参考文献

1. 关于其他递归网络结构，参考 Jordan(1986)，Back and Tsoi(1991)，Frasconi 等（1992）。
2. NARX 模型包括一类重要的非线性离散时间系统（Leontaritis and Billings，1985）。涉及神经网络这方面的讨论可以参考 Chen 等（1990）、Narendra and Parthasarathy（1990）、Lin 等（1996）和 Sieglemann 等（1997）。

   已经证实 NARX 模型十分适合对非线性系统进行建模，如热交换器（Chen 等，1990），污水处理设备（Su and McAvoy，1991；Su 等，1992），用于石油提炼的催化更新系统（Su 等，1992），在生物系统中的多肽移动的非线性振荡（Venkataraman，1994）和语法推理（Giles and Horne，1994）。

   NARX 模型也指非线性自回归滑动平均（NARMA）模型，其中“滑动平均”是对于输入而言。
3. 递归多层感知器是延时递归神经网络（TLRNN）的特例。这一递归网络的一般类允许使用神经网络节点间连接的任意模式；另一方面，递归多层感知器具有连接的层模式。TLRNN 提供下面重要的特性（Lo，1993）：

   (i) 它们包含传统的结构如有限时宽脉冲响应（FIR）。

   (ii) 它们具有解释非线性动态系统中强隐藏状态的内在能力。

   (iii) 它们是非线性动态系统的通用逼近。
4. Omlin and Giles(1996) 指出，用二阶递归网络，任何有限状态自动机可以映射到这样一种网络，且可以保证有限长度的时序序列的正确分类。
5. 可控性和可观测性的严格处理可以参考 Zadeh and Desoer(1963)、Kailath(1980)、Sontag(1990)。
6. 有关神经网络和自动机（实际上是串行机器-自动机的实现）方面的最早工作，即第一篇关于有限状态自动机、人工智能和递归神经网络方面的论文，是 McCulloch and Pitts(1943) 的经典的论文。递归网络（具有瞬时反馈）是这篇论文的第二部分，这在 Kleene(1956) 被解释为一个有限状态自动机。Kleene 的论文出现在由 Shannon 和 McCarthy 编辑的《自动机研究》（Automata Studies）一书中（这本惊世之作的作者还包括 Moore、Minsky、von Neumann、Uttley、McCarthy 和 Shannon 等人）。有时候，Kleene 的论文被作为有限状态机器方面的第一篇文章引用（Perrin，1990）。Minsky(1967) 在他的《计算：有限和无限机器》（Computation：Finite and Infinite Machines）一书中讨论自动机和神经网络。

   所有关于自动机和神经网络方面的早期工作主要考虑怎样将二者结合在一起，即如何建造和设计自动机到神经网络中去。因为大多数自动机（当被实现为串行机器的时候）需要反馈，神经网络必须为递归的。注意早期的工作（除了 Minsky 的）并没有明确地区分自动机（有向图、标记图和无圈图）和串行机器（逻辑延时和反馈延时），大多数情况下仅考虑有限状态自动机，而对于提高自动机的层次到下推自动机和图灵机没有什么兴趣（除了 Minsky 之外）。

   在神经网络的黑暗时代过去之后，关于自动机和神经网络方面的研究在 20 世纪 80 年代又开始了。这个工作可以大概分为下面三个大的领域：（1）学习自动机；（2）自动机关于知识的合成、抽取和提炼；(3) 表示。首先提到自动机和神经网络的是 Jordan(1986)。
7. 使用 McCulloch-Pitts 神经元的单层递归网络不能模拟任何有限状态的机（Goudreau 等，1994），但 Elman 的简单递归网络可以进行这样的模拟（Kremer，1995）。只有局部反馈的递归网络不能表示所有有限状态机（Frasconi and Gori，1996；Giles 等，1995；Kremer，1996）。换句话说，全局反馈的使用是通过神经网络模拟有限状态的必要需求。
8. 通过时间的反向传播的思想，是对于每一个递归网络都可能建立一个前馈网络，使之在一个特定的时间间隔内具有和它相同的行为（Minsky and Papert，1969）。通过时间的反向传播首先在 Werbos(1974) 的博士论文讨论过；也可以参考 Werbos(1990)。这个算法由 Rumelhart 等，(1986b) 独立地重新发现。通过时间的反向传播算法的一个变体由 Williams and Peng(1990) 所讨论。对于算法的综述和相关的问题，可以参考 Williams and Zipser(1995)。

9. 实时递归学习算法在神经元网络文献中的第一次描述是 Williams and Zipser(1989)。其来源可以追溯到 McBride and Narendra(1965) 用于调节任意动态系统参数的系统辨识的论文。

   Williams 和 Zipser 给出的推导是关于完全递归的单层神经网络。它已扩展为更一般的结构；例如，参考 Kechriotis 等 (1994)；Puskorius and Feldkamp(1994)。

10. Schraudolph(2002) 描述了随机元下降 (stochastic meta-descent) (SMD) 算法，其中提出了通过迭代逼近来放弃计算精确的 Hessian 矩阵的概念。特别地，一个特殊的弧度矩阵-向量积被引入到如高斯-牛顿和 Levenberg-Marquardt 方法等迭代逼近二阶梯度方法中，得到改进的稳定性和性能。

11. Singhal and Wu(1989) 也许是第一个展示用扩展卡尔曼滤波器提高监督神经元网络的映射性能。不幸的是，那里讨论的训练算法受限于它计算的复杂性。为克服这个困难，Kollias and Anastassiou(1989)，Shah and Palmieri(1990) 尝试通过将全局问题分为一系列子问题，每个子问题表示一个单一的神经元，以简化扩展卡尔曼滤波器的应用。但是作为一个辨识问题的每一个神经元的处理并不是严格地遵守卡尔曼滤波器理论。还有，这样处理会导致训练过程中的不稳定行为，并且可能得到比别的方法得到的结果还差的解 (Puskorius and Feldkamp，1991)。

12. 在 Prokhorov(2006，2007) 和相关的论文中，由 Nörgaard，Poulsen，and Ravn(2000) 而来的逐次状态估计算法被称为 nprKF 算法，其中 "npr" 是从算法的三个作者的第一个字母中取出。在本章中，我们优先选择将这一算法命名为中心差分卡尔曼滤波器 (CDKF)，这是对这一算法基础的更好描述。

13. 考虑具有变量 x 的函数 $f(x)$。令 f_k 记函数在 $x=x_k$ 时的值。中心差分定义为：

$$\delta f_{k+\frac{1}{2}} = f_{k+1} - f_k \quad \text{对于每个 } k$$

   其中左边的下标是右边两个下标的平均。下面的表高阶中心差分是如何构造的：

x	f				
x_0	f_0				
		$\delta f_{1/2}$			
x_1	f_1		$\delta^2 f_1$		
		$\delta f_{3/2}$		$\delta^2 f_{3/2}$	
x_2	f_2		$\delta^2 f_2$		$\delta^4 f_2$
		$\delta f_{5/2}$		$\delta^2 f_{5/2}$	
x_3	f_3		$\delta^2 f_3$		
		$\delta f_{7/2}$			
x_4	f_4				

   注意表中具有相同下标的元素总是处于水平或中心 (centrally) 展开到表的行上 (Wylie and Barrett，1982)。

14. 以递归多层感知器为例的递归神经网络自适应行为的出现，首先由 Lo and Yu(1995) 讨论。关于这一现象的更多参考文献，参看 Prokhorov 等 (2002) 的综述论文。

习题

状态空间模型

15.1　写出图 15.3 的 Elman 简单递归网络状态空间模型的计算公式。

15.2　证实图 15.4 的递归多层感知器可以用状态空间模型

$$\mathbf{x}_{n+1} = \mathbf{f}(\mathbf{x}_n, \mathbf{u}_n)$$
$$\mathbf{y}_n = \mathbf{g}(\mathbf{x}_n, \mathbf{u}_n)$$

表示，其中 $\mathbf{u}_n$ 表示输入，$\mathbf{y}_n$ 表示输出，$\mathbf{x}_n$ 表示状态，$\mathbf{f}(\cdot,\cdot)$ 和 $\mathbf{g}(\cdot,\cdot)$ 表示向量值非线性函数。

15.3　一个动态系统是否可能是可控的但不可观测的，而且反之亦然？证实你的答案。

15.4　参考 15.4 节的局部可控性问题，证实

(a) 状态 $\mathbf{x}_{n+q}$ 是它过去值 $\mathbf{x}_n$ 和式(15.24)的输入向量 $\mathbf{u}_{q,n}$ 的嵌套非线性函数。

(b) $\mathbf{x}_{n+q}$ 对 $\mathbf{u}_{q,n}$ 的 Jacobi 矩阵在原点求值等于式(15.23)可控性矩阵 $\mathbf{M}_c$。

15.5　参照 15.4 节的局部可观测性问题，证明定义在式(15.30)中的观察向量 $\mathbf{y}_{q,n}$ 对状态 $\mathbf{x}_n$ 的 Jacobi 矩阵在原

点的求值等于式(15.28)的可观察矩阵 $\mathbf{M}_o$。

15.6　非线性动态系统的系统方程由

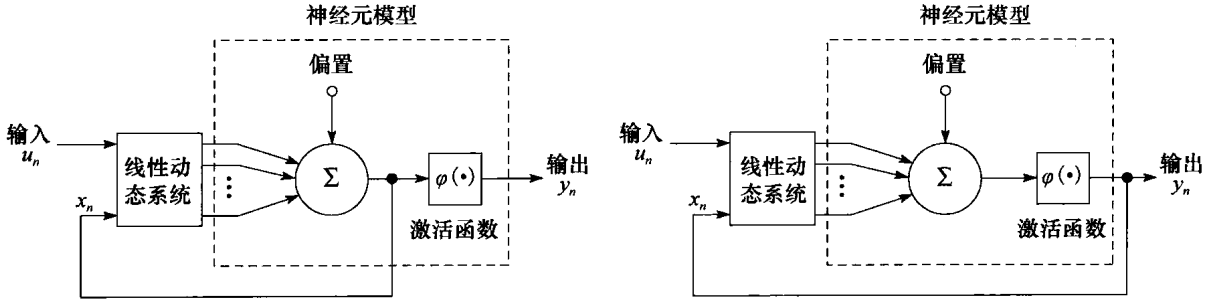
$$\mathbf{x}_{n+1} = \mathbf{f}(\mathbf{x}_n, \mathbf{u}_n)$$

描述，其中 $\mathbf{u}_n$ 是在时刻 n 的输入向量，$\mathbf{x}_n$ 是对应的系统状态。输入 $\mathbf{u}_n$ 在系统方程中以非加性的方式出现。在本题中，我们希望重新写过程方程，使输入 $\mathbf{u}_n$ 以加性的方式出现。这需写成

$$\mathbf{x}'_{n+1} = \mathbf{f}_{new}(\mathbf{x}'_n) + \mathbf{u}'_n$$

给出向量 $\mathbf{x}'_n$ 和 $\mathbf{u}'_n$ 以及函数 $\mathbf{f}_{new}(\cdot)$ 的定义公式。

15.7　图 P15.7 提出在神经元级上的使用局部反馈的递归网络模型的两个例子。在图 P15.7a 部分和图 P15.7b 部分显示的体系结构分别称为*局部激活反馈*和*局部输出反馈*（Tsoi and Back，1994）。对这两个递归网络的体系结构，写出状态空间模型公式。评价它们的可控性和可观察性。



a）局部激活反馈结构　　b）局部输出反馈结构

图　P15.7

有外部输入的非线性自回归（NARX）模型

15.8　考虑图 P15.8 的 NARX 网络，如下：

(a) 构造等价于这个单输入单输出递归网络的等价状态空间模型。

(b) 当图 P15.8 被扩展到包含两个输入和两个输出时重复（a）部分的习题。

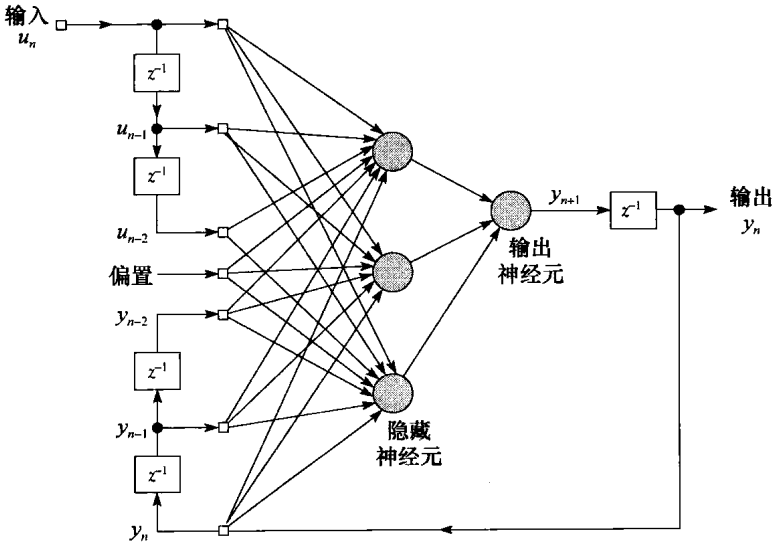


图 P15.8　具有 $q=3$ 个隐藏神经元的 NARX 网络

15.9　建立对应于图 P15.9 中的完全递归网络的 NARX。

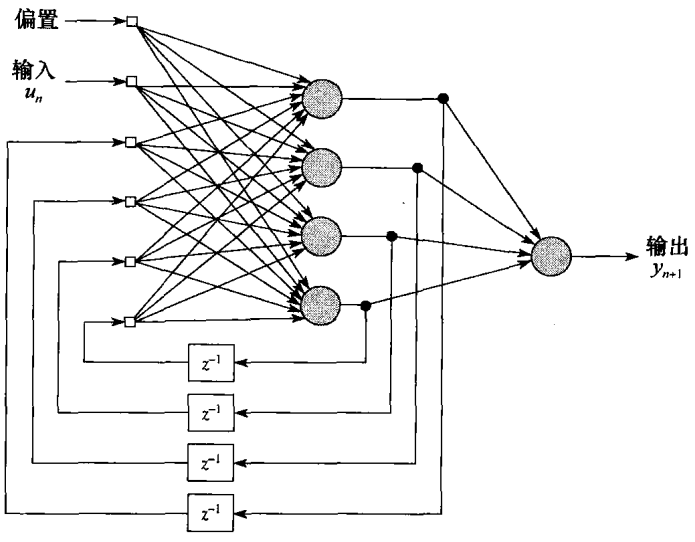


图　P15.9

15.10　任何状态空间模型可以表达成 NARX 模型。反过来的结果如何？任何的 NRAX 模型是否都可以表达成 15.2 节形式的状态空间模型？说明你的结论的理由。

通过时间的反向传播

15.11　展开图 15.3 的状态空间模型的时序行为。

15.12　截断的 BPTT(h) 算法可以看作是分回合的 BPTT 算法的近似。可以通过将分回合 BPTT 算法的一些方面包括进 BPTT(h) 来提高这个近似程度。特别是可以让网络在执行下一个 BPTT 计算前通过 h'个附加步，这里 $h'<h$。通过时间的反向传播的混合形式的重要特征是下一个后向传播在时间步 $n+h'$之后才执行。在此期间，网络过去输入值、网络状态和期望的响应都存储在一个缓冲区里面，但并不对它们进行处理。在这个混合型的算法中给出神经元 j 的局部梯度的公式。

实时递归学习算法

15.13　教师强制递归网络在训练过程中的动态以下面的方式描述

$$\xi_{i,n}=\begin{cases}u_{i,n}, & \text{如果 } i\in\mathcal{A}\\ d_{i,n}, & \text{如果 } i\in\mathcal{C}\\ y_{i,n}, & \text{如果 } i\in\mathcal{B}-\mathcal{C}\end{cases}$$

其中$\mathcal{A}$是记当 ξ_i 是一个外部输入时下标为 i 的集合。$\mathcal{B}$表示当 ξ_i 是一个神经元的输出时下标 i 的集合，C 表示可见的输出神经元的集合。

(a) 证明对这个格式，偏导数 $\partial y_{j,n+1}/\partial w_{kl,n}$由下式给出

$$\frac{\partial y_{j,n+1}}{\partial w_{kl,n}}=\varphi'(v_{j,n})\Big(\sum_{i\in\mathcal{B}-\mathcal{C}}w_{ji,n}\Big(\frac{\partial y_{i,n}}{\partial w_{kl,n}}\Big)+\delta_{kj}\xi_{l,n}\Big)$$

(b) 对于教师强制递归网络推导训练算法。

非线性逐次状态估计器

15.14　描述 DEKF 算法如何训练图 15.3 所示的简单递归网络。对于这个训练也可用 BPTT 算法。

15.15　表 15.2 给出 EKF 算法用于 RMLP 监督训练的总结。利用第 14 章描述的方根滤波理论来构造这一算法的方根修正。

15.16　在第 14 章描述了取样-重要性-再取样（SIR）粒子滤波器。这一滤波器是无导数的；因此可以尝试建议用它来作为递归多层感知器监督训练 EKF 算法的替代。讨论这一方法可能的困难。

计算机实验

15.17　在这一习题中，我们继续在第 6 章的习题 6.25 中关于支持向量机的计算机实验。我们具体考虑图 P6.25 的紧握起的多圆盘结构的困难模式分类实验，为了表示的方便我们将之复制在这里作为图

P15.17。然而这一次，我们根据 15.10 节描述的路线来学习基于扩展卡尔曼滤波器算法的多层感知器的监督训练。

对于多层感知器，利用下面的结构：

- 两个隐藏层，在第一个隐藏层中有 4 个神经元，在第二个隐藏层中有 3 个神经元；对所有的隐藏层神经元都采用 $\varphi(v)=\tanh(v)$ 的激活函数。
- 线性输出层。

为了实现模式分类，生成 100 个回合，每个回合包含 200 个随机分布的训练样本，对图 P15.17 的两个区域具有相同大小的测试数据。做如下事情：

1. 对于变化的回合数，构造由 EKF 算法计算的决策边界以决定“最佳”分类性能。
2. 对被考虑认为是“最佳”的分类性能，决定误分类误差。

最后，比较你用 EKF 算法得到的结果和在习题 6.25 中用支持向量机获得的结果。

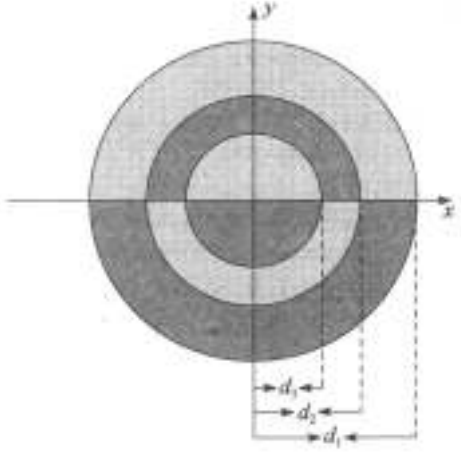


图 P15.17　三个圆的直径：$d_1=3$，$d_2=6$，$d_3=9$

参考文献

Aarts, E., and J. Korst, 1989. *Simulated Annealing and Boltzmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing,* New York: Wiley.

Abarbanel, H.D.I., 1996. *Analysis of Observed Chaotic Data,* New York: Springer-Verlag.

Abraham, R., J.E. Marsden, and T. Ratiu, 1988. *Manifolds, Tensor Analysis, and Applications,* 2d ed., New York: Springer-Verlag.

Abraham, R.H., and C.D. Shaw, 1992. *Dynamics of the Geometry of Behavior,* Reading, MA: Addison-Wesley.

Abramowitz, M., and I.A. Stegun, 1965. *Handbook of Mathematical Functions with Formulas, Graphs and Mathematical Tables,* New York: Dover Publications.

Abu-Mostafa, Y.S., 1995. "Hints," *Neural computation,* vol. 7, pp. 639–671.

Ackley, D.H., G.E. Hinton, and T.J. Sejnowski, 1985. "A Learning Algorithm for Boltzmann Machines," *Cognitive Science,* vol. 9, pp. 147–169.

Adetona, O., E. Garcia, and L.H. Keel, 2000. "A new method for control of discrete nonlinear dynamic systems using neural networks," *IEEE Trans. Neural Networks,* vol. 11, pp. 102–112.

Aiyer, S.V.B., N. Niranjan, and F. Fallside, 1990. "A theoretical investigation into the performance of the Hopfield model," *IEEE Transactions on Neural Networks,* vol. 15, pp. 204–215.

Aizerman, M.A., E.M. Braverman, and L.I. Rozonoer, 1964a. "Theoretical foundations of the potential function method in pattern recognition learning," *Automation and Remote Control,* vol. 25, pp. 821–837.

Aizerman, M. A., E.M. Braverman, and L.I. Rozonoer, 1964b. "The probability problem of pattern recognition learning and the method of potential functions," *Automation and Remote Control,* vol. 25, pp. 1175–1193.

Alspach, D.L. and H.W. Sorenson, 1972. "Nonlinear Bayesian estimation using Gaussian sum approximations," *IEEE Trans. Automatic Control,* vol. 17, pp. 439–448.

Aleksander, I., and H. Morton, 1990, *An Introduction to Neural Computing,* London: Chapman and Hall.

Amari, S., 1998. "Natural gradient works efficiently in learning." *Neural Computation,* vol. 10, pp. 251–276.

Amari, S., 1993. "A universal theorem on learning curves," *Neural Networks,* vol. 6, pp. 161–166.

Amari, S., 1990. "Mathematical foundations of neurocomputing," *Proceedings of the IEEE,* vol. 78, pp. 1443–1463.

Amari, S., 1987. "Differential geometry of a parametric family of invertible systems—Riemanian metric, dual affine connections and divergence," *Mathematical Systems Theory,* vol. 20, pp. 53–82.

Amari, S., 1985. *Differential-Geometrical Methods in Statistics,* New York: Springer-Verlag.

Amari, S., 1983. "Field theory of self-organizing neural nets," *IEEE Transactions on Systems, Man, and Cybernetics* vol. SMC-13, pp. 741–748.

Amari, S., 1980. "Topographic organization of nerve fields," *Bulletin of Mathematical Biology,* vol. 42, pp. 339–364.

Amari, S., 1977a. "Neural theory of association and concept-formation," *Biological Cybernetics,* vol. 26, pp. 175–185.

Amari, S., 1977b. "Dynamics of pattern formation in lateral-inhibition type neural fields," *Biological Cybernetics,* vol. 27, pp. 77–87.

Amari, S., 1972. "Characteristics of random nets of analog neuron-like elements," *IEEE Transactions on Systems, Man, and Cybernetics,* vol. SMC-2, pp. 643–657.

Amari, S., 1967. "A theory of adaptive pattern classifiers," *IEEE Trans. Electronic Computers,* vol. EC-16, pp. 299–307.

Amari, S., and M.A. Arbib, 1977. "Competition and cooperation in neural nets," in J. Metzler, ed., *Systems Neuroscience,* pp. 119–165, New York: Academic Press.

Amari, S., and J.-F. Cardoso, 1997. "Blind source separation—Semiparametric statistical approach," *IEEE Transactions on Signal Processing,* vol. 45, pp. 2692–2700.

Amari, S.,T.-P. Chen, and A. Cichoki, 1997. "Stability analysis of learning algorithms for blind source separation," *Neural Networks,* vol. 10, pp. 1345–1351.

Amari, S., A. Cichoki, and H.H. Yang, 1996. "A new learning algorithm for blind signal separation." *Advances in Neural Information Processing Systems,* vol. 8, pp. 757–763, Cambridge, MA: MIT Press.

Amari, S., and K. Maginu, 1988. "Statistical neurodynamics of associative memory," *Neural Networks,* vol. 1, pp. 63–73.

Amari, S., K. Yoshida, and K.-I. Kanatani, 1977. "A mathematical foundation for statistical neurodynamics," *SIAM Journal of Applied Mathematics,* vol. 33, pp. 95–126.

Ambros-Ingerson, J., R. Granger, and G. Lynch, 1990. "Simulation of paleo-cortex performs hierarchical clustering," *Science,* vol. 247, pp. 1344–1348.

Amit, D.J., 1989. *Modeling Brain Function: The World of Attractor Neural Networks,* New York: Cambridge University Press.

Anastasio T.J., 2003. "Vestibulo-ocular reflex," In M.A. Arbib, ed., *The Handbook of Brain Theory and Neural Networks,* 2d ed., pp. 1192–1196, Cambridge, MA: MIT Press.

Anastasio, T.J., 1993. "Modeling vestibulo-ocular reflex dynamics: From classical analysis to neural networks," in F. Eeckman, ed., *Neural Systems: Analysis and Modeling,* pp. 407–430, Norwell, MA: Kluwer.

Anderson, B.D.O., and J.B. Moore, 1971. *Linear Optimal Control,* Englewood Cliffs, NJ: Prentice-Hall.

Anderson, J.A., 1995. *Introduction to Neural Networks,* Cambridge, MA: MIT Press.

Anderson, J.A., 1993. "The BSB model: A simple nonlinear autoassociative neural network," in *Associative Neural Memories* (M. Hassoun, ed.) pp. 77–103, Oxford: Oxford University Press.

Anderson, J.A., and E. Rosenfeld, eds., 1988. *Neurocomputing: Foundations of Research,* Cambridge, MA: MIT Press.

Anderson, J.A., A. Pellionisz, and E. Rosenfeld, eds., 1990a. *Neurocomputing 2: Directions for Research,* Cambridge, MA: MIT Press.

Anderson, J.A., J.W. Silverstein, S.A. Ritz, and R.S. Jones, 1977. "Distinctive features, categorical perception, and probability learning: Some applications of a neural model," *Psychological Review,* vol. 84, pp. 413–451.

Anderson, J.A., and J.P. Sutton, 1995. "A network of networks: Computation and neurobiology," *World Congress on Neural Networks,* vol. I, pp. 561–568.

Anderson, T.W., 1984. *An Introduction to Multivariate Statistical Analysis,* 2d ed., New York: Wiley.

Ansari, N., and E. Hou, 1997. *Computational Intelligence for Optimization,* Norwell, MA: Kluwer.

Arasaratnam, I., and S. Haykin, 2009. "Cubature Kalman filters," *IEEE Trans. Automatic Control,* vol. 54, June.

Arasaratnam, I., S. Haykin, and R.J. Elliott, 2007. "Discrete-time nonlinear-filtering algorithms using Gauss–Hermite quadrature," *Proc. IEEE,* vol. 95, pp. 953–977.

Arbib, M.A., 1989. *The Metaphorical Brain,* 2d ed., New York: Wiley.

Arbib, M.A., 1987. *Brains, Machines, and Mathematics,* 2d ed., New York: Springer-Verlag.

Arbib, M.A., 2003. *The Handbook of Brain Theory and Neural Networks,* 2d ed., Cambridge, MA: MIT Press.

Arimoto, S., 1972. "An algorithm for calculating the capacity of an arbitrary memoryless channel," *IEEE Trans. Information Theory,* vol. IT-18, pp. 14–20.

Aronszajn, N., 1950. "Theory of reproducing kernels," *Trans. American Mathematical Society,* vol. 68, pp. 337–404.

Arrowsmith, D.K., and C.M. Place, 1990. *An Introduction to Dynamical Systems,* Cambridge, U.K.: Cambridge University Press.

Ash, R.E., 1965. *Information Theory,* New York: Wiley.

Ashby, W.R., 1960. *Design for a Brain,* 2d ed., New York: Wiley.

Ashby, W.R., 1952. *Design for a Brain,* New York: Wiley.

Aspray, W., and A. Burks, 1986. *Papers of John von Neumann on Computing and Computer Theory,* Charles Babbage Institute Reprint Series for the History of Computing, vol. 12. Cambridge, MA: MIT Press.

Atick, J.J., 1992. "Could information theory provide an ecological theory of sensory processing?" *Network: Computation in Neural Systems,* vol. 3, pp. 213–251.

Atick, J.J., and A.N. Redlich, 1990. "Towards a theory of early visual processing," *Neural Computation,* vol. 2, pp. 308–320.

Atiya, A.F., 1987, "Learning on a general network," In *Neural Information Processing Systems,* D.Z. Anderson, ed., pp. 22–30, New York: American Institute of Physics.

Attneave, F., 1954. "Some informational aspects of visual perception," *Psychological Review,* vol. 61, pp. 183–193.

Back, A.D., and A.C. Tsoi, 1991. "FIR and IIR synapses, a new neural network architecture for time series modeling," *Neural Computation,* vol. 3, pp. 375–385.

Bakir, G.H., T. Hofmann, B. Schölkopf, A.J. Smola, B. Taskar, and S.V.N. Vishwanathan, eds., 2007. *Predicting Structured Data,* Cambridge, MA: MIT Press.

Barlow, H.B., 1989. "Unsupervised learning," *Neural Computation,* vol. 1, pp. 295–311.

Barlow, H.B., 1959. "Sensory mechanisms, the reduction of redundancy, and intelligence," in *The Mechanisation of Thought Processes, National Physical Laboratory Symposium No. 10,* Her Majesty's Stationary Office, London.

Barlow, H., and P. Földiák, 1989. "Adaptation and decorrelation in the cortex," in *The Computing Neuron,* R. Durbin, C. Miall, and G. Mitchison, eds., pp. 54–72. Reading, MA: Addison-Wesley.

Barnard, E., and D. Casasent, 1991. "Invariance and neural nets," *IEEE Transactions on Neural Networks,* vol. 2, pp. 498–508.

Barron, A.R., 1993. "Universal approximation bounds for superpositions of a sigmoidal function," *IEEE Transactions on Information Theory,* vol. 39, pp. 930–945.

Barron, A.R., 1992. "Neural net approximation," in *Proceedings of the Seventh Yale Workshop on Adaptive and Learning Systems,* pp. 69–72, New Haven, CT: Yale University.

Barto, A.G., S.J. Bradtke, and S. Singh, 1995. "Learning to act using real-time dynamic programming," *Artificial Intelligence,* vol. 72, pp. 81–138.

Barto, A.G., R.S. Sutton, and C.W. Anderson, 1983. "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE Transactions on Systems, Man, and Cybernetics,* vol. SMC-13, pp. 834–846.

Battiti, R., 1992. "First- and second-order methods for learning: Between steepest descent and Newton's method," *Neural Computation,* vol. 4, pp. 141–166.

Bauer, H.-U., and K.R. Pawelzik, 1992. "Quantifying the neighborhood preservation of self-organizing feature maps," *IEEE Transactions on Neural Networks,* vol. 3, pp. 570–579.

Bauer, H.-U., R. Der, and M. Hermman, 1996. "Controlling the magnification factor of self-organizing feature maps," *Neural Computation,* vol. 8, pp. 757–771.

Baum, E.B., and F. Wilczek, 1988. "Supervised learning of probability distributions by neural networks," in D.Z. Anderson, ed., pp. 52–61, New York: American Institute of Physics.

Beaufays, F., and E.A. Wan, 1994. "Relating real-time backpropagation and backpropagation-through-time: An application of flow graph interreciprocity," *Neural Computation,* vol. 6, pp. 296–306.

Becker, S., 1996. "Mutual information maximization: models of cortical self-organization," *Network: Computation in Neural Systems,* vol. 7, pp. 7–31.

Becker, S., 1991. "Unsupervised learning procedures for neural networks," *International Journal of Neural Systems,* vol. 2, pp. 17–33.

Becker, S., and G.E. Hinton, 1992. "A self-organizing neural network that discovers surfaces in random-dot stereograms," *Nature (London),* vol. 355, pp. 161–163.

Becker, S., and Y. LeCun, 1989. "Improving the convergence of back-propagation learning with second order methods," In D. Touretzky, G.E. Hinton, and T.J. Sejnowski, eds., *Proceedings of the 1988 Connectionist Models Summer School*, pp. 29–37, San Fransisco: Morgan Kaufmann.

Beckerman, M., 1997. *Adaptive Cooperative Systems,* New York: Wiley (Interscience).

Belkin, M., 2003. *Problems of Learning on Manifolds,* Ph.D. thesis, The University of Chicago.

Belkin, M., P. Niyogi, and V. Sindhwani, 2006. "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *J. Machine Learning Research,* vol. 7, pp. 2399–2434.

Bell, A.J. and T.J. Sejnowski, 1997. "The 'Independent Components' of natural scenes are edge filters," *Vision Research,* vol. 37, pp. 3327–3338.

Bell, A.J., and T.J. Sejnowski, 1995. "An information-maximization approach to blind separation and blind deconvolution," *Neural Computation,* vol. 6, pp. 1129–1159.

Bellman, R., 1961. *Adaptive Control Processes: A Guided Tour,* Princeton, NJ: Princeton University Press.

Bellman, R., 1957. *Dynamic Programming,* Princeton, NJ: Princeton University Press.

Bellman, R., and S.E. Dreyfus, 1962. *Applied Dynamic Programming,* Princeton, NJ: Princeton University Press.

Bengio, Y., and Y. LeCun, 2007. "Scaling learning algorithms toward AI," in L. Bottou, O. Chapelle D. DeCosta, and J. Weston, eds., *Large-Scale Kernel Machines*, pp. 321–359, Cambridge, MA MIT Press.

Bengio, Y., P. Simard, and P. Frasconi, 1994. "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks,* vol. 5, pp. 157–166.

Bengtsson, T., P. Bickel, and B. Li, 2008. "Curse-of-dimensionality revisited: Collapse of the particle filter in very large scale systems," *IMS Collections, Probability and Statistics: Essays in Honor of David A. Freedman,* vol. 2, pp. 316–334.

Benveniste, A., M. Métivier, and P. Priouret, 1987. *Adaptive Algorithms and Stochastic Approximation,* New York: Springer-Verlag.

Bertsekas, D.P., 2007. *Dynamic Programming and Optimal Control,* vol. II, 3d ed., Nashua, NH: Athena Scientific.

Bertsekas, D.P., 2005. *Dynamic Programming and Optimal Control,* vol. I, 3d ed., Nashua, NH: Athena Scientific.

Bertsekas, D.P., A. Nedich, and V.S. Borkar, 2004. "Improved temporal difference methods with linear function approximation," in J. Si, A.G. Barto, W.B. Powell, and D. Wunsch II, eds., *Handbook of Learning and Approximate Dynamic Programming*, pp. 235–259, Hobken, NJ: Wiley-Interscience.

Bertsekas, D.P., with A. Nedich and A.E. Ozdaglar, 2003. *Convex Analysis and Optimization,* Nashua, NH: Athena Scientific.

Bertsekas, D.P., and J.N. Tsitsiklis, 2002. *Introduction to Probability,* Nashua, NH: Athena Scientific.

Bertsekas, D.P., 1995. *Nonlinear Programming.* Belmont, MA: Athenas Scientific.

Bertsekas, D.P., and J.N. Tsitsiklis, 1996. *Neuro-Dynamic Programming,* Belmont, MA: Athena Scientific.

Bierman, G.J., and C.L. Thornton, 1977. "Numerical comparison of Kalman filter algorithms: Orbit determination case study," *Automatica*, vol. 13, pp. 23–35.

Bishop, C.M., 1995. *Neural Networks for Pattern Recognition.* Oxford, U.K.: Clarendon Press.

Blahut, R., 1972. "Computation of channel capacity and rate distortion functions," *IEEE Trans. Information Theory*, vol. IT-18, pp. 460–473.

Bobrowski, O., R. Meir, and Y.C. Eldor, 2007. "Bayesian filtering in spiking neural networks: Noise, adaptation, and multisensory integration," *Neural Information Processing Systems (NIPS) Conference*, Vancouver: December.

Bodenhausen, U., and A. Waibel, 1991. "The tempo 2 algorithm: Adjusting time-delays by supervised learning," *Advances in Neural Information Processing Systems,* vol. 3, pp. 155–161, San Mateo, CA: Morgan Kaufmann.

Boltzmann, L., 1872. "Weitere studien über das Wärmegleichgewicht unter gasmolekülen," *Sitzungsberichte der Mathematisch-Naturwissenschaftlichen Classe der Kaiserlichen Akademie der Wissenschaften,* vol. 66, pp. 275–370.

Boothby, W.M., 1986. *An Introduction to Differentiable Manifolds and Riemannian Geometry,* 2d ed., Orlando, FL: Academic Press.

Boser, B., I. Guyon, and V.N. Vapnik, 1992. "A training algorithm for optimal margin classifiers," *Fifth Annual Workshop on Computational Learning Theory,* pp. 144–152. San Mateo, CA: Morgan Kaufmann.

Bottou, L., 2007. *Learning Using Large Datasets*, NIPS 2007 Conference Tutorial Notes, Neural Information Processing Systems (NIPS) Conference, Vancouver: December.

Bottou, L., and C. Lin, 2007. "Support vector machine solvers," in L. Bottou, O. Chapelle, D. DeCosta, and J. Weston, eds., *Large-Scale Kernel Machines,* pp. 1–27, Cambridge, MA: MIT Press.

Boyan, J.A., 2002. "Technical update: Least-squares temporal difference learning," *Machine Learning,* vol. 49, pp. 1–15.

Boyd, S., and L. Vandenberghe, 2004. *Convex Optimization.* Cambridge, U.K., and New York: Cambridge University Press.

Bradtke, S.J., and A.G. Barto, 1996. "Linear least-squares algorithms for temporal difference learning," *Machine Learning,* vol. 22, pp. 33–57.

Braitenberg, V., 1990. "Reading the structure of brains," *Network: Computation in Neural Systems,* vol. 1, pp. 1–12.

Braitenberg, V., 1986. "Two views of the cerebral cortex," in *Brain Theory,* G. Palm and A. Aertsen, eds., pp. 81–96. New York: Springer-Verlag.

Braitenberg, V., 1984. *Vehicles: Experiments in Synthetic Psychology,* Cambridge, MA: MIT Press.

Braitenberg, V., 1977. *On the Texture of Brains,* New York: Springer-Verlag.

Braitenberg, V., 1967. "Is the cerebella cortex a biological clock in the millisecond range?" in *The Cerebellum. Progress in Brain Research,* C.A. Fox and R.S. Snider, eds., vol. 25 pp. 334–346, Amsterdam: Elsevier.

Bregman, A.S., 1990. *Auditory Scene Analysis: The Perceptual Organization of Sound,* Cambridge, MA: MIT Press.

Brodal, A., 1981. *Neurological Anatomy in Relation to Clinical Medicine,* 3d ed., New York: Oxford University Press.

Brogan, W.L., 1985. *Modern Control Theory,* 2d ed., Englewood Cliffs, NJ: Prentice-Hall.

Broomhead, D.S., and D. Lowe, 1988. "Multivariable functional interpolation and adaptive networks," *Complex Systems,* vol. 2, pp. 321–355.

Brown, T.H., E.W. Kairiss, and C.L. Keenan, 1990. "Hebbian synapses: Biophysical mechanisms and algorithms," *Annual Review of Neuroscience,* vol. 13, pp. 475–511.

Bruck, J., 1990. "On the convergence properties of the Hopfield model," *Proceedings of the IEEE,* vol. 78, pp. 1579–1585.

Bryson, A.E., Jr., and Y.C. Ho, 1969. *Applied Optimal Control,* Blaisdell. (Revised printing, 1975, Hemisphere Publishing, Washington, DC).

Cacoullos, T., 1966. "Estimation of a multivariate density," *Annals of the Institute of Statistical Mathematics (Tokyo),* vol. 18, pp. 179–189.

Caianiello, E.R., 1961. "Outline of a theory of thought-processes and thinking machines," *Journal of Theoretical Biology,* vol. 1, pp. 204–235.

Cameron, S.H., 1960. Tech. Report 60–600, *Proceedings of the Bionics Symposium,* pp. 197–212, Wright Air Development Division, Dayton, Ohio.

Cappé, O., S.J. Godsill, and E. Moulines, 2007. "An overview of existing methods and recent advances in sequential Monte Carlo," *Proc. IEEE,* vol. 95, pp. 899–924.

Cappé, O., E. Moulines, and T. Rydén, 2005. *Inference in Hidden Markov Models,* New York and London: Springer.

Cardoso, J.F., 2003. "Dependence, correlation and Gaussianity in independent component analysis," *J. Machine Learning Research,* vol. 4, pp. 1177–1203.

Cardoso, J.F., 2001. "The three easy routes to independent component analysis: Contrasts and geometry," *Proceedings of 3rd International Conference on Independent Component Analysis and Blind Source Separation,* San Diego, December.

Cardoso, J.-F., 1998. "Blind signal separation: A review," *Proceedings of the IEEE,* vol. 86, pp. 2009–2025.

Cardoso, J-F., 1997. "Infomax and maximum likelihood for blind source separation," *IEEE Signal Processing Letters,* vol. 4, pp. 112–114.

Cardoso, J.-F., and B. Laheld, 1996. "Equivariant adaptive source separation," *IEEE Transactions on Signal Processing,* vol. 44, pp. 3017–3030.

Carpenter, G.A., M.A. Cohen, and S. Grossberg, 1987. Technical comments on "Computing with neural networks," *Science,* vol. 235, pp. 1226–1227.

Černy, V., 1985. "Thermodynamic approach to the travelling salesman problem," *Journal of Optimization Theory and Applications,* vol. 45, pp. 41–51.

Changeux, J.P., and A. Danchin, 1976. "Selective stabilization of developing synapses as a mechanism for the specification of neural networks," *Nature,* vol. 264, pp. 705–712.

Chapelle, O., B. Schölkopf, and A. Zien, 2006. *Semi-Supervised Learning,* Cambridge, MA: MIT Press.

Charlin, L., P. Poupart, and R. Shoida, 2007. "Automated hierarchy discovery for planning in partially observable environments," *Advances in Neural Information Processing Systems,* vol. 19, pp. 225–232.

Chatterjee, C., V.P. Roychowdhhury, and E.K.P. Chong, 1998. "On relative convergence properties of principal component algorithms," *IEEE Transactions on Neural Networks,* vol. 9, pp. 319–329.

Chechik, G., A. Globerson, N. Tishby, and Y. Weiss, 2004. "Information bottleneck for Gaussian variables," *Advances in Neural Information Processing Systems,* vol. 16, pp. 1213–1220.

Chen, S., S. Billings, and P. Grant, 1990. "Non-linear system identification using neural networks," *International Journal of Control,* vol. 51, pp. 1191–1214.

Chen, Z., S. Haykin, J.J. Eggermont, and S. Becker, 2007. *Correlative Learning: A Basis for Brain and Adaptive Systems,* New York: Wiley-Interscience.

Cherkassky, V., and F. Mulier, 1998. *Learning from Data: Concepts, Theory and Methods,* New York: Wiley.

Cherry, E.G., 1953. "Some experiments on the recognition of speech, with one and with two ears," *Journal of the Acoustical Society of America,* vol. 25, pp. 975–979.

Cherry, E.C., and W.K. Taylor, 1954. "Some further experiments upon the recognition of speech, with one and with two ears," *Journal of Acoustical Society of America,* vol. 26, pp. 554–559.

Chester, D.L., 1990. "Why two hidden layers are better than one," *International Joint Conference on Neural Networks,* vol. I, pp. 265–268, Washington, D.C.

Chigirev, D. and W. Bialek, 2004. "Optimal manifold representation of data: An information-theoretic approach," *Advances in Neural Information Processing Systems,* vol. 16, pp. 161–168.

Choi, H., and R.G. Baraniuk, 1999. "Multiple basis wavelet denoising using Besov projections," *Proceedings of IEEE International Conference on Image Processing,* pp. 595–599.

Choi, S., A. Cichoki, H.M. Park, and S.Y. Lee, 2005. "Blind source separation and independent component analysis: A review," *Neural Information Processing-Letters and Reviews,* vol. 6, pp. 1–57.

Chung, R.K., 1997. *Spectral Graph Theory,* Regional Conference Series in Mathematics, Number 92, Providence, RI: American Mathematical Society.

Churchland, P.S., and T.J. Sejnowski, 1992. *The Computational Brain,* Cambridge, MA: MIT Press.

Cichocki, A., and S. Amari, 2002. *Adaptive Blind Signal and Image Processing: Learning Algorithms and Applications,* Chichester, NY: Wiley-Interscience.

Cleeremans, A., D. Servan-Schreiber, and J.L. McClelland, 1989. "Finite state automata and simple recurrent networks," *Neural Computation,* vol. 1, pp. 372–381.

Cohen, L., 2005. "The history of noise [on the 100th anniversary of its birth]," *IEEE Signal Processing Magazine,* vol. 22, issue 6, pp. 20–45, November.

Cohen, M. A., and S. Grossberg, 1983. "Absolute stability of global pattern formation and parallel memory storage by competitive neural networks," *IEEE Transactions on Systems, Man, and Cybernetics,* vol. SMC-13, pp. 815–826.

Collins, M., S. Dasgupta, and R.E. Schapire, 2002. "A generation of principal components analysis to the exponential family," *Advances in Neural Information Processing Systems,* vol. 14-1, pp. 617–624, Cambridge, MA: MIT Press.

Comon, P., 1994. "Independent component analysis: A new concept?" *Signal Processing,* vol. 36, pp. 287–314.

Comon, P., 1991. "Independent component analysis," *Proceedings of International Signal Processing Workshop on Higher-order Statistics,* pp. 111–120, Chamrousse, France.

Cook, A.S., 1971. "The complexity of theorem-proving procedures," *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing,* pp. 151–158, New York.

Cook, P.A., 1986. *Nonlinear Dynamical Systems,* London: Prentice-Hall International.

Cools, R., 2002. "Advances in multidimensional integration," *J. Comput. and Applied Math.,* vol. 149, pp. 1–12.

Cools, R., 1997. "Computing cubature formulas: The science behind the art," *Acta Numerica,* vol. 6, pp. 1–54, Cambridge, U.K.: Cambridge University Press.

Cormen, T.H., C.E. Leiserson, and R.R. Rivest, 1990. *Introduction to Algorithms.* Cambridge, MA: MIT Press.

Cortes, C, and V. Vapnik, 1995. "Support vector networks," *Machine Learning,* vol. 20, pp. 273–297.

Cottrell, M., J.C. Fort, and G. Pagés, 1997. "Theoretical aspects of the SOM algorithm," *Proceedings of the Workshop on Self-Organizing Maps,* Espoo, Finland.

Courant, R., and D. Hilbert, 1970. *Methods of Mathematical Physics,* vol. I and II, New York: Wiley Interscience.

Cover, T.M., and J.A. Thomas, 2006. *Elements of Information Theory,* 2d ed., Hoboken, NJ: Wiley-Interscience.

Cover, T.M., 1968. "Capacity problems for linear machines," In L. Kanal, ed., *Pattern Recognition,* pp. 283–289, Washington, DC: Thompson Book Co.

Cover, T.M., 1965. "Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition," *IEEE Transactions on Electronic Computers,* vol. EC-14, pp. 326–334.

Cover, T.M., and P.E. Hart, 1967. "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory,* vol. IT-13, pp. 21–27.

Cowan, J.D., 1968. "Statistical mechanics of nervous nets," in *Neural Networks,* E.R. Caianiello, ed., pp. 181–188, Berlin: Springer-Verlag.

Cragg, B.G., and H.N.V. Tamperley, 1954. "The organization of neurons: A cooperative analogy," *EEG Clinical Neurophysiology,* vol. 6, pp. 85–92.

Craik, K.J.W., 1943. *The Nature of Explanation,* Cambridge, U.K.: Cambridge University Press.

Craven, P., and G. Wahba, 1979. "Smoothing noisy data with spline functions: Estimating the correct degree of smoothing by the method of generalized cross-validation," *Numerische Mathematik.,* vol. 31, pp. 377–403.

Crisan, D., and A. Doucet, 2002. "A survey of convergence results on particle filtering methods for practitioners," *IEEE Trans. Signal Processing,* vol. 50, pp. 736–746.

Crites, R.H., and A.G. Barto, 1996. "Improving elevator performance using reinforcement learning," *Advances in Neural Information Processing Systems,* vol. 8, pp. 1017–1023, Cambridge, MA: MIT Press.

Csiszár, I., and G. Tusnády, 1984. "Information geometry and alternating minimization procedures," *Statistics and Decisions,* Supplement Issue, vol. I, 205–237.

Cucker, F., and S. Smale, 2001. "On the mathematical foundations of learning," *Bulletin (New Series) of the American Mathematical Society,* vol. 39, pp. 1–49.

Cybenko, G., 1995. "Q-learning: A tutorial and extensions." Presented at *Mathematics of Artificial Neural Networks,* Oxford University, Oxford, U.K., July 1995.

Cybenko, G, 1989. "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals, and Systems,* vol. 2, pp. 303–314.

Cybenko, G., 1988. "Approximation by superpositions of a sigmoidal function," Urbana, IL.: University of Illinois.

Dan, Y., J.J. Atick, and R.C. Reid, 1996. "Efficient coding of natural scenes in the lateral geniculate nucleus: Experimental test of a computational theory," *J. of Neuroscience,* vol. 16, pp. 3351–3362.

Darken, C., and J. Moody, 1992. "Towards faster stochastic gradient search," *Advances in Neural Information Processing Systems,* vol. 4, pp. 1009–1016, San Mateo, CA: Morgan Kaufmann.

Darken, C., and J. Moody, 1991. "Note on learning rate schedules for stochastic optimization," in R.P. Lippmann, J.E. Moody, and D.S. Touretzky, *Advances in Neural Information Processing Systems,* pp. 832–838, San Mateo, CA: Morgan Kaufmann.

Darmois, G., 1953. "Analyse générale des liaisons stochastiques," *Rev. Inst. Internal. Stat.,* vol. 21, pp. 2–8.

Daubechies, I., ed., 1993. *Different Perspectives on Wavelets,* American Mathematical Society Short Course, San Antonio, January 11–12.

Daubechies, I., 1992. *Ten Lectures on Wavelets,* SIAM.

Daubechies, I., 1990. "The wavelet transform, time-frequency," *IEEE Transactions on Information Theory,* vol. IT-36, pp. 961–1005.

Daum, F. and J. Huang, 2003. "Curse of dimensionality and particle filters," *Proceedings, IEEE Aerospace Conference*, vol. 4, pp. 1979–1993, March.

Davis, P.J., 1963. *Interpolation and Approximation,* New York: Blaisdell.

Debnath, L., and P. Mikusiński, 1990. *Introduction to Hilbert Spaces with Applications,* New York: Academic Press.

de Figueiredo, R.J.P., and G. Chen, 1993. *Nonlinear Feedback Control Systems,* New York: Academic Press.

Dempster, A.P., N.M. Laird, and D.B. Rubin, 1977. "Maximum likelihood from incomplete data via the EM algorithm," (with discussion), *Journal of the Royal Statistical Society.,* B, vol. 39, pp. 1–38.

Denardo, E.V., 1967. "Contraction mappings in the theory underlying dynamic programming," *SIAM,* Review, vol. 9, pp. 165–177.

DeSieno, D., 1988. "Adding a conscience to competitive learning," *IEEE International Conference on Neural Networks,* vol. I, pp. 117–124, San Diego.

deSilva, C.J.S., and Y. Attikiouzel, 1992. "Hopfield networks as discrete dynamical systems," *International Joint Conference on Neural Networks,* vol. III, pp. 115–120, Baltimore.

DeVito, E., L. Rosasco, A. Caponnetto, U. DeGiovannini, and F. Odone, 2005. "Learning from examples as an inverse problem." *J. Machine Learning Research*, vol. 6, pp. 883–904.

deVries, B., and J.C. Principe, 1992. "The gamma model—A new neural model for temporal processing," *Neural Networks,* vol. 5, pp. 565–576.

Diamantaras, K.I., and S.Y. Kung, 1996. *Principal Component Neural Networks: Theory and Applications,* New York: Wiley.

Ding, C., 2004. *Spectral Clustering,* Tutorial Notes, ICML, Banff, Alberta, Canada, July.

Ding, C., and X. He, 2004. "K-means clustering via principal component analysis," *Proceedings of the Twenty-first International Conference on Machine Learning,* pp. 225–240, Banff, Alberta, Canada.

Diniz, P.S.R., 2002. *Adaptive Filtering: Algorithms and Practical Implementation,* 2d ed., Boston: Kluwer Academic Publishers.

Dong, D.W., and J.J. Atick, 1995. "Temporal decorrelation: A theory of lagged and non-lagged responses in the lateral geniculate nucleus," *Network: Computation in Neural Systems,* vol. 6, pp. 159–178.

Dony, R.D., and S. Haykin, 1995. "Optimally adaptive transform coding," *IEEE Transactions on Image Processing,* vol. 4, pp. 1358–1370.

Dorny, C.N., 1975. *A Vector Space Approach to Models and Optimization,* New York: Wiley (Interscience).

Doucet, A., N. deFreitas, and N. Gordon, eds., 2001. *Sequential Monte Carlo Methods in Practice,* New York: Springer.

Doucet, A., S. Godsill, and C. Andrieu, 2000. "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing,* vol. 10, pp. 197–208.

Doya, K., S. Ishi, A. Pouget, and R.P.N. Rao, eds., 2007. *Bayesian Brain: Probabilistic Approaches to Neural Coding,* Cambridge, MA: MIT Press.

Drineas, P., and M.W. Mahoney, 2005. "On the Nyström method for approximating a Gram matrix for improved kernel-based learning," *J. Machine Learning Research,* vol. 6, pp. 2153–2175.

Duda, R.O., P.E. Hart, and D.G. Stork, 2001. *Pattern Classification,* 2d ed., New York: Wiley-Interscience.

Duda, R.O., and P.E. Hart, 1973. *Pattern Classification and Scene Analysis,* New York: Wiley.

Durbin, R., and G. Michison, 1990. "A dimension reduction framework for understanding cortical maps," *Nature,* vol. 343, pp. 644–647.

Durbin, R., C. Miall, and G. Mitchison, eds, 1989. *The Computing Neuron,* Reading, MA. Addison-Wesley.

Durdanovic, I., E. Cosatto, and H. Graf, 2007. "Large-scale Parallel SVM implementation". In L. Bottou, O. Chapelle, D. DeCosta, and J. Weston, editors, *Large-Scale Kernel Machines*, pp. 105–138, MIT Press.

Eggermont, J.J., 1990. *The Correlative Brain: Theory and Experiment in Neural Interaction,* New York: Springer-Verlag.

Elhilali, M., 2004. *Neural Basis and Computational Strategies for Auditory Processing,* Ph.D. thesis, University of Maryland.

Elliott, R.J., L. Aggoun, and J.B. Moore, 1995. *Hidden Markov Models: Estimation and Control,* New York: Springer-Verlag.

Elman, J.E., E.A. Bates, M.H. Johnson, A. Karmiloff-Smith, D. Parisi, and K. Plinket, 1996. *Rethinking Innateness: A Connectionist Perspective on Development,* Cambridge, MA: MIT Press.

Elman, J.L., 1990. "Finding structure in time," *Cognitive Science,* vol. 14, pp. 179–211.

Erwin, E., K. Obermayer, and K. Schulten, 1995. "Models of orientation and ocular dominance columns in the visual cortex: A critical comparison," *Neural Computation,* vol. 7, pp. 425–468.

Erwin, E., K. Obermayer, and K. Schulten, 1992a. "I: Self-organizing maps: Stationary states, metastability and convergence rate," *Biological Cybernetics,* vol. 67, pp. 35–45.

Erwin, E., K. Obermayer, and K. Schulten, 1992b. "II: Self-organizing maps: Ordering, convergence properties and energy functions," *Biological Cybernetics,* vol. 67, pp. 47–55.

Feldkamp, L.A., T.M. Feldkamp, and D.V. Prokhorov, 2001. "Neural network training with the nprKF," *Proceedings of the International Joint Conference on Neural Networks,* Washington, DC.

Feldkamp, L., and G., Puskorius, 1998. "A signal processing framework based on dynamic neural networks with application to problems in adaptation, filtering, and classification," *Proc. IEEE,* vol. 86, pp. 2259–2277.

Feldkamp, L.A., G.V. Puskorius, and P.C. Moore, 1997. "Adaptation from fixed weight networks," *Information Sciences,* vol. 98, pp. 217–235.

Feller, W., 1968. *An Introduction to Probability Theory and its Applications,* vol. 1, 3d ed., New York: John Wiley; 1st printing, 1950.

Feller, W., 1971. *An Introduction to Probability Theory and its Applications,* 3d ed., vol. II, New York: Wiley; 1st printing, 1967.

Field, D.J., 1994. "What is the goal of sensory coding?" *Neural computation,* vol. 6, pp. 559–601.

Fischler, M.A., and O. Firschein, 1987. *Intelligence: The Eye, The Brain, and The Computer,* Reading, MA: Addison-Wesley.

Fisher, R.A., 1925. "Theory of statistical estimation," *Proceedings of the Cambridge Philosophical Society,* vol. 22, pp. 700–725.

Fletcher, R., 1987. *Practical Methods of Optimization,* 2d ed., New York: Wiley.

Forgey, E., 1965. "Cluster analysis of multivariate data: Efficiency vs. interpretability of classification," *Biometrics,* vol. 21, p. 768 (abstract).

Földiak, P., 1989. "Adaptive network for optimal linear feature extractions," *IEEE International Joint Conference on Neural Networks,* vol. I, pp. 401–405, Washington, DC.

Forney, G.D., Jr., 1973. "The Viterbi algorithm," *Proceedings of the IEEE,* vol. 61, pp. 268–278.

Forte, J.C., and G. Pagés, 1996. "Convergence of stochastic algorithm: From the Kushner and Clark theorem to the Lyapunov functional," *Advances in Applied Probability,* vol. 28, pp. 1072–1094.

Forte, J.C., and G. Pagés, 1995. "On the a.s. convergence of the Kohonen algorithm with a general neighborhood function," *Annals of Applied Probability,* vol. 5, pp. 1177–1216.

Frasconi, P., and M. Gori, 1996. "Computational capabilities of local-feedback recurrent networks acting as finite-state machines," *IEEE Transactions on Neural Networks,* vol. 7, pp. 1521–1524.

Frasconi, P., M. Gori, and G. Soda, 1992. "Local feedback multilayered networks," *Neural Computation,* vol. 4, pp. 120–130.

Fraser, A.M., 1989. "Information and entropy in strange attractors," *IEEE Transactions on Information Theory,* vol. 35, pp. 245–262.

Freeman, W.J., 1975. *Mass Action in the Nervous System,* New York: Academic Press.

Friedman, J.H., 1995. "An overview of prediction learning and function approximation," In V. Cherkassky, J.H. Friedman, and H. Wechsler, eds., *From Statistics to Neural Networks: Theory and Pattern Recognition Applications,* New York: Springer-Verlag.

Fukunaga, K., 1990. *Statistical Pattern Recognition,* 2d ed., New York: Academic Press.

Fukushima, K., 1995. "Neocognitron: A model for visual pattern recognition," in M.A. Arbib, ed., *The Handbook of Brain Theory and Neural Networks,* Cambridge, MA: MIT Press.

Fukushima, K., 1980. "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics,* vol. 36,193–202.

Funahashi, K., 1989. "On the approximate realization of continuous mappings by neural networks," *Neural Networks,* vol. 2, pp. 183–192.

Fyfe, C., 2005. *Hebbian Learning and Negative Feedback Networks,* New York: Springer.

Gabor, D., 1954. "Communication theory and cybernetics," *IRE Transactions on Circuit Theory,* vol. CT-1, pp. 19–31.

Gabor, D., W.P.L. Wilby, and R. Woodcock, 1960. "A universal non-linear filter, predictor, and simulator which optimizes itself by a learning process," *Proceedings of the Institution of Electrical Engineers,* London, vol. 108, pp. 422–435.

Galland, C.C., 1993. "The limitations of deterministic Boltzmann machine learning," *Network,* vol. 4, pp. 355–379.

Gallant, A.R., and H. White, 1988. "There exists a neural network that does not make avoidable mistakes," *IEEE International Conference on Neural Networks,* vol. I, pp. 657–664, San Diego.

Garey, M.R., and D.S. Johnson, 1979. *Computers and Intractability,* New York: W.H. Freeman.

Gee, A.H., 1993. "Problem solving with optimization networks," Ph.D. dissertation, University of Cambridge.

Gee, A.H., S.V.B. Aiyer, and R. Prager, 1993. "An analytical framework for optimizing neural networks." *Neural Networks,* vol. 6, pp. 79–97.

Geisser, S., 1975. "The predictive sample reuse method with applications," *Journal of the American Statistical Association,* vol. 70, pp. 320–328.

Gelfand, A.E., and A.F.M. Smith, 1990. "Sampling-based approaches to calculating marginal densities," *Journal of the American Statistical Association,* vol. 85, pp. 398–409.

Geman, S., and D. Geman, 1984. "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. PAMI-6, pp. 721–741.

Geman, S., E. Bienenstock, and R. Doursat, 1992. "Neural networks and the bias/variance dilemma," *Neural Computation,* vol. 4, pp. 1–58.

Genest, C., and J. MacKay, 1989. "The joy of copulas: Bivariate distributions with uniform marginals," *The American Statistician,* vol. 40, pp. 280–285.

Gersho, A., and R.M. Gray, 1992. *Vector Quantization and Signal Compression,* Norwell, MA: Kluwer.

Gibbs, J.W., 1902. "Elementary principles in statistical mechanics," reproduced in vol. 2 of *Collected Works of J. Willard Gibbs in Two Volumes,* New York: Longmans, Green and Co., 1928.

Gidas, B., 1985. "Global optimization via the Langevin equation," *Proceedings of 24th Conference on Decision and Control,* pp. 774–778, Ft. Lauderdale, FL.

Giles, C.L., 1996. "Dynamically driven recurrent neural networks: Models, learning algorithms, and applications," Tutorial #4, *International Conference on Neural Networks,* Washington, DC.

Giles, C.L., D. Chen, G.Z. Sun, H.H. Chen, Y.C. Lee, and M.W. Goudreau, 1995. "Constructive learning of recurrent neural networks: Limitations of recurrent cascade correlation with a simple solution," *IEEE Transactions on Neural Networks,* vol. 6, pp. 829–836.

Giles, C.L., and B.G. Horne, 1994. "Representation of learning in recurrent neural network architectures," *Proceedings of the Eighth Yale Workshop on Adaptive and Learning Systems,* pp. 128–134, Yale University, New Haven, CT.

Giles, C.L., G.Z. Sun, H.H. Chen, Y.C. Lee, and D. Chen, 1990. "Higher order recurrent networks and grammatical inference," *Advances in Neural Information Processing Systems,* vol. 2, pp. 380–387, San Mateo CA: Morgan Kaufmann.

Girosi, F., 1998. "An equivalence between sparse approximation and support vector machines," *Neural Computation,* vol. 10, pp. 1455–1480.

Girosi, F, M. Jones, and T. Poggio, 1995. "Regularization theory and neural networks architectures," *Neural Computation,* vol. 7, pp. 219–269.

Glauber, R.J., 1963. "Time-dependent statistics of the Ising model," *Journal of Mathematical Physics,* vol. 4, pp. 294–307.

Golden, R.M., 1986. "The 'Brain-State-in-a-Box' neural model is a gradient descent algorithm," *Journal of Mathematical Psychology,* vol. 30, pp. 73–80.

Goles, E., and S. Martinez, 1990. *Neural and Automata Networks,* Dordrecht, The Netherlands: Kluwer.

Golub, G.H., and C.G. Van Loan, 1996. *Matrix Computations,* 3d ed., Baltimore: Johns Hopkins University Press.

Goodwin, G.C., and K.S. Sin, 1984. *Adaptive Filtering Prediction and Control,* Englewood Cliffs, NJ: Prentice-Hall.

Gordon, N.J., D.J. Salmond, and A.F.M. Smith, 1993. "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEE Proceedings-F,* vol. 140, pp. 107–113.

Goudreau, M.W., C.L. Giles, S.T. Chakradhar, and D. Chen, 1994. "First-order vs. second-order single-layer recurrent neural networks," *IEEE Transactions on Neural Networks,* vol. 5, pp. 511–513.

Grassberger, I., and I. Procaccia, 1983. "Measuring the strangeness of strange attractors," *Physica D,* vol. 9, pp. 189–208.

Gray, R.M., 1984. "Vector quantization," *IEEE ASSP Magazine,* vol. 1, pp. 4–29.

Green, M., and D.J.N. Limebeer, 1995. *Linear Robust Control,* Englewood Cliffs, NJ: Prentice Hall.

Greenberg, H.J., 1988. "Equilibria of the brain-state-in-a-box (BSB) neural model," *Neural Networks,* vol. 1, pp. 323–324.

Grenander, U., 1983. *Tutorial in Pattern Theory,* Brown University, Providence, RI.

Griffiths, L.J., and C.W. Jim, 1982. "An alternative approach to linearly constrained optimum beamforming," *IEEE Transactions on Antennas and Propagation,* vol. AP-30, pp. 27–34.

Grossberg, S., 1990. "Content-addressable memory storage by neural networks: A general model and global Liapunov method," In *Computational Neuroscience,* E.L. Schwartz, ed., pp. 56–65, Cambridge, MA: MIT Press.

Grossberg, S., 1988. *Neural Networks and Natural Intelligence,* Cambridge, MA: MIT Press.

Grossberg, S., 1982. *Studies of Mind and Brain,* Boston: Reidel.

Grossberg, S., 1969. "On learning and energy-entropy dependence in recurrent and nonrecurrent signed networks," *Journal of Statistical Physics*, vol. 1, pp. 319–350.

Grossberg, S., 1968. "A prediction theory for some nonlinear functional-difference equations," *Journal of Mathematical Analysis and Applications,* vol. 21, pp. 643–694, vol. 22, pp. 490–522.

Grossberg, S., 1967. "Nonlinear difference—differential equations in prediction and learning theory," *Proceedings of the National Academy of Sciences,* USA, vol. 58, pp. 1329–1334.

Grünwald, P.D., 2007. *the Minimum Description Length principle,* Cambridge, MA: MIT Press.

Guestrin, C., and G. Gordon, 2002. "Distributed planning in hierarchical factored MDPs," *Proceedings of 18th Conference on Uncertainty in Artificial Intelligence,* pp. 197–206, August.

Hadamard, J., 1902. "Sur les problèmes aux derivées partielles et leur signification physique," *Bulletin, Princeton University,* vol. 13, pp. 49–52.

Haft, M., and I.L. van Hemmen, 1998. "Theory and implementations of infomax filters for the retina." *Network: Computations in Neural Systems*, vol. 9, pp. 39–71.

Hagiwara, M., 1992. "Theoretical derivation of momentum term in back-propagation," *International Joint Conference on Neural Networks,* vol. I, pp. 682–686, Baltimore.

Hampson, S.E., 1990. *Connectionistic Problem Solving: Computational Aspects of Biological Learning,* Berlin: Birkhäuser.

Härdle, W., 1990. *Applied Nonparametric Regression,* Cambridge: Cambridge University Press.

Hardy, R.L., 1971. "Multiquadric equations of topography and other irregular surfaces," *Journal of Geophysics Research,* vol. 76, pp. 1905–1915.

Harel, D., 1987. *"Algorithmics: The Spirit of Computing,"* Reading, MA: Addison-Wesley.

Hartline, H.K., 1940. "The receptive fields of optic nerve fibers," *American Journal of Physiology,* vol. 130, pp. 690–699.

Harvey, A., 1989. *Forecasting, Structural Time Series Models and the Kalman Filter,* Cambridge, U.K., and New York: Cambridge University Press.

Hassibi, B., A.H. Sayed, and T. Kailath, 1999. *Indefinite-Quadratic Estimation and Control: A Unified Approach to H^2 and H^∞ Theories,* Studies in Applied and Numerical Mathematics, SIAM, Philadelphia: Society for Industrial and Applied Mathematics.

Hassibi, B., A.H. Sayed, and T. Kailath, 1996. "The H^∞ optimality of the LMS algorithm," *IEEE Transactions on Signal Processing,* vol. 44, pp. 267–280.

Hassibi, B., A.H. Sayed, and T. Kailath, 1993. "LMS is H^∞ optimal," *Proceedings of the IEEE Conference on Decision and Control,* pp. 74–79, San Antonio.

Hassibi, B., and D.G. Stork, 1993. "Second-order-derivatives for network pruning: Optimal brain surgeon," in S.H. Hanson, J.D. Cowan, and C.L. Giles, eds. *Advances in Neural Information Processing Systems,* vol. 5, pp. 164–171, San Francisco: Morgan Kaufmann.

Hassibi, B., D.G. Stork, and G.J. Wolff, 1992. "Optimal brain surgeon and general network pruning," *IEEE International Conference on Neural Networks,* vol. 1, pp. 293–299, San Francisco.

Hassibi, B., and T. Kailath, 1995. "H^∞ optimal training algorithms and their relation to back propagation," *Advances in Neural Information Proccessing Systems,* vol. 7, pp. 191–198.

Hastie, T., R. Tibshirani, and J. Friedman, 2001. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction,* New York: Springer.

Hastie, T., and W. Stuetzle, 1989. "Principal curves," *Journal of the American Statistical Association,* vol. 84, pp. 502–516.

Hastings, W.K., 1970. "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika,* vol. 87, pp. 97–109.

Haykin, S., and K. Kan, 2007. "Coherent ICA: Implications for Auditory Signal Processing," WASPA'07, New Paltz, NY, October.

Haykin, S., 2006. "Statistical Learning Theory of the LMS Algorithm Under Slowly Varying Conditions, Using the Langevin Equation," *40th Asilomar Conference on Signals, Systems, and Computers,* Pacific Grove, CA.

Haykin, S., and Z. Chen, 2006. "The machine cocktail party problem," in S. Haykin, J.C. Principe, T.J. Sejnowski, and J. McWhirter, eds., *New Directions in Statistical Signal Processing: From Systems to Brain,* pp. 51–75, Cambridge, MA: MIT Press.

Haykin, S., and B. Widrow, 2003. *Least-Mean-Square Adaptive Filters,* New York: Wiley-Interscience,

Haykin, S., 2002. *Adaptive Filter Theory,* 4th ed., Englewood Cliffs, NJ: Prentice Hall.

Haykin, S., and C. Deng, 1991. "Classification of radar clutter using neural networks," *IEEE Transactions on Neural Networks,* vol. 2, pp. 589–600.

Haykin, S., and B. Van Veen, 1998. *Signals and Systems,* New York: Wiley.

Hebb, D.O., 1949. *The Organization of Behavior: A Neuropsychological Theory,* New York: Wiley,

Hecht-Nielsen, R., 1995. "Replicator neural networks for universal optimal source coding," *Science,* vol. 269, pp. 1860–1863.

Hecht-Nielsen, R., 1990. *Neurocomputing,* Reading, MA: Addison-Wesley.

Hecht-Nielsen, R., 1987. "Kolmogorov's mapping neural network existence theorem," *First IEEE International Conference on Neural Networks,* vol. III, pp. 11–14, San Diego.

Helstrom, C.W., 1968. *Statistical Theory of Signal Detection,* 2nd edition, Pergamon Press.

Herault, J., and C. Jutten, 1986. "Space or time adaptive signal processing by neural network models," in J.S. Denker, ed., *Neural Networks for Computing.* Proceedings of the AIP Conference, American Institute of Physics, New York, pp. 206–211.

Herault, J., C. Jutten, and B. Ans, 1985. "Detection de grandeurs primitives dans un message composite par une architecture de calcul neuromimetique un apprentissage non supervise." *Procedures of GRETSI,* Nice, France.

Herbrich, R., 2002. *Learning Kernel Classifiers: Theory and Algorithms,* Cambridge, MA: MIT Press.

Hertz, J., A. Krogh, and R.G. Palmer, 1991. *Introduction to the Theory of Neural Computation,* Reading, MA: Addison-Wesley.

Heskes, T.M., 2001. "Self-organizing maps, vector quantization, and mixture modeling," *IEEE Trans. on Neural Networks,* vol. 12, pp. 1299–1305.

Heskes, T.M. and B. Kappen, 1991. "Learning processes and neural networks," *Phys. Rev.,* A44, pp. 2718–2726.

Hestenes, M.R., and E. Stiefel, 1952. "Methods of conjugate gradients for solving linear systems," *Journal of Research of the National Bureau of Standards,* vol. 49, pp. 409–436.

Hiller, F.S., and G.J. Lieberman, 1995. *Introduction to Operations Research,* 6th ed., New York: McGraw-Hill.

Hinton, G.E., 2007. *Deep Belief Nets,* 2007 NIPS Tutorial Notes, *Neural Information Processing Systems Conference,* Vancouver, BC, December.

Hinton, G.E., S. Osindero, and Y. Teh, 2006. "A fast learning algorithm for deep belief nets," *Neural Computation,* vol. 18, pp. 1527–1554.

Hinton, G.E., 1989. "Connectionist learning procedures," *Artificial Intelligence,* vol. 40, pp. 185–234.

Hinton, G.E., and T.J. Sejnowski, 1986. "Learning and relearning in Boltzmann machines," in *Parallel Distributed Processing: Explorations in Microstructure of Cognition,* D.E. Rumelhart and J.L. McClelland, eds., Cambridge, MA: MIT Press.

Hinton, G.E., and T.J. Sejnowski, 1983. "Optimal perceptual inference," *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* pp. 448–453, Washington, DC.

Hirsch, M.W., 1989. "Convergent activation dynamics in continuous time networks," *Neural Networks,* vol. 2, pp. 331–349.

Hirsch, M.W., and S. Smale, 1974. *Differential Equations, Dynamical Systems, and Linear Algebra,* New York: Academic Press.

Ho, Y.C., and R.C.K. Lee, 1964. "A Bayesian approach to problems in stochastic estimation and control," *IEEE Trans. Automatic Control,* vol. AC-9, pp. 333–339.

Hochreiter, S., 1991. *Untersuchungen zu dynamischen neuronalen Netzen,* diploma thesis, Technische Universität Munchen, Germany.

Hodgkin, A.L., and A.F. Huxley, 1952. "A quantitative description of membrane current and its application to conduction and excitation in nerve," *Journal of Physiology,* vol. 117, pp. 500–544.

Holland, J.H., 1992. *Adaptation in Natural and Artificial Systems,* Cambridge, MA: MIT Press.

Holmström, L., P. Koistinen, J. Laaksonen, and E. Oja, 1997. "Comparison of neural and statistical classifiers—taxonomy and two case studies," *IEEE Trans. on Neural Networks,* vol. 8, pp. 5–17.

Honkela, T., 2007. "Philosphical aspects of neural, probabilistic and fuzzy modeling of language use and translation," *Proceedings of IJCNN, International Joint Conference on Neural Networks,* pp. 2881–2886, Orlando, FL, August.

Honkela, T., V. Pulkki, and T. Kohonen, 1995. "Contextual relations of words in Grimm tales, analyzed by self-organizing maps," *Proceedings of International Conference on Artificial Neural Networks,* ICANN-95, vol. II, pp. 3–7, Paris.

Hopcroft, J., and U. Ullman, 1979. *Introduction to Automata Theory, Languages and Computation,* Reading MA: Addison-Wesley.

Hopfield, J.J., 1995. "Pattern recognition computation using action potential timing for stimulus representation," *Nature,* vol. 376, pp. 33–36.

Hopfield, J.J., 1994. "Neurons, dynamics and computation," *Physics Today,* vol. 47, pp. 40–46, February.

Hopfield, J.J., 1987. "Learning algorithms and probability distributions in feed-forward and feed-back networks," *Proceedings of the National Academy of Sciences, USA,* vol. 84, pp. 8429–8433.

Hopfield, J.J., 1984. "Neurons with graded response have collective computational properties like those of two-state neurons," *Proceedings of the National Academy of Sciences, USA,* vol. 81, pp. 3088–3092.

Hopfield, J.J., 1982. "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the National Academy of Sciences, USA,* vol. 79, pp. 2554–2558.

Hopfield, J.J., and T.W. Tank, 1985. "'Neural' computation of decisions in optimization problems," *Biological Cybernetics,* vol. 52, pp. 141–152.

Hornik, K., M. Stinchcombe, and H. White, 1990. "Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks," *Neural Networks,* vol. 3, pp. 551–560.

Hornik, K., M. Stinchcombe, and H. White, 1989. "Multilayer feedforward networks are universal approximators," *Neural Networks,* vol. 2, pp. 359–366.

Hotelling, H., 1936. "Relations between two sets of variates," *Biometrika,* vol. 28, pp. 321–377.

Hotelling, H., 1935. "The most predictable criterion," *J. Educational Psychology,* vol. 26, pp. 139–142.

Hotelling, H., 1933. "Analysis of a complex of statistical variables into principal components," *Journal of Educational Psychology,* vol. 24, pp. 417–441, 498–520.

Howard, R.A., 1960. *Dynamic Programming and Markov Processes,* Cambridge, MA: MIT Press.

Hubel, D.H., 1988. *Eye, Brain, and Vision,* New York: Scientific American Library.

Hubel, D.H., and T.N. Wiesel, 1977. "Functional architecture of macaque visual cortex," *Proceedings of the Royal Society,* B, vol. 198, pp. 1–59, London.

Hubel, D.H., and T.N. Wiesel, 1962. "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *Journal of Physiology,* vol. 160, pp. 106–154, London.

Huber, P.J., 1985. "Projection pursuit," *Annals of Statistics,* vol. 13, pp. 435–475.

Huber, P.J., 1981. *Robust Statistics,* New York: Wiley.

Huber, P.J., 1964. "Robust estimation of a location parameter," *Annals of Mathematical Statistics,* vol. 35, pp. 73–101.

Hush, D., P. Kelly, C. Scovel, and I. Steinwart, 2006. "QP algorithms with guaranteed accuracy and run time for support vector machines," *J. Machine Learning Research,* vol. 7, pp. 733–769.

Hush, D.R., and B.G. Home, 1993. "Progress in supervised neural networks: What's new since Lippmann?" *IEEE Signal Processing Magazine,* vol. 10, pp. 8–39.

Hush, D.R., and J.M. Salas, 1988. "Improving the learning rate of back-propagation with the gradient reuse algorithm," *IEEE International Conference on Neural Networks,* vol. I, pp. 441–447, San Diego.

Hyvärinen, A., J. Karhunen, and E. Oja, 2001. *Independent Component Analysis,* New York: Wiley-Interscience.

Hyvärinen, A. and E. Oja, 2000. "Independent component analysis: Algorithms and applications," *Neural Networks,* vol. 13, pp. 411–430.

Hyvärinen, A. and E. Oja, 1997. "A fast fixed-point algorithm for independent component analysis," *Neural Computation,* vol. 9, pp. 1483–1492.

Ito, K., and K. Xing, 2000. "Gaussian filters for nonlinear filtering problems," *IEEE Trans. Automatic Control,* vol. 45, pp. 910–927.

Izhikevich, E., and G.M. Edelman, 2008. "Large-scale model of mammalian thalamocortical systems," *Proceedings of the National Academy of Sciences,* vol. 105, pp. 3593–3598.

Izhikevich, E.M., 2007a. *Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting,* Cambridge, MA: MIT Press.

Izhikevich, E.M., 2007b. "Solving the distal reward problem through linkage of STDP and dopamine signaling", Cerebral Cortex, vol. 17, pp. 2443–2452.

Jackson, E.A., 1989. *Perspectives of Nonlinear Dynamics,* vol. 1, Cambridge, U.K.: Cambridge University Press.

Jackson, E.A., 1990. *Perspectives of Nonlinear Dynamics,* vol. 2, Cambridge, U.K.: Cambridge University Press.

Jackson, J.D., 1975. *Classical Electrodynamics,* 2d ed., New York: Wiley.

Jacobs, R.A., 1988. "Increased rates of convergence through learning rate adaptation," *Neural Networks,* vol. 1, pp. 295–307.

Jayant, N.S., and P. Noll, 1984. *Digital Coding of Waveforms,* Englewood Cliffs, NJ: Prentice-Hall.

Jaynes, E.T., 2003. *Probability Theory: The Logic of Science,* Cambridge, U.K., and New York: Cambridge University Press.

Jaynes, E.T., 1982. "On the rationale of maximum-entropy methods," *Proceedings of the IEEE,* vol. 70, pp. 939–952.

Jaynes, E.T., 1957. "Information theory and statistical mechanics," *Physical Review,* vol. 106, pp. 620–630; "Information theory and statistical mechanic II," *Physical Review,* vol. 108, pp. 171–190.

Jazwinski, A.H., 1970. *Stochastic Processes and Filtering Theory,* New York: Academic Press.

Jelinek, F., 1997. *Statistical Methods for Speech Recognition,* Cambridge, MA: MIT Press.

Joachims, T., 1999. "Making large-scale SVM learning practical," in B. Schölkopf, C.J.C. Burges, and A.J. Smola, eds., *Advances in Kernel Methods—Support Vector Learning,* pp. 169–184, Cambridge, MA: MIT Press.

Johansson, E.M., F.U. Dowla, and D.M. Goodman, 1990. "Back-propagation learning for multi-layer feedforward neural networks using the conjugate gradient method," Report UCRL-JC-104850, Lawrence Livermore National Laboratory, CA.

Johnson, D.S., C.R. Aragon, L.A. McGeoch, and C. Schevon, 1989. "Optimization by simulated annealing: An experimental evaluation," *Operations Research,* vol. 37, pp. 865–892.

Jolliffe, I.T., 1986. *Principal Component Analysis,* New York: Springer-Verlag.

Jordan, M.I., 1986. "Attractor dynamics and parallelism in a connectionist sequential machine," *The Eighth Annual Conference of the Cognitive Science Society,* pp. 531–546, Amherst, MA.

Jordan, M.I., ed., 1998. *Learning in Graphical Models,* Boston: Kluwer.

Joseph, R.D., 1960. "The number of orthants in n-space intersected by an s-dimensional subspace," Technical Memo 8, Project PARA, Cornell Aeronautical Lab., Buffalo.

Julier, S.J., and J.K. Ulhmann, 2004. "Unscented filtering and nonlinear estimation," *Proc. IEEE,* vol. 92, pp. 401–422.

Julier, S.J., J.K. Ulhmann, and H.F. Durrent-Whyte, 2000. "A new method for nonlinear transformation of means and covariances in filters and estimation," *IEEE Trans. Automatic Control,* vol. 45, pp. 472–482.

Jutten, C. and A. Taleb, 2000. "Source separation: from dusk till dawn," *Proceedings of 2nd International Workshop on Independent Component Analysis and Blind Source Separation,* Helsinki, June.

Jutten, C., and J. Herault, 1991. "Blind separation of sources, Part I: An adaptive algorithm based on neuromimetic architecture," *Signal Processing,* vol. 24, pp. 1–10.

Kaas, J.H., M.M. Merzenich, and H.P. Killackey, 1983. "The reorganization of somatosensory cortex following peripheral nerve damage in adult and developing mammals," *Annual Review of Neurosciences,* vol. 6, pp. 325–356.

Kailath, T., 1980. *Linear Systems,* Englewood Cliffs, NJ: Prentice-Hall.

Kailath, T., 1971. "RKHS approach to detection and estimation problems—Part I: Deterministic signals in Gaussian noise," *IEEE Transactions of Information Theory,* vol. IT-17, pp. 530–549.

Kailath, T., 1968. "An innovations approach to least-squares estimation: Part 1. Linear filtering in additive white noise," *IEEE Transactions of Automatic Control,* vol. AC-13, pp. 646–655.

Kakade, S., 2002. "A natural policy gradient," *Advances in Neural Information Processing Systems,* vol. 14-2, pp. 1531–1538, Cambridge, MA: MIT Press.

Kalman, R.E., 1960. "A new approach to linear filtering and prediction problems," *Transactions of the ASME, Journal of Basic Engineering,* vol. 82, pp. 35–45.

Kaminski, P.G., A.E. Bryson, Jr., and S.F. Schmidt, 1971. "Discrete square root filtering: A survey of current techniques," *IEEE Trans. Automatic Control,* vol. AC-16, pp. 727–735.

Kammler, D. W., 2000, *A First Course in Fourier Analysis*, Prentice-Hall

Kan, K., 2007. *Coherent Independent Component Analysis: Theory and Applications,* M.Eng. thesis, McMaster University, Hamilton, Ontario, Canada.

Kandel, E.R., J.H. Schwartz, and T.M. Jessell, eds., 1991. *Principles of Neural Science,* 3d ed., Norwalk, CT: Appleton & Lange.

Kaplan, J.L., and J.A. Yorke, 1979. "Chaotic behavior of multidimensional difference equations," in H.-O. Peitgen and H.-O. Walker, eds., *Functional Differential Equations and Approximations of Fixed Points,* pp. 204–227, Berlin: Springer.

Kappen, H.J., and F.B. Rodriguez, 1998. "Efficient learning in Boltzmann machines using linear response theory," *Neural Computation,* vol. 10, pp. 1137–1156.

Karhunen, J., and J. Joutsensalo, 1995. "Generalizations of principal component analysis, optimization problems, and neural networks," *Neural Networks,* vol. 8, pp. 549–562.

Karhunen, J. and E. Oja, 1982. "New methods for stochastic approximation of truncated Karhunen–Loève expansions," *IEEE Proceedings of the 6th International Conference on Pattern Recognition,* pp. 550–553, October.

Karhunen, K., 1947. "Über lineare methoden in der Wahrscheinlichkeitsrechnung," *Annales Academiae Scientiarum Fennicae, Series AI: Mathematica-Physica,* vol. 37, pp. 3–79, (Transl.: RAND Corp., Santa Monica, CA, Rep. T-131, Aug. 1960).

Karush, W., 1939. "Minima of functions of several variables with inequalities as side conditions," master's thesis, Department of Mathematics, University of Chicago.

Katz, B., 1966. *Nerve, Muscle and Synapse,* New York: McGraw-Hill.

Kawamoto, A.H., and J.A. Anderson, 1985. "A neural network model of multistable perception," *Acta Psychologica,* vol. 59, pp. 35–65.

Kearns, M., 1996. "A bound on the error of cross validation using the approximation and estimation rates, with consequences for the training-test split," *Advances in Neural Information Processing Systems,* vol. 8, pp. 183–189, Cambridge, MA: MIT Press.

Kearns, M.J., and U.V. Vazirani, 1994. *An Introduction to Computational Learning Theory,* Cambridge, MA: MIT Press.

Kechriotis, G., E. Zervas, and E.S. Manolakos, 1994. "Using recurrent neural networks for adaptive communication channel equalization," *IEEE Transactions on Neural Networks,* vol. 5, pp. 267–278.

Kerlirzin, P., and F. Vallet, 1993. "Robustness in multilayer perceptrons," *Neural Computation,* vol. 5, pp. 473–482.

Khalil, H.K., 1992. *Nonlinear Systems,* Englewood Cliffs, NJ: Prentice Hall.

Kim, K.I., M.O. Franz, and B. Schölkopf, 2005. "Iterative kernel principal component analysis for image denoising," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 27, pp. 1351–1366.

Kimeldorf, G.S., and G. Wahba, 1971. "Some results on Tchebycheffian spline functions," *J. Math. Apal. Appli.*, vol. 33, pp. 82–95.

Kimeldorf, G.S., and G. Wahba, 1970, "A correspondence between Bayesian estimation on stochastic processes and smoothing by splines," *Annals of Mathematical Statistics,* vol. 41, pp. 495–502.

Kirkpatrick, S., 1984. "Optimization by simulated annealing: Quantitative studies," *Journal of Statistical Physics,* vol. 34, pp. 975–986.

Kirkpatrick, S., and D. Sherrington, 1978. "Infinite-ranged models of spin-glasses," *Physical Review,* Series B, vol. 17, pp. 4384–4403.

Kirkpatrick, S., C.D. Gelatt, Jr., and M.P. Vecchi, 1983. "Optimization by simulated annealing," *Science,* vol. 220, pp. 671–680.

Kirsch, A., 1996. *An Introduction to the Mathematical Theory of Inverse Problems,* New York: Springer-Verlag.

Kleene, S.C., 1956. "Representation of events in nerve nets and finite automata," in C.E. Shannon and J. McCarthy, eds., *Automata Studies,* Princeton, NJ: Princeton University Press.

Kline, M., 1972. *Mathematical Thought from Ancient to Modern Times,* Oxford University Press.

Kmenta, J., 1971. *Elements of Econometrics,* New York: Macmillan.

Knill, D.C., and W. Richards, eds., 1996. *Perception as Bayesian Inference,* Cambridge, U.K., and New York: Cambridge University Press.

Knudsen, E.I., S. duLac, and S.D. Esterly, 1987. "Computational maps in the brain," *Annual Review of Neuroscience,* vol. 10, pp. 41–65.

Koch, C., 1999. *Biophysics of Computation: Information Processing in Single Neurons,* New York: Oxford University Press.

Kohonen, T., 1997a. "Exploration of very large databases by self-organizing maps," *1997 International Conference on Neural Networks,* vol. I, pp. PL1–PL6, Houston.

Kohonen, T., 1997b. *Self-Organizing Maps,* 2d ed., Berlin: Springer-Verlag.

Kohonen, T., 1993. "Things you haven't heard about the self-organizing map," *Proceedings of the IEEE International Conference on neural networks,* pp. 1147–1156, San Francisco.

Kohonen, T., 1990. "The self-organizing map," *Proceedings of the Institute of Electrical and Electronics Engineers,* vol. 78, pp. 1464–1480.

Kohonen, T., 1982. "Self-organized formation of topologically correct feature maps," *Biological Cybernetics,* vol. 43, pp. 59–69.

Kohonen, T., 1972. "Correlation matrix memories," *IEEE Transactions on Computers,* vol. C-21, pp. 353–359.

Kolen, J., and S. Kremer, eds., 2001. *A Field Guide to Dynamical Recurrent Networks,* New York: IEEE Press.

Kollias, S., and D. Anastassiou, 1989. "An adaptive least squares algorithm for the efficient training of artificial neural networks," *IEEE Transactions on Circuits and Systems,* vol. 36, pp. 1092–1101.

Kolmogorov, A., 1965. "Three approaches to the quantitative definition of information," *Problems of Information Transmission,* vol. 1, issue 1, pp. 1–7.

Kolmogorov, A.N., 1942. "Interpolation and extrapolation of stationary random sequences," translated by the Rand Corporation, Santa Monica, CA., April 1962.

Kramer, M.A., 1991. "Nonlinear principal component analysis using autoassociative neural networks," *AIChE Journal,* vol. 37, pp. 233–243.

Kramer, A.H., and A. Sangiovanni-Vincentelli, 1989. "Efficient parallel learning algorithms for neural networks," *Advances in neural Information Processing Systems,* vol. 1, pp. 40–48, San Mateo, CA: Morgan Kaufmann.

Kremer, S.C., 1996. "Comments on constructive learning of recurrent neural networks: Limitations of recurrent cascade correlation and a simple solution," *IEEE Transactions on Neural Networks,* vol. 7, pp. 1047–1049.

Kremer, S.C., 1995. "On the computational power of Elman-style recurrent networks," *IEEE Transactions on Neural Networks,* vol. 6, pp. 1000–1004.

Kreyszig, E., 1988. *Advanced Engineering Mathematics,* 6th ed., New York: Wiley.

Krzyżak, A., T. Linder, and G. Lugosi, 1996. "Nonparametric estimation and classification using radial basis functions," *IEEE Transactions on Neural Networks,* vol. 7, pp. 475–487.

Kuffler, S.W., J.G. Nicholls, and A.R. Martin, 1984. *From Neuron to Brain: A Cellular Approach to the Function of the Nervous System,* 2d ed., Sunderland, MA: Sinauer Associates.

Kullback, S., 1968. *Information Theory and Statistics,* Gloucester, MA: Peter Smith.

Kuhn, H.W., 1976. "Nonlinear programming: A historical view," in R.N. Cottle and C.E. Lemke, eds., *SIAM-AMS Proceedings,* vol. IX, American Mathematical Society, pp. 1–26.

Kuhn, H.W., and A.W. Tucker, 1951. "Nonlinear programming," in J. Neyman, ed., *Proceedings of the 2nd Berkley Symposium on Mathematical Statistics and Probabilities,* pp. 481–492, Monterey CA: University of California Press.

Kung, S.Y., and K.I. Diamantaras, 1990. "A neural network learning algorithm for adaptive principal component extraction (APEX)." *IEEE International Conference on Acoustics, Speech, and Signal Processing,* vol. 2, pp. 861–864, Albuquerque, NM.

Kushner, H.J., and D.S. Clark, 1978. *Stochastic Approximation Methods for Costrained and Unconstrained Systems,* New York: Springer-Verlag.

Laaksonen, J., and V. Viitanieni, 2006. "Emergence of ontological relations from visual data with self-organizing maps," *Proceedings of SCAI'06, the 9th Scandanavian Conference on Artificial Intelligence,* pp. 31–38, Espoo, Finland, October.

Laaksonen, J.T., J.M. Koskela, and E. Oja, 2004. "Class distributions on SOM surfaces for feature extraction and object retrieval," *Neural Networks,* vol. 17, pp. 1121–1133.

Lagoudakis, M.G., and R. Parr, 2003. "Least-squares policy iteration," *J. Machine Learning Research,* vol. 4, pp. 1107–1149.

Lanczos, C., 1964. *Linear Differential Operators,* London: Van Nostrand.

Landau, Y.D., 1979. *Adaptive Control: The Model Reference Approach,* New York: Marcel Dekker.

Landau, L.D., and E.M. Lifshitz, 1980. *Statistical Physics: Part 1,* 3d ed., London: Pergamon Press.

Lanford, O.E., 1981. "Strange attractors and turbulence," in H.L. Swinney and J.P. Gollub, eds., *Hydrodynamic Instabilities and the Transition to Turbulence,* New York: Springer-Verlag.

Lang, S. 2002. *Introduction to Differentiable Manifolds*, New York: Springer.

LeCun, Y., 1993. *Efficient Learning and Second-order Methods, A Tutorial at NIPS 93,* Denver.

LeCun, Y., 1989. "Generalization and network design strategies," Technical Report CRG-TR-89-4, Department of Computer Science, University of Toronto, Ontario, Canada.

LeCun, Y., 1985. "Une procedure d'apprentissage pour reseau a seuil assymetrique." *Cognitiva,* vol. 85, pp. 599–604.

LeCun, Y., and Y., Bengio, 2003. "Convolutional Networks for Images, Speech, and Time Series," in M.A. Arbib, ed., *The Handbook of Brain Theory and Neural Networks,* 2d ed., Cambridge, MA: MIT Press.

LeCun, Y., B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, and L.D. Jackel, 1990. "Handwritten digit recognition with a back-propagation network," *Advances in Neural Information Processing,* vol. 2, pp. 396–404, San Mateo, CA: Morgan Kaufmann.

LeCun, Y., L. Bottou, and Y. Bengio, 1997. "Reading checks with multilayer graph transformer networks," *IEEE International Conference on Acoustics, Speech and Signal Processing,* pp. 151–154, Munich, Germany.

LeCun, Y., L. Bottou, Y. Bengio, and R. Haffner, 1998. "Gradient-based learning applied to document recognition," *Procedings of the IEEE*, vol. 86, pp. 2278–2324.

LeCun, Y., J.S. Denker, and S.A. Solla, 1990. "Optimal brain damage," *Advances in Neural Information Processing Systems,* vol. 2, pp. 598–605, San Mateo, CA: Morgan Kaufmann.

LeCun, Y, I. Kanter, and S.A. Solla, 1991. "Second order properties of error surfaces: Learning time and generalization," *Advances in Neural Information Processing Systems,* vol. 3, pp. 918–924, Cambridge, MA: MIT Press.

Lee, T.S., and D. Mumford, 2003. "Hierarchical Bayesian inference in the visual cortex," *J. Optical Society of America,* vol. 20, pp. 1434–1448.

Lefebvre, W.C., 1991. *An Object Oriented Approach for the Analysis of Neural Networks,* master's thesis, University of Florida, Gainesville, FL.

Leon-Garcia, A., 1994. *Probability and Random Processes for Electrical Engineering,* 2d ed., Reading, MA: Addison-Wesley.

Leontaritis, I., and S. Billings, 1985. "Input–output parametric models for nonlinear systems: Part I: Deterministic nonlinear systems," *International Journal of Control,* vol. 41, pp. 303–328.

Levenberg, K., 1944. "A method for the solution of certain non-linear problems in least squares," *Quart. Appl. Math.,* vol. 12, pp. 164–168.

Levin, A.V., and K.S. Narendra, 1996. "Control of nonlinear dynamical systems using neural networks—Part II: Observability, identification, and control," *IEEE Transactions on Neural Networks,* vol. 7, pp. 30–42.

Levin, A.V., and K.S. Narendra, 1993. "Control of nonlinear dynamical systems using neural networks—Controllability and stabilization," *IEEE Transactions on Neural Networks,* vol. 4, pp. 192–206.

Lewis, F.L., and V.L. Syrmas, 1995. *Optimal Control,* 2d ed., New York: Wiley (Interscience).

Li, M., and P. Vitányi, 1993. *An Introduction to Kolmogorov Complexity and Its Applications*, New York: Springer-Verlag.

Lichtenberg, A.J., and M.A. Lieberman, 1992. *Regular and Chaotic Dynamics,* 2d ed., New York: Springer-Verlag.

Light, W.A., 1992a. "Some aspects of radial basis function approximation," in *Approximation Theory, Spline Functions and Applications,* S.P. Singh, ed., NATO ASI vol. 256, pp. 163–190, Boston: Kluwer Academic Publishers.

Light, W., 1992b. "Ridge functions, sigmoidal functions and neural networks," in E.W. Cheney, C.K. Chui, and L.L. Schumaker, eds., *Approximation Theory VII,* pp. 163–206, Boston: Academic Press.

Lin, S., and D.J. Costello, 2004. *Error Control Coding,* 2d ed., Upper Saddle River, NJ: Prentice Hall.

Lin, J.K., D.G. Grier, and J.D. Cowan, 1997. "Faithful representation of separable distributions," *Neural Computation,* vol. 9, pp. 1305–1320.

Lin, T., B.G. Horne, P. Tino, and C.L. Giles, 1996. "Learning long-term dependencies in NARX recurrent neural networks," *IEEE Transactions on Neural Networks,* vol. 7, pp. 1329–1338.

Linsker, R., 1993. "Deriving receptive fields using an optimal encoding criterion," *Advances in Neural Information Processing Systems,* vol. 5, pp. 953–960, San Mateo, CA: Morgan Kaufmann.

Linsker, R., 1990a. "Designing a sensory processing system: What can be learned from principal components analysis?" *Proceedings of the International Joint Conference on Neural Networks,* vol. 2, pp. 291–297, Washington, DC.

Linsker, R., 1990b. "Perceptual neural organization: Some approaches based on network models and information theory," *Annual Review of Neuroscience,* vol. 13, pp. 257–281.

Linsker, R., 1989a. "An application of the principle of maximum information preservation to linear systems," *Advances in Neural Information Processing Systems,* vol. 1, pp. 186–194, San Mateo, CA: Morgan Kaufmann.

Linsker, R., 1989b. "How to generate ordered maps by maximizing the mutual information between input and output signals," *Neural computation,* vol. 1, pp. 402–411.

Linsker, R., 1988a. "Self-organization in a perceptual network," *Computer,* vol. 21, pp. 105–117.

Linsker, R., 1988b. "Towards an organizing principle for a layered perceptual network," in *Neural Information Processing Systems,* D.Z. Anderson, ed., pp. 485–494, New York: American Institute of Physics.

Linsker, R., 1987. "Towards an organizing principle for perception: Hebbian synapses and the principle of optimal neural encoding," *IBM Research Report RC12820,* IBM Research, Yorktown Heights, NY.

Linsker, R., 1986. "From basic network principles to neural architecture" (series), *Proceedings of the National Academy of Sciences, USA,* vol. 83, pp. 7508–7512, 8390–8394, 8779–8783.

Lippmann, R.P., 1987. "An introduction to computing with neural nets," *IEEE ASSP Magazine,* vol. 4, pp. 4–22.

Lippmann, R.P., 1989a. "Review of neural networks for speech recognition," *Neural Computation,* vol. 1, pp. 1–38.

Lippmann, R.P., 1989b. "Pattern classification using neural networks," *IEEE Communications Magazine,* vol. 27, pp. 47–64.

Little, W.A., 1974. "The existence of persistent states in the brain," *Mathematical Biosciences,* vol. 19, pp. 101–120.

Little, W.A., and G.L. Shaw, 1978. "Analytic study of the memory storage capacity of a neural network," *Mathematical Biosciences,* vol. 39, pp. 281–290.

Little, W. A., and G.L. Shaw, 1975. "A statistical theory of short and long term memory," *Behavioral Biology,* vol. 14, pp. 115–133.

Littman, M.L., R.S. Sutton, and S. Singh, 2002. "Predictive representations of state," *Advances in Neural Information Processing Systems,* vol. 14, pp. 1555–1561.

Liu, J.S., and R. Chen, 1998. "Sequential Monte Carlo methods for dynamical systems," *J. American Statistical Association,* vol. 93, pp. 1032–1044.

Liu, J.S., 1996. "Metropolized independent sampling with comparisons to rejection sampling and importance sampling," *Statistics and Computing,* vol. 6, pp. 113–119.

Liu, W., P.P. Pokharel, and J.C. Principe, 2008. "The kernel least-mean-square algorithm," *IEEE Trans. Signal Processing,* vol. 56, pp. 543–554.

Livesey, M., 1991. "Clamping in Boltzmann machines," *IEEE Transactions on Neural Networks,* vol. 2, pp. 143–148.

Ljung, L., 1987. *System Identification: Theory for the User.* Englewood Cliffs, NJ: Prentice-Hall.

Ljung, L., 1977. "Analysis of recursive stochastic algorithms," *IEEE Transactions on Automatic Control,* vol. AC-22, pp. 551–575.

Ljung, L., and T. Glad, 1994. *Modeling of Dynamic Systems,* Englewood Cliffs, NJ: Prentice Hall.

Lloyd, S.P., 1957. "Least squares quantization in PCM," unpublished Bell Laboratories technical note. Published later under the same title in *IEEE Transactions on Information Theory,* vol. IT-28, pp. 127–135, 1982.

Lo, J.T., 2001. "Adaptive vs. accommodative neural networks for adaptive system identification," *Proceedings of the International Joint Conference on Neural Networks,* pp. 2001–2006, Washington, DC.

Lo, J.T., and L. Yu, 1995a. "Recursive neural filters and dynamical range transformers," *Proc. IEEE,* vol. 92, pp. 514–535.

Lo, J.T., and L. Yu, 1995b. Adaptive neural filtering by using the innovations process, *Proceedings of the 1995 World Congress on Neural Networks,* vol. II, pp. 29–35, July.

Lo, J.T., 1993. "Dynamical system identification by recurrent multilayer perceptrons," *Proceedings of the 1993 World Congress on Neural Networks,* Portland, OR.

Lo, Z.-P., M. Fujita, and B. Bavarian, 1991. "Analysis of neighborhood interaction in Kohonen neural networks," *6th International Parallel Processing Symposium Proceedings,* pp. 247–249, Los Alamitos, CA.

Lo, Z.-P., Y. Yu and B. Bavarian, 1993. "Analysis of the convergence properties of topology preserving neural networks," *IEEE Transactions on Neural Networks,* vol. 4, pp. 207–220.

Lockery, S.R., Y. Fang, and T.J. Sejnowski, 1990. "A dynamical neural network model of sensorimotor transformations in the leech," *International Joint Conference on Neural Networks,* vol. I, pp. 183–188, San Diego, CA.

Loève, M., 1963. *Probability Theory,* 3d ed., New York: Van Nostrand.

Lorentz, G.G., 1976. "The 13th problem of Hilbert," *Proceedings of Symposia in Pure Mathematics,* vol. 28, pp. 419–430.

Lorentz, G.G., 1966. *Approximation of Functions,* Orlando, FL: Holt, Rinehart & Winston.

Lorenz, E.N., 1963. "Deterministic non-periodic flows," *Journal of Atmospheric Sciences,* vol. 20, pp. 130–141.

Lowe, D., 1989. "Adaptive radial basis function nonlinearities, and the problem of generalisation," *First IEE International Conference on Artificial Neural Networks,* pp. 171–175, London.

Lowe, D., 1991a. "What have neural networks to offer statistical pattern processing?" *Proceedings of the SPIE Conference on Adaptive Signal Processing,* pp. 460–471, San Diego.

Lowe, D., 1991b. "On the iterative inversion of RBF networks: A statistical interpretation," *Second IEE International Conference on Artificial Neural Networks,* pp. 29–33, Bournemouth, U.K.

Lowe, D., and A.R. Webb, 1991a. "Time series prediction by adaptive networks: A dynamical systems perspective," *IEE Proceedings (London), Part F,* vol. 138, pp. 17–24.

Lowe, D., and A.R. Webb, 1991b. "Optimized feature extraction and the Bayes decision in feed-forward classifier networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* PAMI-13, 355–364.

Luenberger, D.G., 1984. *Linear and Nonlinear Programming,* 2d ed., Reading, MA: Addison-Wesley.

Luttrell, S.P., 1994. "A Bayesian analysis of self-organizing maps," *Neural Computation,* vol. 6, pp. 767–794.

Luttrell, S.P., 1991. "Code vector density in topographic mappings: Scalar case," *IEEE Transactions on Neural Networks,* vol. 2, pp. 427–436.

Luttrell, S.P., 1989a. "Hierarchical vector quantization," *IEE Proceedings (London),* vol. 136 (Part I), pp. 405–413.

Luttrell, S.P., 1989b. "Self-organization: A derivation from first principle of a class of learning algorithms," *IEEE Conference on Neural Networks,* pp. 495–498, Washington, DC.

Lyapunov, A.M., 1892. *The General Problem of Motion Stability* (in Russian). (Translated in English by F. Abramovici and M. Shimshoni, under the title *Stability of Motion,* New York: Academic Press, 1966.)

Maass, W., 1993. "Bounds for the computational power and learning complexity of analog neural nets," *Proceedings of the 25th Annual ACM Symposium on the Theory of Computing,* pp. 335–344, New York: ACM Press.

MacKay, D.J.C., 2003. *Information Theory, Inference, and Learning Algorithms,* Cambridge, U.K., and New York: Cambridge University Press.

Mackey, M.C., and L. Glass, 1977. "Oscillations and chaos in physiological control systems," *Science,* vol. 197, pp. 287–289.

MacQueen, J., 1967. "Some methods for classification and analysis of multivariate observation," in *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability,* L.M. LeCun and J. Neyman, eds., vol. 1, pp. 281–297, Berkeley: University of California Press.

Mahowald, M.A., and C. Mead, 1989. "Silicon retina," in *Analog VLSI and Neural Systems* (C. Mead), Chapter 15. Reading, MA: Addison-Wesley.

Mallat, S., 1998. *A Wavelet tour of signal processing*, San Diego: Academic Press.

Mandelbrot, B.B., 1982. *The Fractal Geometry of Nature,* San Francisco: Freeman.

Mañé, R., 1981. "On the dimension of the compact invariant sets of certain non-linear maps," in D. Rand and L.S. Young, eds., *Dynamical Systems and Turbulence,* Lecture Notes in Mathematics, vol. 898, pp. 230–242, Berlin: Springer-Verlag.

Marquardt, D.W. 1963. "An algorithm for least-squares estimation of nonlinear parameters," *J. Soc. Indust. Appli. Math.*, vol. 11, no. 2, pp. 431–441, June.

Marr, D., 1982. *Vision,* New York: W.H. Freeman and Company.

Mason, S.J., 1953. "Feedback theory—Some properties of signal-flow graphs," *Proceedings of the Institute of Radio Engineers,* vol. 41, pp. 1144–1156.

Mason, S.J., 1956. "Feedback theory—Further properties of signal-flow graphs," *Proceedings of the Institute of Radio Engineers,* vol. 44, pp. 920–926.

Maybeck, P.S., 1982. *Stochastic Models, Estimation, and Control,* vol. 2, New York: Academic Press.

Maybeck, P.S., 1979. *Stochastic Models, Estimation, and Control,* vol. 1, New York: Academic Press.

Mazaika, P.K., 1987. "A mathematical model of the Boltzmann machine," *IEEE First International Conference on Neural Networks,* vol. III, pp. 157–163, San Diego.

McBride, L.E., Jr., and K.S. Narendra, 1965. "Optimization of time-varying systems," *IEEE Transactions on Automatic Control,* vol. AC-10, pp. 289–294.

McCulloch, W.S., 1988. *Embodiments of Mind,* Cambridge, MA: MIT Press.

McCulloch, W.S., and W. Pitts, 1943. "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics,* vol. 5, pp. 115–133.

McLachlan, G.J., and T. Krishnan, 1997. *The EM Algorithm and Extensions,* New York: Wiley (Interscience).

McQueen, J., 1967. "Some methods for classification and analysis of multivariate observations," *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability,* vol. 1, pp. 281–297, Berkeley, CA: University of California Press.

Mead, C.A., 1989. *Analog VLSI and Neural Systems,* Reading, MA: Addison-Wesley.

Mendel, J.M., 1995. *Lessons in Estimation Theory for Signal Processing, Communications, and Control.* Englewood Cliffs, NJ: Prentice Hall.

Mennon, A., K. Mehrotra, C.K. Mohan, and S. Ranka, 1996. "Characterization of a class of sigmoid functions with applications to neural networks," *Neural Networks,* vol. 9, pp. 819–835.

Mercer, J., 1909. "Functions of positive and negative type, and their connection with the theory of integral equations," *Transactions of the London Philosophical Society (A),* vol. 209, pp. 415–446.

Metropolis, N., A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, 1953. Equations of state calculations by fast computing machines, *Journal of Chemical Physics,* vol. 21, pp. 1087–1092.

Meyer, Y., 1993. *Wavelets: Algorithms and Applications,* SIAM (translated from French and revised by R.D. Ryan), Philadelphia: Society for Industrial and Applied Mathematics.

Micchelli, C.A., 1986. "Interpolation of scattered data: Distance matrices and conditionally positive definite functions," *Constructive Approximation,* vol. 2, pp. 11–22.

Miller, D., A.V. Rao, K. Rose, and A. Gersho, 1996. "A global optimization technique for statistical classifier design," *IEEE Transactions on Signal Processing,* vol. 44, pp. 3108–3122.

Miller, D., and K. Rose, 1994. "Combined source-channel vector quantization using deterministic annealing," *IEEE Transactions on Communications,* vol. 42, pp. 347–356.

Miller, R., 1987. "Representation of brief temporal patterns, Hebbian synapses, and the left-hemisphere dominance for phoneme recognition," *Psychobiology,* vol. 15, pp. 241–247.

Minsky, M.L., 1986. Society of Mind, New York: Simon and Schuster.

Minsky, M.L., 1967. *Computation: Finite and Infinite Machines.* Englewood Cliffs, NJ: Prentice-Hall.

Minsky, M.L., 1954. "Theory of neural-analog reinforcement systems and its application to the brain-model problem," Ph.D. thesis, Princeton University, Princeton, NJ.

Minsky, M.L., and S.A. Papert, 1988. *Perceptrons,* expanded edition, Cambridge, MA: MIT Press.

Minsky, M.L., and S.A. Papert, 1969. *Perceptrons,* Cambridge, MA: MIT Press.

Minsky, M.L., and O.G. Selfridge, 1961. "Learning in random nets," *Information Theory, Fourth London Symposium,* London: Buttenvorths.

Møller, M.F., 1993. "A scaled conjugate gradient algorithm for fast supervised learning," *Neural Networks,* vol. 6, pp. 525–534.

Moody, J., and C.J. Darken, 1989. "Fast learning in networks of locally-tuned processing units," *Neural Computation,* vol. 1, pp. 281–294.

Morita, M., 1993. "Associative memory with nonmonotonic dynamics," *Neural Networks,* vol. 6, pp. 115–126.

Morozov, V.A., 1993. *Regularization Methods for Ill-Posed Problems,* Boca Raton, FL: CRC Press.

Müller, K., A. Ziehe, N. Murata, and S. Amari, 1998. "On-line learning in switching and drifting environments with application to blind source separation," in D. Saad, ed., *On-line Learning in Neural Networks,* pp. 93–110, Cambridge, U.K., and New York: Cambridge University Press.

Mumford, D., 1994. "Neural architectures for pattern-theoretic problems," in C. Koch and J. Davis, eds., *Large-Scale Theories of the Cortex,* pp. 125–152, Cambridge, MA: MIT Press.

Murata, N., 1998. "A statistical study of on-line learning," in D. Saad, ed., *On-line Learning in Neural Networks,* pp. 63–92, Cambridge, U.K., and New York: Cambridge University Press.

Murray, M.K., and J.W. Rice, 1993. *Differential Geometry and Statistics,* New York: Chapman and Hall.

Nadal, J.-P., and N. Parga, 1997, "Redundancy reduction and independent component analysis: Conditions on cumulants and adaptive approaches," *Neural Computation,* vol. 9, pp. 1421–1456.

Nadal, J.-P., and N. Parga, 1994, "Nonlinear neurons in the low-noise limit: A factorial code maximizes information transfer, *Network,* vol. 5, pp. 565–581.

Nadaraya, E.A. 1965. "On nonparametric estimation of density functions and regression curves," *Theory of Probability and its Applications*, vol. 10, pp. 186–190.

Nadaraya, É.A., 1964. "On estimating regression," *Theory of Probability and its Applications*, vol. 9, issue 1, pp. 141–142.

Narendra, K.S., and A.M. Annaswamy, 1989. *Stable Adaptive Systems*, Englewood Cliffs, NJ: Prentice Hall.

Narendra, K.S., and K. Parthasarathy, 1990. "Identification and control of dynamical systems using neural networks," *IEEE Transactions on Neural Networks*, vol. 1, pp. 4–27.

Neal, R.M., 1995. *Bayesian Learning for Neural Networks*, Ph.D. Thesis, University of Toronto, Canada.

Neal, R.M., 1993. "Bayesian learning via stochastic dynamics," *Advances in Neural Information Processing Systems*, vol. 5, pp. 475–482, San Mateo, CA: Morgan Kaufmann.

Neal, R.M., 1992. "Connectionist learning of belief networks," *Artificial Intelligence*, vol. 56, pp. 71–113.

Nelsen, R.B., 2006. *An Introduction to Copulas*, 2d ed., New York: Springer.

Newcomb, S., 1886. "A generalized theory of the combination of observations so as to obtain the best result," *American Journal of Mathematics*, vol. 8, pp. 343–366.

Newell, A., and H.A. Simon, 1972. *Human Problem Solving*, Englewood Cliffs, NJ: Prentice-Hall.

Nguyen, D., and B. Widrow, 1989. "The truck backer-upper: An example of self-learning in neural networks," *International Joint Conference on Neural Networks*, vol. II, pp. 357–363, Washington, DC.

Nie, J., and S. Haykin, 1999. "A Q-learning-based dynamic channel assignment technique for mobile communication systems," *IEEE Transactions on Vehicular Technology*, vol. 48, p. 1676–1687.

Nilsson, N.J., 1980. *Principles of Artificial Intelligence*, New York: Springer-Verlag.

Nilsson, N.J., 1965. *Learning Machines: Foundations of Trainable Pattern-Classifying Systems*, New York: McGraw-Hill.

Niyogi, P., and F. Girosi, 1996. "On the relationship between generalization error, hypothesis complexity, and sample complexity for radial basis functions," *Neural Computation*, vol. 8, pp. 819–842.

Novikoff, A.B.J., 1962. "On convergence proofs for perceptrons," in *Proceedings of the Symposium on the Mathematical Theory of Automata*, pp. 615–622, Brooklyn, NY: Polytechnic Institute of Brooklyn.

Nørgaard, M., N.K. Poulsen, and O. Ravn, 2000. "New developments in state estimation for non-linear systems," *Automatica*, vol. 36, pp. 1627–1638.

Obermayer, K., H. Ritter, and K. Schulten, 1991. "Development and spatial structure of cortical feature maps: A model study," *Advances in Neural Information Processing Systems*, vol. 3, pp. 11–17, San Mateo, CA: Morgan Kaufmann.

Oja, E., 1992. "Principal components, minor components, and linear neural networks," *Neural Networks*, vol. 5, 927–936.

Oja, E., 1983. *Subspace Methods of Pattern Recognition*, Letchworth, England: Research Studies Press.

Oja, E., 1982. "A simplified neuron model as a principal component analyzer," *Journal of Mathematical Biology*, vol. 15, pp. 267–273.

Oja, E., and J. Karhunen, 1985. "A stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix," *Journal of Mathematical Analysis and Applications*, vol. 106, pp. 69–84.

Olshausen, B.A., and D.J. Field, 1997. Sparse coding with an overcomplete basis set: A strategy employed by VI? *Vision Research*, vol. 37, pp. 3311–3325.

Olshausen, B.A., and D.J. Field, 1996. "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, pp. 607–609.

Omlin, C.W., and C.L. Giles, 1996. "Constructing deterministic finite-state automata in recurrent neural networks," *Journal of the Association for Computing Machinery*, vol. 43, pp. 937–972.

Omura, J.K., 1969. "On the Viterbi decoding algorithm," *IEEE Trans. Information Theory*, vol. IT-15, pp. 177–179.

Opper, M., 1996. "Online versus offline learning from random examples: General results," *Phys. Rev. Lett.*, vol. 77, pp. 4671–4674.

Orr, G.B., and K. Müller, 1998. *Neural Networks: Tricks of the Trade* (Outgrowth of a 1996 NIPS Workshop), Berlin and New York: Springer.

Osuna, E., R. Freund, and F. Girosi, 1997. "An improved training algorithm for support vector machines," *Neural Networks for Signal Processing* VII, Proceedings of the 1997 IEEE Workshop, pp. 276–285, Amelia Island, FL.

Ott, E., 1993. *Chaos in Dynamical Systems,* Cambridge, MA: Cambridge University Press.

Packard, N.H., J.P. Crutchfield, J.D. Farmer, and R.S. Shaw, 1980. "Geometry from a time series," *Physical Review Letters,* vol. 45, pp. 712–716.

Palm, G., 1982. *Neural Assemblies: An Alternative Approach,* New York: Springer-Verlag.

Papoulis, A., 1984. *Probability, Random Variables, and Stochastic Processes,* 2d ed., New York: McGraw-Hill.

Parisi, G., 1988. *Statistical Field Theory,* Reading, MA: Addison-Wesley.

Park, J., and I.W. Sandberg, 1991. "Universal approximation using radial-basis-function networks," *Neural Computation,* vol. 3, pp. 246–257.

Parker, D.B., 1987. "Optimal algorithms for adaptive networks." Second order back propagation, second order direct propagation and second order Hebbian learning." *IEEE 1st International Conference on Neural Networks*, vol. 2, pp. 593–600, San Diego, CA.

Parker, T.S., and L.O., Chua, 1989. *Practical Numerical Algorithms for Chaotic Systems,* New York: Springer.

Parzen, E., 1962. "On estimation of a probability density function and mode," *Annals of Mathematical Statistics,* vol. 33, pp. 1065–1076.

Paulin, M.G., 1997. "Neural representations of moving systems," *International Journal of Neurobiology,* vol. 41, pp. 515–533.

Pavlov, I.P., 1927. *Conditional Reflexes: An Investigation of the Physiological Activity of the Cerebral Cortex,* (Translation from the Russian by G.V. Anrep), New York: Oxford University Press.

Pearl, J., 1988. *Probabilistic Reasoning in Intelligent Systems,* San Mateo, CA: Morgan Kaufmann. (revised 2nd printing, 1991).

Pearlmutter, B.A., 1994. "Fast exact multiplication by the Hessian," *Neural Computation,* vol. 6, issue 1, pp. 147–160.

Pearson, K., 1901. "On lines and planes of closest fit to systems of points in space," *Philosophical Magazine,* vol. 2, pp. 559–572.

Peretto, P. 1984. "Collective properties of neural networks: A statistical physics approach," *Biological Cybernetics,* vol. 50, pp. 51–62.

Peretto, P., and J.-J Niez, 1986. "Stochastic dynamics of neural networks," *IEEE Transactions on Systems, Man, and Cybernetics,* vol. SMC-16, pp. 73–83.

Perrin, D., 1990. "Finite automata," in J. van Leeuwen, ed., *Handbook of Theoretical Computer Science, Volume B: Formal Models and Semantics,* Chapter 1, pp. 3–57, Cambridge, MA: MIT Press.

Pham, D.T., and P. Garrat, 1997. "Blind separation of mixture of independent sources through a quasi-maximum likelihood approach," *IEEE Transactions on Signal Processing,* vol. 45, pp. 1712–1725.

Pham, D.T., P. Garrat, and C. Jutten, 1992. "Separation of a mixture of independent sources through a maximum likelihood approach," *Proceedings of EUSIPCO,* pp. 771–774.

Phillips, D., 1962. "A technique for the numerical solution of certain integral equations of the first kind," *Journal of Association for Computing Machinery,* vol. 9, pp. 84–97.

Pineau, J., G. Gordon, and S. Thrun, 2006. "Anytime point-based approximations for large POMDPs." *Journal of Artificial Intelligence Research,* Vol. 27, pp. 335–380.

Pineda, F.J., 1988a. "Generalization of backpropagation to recurrent and higher order neural networks," in *Neural Information Processing Systems,* D.Z. Anderson, ed., pp. 602–611, New York: American Institute of Physics.

Pineda, F.J., 1988b. "Dynamics and architecture in neural computation," *Journal of Complexity,* vol. 4, pp. 216–245.

Pineda, F.J., 1987. "Generalization of back-propagation to recurrent neural networks," *Physical Review Letters,* vol. 59, pp. 2229–2232.

Pitts, W., and W.S. McCulloch, 1947. "How we know universals: The perception of auditory and visual forms," *Bulletin of Mathematical Biophysics,* vol. 9, pp. 127–147.

Pizurica, A., and W. Phillips, 2006. "Estimating the probability of the presence of a signal of interest in multiresolution single- and multiband image denoising," *IEEE Trans. on Image Processing,* vol. 15, pp. 654–665.

Platt, J., 1999. "Fast training of support vector machines using sequential minimal optimization," in B. Schölkopf, C.J.C. Burges, and A.J. Smola, eds., *Advances in Kernel Methods—Support Vector Learning,* pp. 185–208, Cambridge, MA: MIT Press.

Poggio, T., and F. Girosi, 1990a. "Networks for approximation and learning," *Proceedings of the IEEE,* vol. 78, pp. 1481–1497.

Poggio,T., and F. Girosi, 1990b." Regularization algorithms for learning that are equivalent to multilayer networks," *Science,* vol. 247, pp. 978–982.

Poggio, T., and C. Koch, 1985. "Ill-posed problems in early vision: From computational theory to analogue networks," *Proceedings of the Royal Society of London,* Series B, vol. 226, pp. 303–323.

Poggio, T., V. Torre, and C. Koch, 1985. "Computational vision and regularization theory," *Nature,* vol. 317, pp. 314–319.

Polak, E., and G. Ribiére, 1969. "Note sur la convergence de méthodes de directions conjuguées," *Revue Française d' Informatique et de Recherche Opérationnelle* vol. 16, pp. 35–43.

Powell, M.J.D., 1992. "The theory of radial basis function approximation in 1990," in W. Light, ed., *Advances in Numerical Analysis Vol. II: Wavelets, Subdivision Algorithms, and Radial Basis Functions,* pp. 105–210, Oxford: Oxford Science Publications.

Powell, M.J.D., 1988. "Radial basis function approximations to polynomials," *Numerical Analysis 1987 Proceedings,* pp. 223–241, Dundee, UK.

Powell, M.J.D., 1987. "A review of algorithms for nonlinear equations and unconstrained optimization," *ICIAM'87: Proceedings of the First International Conference on Industrial and Applied Mathematics,* Philadelphia, Society for Industrial and Applied Mathematics, pp. 220–264.

Powell, M.J.D., 1985. "Radial basis functions for multivariable interpolation: A review," *IMA Conference on Algorithms for the Approximation of Functions and Data,* pp. 143–167, RMCS, Shrivenham, U.K.

Prechelt, L., 1998. *Early Stopping—But When?* in *Neural Networks: Tricks of the Trade,* ed. G. Orr and K. Müller, Lecture Notes in Computer Science, no. 1524. Berlin: Springer, pp. 55–69.

Preisendorfer, R.W., 1988. *Principal Component Analysis in Meteorology and Oceanography,* New York: Elsevier.

Press, W.H., and S.A. Teukolsky, 1990. "Orthogonal polynomials and Gaussian quadrature with nonclassical weighting functions," *Computers in Physics,* pp. 423–426.

Press, W.H., P.B. Flannery, S.A. Teukolsky, and W.T. Vetterling, 1988. *Numerical Recipes in C: The Art of Scientific Computing,* Cambridge and New York: Cambridge University Press.

Prokhorov, D.V., 2007. "Training recurrent neurocontrollers for real-time applications," *IEEE Trans. Neural Networks,* vol. 16, pp. 1003–1015.

Prokhorov, D.V., 2006. "Training recurrent neurocontrollers for rubustness with derivation-free Kalman filter," *IEEE Trans. Neural Networks,* vol. 17, pp. 1606–1616.

Prokhorov, D.V., L.A. Feldkamp, and I.Y. Tyukin, 2002. "Adaptive behavior with fixed weights in RNN: An overview," *Proceedings of the International Joint Conference on Neural Networks,* Hawaii.

Prokhorov, D., G. Puskorius, and L. Feldkamp, 2001. "Dynamical neural networks for control," in J. Kolen and S. Kremer, eds., *A Field Guide to Dynamical Recurrent Networks,* New York: IEEE Press, pp. 257–289.

Prokhorov, D.V., and D.C. Wunsch, II, 1997. "Adaptive critic designs," *IEEE Transactions on Neural Networks,* vol. 8, pp. 997–1007.

Puskorius, G., and L. Feldkamp, 2001. "Parameter-based Kalman filter training: Theory and implementation," in S. Haykin, ed., *Kalman Filtering and Neural Networks,* New York: Wiley.

Puskorius, G.V., L.A. Feldkamp, and L.I. Davis, Jr., 1996. "Dynamic neural network methods applied to on-vehicle idle speed control," *Proceedings of the IEEE,* vol. 84, pp. 1407–1420.

Puskorius, G.V., and L.A. Feldkamp, 1994. "Neurocontrol of nonlinear dynamical systems with Kalman filter-trained recurrent networks," *IEEE Transactions on Neural Networks,* vol. 5, pp. 279–297.

Puskorius, G.V., and L.A. Feldkamp, 1991. "Decoupled extended Kalman filter training of feedforward layered networks," *International Joint Conference on Neural Networks,* vol. 1, pp. 771–777, Seattle.

Rabiner, L.R., 1989. "A tutorial on hidden Markov models," *Proceedings of the IEEE,* vol. 73, pp. 1349–1387.

Rabiner, L.R., and B.H. Juang, 1986. "An introduction to hidden Markkov models," *IEEE ASSP Magazine,* vol. 3, pp. 4–16.

Ramón y Cajál, S., 1911, *Histologie du Systéms Nerveux de l'homme et des vertébrés,* Paris: Maloine.

Rao, A., D. Miller, K. Rose, and A. Gersho, 1997a. "Mixture of experts regression modeling by deterministic annealing," *IEEE Transactions on Signal Processing,* vol. 45, pp. 2811–2820.

Rao, A., K. Rose, and A. Gersho, 1997b. "A deterministic annealing approach to discriminative hidden Markov model design," *Neural Networks for Signal Processing VII, Proceedings of the 1997 IEEE Workshop,* pp. 266–275, Amelia Island, FL.

Rao, C.R., 1973. *Linear Statistical Inference and Its Applications,* New York: Wiley.

Rao, R.P.N., B.A. Olshausen, and M.S. Lewicki, eds., 2002. *Probabilistics Models of the Brain,* Cambridge, MA: MIT Press.

Rao, R.P.N., and T.J. Sejnowski, 2003. "Self-organizing neural systems based on predictive learning," *Philosophical Transactions of the Royal Society of London,* vol. A.361, pp. 1149–1175.

Rao, R.P.N., and D.H. Ballard, 1999. "Predictive coding in the visual cortex: A functional interpretation of some extra-classical receptive-field effects," *Nature Neuroscience,* vol. 3, pp. 79–87.

Rao, R.P.N., and D.H. Ballard, 1997. "Dynamic model of visual recognition predicts neural response properties in the visual cortex," *Neural Computation,* vol. 9, pp. 721–763.

Rashevsky, N., 1938. *Mathematical Biophysics,* Chicago: University of Chicago Press.

Reed, R.D., and R.J. Marks, II, 1999. *Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks,* Cambridge, MA: MIT Press.

Reif, F., 1965. *Fundamentals of Statistical and Thermal Physics,* New York: McGraw-Hill.

Renals, S., 1989. "Radial basis function network for speech pattern classification," *Electronics Letters,* vol. 25, pp. 437–439.

Rényi, A., 1970. *Probability Theory,* North-Holland, Amsterdam.

Rényi, A. 1960. "On measures of entropy and information," *Proceedings of the 4th Berkeley Symposium on Mathematics, Statistics, and Probability,* pp. 547–561.

Richards, W., ed., 1988. *Natural Computation,* Cambridge, MA: MIT Press.

Rieke, F., D. Warland, R. van Steveninck, and W. Bialek, 1997. *Spikes: Exploring the Neural Code,* Cambridge, MA: MIT Press.

Rifkin, R.M., 2002. *Everything old is new again: A fresh look at historical approaches in machine learning,* Ph.D. thesis, MIT.

Rissanen, J., 1989. *Stochastic Complexity in Statistical Inquiry,* Singapore: World Scientific.

Rissanen, J., 1978. "Modeling by shortest data description," *Automatica,* vol. 14, pp. 465–471.

Ristic, B., S. Arulampalam, and N. Gordon, 2004. *Beyond the Kalman Filter: Particle Filters for Tracking Applications,* Boston: Artech House.

Ritter, H., 2003. "Self-organizing feature maps." In M.A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks,* 2nd edition, pp. 1005–1010.

Ritter, H., 1991. "Asymptotic level density for a class of vector quantization processes," *IEEE Transactions on Neural Networks,* vol. 2, pp. 173–175.

Ritter, H., and T. Kohonen, 1989. "Self-organizing semantic maps," *Biological Cybernetics,* vol. 61, pp. 241–254.

Ritter, H., T. Martinetz, and K. Schulten, 1992. *Neural Computation and Self-Organizing Maps: An Introduction,* Reading, MA: Addison-Wesley.

Robbins, H., and S. Monro, 1951. "A stochastic approximation method," *Annals of Mathematical Statistics,* vol. 22, pp. 400–407.

Robert, C.P., 2001. *The Bayesian Choice,* New York: Springer.

Robert, C.P., and G. Casella, 1999. *Monte Carlo Statistical Methods,* New York: Springer.

Roberts, S., and R. Everson, editors, 2001. *Independent Component Analysis: Principles and Practice,* Cambridge, U.K., and New York: Cambridge University Press.

Rochester, N., J.H. Holland, L.H. Haibt, and W.L. Duda, 1956. "Tests on a cell assembly theory of the action of the brain, using a large digital computer," *IRE Transactions on Information Theory,* vol. IT-2, pp. 80–93.

Rose, K., 1998. "Deterministic annealing for clustering, compression, classification, regression, and related optimization problems," *Proceedings of the IEEE,* vol. 86, pp. 2210–2239.

Rose, K., 1991. *Deterministic Annealing, Clustering, and Optimization,* Ph.D. Thesis, California Institute of Technology, Pasadena, CA.

Rose, K., E. Gurewitz, and G.C. Fox, 1992. "Vector quantization by deterministic annealing," *IEEE Transactions on Information Theory,* vol. 38, pp. 1249–1257.

Rose, K., E. Gurewitz, and G.C. Fox, 1990. "Statistical mechanics and phase transitions in clustering," *Physical Review Letters,* vol. 65, pp. 945–948.

Rosenberg, S., 1997. *The Laplacian on a Riemannian Manifold,* Cambridge, U.K., and New York: Cambridge University Press.

Rosenblatt, F., 1962. *Principles of Neurodynamics,* Washington, DC: Spartan Books.

Rosenblatt, F., 1958. "The Perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review,* vol. 65, pp. 386–408.

Rosenblatt, M., 1970. "Density estimates and Markov sequences," in M. Puri, ed., *Nonparametric Techniques in Statistical Inference,* pp. 199–213, London: Cambridge University Press.

Rosenblatt, M., 1956. "Remarks on some nonparametric estimates of a density function," *Annals of Mathematical Statistics.,* vol. 27, pp. 832–837.

Ross, S.M., 1983. *Introduction to Stochastic Dynamic Programming,* New York: Academic Press.

Roth, Z., and Y. Baram, 1996. "Multi-dimensional density shaping by sigmoids," *IEEE Transactions on Neural Networks,* vol. 7, pp. 1291–1298.

Roussas, G., ed., 1991. *Nonparametric Functional Estimation and Related Topics,* The Netherlands: Kluwer.

Roy, N., and G. Gordon, 2003. "Exponential family PCA for belief compression in POMDPs," *Advances in Neural Information Processing Systems,* vol. 15, pp. 1667–1674.

Roy, S., and J.J. Shynk, 1990. "Analysis of the momentum LMS algorithm," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* vol. ASSP-38, pp. 2088–2098.

Rubin, D.B., 1988. "Using the SIR algorithm to simulate posterior distribution," in J.M. Bernardo, M.H. DeGroot, D.V. Lindley, and A.F.M. Smith, eds., *Bayesian Statistics,* vol. 3, pp. 395–402, Oxford, U.K.: Oxford University Press.

Rubner, J., and K. Schulten, 1990. "Development of feature detectors by self-organization," *Biological Cybernetics,* vol. 62, pp. 193–199.

Rubner, J., and P. Tavan, 1989. "A self-organizing network for principal component analysis," *Europhysics Letters,* vol. 10, pp. 693–698.

Ruderman, D.L., 1997. "Origins of scaling in natural images," *Vision Research,* vol. 37, pp. 3385–3395.

Rueckl, J.G., K.R. Cave, and S.M. Kosslyn, 1989. "Why are 'what' and 'where' processed by separate cortical visual systems? A computational investigation," *J. Cognitive Neuroscience,* vol. 1, pp. 171–186.

Rumelhart, D.E., and J.L. McClelland, eds., 1986. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition,* vol. 1, Cambridge, MA: MIT Press.

Rumelhart, D.E., and D. Zipser, 1985. "Feature discovery by competitive learning," *Cognitive Science,* vol. 9, pp. 75–112.

Rumelhart, D.E., G.E. Hinton, and R.J. Williams, 1986a. "Learning representations of back-propagation errors," *Nature (London),* vol. 323, pp. 533–536.

Rumelhart, D.E., G. E. Hinton, and R.J. Williams, 1986b. "Learning internal representations by error propagation," in D.E. Rumelhart and J.L. McCleland, eds., vol 1, Chapter 8, Cambridge, MA: MIT Press.

Russell, S.J., and P. Novig, 1995. *Artificial Intelligence: A Modem Approach,* Upper Saddle River, NJ: Prentice Hall.

Saarinen, S., R.B. Bramley, and G. Cybenko, 1992. "Neural networks, backpropagation, and automatic differentiation," in *Automatic Differentiation of Algorithms: Theory, Implementation, and Application,* A. Griewank and G.F. Corliss, eds., pp. 31–42, Philadelphia: SIAM.

Saarinen, S., R. Bramley, and G. Cybenko, 1991. "The numerical solution of neural network training problems," *CRSD Report No. 1089,* Center for Supercomputing Research and Development, University of Illinois, Urbana, IL.

Saerens, M., and A. Soquet, 1991. "Neural controller based on back-propagation algorithm," *IEE Proceedings (London), Part F,* vol. 138, pp. 55–62.

Salomon, R., and J.L. van Hemmen, 1996. "Accelerating backpropagation through dynamic self-adaptation," *Neural Networks,* vol. 9, pp. 589–601.

Samuel, A.L., 1959. "Some studies in machine learning using the game of checkers," *IBM Journal of Research and Development,* vol. 3, pp. 211–229.

Sandberg, I.W., 1991. "Structure theorems for nonlinear systems," *Multidimensional Systems and Signal Processing,* vol. 2, pp. 267–286.

Sandberg, I.W., L. Xu, 1997a. "Uniform approximation of multidimensional myopic maps," *IEEE Transactions on Circuits and Systems,* vol. 44, pp. 477–485.

Sandberg, I.W., and L. Xu, 1997b. "Uniform approximation and gamma networks," *Neural Networks,* vol. 10, pp. 781–784.

Sanger, T.D., 1990. "Analysis of the two-dimensional receptive fields learned by the Hebbian algorithm in response to random input," *Biological Cybernetics,* vol. 63, pp. 221–228.

Sanger, T.D., 1989a. "An optimality principle for unsupervised learning," *Advances in Neural Information Processing Systems,* vol. 1, pp. 11–19, San Mateo, CA: Morgan Kaufmann.

Sanger, T.D., 1989b. "Optimal unsupervised learning in a single-layer linear feedforward neural network," *Neural Networks,* vol. 12, pp. 459–473.

Sauer, T., J.A. Yorke, and M. Casdagli, 1991. "Embedology," *Journal of Statistical Physics,* vol. 65, pp. 579–617.

Saul, L.K., and M.I. Jordan, 1995. "Boltzmann chains and hidden Markov models," *Advances in Neural Information Processing Systems,* vol. 7, pp. 435–442.

Scharf, L.L., 1991. *Statistical Signal Processing: Detection, Estimation, and Time Series Analysis,* Reading, MA: Addison-Wesley.

Schei, T.S., 1997. "A finite-difference method for linearization in nonlinear estimation algorithms," *Automatica,* vol. 33, pp. 2051–2058.

Schiffman, W.H., and H.W. Geffers, 1993. "Adaptive control of dynamic systems by back propagation networks," *Neural Networks,* vol. 6, pp. 517–524.

Schölkopf, B., and A.J. Smola, 2002. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond,* Cambridge, MA: MIT Press.

Schölkopf, B., A.J. Smola, and K. Müller, 1998. "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation,* vol. 10, pp. 1299–1319.

Schölkopf, B., 1997. *Support Vector Learning,* Munich, Germany: R. Oldenbourg Verlag.

Schraudolf, N.N., J. Yu, and S. Günter, 2007. "A stochastic quasi-Newton method for on-line convex optimization," *Proceedings of 11th Intl. Conf. Artificial Intelligence and Statistics,* Puerto Rico, pp. 433–440.

Schraudolph, N.N., 2002. "Fast curvature matrix–vector products for second-order gradient descent," *Neural Computation,* vol. 4, pp. 1723–1738.

Schultz, W., 2007. "Reward signals", *Scholarpedia,* vol. 2, issue 6, 16 pages.

Schultz, W., 1998. "Predictive reward signal of dopamine neurons", *J. Neurophysiology,* vol. 80, pp. 1–27.

Schumaker, L.L., 1981, *Spline Functions: Basic Theory,* New York: Wiley.

Seber, G.A.F., and C.J. Wild, 1989. *Nonlinear Regression,* New York: Wiley.

Sejnowski, T.J., 1977a. "Strong covariance with nonlinearly interacting neurons," *Journal of Mathematical Biology,* vol. 4, pp. 303–321.

Sejnowski, T.J., 1977b. "Statistical constraints on synaptic plasticity," *Journal of Theoretical Biology,* vol. 69, pp. 385–389.

Sejnowski, T.J., and C.R. Rosenberg, 1987. "Parallel networks that learn to pronounce English text," *Complex Systems,* vol. 1, pp. 145–168.

Selfridge, O.G., 1958. "Pandemonium: A paradigm for learning," *Mechanization of Thought Processes, Proceedings of a Symposium held at the National Physical Laboratory,* pp. 513–526, London, November. (Reproduced in J.A. Anderson and E. Rosenfeld, editors, *Neurocomputing,* pp. 117–122, Cambridge, MA: MIT Press, 1988.)

Shah, S., and F. Palmieri, 1990. "MEKA—A fast, local algorithm for training feedforward neural networks," *International Joint Conference on Neural Networks,* vol. 3, pp. 41–46, San Diego.

Shamma, S.A., 2001. "On the role of space and time in auditory processing," *Trends in Cognitive Sciences,* vol. 5, pp. 340–348.

Shamma, S., 1989. "Spatial and temporal processing in central auditory networks," in *Methods in Neural Modeling,* C. Koch and I. Segev, Eds., Cambridge, MA: MIT Press.

Shanno, D.F., 1978. "Conjugate gradient methods with inexact line searches," *Mathematics of Operations Research,* vol. 3, pp. 244–256.

Shannon, C.E., 1948. "A mathematical theory of communication," *Bell System Technical Journal,* vol. 27, pp. 379–423, 623–656.

Shannon, C.E., and W. Weaver, 1949. *The Mathematical Theory of Communication,* Urbana, IL: The University of Illinois Press.

Shannon, C.E., and J. McCarthy, eds., 1956. *Automata Studies,* Princeton, NJ: Princeton University Press.

Shawe-Taylor, J., and N. Cristianini, 2004. *Kernel Methods for Pattern Analysis,* Cambridge, U.K., and New York: Cambridge University Press.

Shepherd, G.M., 1988. *Neurobiology,* 2d ed., New York: Oxford University Press.

Shepherd, G.M., ed., 1990. *The Synoptic Organization of the Brain,* 3d ed., New York: Oxford University Press.

Shepherd, G.M., and C. Koch, 1990. "Introduction to synaptic circuits," in *The Synaptic Organization of the Brain,* G.M. Shepherd, ed., pp. 3–31. New York: Oxford University Press.

Sherrington, C.S., 1933. *The Brain and Its Mechanism,* London: Cambridge University Press.

Sherrington, C.S., 1906. *The Integrative Action of the Nervous System,* New York: Oxford University Press.

Sherrington, D., and S. Kirkpatrick, 1975. "Spin-glasses," *Physical Review Letters,* vol. 35, p. 1972.

Shewchuk, J.R., 1994. *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain,* School of Computer Science, Carnegie Mellon University, Pittsburgh, August 4, 1994.

Shore, J.E., and R.W. Johnson, 1980. "Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy," *IEEE Transactions on Information Theory,* vol. IT-26, pp. 26–37.

Shynk, J.J., 1990. "Performance surfaces of a single-layer perceptron," *IEEE Transactions or Neural Networks,* 1, 268–274.

Shynk, J.J. and N.J. Bershad, 1991. Steady-state analysis of a single-layer perceptron based on a system identification model with bias terms," *IEEE Transactions on Circuits and Systems,* vol. CAS-38, pp. 1030–1042.

Si, J., A.G. Barto, W.B. Powell, and D. Wunsch II, eds., 2004. *Handbook of Learning and Approximate Dynamic Programming,* Hoboken, NJ: Wiley-Interscience.

Siegelmann, H.T., B.G. Horne, and C.L. Giles, 1997. "Computational capabilities of recurrent NARX neural networks," *Systems, Man, and Cybernetics, Part B: Cybernetics,* vol. 27, pp. 208–215.

Siegelmann, H.T., and E.D. Sontag, 1991. "Turing computability with neural nets," *Applied Mathematics Letters,* vol. 4, pp. 77–80.

Silver, D., R.S. Sutton, and M. Müller, 2008. "Sample-based learning and search with permanent and transient memories," *Proceedings of the 25th International Conference on Machine Learning,* Helsinki, Finland.

Simmons, J.A., P.A. Saillant, and S.P. Dear, 1992. "Through a bat's ear," *IEEE Spectrum,* vol. 29, issue 3, pp. 46–48, March.

Sindhwani, V., M. Belkin, and P. Niyogi, 2006. "The geometric basis of semi-supervised learning," in O. Chapelle, B. Schölkopf, and A. Zien, eds., *Semi-Supervised Learning,* pp. 217–235, Cambridge, MA: MIT Press.

Sindhwani, V., P. Niyogi, and M. Belkin, 2005. "Beyond the point cloud: From transductive to semi-supervised learning," *Proceedings of the 22nd International Conference on Machine Learning,* Bonn, Germany.

Singh, S., and D. Bertsekas, 1997. "Reinforcement learning for dynamic channel allocation in cellular telephone systems," *Advances in Neural Information Processing Systems,* vol. 9, pp. 974–980, Cambridge, MA: MIT Press.

Singhal, S., and L. Wu, 1989. "Training feed-forward networks with the extended Kalman filter," *IEEE International Conference on Acoustics, Speech, and Signal Processing,* pp. 1187–1190, Glasgow, Scotland.

Sklar, A. (1959), "Fonctions de repartition 'a n dimensions et leurs marges," Publ. Inst. Statist. Univ. Paris 8, pp. 229–231.

Slepian, D., 1973. *Key papers in the development of information theory,* New York: IEEE Press.

Sloane, N.J.A., and A.D. Wyner, 1993. *Claude Shannon: Collected Papers,* New York: IEEE Press.

Slonim, N., N. Friedman, and N. Tishby, 2006. "Multivariate information bottleneck," *Neural Computation,* vol. 18, pp. 1739–1789.

Slotine, J.-J., and W. Li, 1991. Applied Nonlinear Control, Englewood Cliffs, NJ: Prentice Hall.

Smith, T. 2007. *Probabilistic Planning for Robotic Exploration,* Ph.D. thesis, Carnegie Mellon University, Pittsburgh.

Smolensky, P., 1986. "Information processing in dynamical systems: Foundations of Information Theory," in D.E. Rumelhart, J.L. McLelland, and the PDP Research Group, *Parallel Distributed Processing, Volume 1: Foundations,* Chapter 6, pp. 194–281, Cambridge, MA: MIT Press.

Snyder, D.L., 1975. *Random Point Processes,* New York: Wiley-Interscience.

Snyder, D.L., 1972. "Filtering and detection for doubly stochastic Poisson processes," *IEEE Trans. Information Theory,* vol. IT-18, pp. 91–102.

Sompolinsky, H., N. Barkai, and H.S. Seung, 1995. "On-line learning and dicotomies: Algorithms and learning curves," in J.-H. Oh, C. Kwon, and S. Cho, eds., *Neural Networks: The Statistical Mechanics Perspective,* pp. 105–130, Singapore and River Edge, NJ: World Scientific.

Sondik, E.J., 1971. *The Optimal Control of Partially Observable Markov Processes,* Ph.D. thesis, Stanford University.

Sontag, E.D., 1992. "Feedback stabilization using two-hidden-layer nets," *IEEE Transactions on Neural Networks,* vol. 3, pp. 981–990.

Sontag, E.D., 1990. *Mathematical Control Theory: Deterministic Finite Dimensional Systems,* New York: Springer-Verlag.

Sontag, E.D., 1989. "Sigmoids distinguish more efficiently than Heavisides," *Neural Computation,* vol. 1, pp. 470–472.

Southwell, R.V., 1946. *Relaxation Methods in Theoretical Physics,* New York: Oxford University Press.

Specht, D.F., 1991. "A general regression neural network," *IEEE Transactions on Neural Networks,* vol. 2, pp. 568–576.

Sperduti, A., 1997. "On the computational power of recurrent neural networks for structures," *Neural Networks,* vol. 10, pp. 395–400.

Sprecher, D.A., 1965. "On the structure of continuous functions of several variables," *Transactions of the American Mathematical Society,* vol. 115, pp. 340–355.

Steinbuch, K., 1961. "Die Lernmatrix." *Kybernetik,* vol. 1, pp. 36–45.

Steinwart, I., 2003. "Sparseness of support vector machines," *J. Machine Learning Research,* vol. 4, pp. 1071–1105.

Stent, G.S., 1973. "A physiological mechanism for Hebb's postulate of learning," *Proceedings of the National Academy of Sciences, USA,* vol. 70, pp. 997–1001.

Sterling, P., 1990. "Retina," in *The Synoptic Organization of the Brain,* G.M. Shepherd, ed., 3d ed., pp. 170–213, New York: Oxford University Press.

Stewart, G.W., 1973. *Introduction to Matrix Computations,* New York: Academic Press.

Stone, M., 1978. "Cross-validation: A review," *Mathematische Operationsforschung Statistischen, Serie Statistics,* vol. 9, pp. 127–139.

Stone, M., 1974. "Cross-validatory choice and assessment of statistical predictions," *Journal of the Royal Statistical Society,* vol. B36, pp. 111–133.

Strang, G., 1980. *Linear Algebra and its Applications,* New York: Academic Press.

Stroud, A.H., 1971. *Approximate Calculation of Multiple Integrals,* Englewood Cliffs, NJ: Prentice-Hall.

Stroud, A.H., 1966. *Gaussian Quadrature Formulas,* Englewood Cliffs, NJ: Prentice-Hall.

Su, H.-T., and T. McAvoy, 1991. "Identification of chemical processes using recurrent networks," *Proceedings of the 10th American Controls Conference,* vol. 3, pp. 2314–2319, Boston.

Su, H.-T., T. McAvoy, and P. Werbos, 1992. "Long-term predictions of chemical processes using recurrent neural networks: A parallel training approach," *Industrial Engineering and Chemical Research,* vol. 31, pp. 1338–1352.

Suga, N., 1990a. "Cortical computational maps for auditory imaging," *Neural Networks,* vol. 3, pp. 3–21.

Suga, N., 1990b. "Computations of velocity and range in the bat auditory system for echo location," in *Computational Neuroscience,* E.L. Schwartz, ed., pp. 213–231, Cambridge, MA: MIT Press.

Suga, N., 1990c. "Biosonar and neural computation in bats," *Scientific American,* vol. 262, pp. 60–68.

Suga, N., 1985. "The extent to which bisonar information is represented in the bat auditory cortex," in *Dynamic Aspects of Neocortical Function,* G.M. Edelman, W.E. Gall, and W.M. Cowan, eds. pp. 653–695, New York: Wiley (Interscience).

Suga, N., 2003, "Echolocation: Chocleotopic and computational maps," in M.A. Arbib, ed., *The Handbook of Brain Theory and Neural Networks,* 2d edition, pp. 381–387, Cambridge, MA: MIT Press.

Sutton, R.S., 1988. "Learning to predict by the methods of temporal differences," *Machine Learning,* vol. 3, pp. 9–44.

Sutton, R.S., 1984. "Temporal credit assignment in reinforcement learning," Ph.D. dissertation, University of Massachusetts, Amherst, MA.

Sutton, R.S., ed., 1992. Special Issue on Reinforcement Learning, *Machine Learning,* vol. 8, pp. 1–395.

Sutton, R.S., and A.G. Barto, 1998. *Reinforcement Learning: An Introduction,* Cambridge, MA: MIT Press.

Sutton, R.S., and B. Tanner, 2005. "Temporal Difference Networks," *Advances in Neural Information Processing Systems,* vol. 17, pp. 1377–1384.

Suykens, J.A., T. Van Gestel, J. DeBrabanter, B. DeMoor, and J. Vanderwalle, 2002. *Least-Squares Support Vector Machines,* River Edge, NJ: World Scientific.

Suykens, J.A.K., J.P.L. Vandewalle, and B.L.R. DeMoor, 1996. *Artificial Neural Networks for Modeling and Control of Non-Linear Systems,* Dordrecht, The Netherlands: Kluwer.

Takens, F., 1981. "On the numerical determination of the dimension of an attractor," in D. Rand and L.S. Young, eds., *Dynamical Systems and Turbulence,* Annual Notes in Mathematics, vol. 898, pp. 366–381, Berlin: Springer-Verlag.

Tapia, R.A., and J.R. Thompson, 1978. *Nonparametric Probability Density Estimation,* Baltimore: The Johns Hopkins University Press.

Tesauro, G., 1995. "Temporal difference learning and TD-gamma," *Communications of the Association for Computing Machinery,* vol. 38, pp. 58–68.

Tesauro, G., 1994. "TD-Gammon, A self-teaching Backgammon program, achieves master-level play," *Neural Computation,* vol. 6, pp. 215–219.

Tesauro, G., 1992. "Practical issues in temporal difference learning," *Machine Learning,* vol. 8, pp. 257–277.

Tesauro, G., 1989. "Neurogammon wins computer olympiad," *Neural Computation,* vol. 1, pp. 321–323.

Tesauro, G., and R. Janssens, 1988. "Scaling relationships in back-propagation learning," *Complex Systems,* vol. 2, pp. 39–44.

Theodoridis, S., and K. Koutroumbas, 2003. *Pattern Recognition,* 2d ed., Amsterdam and Boston: Academic Press.

Thorndike, E.L., 1911. *Animal Intelligence,* Darien, CT: Hafner.

Thrun, S.B., 1992. "The role of exploration in learning control," in *Handbook of Intelligent Control,* D.A. White and D.A. Sofge, eds., pp. 527–559, New York: Van Nostrand Reinhold.

Tichavsky, P., Z. Koldovsky, and E. Oja, 2006. "Performance analysis of FastICA algorithm and Cramér–Rao bounds for linear independent component analysis," *IEEE Trans. Signal Processing,* vol. 54, pp. 1189–1203.

Tikhonov, A.N., 1973. "On regularization of ill-posed problems," *Doklady Akademii Nauk USSR,* vol. 153, pp. 49–52.

Tikhonov, A.N., 1963. "On solving incorrectly posed problems and method of regularization," *Doklady Akademii Nauk USSR,* vol. 151, pp. 501–504.

Tikhonov, A.N., and V.Y. Arsenin, 1977. *Solutions of Ill-posed Problems,* Washington, DC: W.H. Winston.

Tishby, N., and N. Slonim, 2001. "Data Clustering by Markovian relaxation and the information bottleneck method," *Advances in Neural Information Processing Systems,* vol. 13, pp. 640–646.

Tishby, N., F.C. Pereira, and W. Bialek, 1999. "The information bottleneck method," *Proceedings of the 37th Annual Allerton Conference on Communications, Control and Computing,* pp. 368–377.

Touretzky, D.S., and D.A. Pomerleau, 1989. "What is hidden in the hidden layers?" *Byte,* vol. 14, pp. 227–233.

Tsitsiklis, J.N., 1994. "Asynchronous stochastic approximation and Q-learning," *Machine Learning,* vol. 16, pp. 185–202.

Tsoi, A.C., and A.D. Back, 1994. "Locally recurrent globally feedforward networks: A critical review," *IEEE Transactions on Neural Networks,* vol. 5, pp. 229–239.

Tu, Z.W., X. R. Chen, A.L. Yiulle, and S.C. Zhu, 2005. "Image parsing: Unifying segmentation, detection, and recognition," *International J. Computer Vision,* vol. 63, pp. 113–140.

Turing, A.M., 1952. "The chemical basis of morphogenesis," *Philosophical Transactions of the Royal Society, B,* vol. 237, pp. 5–72.

Turing, A.M., 1950. "Computing machinery and intelligence," *Mind,* vol. 59, pp. 433–460.

Turing, A.M., 1936. "On computable numbers with an application to the Entscheidungs problem," *Proceedings of the London Mathematical Society,* Series 2, vol. 42, pp. 230–265. Correction published in vol. 43, pp. 544–546.

Ukrainec, A.M., and S. Haykin, 1996. "A modular neural network for enhancement of cross-polar radar targets," *Neural Networks,* vol. 9, pp. 143–168.

Ukrainec, A., and S. Haykin, 1992. "Enhancement of radar images using mutual information based unsupervised neural networks," *Canadian Conference on Electrical and Computer Engineering,* pp. MA6.9.1–MA6.9.4, Toronto, Canada.

Uttley, A.M., 1979. *Information Transmission in the Nervous System,* London: Academic Press.

Uttley, A.M., 1970. "The informon: A network for adaptive pattern recognition," *Journal of Theoretical Biology,* vol. 27, pp. 31–67.

Uttley, A.M., 1966. "The transmission of information and the effect of local feedback in theoretical and neural networks," *Brain Research,* vol. 102, pp. 23–35.

Uttley, A.M., 1956. "A theory of the mechanism of learning based on the computation of conditional probabilities," *Proceedings of the First International Conference on Cybernetics,* Namur, Gauthier-Villars, Paris.

Valiant, L.G., 1984. "A theory of the learnable," *Communications of the Association for Computing Machinery,* vol. 27, pp. 1134–1142.

Van Essen, D.C., C.H. Anderson, and D.J. Felleman, 1992. "Information processing in the primate visual system: An integrated systems perspective," *Science,* vol. 255, pp. 419–423.

Van Hulle, M.M., 2005. "Maximum likelihood topographic map formation," *Neural Computation,* vol. 17, pp. 503–513.

Van Hulle, M.M., 2002a. "Kernel-based topographic map formation by local density modeling," *Neural Computation,* vol. 14, pp. 1561–1573.

Van Hulle, M.M., 2002b. "Joint entropy maximization in kernel-based topographic maps," *Neural Computation,* vol. 14, pp. 1887–1906.

Van Hulle, M.M., 1997. "Nonparametric density estimation and regression achieved with topographic maps maximizing the information-theoretic entropy of their outputs," *Biological Cybernetics,* vol. 77, pp. 49–61.

Van Hulle, M.M., 1996. "Topographic map formation by maximizing unconditional entropy: A plausible strategy for "on-line" unsupervised competitive learning and nonparametric density estimation," *IEEE Transactions on Neural Networks,* vol. 7, pp. 1299–1305.

van Laarhoven, P.J.M., and E.H.L. Aarts, 1988. *Simulated Annealing: Theory and Applications,* Boston: Kluwer Academic Publishers.

Van Trees, H.L., 1968. *Detection, Estimation, and Modulation Theory,* Part I, New York: Wiley.

Vapnik, V.N., 1998. *Statistical Learning Theory,* New York: Wiley.

Vapnik, V.N., 1995. *The Nature of Statistical Learning Theory,* New York: Springer-Verlag.

Vapnik, V.N., 1992. "Principles of risk minimization for learning theory," *Advances in Neural Information Processing Systems,* vol. 4, pp. 831–838, San Mateo, CA: Morgan Kaufmann.

Vapnik, V.N., 1982. *Estimation of Dependences Based on Empirical Data,* New York: Springer-Verlag.

Vapnik, V.N., and A.Ya. Chervonenkis, 1971. "On the uniform convergence of relative frequencies of events to their probabilities," *Theoretical Probability and Its Applications,* vol. 17, pp. 264–280.

Vapnik, V.N., and A. Ya. Chervonenkis, 1964. "A note on a class of perceptrons," *Automation and Remote Control,* vol. 25, pp. 103–109.

Venkataraman, S., 1994. "On encoding nonlinear oscillations in neural networks for locomotion," *Proceedings of the 8th Yale Workshop on Adaptive and Learning Systems,* pp. 14–20, New Haven, CT.

Vidyasagar, M., 1997. *A Theory of Learning and Generalization,* London: Springer-Verlag.

Vidyasagar, M., 1993. *Nonlinear Systems Analysis,* 2d ed., Englewood Cliffs, NJ: Prentice Hall.

Viterbi, A.J., 1967. "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory,* vol. IT-13, pp. 260–269.

von der Malsburg, C., 1990a. "Network self-organization," in *An Introduction to Neural and Electronic Networks,* S.F. Zornetzer, J.L. Davis, and C. Lau, eds., pp. 421–432, San Diego: Academic Press.

von der Malsburg, C., 1990b. "Considerations for a visual architecture," in *Advanced Neural Computers,* R. Eckmiller, ed., pp. 303–312, Amsterdam: North-Holland.

von der Malsburg, C., 1981. "The correlation theory of brain function," *Internal Report 81–2,* Department of Neurobiology, Max-Plak-Institute for Biophysical Chemistry, Göttingen, Germany.

von der Malsburg, C., 1973. "Self-organization of orientation sensitive cells in the striate cortex," *Kybernetik,* vol. 14, pp. 85–100.

von der Malsburg, C., and W. Schneider, 1986. "A neural cocktail party processor," *Biological Cybernetics,* vol. 54, pp. 29–40.

von Neumann, J., 1986. *Papers of John von Neumann on Computing and Computer Theory,* W. Aspray and A. Burks, eds., Cambridge, MA: MIT Press.

von Neumann, J., 1958. *The Computer and the Brain,* New Haven, CT: Yale University Press.

von Neumann, J., 1956. "Probabilistic logics and the synthesis of reliable organisms from unreliable components," in *Automata Studies,* C.E. Shannon and J. McCarthy, eds., pp. 43–98, Princeton, NJ: Princeton University Press.

Wahba, G., 1990. *Spline Models for Observational Data,* SIAM.

Wahba, G., D.R. Johnson, F. Gao, and J. Gong, 1995. "Adaptive tuning of numerical weather prediction models: Randomized GCV in three and four dimensional data assimilation," *Monthly Weather Review,* vol. 123, pp. 3358–3369.

Watkins, C.J.C.H., 1989. *Learning from Delayed Rewards,* Ph.D. thesis, University of Cambridge, Cambridge, U.K.

Watkins, C.J.C.H., and P. Dayan, 1992. "Q-learning," *Machine Learning,* vol. 8, pp. 279–292.

Watrous, R.L. 1987. "Learning algorithms for connectionist networks: Applied gradient methods of nonlinear optimization," *First IEEE International Conference on Neural Networks,* vol. 2, pp. 619–627, San Diego.

Watson, G.S., 1964. "Smooth regression analysis," *Sankhyā: The Indian Journal of Statistics, Series A,* vol. 26, pp. 359–372.

Wax, W., and T. Kailath, 1985. "Detection of signals by information theoretic criteria," *IEEE Trans. Acoustics, Speech and Signal Processing,* vol. ASSP32, pp. 387–392.

Webb, A.R., 1994. "Functional approximation by feed-forward networks: A least-squares approach to generalisation," *IEEE Transactions on Neural Networks,* vol. 5, pp. 480–488.

Webb, A.R., and D. Lowe, 1990. "The optimal internal representation of multilayer classifier networks performs nonlinear discriminant analysis," *Neural Networks,* vol. 3, pp. 367–375.

Weierstrass, K., 1885. "Uber die analytische Darstellbarkeit sogenannter willkurlicher Funktionen einer reellen veranderlichen," *Sitzungsberichte der Akademie der Wissenschaften, Berlin,* pp. 633–639, 789–905.

Werbos, P., 2004. "ADP: Goals, opportunities and principles," in J. Si, A.G. Barto, W.B. Powell, and D. Wunsch II, eds., *Handbook of Learning and Approximate Dynamic Programming,* Hoboken, NJ: Wiley-Interscience.

Werbos, P.J., 1992. "Neural networks and the human mind: New mathematics fits humanistic insight," *IEEE International Conference on Systems, Man, and Cybernetics,* vol. 1, pp. 78–83, Chicago.

Werbos, P.J., 1990. "Backpropagation through time: What it does and how to do it," *Proceedings of the IEEE,* vol. 78, pp. 1550–1560.

Werbos, P.J., 1989. "Backpropagation and neurocontrol: A review and prospectus," *International Joint Conference on Neural Networks,* vol. I, pp. 209–216, Washington, DC.

Werbos, P., 1977. "Advanced forecasting for global crisis warning and models of intelligence," *General Systems Yearbook,* vol. 22, pp. 25–38.

Werbos, P.J., 1974. "Beyond regression: New tools for prediction and analysis in the behavioral sciences," Ph.D. thesis, Harvard University, Cambridge, MA.

Whitney, H., 1936. "Differentiable manifolds," *Annals of Mathematics,* vol. 37, pp. 645–680.

Whittaker, E.T., 1923. "On a new method of graduation," *Proceedings of the Edinburgh Mathematical Society,* vol. 41, pp. 63–75.

Widrow, B., N.K. Gupta, and S. Maitra, 1973. "Punish/reward: Learning with a critic in adaptive threshold systems," *IEEE Transactions of Systems, Man, and Cybernetics,* vol. SMC-3, pp. 455–465.

Widrow, B., and M.E. Hoff, Jr., 1960. "Adaptive Switching Circuits," *IRE WESCON Conv. Rec.,* Pt. 4, pp. 96–104.

Widrow, B., and M.A. Lehr, 1990. "30 years of adaptive neural networks: Perceptron, madaline, and back-propagation," *Proceedings of the Institute of Electrical and Electronics Engineers,* vol. 78, pp. 1415–1442.

Widrow, B., P.E. Mantey, L.J. Griffiths, and B.B. Goode, 1967. "Adaptive antenna systems," *Proceedings of the IEEE,* vol. 55, pp. 2143–2159.

Widrow, B., and S.D. Stearns, 1985. *Adaptive Signal Processing,* Englewood Cliffs, NJ: Prentice-Hall.

Widrow, B., and E. Walach, 1996. *Adaptive Inverse Control,* Upper Saddle River, NJ: Prentice Hall.

Wieland, A., and R. Leighton, 1987. "Geometric analysis of neural network capabilities," first *IEEE International Conference on Neural Networks,* vol. III, pp. 385–392, San Diego.

Wiener, N., 1961. *Cybernetics,* 2d ed., New York: Wiley.

Wiener, N., 1958. *Nonlinear Problems in Random Theory,* New York: Wiley.

Wiener, N., 1949. *Extrapolation, Interpolation, and Smoothing of Stationary Time Series with Engineering Applications,* Cambridge, MA: MIT Press. (This was originally issued as a classified National Defense Research Report, February 1942).

Wiener, N., 1948. *Cybernetics: Or Control and Communication in the Animal and the Machine,* New York: Wiley.

Wilks, S.S., 1962. *Mathematical Statistics,* New York: Wiley.

Williams, R.J., and J. Peng, 1990. "An efficient gradient-based algorithm for on-line training of recurrent network trajectories," *Neural Computation,* vol. 2, pp. 490–501.

Williams, R.J., and D. Zipser, 1995. "Gradient-based learning algorithms for recurrent networks and their computational complexity," in Y. Chauvin and D.E. Rumelhart, eds., *Backpropagation: Theory, Architectures, and Applications,* pp. 433–486, Hillsdale, NJ: Lawrence Erlbaum.

Williams, R.J., and D. Zipser, 1989. "A learning algorithm for continually running fully recurrent neural networks," *Neural Computation,* vol. 1, pp. 270–280.

Willshaw, D.J., O.P. Buneman, and H.C. Longuet-Higgins, 1969. "Non-holographic associative memory," *Nature (London),* vol. 222, pp. 960–962.

Willshaw, D.J., and C. von der Malsburg, 1976. "How patterned neural connections can be set up by self-organization," *Proceedings of the Royal Society of London Series B,* vol. 194, pp. 431–445.

Wilson, G.V., and G.S. Pawley, 1988. "On the stability of the travelling salesman problem algorithm of Hopfield and Tank," *Biological Cybernetics,* vol. 58, pp. 63–70.

Wilson, H.R., and J.D. Gowan, 1972. "Excitatory and inhibitory interactions in localized populations of model neurons," *Journal of Biophysics,* vol. 12, pp. 1–24.

Winder, R.O., 1961. "Single stage threshold logic," *Switching Circuit Theory and Logical Design,* AIEE Special Publications, vol. S-134, pp. 321–332.

Winograd, S., and J.D. Cowan, 1963. *Reliable Computation in the Presence of Noise,* Cambridge, MA: MIT Press.

Wood, N.L., and N. Cowan, 1995. "The cocktail party phenomenon revisited: Attention and memory in the classic selective listening procedure of Cherry (1953)," *Journal of Experimental Psychology: General,* vol. 124, pp. 243–262.

Woods, W.A., 1986. "Important issues in knowledge representation," *Proceedings of the Institute of Electrical and Electronics Engineers,* vol. 74, pp. 1322–1334.

Wu, C.F.J., 1983. "On the convergence properties of the EM algorithm," *Annals of Statistics,* vol. 11, pp. 95–103.

Wylie, C.R., and L.C. Barrett, 1982. *Advanced Engineering Mathematics,* 5th ed., New York: McGraw-Hill.

Xu, L., A. Krzyzak, and A. Yuille, 1994. "On radial basis function nets and kernel regression: Statistical consistency, convergency rates, and receptive field size," *Neural Networks,* vol. 7, pp. 609–628.

Xu, L., E. Oja, and C.Y. Suen, 1992. "Modified Hebbian learning for curve and surface fitting," *Neural Networks,* vol. 5, pp. 441–457.

Yee, P., and S. Haykin, 2001. *Regularized Radial Basis Function Networks: Theory and Applications,* New York: Wiley-Interscience.

Yockey, H.P., 1992. *Information Theory and Molecular Biology,* Cambridge, U.K.: Cambridge University Press.

Yom-Tov, E., 2007. "A distributed sequential solver for large-scale SVMs." In L. Bottou, O. Chapelle, D. DeCosta, and J. Weston, editors, *Large-Scale Kernel Machines*, pp.139–154, Cambridge: MIT Press.

Yoshizawa, S., M. Morita, and S. Amari, 1993. "Capacity of associative memory using a nonmonotonic neuron model," *Neural Networks,* vol. 6, pp. 167–176.

Young, P.C., 1984. *Recursive Estimation and Time-Series Analysis,* Berlin and New York: Springer-Verlag.

Younger, S., S. Hockreiter, and P. Conwell, 2001. "Meta-learning with backpropagation," *Proceedings of the International Joint Conference on Neural Networks,* pp. 2001–2006, Washington, DC.

Younger, S., P. Conwell, and N. Cotter, 1999. "Fixed-weight on-line learning," *IEEE Trans. Neural Networks,* vol. 10, pp. 272–283.

Zadeh, L.A., and C.A. Desoer, 1963. *Linear System Theory: The State Space Approach,* New York: McGraw-Hill.

Zames, G., 1981. "Feedback and optimal sensitivity: Model reference transformations, multiplicative seminorms, and approximate inverses," *IEEE Transactions on Automatic Control,* vol. AC-26, pp. 301–320.

Zames, G., and B.A. Francis, 1983. "Feedback, minimax, sensitivity, and optimal robustness," *IEEE Transactions on Automatic Control,* vol. AC-28, pp. 585–601.

Zhou, K., and J.C. Doyle, 1998. *Essentials of Robust Control,* Englewood Cliffs, NJ: Prentice Hall.