Biomedical Signal Analysis

CONTEMPORARY METHODS AND APPLICATIONS

Fabian J. Theis and
Anke Meyer-Bäse

Biomedical Signal Analysis

Biomedical Signal Analysis: Contemporary Methods and Applications

Fabian J. Theis and Anke Meyer-Bäse

The MIT Press
Cambridge, Massachusetts
London, England

This book was set in LaTeX by Fabian J. Theis and Anke Meyer-Bäse.

Printed and bound in the United States of America.

Library of Congress Cataloging-in-Publication Data

Theis, Fabian J.
Biomedical signal analysis: contemporary methods and applications / Fabian J. Theis and Anke Meyer-Bäse.
p. cm.
Includes bibliographical references and index.
ISBN 978-0-262-01328-4 (hardcover : alk. paper) 1. Magnetic resonance imaging. 2. Image processing. 3. Diagnostic imaging. I. Meyer-Bäse, Anke. II. Title.
RC78.7.N83T445 2009
616.07'54-dc22

10 9 8 7 6 5 4 3 2 1

Preface

If we knew what we were doing, it wouldn't be called research, would it?
Albert Einstein (1879–1955)

Our nation's strongest information technology (IT) industry advances are occurring in the life sciences, and it is believed that IT will play an increasingly important role in information-based medicine. Nowadays, the research and economic benefits are found at the intersection of biosciences and information technology, while future years will see a greater adoption of systems-oriented perspectives that will help change the way we think about diseases, their diagnosis, and their treatment. On the other hand, medical imaging is positioned to become a substantial beneficiary of, and a main contributor to, the emerging field of systems biology.

In this important context, innovative projects in the very broad field of biomedical signal analysis are now taking place in medical imaging, systems biology, and proteomics. Medical imaging and biomedical signal analysis are today becoming one of the most important visualization and interpretation methods in biology and medicine. The period since 2000 has witnessed a tremendous development of new, powerful instruments for detecting, storing, transmitting, analyzing, and displaying images. These instruments are greatly amplifying the ability of biochemists, biologists, medical scientists, and physicians to see their objects of study and to obtain quantitative measurements to support scientific hypotheses and medical diagnoses.

An awareness of the power of computer-aided analytical techniques, coupled with a continuing need to derive more information from medical images, has led to a growing application of digital processing techniques for the problems of medicine. The most challenging aspect herein lies in the development of integrated systems for use in the clinical sector. Design, implementation, and validation of complex medical systems require not solely medical expertise but also a tight collaboration between physicians and biologists, on the one hand, and engineers and physicists, on the other.

The very recent years have proclaimed systems biology as the future of biomedicine since it will combine theoretical and experimental approaches to better understand some of the key aspects of human health. The origins of many human diseases, such as cancer, diabetes, and cardiovascular and neural disorders, are determined by the functioning and malfunctioning of signaling components. Understanding how individual

components function within the context of an entire system under a plentitude of situations is extremely important to elucidate the emergence of pathophysiology as a result of interactions between aberrant signaling pathways. This poses a new challenge to today's pharmaceutical industry, where both bioinformatics and systems biology/modeling will play a crucial role. Bioinformatics enables the processing of the enormous amount of data stemming from high-throughput screening methods while modeling helps in predicting possible side effects, as well as determining optimal dosages and treatment strategies. Both techniques aid in a mechanistic understanding of both disease and drug action, and will enable further progress in pharmaceutics by facilitating the transfer from the "black-box" approach to drug discovery.

The goal of the present book is to present a complete range of proven and new methods which play a leading role in the improvement of biomedical signal analysis and interpretation.

Chapter 1 provides an introduction to biomedical signal analysis. It will give an overview on several processing and imaging techniques that will disambiguate mixtures of observed components being observed in the biomedical analysis. Chapter 2 contains a description of methods for spectral transformations. Signal processing techniques that extract the information required to explore complex organization levels are described . Methods such as continuous and discrete Fourier transforms and derived techniques as discrete cosine and sine transform will be elucidated. Chapter 3 deals with principal component analysis, representing an important step in demixing groups of components. The theoretical aspects of blind source separation or independent component analysis (ICA) are described in chapter 4. Several state-of-the-art ICA techniques are explained and many practical issues are presented, since the mixture of components represents a very important paradigm in biosignal processing. Chapter 5 presents a new signal processing technique, the dependent component analysis and practical modeling of relevant architectures. Neural networks have been an emerging technique since the 1980s and have established themselves as an effective parallel processing technique in pattern recognition. The foundations of these networks are described in chapter 6. Besides neural networks, fuzzy logic methods represent one of the most recent techniques applied to data analysis in medical imaging. They are always of interest when we have to deal with imperfect knowledge, when a precise modeling of a system is difficult,

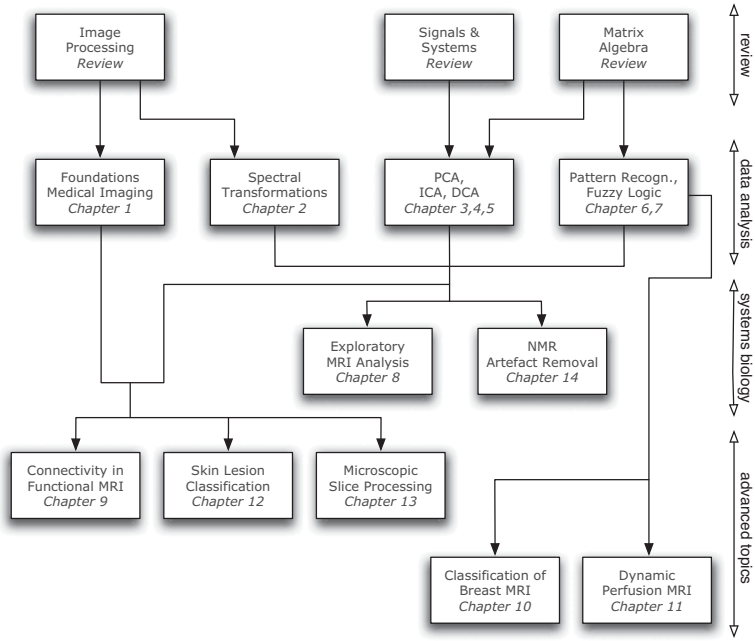


Figure 1
Overview of material covered in this volume and a flow diagram of the chapters.

and when we have to cope with both uncertain and imprecise knowledge. Chapter 7 develops the foundations of fuzzy logic and of several fuzzy c-means clustering and adaptive algorithms. Chapters 8 through 14 show the application of the theoretical tools to practical problems encountered in everyday biosignal processing. Challenging topics ranging from exploratory data analysis and low frequency connectivity analysis in fMRI, to MRI signal processing such as lesion detection in breast MRI, and cerebral time-series analysis in contrast-enhanced perfusion MRI time series are presented, and solutions based on the introduced techniques are outlined and explained in detail. In addition, applications to skin lesion classification, microscopic slice image processing, and automatic labeling, as well as mass spectrometry, are described.

An overview of the chapters is given in order to provide guidance through the material, and thus to address specific needs of very diverse audiences. The basic structure of the book is depicted in figure 1.

The selected topics support several options for reference material and graduate courses aimed to address specific needs of a very diverse audience:

- **Modern biomedical data analysis techniques:** Chapters 2 to 7 provide theoretical aspects and simple implementations of advanced topics. Potential readers: graduate students and bioengineering professionals.
- **Selected topics of computer-assisted radiology:** chapters 1, 2, 3, 4, 6, 7 (section 7.5) and 10 to 14. Potential readers: graduate students, radiologists, and biophysicists.

The book is also designed to be accessible to the independent reader. The table of contents and end-of-chapter summaries should enable the reader to quickly determine which chapters he or she wants to study in most depth. The dependency diagram in figure 1 serves as an aid to the independent reader by helping him or her to determine in what order material in the book may be covered.

The emphasis of the book is on the compilation and organization of a breadth of new approaches, modelling, and applications from signal processing, exploratory data analysis, and systems theory relevant to biosignal modeling. More than 300 references are included and are the basis of an in-depth study. The authors hope that the book will complement existing books on biomedical signal analysis, which focus primarily on time-frequency representations and feature extraction.

Only basic knowledge of digital signal processing, linear algebra, and probability is necessary to fully appreciate the topics considered in this book. Therefore, the authors hope that the book will receive widespread attention in an interdisciplinary scientific community: for those new to the field as a novel synthesis, and as a unique reference tool for experienced researchers.

Acknowledgments

A book does not just "happen" but requires a significant commitment from its author as well as a stimulating and supporting environment. The authors have been very fortunate in this respect.

FT wants to acknowledge the excellent scientific and educational

environment at the University of Regensburg, then at the Max-Planck-Institute of Dynamics and Self-Organization at Göttingen, and finally at the Helmholtz Zentrum München. Moreover, he is deeply grateful for the support of his former professor Elmar W. Lang, who not only directed him into this field of research but has always been a valuable discussion partner since. In addition, FT thanks Theo Geisel for the great opportunities at the MPI at Göttingen, which opened up a whole new area of research and collaboration to him. Similarly, deep thanks are extended to Hans-Werner Mewes for his mentoring and support after FT's start in the field of systems biology. FT acknowledges funding by the BMBF (Bernstein fellow) and the Helmholtz Alliance on Systems Biology (project CoReNe). For the tremendous effort during the copy editing, FT wants to thank Dennis Rickert and Andre Arberer.

A book, particularly one that focuses on a multitude of methods and applications, is not intellectually composed by only two persons. FT wants to thank his collaborators Kurt Stadlthanner, Elmar Lang, Christoph Bauer, Hans Stockmeier, Ingo Keck, Peter Gruber, Harold Gutch, Cédric Févotte, Motoaki Kawanabe, Dominik Hartl, Goncalo García, Carlos Puntonet, and Zaccharias Kohl for the interesting projects, theoretical insights, and great applications. In this book, I have tried to summarize some of our contributions in a concise but well-founded manner. Finally, FT extends his deepest thanks to his family and friends, in particular his wife, Michaela, and his two sons, Jakob and Korbinian, who are the coolest nonscience subjects ever.

The environment in the Department of Electrical and Computer Engineering and the College of Engineering at Florida State University was also conducive to this task. AMB's thanks to Dean Ching-Jen Chen and to the chair, Reginald Perry. Furthermore she would like to thank her graduate students, who used earlier versions of the notes and provided both valuable feedback and continuous motivation.

AMB is deeply indebted to Prof. Heinrich Werner, Thomas Martinetz, Heinz-Otto Peitgen, Dagmar Schipanski, Maria Kallergi, Claudia Berman, Leonard Tung, Jim Zheng, Simon Foo, Bruce Harvey, Krishna Arora, Rodney Roberts, Uwe Meyer-Bäse, Helge Ritter, Henning Scheich, Werner Endres, Rolf Kammerer, DeWitt Sumners, Monica Hurdal, and Mrs. Duo Liu.

AMB is grateful to Prof. Andrew Laine of Columbia University, who provided data, and support and inspired this book project. Her thanks

to Dr. Axel Wismüller from the University of Munich, who is the only "real" bioengineer she has met, who provided her with material and expertise, and who is one of her most helpful colleagues. She also wishes to thank Dr. habil. Marek Ogiela from the AGH University of Science and Technology in Poland for proofreading the sections regarding syntactic pattern recognition and for his valuable expert tips on applications of structural pattern recognition techniques in bioimaging. Finally, watching her daughter Lisa-Marie laugh and play rewarded AMB for the many hours spent with the manuscript.

The efforts of the professional staff at MIT Press, especially Susan Buckley, Katherine Almeida and Robert Prior, deserve special thanks.

We end with some remarks about the form of this book.

Conventions We set technical terms in italics at first use e.g. *new definition.*

Exercises At a number of places, particularly at the end of each theoretical chapter, we include exercises. Attempting the exercises may help you to improve your understanding. If you do not have time to complete the exercises, just making sure that you understand what each exercise is asking will be of benefit.

Experiments and intuitions Often we want you to reflect on your opinion on a particular claim, or to try a small psychological experiment on yourself. In some cases, reading ahead without thinking about the problem or doing the experiment may spoil your intuition about a problem, or may mean that you know what the "correct" result is.

Citations and References As we mentioned above, we have kept citations in the running text to an absolute minimum. Instead, at the end of each chapter, we have included a section titled Further Reading, where we give details of not only the original references where content presented in the chapter first appeared, but also details of how one can follow up certain topics in more depth. These references are also collected in a bibliography at the end of the book.

Index An integrated index is supplied at the end of the book. This is intended to help those who do not read the book from cover to cover to come to grips with the jargon. The index gives the page reference where the term in question was first introduced and defined, as well as page references where the various topics are discussed.

September 2009 Fabian J. Theis and Anke Meyer-Bäse

I METHODS

1 Foundations of Medical Imaging and Signal Recording

Computer processing and analysis of medical images, as well as experimental data analysis of physiological signals, have evolved since the late 1980s from a variety of directions, ranging from signal and imaging acquisition equipment to areas such as digital signal and image processing, computer vision, and pattern recognition.

The most important physiological signals, such as electrocardiograms (ECG), electromyograms (EMG), electroencephalograms (EEG), and magnetoencephalograms (MEG), represent analog signals that are digitized for the purposes of storage and data analysis.

The nature of medical images is very broad; it is as simple as an chest X-ray or as sophisticated as noninvasive brain imaging, such as functional magnetic resonance imaging (fMRI).

While medical imaging is concerned with the interaction of all forms of radiation with tissue and the clinical extraction of relevant information, its analysis encompasses the measurement of anatomical and physiological parameters from images, image processing, and motion and change detection from image sequences.

This chapter gives an overview of biological signal and image analysis, and describes the basic model for computer-aided systems as a common basis enabling the study of several problems of medical-imaging-based diagnostics.

1.1 Biosignal Recording

Biosignals represent space-time records with one or multiple independent or dependent variables that capture some aspect of a biological event. They can be either deterministic or random in nature. Deterministic signals very often can be compact, described by syntactic techniques, while random signals are mainly described by statistical techniques.

In this section, we will present the most common biosignals and the events from which they were generated. Table 1.1 describes these signals.

Biosignals are usually divided into the following groups:

• Bioelectrical (electrophysiological) signals: Electrical and chemical transmissions form the electrophysiological communication between neu-

Table 1.1
Most common biosignals [56].

Event	Signal
Heart electrical conduction at limb surfaces	Electrocardiogram (ECG)
Surface CNS electrical activity	Electroencephalogram (EEG)
Magnetic fields of neural activity	Magnetoencephalogram (MEG)
Muscle electrical activity	Electromyogram (EMG)

ral and muscle cells. Signal transmission between cells takes place as each cell becomes depolarized relative to its resting membrane potential. These changes are recorded by electrodes in contact with the physiological tissue that conducts electricity. While surface electrodes capture bioelectric signals of groups of correlated nerve or muscle cell potentials, intracellular electrodes show the difference in electric potential across an individual cell membrane.

• Biomechanical signals: They are produced by tissue motion or force with highly correlated time-series from sample to sample, enabling an accurate modeling of the signal over long time periods.

• Biomagnetic signals: Body organs produce weak magnetic fields as they undergo electrical changes, and these biosignals can be used to produce three-dimensional images.

• Biochemical signals: They provide functional physiological information and show the levels and changes of various biochemicals. Chemicals such as glucose and metabolites can be also measured.

Electroencephalogram (EEG)

The basis of this method lies in the recording over time of the electric field generated by neural activity through electrodes attached to the scalp. The electrode at each position records the difference in potential between this electrode and a reference one. EEG is employed for spontaneous brain activity, as well as after averaging several presentations of the stimulus. These responses are processed either in the time or in the frequency domain.

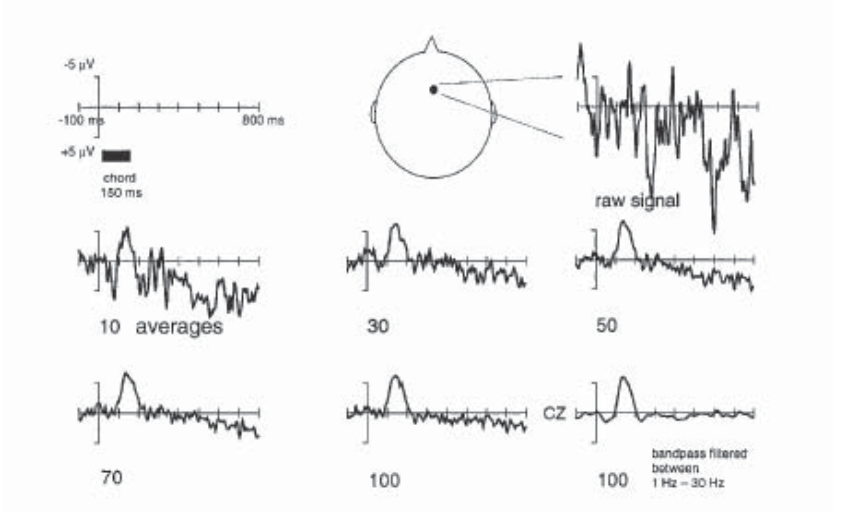


Figure 1.1
EEG signal processing. The EEG signal is displayed in the upper right corner, and the filtered signals averaged is shown below [243].

Magnetoencephalogram (MEG)

The magnetoencephalogram is a technique that records based on ultrasensitive superconducting sensors (SQUIDS), which are placed on a helmet-shaped device. The magnetic fields generated by the neural activity thus allow clinicians to monitor brain activity at different locations and represent different brain functions. As with EEG, the magnetic fields result from coherent activity of dendrites of pyramidal cells. The processing methods are the same as in EEG in regard to both spontaneous and averaged activity. Both EEG and MEG have their own advantages. In MEG, the measured magnetic fields are not affected by the conductivity boundaries, as is the case with EEG. On the other hand, EEG, compared to MEG, enables the localization of all possible orientations of neural sources.

Electrocardiogram (ECG)

The electrocardiogram (ECG) is the recording of the heart's electric activity of repolarization and depolarization of the atrial and ventricular chambers of the heart. Depolarization is the sudden influx of cations

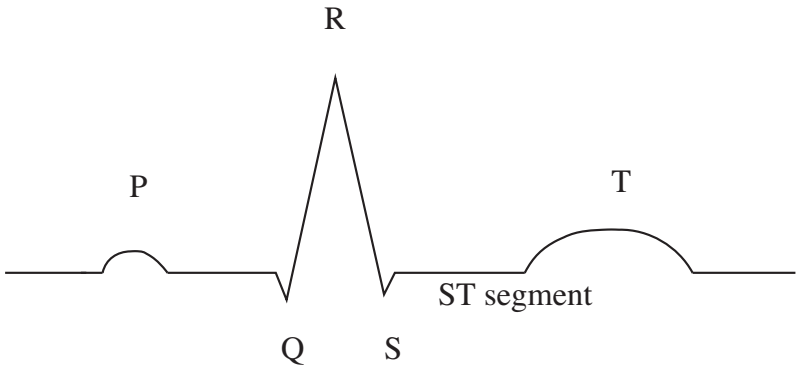


Figure 1.2
Typical waveform of an ECG. The P-wave denotes the atrial depolarization, and the QRS-wave the ventricular depolarization. The T-wave describes the ventricular recovery.

when the membrane becomes permeable, and repolarization is the recovery phase of the ion concentrations returning to normal.

The waveform of the typical ECG is displayed in figure 1.2 with the typical deflections labeled P, QRS, and T, corresponding to atrial contraction (depolarization), ventricular depolarization, and ventricular repolarization, respectively.

The interpretation of an ECG is based on (a) morphology of waves and (b) timing of events and variations observed over many beats.

The diagnostic changes observed in the ECG are permanent or transient occlusion of coronary arteries, heart enlargement, conduction defects, rhythm, and ionic effects.

Electromyogram (EMG)

The electromyogram records the electrical activity of muscles and is used in the clinical environment for the detection of diseases and conditions such as muscular distrophy or disk herniation. There are two types of EMG: intramuscular and surface EMG (sEMG). Intramuscular EMG is performed by inserting a needle which serves as an electrode into the muscle. The action potential represents a waveform of a certain size and shape. Surface EMG (sEMG) is done by placing an electrode on the skin over a muscle in order to detect electrical activity of this muscle.

1.2 Medical Image Analysis

Medical imaging techniques, mostly noninvasive, play an important role in disciplines such as medicine, psychology, and linguistics. The four main medical imaging signals are (1) x-ray transmission, (2) gamma-ray transmission, (3) ultrasound echoes, and (4) nuclear magnetic resonance induction. This is illustrated in table 1.2, where US is ultrasound and MR is magnetic resonance.

Table 1.2
Range of application of the most important radiological imaging modalities [173].

X-rays	Breast, lung, bone
γ-rays	Brain, organ parenchyma, heart function
MR	Soft tissue, disks, brain
US	Fetus, pathological changes, internal organs

The most frequently used medical imaging modalities are illustrated in figure 1.3.

Figure 1.3a and 1.3b illustrate ionizing radiation. Projection radiography and computed tomography are based on x-ray transmission through the body and the selective attenuation of these rays by the body's tissue to produce an image. Since they transmit energy through the body, x-rays belong to transmission imaging modalities, in contrast to emission imaging modalities found in nuclear medicine, where the radioactive sources are localized within the body. They are based on injecting radioactive compounds into the body which finally move to certain regions or body parts, which then emit gamma-rays of intensity proportional to the local concentration of the compounds.

Magnetic resonance imaging is visualized in figure 1.3(c) and is based on the property of nuclear magnetic resonance. This means that protons tends to align themselves with this magnetic field. Regions within the body can be selectively excited such that these protons tip away from the magnetic field direction. The returning of the protons to alignment with the field causes a precession. This produces a radio-frequency (RF) electromagnetic signature which can be detected by an antenna.

Figure 1.3(d) presents the concept of ultrasound imaging: high frequency acoustic waves are sent into the body and the received echoes are used to create an image.

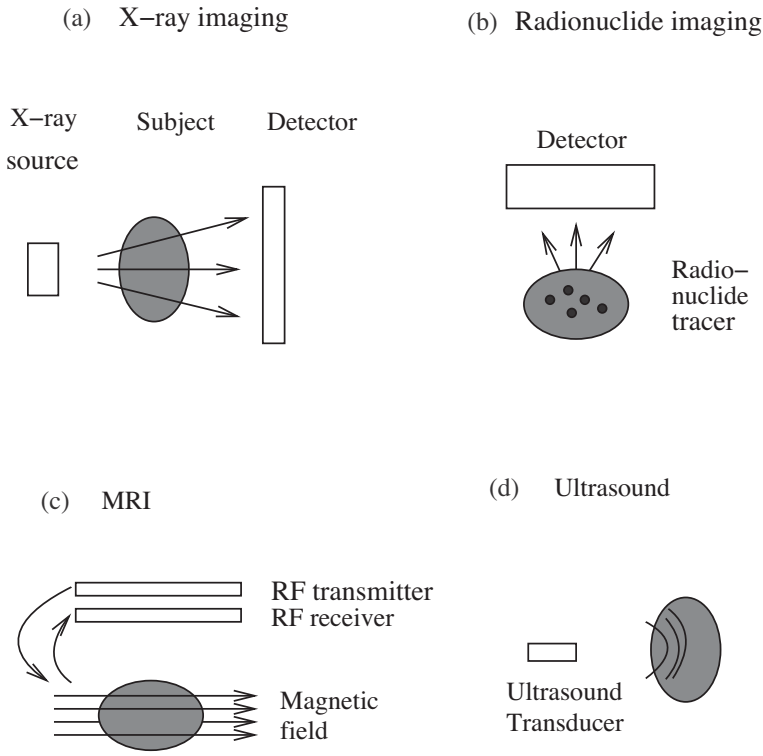


Figure 1.3
Schematic representations of the most frequent used medical imaging modalities [153].

In this chapter, we discuss the four main medical imaging signals introduced in figure 1.3. The medical physics behind these imaging modalities, as well as the image analysis challenges, will be presented. Since the goal of medical imaging is to be automated as much as possible, we will give an overview of computer-aided diagnostic systems in section 1.3. Their main component, the workstation, is described in great detail.

For further details on medical imaging, readers are referred to [51, 164, 280].

Imaging with Ionizing Radiation

X-ray, the most widespread medical imaging modality, was discovered by W. C. Röntgen in 1895. X-rays represent a form of ionizing radiation

with a typical energy range between 25 keV and 500 keV for medical imaging. A conventional radiographic system contains an X-ray tube that generates a short pulse of X-rays that travels through the human body. X-ray photons that are not absorbed or scattered reach the large area detector, creating an image on a film. The attenuation has a spatial pattern. This energy- and material-dependent effect is captured by the basic imaging equation

$$I_d = \int_0^{E_{max}} S_0(E) E \exp\left[- \int_0^d \mu(s;E) ds \right] dE \tag{1.1}$$

where $S_0(E)$ is the X-ray spectrum and $\mu(s;E)$ is the linear attenuation coefficient along the line between the source and the detector; s is the distance from the origin, and d is the source-to-detector distance.

The image quality is influenced by the noise stemming from the random nature of the X-rays or their transmission. Figure 1.4 is a thorax X-ray.

A popular imaging modality is *computed tomography (CT)*, introduced by Hounsfield in 1972, that eliminates the artifacts stemming from overlying tissues and thus hampering a correct diagnosis. In CT, x-ray projections are collected around the patient. CT can be seen as a series of conventional X-rays taken as the patient is rotated slightly around an axis. The films show 2-D projections at different angles of a 3-D body. A horizontal line in a film visualizes a 1-D projection of a 2-D axial cross section of the body. The collection of horizontal lines stemming from films at the same height presents a one-axial cross section. The 2-D cross-sectional slices of the subject are reconstructed from the projection data based on the Radon transform [51], an integral transform introduced by J. Radon in 1917. This transformation collects 1-D projections of a 2-D object over many angles, and the reconstruction is based on a filtered backpropagation, which is the most frequently employed reconstruction algorithm. The projection-slice theorem, which forms the basis of the reconstructions, states that a 1-D Fourier transform of a projection is a slice of the 2-D Fourier transform of the object. Figure 1.5 visualizes this.

The basic imaging equation is similar to conventional radiography, the sole difference being that an ensemble of projections is employed in the reconstruction of the cross-sectional images:

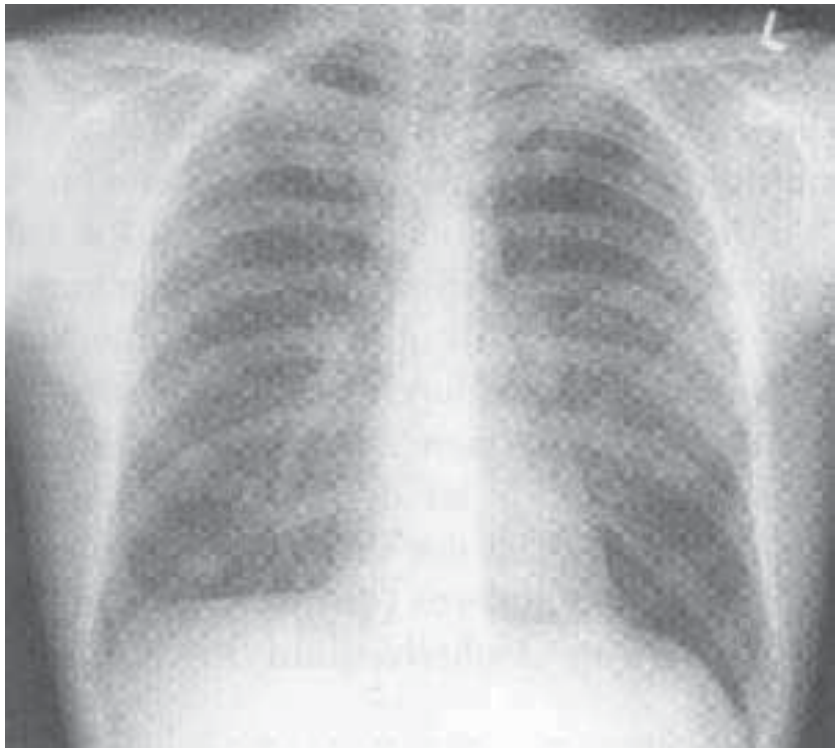


Figure 1.4
Thorax X-ray. (Courtesy of Publicis-MCD-Verlag.)

$$I_d = I_0 \exp\left[-\int_0^d \mu(s;\bar{E})ds\right]dE \tag{1.2}$$

where I_0 is the reference intensity and $\bar{E}$ is the effective energy.

The major advantages of CT over projection radiography are (1) eliminating the superposition of images of structures outside the region of interest; (2) providing a high-contrast resolution such that differences between tissues of physical density of less than 1% become visible; and (3) being a tomographic and potentially 3-D method allowing the analysis of isolated cross-sectional visual slices of the body. The most common artifacts in CT images are aliasing and beam hardening. CT represents an important tool in medical imaging, being used to provide

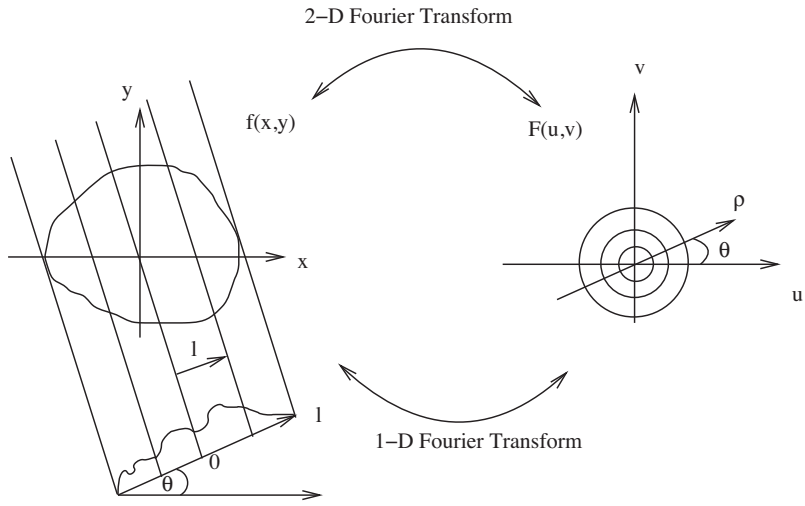


Figure 1.5
Visualization of the projection-slice theorem.

more information than X-rays or ultrasound. It is employed mostly in the diagnosis of cerebrovascular diseases, acute and chronic changes of the lung parenchyma, supporting ECG, and a detailed diagnosis of abdominal and pelvic organs. A CT image is shown in figure 1.6.

Nuclear medicine began in the late 1930s, and many of its procedures use radiopharmaceuticals. Its beginning marked the use of radioactive iodine to treat thyroid disease. Like x-ray imaging, nuclear medicine imaging developed from projection imaging to tomographic imaging. Nuclear medicine is based on ionizing radiation, and image generation is similar to an x-ray's, but with an emphasis on the physiological function rather than anatomy. However, in *nuclear medicine*, radiotracers, and thus the source of emission, are introduced into the body. This technique is a functional imaging modality: the physiology and biochemistry of the body determine the spatial distribution of measurable radiation of the radiotracer. In nuclear medicine, different radiotracers visualize different functions and thus provide different information. In other words, a variety of physiological and biochemical functions can be visualized by different radiotracers. The emissions from a patient are recorded by

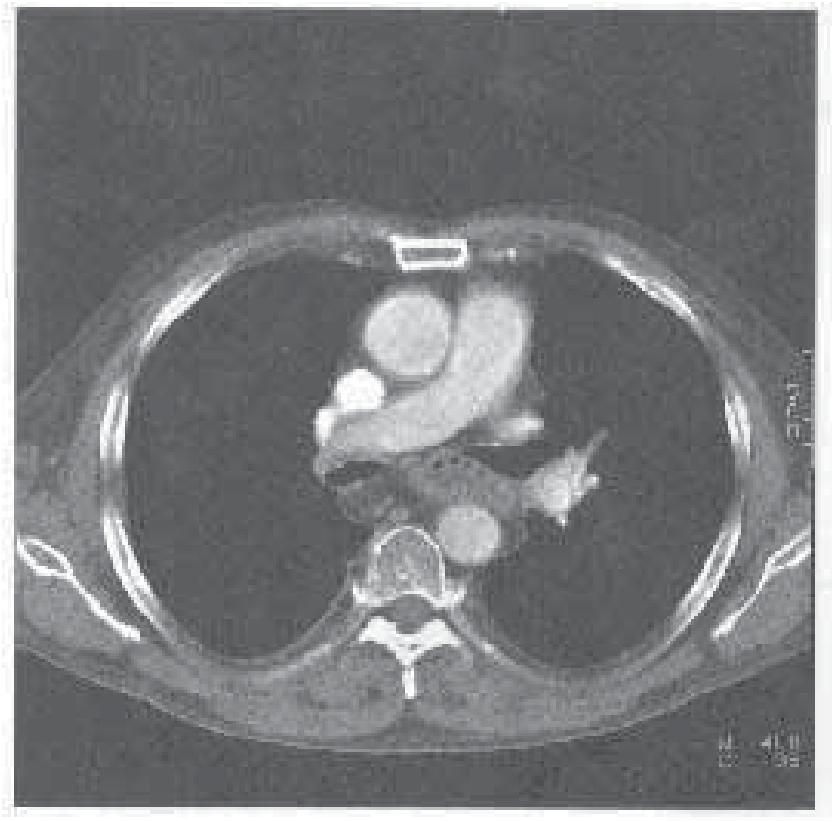


Figure 1.6
CT of mediastinum and lungs. (Courtesy of Publicis-MCD-Verlag.)

scintillation cameras (external imaging devices) and converted into a planar (2-D) image, or cross-sectional images.

Nuclear medicine is relevant for clinical diagnosis and treatment covering a broad range of applications: tumor diagnosis and therapy, acute care, cardiology, neurology, and renal and gastrointestinal disorders.

Based on radiopharmaceutical disintegration, the three basic imaging modalities in nuclear medicine are usually divided into two main areas: (1) planar imaging and *single-photon emission computed tomography (SPECT),* using gamma-emitters as radiotracers, and (2) *positron emission tomography (PET)* using positrons as radiotracers. Projection

imaging, called also planar scintigraphy, uses the Anger scintillation camera, an electronic detection instrument. This imaging modality is based on the detection and estimation of the position of individual scintillation events on the face of an Anger camera. The fundamental imaging equation contains two important components: activity as the desired parameter, and attenuation as an undesired but extremely important additional part.

The fundamental imaging equation is:

$$\varphi(x,y) = \int_{\infty}^{0} \frac{A(x,y,z)}{4\pi z^2} \exp\left(-\int_{z}^{0} \mu(x,y,z';E)dz'\right)dz \tag{1.3}$$

where $A(x,y,z)$ represents the activity in the body and E, the energy of the photon. The image quality is determined mainly by camera resolution and noise stemming from the sensitivity of the system, activity of the injected substance, and acquisition time.

On the other hand, SPECT uses a rotating Anger scintillation camera to obtain projection data from multiple angles. Single-photon emission uses nuclei that disintegrate by emitting a single γ-photon, which is measured with a gamma-camera system. SPECT is a slice-oriented technique, in the sense that the obtained data are tomographically reconstructed to produce a 3-D data set or thin (2-D) slices. This imaging modality can be viewed as a collection of projection images where each is a conventional planar scintigram. The basic imaging equation contains two inseparable terms, activity and attenuation. Before giving the imaging equation, we need some geometric considerations: if x and y are rectlinear coordinates in the plane, the line equation in the plane is given as

$$L(l,\theta) = \{(x,y)|x\cos\theta + y\sin\theta = l\} \tag{1.4}$$

with l being the lateral position of the line and θ the angle of a unit normal to the line. Figure 1.7 visualizes this.

This yields the following parameterization for the coordinates $x(s)$ and $y(s)$:

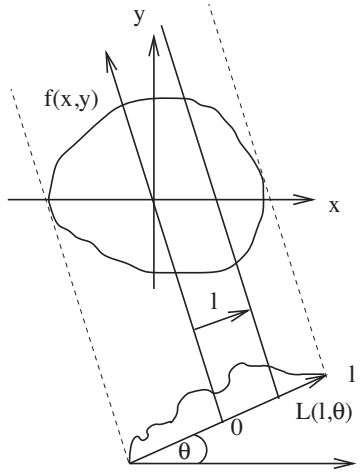


Figure 1.7
Geometric representations of lines and projections.

$$x(s) = l\cos\theta - s\sin\theta \tag{1.5}$$

$$y(s) = l\sin\theta + s\cos\theta \tag{1.6}$$

Thus, the line integral of a function $f(x, y)$ is given as

$$g(l,\theta) = \int_{-\infty}^{\infty} f(x(s), y(s))ds \tag{1.7}$$

For a fixed angle θ, $g(l, \theta)$ represents a projection, while for all l and θ it is called the *2-D radon transformation* of $f(x, y)$.

The imaging equation for SPECT, ignoring the effect of the attenuation term, is:

$$\varphi(l,\theta) = \int_{-\infty}^{\infty} A(x(s), y(s))ds \tag{1.8}$$

where $A(x(s), y(s))$ describes the radioactivity within the 3-D body and is the inverse 2-D Radon transform of $\varphi(l, \theta)$. Therefore, there is no closed-form solution for attenuation correction in SPECT. SPECT represents an important imaging technique by providing an accurate

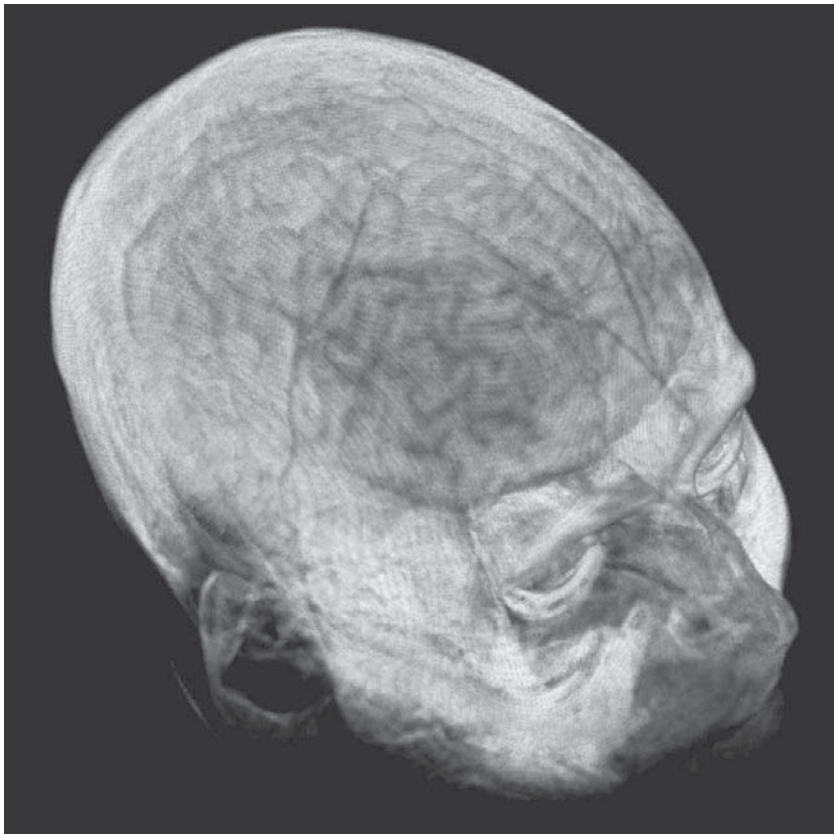


Figure 1.8
SPECT brain study. (Image courtesy Dr. A. Wismüller, Dept. of Radiology, University of Munich.)

localization in 3-D space and is used to provide functional images of organs. Its main applications are in functional cardiac and brain imaging. Figure 1.8 is an image of a SPECT brain study.

PET is a technique having no analogy to other imaging modalities. The radionuclides employed for PET emit positrons instead of γ-rays. These positrons, antiparticles of electrons, are measured and their positions are computed. The reconstruction is produced by using algorithms of filtered backprojection. The imaging equation in PET is similar to that in SPECT, with one difference: The limits of integration for the

attenuation term span the entire body because of the coincidence detection of paired γ-rays, the so-called annihilation photons. The imaging equation is given as

$$\varphi(l,\theta) = K \int_{-R}^{R} A(x(s), y(s))ds \tag{1.9}$$

where K represents a constant that includes the constant factors, such as detector area and efficiency, that influence φ. The image quality in both SPECT and PET is limited by resolution, scatter, and noise. PET has its main clinical application in oncology, neurology, and psychiatry. An important area is neurological disorders, such as early detection of Alzheimers disease, dementia, and epilepsy.

Magnetic Resonance Imaging

Magnetic resonance imaging (MRI) is a non-invasive imaging method used to render images of the inside of the body. Since the late 1970s, it has become one of the key bioimaging modalities in medicine. It reveals pathological and physiological changes in bod tissues as nuclear medicine does, in addition to structural details of organs as CT does.

The MRI signal stems from the nuclear magnetism of hydrogen atoms located in the fat and water of the human body, and is based on the physical principle of nuclear magnetic resonance (NMR). NMR is concerned with the charge and angular momentum possessed by certain nuclei. Nuclei have positive charge and, in the case of an odd atomic number or mass number, an angular momentum Φ. By having spin, these nuclei are NMR-active. Each nucleus that has a spin also has a microscopic magnetic field. When an external electric field is applied, the spins tend to align with that field. This property is called nuclear magnetism. Thus, the spin systems become macroscopically magnetized.

In MR imaging, we look at the macroscopic magnetization by considering a specific spin system (hydrogen atoms) within a sample. The "sample" represents a small volume of tissue (i.e., a voxel). Applying a static magnetic field $\mathbf{B}_0$ causes the spin system to become magnetized, and it can be modeled by a bulk magnetization vector $\mathbf{M}$. In the undisturbed state, $\mathbf{M}$ will reach an equilibrium value $\mathbf{M}_0$ parallel to the direction of $\mathbf{B}_0$, see figure 1.10(a).

It's very important to note that $\mathbf{M}(\mathbf{r}, t)$ is a function of time and

of the 3-D coordinate $\mathbf{r}$ that can be manipulated spatially by external radio-frequency excitations and magnetic fields.

At a given voxel, the value of an MR image is characterized by two important factors: the tissue properties and the scanner imaging protocol. The most relevant tissue properties are the relaxation parameters T_1 and T_2 and the proton density. The proton density is defined as the number of targeted nuclei per unit volume. The scanner software and hardware manipulate the magnetization vector $\mathbf{M}$ over time and space based on the so-called pulse sequence.

In the following text, we will focus on a particular voxel and give the equations of motion for $\mathbf{M}(t)$ as a function of time t. These equations are based on the Bloch equations and describe a precession of the magnetization vector around the external applied magnetic field with a frequency ω_0, which is known as the resonance or Larmor frequency.

The magnetization vector $\mathbf{M}(t)$ has two components:

1. The longitudinal magnetization given by $M_z(t)$, the z-component of $\mathbf{M}(t)$
2. The transverse magnetization vector $M_{xy}(t)$, a complex quantity, which combines two orthogonal components:

$$M_{xy}(t) = M_x(t) + jM_y(t) \tag{1.10}$$

where φ is the angle of the complex number M_{xy}, known as the phase angle, given as

$$\varphi = \tan^{-1} \frac{M_x}{M_y} \tag{1.11}$$

Since $\mathbf{M}(t)$ is a magnetic moment, it will have a torque if an external time-varying magnetic field $\mathbf{B}(t)$ is applied. If this field is static and oriented parallel to the z-direction, then $\mathbf{B}(t) = \mathbf{B}_0$.

The magnetization vector $\mathbf{M}$ precesses if it is initially oriented away from the $\mathbf{B}_0$. The spin system can also be excited by using RF signals, such that RF signals are produced as output by the stimulated system. This RF excitation is achieved by applying $\mathbf{B}_1$ at the Larmor frequency rather than keeping it constant, and allows tracking the position of $\mathbf{M}(t)$. However, the precession is not perpetual, and we will show that there

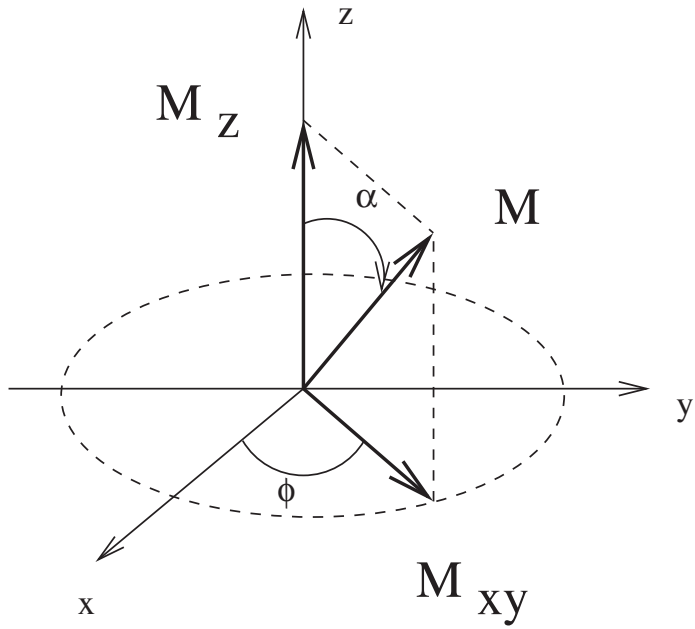


Figure 1.9
The magnetization vector **M** precesses about the z-axis.

are two independent mechanisms to dampen the motion and cause the received signal to vanish: the longitudinal and transversal relaxations.

The RF excitation pushes $\mathbf{M}(t)$ down at an angle α toward the xy-plane if $\mathbf{B}_1$ is along the direction of the y-axis. At $\alpha = 0$, we have $M_z = 0$ and the magnetization vector rotates in the xy-plane with a frequency equal to the Larmor frequency. The $\mathbf{B}_1$ pulse needed for an angle $\alpha = \pi/2$ is called the 90 pulse. The magnetization vector returns to its equilibrium state, and the relaxation process is described by

$$M_z(t) = M_0 \left[1 - \exp\left(-\frac{t}{T_1}\right)\right] \tag{1.12}$$

and depends on the longitudinal or spin-lattice relaxation time (T_1) (See figure 1.9.

Transverse or spin-spin relaxation is the effect of perturbations caused by neighboring spins as they change their phase relative to others. This dephasing leads to a loss of the signal in the receiver antenna. The resulting signal is called free induction decay (FID). The return of the transverse magnetization $\mathbf{M}_{xy}$ to equilibrium is described by

$$M_{xy}(t) = M_{x_0 y_0} \exp\left(-\frac{t}{T_2}\right) \tag{1.13}$$

where T_2 is the spin-spin relaxation time. T_2 is tissue-dependent and produces the contrast in MR images. However, the received signal decays faster than T_2. Local perturbations in the static field $\mathbf{B}_0$ give rise to a faster time constant T_2^*, where $T_2^* < T_2$. Figure 1.10(b) visualizes this situation. The decay associated with the external field effects is modeled by the time constant $T_2^{'}$. The relationship between the three transverse relaxation constants is modelled by

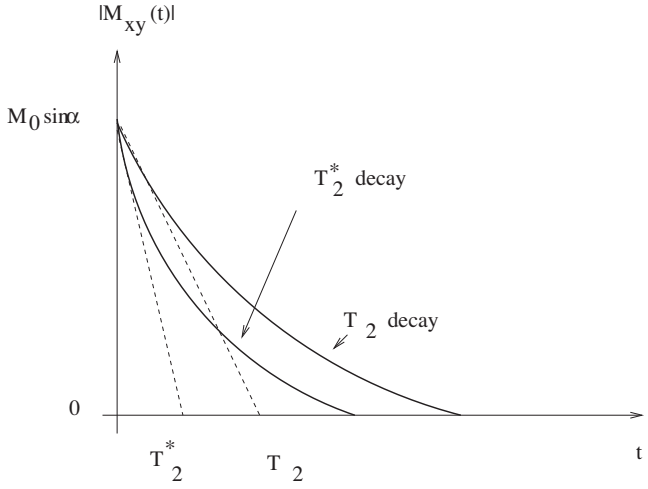
$$\frac{1}{T_2^*} = \frac{1}{T_2} + \frac{1}{T_2^{'}} \tag{1.14}$$

It's important to note that both T_1 and T_2 are tissue-dependent and that for all materials $T_2 \leq T_1$.

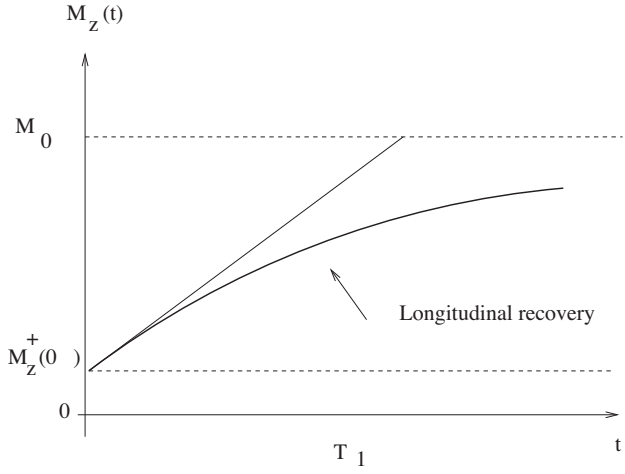
Valuable information is obtained from measuring the temporal course of the T1/T2 relaxation process after applying an RF pulse sequence. This measured time course is converted from the time to the frequency domain based on the Fourier transform. The amplitude in the spectrum appears at the resonance frequency of hydrogen nucleons in water (see figure 1.11).

A contrast between tissues can be seen if the measured signal is different in those tissues. In order to achieve this, two possibilities are available: the intrinsic NMR properties, such as P_D, T_1, and T_2, and the characteristics of the externally applied excitation. It is possible to control the tip angle α and to use sophisticated pulse sequences such as the spin-echo sequence. A 90° pulse has a period of TR seconds (repetition time) and is followed by a 180° pulse after TE seconds (echo time). This second pulse partially rephases the spins and produces an echo signal.

Figure 1.12 shows a brain scan as T_1-weighted, T_2-weighted, and hydrogen density-weighted images.



(a)



(b)

Figure 1.10
(a) Transverse and (b) longitudinal relaxation.

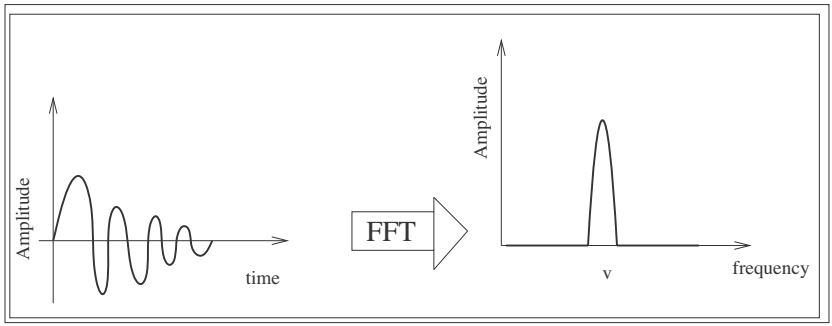


Figure 1.11
Frequency-domain transformation of the measured temporal course. The amplitude in the spectrum is exhibited at the Larmor frequency.

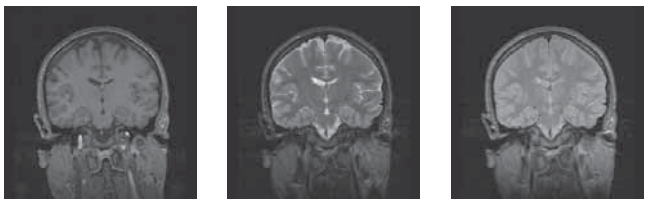


Figure 1.12
Brain MRI showing (a) T_1, (b) T_2, and (c) hydrogen density-weighted images. (Image courtesy Dr. A. Wismüller, Dept. of Radiology, University of Munich.)

"Weighted" means that the differences in intensity observed between different tissues are mainly caused by the differences in T_1, T_2, and P_D, respectively, of the tissues. The basic way to create contrast based on the above parameters is show in table 1.3.

The pixel intensity $I(x, y)$ of an MR image obtained using a spin-echo sequence is given by

$$I(x,y) \propto P_D(x,y) \underbrace{\left(1 - \exp\left[-\frac{T_R}{T_1}\right]\right)}_{\text{T1}-\text{weighting}} \underbrace{\exp\left[-\frac{T_E}{T_2}\right]}_{\text{T2}-\text{weighting}} \tag{1.15}$$

Varying the values of T_R and T_E will control the sensitivity of the signal to the T_1/T_2 relaxation process and will produce different weighted

Table 1.3
Basic way to create contrast depending on P_D, T_1, and T_2.

Contrast	Scanner Parameters
P_D	Long T_R, read FID or use short T_E
T_2	Long T_R, $T_E \approx T_2$
T_1	Read FID or use short T_E, $T_R \approx T_1$

contrast images. If, for example, T_R is much larger than T_1 for all tissues in the *region of interest* (ROI), then the T_1 weighting term converges to zero and there is no sensitivity of the signal to the T_1 relaxation process. The same holds whan T_E is much smaller than T_2 for all tissues. When both T_1 and T_2 sensitivities decrease, the pixel density depends only on the proton density $P_D(x, y)$.

The MR image quality depends not only on contrast but also on sampling and noise. To summarize, the advantages of MRI as an imaging tool are (1) excellent contrasts between the various organs and tumors essential for image quality, (2) the 3-D nature of the image, and (3) the contrast provided by the T_1 and T_2 relaxation mechanism, as one of the most important imaging modalities.

An important technique in MRI is *multispectral magnetic resonance imaging*. A sequence of 3-D MRI images of the same ROI is recorded assuming that the images are correctly registered. This imaging type enables the discrimination of different tissue types.

To further enhance the contrast between tissue types, contrast agents (CA) are used to manipulate the relaxation times. CAs are intravenously administrated, and during that time a signal enhancement is achieved for tissue with increased vascularity.

Functional magnetic resonance imaging (fMRI) is a novel noninvasive technique for the study of cognitive functions of the brain [189]. The basis of this technique is the fact that the MRI signal is susceptible to changes of hemodynamic parameters, such as bood flow, blood volume, and oxygenation, that arise during neural activity. The most commonly used fMRI signal is the blood oxygenation level-dependent (BOLD) contrast. The BOLD temporal response changes when the local deoxyhemoglobin concentration decreases in an area of neuronal activity. This fact is reflected in T_2^*- and T_2-weighted MR images.

The two underlying characteristics of hemodynamic effects are spatial and temporal. While vasculature is mainly responsible for spatial

effects, the temporal effects are responsible for the delay of the detected MR signal changes in response to neural activity and a longer duration of the dispersion of the hemodynamic changes. The temporal aspects impose two different types of fMRI experiments: "block" designs and "event-related" designs. The block designs are characterized by an experimental task performed in an alternating sequence of 20-60 sec blocks. In event-related designs, multiple stimuli are presented randomly and the corresponding hemodynamic response to each is measured. The main concept behind this type of experiment is the almost linear response to multiple stimulus presentations. fMRI, with high temporal and spatial resolution, is a powerful technique for visualizing rapid and fine activation patterns of the human brain. The functional localization is based on the evident correlation between neuronal activities and MR signal changes. As is known from both theoretical estimations and experimental results [187], an activated signal variation appears very low on a clinical scanner. This motivates the application of analysis methods to determine the response waveforms and associated activated regions.

The main advantages of this technique are (1) noninvasive recording of brain signals without any risk of radiation, unlike CT; (2) excellent spatial and temporal resolution, and (3) integration of fMRI with other techniques, such as MEG and EEG, to study the human brain.

fMRI's main feature is to image brain activity in vivo. Therefore its applications lie in the diagnosis, interpretation, and treatment evaluation of clinical disorders of cognitive brain functions. The most important clinical application lies in preoperative planning and risk assessment in intractable focal epilepsy. In pharmacology, fMRI is a valuable tool in determining how the brain is responding to a drug. Furthermore in clinical applications, the importance of fMRI in understanding neurological and psychiatric disorders and refining the diagnosis is growing.

Ultrasound and Acoustic Imaging

Ultrasound is a leading imaging modality and has been extensively studied since the early 1950s. It is a noninvasive imaging modality which produces oscillations of 1 to 10 MHz when passing through soft tissues and fluid.

The cost effectiveness and the portability of ultrasound have made this technique extremely popular. Its importance in diagnostic radiology is unquestionable, enabling the imaging of pathological changes of inner

organs and blood vessels, and supporting breast cancer detection.

The principle of the ultrasonic imaging is very simple: the acoustic wave launched by a transducer into the body interacts with tissue and blood, and some of the energy that is not absorbed returns to the transducer and is detected by it. As a result, "ultrasonic signatures" emerge from the interaction of ultrasound energy with different tissue types that are subsequently used for diagnosis.

The speed of sound in tissue is a function of tissue type, temperature, and pressure. Table 1.4 gives examples of acoustic properties of some materials and biological tissues. Because of scattering, absorption or reflection, an attenuation of the acoustic wave is observed. The attenuation is described by an exponential function of the distance, described by $A(x) = A_0 \exp(-\alpha x)$, where A is the amplitude, A_0 is a constant, α is the attenuation factor, and x is the distance. The important characteristics of the returning signal, such as amplitude and phase, provide pertinent information about the interaction and the type of medium that is crossed. The basic imaging equation is the pulse-echo equation, which gives a relation among the excitation pulse, the transducer face, the object reflectivity, and the received signal.

Ultrasound has the following imaging modes:

- A-mode (amplitude mode): the most simple method that displays the envelope of pulse-echoes versus time. It is mostly used in ophthalmology to determine the relative distances between different regions of the eye, and also in localization of the brain midline or of a myocardial infarction. Figure 1.13 visualizes this aspect.
- B-mode (brightness mode): produced by scanning the transducer beam in a plane, as shown in figure 1.14. It can be used for both stationary and moving structures, such as cardiac valve motion.
- M-mode (motion mode): displays the A-mode signal corresponding to repeated pulses in a separate column of a 2-D image. It is mostly employed in conjunction with ECG for motion of the heart valves.

The two basic techniques used to achieve a better sensitivity of the echoes along the dominant (steered) direction are the following:

- Beam forming: increases the transducer's directional sensitivity
- Dynamic focusing: increases the transducer's sensitivity to a particular point in space at a particular time

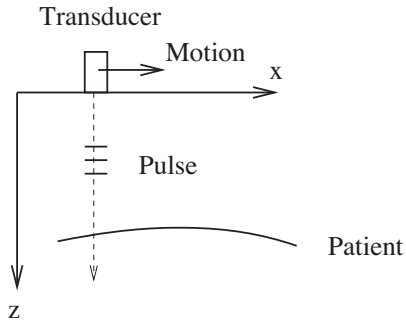


Figure 1.13
A-mode display.

Table 1.4
Acoustical properties of some materials and biological tissues .

Medium	Speed of sound (m/sec)	Impedance (10^6kg/m^2s)	Attenuation (dB/cm at 1MHZ)
Air	344	0.0004	12
Water	1480	1.48	0.0025
Fat	1410	1.38	0.63
Muscle	1566	1.70	1.2-3.3
Liver	1540	1.65	0.94
Bone	4080	7.80	20.0

1.3 Computer-Aided Diagnosis (CAD) Systems

The important advances in computer vision, paired with artificial intelligence techniques and data mining, have facilitated the development of automatic medical image analysis and interpretation. Computer-aided diagnosis (CAD) systems are the result of these research endeavors and provide a parallel second opinion in order to assist clinicians in detecting abnormalities, predicting the diseases progress, and obtaining a differential diagnosis of lesions.

Modern CAD systems are becoming very sophisticated tools with a user-friendly graphical interface supporting the interactions with clinicians during the diagnostic process. They have a multilayer architecture with many modules, such as image processing, databases, and a graphical interface.

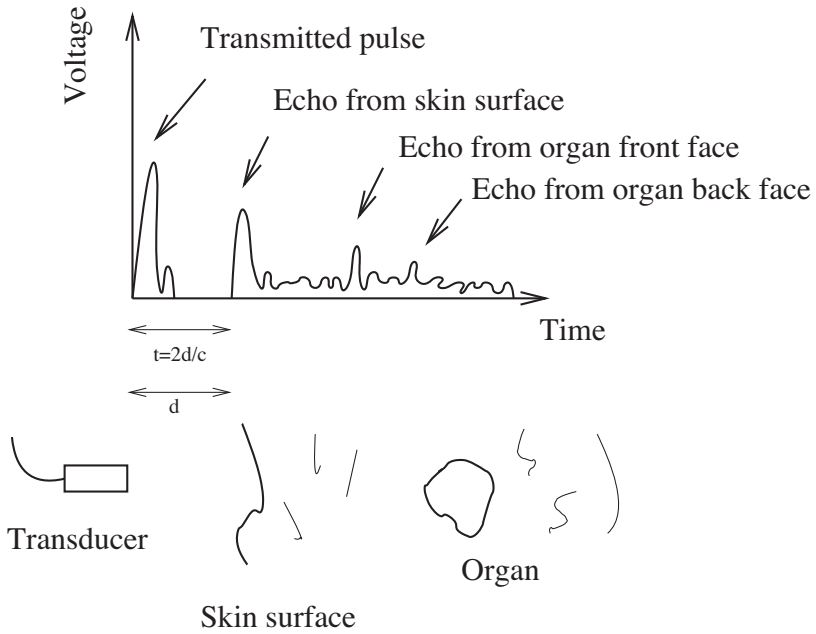


Figure 1.14
B-mode scanner.

A typical CAD system is described in [205]. It has three layers: data layer, application layer, and presentation layer, as shown in figure 1.15.

The functions of each layer are described below.

- Data layer: has a database management system which is responsible for archiving and distributing data
- Application layer: has a management application server for database access and presentation to graphical user interface, a WWW server to ensure remote access to the CAD system, and a CAD workstation for image processing
- Presentation layer: has the Eeb viewer to allow a fast remote access to the system, and at the user site it grants access to the whole system.

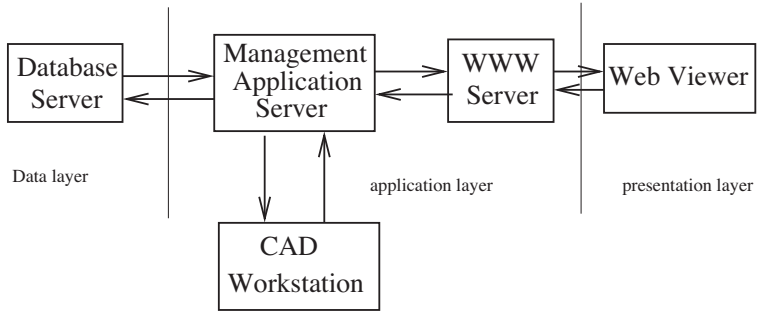


Figure 1.15
Multilayer structure of a CAD system [205].)

CAD Workstation

A typical CAD system's architecture is shown in figure 1.16. It has four important components: (1) image preprocessing, (2) definition of a region of interest (ROI), (3) extraction and selection of features, and (4) classification of the selected ROI.

These basic components are described in the following:

• Image preprocessing: The goal is to improve the quality of the image based on denoising and enhancing the edges of the image or its contrast. This task is crucial for subsequent tasks.

• Definition of an ROI: ROIs are mostly determined by growing seeded regions and by active contour models that correctly approximate the shapes of organ boundaries.

• Extraction and selection of features: These are crucial for the subsequent classification and are based on finding mathematical methods for reducing the sizes of measurements of medical images. Feature extraction is typically carried out in the spectral or spatial domains and considers the whole image content and maps it onto a lower-dimensional feature space. On the other hand, feature selection considers only the information necessary to achieve a robust and accurate classification. The methods employed for removing redundant information are exhaustive, heuristic, or nondeterministic.

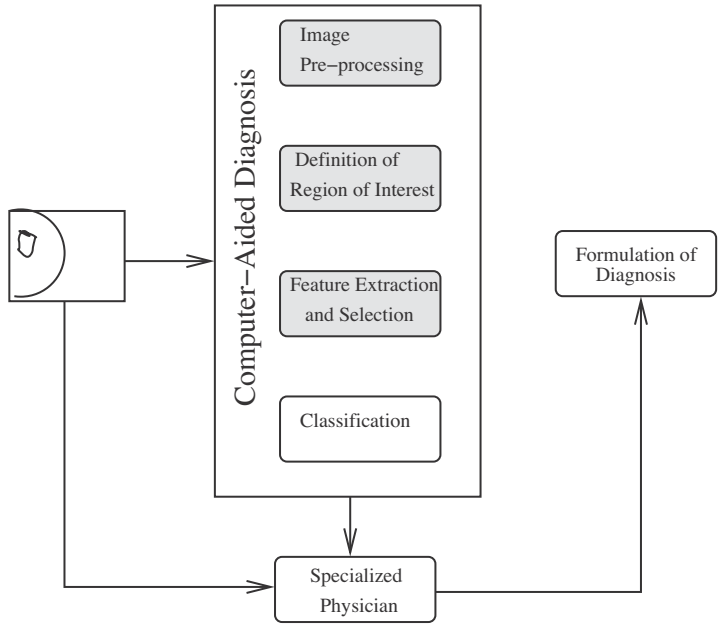


Figure 1.16
Typical architecture of a CAD workstation.

• Classification of the selected ROI: Classification, either supervised or unsupervised, assigns a given set of features describing the ROI to its proper class. These classes can be in medical imaging of tumors, diseases, or physiological signal groups. Several supervised and unsupervised classification algorithms have been applied in the context of breast tumor diagnosis [171, 201, 294].

2 Spectral Transformations

Pattern recognition tasks require the conversion of biosignals in features describing the collected sensor data in a compact form. Ideally, this should pertain only to relevant information. Feature extraction is an important technique in pattern recognition by determining descriptors for reducing dimensionality of pattern representation. A lower-dimensional representation of a signal is a *feature*. It plays a key role in determining the discriminating properties of signal classes. The choice of features, or measurements, has an important influence on (1) accuracy of classification, (2) time needed for classification, (3) number of examples needed for learning, and (4) cost of performing classification.

A carefully selected feature should remain unchanged if there are variations within a signal class, and it should reveal important differences when discriminating between patterns of different signal classes. In other words, patterns are described with as little loss as possible of pertinent information.

There are four known categories in the literature for extracting features [54]:

1. Nontransformed structural characteristics: moments, power, amplitude information, energy, etc.
2. Transformed signal characteristics: frequency and amplitude spectra, subspace transformation methods, etc.
3. Structural descriptions: formal languages and their grammars, parsing techniques, and string matching techniques
4. Graph descriptors: attributed graphs, relational graphs, and semantic networks

Transformed signal characteristics form the most relevant category for biosignal processing and feature extraction. The basic idea employed in transformed signal characteristics is to find such transform-based features with a high information density of the original input and a low redundancy. To understand this aspect better, let us consider a radiographic image. The pixels (input samples) at the various positions have a large degree of correlation. Gray values only introduce redundant information for the subsequent classification. For example, by using the wavelet transform we obtain a feature set based on the wavelet

coefficients which retains only the important image information residing in some few coefficients. These coefficients preserve the high correlation between the pixels.

There are several methods for obtaining transformed signal characteristics. For example, Karhunen-Loeve transform and singular value decomposition are problem-dependent and the result of an optimization process [70, 264]. They are optimal in terms of decorrelation and information concentration properties, but at the same time are too computationally expensive. On the other hand, transforms which use fixed basis vectors (images), such as the Fourier and wavelet transforms, exhibit low computational complexity while being suboptimal in terms of decorrelation and redundancy.

We will review the most important methods for obtaining transformed signal characteristics, such as the continuous and discrete Fourier transform, the discrete cosine and sine transform, and the wavelet transform.

2.1 Frequency Domain Representations

In this section, we will show that Fourier analysis offers the rigorous language needed to define and design modern bioengineering systems. Several continuous and discrete representations derived from the Fourier transform are presented. Thus, it becomes evident that these techniques represent an important concept in the analysis and interpretation of biological signals.

Continuous Fourier Transform

One of the most important tasks in processing of biomedical signals is to decompose a signal intp its frequency components and to determine the corresponding amplitudes. The standard analysis for continuous time signals is performed by the classical Fourier transform. The Fourier transform is defined by the following equation:

$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-j\omega t} dt \tag{2.1}$$

while the inverse transform is given as

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(\omega) e^{j\omega t} d\omega \tag{2.2}$$

The direct transform extracts spectrum information from the signal, and the inverse transform synthesizes the time-domain signal from the spectral information.

EXAMPLE 2.1: We consider the following exponential signal

$$f(t) = e^{-5t} u(t) \tag{2.3}$$

where $u(t)$ is the step function. The Fourier transform is given as

$$F(j\omega) = \int_0^{\infty} e^{-5t} e^{-j\omega t} dt = \int_0^{\infty} e^{-5+j\omega t} dt = \frac{1}{5 + j\omega} \tag{2.4}$$

For real-world problems, we employ the existing properties of the Fourier transform that help to simplify the frequency domain transformations [190]. However, the major drawback of the classical Fourier transform is its inability to deal with nonstationary signals. Since it considers the whole time domain, it misses the local changes of high-frequency components in the signal. In summary, it is assumed that the signal properties (amplitudes, frequency, and phases) will not change with time and will stay the same for the whole length of the window. To overcome these disadvantages, the *short-time Fourier transform* was proposed by Gabor in 1946 [88]. The short-time Fourier transform is defined as

$$F(\omega, \tau) = \int_{-\infty}^{\infty} f(t) g^*(t - \tau) e^{-j\omega t} dt \tag{2.5}$$

where a window $g(t)$ is positioned at some point τ on the time axis. Thus, this new transform works by sweeping a short-time window over the time signal, and thus determines the frequency content in each considered time interval.

The transform modulates the signal with a window function $g(t)$. In this context ω and τ are the modulation and translation parameters. The window $g(t)$ has a fixed time duration and a fixed frequency resolution. Although the frequency and time domains are different, when used to represent functions, they are linked: A precise information about time can be achieved only at the cost of some uncertainty about frequency, and vice versa. This important aspect is captured by the *Heisenberg*

Uncertainty Principle [195] in information processing.

The uncertainty principle states that for each transformation pair $g(t) \longleftrightarrow G(\omega)$, the relationship

$$\sigma_t \sigma_\omega \geq \frac{1}{2} \tag{2.6}$$

holds. σ_T and σ_ω represent the squared variances of $g(t)$ and $G(\omega)$:

$$\begin{aligned} \sigma_T^2 &= \frac{\int t^2 |g(t)|^2 dt}{\int |g(t)|^2 dt} \\ \sigma_\omega^2 &= \frac{\int \omega^2 |G(\omega)|^2 d\omega}{\int |G(\omega)|^2 d\omega} \end{aligned} \tag{2.7}$$

where $g(t)$ is defined as a prototype function. The lower bound is given by the Gaussian function $f(t) = e^{-t^2}$. As τ increases, the prototype function is shifted on the time axis such that the window length remains unchanged. Figure 2.1 graphically visualizes this principle, where each basis function used in the representation of a function is interpreted as a tile in a time-frequency plane. This tile, the so-called Heisenberg cell, describes the energy concentration of the basis function. All these tiles have the same form and area. Thus, each element σ_T and σ_ω of the resolution rectangle of the area $\sigma_T \sigma_\omega$ remains unchanged for each frequency ω and time shift τ.

The short-time Fourier transform can be interpreted as a filtering of signal $f(t)$ by a filter bank in which each filter is centered at a different frequency but has the same bandwidth. It can be seen immediately that a problem arises since both low- and high-frequency components are analyzed by the same window length, and thus an unsatisfactory overall localization of events is achieved. A solution to this problem is given by choosing a window of variable length such that a larger one can analyze long-time, low-frequency components while a shorter one can detect high-frequency, short-time components. This exactly is accomplished by the wavelet transform.

Discrete Fourier Transform

An alternative Fourier representation that pertains to finite-duration sequences is the *discrete Fourier transform* (DFT). This transform represents a sequence rather than a function of a continuous variable, and

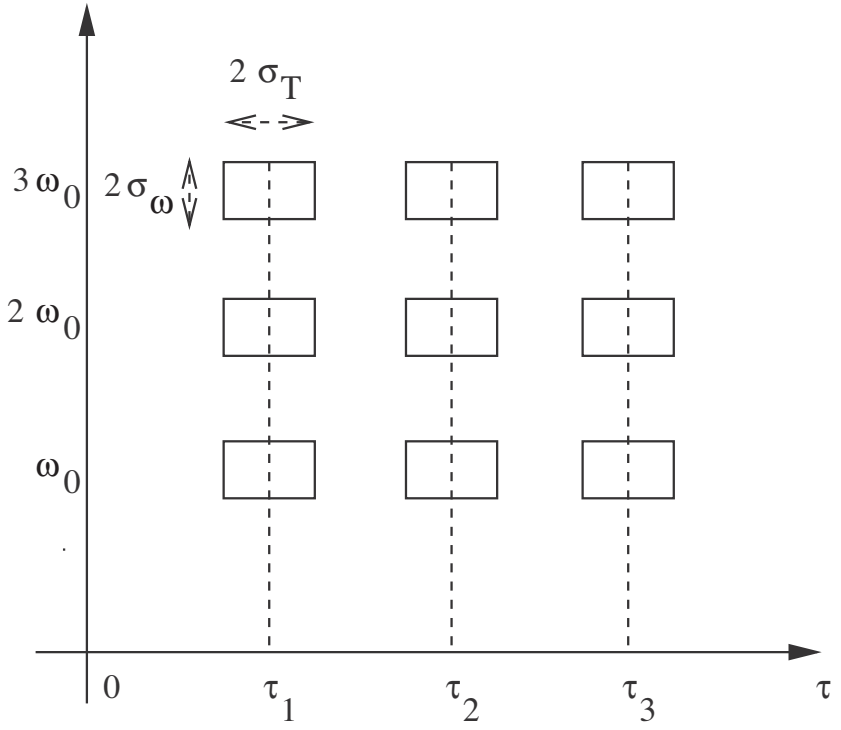


Figure 2.1
Short-time Fourier transform: time-frequency space and resolution cells.

captures samples equally spaced in frequency. The DFT analyzes a signal in terms of its frequency components by finding the signal's magnitude and phase spectra, and exists for both one- and two-dimensional cases.

Let us consider N sampled values $x(0), \ldots, x(N-1)$. Their DFT is given by

$$y(k) = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi}{N}kn}, \quad k = 0, 1, \ldots, N-1 \tag{2.8}$$

and the corresponding inverse transform is

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} y(k) e^{j\frac{2\pi}{N}kn}, \quad n = 0, 1, \ldots, N-1 \tag{2.9}$$

with j $\equiv \sqrt{-1}$. All $x(n)$ and $y(k)$ can be concatenated in the form of two $N \times 1$ vectors. Let us also define

$$W_N \equiv e^{-j\frac{2\pi}{N}} \tag{2.10}$$

such that equations (2.8) and (2.9) can be written in the matrix form

$$\mathbf{y} = \mathbf{W^{-1}x}, \quad \mathbf{x} = \mathbf{Wy} \tag{2.11}$$

with

$$\mathbf{W} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & W_N & W_N^2 & \cdots & W_N^{N-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & W_N^{N-1} & W_N^{2(N-1)} & \cdots & W_N^{(N-1)(N-1)} \end{bmatrix} \tag{2.12}$$

where $\mathbf{W}$ is an unitary and symmetric matrix.

Let us choose as an example the case $N = 2$.

EXAMPLE 2.2: We then obtain for $N = 2$

$$\mathbf{W} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

We see that the columns of $\mathbf{W}$ correspond to the basis vectors

$$\mathbf{w_0} = [1, 1]^T$$

$$\mathbf{w_1} = [1, -1]^T$$

and, based on them, we can reconstruct the original signal:

$$\mathbf{x} = \sum_{i=0}^{1} y(i)\mathbf{w_i}$$

Unfortunately, the DFT has the same drawbacks as the continuous-time Fourier transform when it comes to nonstationary signals: (a) the behavior of a signal within a given window is analyzed; (b) accurate representation is possible only for signals stationary within a window; and (c) good time and frequency resolution cannot be achieved simultaneously, as illustrated by table 2.1.

Table 2.1
Time and frequency resolution by window width.

Narrow window	Good time resolution	Poor frequency resolution
Wide window	Poor time resolution	Good frequency resolution

The two-dimensional DFT for an $N \times N$ image is defined as

$$Y(k,l) = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} X(m,n) W_N^{km} W_N^{ln} \tag{2.13}$$

and its inverse DFT is given by

$$X(m,n) = \frac{1}{N^2} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} Y(k,l) W_N^{-km} W_N^{-ln} \tag{2.14}$$

The corresponding matrix representation yields

$$\mathbf{Y} = \tilde{\mathbf{W}}\mathbf{X}\tilde{\mathbf{W}}, \quad \mathbf{X} = \mathbf{W}\mathbf{Y}\mathbf{W} \tag{2.15}$$

We immediately see that the two-dimensional DFT represents a separable transformation with the basis images $\mathbf{w_i}\mathbf{w_j^T}, i, j = 0, 1, \ldots, N-1$.

Discrete Cosine and Sine Transform

Another very useful transformation is the *discrete cosine transform* (DCT), which plays an important role in image compression and has become an international standard for transform coding systems. Its main advantage is that it can be implemented in a single integrated circuit having all relevant information packed into a few coefficients. In addition, it minimizes blocking artifacts that usually accompany block-based transformations. In the following, we will review the DCT for both the one- and two-dimensional cases.

For N given input samples the DCT is defined as

$$y(k) = \alpha(k) \sum_{n=0}^{N-1} x(n) \cos\left(\frac{\pi(2n+1)k}{2N}\right), \quad k = 0, 1, \ldots, N-1 \tag{2.16}$$

Its inverse transform is given by

$$x(n) = \sum_{k=0}^{N-1} \alpha(k) y(n) \cos\left(\frac{\pi(2n+1)k}{2N}\right), \quad n = 0, 1, \ldots, N-1 \quad (2.17)$$

with

$$\alpha(0) = \sqrt{\frac{1}{N}}, \quad k = 0 \quad \text{and} \quad \alpha(k) = \sqrt{\frac{2}{N}}, \quad 1 \leq k \leq N-1 \quad (2.18)$$

The vector form of the DCT is given by

$$\mathbf{y} = \mathbf{C^T x} \quad (2.19)$$

while for the elements of the matrix $\mathbf{C}$ we have

$$C(n,k) = \sqrt{\frac{1}{N}}, \quad k = 0, \quad 0 \leq n \leq N-1$$

and

$$C(n,k) = \sqrt{\frac{2}{N}} \cos\left(\frac{\pi(2n+1)k}{2N}\right),$$
$$1 \leq k \leq N-1, \quad 0 \leq n \leq N-1.$$

$\mathbf{C}$ represents an orthogonal matrix with real numbers as elements:

$$\mathbf{C^{-1}} = \mathbf{C^T}.$$

In the two-dimensional case the DCT becomes

$$\mathbf{Y} = \mathbf{C^T X C}, \quad \mathbf{X} = \mathbf{C Y C^T}. \quad (2.20)$$

Unlike the DFT, the DCT is real-valued. Also, its basis sequences are cosines. Compared with the DFT, which requires periodicity, this transform involves indirect assumptions about both periodicity and even symmetry.

Another orthogonal transform is the *discrete sine transform* (DST), defined as

$$S(k,n) = \sqrt{\frac{2}{N+1}} \sin\left(\frac{\pi(n+1)(k+1)}{N+1}\right), \quad k,n = 0,1,\ldots,N-1 \tag{2.21}$$

Its basis sequences in the orthonormal transformation are sine functions. Both DCT and DST have excellent information concentration properties since they concentrate most of the energy in a few coefficients.

Other important transforms are the Haar, wavelet, Hadamard, and Walsh transforms [48, 264]. Because of the powerful properties of the wavelet transform and its extensive application opportunities in biomedical engineering, the next section is dedicated solely to the wavelet transform.

2.2 The Wavelet Transform

Modern transform techniques such as the wavelet transform are gaining an increasing importance in biomedical signal and image processing. They provide enhanced processing capabilities compared to the traditional ones in terms of denoising, compression, enhancement, and edge and feature extraction. These techniques fall under the categories of multiresolution analysis, time-frequency analysis, or pyramid algorithms. The wavelet transform is based on wavelets, which are small waves of varying frequency and limited duration, and thus represents a deviation from the traditional Fourier transform concept that has sinusoids as basis functions. In addition to the traditional Fourier transform, they provide not only frequency but also temporal information on the signal.

In this section, we present the theory and the different types of wavelet transforms. A wavelet represents a basis function in continuous time and can serve as an important component in a function representation: any function $f(t)$ can be represented by a linear combination of basis functions, such as wavelets. The most important aspect of the wavelet basis is that all wavelet functions are constructed from a single mother wavelet. This wavelet is a small wave or a pulse.

Wavelet transforms are an alternative to the short-time Fourier transform. Their most important feature is that they analyze different frequency components of a signal with different resolutions. In other words, they address exactly the concern raised in connection with the

short-time Fourier transform. Implementing different resolutions at different frequencies requires the notion of functions at different scales. Like scales on a map, small scales show fine details while large scales show only coarse features. A scaled version of a function $\psi(t)$ is the function $\psi(t/a)$, for any scale a. When $a > 1$, a function of lower frequency is obtained that is able to describe slowly changing signals. When $a < 1$, a function of higher frequency is obtained that can detect fast signal changes. It is important to note that the scale is inversely proportional to the frequency.

Wavelet functions are localized in frequency in the same way sinusoids are, but they differ from sinusoids by being localized in time as well. There are several wavelet families, each having a characteristic shape, and the basic scale for each family covers a known, fixed interval of time. The time spans of the other wavelets in the family widen for larger scales and narrow for smaller scales. Thus, wavelet functions can offer either good time resolution or good frequency resolution: good time resolution is associated with narrow, small-scale windows, while good frequency resolution is associated with wide, large-scale windows.

To determine what frequencies are present in a signal and when they occur, the wavelet functions at each scale must be translated through the signal, to enable comparison with the signal in different time intervals. A scaled and translated version of the wavelet function $\psi(t)$ is the function $\psi(\frac{t-b}{a})$, for any scale a and translation b. A wavelet function similar to the signal in frequency produces a large wavelet transform. If the wavelet function is dissimilar to the signal, a small transform will arise. A signal can be coded using these wavelets if it can be decomposed into scaled and translated copies of the basic wavelet function. The widest wavelet responds to the slowest signal variations, and thus describes the coarsest features in the signal. Smaller scale wavelets respond best to high frequencies in the signal and detect rapid signal changes, thus providing detailed information about this signal. In summary, smaller scales correspond to higher frequencies, and larger scales to lower frequencies. A signal is coded through the wavelet transform by comparing the signal against many scalings and translations of a wavelet function.

The *wavelet transform* (WT) is produced by a translation and dilation of a so-called prototype function ψ. Figure 2.2 illustrates a typical wavelet and its scalings. The bandpass characteristics of ψ and the time-

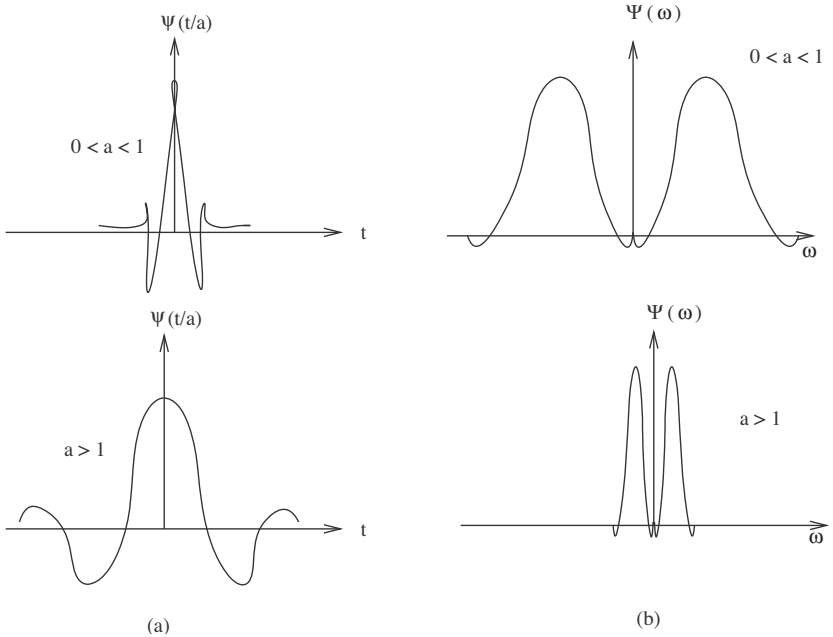


Figure 2.2
Wavelet in time and frequency domains: (a) scale parameter $0 < a < 1$, (b) scale parameter $a > 1$.

frequency resolution of the WT can easily be detected.

The foundation of the WT is based on the scaling property of the Fourier transform. If

$$\psi(t) \longleftrightarrow \Psi(\omega)$$

represents a Fourier transform pair, then we have

$$\frac{1}{\sqrt{a}}\Psi\left(\frac{t}{a}\right) \longleftrightarrow \sqrt{a}\Psi(a\omega) \tag{2.22}$$

with $a > 0$ being a continuous variable. A contraction in the time domain produces an expansion in the frequency domain, and vice versa. Figure 2.3 illustrates the corresponding resolution cells in the time-frequency domain. The figure makes visual the underlying property of wavelets: they are localized in both time and frequency. The functions $e^{j\omega t}$ are

perfectly localized at ω, they extend over all time; wavelets, on the other hand, that are not at a single frequency are limited to finite time. As we rescale, the frequency increases by a certain quantity, and at the same time the time interval decreases by the same quantity. Thus the uncertainty principle holds.

A wavelet can be defined by the scale and shift parameters a and b,

$$\psi_{ab}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right) \tag{2.23}$$

while the WT is given by the inner product

$$W(a,b) = \int_{-\infty}^{\infty} \psi_{ab}(t) f^*(t) dt =< \psi_{ab}, f > \tag{2.24}$$

with $a \in R^+, b \in R$.

The WT defines an $L^2(R) \rightarrow L^2(R^2)$ mapping which has a better time-frequency localization than the short-time Fourier transform.

In the following, we will describe the *continuous wavelet transform* (CWT) and show an admissibility condition which is necessary to ensure the inversion of the WT. Also, we will define the *discrete wavelet transform* (DWT), which is generated by sampling the wavelet parameters (a, b) on a grid or lattice. The quality of the reconstructed signals based on the transform values depends on the coarseness of the sampling grid. A finer sampling grid leads to more accurate signal reconstruction at the cost of redundancy; a coarse sampling grid is associated with loss of information. To address these important issues, the concept of frames is now presented.

The Continuous Wavelet Transform

The CWT transforms a continuous function into a highly redundant function of two continuous variables, translation and scale. The resulting transformation is important for time-frequency analysis and is easy to interpret.

The CWT is defined as the mapping of the function $f(t)$ on the time-scale space by

$$W_f(a,b) = \int_{-\infty}^{\infty} \psi_{ab}(t) f(t) dt =< \psi_{ab}(t), f(t) > \tag{2.25}$$

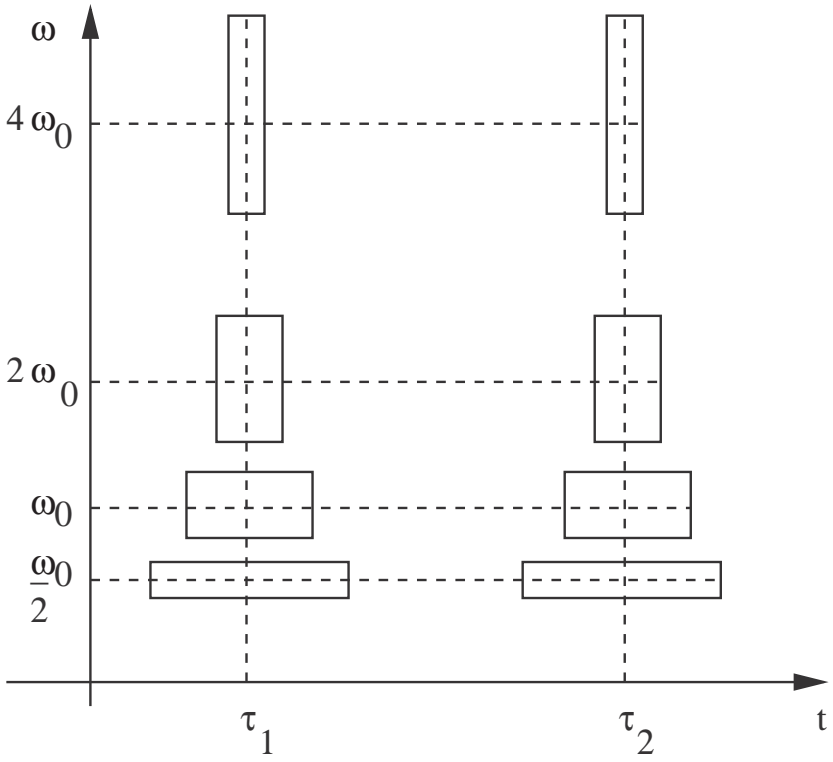


Figure 2.3
Wavelet transform: time-frequency domain and resolution cells.

The CWT is invertible if and only if the resolution of identity holds:

$$f(t) = \underbrace{\frac{1}{C_\psi}\int_{-\infty}^{\infty}\int_{0}^{\infty}\frac{dadb}{a^2}}_{\text{Summation}} \underbrace{W_f(a,b)}_{\text{Waveletcoefficients}} \underbrace{\psi_{ab}(t)}_{\text{Wavelet}} \tag{2.26}$$

where

$$C_\psi = \int_o^\infty \frac{|\Psi(\omega)|^2}{\omega} d\omega \tag{2.27}$$

assuming that a real-valued $\psi(t)$ fulfills the admissibility condition. If

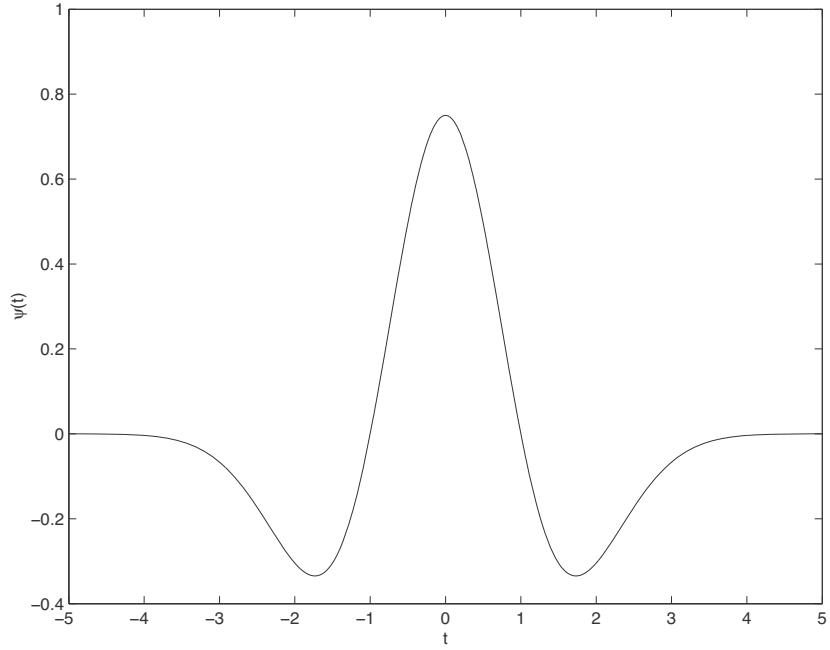


Figure 2.4
Mexican-hat wavelet.

$C_\psi < \infty$, then the wavelet is called admissible. Then for the gain we get

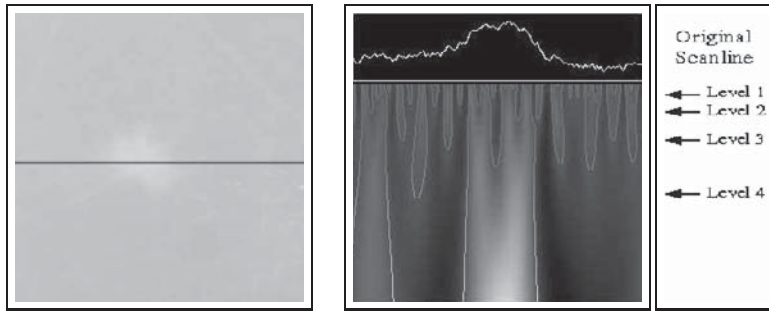
$$\Psi(0) = \int_{-\infty}^{\infty} \psi(t)dt = 0 \tag{2.28}$$

We immediately see that $\psi(t)$ corresponds to the impulse response of a bandpass filter and has a decay rate of $|t|^{1-\varepsilon}$. It is important to note that based on the admissibility condition, it can be shown that the CWT is complete if $W_f(a, b)$ is known for all a, b.

The *Mexican-hat wavelet*

$$\psi(t) = (\frac{2}{\sqrt{3}}\pi^{-\frac{1}{4}})(1 - t^2)e^{-\frac{t^2}{2}} \tag{2.29}$$

is visualized in figure 2.4. It has a distinctive symmetric shape, and it has an average value of zero and dies out rapidly as $|t| \to \infty$. There is no scaling function associated with the Mexican hat wavelet.



(a) (b)

Figure 2.5
Continuous wavelet transform: (a) scan line and (b) multi-scale coefficients. (Images courtesy of Dr. A. Laine, Columbia University.)

Figure 2.5 illustrates the multiscale coefficients describing a spiculated mass. Figure 2.5a shows the scan line through a mammographic image with a mass (8 mm), and figure 2.5b visualizes the multi scale coefficients at various levels.

The short-time Fourier transform finds a decomposition of a signal into a set of equal-bandwidth functions across the frequency spectrum. The WT provides a decomposition of a signal based on a set of bandpass functions that are placed over the entire spectrum. The WT can be seen as a signal decomposition based on a set of constant-Q bandpasses. In other words, we have an octave decomposition, logarithmic decomposition, or constant-Q decomposition on the frequency scale. The bandwidth of each of the filters in the bank is the same in a logarithmic scale or, equivalently, the ratio of the filters bandwidth to the respective central frequency is constant.

2.3 The Discrete Wavelet Transformation

The CWT has two major drawbacks: redundancy and lack of practical relevance. The first is based on the nature of the WT; the latter is because the transformation parameters are continuous. A solution to these problems can be achieved by sampling both parameters (a, b) such that a set of wavelet functions in the form of discrete parameters is

obtained. We also have to look into the following problems:

1. Is the set of discrete wavelets complete in $L^2(R)$?
2. If complete, is the set at the same time also redundant?
3. If complete, then how coarse must the sampling grid be, such that the set is minimal or nonredundant?

A response to these questions will be given in this section, and we also will show that the most compact set is the orthonormal wavelet set.

The sampling grid is defined as follows [4]:

$$a = a_0^m b = nb_0 a_0^m \tag{2.30}$$

where

$$\psi_{mn}(t) = a^{-m/2}\psi(a_0^{-m}t - nb_0) \tag{2.31}$$

with $m, n \in Z$. If we consider this set to be complete in $L^2(R)$ for a given choice of $\psi(t), a, b$, then $\{\psi_{mn}\}$ is an *affine wavelet*. $f(t) \in L^2(R)$ represents a wavelet synthesis. It recombines the components of a signal to reproduce the original signal $f(t)$. If we have a wavelet basis, we can determine a wavelet series expansion. Thus, any square-integrable (finite energy) function $f(t)$ can be expanded in wavelets:

$$f(t) = \sum_m \sum_n d_{m,n}\psi_{mn}(t) \tag{2.32}$$

The wavelet coefficient $d_{m,n}$ can be expressed as the inner product

$$d_{m,n} =< f(t), \psi_{mn}(t) >= \frac{1}{a_0^{m/2}} \int f(t)\psi(a_0^{-m}t - nb_0)dt \tag{2.33}$$

These complete sets are called frames. An analysis frame is a set of vectors ψ_{mn} such that

$$A||f||^2 \leq \sum_m \sum_n | < f, \psi_{mn} > |^2 \leq B||f||^2 \tag{2.34}$$

with

$$||f||^2 \triangleq \int |f(t)|^2 dt \tag{2.35}$$

$A, B > 0$ are the frame bounds. A tight, exact frame that has $A = B = 1$ represents an orthonormal basis for $L^2(R)$. A notable characteristic of orthonormal wavelets $\{\psi_{mn}(t)\}$ is

$$\int \psi_{mn}(t)\psi_{m'n'}(t)dt = \begin{cases} 1, & m = m', n = n' \\ 0, & \text{else} \end{cases} \tag{2.36}$$

In addition they areorthonormal in both indices. This means that for the same scale m they are orthonormal both in time and across the scales.

For the scaling functions the orthonormal condition holds only for a given scale

$$\int \varphi_{mn}(t)\varphi_{ml}(t)dt = \delta_{n-l} \tag{2.37}$$

The scaling function can be visualized as a low-pass filter. While scaling functions alone can code a signal to any desired degree of accuracy, efficiency can be gained by using the wavelet functions. Any signal $f \in L^2(R)$ at the scale m can be approximated by its projections on the scale space.

The similarity between ordinary convolution and the analysis equations suggests that the scaling function coefficients and the wavelet function coefficients may be viewed as impulse responses of filters, as shown in Figure 2.6. The convolution of $f(t)$ with $\psi_m(t)$ is given by

$$y_m(t) = \int f(\tau)\psi_m(\tau - t)d\tau \tag{2.38}$$

where

$$\psi_m(t) = 2^{-m/2}\psi(2^{-m}t) \tag{2.39}$$

Sampling $y_m(t)$ at $n2^m$ yields

$$y_m(n2^m) = 2^{-m/2} \int f(\tau)\psi(2^{-m}\tau - n)d\tau = d_{m,n} \tag{2.40}$$

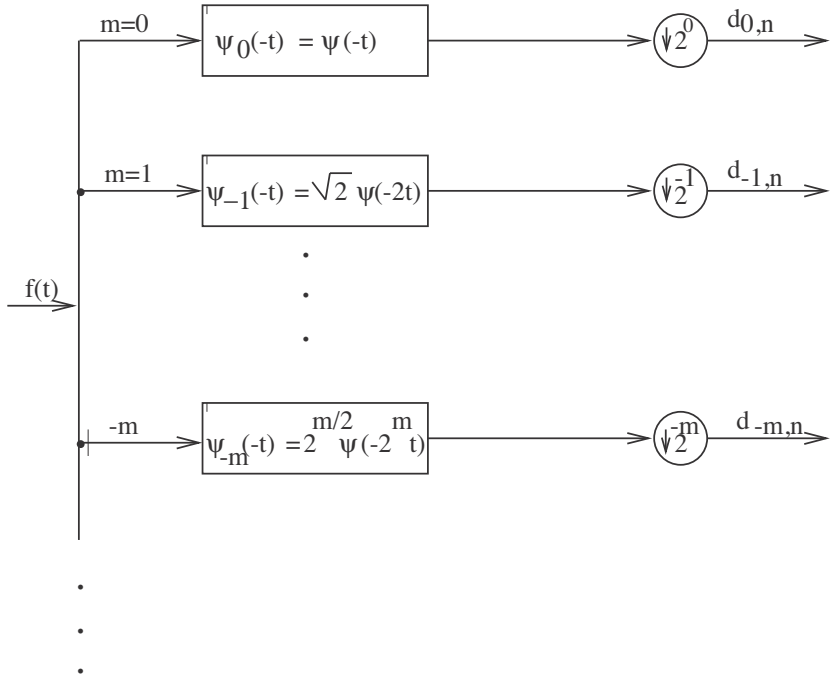


Figure 2.6
Filter bank representation of DWT.

Whereas in the filter bank representation of the short-time Fourier transform all subsamplers are identical, the subsamplers of the filter bank corresponding to the wavelet transform are dependent on position or scale.

The DWT dyadic sampling grid in figure 2.7 visualizes this aspect. Every single point represents a wavelet basis function $\psi_{mn}(t)$ at the scale 2^{-m} and shifted by $n2^{-m}$.

2.4 Multiscale Signal Decomposition

The goal of this section is to highlight an important aspect of the wavelet transform that accounts for its success as a method in pattern recognition: the decomposition of the whole function space into subspaces. This implies that there is a piece of the function $f(t)$ in each subspace. Those

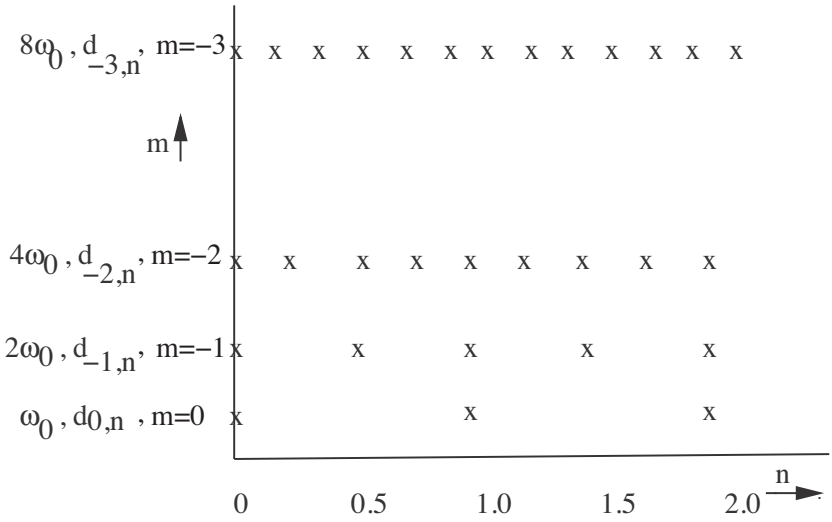


Figure 2.7
Dyadic sampling grid for the DWT.

pieces (or projections) give finer and finer details of $f(t)$. For audio signals, these scales are essentially octaves. They represent higher and higher frequencies. For images and all other signals, the simultaneous appearance of multiple scales is known as *multiresolution*.

Mallat and Meyer's method [165] for signal decomposition based on orthonormal wavelets with compact carrier will be reviewed here. We will establish a link between these wavelet families and the hierarchic filter banks. In the last part of this section, we will show that the FIR PR–QMF hold the regularization property, and produce orthonormal wavelet bases.

Multiscale-Analysis Spaces

Multiscale signal analysis provides the key to the link between wavelets and pyramidal dyadic trees. A wavelet family is used to decompose a signal into scaled and translated copies of a basic function. As stated before, the wavelet family consists of scaling and wavelet functions. *Scaling functions* $\varphi(t)$ alone are adequate to code a signal completely, but a decomposition based on both scaling and wavelet functions is most efficient.

In mathematical terminology, a function $f(t)$ in the whole space has a piece in each subspace. Those pieces contain more and more of the full information in $f(t)$. These successive approximations converge to a limit which represents the function $f \in L^2$. At the same time they describe different resolution levels, as is known from the pyramidal representation.

A multiscale analysis is based on a sequence of subspaces $\{V_m | m \in Z\}$ in $L^2(R)$ satisfying the following requirements:

• Inclusion: Each subspace V_j is contained in the next subspace. A function $f \in L^2(R)$ in one subspace is in all the higher (finer) subspaces:

$$\cdots V_2 \subset V_1 \subset V_0 \subset V_{-1} \subset V_{-2} \cdots \qquad (2.41)$$
$$\leftarrow coarser \qquad finer \rightarrow$$

• Completeness: A function in the whole space has a part in each subspace.

$$\bigcap_{m \in Z} V_m = 0 \quad \bigcup_{m \in Z} V_m = L^2(R) \qquad (2.42)$$

• Scale invariance:

$$f(x) \in V_m \Longleftrightarrow f(2x) \in V_{m-1} \quad \text{for any function} \quad f \in L^2(R) \qquad (2.43)$$

• Basis-frame property: This requirement for multiresolution concerns a basis for each space V_j. There is a scaling function $\varphi(t) \in V_0$, such that $\forall m \in Z$, the set

$$\{\varphi_{mn}(t) = 2^{-m/2}\varphi(2^{-m}t - n)\} \qquad (2.44)$$

forms an orthonormal basis for V_m:

$$\int \varphi_{mn}(t)\varphi_{mn'}(t)dt = \delta_{n-n'} \qquad (2.45)$$

In the following, we will mathematically review the multiresolution concept based on scaling and wavelet functions, and thus define the approximation and detail operators.

Let $\varphi_{mn}(t)$ with $m \in Z$ be defined as

$$\{\varphi_{mn}(t) = 2^{-m/2}\varphi(2^{-m}t - n)\} \tag{2.46}$$

Then the approximation operator P_m on functions $f(t) \in L^2(R)$ is defined by

$$P_m f(t) = \sum_n < f, \varphi_{mn} > \varphi_{mn}(t) \tag{2.47}$$

and the detail operator Q_m on functions $f(t) \in L^2(R)$ is defined by

$$Q_m f(t) = P_{m-1} f(t) - P_m f(t) \tag{2.48}$$

It can easily be shown that $\forall m \in Z, \{\varphi_{mn}(t)\}$ is an orthonormal basis for V_m [278], and that for all functions $f(t) \in L^2(R)$,

$$\lim_{m \to -\infty} ||P_m f(t) - f(t)||_2 = 0 \tag{2.49}$$

and

$$\lim_{m \to \infty} ||P_m f(t)||_2 = 0 \tag{2.50}$$

An important feature of every scaling function $\varphi(t)$ is that it can be built from translations of double-frequency copies of itself, $\varphi(2t)$, according to

$$\varphi(t) = 2 \sum_n h_0(n)\varphi(2t - n) \tag{2.51}$$

This equation is called a multiresolution-analysis equation. Since $\varphi(t) = \varphi_{00}(t)$, both m and n can be set to 0 to obtain the above simpler expression. The equation expresses the fact that each scaling function in a wavelet family can be expressed as a weighted sum of scaling functions at the next finer scale. The set of coefficients $\{h_0(n)\}$ is called the scaling function coefficients and behaves as a low-pass filter.

Wavelet functions can also be built from translations of $\varphi(2t)$:

$$\psi(t) = 2 \sum_n h_1(n)\varphi(2t - n) \tag{2.52}$$

This equation is called the *fundamental wavelet equation*. The set of coefficients $\{h_1(n)\}$ is called the wavelet function coefficients and behaves as a high-pass filter. This equation expresses the fact that each wavelet function in a wavelet family can be written as a weighted sum of scaling functions at the next finer scale.

The following theorem provides an algorithm for constructing a wavelet orthonormal basis, given a multiscale analysis.

THEOREM 2.1:

Let $\{V_m\}$ be a multiscale analysis with scaling function $\varphi(t)$ and scaling filter $h_0(n)$.

Define the wavelet filter $h_1(n)$ by

$$h_1(n) = (-1)^{n+1} h_0(N-1-n) \tag{2.53}$$

and the wavelet $\psi(t)$ by equation (2.52).

Then

$$\{\psi_{mn}(t)\} \tag{2.54}$$

is a wavelet orthonormal basis on R.

Alternatively, given any $L \in Z$,

$$\{\varphi_{Ln}(t)\}_{n \in Z} \bigcup \{\psi_{mn}(t)\}_{m,n \in Z} \tag{2.55}$$

is an orthonormal basis on R.

The proof can be found in [278]. Some very important facts representing the key statements of multiresolution follow:

(a) $\{\psi_{mn}(t)\}$ is an orthonormal basis for W_m.

(b) If $m \neq m'$, then $W_m \perp W_{m'}$.

(c) $\forall m \in Z$, $V_m \perp W_m$ where W_m is the orthogonal complement of V_m in V_{m-1}.

(d) In $\forall m \in Z$, $V_{m-1} = V_m \oplus W_m$, $\oplus$ stands for orthogonal sum. This means that the two subspaces are orthogonal and that every function in V_{m-1} is a sum of functions in V_m and W_m. Thus every function $f(t) \in V_{m-1}$ is composed of two subfunctions, $f_1(t) \in V_m$ and $f_2(t) \in W_m$, such that

Table 2.2
Properties of orthonormal wavelets.

$\varphi(t)$	=	$\sum h_0(n)\varphi(2t-n)$
$\psi(t)$	=	$\sum h_1(n)\psi(2t-n)$
$h_1(n)$	=	$(-1)^{n+1}h_0(N-1-n)$
$<\psi_{mn}(t),\psi_{kl}(t)>$	=	$\delta_{m-k}\delta_{n-l}$
$<\varphi_{mn}(t),\varphi_{mn'}(t)>$	=	$\delta_{n-n'}$
$<\varphi_{mn}(t),\psi_{kl}(t)>$	=	0

$f(t) = f_1(t) + f_2(t)$ and $< f_1(t), f_2(t) >= 0$.
The most important part of multiresolution is that the spaces W_m represent the differences between the spaces V_m, while the spaces V_m are the sums of W_m.

(e) Every function $f(t) \in L^2(R)$ can be expressed as

$$f(t) = \sum_m f_m(t), \tag{2.56}$$

where $f_m(t) \in W_m$ and $< f_m(t), f_{m'} >= 0$. This can be usually written as

$$\cdots \oplus W_j \oplus W_{j-1} \cdots \oplus W_0 \cdots \oplus W_{-j+1} \oplus W_{-j+2} \cdots = L^2(R). \tag{2.57}$$

Although scaling functions alone can code a signal to any desired degree of accuracy, efficiency can be gained by using the wavelet functions. This leads to a new understanding of the concept of multiresolution. Multiresolution can be described based on wavelet W_j and scaling subspaces V_j. This means that the subspace formed by the wavelet functions covers the difference between the subspaces covered by the scaling functions at two adjacent scales.

The mathematical properties of orthonormal wavelets with compact carriers are summarized in table 2.2 [4].

A Very Simple Wavelet: The Haar Wavelet

The *Haar wavelet* is one of the simplest and oldest known orthonormal wavelets. However, it has didactic value because it helps to visualize the multiresolution concept.

Let V_m be the space of piecewise constant functions

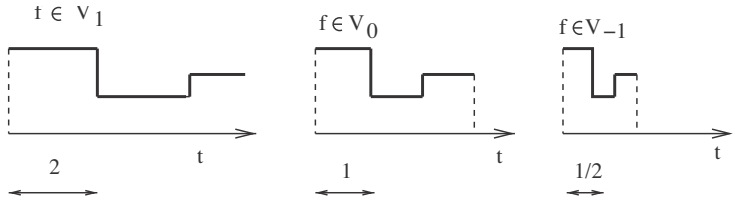


Figure 2.8
Piecewise constant functions in V_1, V_0 and V_{-1}.

$$V_m = \{f(t) \in L^2(R); \quad \text{f is constant in} \quad [2^m n, 2^m(n+1)] \quad \forall n \in Z\}. \tag{2.58}$$

Figure 2.8 illustrates such a function.

We can easily see that $\cdots V_1 \subset V_0 \subset V_{-1} \cdots$ and $f(t) \in V_0 \longleftrightarrow f(2t) \in V_{-1}$, and that the inclusion property is fulfilled. The function $f(2t)$ has the same shape as $f(t)$ but is compressed to half the width.

The scaling function of the Haar wavelet $\varphi(t)$ is given by

$$\varphi(t) = \begin{cases} 1, & 0 \le t \le 1 \\ 0, & \text{else} \end{cases} \tag{2.59}$$

and defines an orthonormal basis for V_0. Since for $n \neq m$, $\varphi(t-n)$ and $\varphi(t-m)$ do not overlap, we obtain

$$\int \varphi(t-n)\varphi(t-m)dt = \delta_{n-m} \tag{2.60}$$

The Fourier transform of the scaling function yields

$$\Phi(\omega) = e^{-j\frac{\omega}{2}} \frac{\sin \omega/2}{\omega/2}. \tag{2.61}$$

Figure 2.9 shows that $\varphi(t)$ can be written as the linear combination of even and odd translations of $\varphi(2t)$:

$$\varphi(t) = \varphi(2t) + \varphi(2t-1) \tag{2.62}$$

Since $V_{-1} = V_0 \oplus W_0$ and $Q_0 f = (P_{-1}f - P_0 f) \in W_0$ represent the details from scale 0 to -1, it is easy to see that $\psi(t-n)$ spans W_0. The

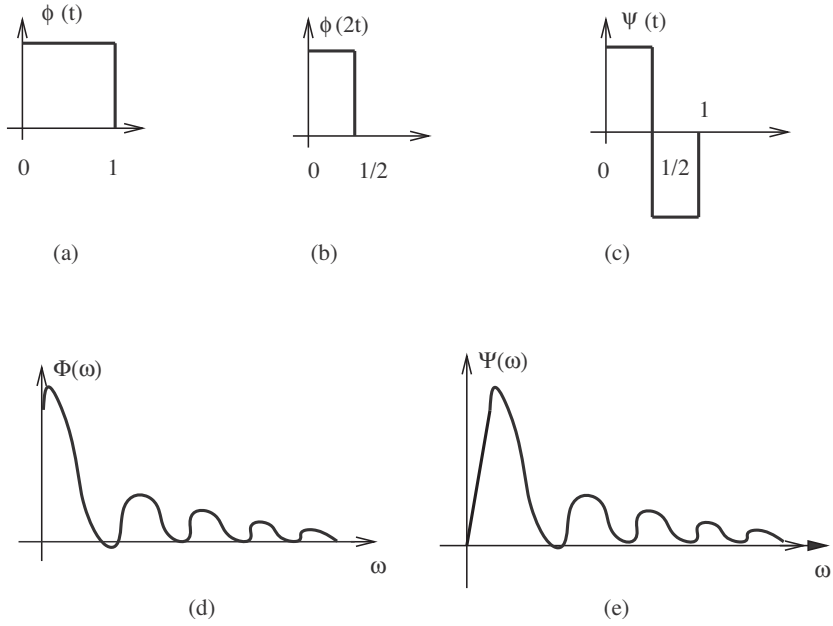


Figure 2.9
(a) and (b) Haar basis functions; (c) Haar wavelet; (d) Fourier transform of the scaling function; (e) Haar wavelet function.

Haar mother wavelet function is given by

$$\psi(t) = \varphi(2t) - \varphi(2t-1) = \begin{cases} 1, & 0 \le t < 1/2 \\ -1, & 1/2 \le t < 1 \\ 0, & \text{else} \end{cases} \tag{2.63}$$

The Haar wavelet function is an up-down square wave, and can be described by a half-box minus a shifted half-box. We also can see that the wavelet function can be computed directly from the scaling functions. In the Fourier domain it describes a bandpass, as can be easily seen from figure 2.9e. This is given by

$$\Psi(\omega) = je^{-j\frac{\omega}{2}} \frac{\sin^2 \omega/4}{\omega/4}. \tag{2.64}$$

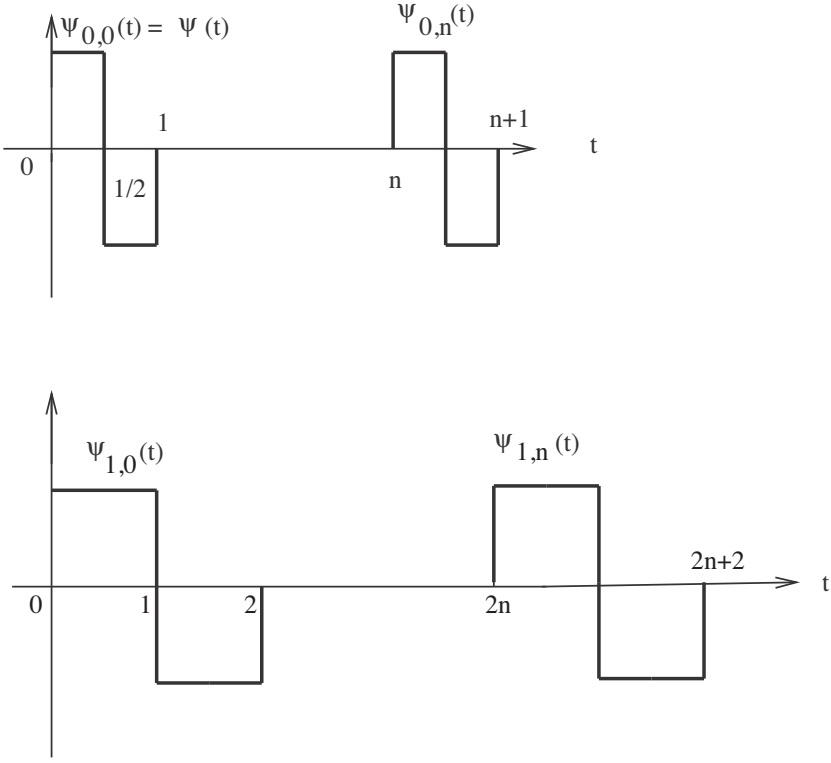


Figure 2.10
Typical Haar wavelet for the scales 0 and 1.

We can easily show that

$$\varphi_{m+1,n} = \frac{1}{\sqrt{2}}[\varphi_{m,2n} + \varphi_{m,2n+1}]$$

and

$$\psi_{m+1,n} = \frac{1}{\sqrt{2}}[\varphi_{m,2n} - \varphi_{m,2n+1}]. \tag{2.65}$$

Figure 2.10 illustrates a typical Haar wavelet for the scales 0 and 1. Figure 2.11 shows the approximations $P_0 f$, $P_{-1} f$ and the detail $Q_0 f$ for a function f. As stated in the context of multiresolution, the detail $Q_0 f$ is added to the coarser approximation $P_0 f$ in order to obtain the finer

approximation $P_{-1}f$.

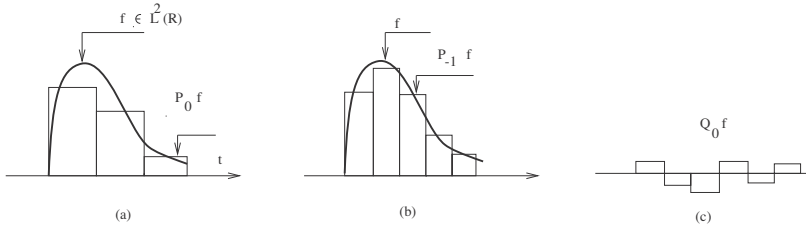


Figure 2.11
Approximation of (a) P_0f, (b) P_{-1}f, and (c) the detail signal Q_0f, with P_0f+Q_0f=P_{-1}f.

The scaling function coefficients for the Haar wavelet at scale m are given by

$$c_{m,n} =< f, \varphi_{mn} >= 2^{-m/2} \int_{2^m n}^{2^m (n+1)} f(t)dt \tag{2.66}$$

This yields an approximation of f at scale m:

$$P_m f = \sum_n c_{m,n} \varphi_{mn}(t) = \sum_n c_{m,n} 2^{-m/2} \varphi(2^{-m}t - n) \tag{2.67}$$

In spite of their simplicity, the Haar wavelets exhibit some undesirable properties which pose a difficulty in many practical applications. Other wavelet families, such as Daubechies wavelets and Coiflet basis [4, 278] are more attractive in practice. Daubechies wavelets are quite often used in image compression. The scaling function coefficients $h_0(n)$ and the wavelet function coefficients $h_1(n)$ for the Daubechies-4 family are nearly impossible to determine. They were obtained based on iterative methods [38].

Multiscale Signal Decomposition and Reconstruction

In this section we will illustrate multiscale pyramid decomposition. Based on a wavelet family, a signal can be decomposed into scaled and translated copies of a basic function. As discussed in the preceeding sections, a wavelet family consists of scaling functions, which are scalings and translations of a father wavelet, and wavelet functions, which are

scalings and translations of a mother wavelet. We will show an efficient signal coding that uses scaling and wavelet functions at two successive scales. In other words, we give a recursive algorithm which supports the computation of wavelet coefficients of a function $f(t) \in L^2(R)$.

Assume we have a signal or a sequence of data $\{c_0(n)|n \in Z\}$, and $c_0(n)$ is the nth scaling coefficient for a given function $f(t)$:

$$c_{0,n} =< f, \varphi_{0n} >$$

for each $n \in Z$. This assumption makes the recursive algorithm work.

The decomposition and reconstruction algorithm is given by theorem 2.2 [278].

THEOREM 2.2:

Let $\{V_k\}$ be a multiscale analysis with associated scaling function $\varphi(t)$ and scaling filter $h_0(n)$. The wavelet filter $h_1(n)$ is defined by equation (2.52), and the wavelet function is defined by equation (2.53).

Given a function $f(t) \in L^2(R)$, define for $n \in Z$

$$c_{0,n} =< f, \varphi_{0n} > \tag{2.68}$$

and for every $m \in N$ and $n \in Z$,

$$c_{m,n} =< f, \varphi_{mn} > \quad \text{and} \quad d_{m,n} =< f, \psi_{mn} > \tag{2.69}$$

Then the decomposition algorithm is given by

$$c_{m+1,n} = \sqrt{2}\sum_k c_{m,k}h_0(k-2n) \quad d_{m+1,n} = \sqrt{2}\sum_k d_{m,k}h_1(k-2n) \tag{2.70}$$

and the reconstruction algorithm is given by

$$c_{m,n} = \sqrt{2}\sum_k c_{m+1,n}h_0(n-2k) + \sqrt{2}\sum_k d_{m+1,n}h_1(n-2k) \tag{2.71}$$

From equation (2.70) we obtain for $m = 1$ at resolution $1/2$ the wavelet $d_{1,n}$ and the scaling coefficients $c_{1,n}$:

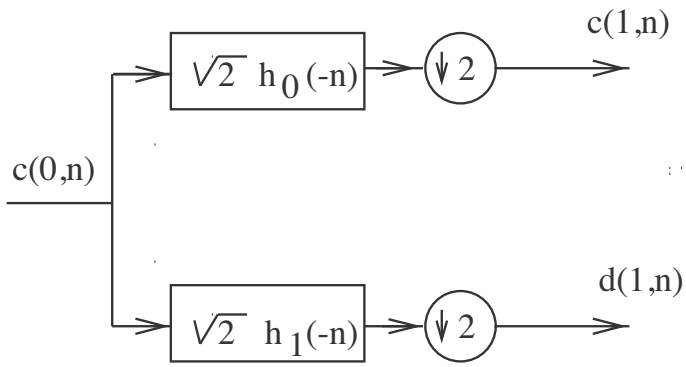


Figure 2.12
First level of the multiscale signal decomposition.

$$c_{1,n} = \sqrt{2} \sum h_0(k - 2n) c_{0,k} \tag{2.72}$$

and

$$d_{1,n} = \sqrt{2} \sum h_1(k - 2n) c_{0,k} \tag{2.73}$$

These last two analysis equations relate the DWT coefficients at a finer scale to the DWT coefficients at a coarser scale. The analysis operations are similar to ordinary convolution. The similarity between ordinary convolution and the analysis equations suggests that the scaling function coefficients and wavelet function coefficients may be viewed as impulse responses of filters. In fact, the set $\{h_0(-n), h_1(-n)\}$ can be viewed as a paraunitary FIR filter pair. Figure 2.12 illustrates this.

The discrete signal $d_{1,n}$ is the WT coefficient the resolution 1/2 and describes the detail signal or difference between the original signal $c_{0,n}$ and its smooth undersampled approximation $c_{1,n}$.

For $m = 2$, we obtain at the resolution 1/4 the coefficients of the

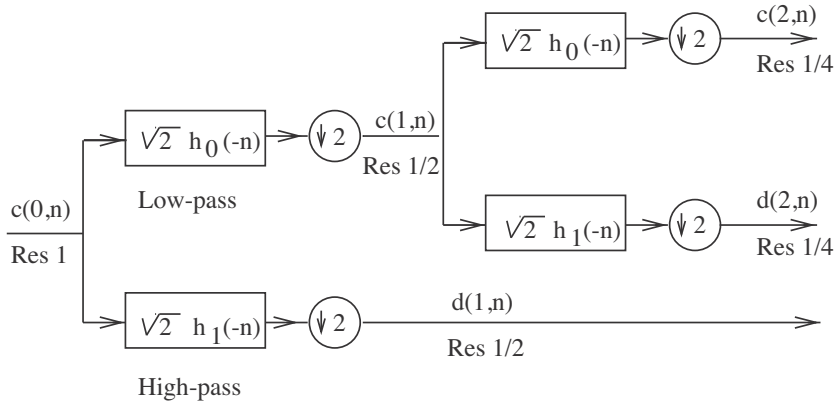


Figure 2.13
Multiscale pyramid decomposition.

smoothed signal (approximation) and the detail signal (approximation error) as

$$c_{2,n} = \sqrt{2} \sum c_{1,k} h_0(k - 2n) \tag{2.74}$$

$$d_{2,n} = \sqrt{2} \sum c_{1,k} h_1(k - 2n) \tag{2.75}$$

These relationships are illustrated in the two-level multiscale pyramid in figure 2.13.

Wavelet synthesis is the process of recombining the components of a signal to reconstruct the original signal. The inverse discrete wavelet transformation, or IDWT, performs this operation. To obtain $c_{0,n}$, the terms $c_{1,n}$ and $d_{1,n}$ are upsampled and convoluted with the filters $h_0(n)$ and $h_1(n)$, as shown in figure 2.14.

The results of the multiscale decomposition and reconstruction of a dyadic subband tree are shown in figure 2.15 and describe the analysis and synthesis part of a two-band PR-QMF bank.

It is important to note that the recursive algorithms for decomposition and reconstruction can easily be extended for a two-dimensional signal (image) [278] and play an important role in image compression.

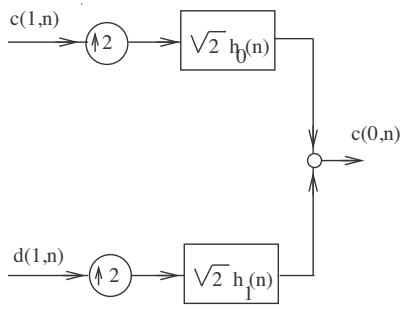


Figure 2.14
Reconstruction of a one-level multiscale signal decomposition.

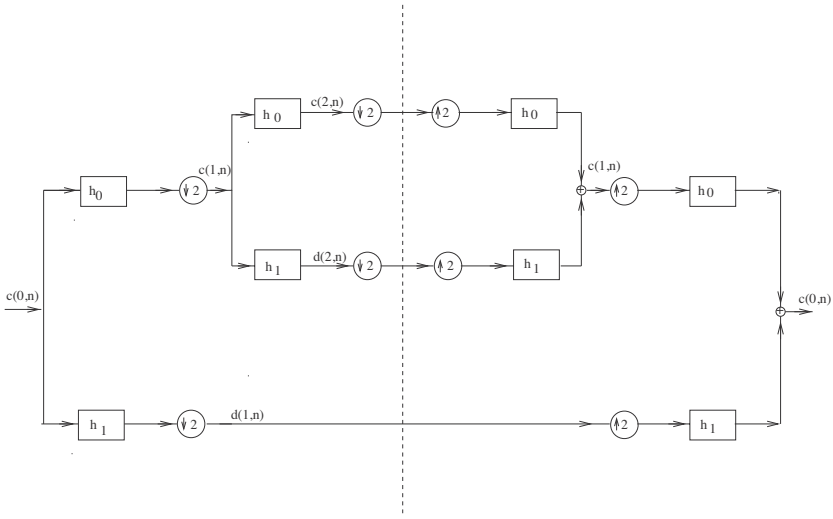


Figure 2.15
Multiscale analysis and synthesis.

Wavelet Transformation at a Finite Resolution

In this section we will show that a function can be approximated to a desired degree by summing the scaling function and as many wavelet detail functions as necessary. Let $f \in V_0$ be defined as

$$f(t) = \sum c_{0,n}\varphi(t-n) \tag{2.76}$$

As stated in previous sections, it also can be represented as a sum of a signal at a coarser resolution (approximation) plus a detailed signal (approximation error):

$$f(t) = f_v^1(t) + f_w^1(t) = \sum c_{1,n}2^{\frac{1}{2}}\varphi\left(\frac{t}{2}-n\right) + \sum d_{1,n}2^{\frac{1}{2}}\psi\left(\frac{t}{2}-n\right) \tag{2.77}$$

The coarse approximation $f_v^1(t)$ can be rewritten as

$$f_v^1(t) = f_v^2(t) + f_w^2(t) \tag{2.78}$$

such that

$$f(t) = f_v^2(t) + f_w^2(t) + f_w^1(t) \tag{2.79}$$

Continuing with this procedure we have at scale J for $f_v^J(t)$

$$f(t) = f_v^J(t) + f_w^J(t) + f_w^{J-1}(t) + \cdots + f_w^1 \tag{2.80}$$

or

$$f(t) = \sum_{n=-\infty}^{\infty} c_{J,n}\varphi_{J,n}(t) + \sum_{m=1}^{J}\sum_{n=-\infty}^{\infty} d_{m,n}\psi_{m,n}(t) \tag{2.81}$$

This equation describes a wavelet series expansion of function $f(t)$ in terms of the wavelet $\psi(t)$ and scaling function $\varphi(t)$ for an arbitrary scale J. In comparison, the pure WT,

$$f(t) = \sum_m \sum_n d_{m,n}\psi_{mn}(t) \tag{2.82}$$

requires an infinite number of resolutions for a complete signal representation.

From equation (2.82) we can see that $f(t)$ is given by a coarse approximation at the scale L and a sum of L detail components (wavelet components) at different resolutions.

Example 2.3: Consider the simple function

$$y = \begin{cases} t^2, & 0 \le t \le 1 \\ 0, & \text{else} \end{cases} \tag{2.83}$$

Using Haar wavelets and the starting scale $J = 0$, we can easily determine the following expansion coefficients:

$$\begin{aligned} c_{0,0} &= \int_0^1 t^2 \varphi_{0,0}(t)dt = \frac{1}{3} \\ d_{0,0} &= \int_0^1 t^2 \psi_{0,0}(t)dt = -\frac{1}{4} \\ d_{1,0} &= \int_0^1 t^2 \psi_{1,0}(t)dt = -\frac{\sqrt{2}}{32} \\ d_{1,1} &= \int_0^1 t^2 \psi_{1,1}(t)dt = -\frac{3\sqrt{2}}{32} \end{aligned} \tag{2.84}$$

Thus, we obtain the wavelet series expansion

$$y = \frac{1}{3}\varphi_{0,0}(t) - \frac{1}{4}\psi_{0,0}(t) - \frac{\sqrt{2}}{32}\psi_{1,0}(t) - \frac{3\sqrt{2}}{32}\psi_{1,1}(t) + \cdots \tag{2.85}$$

2.5 Overview: Types of Wavelet Transforms

The goal of this section is to provide an overview of the most frequently used wavelet types. Figure 2.16 illustrates the block diagram of the generalized time-discrete filter bank transform. It is important to point out that there is a strong analogy between filter banks and wavelet bases: the low-pass filter coefficients of the filter bank determine the scaling functions while the high-pass filter coefficients produce the wavelets.

The mathematical representation of the direct and inverse generalized time–discrete filter bank transform is

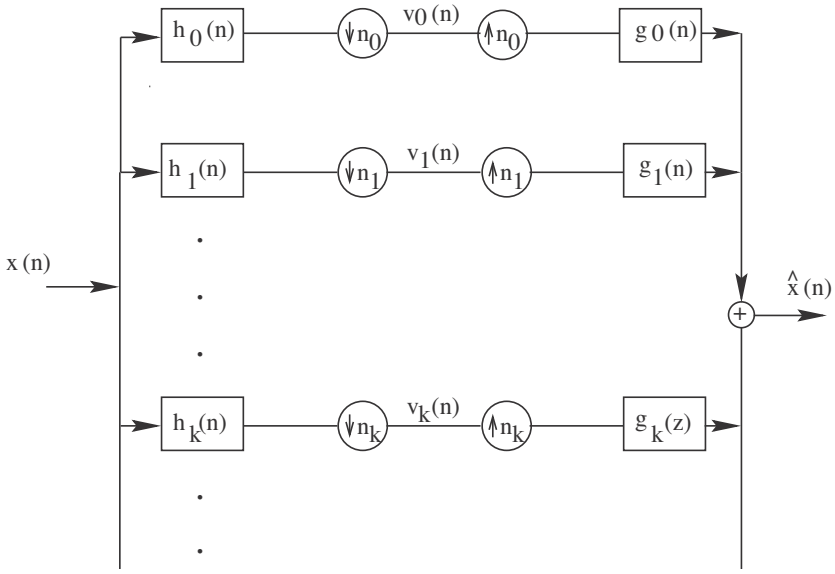


Figure 2.16
Generalized time-discrete filter bank transform.

$$v_k(n) = \sum_{m=-\infty}^{\infty} x(m) h_k(n_k n - m), \quad 0 \le k \le M-1 \tag{2.86}$$

and

$$\widehat{x}(n) = \sum_{k=0}^{M-1} \sum_{m=-\infty}^{\infty} v_k(m) g_k(n - n_k m) \tag{2.87}$$

Based on this representation, we can derive as functions of n_k, $h_k(n)$, and $g_k(n)$ the following special cases [78]:

1. **Orthonormal wavelets:** $n_k = 2^k$ with $0 \le k \le M-2$ and $n_{M-1} = n_{M-2}$. The basis function fulfills the orthonormality condition (2.36).
2. **Orthonormal wavelet packets:** They represent a generalization of the orthonormal wavelets because they use the recursive decomposition-reconstruction structure which is applied to all bands. The following holds: $n_k = 2^L$ with $0 \le k \le 2^L - 1$.

3. **Biorthogonal wavelets:** They have properties similar to those of the orthogonal wavelets but are less restrictive.
4. **Generalized filter bank representations:** They represent a generalization of the (bi)orthogonal wavelet packets. Each band is split into two subbands. The basis functions fulfill the biorthonormality condition:

$$\sum_{m=-\infty}^{\infty} g_c(m - n_c l) h_k(n_k n - m) = \delta(c - k)\delta(l - n). \tag{2.88}$$

5. **Oversampled wavelets:** There is no downsampling or oversampling required, and $n_k = 1$ holds for all bands.

The first four wavelet types are known as *nonredundant wavelet representations*. For the representation of oversampled wavelets, more analysis functions ($\{u_k(n)\}$) than basis functions are required. The analysis and synthesis functions must fulfill

$$\sum_{k=0}^{M-1} \sum_{m=-\infty}^{\infty} g_k(m - l) h_k(n - m) = \delta(l - n). \tag{2.89}$$

This condition holds only in the case of linear dependency. This means that some functions are represented as linear combinations of others.

2.6 The Two-Dimensional Discrete Wavelet Transform

For any wavelet orthonormal basis $\{\psi_{j,n}\}_{(j,n)\in Z^2}$ in $L^2(R)$, there also exists a separable wavelet orthonormal basis in $L^2(R)$:

$$\{\psi_{j,n}(x)\psi_{l,m}(y)\}_{(j,l,n,m)\in Z^4} \tag{2.90}$$

The functions $\psi_{j,n}(x)\psi_{l,m}(y)$ mix the information at two different scales 2^j and 2^l, across x and y. This technique leads to a building procedure based on separable wavelets whose elements represent products of function dilation at the same scale. These multiscale approximations are mostly applied in image processing because they facilitate the processing of images at several detail levels. Low-resolution images can be represented using fewer pixels while preserving the features necessary for recognition tasks.

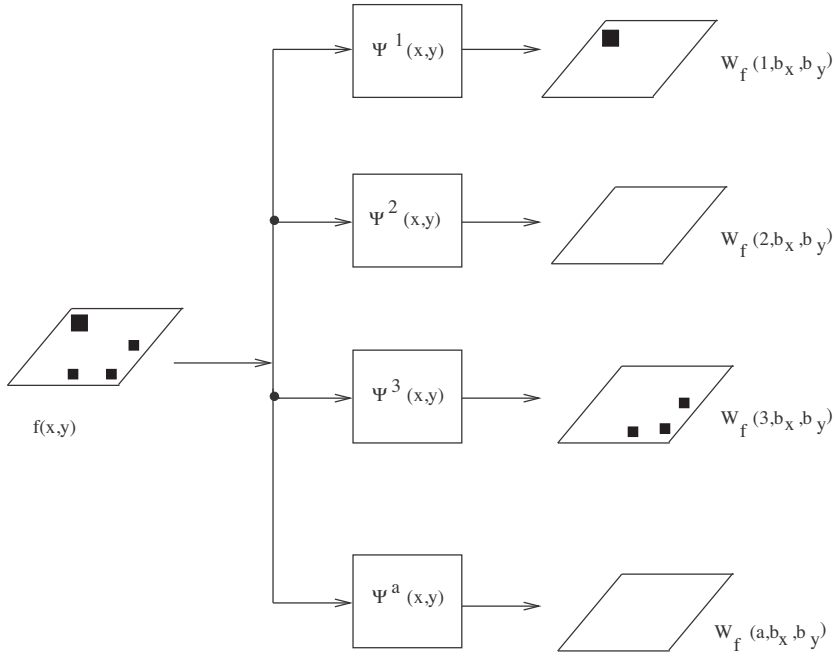


Figure 2.17
Filter bank analogy of the WT of an image.

The theory presented for the one-dimensional WT can easily be extended to two-dimensional signals such as images. In two dimensions, a 2-D scaling function, $\varphi(x, y)$ and three 2D wavelets $\psi^1(x, y), \psi^2(x, y)$, and $\psi^3(x, y)$ are required. Figure 2.17 shows a 2-D filter bank. Each filter $\psi_a(x, y)$ represents a 2-D impulse response, and its output, a bandpass filtered version of the original image. The set of the filtered images describes the WT.

In the following, we will assume that the 2-D scaling functions are separable. That is:

$$\varphi(x, y) = \varphi(x)\varphi(y) \tag{2.91}$$

where $\varphi(x)$ is a one–dimensional scaling function. If we define $\psi(x)$, the companion wavelet function, as shown in equation (2.52), then based on the following three basis functions,

$$\psi^1(x,y) = \varphi(x)\psi(y) \quad \psi^2(x,y) = \psi(x)\varphi(y) \quad \psi^3(x,y) = \psi(x)\psi(y) \tag{2.92}$$

we set up the foundation for the 2-D wavelet transform. Each of them is the product of a one-dimensional scaling function φ and a wavelet function ψ. They are "directionally sensitive" wavelets because they measure functional variations, either intensity or gray-level variations, along different directions: ψ^1 measures variations along the columns (horizontal edges), ψ^2 is sensitive to variations along rows (vertical edges), and ψ^3 corresponds to variations along diagonals. This directional sensitivity is an implication of the separability condition.

To better understand the 2-D WT, let us consider $f_1(x,y)$, an $N \times N$ image, where the subscript describes the scale and N is a power of 2. For $j = 0$, the scale is given by $2^j = 2^0 = 1$, and corresponds to the original image. Allowing j to become larger doubles the scale and halves the resolution.

An image can be expanded in terms of the 2-D WT. At each decomposition level, the image can be decomposed into four subimages a quarter of the size of the original, as shown in figure 2.18. Each of these images stems from an inner product of the original image with the subsampled version in x and y by a factor of 2. For the first level ($j = 1$), we obtain

$$\begin{aligned} f_2^0(m,n) &= < f_1(x,y), \varphi(x-2m, y-2n) > \\ f_2^1(m,n) &= < f_1(x,y), \psi^1(x-2m, y-2n) > \\ f_2^2(m,n) &= < f_1(x,y), \psi^2(x-2m, y-2n) > \\ f_2^3(m,n) &= < f_1(x,y), \psi^3(x-2m, y-2n) > . \end{aligned} \tag{2.93}$$

For the subsequent levels ($j > 1$), $f_{2^j}^0(x,y)$ is decomposed in a similar way, and four quarter-size images at level 2^{j+1} are formed. This procedure is visualized in figure 2.18.

The inner products can also be written as a convolution:

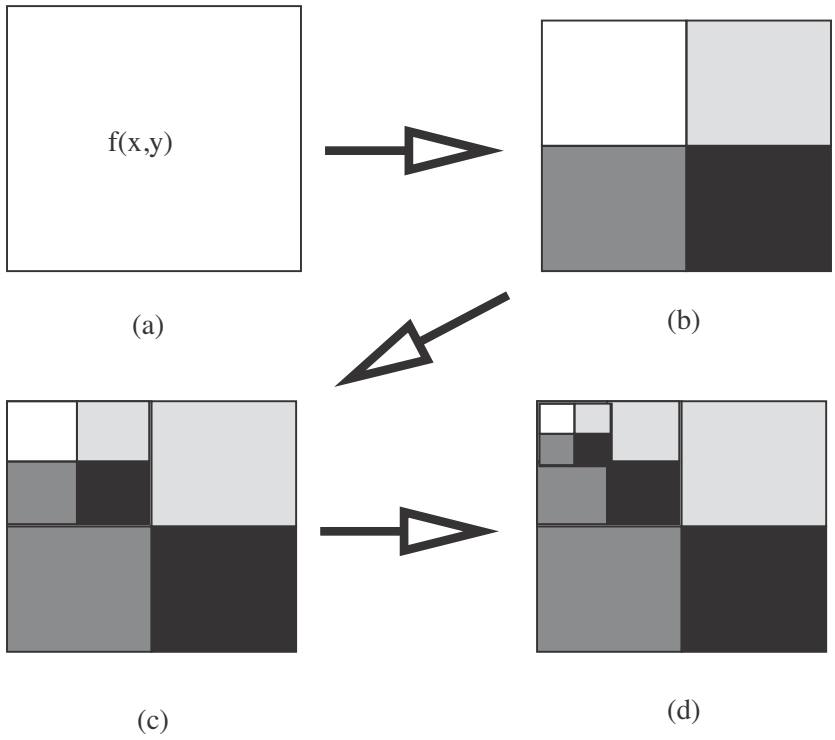


Figure 2.18
2-D discrete wavelet transform: (a) original image; (b) first, (c) second, and (d) third levels.

$$
\begin{aligned}
f^0_{2^{j+1}}(m,n) &= \left\{[f^0_{2^j}(x,y) * \varphi(x,y)](2m,2n)\right\} \\
f^1_{2^{j+1}}(m,n) &= \left\{[f^0_{2^j}(x,y) * \psi^1(x,y)](2m,2n)\right\} \\
f^2_{2^{j+1}}(m,n) &= \left\{[f^0_{2^j}(x,y) * \psi^2(x,y)](2m,2n)\right\} \\
f^3_{2^{j+1}}(m,n) &= \left\{[f^0_{2^j}(x,y) * \psi^3(x,y)](2m,2n)\right\}.
\end{aligned}
\tag{2.94}
$$

The scaling and the wavelet functions are separable, and therefore we can replace every convolution by a 1-D convolution on the rows and columns of $f^0_{2^j}$. Figure 2.20 illustrates this fact. At level 1, we convolve the rows of the image $f_1(x,y)$ with $h_0(x)$ and with $h_1(x)$, then eliminate the odd-numbered columns (the leftmost is set to zero) of the two

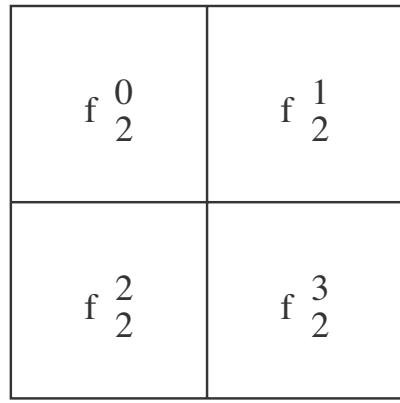


Figure 2.19
DWT decomposition in the frequency domain.

resulting arrays. The columns of each $N/2 \times N$ are then convolved with $h_0(x)$ and $h_1(x)$, and the odd-numbered rows are eliminated (the top row is set to zero). As an end result we obtain the four $N/2 \times N/2$ arrays required for that level of the WT. Figure 2.19 illustrates the localization of the four newly obtained images in the frequency domain. $f^0_{2^j}(x, y)$ describes the low-frequency information of the previous level, while $f^1_{2^j}(x, y)$, $f^2_{2^j}(x, y)$, and $f^3_{2^j}(x, y)$ represent the horizontal, vertical, and diagonal edge information.

The inverse WT is shown in figure 2.20. At each level, each of the arrays obtained on the previous level is upsampled by inserting a column of zeros to the left of each column. The rows are then convolved with either $h_0(x)$ or $h_1(x)$, and the resulting $N/2 \times N$ arrays are added together in pairs. As a result, we get two arrays which are oversampled to achieve an $N \times N$ array by inserting a row of zeros above each row. Next, the columns of the two new arrays are convolved with $h_0(x)$ and $h_1(x)$, and the two resulting arrays are added together. The result shows the reconstructed image for a given level.

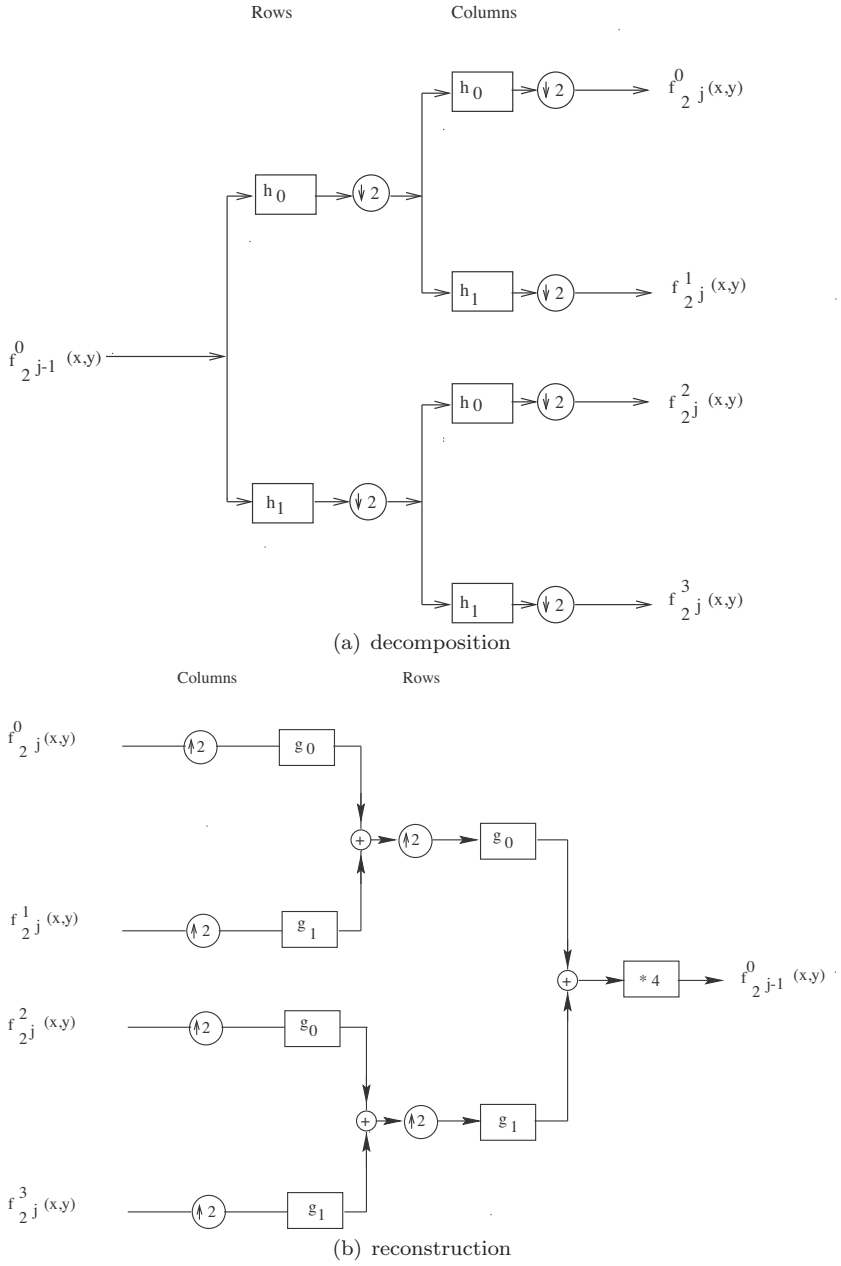


Figure 2.20
Image decomposition (a) and reconstruction (b) based on discrete WT.

EXERCISES

1. Consider the continuous-time signal

$$f(t) = 3\cos(400\pi t) + 5\sin(1200\pi t) + 6\cos(4400\pi t) + 2\sin(5200\pi t). \tag{2.95}$$

Determine its continuous Fourier transform.

2. Compute the DFT for the following signal:

$$x[n] = \cos(2\pi rn/N), \quad 0 \le n \le N-1, 0 \le r \le N-1 \tag{2.96}$$

3. Prove the linearity property for the discrete cosine transform (DCT) and discrete sine transform (DST).

4. What is the difference between the continuous and discrete wavelet transforms?

5. Comment on the differences and applicability of the discrete cosine transform and the wavelet transform to medical image compression.

6. Show if the scaling function

$$\varphi(t) = \begin{cases} 1, & 0.5 \le t < 1 \\ 0, & \text{else} \end{cases}$$

satisfies the inclusion requirement of the multiresolution analysis.

7. Compute the Haar transform of the image

$$\mathbf{I} = \begin{bmatrix} 4 & -1 \\ 8 & 2 \end{bmatrix} \tag{2.97}$$

8. Consider the following function

$$\varphi(t) = \begin{cases} t^3, & 0 \le t < 1 \\ 0, & \text{else} \end{cases}$$

Using the Haar wavelet and starting at scale 0, give a multiscale decomposition of this signal.

9. Plot the wavelet $\psi_{5,5}(t)$ for the Haar wavelet function. Express $\psi_{5,5}$

in terms of the Haar scaling function.

10. Verify if the following holds for the Haar wavelet family:
a) $\varphi(2t) = \sum h_0(n)\varphi(4t-k)$ and
b) $\psi(2t) = \sum h_1(n)\psi(4t-k)$.

11. The function $f(t)$ is given as

$$f(t) = \begin{cases} 8, & 0 \le t < 4 \\ 0, & \text{else} \end{cases}$$

Plot the following scaled and/or translated versions of $f(t)$:
a) $f(t-1)$
b) $f(2t)$
c) $f(2t-1)$
d) $f(8t)$

12. Write a program to compute the CWT of a medical image and use it to determine a small region of interest (tumor) in the image.

13. Write a program to compute the DWT of a medical image of an aneurysm and use this program to detect edges in the image.

14. Write a program to compute the DWT of a medical image and use this program to denoise the image by hard thresholding. Hint: First choose the number of levels or scales for the decomposition and then set to zero all elements whose absolute values are lower than the threshold.

3 Information Theory and Principal Component Analysis

In this chapter, we introduce algorithms for data analysis based on statistical quantities. This probabilistic approach to explorative data analysis has become an important branch in machine learning with many applications in life sciences.

We first give a short, somewhat technical review of necessary concepts from probability and estimation theory. We then introduce some key elements from information theory, such as entropy and mutual information. As a first data analysis method, we finish this chapter by discussing an important and often used preprocessing technique, principal component analysis.

3.1 Probability Theory

In this section we summarize some important facts from probability theory which are needed later. The basic measure theory required for the probability theoretic part can be found in many books, such as [22].

Random Functions

In this section we follow the first chapter of [23]. We give only proofs that are not in [244].

DEFINITION 3.1: A *probability space* $(\Omega, \mathbf{A}, P)$ consists of a set Ω, a σ-algebra $\mathbf{A}$ on Ω, and a measure P called *probability measure* on $\mathbf{A}$ with $P(\Omega) = 1$.

While this may sound confusing, the intuitive notion is very simple: For some subsets of our space Ω, we specify how probable they are. Clearly, we want intersections and unions also to have probabilities, and this (in addition to some technicality with respect to infinite unions) is what is implied by the σ-algebra.

Elements of $\mathbf{A}$ are called events, and $P(A)$ is called the *probability of the event* A. By definition we have

$$0 \leq P(A) \leq 1.$$

As usual we us $L^1(\Omega, \mathbb{R}^n)$ to denote the Banach space of all equivalence classes of integrable functions from Ω to $\mathbb{R}^n$, and $L^2(\Omega, \mathbb{R}^n)$ to denote the Hilbert space of all equivalence classes of square-integrable functions. Note that this is a subset.

The notion of a random variable is one of the key concepts of probability theory.

DEFINITION 3.2: If $(\Omega, \mathbf{A}, P)$ is a probability space and $(\Omega', \mathbf{A}')$ is a measurable space, then an $(\mathbf{A}, \mathbf{A}')$-measurable mapping $X : \Omega \longrightarrow \Omega'$ is called a *random function* with values in Ω'.

If $(\Omega', \mathbf{A}') = (\mathbb{R}, \mathbf{B}(\mathbb{R}))$ are the real numbers together with the *Borel sigma algebra* (i.e. the sigma algebra generated by the half-open intervals), then such a random function is also called a *random variable*. Note that an $X : \Omega \to \mathbb{R}$ is a random variable over the probability space $(\Omega, \mathbf{A})$ if and only if $X^{-1}(a, b] \in \mathbf{A}$ for all $-\infty \leq a < b \leq \infty$. Similarly, for $(\Omega', \mathbf{A}') = (\mathbb{R}^n, \mathbf{B}(\mathbb{R}^n))$ we speak of a *random vector*.

Although initially possibly confusing due to the notation, a function X from some probability space to the real numbers is a random function if it assigns a probability to intervals of $\mathbb{R}$. Later we will see under what (weak) conditions we can simply assign a density to this function X. Then this coincides with the possibly more intuitive notion of a probability density on $\mathbb{R}$. In this chapter we use capitals for random functions in order to not confuse them with points from $\mathbb{R}^n$. In later chapters, such confusion will rarely occur, and we will often use x or $x(t)$ to describe a random function.

Given a random function $X : (\Omega, \mathbf{A}, P) \to (\Omega', \mathbf{A}', P')$, we define a mapping

$$\begin{array}{rcl} X(P) : \mathbf{A}' & \longrightarrow & \mathbb{R}_0^+ \\ A' & \longmapsto & X(P)(A') := P\{X \in A'\} := P(X^{-1}(A')). \end{array}$$

Since $P\{X \in \Omega'\} = P(\Omega) = 1$, this defines a probability measure on $\mathbf{A}'$ called the *image measure* $X(P)$ of P under X.

DEFINITION 3.3: Let X be a random function. The image measure $X(P)$ is called the *distribution* of X with respect to P, and we write

$$P_X := X(P).$$

For $A' \in \mathbf{A}'$ we have

$$P_X(A') = P\{X \in A'\}.$$

DEFINITION 3.4: If $\mathbf{X} : \Omega \longrightarrow \mathbb{R}^n$ denotes a random vector on a probability space $(\Omega, \mathbf{A}, P)$, then

$$\begin{array}{rcl} F_{\mathbf{X}} : \mathbb{R}^n & \longrightarrow & [0,1] \\ (x_1, \ldots, x_n) & \longmapsto & P_{\mathbf{X}}((-\infty, x_1] \times \ldots \times (-\infty, x_n]) \end{array}$$

is called the *distribution function* of X with respect to P.

If $n = 1$, then X is a random variable. Then its distribution function F_X is monotonic-increasing, and right-continuous and $\lim_{x \to -\infty} X(x) = 0$, $\lim_{x \to \infty} X(x) = 1$.

If the image measure $P_{\mathbf{X}}$ of a random vector $\mathbf{X}$ on $\mathbb{R}^n$ can be written as

$$P_{\mathbf{X}} = p_{\mathbf{X}} \lambda^n,$$

with a function $p_{\mathbf{X}} : \mathbb{R}^n \to \mathbb{R}$ and the Lebesgue-measure λ^n on $\mathbb{R}^n$, then the random vector is said to be *continuous* and $p_{\mathbf{X}}$ is called the *density of* $\mathbf{X}$. $\mathbf{X}$ has a density according to the Radon-Nikodym theorem [22] if $\mathbf{X}$ is continuous with respect to the Lebesgue-measure.

For example, if a random variable has a density

$$p_X = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-m)^2}{2\sigma^2}\right)$$

with $\sigma > 0, m \in \mathbb{R}$, then it is said to be a *Gaussian random variable*. If $\sigma = 1$ and $m = 0$, it is called *normal*.

Note that if $\mathbf{X}$ is a random vector with density $p_{\mathbf{X}}$, then $\frac{\partial^n}{\partial x_1 \ldots \partial x_n} F_{\mathbf{X}}$ exists almost everywhere and

$$\frac{\partial^n}{\partial x_1 \ldots \partial x_n} F_{\mathbf{X}} = p_{\mathbf{X}}$$

also exists almost everywhere.

THEOREM 3.1 TRANSFORMATION OF DENSITIES: Let $\mathbf{X}$ be an n-dimensional random vector with density $p_{\mathbf{X}}$ and $\mathbf{h} : U \longrightarrow V$ a C^1-diffeomorphism with $U, V \subset \mathbb{R}^n$ open and $\operatorname{supp} p_{\mathbf{X}} \subset U$. Then $\mathbf{h} \circ \mathbf{X}$ has

the density

$$p_{\mathbf{h}\circ\mathbf{X}} \circ \mathbf{h} = |\det D\mathbf{h}|^{-1}\ p_{\mathbf{X}}.$$

Expectation and moments

DEFINITION 3.5: Let $\mathbf{X}$ be a random vector on a probability space $(\Omega, \mathbf{A}, P)$. If $\mathbf{X}$ is P-integrable ($\mathbf{X} \in L^1(\Omega, \mathbb{R}^n)$), then

$$\mathbf{E}(\mathbf{X}) := \int_\Omega \mathbf{X} dP$$

is called the *expectation of* $\mathbf{X}$.

$\mathbf{E}(\mathbf{X})$ is also called the *mean* of $\mathbf{X}$ or the *first-order moment*.

LEMMA 3.1: If $\mathbf{X} \in L^1(\Omega, \mathbb{R}^n)$ then

$$\mathbf{E}(\mathbf{X}) = \int_{\mathbb{R}^n} \mathbf{x}\, dP_{\mathbf{X}}.$$

Hence $\mathbf{E}(\mathbf{X})$ is a *probability theoretic notion* (i.e. it depends only on the distribution $P_{\mathbf{X}}$ of $\mathbf{X}$). If $\mathbf{X}$ has a density $p_{\mathbf{X}}$, then

$$\mathbf{E}(\mathbf{X}) = \int_{\mathbb{R}^n} \mathbf{x} p_{\mathbf{X}}(\mathbf{x}) d\mathbf{x}.$$

The expectation is a linear mapping of the vector space $L^1(\Omega, \mathbb{R}^n)$ to $\mathbb{R}^n$, so $\mathbf{E}(\mathbf{AX}) = \mathbf{A}\mathbf{E}(\mathbf{X})$ for a matrix $\mathbf{A}$.

DEFINITION 3.6: Let $\mathbf{X} : \Omega \to \mathbb{R}^n$ be an L^2 random vector. Then

$$\begin{aligned} \mathbf{R}_{\mathbf{X}} := \mathrm{Cor}(\mathbf{X}) &:= \mathbf{E}(\mathbf{X}\mathbf{X}^\top) \\ \mathbf{C}_{\mathbf{X}} := \mathrm{Cov}(\mathbf{X}) &:= \mathbf{E}((\mathbf{X} - \mathbf{E}(\mathbf{X}))(\mathbf{X} - \mathbf{E}(\mathbf{X}))^\top) \end{aligned}$$

exist, and are called the *correlation* (respectively *covariance*) of $\mathbf{X}$.

Note that $\mathbf{X}$ is then also L^1 (i.e. integrable) and therefore $\mathbf{E}(\mathbf{X})$ exists. $\mathbf{R}_{\mathbf{X}}$ and $\mathbf{C}_{\mathbf{X}}$ are symmetric and positive semidefinite (i.e. $\mathbf{a}^\top \mathbf{R}_{\mathbf{X}} \mathbf{a} \geq 0$ for all $\mathbf{a} \in \mathbb{R}^n$). If $\mathbf{X}$ has no *deterministic* component (i.e. a component with constant image), then the two matrices are positive-definite, meaning that $\mathbf{a}^\top \mathbf{R}_{\mathbf{X}} \mathbf{a} > 0$ for $\mathbf{a} \neq 0$. Since the above equations are quadratic in $\mathbf{X}$, the components of $\mathbf{R}$ are called the *second-order moments* of $\mathbf{X}$

and the components of $\mathbf{C}$ are the *central second-order moments*. If $n = 1$, then

$$\operatorname{var} X := \sigma_X := E((X - m_X)^2) = C_X$$

is called the *variance* of X. Its square root σ_X is called the *standard deviation* of X.

The central moments and the general second-order ones are related as follows:

$$\mathbf{R_X} = \mathbf{C_X} + \mathbf{m_X}\mathbf{m_X}^\top.$$

Decorrelation and Independence

We are interested in analyzing the structure of random vectors. A simple question to ask is how strongly they depend on each other. This we can measure in first approximation using correlations. By taking into account higher-order correlations, we later arrive at the notion of dependent and independent random vectors.

DEFINITION 3.7: Let $\mathbf{X} : \Omega \to \mathbb{R}^n$ be an arbitrary random vector. If $\operatorname{Cov}(\mathbf{X})$ is diagonal, then $\mathbf{X}$ is called (mutually) *decorrelated*. $\mathbf{X}$ is said to be white or *whitened* if $\mathbf{E}(\mathbf{X}) = 0$ and $\operatorname{Cov}(\mathbf{X}) = \mathbf{I}$ (i.e. if $\mathbf{X}$ is centered and decorrelated with unit variance components). A *whitening transformation* of $\mathbf{X}$ is a matrix $\mathbf{W} \in \mathrm{Gl}(n)$ such that $\mathbf{WX}$ is whitened.

Note that $\mathbf{X}$ is white if and only if $\mathbf{AX}$ is white for an orthogonal matrix $\mathbf{A} \in O(n) = \{\mathbf{A} \in \mathrm{Gl}(n) | \mathbf{A}\mathbf{A}^\top = \mathbf{I}\}$, which follows directly from

$$\operatorname{Cov}(\mathbf{AX}) = \mathbf{A} \operatorname{Cov}(\mathbf{X}) \mathbf{A}^\top.$$

LEMMA 3.2: Given a centered random vector $\mathbf{X}$ with nondeterministic components, there exists a whitening transformation of X, and it is unique modulo $O(n)$.

Proof Let $\mathbf{C} := \operatorname{Cov}(\mathbf{X})$ be the covariance matrix of $\mathbf{X}$. $\mathbf{C}$ is symmetric, so there exists $\mathbf{V} \in O(n)$ such that $\mathbf{VCV}^\top = \mathbf{D}$ with $\mathbf{D} \in \mathrm{Gl}(n)$ diagonal and positive. Set $\mathbf{W} := \mathbf{D}^{-1/2}\mathbf{V}$, where $\mathbf{D}^{-1/2}$ denotes a diagonal matrix (square root) with $\mathbf{D}^{-1/2}\mathbf{D}^{-1/2} = \mathbf{D}^{-1}$. Then, using the fact that $\mathbf{X}$ is

centered, we get

$$\begin{aligned}\mathrm{Cov}(\mathbf{WX}) &= \mathbf{E}(\mathbf{WXX}^\top\mathbf{W}^\top)\\ &= \mathbf{WCW}^\top\\ &= \mathbf{D}^{-1/2}\mathbf{VCV}^\top\mathbf{D}^{-1/2}\\ &= \mathbf{D}^{-1/2}\mathbf{DD}^{-1/2} = \mathbf{I}.\end{aligned}$$

If $\mathbf{V}$ is another whitening transformation of $\mathbf{X}$, then

$$\mathbf{I} = \mathrm{Cov}(\mathbf{VX}) = \mathrm{Cov}(\mathbf{VW}^{-1}\mathbf{WX}) = \mathbf{VW}^{-1}\mathbf{W}^{-\top}\mathbf{V}^\top$$

so $\mathbf{VW}^{-1} \in O(n)$. ■

So decorrelation clearly gives insight into the structure of a random vector but does not yield a unique transformation. We will therefore turn to a more stringent constraint.

Definition 3.8: A finite sequence $(X_i)_{i=1,\ldots,n}$ of random functions with values in the probability space Ω_i with σ-algebra $\mathbf{A}_i$ is called *independent* if

$$P\{X_1 \in A_1, \ldots, X_n \in A_n\} := P\left(\bigcap_{i=1}^{n} X_i^{-1}(A_i)\right) = \prod_{i=1}^{n} P\{X_i \in A_i\}$$

for all $A_i \in \mathbf{A}_i$, $i = 1, \ldots, n$. A random vector $\mathbf{X}$ is called *independent* if the family $(X_i)_i := (\pi_i \circ \mathbf{X})_i$ of its components is independent.

Here π_i denotes the projection onto the i-th coordinate. If $\mathbf{X}$ is a random vector with density $p_{\mathbf{X}}$, then it is independent if and only if the density factorizes into one-dimensional functions. That is,

$$p_{\mathbf{X}}(x_1, \ldots, x_n) = p_{X_1}(x_1) \ldots p_{X_n}(x_n)$$

for all $(x_1, \ldots, x_n) \in \mathbb{R}^n$. Here, the p_{X_i} are also often called the *marginal densities* of $\mathbf{X}$.

Note that it is easy to see that independence is a probability theoretic term. Examples for independent random vectors will be given later.

Definition 3.9: Given two n- respectively m-dimensional random vectors $\mathbf{X}$ and $\mathbf{Y}$ with densities, the joint density $p_{\mathbf{X},\mathbf{Y}}$ is the density of the $n+m$-dimensional random vector $(\mathbf{X}, \mathbf{Y})^\top$. For given $\mathbf{y}_0 \in \mathbb{R}^m$

with $p_{\mathbf{Y}}(\mathbf{y}_0) \neq 0$, the *conditional density* of $\mathbf{X}$ with respect to $\mathbf{Y}$ is the function

$$p_{\mathbf{X}|\mathbf{Y}}(\mathbf{x}|\mathbf{y}_0) = \frac{p_{\mathbf{X},\mathbf{Y}}(\mathbf{x}, \mathbf{y}_0)}{p_{\mathbf{Y}}(\mathbf{y}_0)}$$

for $\mathbf{x} \in \mathbb{R}^n$.

Indeed, it is possible to define a conditional random vector $\mathbf{X}|\mathbf{Y}$ with density $p_{\mathbf{X}|\mathbf{Y}}(\mathbf{x}|\mathbf{y}_0)$.

Note that if $\mathbf{X}$ and $\mathbf{Y}$ are independent, meaning that their joint density factorizes, then $p_{\mathbf{X}|\mathbf{Y}}(\mathbf{x}|\mathbf{y}_0) = p_{\mathbf{X}}$. More generally we get

$$p_{\mathbf{X}|\mathbf{Y}}(\mathbf{x}_0|\mathbf{y}_0) = p_{\mathbf{X}|\mathbf{Y}}(\mathbf{x}_0|\mathbf{y}_0)p_{\mathbf{Y}}(\mathbf{y}_0) = p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}_0|\mathbf{x}_0)p_{\mathbf{X}}(\mathbf{x}_0),$$

so we have shown *Bayes's rule*:

$$p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}_0|\mathbf{x}_0) = \frac{p_{\mathbf{X}|\mathbf{Y}}(\mathbf{x}_0|\mathbf{y}_0)p_{\mathbf{Y}}(\mathbf{y}_0)}{p_{\mathbf{X}}(\mathbf{x}_0)}$$

Operations on Random Vectors

In this section we present two different methods for constructing new random vectors out of given ones in order to get certain properties. The first of these properties is the vanishing mean.

Definition 3.10: A random vector $\mathbf{X} : \Omega \to \mathbb{R}^n$ is called *centered* if $\mathbf{E}(\mathbf{X}) = 0$.

Lemma 3.3: Let $\mathbf{X} : \Omega \to \mathbb{R}^n$ be a random vector. Then $\mathbf{X} - \mathbf{E}(\mathbf{X})$ is centered.

Proof $\mathbf{E}(\mathbf{X} - \mathbf{E}(\mathbf{X})) = \mathbf{E}(\mathbf{X}) - \mathbf{E}(\mathbf{X}) = 0$. ■

Another construction we want to make is the restriction of a random vector in the sense that only samples from a given region are taken into account. This notion is formalized in next lemma 3.4.

Lemma 3.4: Let $\mathbf{X} : \Omega \to \mathbb{R}^n$ be a random vector, and let $U \subset \mathbb{R}^n$ be measurable with $P_{\mathbf{X}}(U) = P(\mathbf{X}^{-1}(U)) > 0$. Then

$$\begin{array}{rcl} \mathbf{X}|U : \mathbf{X}^{-1}(U) & \longrightarrow & \mathbb{R}^n \\ \omega & \longmapsto & \mathbf{X}(\omega) \end{array}$$

defines a new random vector on $\left(\mathbf{X}^{-1}(U), \mathbf{A}'\right)$ with σ-algebra $\mathbf{A}' := \{A \in \mathbf{A} \,|\, A \subset \mathbf{X}^{-1}(U)\}$ and probability measure

$$P'(A) = \frac{P(A)}{P_{\mathbf{X}}(U)}$$

for $A \in \mathbf{A}'$. It is called the *restriction* of $\mathbf{X}$ to U.

LEMMA 3.5 TRANSFORMATION PROPERTIES OF RESTRICTION: Let $\mathbf{X}, \mathbf{Y} : \Omega \to \mathbb{R}$ be random variables with densities $p_{\mathbf{X}}$ and $p_{\mathbf{Y}}$ respectively, and let $U \subset \mathbb{R}^n$ with $P_{\mathbf{X}}(U), P_{\mathbf{Y}}(U) > 0$.

i. $(\lambda\mathbf{X})|(\lambda U) = \lambda\mathbf{X}|U$ if $\lambda \in \mathbb{R}$.
ii. $(\mathbf{A}\mathbf{X})|(\mathbf{A}U) = \mathbf{A}(\mathbf{X}|U)$ if $\mathbf{A} \in \mathrm{Gl}(n)$.
iii. If $\mathbf{X}$ is independent and $U = [a_1, b_1] \times \ldots \times [a_n, b_n]$, then $\mathbf{X}|U$ is independent.

We can construct samples of $\mathbf{X}|U$ given samples $\mathbf{x}_1, \ldots, \mathbf{x}_s$ of $\mathbf{X}$ by taking all samples that lie in U.

Examples of Probability Distributions

In this section, we give some important examples of random vectors. In particular, Gaussian distributed random vectors will play a key role in ICA. The probability density functions of the following random vectors in the one-dimensional case are plotted in figure 3.4.

Uniform Density

For a subset $K \subset \mathbb{R}^n$ let χ_K denote the *characteristic function* of K:

$$\begin{array}{rccl} \chi_K : \mathbb{R}^n & \longrightarrow & \mathbb{R} & \\ \mathbf{x} & \longmapsto & \begin{cases} 1 & \mathbf{x} \in K \\ 0 & \mathbf{x} \notin K \end{cases} \end{array}$$

DEFINITION 3.11: Let $K \subset \mathbb{R}^n$, be a measurable set. A random vector $\mathbf{X} : \Omega \to \mathbb{R}^n$ is said to be uniform in K if its density function $p_{\mathbf{X}}$ exists and is of the form

$$p_{\mathbf{X}} = \frac{1}{\mathrm{vol}(K)} \chi_K$$

.

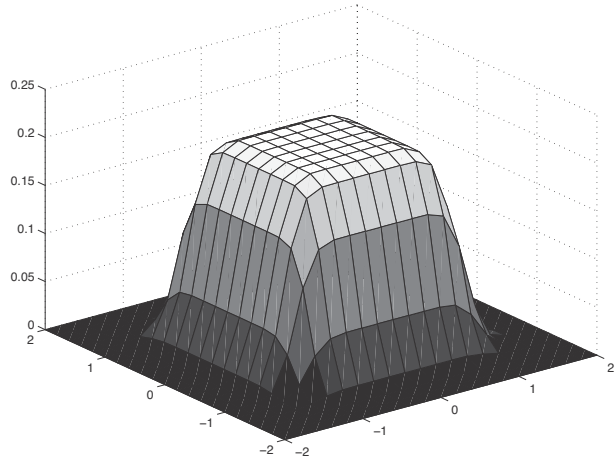


Figure 3.1
Smoothed density of a two-dimensional random vector, uniform in $[-1,1]^2$ uniform distribution.

Figure 3.1 shows a plot of the density of a uniform two-dimensional random vector.

Gaussian Density

DEFINITION 3.12: A random vector $\mathbf{X} : \Omega \to \mathbb{R}^n$ is said to be *Gaussian* if its density function $p_{\mathbf{X}}$ exists and is of the form

$$p_{\mathbf{X}}(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^n \det \mathbf{C}}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^\top \mathbf{C}^{-1}(\mathbf{x}-\mu)\right)$$

where $\mu \in \mathbb{R}^n$ and $\mathbf{C}$ is symmetric and positive-definite.

If $\mathbf{X}$ is Gaussian with μ and $\mathbf{C}$, as above, then $\mathbf{E}(\mathbf{X}) = \mu$ and $\mathrm{Cov}(\mathbf{X}) = \mathbf{C}$. A white Gaussian random vector is called normal. In the one-dimensional case a Gaussian random variable with mean $\mu \in \mathbb{R}$ and variance $\sigma^2 > 0$ has the density

$$p_X(x) = \frac{1}{\sqrt{(2\pi)\sigma}} \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right).$$

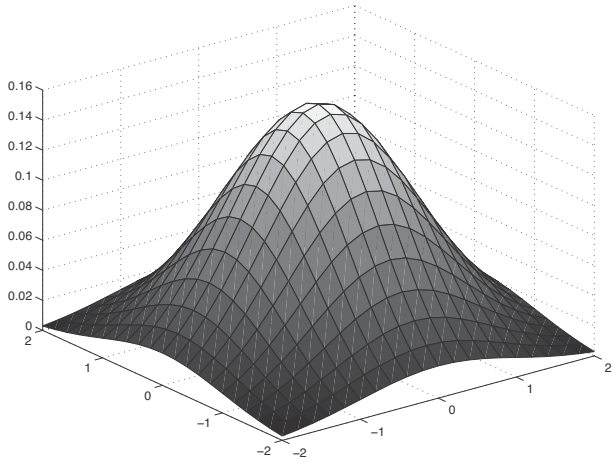


Figure 3.2
Density of a two-dimensional normal distribution i.e. a Gaussian with zero mean and unit variance.

The density of a two-dimensional Gaussian is shown in figure 3.2.

Note that a Gaussian random vector is independent if and only if it is decorrelated. Only mean and variance are needed to describe Gaussians, so it is not surprising that detection of second-order information (decorrelation) already leads to independence. Furthermore, note that the conditional density of a Gaussian is again Gaussian.

LEMMA 3.6: Let $\mathbf{X}$ be a Gaussian n-dimensional random vector and let $\mathbf{A} \in \mathrm{Gl}(n)$. Then $\mathbf{AX}$ is Gaussian. If $\mathbf{X}$ is independent, then $\mathbf{AX}$ is independent if and only if $\mathbf{A} \in O(n)$.

Proof The first- and second-order moments of $\mathbf{X}$ do not change by being multiplied by an orthogonal matrix, so if $\mathbf{A} \in O(n)$, then $\mathbf{AX}$ is independent. If, however, $\mathbf{AX}$ is independent, then $\mathbf{I} = \mathrm{Cov}(\mathbf{X}) = \mathbf{A}\,\mathrm{Cov}(\mathbf{X})\mathbf{A}^\top = \mathbf{A}\mathbf{A}^\top$, so $\mathbf{A} \in O(n)$. ■

Laplacian Density

DEFINITION 3.13: A random vector $\mathbf{X} : \Omega \to \mathbb{R}^n$ is said to be *Laplacian* if its density function $p_{\mathbf{X}}$ exists and is of the form

$$p_{\mathbf{X}}(\mathbf{x}) = \frac{\lambda}{2} \exp\left(-\lambda |\mathbf{x}|_1\right) = \frac{\lambda}{2} \exp\left(-\lambda \sum_{i=1}^{n} |x_i|\right)$$

for a fixed $\lambda > 0$.

Here $|\mathbf{x}|_1 := \sum_{i=1}^{n} |x_i|$ denotes the 1-*norm* of $\mathbf{x}$.

More generally, we can take the p-norm on $\mathbb{R}^n$ to generate γ-*distributions* or *generalized Laplacians* or *generalized Gaussians* [152]. They have the density

$$p_{\mathbf{X}}(\mathbf{x}) = C(\gamma) \exp\left(-\lambda |\mathbf{x}|_\gamma^\gamma\right) = C(\gamma) \exp\left(-\lambda \sum_{i=1}^{n} |x_i|^\gamma\right)$$

for fixed $\gamma > 0$. For the case $\gamma = 2$ we get an independent Gaussian distribution, for $\gamma = 1$ a Laplacian, and for smaller γ we get distributions with even higher kurtosis.

In figure 3.3 the density of a two-dimensional Laplacian is plotted.

Higher-Order Moments and Kurtosis

The covariance is the main second-order statistical measure used to compare two or more random variables. It basically consists of the second moment $\alpha_2(X) := E(X^2)$ of a random variable and combinations. In so-called *higher-order statistics*, too, higher *moments* $\alpha_j(X) := E(X^j)$ or *central moments* $\mu_j(X) := E((X - E(X))^j)$ are used to analyze a random variable $X : \Omega \to \mathbb{R}$.

By definition, we have $\alpha_1(X) = E(X)$ and $\mu_2(X) = \mathrm{var}(X)$. The third central moment $\mu_3(X) = E((X - E(X))^3)$, is called *skewness* of X. It measures asymmetry of its density; obviously it vanishes if X is distributed symmetrically around its mean.

Consider now the fourth moment $\alpha_4(X) = E(X^4)$ and the central moment $\mu_4(X) = E((X - E(X))^4)$. They are often used in order to determine how much a random variable is Gaussian. Instead of using the moments themselves, a combination called kurtosis is used.

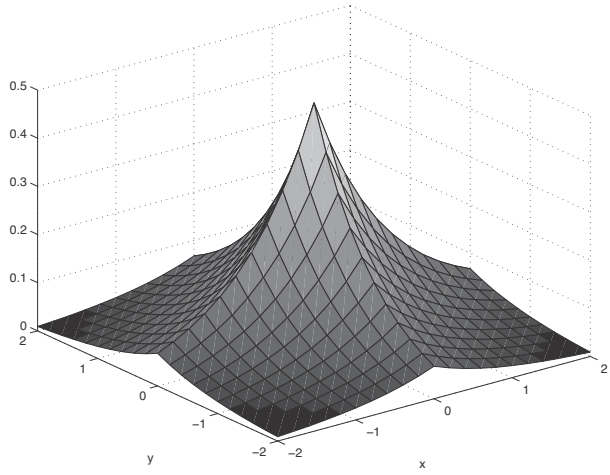


Figure 3.3
Density of a two-dimensional Laplacian random vector.

DEFINITION 3.14: Let $X : \Omega \to \mathbb{R}$ be a random variable such that

$$\operatorname{kurt}(X) := E(X^4) - 3(E(X^2))^2$$

exists. Then $\operatorname{kurt}(X)$ is called the *kurtosis* of X.

LEMMA 3.7 PROPERTIES OF THE KURTOSIS: Let $X, Y : \Omega \to \mathbb{R}$ be random variables with existing kurtosis.

i. $\operatorname{kurt}(\lambda X) = \lambda^4 \operatorname{kurt}(X)$ if $\lambda \in \mathbb{R}$.
ii. $\operatorname{kurt}(X + Y) = \operatorname{kurt}(X) + \operatorname{kurt}(Y)$ if X and Y are independent.
iii. $\operatorname{kurt}(X) = 0$ if X is Gaussian.
iv. $\operatorname{kurt}(X) < 0$ if X is uniform.
v. $\operatorname{kurt}(X) > 0$ if X is Laplacian.

Thus the kurtosis of a Gaussian vanishes. This leads to definition 3.15.

DEFINITION 3.15: Let $X : \Omega \to \mathbb{R}$ be a random variable with existing kurtosis $\operatorname{kurt}(X)$. If $\operatorname{kurt}(X) > 0$ X is called *super-Gaussian* or lep-

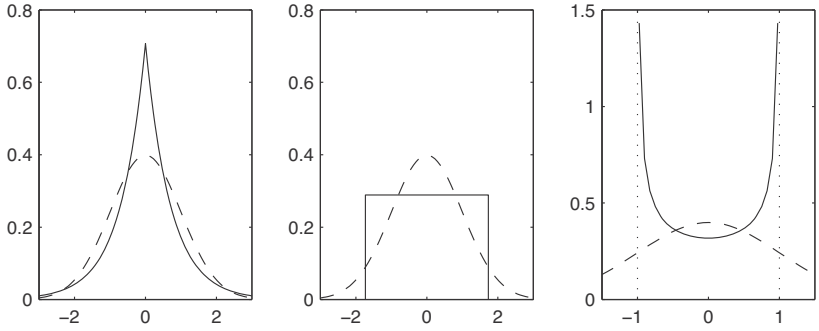


Figure 3.4
Random variables with different kurtosis. In each picture a Gaussian (kurt = 0) with zero mean and unit variance is plotted in dashed lines. The left figure shows a Laplacian distribution with $\lambda = \sqrt{2}$. In the middle figure a uniform density in $[-\sqrt{3}, \sqrt{3}]$ is shown. It has zero mean and kurtosis -1.2. The right picture shows the sub-Gaussian random variable $X := \frac{1}{\pi}\cos(Y)$ with Y uniform in $[-\pi, \pi]$. Its kurtosis is $-\frac{21}{8}$, [105]. Figures courtesy of Dr. Christoph Bauer [19].

tokurtic. If $\text{kurt}(X) < 0$, X is called *sub-Gaussian* or platykurtic. If $\text{kurt}(X) = 0$, X is said to be *mesokurtic*.

By lemma 3.7, Laplacians are superGaussian, and uniform densities are sub-Gaussian densities. In practice, superGaussian variables are often pictured as having sharper peaks and longer tails than Gaussians, whereas sub-Gaussians tend to be flatter or multimodal, as those two examples confirm. See figure 3.4 for these and more examples.

Sampling

Above, we spoke about only random functions. In actual experiments those are not known, but some samples (i.e. some values) of the random function are known. Sampling is defined in this section.

Definition 3.16: Given a finite independent sequence $(X_i)_{i=1,\dots n}$ of random functions on a probability space $(\Omega, \mathbf{A}, P)$ with the same distribution function F and an element $\omega \in \Omega$. Then the n elements $X_i(\omega)$, $i = 1, \dots, n$ are called *i.i.d. samples* of the distribution F.

Here "i.i.d." stands for "independent identically distributed". Thus sampling means executing the same experiments independently n times.

THEOREM 3.2 STRONG THEOREM OF LARGE NUMBERS: Given a pairwise i.i.d. sequence $(X_i)_{i\in\mathbf{N}}$ in $L^1(\Omega)$, then

$$\lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}(X_i(\omega)-E(X_i))=\lim_{n\to\infty}\left(\frac{1}{n}\sum_{i=1}^{n}X_i(\omega)\right)-E(X_1)=0$$

for almost all $\omega\in\Omega$.

Thus for almost all $\omega\in\Omega$ the mean of i.i.d. samples of a distribution F converge to the expectation of F if it exists. This basically means that the more samples you have, the better you can approximate a measure variable.

Theorem 3.3 explains why Gaussian random variables are so interesting and why they occur very frequently in nature.

THEOREM 3.3 CENTRAL LIMIT THEOREM: Given a pairwise i.i.d. sequence $(X_i)_{i\in\mathbf{N}}$ in $L^1(\Omega)$, and let $Y_k := \sum_{i=1}^{k} X_i$ be its sum and $Z_k := \frac{Y_k - E(Y_k)}{\operatorname{var} Y_k}$ be the normalized sum. Then the distribution of Z_k converges to a normal distribution for $k\to\infty$.

3.2 Estimation Theory

We have shown how to formulate observations subject to noise in the framework of probability theory; moreover, we have calculated some quantities such as moments within this framework. However, the full formulation clearly relies on the fact that the full random vector is known — which in practice cannot be expected. Indeed, instead of this asymptotic knowledge, only a few (or hopefully many) samples of a random vector are given, and we have to *estimate* the quantities of interest from the smaller set of samples. In this section we will show how to formulate such estimations and how to do this in practice.

Definitions and Examples

Often it is necessary to estimate parameters in a probabilistic model given a few scalar measurements or samples. The goal, given T scalars

$x(1), \ldots, x(T) \in \mathbb{R}$ is to estimate parameters $\theta_1, \ldots, \theta_n$. Such a mapping $\hat{\boldsymbol{\theta}} : \mathbb{R}^T \to \mathbb{R}^n$ is called an *estimator*.

Two examples of such estimators are the *sample mean* estimator

$$\hat{\mu} = \frac{1}{T} \sum_{i=1}^{T} x(i)$$

and the *sample variance* estimator (for $T > 1$)

$$\hat{\sigma}^2 = \frac{1}{T-1} \sum_{i=1}^{T} (x(i) - \hat{\mu}(x))^2.$$

Note that we divide by $T - 1$, not by T; this makes $\hat{\sigma}^2$ unbiased, as we will see.

In practice we distinguish between *deterministic* and *random estimators*; for the latter a distribution of the θ has to be given. Usually, an estimator is given not only for fixed T but also for all $T \in \mathbf{N}$. Instead of writing $\hat{\theta}^{(T)}$, we omit the index and write $\hat{\theta}$ for the whole family.

Such a family of estimators is said to be *online* if it can be calculated recursively:

$$\hat{\theta}^{(T+1)} = h(x(T+1), \hat{\theta}^{(T)})$$

for a fixed function h independent of T. Otherwise it is called a *batch*.

An example of an online estimator is the sample mean:

$$\hat{\mu}^{(T+1)} = \frac{T}{T+1} \hat{\mu}^{(T)} + \frac{1}{T+1} x(T+1)$$

For a given random vector X, let $\theta(X) \in \mathbb{R}$ be the value to be estimated, and let $\hat{\theta}$ be an estimator. Then

$$\tilde{\theta}(X, x(1), \ldots, x(T)) := \theta(X) - \hat{\theta}(x(1), \ldots, x(T))$$

is called the *estimation error* of $\theta(X)$ with respect to the observations $x(1), \ldots, x(T)$. If the $x(i)$ are samples of X, then $\tilde{\theta}$ should be as close to zero as possible.

Definition 3.17: If $X_1, \ldots, X_T$ are independent random variables with distribution as X, then $\hat{\theta}$ is said to be an *unbiased estimator* of θ if

$$E(\theta(X)) = E(\hat{\theta}(X_1, \ldots, X_T))$$

Similarly, it is possible to define an *asymptotically unbiased estimator* by requiring the above only in the limit. In this case such an estimator is said to be *consistent*. Note that a consistent estimator is of course not necessarily unbiased.

The sample mean $\hat{\mu}$ is an unbiased estimator of the mean of a random variable:

$$E(\hat{\mu}(X)) = \frac{1}{T}\sum_{i=1}^{T} E(x(i)) = \frac{1}{T}TE(X) = E(X)$$

Maximum Likelihood Estimation

Now we define a special random estimator that is based on partial knowledge of the distribution that is to be estimated. Namely, for given samples $x(1), \ldots, x(T)$ of a random variable X, the *maximum likelihood estimator* $\hat{\theta}_{ML}$ is chosen such that the conditional probability $p(x(1), \ldots, x(T)|\hat{\theta}_{ML})$ is maximal. This means that $\hat{\theta}_{ML}$ takes the most likely value given the observations $x(j)$.

If $\theta \mapsto p(x(1), \ldots, x(T)|\theta)$ is continuously differentiable, then by the above condition and the fact that the logarithm is strongly monotonously increasing, we get the *likelihood equation*

$$\frac{\partial}{\partial\theta_i} \ln p(x(1), \ldots, x(T)|\theta)\bigg|_{\theta=\hat{\theta}_{ML}} = 0$$

for $i = 1, \ldots, n$ if n is the dimension of the (here) multidimensional estimator θ. Here $\ln p(x(1), \ldots, x(T)|\theta)$ is also called the *log likelihood*. Using

$$p(x(1), \ldots, x(T)|\theta) = \prod_{j=1}^{T} p(x(j)|\theta),$$

the likelihood equation reads

$$\frac{\partial}{\partial\theta_i} \sum_{j=1}^{T} \ln p(x(j)|\theta)\bigg|_{\theta=\hat{\theta}_{ML}} = 0.$$

For example, assume that $x(1), \ldots, x(T)$ are samples of a Gaussian with unknown mean μ and variance σ^2, which are both to be estimated

from the samples. The conditional probability from above is

$$p(x(1),\ldots,x(T)|\mu,\sigma^2) = (2\pi\sigma^2)^{-T/2}\exp\left(-\frac{1}{2\sigma^2}\sum_{j=1}^{T}(x(j)-\mu)^2\right)$$

and hence the log likelihood is

$$\ln p(x(1),\ldots,x(T)|\mu,\sigma^2) = -\frac{T}{2}\ln(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{j=1}^{T}(x(j)-\mu)^2.$$

The likelihood equation then gives the following two equations at the maximum-likelihood estimates $(\hat{\mu}_{ML}, \hat{\sigma}^2_{ML})$:

$$\begin{aligned}
\frac{\partial}{\partial\mu}\ln p(x(1),\ldots,x(T)|\hat{\mu}_{ML},\hat{\sigma}^2_{ML}) &= \frac{1}{\hat{\sigma}^2_{ML}}\sum_{j=1}^{T}(x(j)-\hat{\mu}_{ML}) = 0 \\
\frac{\partial}{\partial\sigma^2}\ln p(x(1),\ldots,x(T)|\hat{\mu}_{ML},\hat{\sigma}^2_{ML}) &= -\frac{T}{2\hat{\sigma}^2_{ML}} + \\
&\quad \frac{1}{2\hat{\sigma}^4_{ML}}\sum_{j=1}^{T}(x(j)-\hat{\mu}_{ML}) = 0
\end{aligned}$$

From the first one, we get the maximum-likelihood estimate for the mean

$$\hat{\mu}_{ML} = \frac{1}{T}\sum_{j=1}^{T}x(j)$$

which is precisely the sample mean estimator. From the second equation, the maximum-likelihood estimator for the variance is calculated as follows:

$$\hat{\sigma}^2_{ML} = \frac{1}{T}\sum_{j=1}^{T}(x(j)-\hat{\mu}_{ML})^2.$$

Note that this estimator is not unbiased, only asymptotically unbiased, and it does not coincide with the sample variance.

3.3 Information Theory

After introducing the necessary probability theoretic terminology, we now want to define the terms entropy and mutual information. These

notions are important for formulating the hypothesis of structural independence, for example, and have been heavily used in the field of computational neuroscience to interprete data in the framework of some testable theory.

Note that in physics one often distinguishes between discrete and continuous entropy; we will speak only of entropies of random vectors with densities[1]. However, one can easily see that the discrete entropy converges to the continuous one for a growing number of discrete events up to a divergent term that has to be subtracted; this is a common technique in stochastics when going from finite to infinite variables.

DEFINITION 3.18: Let $\mathbf{X}$ be an n-dimensional random vector with density $p_{\mathbf{X}}$ such that the integral

$$H(\mathbf{X}) := -\int_{\mathbb{R}^n} p_{\mathbf{X}}(\mathbf{x}) \log(p_{\mathbf{X}}(\mathbf{x}))d\mathbf{x} = -E_{\mathbf{X}}(\log p_{\mathbf{X}})$$

exists. Then $H(\mathbf{X})$ is called the (differential) *entropy* or *Boltzmann-Gibbs entropy* of $\mathbf{X}$.

Note that $H(\mathbf{X})$ is not necessarily well-defined, since the integral does not always exist.

The entropy of a uniform random variable, for example, can be calculated as follows. Let X have the density $p_X = \frac{1}{a}\chi_{[0,a]}$ for variable $a > 0$. Then the entropy of X is given by

$$H(X) = -\int_0^{\frac{1}{a}} \log \frac{1}{a} = \log a.$$

Note that the entropy is obviously invariant under translation. Its more general transformation properties are given in theorem 3.4.

THEOREM 3.4 ENTROPY TRANSFORMATION: Let $\mathbf{X}$ be a n-dimensional random variable with existing entropy $H(\mathbf{X})$ and $\mathbf{h} : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ a C^1-diffeomorphism. Then $H(\mathbf{h} \circ \mathbf{X})$ exists and

$$H(\mathbf{h} \circ \mathbf{X}) = H(\mathbf{X}) + E_{\mathbf{X}}(\log |\det \mathbf{Dh}|).$$

1 There is also the more general notion of densities in the distribution sense — this would generalize both entropy terms

THEOREM 3.5 GIBBS INEQUALITY FOR RANDOM VARIABLES: Let $\mathbf{X}$ and $\mathbf{Y}$ be two n-dimensional random vectors with densities $p_{\mathbf{X}}$ and $p_{\mathbf{Y}}$. If $p_{\mathbf{X}} \log p_{\mathbf{X}}$ and $p_{\mathbf{X}} \log p_{\mathbf{Y}}$ are integrable, then

$$H(\mathbf{X}) \leq -\int_{\mathbb{R}^n} p_{\mathbf{X}} \log p_{\mathbf{Y}}$$

and equality holds if and only if $p_{\mathbf{X}} = p_{\mathbf{Y}}$.

The entropy measures "unorder" of a random variable in the sense that it is maximal for maximal unorder:

LEMMA 3.8: Let $A \subset \mathbb{R}^n$ be measurable of the finite Lebesgue measure $\lambda(A) < \infty$. Then the maximum of the entropies of all n-dimensional random vectors $\mathbf{X}$ with density functions having support in A and for which $H(\mathbf{X})$ exists is obtained exactly at the random vector $\mathbf{X}_*$ being uniformly distributed in A.

So for the random vector $\mathbf{X}_*$ the density $p_* := \lambda(A)^{-1}\chi_A$ satisfies: All $\mathbf{X}$ as above with density $p_{\mathbf{X}} \neq p_*$ satisfy $H(\mathbf{X}) < H(\mathbf{X}_*) = \log \lambda(A)$.

Proof Let $\mathbf{X}$ be as above with density $p_{\mathbf{X}}$. The Gibbs inequality for $\mathbf{X}$ and $\mathbf{X}_*$ then shows that

$$H(\mathbf{X}) \leq -\int_{\mathbb{R}^n} p_{\mathbf{X}} \log p_* = -\log\left(\frac{1}{\lambda(A)}\right)\int_A p_{\mathbf{X}} = \log \lambda(A) = H(\mathbf{X}_*)$$

and equality holds if and only if $p_{\mathbf{X}} = p_*$. ■

For a given random vector $\mathbf{X}$ in L^2, denote $\mathbf{X}_{\text{gauss}}$ the Gaussian with mean $\mathbf{E}(\mathbf{X})$ and covariance $\text{Cov}(\mathbf{X})$. Lemma 3.9 is the non-finite generalization of the above lemma. It shows that the Gaussian has maximal entropy over all random vectors with the same first- and second-order moments.

LEMMA 3.9: Given an L^2-random vector $\mathbf{X}$, the following inequality holds:

$$H(\mathbf{X}_{\text{gauss}}) \geq H(\mathbf{X})$$

Another information theoretic function measuring distance from a Gaussian can be defined using this lemma.

DEFINITION 3.19: Let $\mathbf{X}$ be an n-dimensional random variable with existing entropy. Then

$$J(\mathbf{X}) := H(\mathbf{X}_{\text{gauss}}) - H(\mathbf{X})$$

is called the *negentropy* of $\mathbf{X}$.

According to lemma 3.9, $J(\mathbf{X}) \geq 0$, and if $\mathbf{X}$ is Gaussian, then $J(\mathbf{X}) = 0$. Note that the entropy of an n-dimensional Gaussian can be calculated as

$$H(\mathbf{X}_{\text{gauss}}) = \frac{1}{2} \log |\det \text{Cov}(\mathbf{X}_{\text{gauss}})| + \frac{n}{2}(1 + \log 2\pi),$$

so by definition

$$J(\mathbf{X}) := \frac{1}{2} \log |\det \text{Cov}(\mathbf{X})| + \frac{n}{2}(1 + \log 2\pi) - H(\mathbf{X}).$$

Using the transformational properties of the entropy, it is obvious that the negentropy is invariant under $\text{Gl}(n)$, because

$$\begin{aligned} J(\mathbf{AX}) &= H((\mathbf{AX})_{\text{gauss}}) - H(\mathbf{AX}) \\ &= H(\mathbf{X}_{\text{gauss}}) + \log \det \mathbf{A} - H(\mathbf{X}) - \log \det \mathbf{A} = J(\mathbf{X}) \end{aligned}$$

for $\mathbf{A} \in \text{Gl}(n)$.

The negentropy of a random variable can be approximated by its moments as follows:

$$J(X) = \frac{1}{12} E(X^3)^2 + \frac{1}{48} \text{kurt}(X)^2 + \ldots \tag{3.1}$$

DEFINITION 3.20: Let $\mathbf{X}$ and $\mathbf{Y}$ be two Lebesgue-continuous n-dimensional random vectors with densities $p_{\mathbf{X}}$ and $p_{\mathbf{Y}}$ such that $p_{\mathbf{X}} \log p_{\mathbf{X}}$ and $p_{\mathbf{X}} \log p_{\mathbf{Y}}$ are integrable. Then

$$K(\mathbf{X}, \mathbf{Y}) := \int_{\mathbb{R}^n} p_{\mathbf{X}} \log \frac{p_{\mathbf{X}}}{p_{\mathbf{Y}}} dx$$

is called the *Kullback-Leibler divergence* or *relative entropy* of $\mathbf{X}$ and $\mathbf{Y}$.

The Kullback-Leibler divergence measures the similarity between two random variables:

THEOREM 3.6: Let $\mathbf{X}$ and $\mathbf{Y}$ be two random variables with existing $K(\mathbf{X}, \mathbf{Y})$. Then $K(\mathbf{X}, \mathbf{Y}) \geq 0$, and equality holds if and only if $\mathbf{X}$ and $\mathbf{Y}$ have the same distribution.

DEFINITION 3.21: Let $\mathbf{X}$ be an n-dimensional random vector with density $p_{\mathbf{X}}$. If $H(X_i)$ exists, it is called the *marginal entropy* of $\mathbf{X}$ in the component i. If $H(X_i)$ exists for all i, then $\sum_{i=1}^{n} H(X_i)$ is called the *marginal entropy* of X.

THEOREM 3.7: The marginal entropy of $\mathbf{X}$ equals $H(\mathbf{X})$ if and only if $\mathbf{X}$ is independent; if not, it is greater than $H(\mathbf{X})$.

DEFINITION 3.22: Let $\mathbf{X}$ be an n-dimensional random variable with existing entropy and marginal entropy. Then

$$I(\mathbf{X}) := \left(\sum_{i=1}^{n} H(X_i) \right) - H(\mathbf{X}) = K(p_{\mathbf{X}}, \prod_{i=1}^{n} p_{X,i})$$

is called the *mutual information (MI)* of $\mathbf{X}$.

The mutual information is a scaling-invariant and permutation-invariant measure of independence of random vectors.

COROLLARY 3.1: $I(\mathbf{X}) \geq 0$ and $I(\mathbf{X}) = 0$ if and only if $\mathbf{X}$ is independent.

THEOREM 3.8 TRANSFORMATION OF MI: Let $\mathbf{X}$ be an n-dimensional random vector with existing $I(\mathbf{X})$. If $h(x_1, \ldots, x_n) = h_1(x_1) \times \ldots \times h_n(x_n)$ is a componentwise $\mathbf{C}^1-$diffeomorphism, then $I(h \circ \mathbf{X})$ exists and

$$I(h \circ \mathbf{X}) = I(\mathbf{X}).$$

Therefore, if $\mathbf{P} \in \mathrm{Gl}(n)$ is a permutation matrix, $\mathbf{L} \in \mathrm{Gl}(n)$ is a diagonal matrix (scaling matrix), and if $\mathbf{c} \in \mathbb{R}^n$, then $I(\mathbf{LPX}+\mathbf{c})$ exists and equals $I(\mathbf{X})$:

$$I(\mathbf{LPX} + \mathbf{c}) = I(\mathbf{X}).$$

Under certain conditions, independence (i.e., the zeros of mutual

information) is invariant under $\mathrm{Gl}(n)$ if and only if the matrix a scaling and a permutation.

THEOREM 3.9 INVARIANCE OF INDEPENDENCE: Let $\mathbf{X}$ be an independent n-dimensional random vector with at most one Gaussian component and existing covariance, and let $\mathbf{A} \in \mathrm{Gl}(n)$. If $\mathbf{AX}$ is again independent, then $\mathbf{A}$ is the product of a scaling and permutation matrix.

This has been shown by Comon [59]; it is a corollary of the Skitovitch-Darmois theorem, which shows a nontrivial connection between Gaussian distributions and stochastic independence. More precisely, it states that if two linear combinations of non-Gaussian independent random variables are again independent, then each original random variable can appear in only one of the two linear combinations. It has been proved independently by Darmois [62] and Skitovitch [233]; in a more accessible form, the proof can be found in [128]. A short version of this proof is presented in the appendix of [245].

Note that if $\mathbf{X}$ is allowed to have more than one Gaussian component, then obviously the above theorem cannot be correct: For example, if $\mathbf{X}$ is a two-dimensional decorrelated (hence independent) Gaussian, then according to lemma 3.6, $\mathbf{AX}$ is independent for any matrix $\mathbf{A} \in O(n)$.

3.4 Principal Component Analysis

Principal component analysis (PCA), also called Karhunen-Loève transformation, is one of the most common multivariate data analysis tools based on early works of Pearson [198]. It tries to (mostly linearly) transform given data into data in a feature space, where a few "main features" already make up most of the data; the new basis vectors are called *principal components*. We will see that this is closely connected to data whitening.

PCA decorrelates data, so it is a second-order analysis technique. ICA, as we will see, uses the much richer requirement of independence, often enforced by the mutual information; hence ICA is said to use higher-order statistics. Here, we will define only linear PCA.

Directions of Maximal Variance

Originally, PCA was formulated as a dimension reduction technique. In its simplest form, it tries to iteratively determine the most "interesting" signal component in the data, and then continue the search in the complement of this component. For any such dimension reduction or deflation technique, we need to specify how to differentiate between signal and noise in this projection. In PCA, this is achieved by considering data to be interesting if it has high variance.

Note that from here on, for simplicity we specify random vectors as lowercase letters. Given a random vector $\mathbf{x} : \Omega \to \mathbb{R}^n$ with existing covariance, we first center it and may then assume $E(\mathbf{x}) = 0$. The projection is defined as follows:

$$\begin{aligned} f : S^{n-1} \subset \mathbb{R}^n &\longrightarrow \mathbb{R} \\ \mathbf{w} &\longmapsto \operatorname{var}(\mathbf{w}^\top \mathbf{x}), \end{aligned} \tag{3.2}$$

where

$$S^{n-1} := \{\mathbf{w} \in \mathbb{R}^n | \, |\mathbf{w}| = 1\}$$

denotes the $(n-1)$-dimensional unit sphere in $\mathbb{R}^n$, and $|\mathbf{w}| = \left(\sum_i w_i^2\right)^{1/2}$ denotes the *Euclidean norm*.

Without the restriction to unit norm, maximization of f would be ill-posed, so clearly such a constraint is necessary. The first principal component of $\mathbf{x}$ is now defined as the random variable

$$y_1 := \mathbf{w}_1^\top \mathbf{x} = \sum_i (\mathbf{w}_1)_i x_i$$

generated by projecting $\mathbf{x}$ along a global maximum $\mathbf{w}_1$ of f.

The function f may, for instance, be maximized by a local algorithm, such as gradient ascent constrained on the unit sphere (e.g. by normalization of $\mathbf{w}$ after each update).

A second principal component y_2 is calculated by assuming that the projection $\mathbf{w}_2$ also maximizes f, but at the same time y_2 is decorrelated from y_1, so $E(y_1 y_2) = 0$ (note that the y_i are centered because $\mathbf{x}$ is centered). Iteratively, we can determine principal components y_i. Such an iterative projection method is called *deflation* and will be studied in more detail for a different projection in the setting of ICA (see section 4.5).

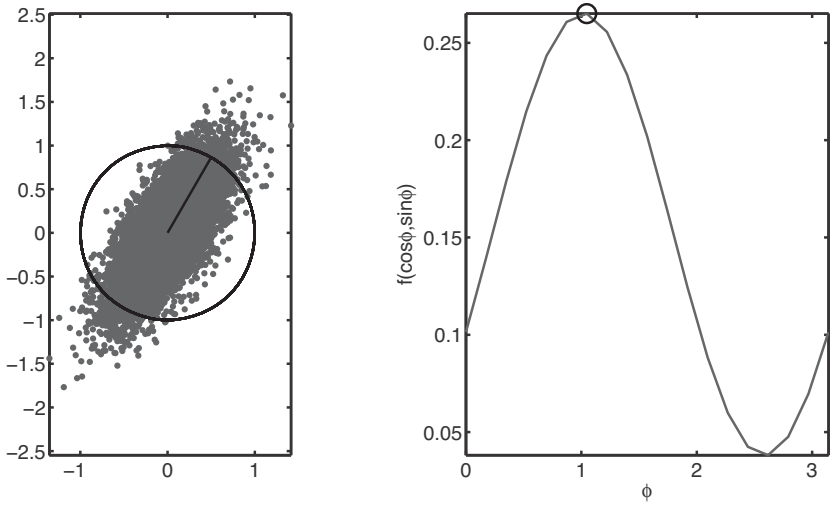


Figure 3.5
Searching for the first principal component in a two-dimensional correlated Gaussian random vector.

As an example, we consider a two-dimensional Gaussian random vector $\mathbf{x}$ centered at 0 with covariance

$$\mathrm{Cov}(\mathbf{x}) = \frac{1}{10}\begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix}.$$

In figure 3.5, we sampled 10^4 samples from $\mathbf{x}$ and numerically determined f for $\mathbf{w} = (\cos\varphi, \sin\varphi)$ with $\varphi \in [0, \pi)$. The resulting function $f(\mathbf{w})$ is shown in the figure. It is maximal at $\varphi = 1.05$ that is $\mathbf{w}_1 = (0.5, 0.86)$. This equals the eigenvector of $\mathrm{Cov}(\mathbf{x})$ corresponding to the (largest) eigenvalue 0.26, which will be explained in the next section.

Batch PCA

Here we will use the fact that the function f represents a second-order optimization problem, so that it can be solved in closed form: We rewrite

$$f(\mathbf{w}) = \mathrm{var}(\mathbf{w}^\top \mathbf{x}) = E((\mathbf{w}^\top \mathbf{x})^2) = E(\mathbf{w}\top \mathbf{x}\mathbf{x}^\top \mathbf{w}) = \mathbf{w}^\top \mathrm{Cov}(\mathbf{x})\mathbf{w}$$

This maximization can be explicitly performed by first calculating an eigenvalue decomposition of the symmetric matrix

$$\mathrm{Cov}(\mathbf{x}) = \mathbf{E}\mathbf{D}\mathbf{E}^\top$$

with orthogonal matrix $\mathbf{E}$ and diagonal matrix $\mathbf{D}$ with eigenvalues $d_{11} \geq d_{22} \geq \ldots \geq 0$.

For simplicity, we may assume pairwise different eigenvalues. Then $d_{11} > d_{22} > \ldots$. Using the decomposition, we can further rewrite

$$f(\mathbf{w}) = \mathbf{w}^\top \mathrm{Cov}(\mathbf{x})\mathbf{w} = (\mathbf{E}^\top \mathbf{w})^\top \mathbf{D}(\mathbf{E}^\top \mathbf{w}) = \sum_i d_{ii} v_i^2$$

with $\mathbf{v} := \mathbf{E}^\top \mathbf{w}$. $\mathbf{E}$ is orthogonal, so $|\mathbf{v}| = 1$, and hence $f(\mathbf{v})$ is maximal if $v_i = 0$ for $i > 1$ (i.e. if up to a sign $\mathbf{v}$ equals the first unit vector). This means that $\mathbf{w}_1 = \pm\mathbf{e}_1$ if $\mathbf{E} = (\mathbf{e}_1 \ldots \mathbf{e}_n)$, so f is maximal at the eigenvector of the covariance corresponding to the maximal eigenvalue.

In order to calculate the other principal components, we furthermore assume decorrelation with the previously calculated ones, so

$$0 = E(y_i y_j) = E(\mathbf{w}_i^\top \mathbf{x}\mathbf{x}^\top \mathbf{w}_j^\top) = \mathbf{w}_i^\top \mathrm{Cov}(\mathbf{x})\mathbf{w}_j.$$

For the second principal component, this means

$$0 = \mathbf{w}_1^\top \mathrm{Cov}(\mathbf{x})\mathbf{w}_2 = \mathbf{w}_1^\top \mathbf{E}\mathbf{D}\mathbf{E}^\top \mathbf{w}_2 = d_{11}\mathbf{e}_1^\top \mathbf{w}_2$$

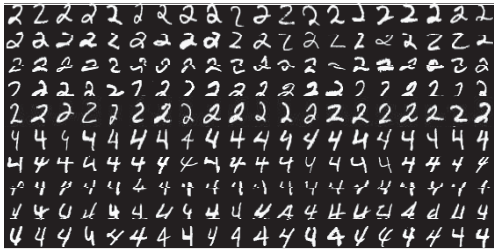
so $\mathbf{w}_2$ is orthogonal on $\mathbf{e}_1$. Hence we want to solve maximization of f in the subspace orthogonal to $\mathbf{e}_1$, which, using the same calculation as above, is clearly maximized by $\mathbf{w}_2 = \mathbf{e}_2$.

Iteratively this shows that we can determine the principal components by calculating an eigenvalue decomposition of the data covariance, and then project the data onto the eigenvectors corresponding to the first few largest eigenvalues.

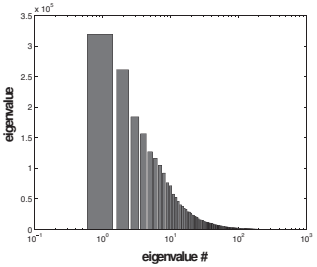
By construction the principal components are mutually decorrelated. If we further normalize their power, this corresponds to a whitening of the data. According to lemma 3.2, this is unique except for orthogonal transformation.

Example

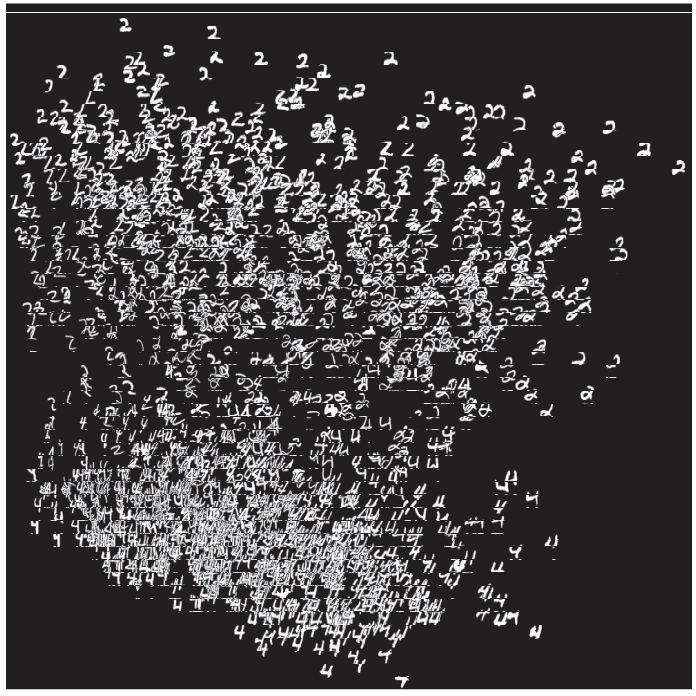
As a first example, we consider a set of handwritten digits (from the NIST image database). They consist of 1000 28x28 gray-scale images, in our case only of digits 2 and 4 (see figure 3.6(a)). We want to



(a) digits data set



(b) digits data set



(c) PCA result

Figure 3.6
NIST digits data set. In (a), we show a few samples of the 1000 28x28 gray-scale pictures of the digits 2 and 4 used in the analysis. (b) shows the eigenvalue distribution of the covariance matrix i.e. the power of each principal component, and (c) a projection onto the first two principal components. At each two-dimensional location, the corresponding picture is plotted. Clearly, the first two PCs already capture the differences between the two digits.

understand the structure of this 28^2-dimensional space given by the samples $\mathbf{x}(1), \ldots, \mathbf{x}(1000)$. For this we determine a dimension reduction onto its first few principal components.

We calculate the 784×784-dimensional covariance matrix and plot the eigenvalues in decreasing order (figure 3.6(b)). No clear cutoff can be determined from the eigenvalue distribution. However, by choosing only the first two eigenvalues (0.25% of all eigenvalues), we already capture 22.6% of the total eigenvalues:

$$\frac{d_{11} + d_{22}}{\sum_{i=1}^{784} d_{ii}} \approx 0.226.$$

And indeed, the first two eigenvalues are already sufficient to distinguish between the general shapes 2 and 4, as can be seen in the plot figure 3.6(c), where the 4s have a significantly lower second PC than the 2s.

From the previous analysis, we can deduce that the first few PCs already capture important information of the data. This implies that we might be able to represent our data set using only the first few PCs, which results in a compression method. In figure 3.7, we show the truncated PCA expansion

$$\hat{\mathbf{x}} = \sum_{i=1}^{k} \mathbf{e}_i y_i$$

when varying the truncation index k. The resulting error $E(|\hat{\mathbf{x}} - \mathbf{x}|)^2$ is precisely the sum of the remaining eigenvalues. We see that with only a few eigenvalues, we can already capture the basic digit shapes.

EXERCISES

1. Calculate the first four centered moments of a in a $[0, a]$ uniform random variable.
2. Show that the variance of the sum $\sum_i X_i$ of uncorrelated random variables X_i equals the sum of the variances $\operatorname{var} X_i$.
3. Show that the kurtosis of a Gaussian random variable vanishes, and prove that the uneven moments of a symmetric density vanish as well.

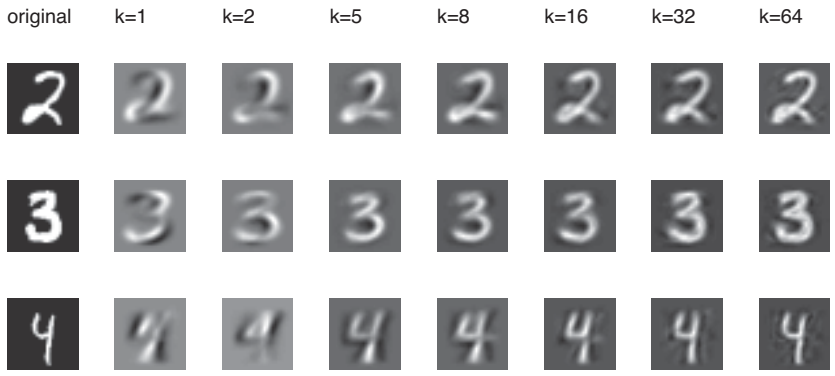


Figure 3.7
Digits 2, 3 and 4 filtered using the first few principal components.

4. *Linear least-squares fitting*. Consider the following estimation problem: assume that an n-dimensional data vector $\mathbf{x}$ follows the linear model

$$\mathbf{x} = \mathbf{A}\boldsymbol{\theta} + \mathbf{y}$$

with known $\mathbf{n} \times m$ data matrix $\mathbf{A}$, unknown parameter $\boldsymbol{\theta} \in \mathbb{R}^m$ and unknown measurement errors $\mathbf{y}$. The interesting case is if $n > m$. We determine the parameter vector $\hat{\boldsymbol{\theta}}_{LS}$ by minimizing the squared error $\sum_i y_i^2$ that is by minimizing

$$f(\boldsymbol{\theta}) = \frac{1}{2}|\mathbf{y}|^2 = \frac{1}{2}(\mathbf{x} - \mathbf{A}\boldsymbol{\theta})^\top(\mathbf{x} - \mathbf{A}\boldsymbol{\theta}).$$

a) Show that $\boldsymbol{\theta}_{LS}$ fulfills the normal equation

$$\mathbf{A}^\top\mathbf{A}\boldsymbol{\theta}_{LS} = \mathbf{A}^\top\mathbf{x}.$$

b) If $\mathbf{A}$ is full rank, we can solve this explicitly by using its pseudoinverse:

$$\boldsymbol{\theta}_{LS} = (\mathbf{A}^\top\mathbf{A})^{-1}\mathbf{A}^\top\mathbf{x}.$$

Show that if we assume that $\mathbf{y}$ is a zero-mean random vector, the least-squares estimator is unbiased.

c) Calculate the error covariance matrix $\mathrm{Cov}(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}_{LS})$ if the noise $\mathbf{y}$ is decorrelated of equal variance σ^2.

5. Compute the entropy of one-dimensional Gaussian, Laplacian and uniform distributions.

6. Show theoretically and numerically that the negentropy of a general Laplacian $p_\sigma(x) = (\sqrt{2}\sigma)^{-1} \exp(\sqrt{2}|x|/\sigma)$ is independent of its variance σ.

7. Implement a gradient ascent algorithm for optimizing the PCA cost function f from equation (3.2).

8. Generalize the gradient ascent algorithm to a multicomponent extraction algorithm by deflation. Compare this to the batch-PCA solution, using a 3-dimensional Gaussian with nontrivial covariance structure.

9. Generate two uniform, independent signals s_1, s_2 with different variances and mix these with some matrix $\mathbf{A}$: $\mathbf{x} := \mathbf{A}\mathbf{s}$. Calculate the PCA matrix $\mathbf{W}$ of $\mathbf{x}$ both analytically and numerically.

10. Prove that in exercise 9, if $\mathbf{s}$ is Gaussian, then $\mathbf{WA}$ is orthogonal. Confirm this by computer simulation and study the dependence on small sample numbers.

4 Independent Component Analysis and Blind Source Separation

Biostatistics deals with the analysis of high-dimensional data sets originating from biological or biomedical problems. An important challenge in this analysis is to identify underlying statistical patterns that facilitate the interpretation of the data set using techniques from machine learning. A possible approach is to learn a more meaningful representation of the data set, which maximizes certain statistical features. Such often linear representations have several potential applications including the decomposition of objects into "natural" components [150], redundancy and dimensionality reduction [87], biomedical data analysis, microarray data mining or enhancement, feature extraction of images in nuclear medicine, etc. [6, 34, 57, 123, 163, 177].

In this chapter, we review a representation model based on the statistical independence of the underlying sources. We show that in contrast to the correlation-based approach in PCA (see chapter 3), we are now able to uniquely identify the hidden sources.

4.1 Introduction

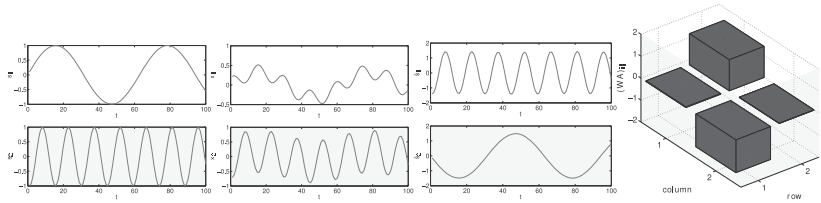
Assume the data is given by a multivariate time series $\mathbf{x}(t) \in \mathbb{R}^m$, where t indexes time, space, or some other quantity. Data analysis can be defined as finding a meaningful representation of $\mathbf{x}(t)$ that is, as $\mathbf{x}(t) = \mathbf{f}(\mathbf{s}(t))$ with unknown features $\mathbf{s}(t) \in \mathbb{R}^m$ and mixing mapping $\mathbf{f}$. Often, $\mathbf{f}$ is assumed to be linear, so we are dealing with the situation

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) \tag{4.1}$$

with a mixing matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$. Often, white noise $\mathbf{n}(t)$ is added to the model, yielding $\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) + \mathbf{n}(t)$; this can be included in $\mathbf{s}(t)$ by increasing its dimension. In equation (4.1), the analysis problem is reformulated as the search for a (possibly overcomplete) basis, in which the feature signal $\mathbf{s}(t)$ allows more insight into the data than $\mathbf{x}(t)$ does. This of course has to be specified within a statistical framework.

There are two general approaches to finding data representations or models as in equation (4.1):

- Supervised analysis: Additional information, for example in the form



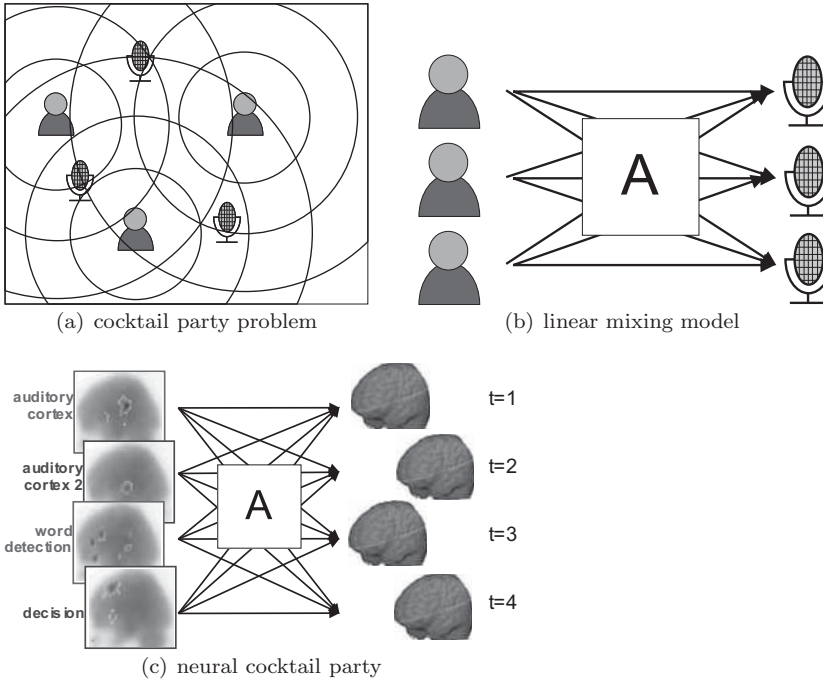
(a) sources $\mathbf{s}(t)$ (b) mixtures $\mathbf{x}(t)$ (c) recoveries (d) $\mathbf{WA}$

Figure 4.1
Two-dimensional example of ICA-based source separation. The observed mixture signal (b) is composed of two unknown source signals (a), using a linear mapping. Application of ICA (here: Hessian ICA) yields the recovered sources (c), which coincide with the original sources up to permutation and scaling: $\hat{s}_1(t) \approx 1.5 s_2(t)$ and $\hat{s}_2(t) \approx -1.5 s_1(t)$. The composition of mixing matrix $\mathbf{A}$ and separating matrix $\mathbf{W}$ equals a unit matrix (d) up to the unavoidable indeterminacies of scaling and permutation.

of input-output pairs $(\mathbf{x}(t_1), \mathbf{s}(t_1)), \ldots, (\mathbf{x}(t_T), \mathbf{s}(t_T))$. These training samples can be used for interpolation and learning of the map $\mathbf{f}$ or the basis $\mathbf{A}$ (regression). If the sources $\mathbf{s}$ are discrete, this leads to a classification problem. The resulting map $\mathbf{f}$ can then be used for prediction.

- Unsupervised models: Instead of samples, weak statistical assumptions are made on either $\mathbf{s}(t)$ or $\mathbf{f}/\mathbf{A}$. A common assumption, for example, is that the source components $s_i(t)$ are mutually independent, which results in an analysis methods called *independent component analysis (ICA)*.

Here, we will focus mostly on the second situation. The unsupervised analysis is often called *blind source separation (BSS)*, since neither features or "sources" $\mathbf{s}(t)$ nor mixing mapping $\mathbf{f}$ are assumed to be known. The field of BSS has been rather intensively studied by the community for more than a decade. Since the introduction of a neural-network-based BSS solution by Hérault and Jutten [112], various algorithms have been proposed to solve the blind source separation problem [25, 46, 59, 124, 259]. Good textbook-level introductions to the topic are given by Hyvärinen et al. [123] and Cichocki and Amari [57]. Recent research centers on generalizations and applications. The first part of this volume deals with such extended models and algorithms; some applications will be presented later.



(a) cocktail party problem (b) linear mixing model

(c) neural cocktail party

Figure 4.2
Cocktail party problem: (a) a linear superposition of the speakers is recorded at each microphone. This can be written as the mixing model $\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t)$ equation (4.1) with speaker voices $\mathbf{s}(t)$ and activity $\mathbf{x}(t)$ at the microphones (b). Possible applications lie in neuroscience: given multiple activity recordings of the human brain, the goal is to identify the underlying hidden sources that make up the total activity (c). See plate 1 for the color version of this figure.

A common model for BSS is realized by the *independent component analysis (ICA)* model [59], in which the underlying signals $\mathbf{s}(t)$ are assumed to be statistically independent. Let us first concentrate on the linear case, i.e. $\mathbf{f} = \mathbf{A}$ linear. Then we search for a decomposition $\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t)$ of the observed data set $\mathbf{x}(t) = (x_1(t), \ldots, x_n(t))^\top$ into *independent* signals $\mathbf{s}(t) = (s_1(t), \ldots, s_n(t))^\top$. For example, consider figure 4.1. The goal is to decompose two time series (b) into two source signals (a). Visually, this is a simple task—obviously the data is composed of two sinusoids with different frequency; but how to do this algorithmically? And how to formulate a feasible model?

A typical application of BSS lies in the cocktail party problem. At

a cocktail party, a set of microphones records the conversations of the guests. Each microphone records a linear superposition of the conversations, and at each microphone, a slightly different superposition is recorded, depending on the position (see figure 4.2). In the following we will see that given some rather weak assumptions on the conversations themselves, such as independence of the various speakers, it is then possible to recover the original sources and the mixing matrix (which encodes the position of the speakers) using only the signals recorded at the microphones. Note that in real-world situations the nice linear mixing situation deteriorates due to noise, convolutions, and nonlinearities.

To summarize, for a given random vector, independent component analysis (ICA) tries to find its statistically independent components. This idea can also be used to solve the blind source separation (BSS) problem which is, given only the mixtures of some underlying independent source signals, to separate the mixed signals (henceforth called sensor signals), thus recovering the original sources. Figure 4.3 shows how to apply ICA to separate three simple signals. Here neither the sources nor the mixing process is known; hence the term *blind* source separation. In contrast to correlation-based transformations such as principal component analysis (PCA), ICA renders the output signals as statistically independent as possible by evaluating higher-order statistics. The idea of ICA was first expressed by Jutten and Herault [112], [127], while the term "ICA" was later coined by Comon in [59]. However, the field became popular only with the seminal paper by Bell and Sejnowski [25] who elaborated upon the Infomax principle, which was first advocated by Linsker [157], [158]. Cardoso and Laheld [44], as well as Amari [8], later simplified the Infomax learning rule introducing by the concept of a natural gradient which accounts for the non-Euclidean Riemannian structure of the space of weight matrices. Many other ICA algorithms have been proposed, the FastICA algorithm [120] being the one of the most efficient and commonly used ones.

Recently, geometric ICA algorithms based on Kohonen-like clustering algorithms have received further attention due to their relative ease of implementation [217], [218]. They have been applied successfully to the analysis of real-world biomedical data [20] [216] and have been extended to nonlinear ICA problems, too [215].

We will now precisely define the two fundamental terms independent component analysis and blind source separation.

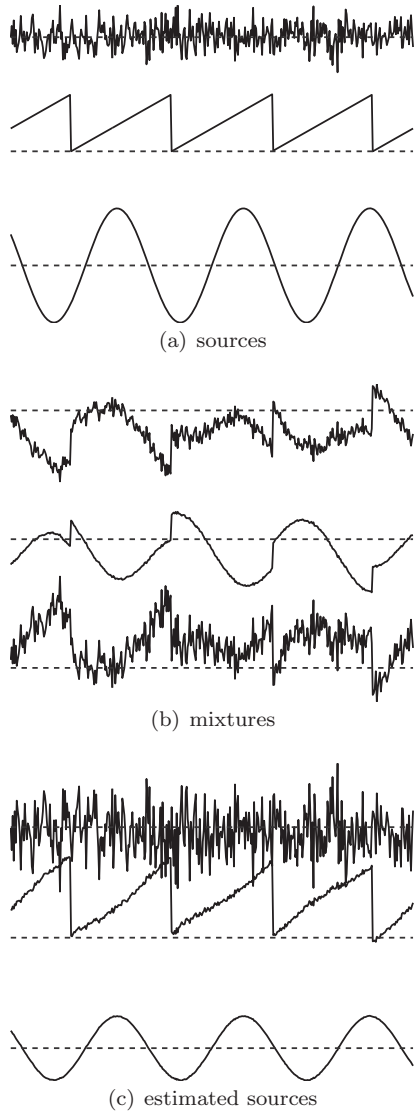


Figure 4.3
Use of ICA for performing BSS. (a) shows the three source signals, which were linearly mixed to give mixture signal as shown (b). We separated these signals using FastICA (see section 4.5). When comparing the estimated sources (c) with the original ones, we observe that they have been recovered very well. Here, we have manually chosen signs and order for visual purposes; in general the sign cannot be recovered — it is part of the ICA indeterminacies (see section 4.2).

4.2 Independent Component Analysis

In independent component analysis, a random vector $\mathbf{x} : \Omega \to \mathbb{R}^m$ called a *mixed vector* is given, and the task is to find a transformation $f(\mathbf{x})$ of $\mathbf{x}$ out of a given analysis model such that $\mathbf{x}$ is as statistically independent as possible.

Definition

First we will define ICA in its most general sense. Later we will mainly restrict ourselves to linear ICA.

DEFINITION 4.1 ICA: Let $\mathbf{x} : \Omega \to \mathbb{R}^m$ be a random vector. A measurable mapping $\mathbf{g} : \mathbb{R}^m \to \mathbb{R}^n$ is called an *independent component analysis (ICA)* of $\mathbf{x}$ if $\mathbf{y} := \mathbf{g}(\mathbf{x})$ is independent. The components Y_i of $\mathbf{y}$ are said to be the *independent components (ICs)* of $\mathbf{x}$.

We speak of *square ICA* if $m = n$. Usually, $\mathbf{g}$ is then assumed to be invertible.

Properties

It is well-known [125] that without additional restrictions to the mapping $\mathbf{g}$, ICA has too many inherent indeterminacies, meaning that there exists a very large set of ICAs which is not easily described. For this, Hyvärinen and Pajunen construct two fundamentally different (nonlinear) decompositions of an arbitrary random vector, thus showing that independence in this general case is too weak a condition.

Note that if $\mathbf{g}$ is an ICA of $\mathbf{x}$, then $I(\mathbf{g}(\mathbf{x})) = 0$. So if there is some parametric way of describing all allowed maps $\mathbf{g}$, a possible algorithm to find ICAs is simply to minimize the mutual information with respect to $\mathbf{g}$:

$$\mathbf{g}_0 = \operatorname{argmin}_{\mathbf{g}} I(\mathbf{g}(\mathbf{x})).$$

This is called *minimum mutual information (MMI)*. Of course, in practice the mutual information is very hard to calculate, so approximations of I will have to be found. Sections 4.5, 4.6, and 4.7 will present some classical ICA algorithms. Often, instead of minimizing the mutual information, the output entropy is maximized, which is kwown as the principle of *maximum entropy (ME)*. This will be discussed in more de-

tail in section 4.6. Connections between those two ideas were given by Yang and Amari in the linear case [290], where they prove that under the assumption of vanishing expectation of the sources, ME does not change the solutions of MMI except for scaling and permutation. A generalization of these ideas to nonlinear ICA problems is shown in [261] and [252].

It was mentioned that without restriction to the demixing mapping, the above problem has too many solutions. In any case, knowing the invariance of mutual information under componentwise nonlinearities (theorem 3.8), we see that if $\mathbf{g}$ is an ICA of $\mathbf{x}$ and if $\mathbf{h}$ is a componentwise diffeomorphism of $\mathbb{R}^n$, then also $\mathbf{h}(\mathbf{g})$ is an ICA of $\mathbf{x}$. Here $\mathbf{h} : \mathbb{R}^n \to \mathbb{R}^n$ is said to be *componentwise* if it can be decomposed into

$$\mathbf{h} = h_1 \times \ldots \times h_n$$

with one-dimensional mappings $h_i : \mathbb{R} \to \mathbb{R}$.

Linear ICA

Definition 4.2 Linear ICA: Let $\mathbf{x} : \Omega \to \mathbb{R}^m$ be a random vector. A full-rank matrix $\mathbf{W} \in \mathrm{Mat}(m \times n; \mathbb{R})$ is called a *linear ICA* of $\mathbf{x}$ if it is an ICA of $\mathbf{x}$ (i.e. if $\mathbf{y} := \mathbf{W}\mathbf{x}$ is independent).

Thus, in the case of square linear ICA, $\mathbf{W} \in \mathrm{Gl}(n)$. In the following, we will often omit the term "linear" if it is clear that we are speaking of linear ICA. Note that an ICA of $\mathbf{x}$ is always a PCA of $\mathbf{x}$ but not necessarily vice versa. The converse holds only if the signals are deterministic or Gaussian.

The inherent indeterminacies of ICA translate into the linear case as scaling and permutation indeterminacies, because those are the only linear mappings that are componentwise - and these mappings are invariants of independence (theorem 3.8). Scaling and permutation indeterminacy mean nothing more than that by requiring only independence, it is not possible to give an inherent order (hence permutations) and a scaling of the independent components.

One of the specialities of linear ICA, however, is that these are already all indeterminacies, as has been shown by Comon [59].

Theorem 4.1 Indeterminacies of linear ICA: Let $\mathbf{x} : \Omega \to \mathbb{R}^m$

be a random vector with existing covariance, and let $\mathbf{W}, \mathbf{V} \in \mathrm{Gl}(m)$ be two linear ICAs of $\mathbf{x}$ such that $\mathbf{Wx}$ has at most one Gaussian component. Then their inverses are equivalent i.e. there exists a permutation $\mathbf{P}$ and a scaling $\mathbf{L}$ with

$$\mathbf{PLW} = \mathbf{V}.$$

Proof This follows directly from theorem 3.9: $\mathbf{Wx}$ is independent, and by assumption $(\mathbf{VW}^{-1})(\mathbf{Wx})$, so $\mathbf{VW}^{-1}$ is the product of a scaling and permutation matrix, and therefore $\mathbf{W}^{-1}$ equals $\mathbf{V}^{-1}$ except for right-multiplication by a scaling and permutation matrix. ■

Note that this theorem also obviously holds for the case $m > n$, which can easily be shown using projections.

In order to solve linear ICA, we could again use the MMI algorithm from above,

$$\mathbf{W}_0 = \mathrm{argmin}_{\mathbf{W}}\, I(\mathbf{Wx}),$$

because elements in $\mathrm{Gl}(n) \subset \mathbb{R}^{n^2}$ are easily parameterizable. Still, the mutual information has to be approximated.

4.3 Blind Source Separation

In blind source separation, a random vector $\mathbf{x} : \Omega \to \mathbb{R}^m$ called a *mixed vector* is given; it comes from an independent random vector $\mathbf{s} : \Omega \to \mathbb{R}^n$, which will be called a *source vector*, by mixing with a *mixing function* $\boldsymbol{\mu} : \mathbb{R}^n \longrightarrow \mathbb{R}^m$ (ie. $\mathbf{x} = \boldsymbol{\mu}(\mathbf{s})$). Only the mixed vector is known, and the task is to recover $\boldsymbol{\mu}$ and then $\mathbf{s}$. If we find an ICA of $\mathbf{x}$, some kind of inversion thereof could possibly give $\boldsymbol{\mu}$.

In the square case ($m = n$), $\boldsymbol{\mu}$ is usually assumed to be invertible, so reconstruction of $\boldsymbol{\mu}$ directly gives $\mathbf{s}$ via $\mathbf{s} = \boldsymbol{\mu}^{-1}(\mathbf{x})$. This means that if we assume that the inverse of the mixing function already lies in the transformation space, then we know that the global minimum of the contrast function (usually the mutual information) has value 0, so a global maximum will indeed give us an independent random vector. Of course we cannot hope that $\boldsymbol{\mu}^{-1}$ will be found because uniqueness in this general setting cannot be achieved (section 4.2) — in contrast to the linear case, as shown in section 4.2. This will usually impose restrictions on the used model.

Definition

DEFINITION 4.3 BSS: Let $\mathbf{s} : \Omega \to \mathbb{R}^n$ be an independent random vector, and let $\boldsymbol{\mu} : \mathbb{R}^n \longrightarrow \mathbb{R}^m$ be a measurable mapping. An ICA of $\mathbf{x} := \boldsymbol{\mu}(\mathbf{s})$ is called a *BSS* of $(\mathbf{s}, \boldsymbol{\mu})$. Given a full-rank matrix $\mathbf{A} \in \mathrm{Mat}(n \times m; \mathbb{R})$, called a *mixing matrix*, a linear ICA of $\mathbf{x} := \mathbf{As}$ is called a *linear BSS* of $(\mathbf{s}, \mathbf{A})$.

Again, we speak of *square BSS* if $m = n$. In the linear case this means that the mixing matrix $\mathbf{A}$ is invertible: $\mathbf{A} \in \mathrm{Gl}(n)$.

If $m > n$, the model above is called *overdetermined* or *undercomplete*. In the case $m < n$ (i.e. in the case of less mixtures than sources) we speak of *underdetermined* or *overcomplete BSS*.

Given an independent random vector $\mathbf{s} : \Omega \to \mathbb{R}^n$ and an invertible matrix $\mathbf{A} \in \mathrm{Gl}(n)$, denote $BSS(\mathbf{s}, \mathbf{A})$ all invertible matrices $\mathbf{B} \in \mathrm{Gl}(n)$ such that $\mathbf{BAs}$ is independent (i.e. the set of all square linear BSSs of $\mathbf{As}$).

Properties

In the following we will mostly deal only with the linear case. So the goal of BSS - one of the main applications of ICA - is to find the unknown mixing matrix $\mathbf{A}$, given only the observations/mixtures $\mathbf{x}$. Using theorem 4.2, we see that in the linear case this is indeed possible, except for the usual indeterminacies scaling and permutation.

THEOREM 4.2 INDETERMINACIES OF LINEAR BSS: Let $\mathbf{s} : \Omega \to \mathbb{R}^n$ be an independent random vector with existing covariance having at most one Gaussian component, and let $\mathbf{A} \in \mathrm{Gl}(n)$. If $\mathbf{W}$ is a BSS of $(\mathbf{s}, \mathbf{A})$, then $\mathbf{W}^{-1} \sim \mathbf{A}$.

Proof This follows directly from theorem 4.2 because both $\mathbf{A}^{-1}$ and $\mathbf{W}$ are ICAs of $\mathbf{x} := \mathbf{As}$. ■

So in this case $BSS(\mathbf{s}, \mathbf{A}) = \Pi(n)\mathbf{A}^{-1}$, where $\Pi(n)$ denotes the group of products of $n \times n$ scaling and permutation matrices.

Linear BSS

In this section, we show that in linear BSS, some additional model assumptions are possible.

The general problem of square linear BSS deals with an arbitrary source random vector $\mathbf{s}$ and an arbitrary invertible matrix $\mathbf{A}$. In this section, we will show that we can make some further assumptions about those two elements.

First of all, note that in both ICA and BSS we can assume the sources to be centered, that is $\mathbf{E}(\mathbf{s}) = 0$, because the coordinate transformation

$$\begin{aligned} \mathbf{x}' &= \mathbf{x} - \mathbf{E}(\mathbf{x}) \\ \mathbf{y}' &= \mathbf{W}\mathbf{x}' = \mathbf{W}\mathbf{x} - \mathbf{W}\mathbf{E}(\mathbf{x}) \end{aligned}$$

gives centered variables that fulfill the same model requirements (independence). The same holds if we assume the BSS model and $\mathbf{x} := \mathbf{A}\mathbf{s}$.

Now denote

$$\mathbf{A} := (\mathbf{a}_1 | \dots | \mathbf{a}_n)$$

with $\mathbf{a}_i \in \mathbb{R}^n$ being the columns of $\mathbf{A}$. Scaling indeterminacy can be read as follows:

$$\begin{aligned} \mathbf{x} &= \mathbf{A}\mathbf{s} \\ &= (\mathbf{a}_1 | \dots | \mathbf{a}_n)\mathbf{s} \\ &= \sum_{i=1}^{n} \mathbf{a}_i s_i \\ &= \sum_{i=1}^{n} \left(\frac{1}{\alpha_i} \mathbf{a}_i \right) (\alpha_i s_i) \end{aligned}$$

where $\alpha_i \in \mathbb{R}, \alpha_i \neq 0$. Multiplying the sources with nonzero constants does not change their independence, so $\mathbf{A}$ can be found only up to scaling. Furthermore permuting the sum in the index i above does not change the model, so only the *set* of columns of $\mathbf{A}$ can be found, but not their order; hence the permutation indeterminacy. In order to reduce the set of solutions, some kind of *normalization* is often used. For example, in the model we could assume that $\mathrm{var}(s_i) = 1$ (i.e. that the sources have unit variances or that $|\mathbf{a}_i| = 1$). These conditions would restrict choices for the α_i to only two (*sign indeterminacy*). Permutation indeterminacy could be reduced by arbitrarily requiring some order of

the source components, for example, using some higher-order moment (like kurtosis); in practice, however, this is not very common.

We will show that we can make some further assumptions using PCA as follows. For this we assume that the sources (and hence the mixtures) have existing covariance. This is equivalent to requiring existing $\mathrm{var}(s_i)$.

Assume that $\mathrm{var}(s_i) = 1$. Then the sources are white, that is $\mathrm{Cov}(\mathbf{s}) = \mathbf{I}$. We claim that we can also assume $\mathrm{Cov}(\mathbf{x}) = \mathbf{I}$. For this, let $\mathbf{V}$ be a whitening matrix (principal component analysis, section 3.4) of $\mathbf{x}$. Then $\mathbf{z} := \mathbf{V}\mathbf{x}$ has unit covariance by definition. Calculating an ICA $\mathbf{y}' := \mathbf{W}'\mathbf{z}$ of $\mathbf{z}$ then gives an ICA of $\mathbf{x}$ by $\mathbf{W} := \mathbf{W}'\mathbf{V}$, because by construction $\mathbf{W}'\mathbf{V}\mathbf{x}$ is independent.

Furthermore, having applied PCA makes $\mathbf{A}$ and $\mathbf{W}$ orthogonal (i.e. $\mathbf{A}\mathbf{A}^\top = \mathbf{I}$): As shown above, we can assume $\mathrm{Cov}(\mathbf{s}) = \mathrm{Cov}(\mathbf{x}) = \mathbf{I}$. Then

$$\mathbf{I} = \mathrm{Cov}(\mathbf{x}) = \mathbf{A}\,\mathrm{Cov}(\mathbf{x})\mathbf{A}^\top = \mathbf{A}\mathbf{A}^\top$$

and similarly $\mathbf{W} \in O(n)$ if we require $\mathrm{Cov}(\mathbf{y}) = \mathbf{I}$. This method of *prewhitening* considerably simplifies the BSS problem. Using the well-known techniques of PCA, the number of parameters to be found has been reduced from n^2 to "only" $\frac{1}{2}n(n-1)$, which is the dimension of $O(n)$.

4.4 Uniqueness of Independent Component Analysis

Application of ICA to BSS tacitly assumes that the data follow the model equation (4.1), that is $\mathbf{x}(t)$ admits a decomposition into independent sources, and we want to find this decomposition. But neither the mixing function $\mathbf{f}$ nor the source signals $\mathbf{s}(t)$ are known, so we should expect to find *many* solutions for this problem. Indeed, the order of the sources cannot be recovered—the speakers at the cocktail party do not have numbers—so there is always an inherent permutation indeterminacy. Moreover, also the strength of each source also cannot be extracted from this model alone, because $\mathbf{f}$ and $\mathbf{s}(t)$ can interchange so-called scaling factors. In other words, by not knowing the power of each speaker at the cocktail party, we can extract only his or her speech, but not the volume—he or she could also be standing farther away from the microphones, but shouting instead of speaking in a normal voice.

One of the key questions in ICA-based source separation is whether

there are any other indeterminacies. Without fully answering this question, ICA algorithms cannot be applied to BSS, because we would not have any clue how to relate the resulting sources to the original ones. But apparently, the set of indeterminacies cannot be very large—after all, at a cocktail party we are able to distinguish the various speakers.

In 1994, Comon was able to answer this question [59] in the linear case where $\mathbf{f} = \mathbf{A}$ by reducing it to the Darmois-Skitovitch theorem [62, 233, 234]. Essentially, he showed that if the sources contain at most one Gaussian component, the indeterminacies of the above model are only scaling and permutation. This positive answer more or less made the field popular; from then on, the number of papers published in this field each year increased considerably. However, it may be argued that Comon's proof lacked two points: by using the rather difficult-to-prove old theorem by Darmois and Skitovitch, the central question *why* there are no more indeterminacies is not at all obvious. Hence not many attempts have been made to extend it to more general situations. Furthermore, no algorithm can be extracted from the proof, because it is nonconstructive.

In [246], a somewhat different approach was taken. Instead of using Comon's idea of minimal mutual information, the condition of source independence was formulated in a different way: in simple terms, a two-dimensional source vector $\mathbf{s}$ is independent if its density $p_{\mathbf{s}}$ factorizes into two one-component densities, p_{s_1} and p_{s_2}. But this is the case only if $\ln p_{\mathbf{s}}$ is the sum of one-dimensional functions, each depending on a different variable. Hence, taking the differential with respect to s_1 and then to s_2 always yields zero. In other words, the Hessian $\mathbf{H}_{\ln p_{\mathbf{s}}}$ of the logarithmic densities of the sources is diagonal—this is what we meant by $p_{\mathbf{s}}$ being a "separated function" in [246]. Using only this property, Comon's uniqueness theorem [246], can be shown without having to resort to the Darmois- Skitovitch theorem; the following is a reformulation of theorem 4.2.

THEOREM 4.3 SEPARABILITY OF LINEAR BSS: Let $\mathbf{A} \in \mathrm{Gl}(n;\mathbb{R})$ and $\mathbf{s}$ be an independent random vector. Assume that $\mathbf{s}$ has at most one Gaussian component and that the covariance of $\mathbf{s}$ exists. Then $\mathbf{As}$ is independent if and only if $\mathbf{A}$ is the product of a scaling and permutation matrix.

Instead of a multivariate random process $\mathbf{s}(t)$, the theorem is formulated for a random vector $\mathbf{s}$, which is equivalent to assuming an i.i.d. process. Moreover, the assumption of equal source (n) and mixture dimensions (m) is made, although relaxation to the undercomplete case ($1 < n < m$) is straightforward, and to the overcomplete case ($n > m > 1$) is possible [73]. The assumption of at most one Gaussian component is crucial, since independence of white, multivariate Gaussians is invariant under orthogonal transformation, abd so theorem 4.3 cannot hold in this case.

An algorithm for separation: Hessian ICA

The proof of theorem 4.3 is constructive, and the exception of the Gaussians comes into play naturally as zeros of a certain differential equation. The idea of why separation is possible becomes quite clear now. Furthermore, an algorithm can be extracted from the pattern used in the proof.

After decorrelation, we can assume that the mixing matrix $\mathbf{A}$ is orthogonal. By using the transformation properties of the Hessian matrix, we can employ the linear relationship $\mathbf{x} = \mathbf{A}\mathbf{s}$ to get

$$\mathbf{H}_{\ln p_\mathbf{x}} = \mathbf{A}^\top \mathbf{H}_{\ln p_\mathbf{s}} \mathbf{A} \tag{4.2}$$

for the Hessian of the mixtures. The key idea, as we have seen in the previous section, is that due to statistical independence, the source Hessian $\mathbf{H}_{\ln p_\mathbf{s}}$ is diagonal everywhere. Therefore equation (4.2) represents a diagonalization of the mixture Hessian, and the diagonalizer equals the mixing matrix $\mathbf{A}$. Such a diagonalization is unique if the eigenspaces of the Hessian are one-dimensional at some point, and this is precisely the case if $\mathbf{x}(t)$ contains at most one Gaussian component [246], lemma 5. Hence, the mixing matrix and the sources can be extracted algorithmically by simply diagonalizing the mixture Hessian evaluated at some point. The *Hessian ICA* algorithm consists of local Hessian diagonalization of the logarithmic density (or equivalently the easier-to-estimate characteristic function). In order to improve robustness, multiple matrices are jointly diagonalized. Applying this algorithm to the mixtures from our example from figure 4.1 yields very well recovered sources in figure 4.1(c) with a high SIR: 23 and 42 dB.

A similar algorithm has been proposed by Lin [155], but without

considering the necessary assumptions for successful algorithm application. In [246] conditions are given for when to apply this algorithm, and showed that points satisfying these conditions can indeed be found if the sources contain at most one Gaussian component ([246], lemma 5). Lin used a discrete approximation of the derivative operator to approximate the Hessian; we suggested using kernel-based density estimation, which can be directly differentiated. A similar algorithm based on Hessian diagonalization was proposed by Yeredor [291], using the character of a random vector. However, the character is complex-valued, and additional care has to be taken when applying a complex logarithm. Basically, this is well-defined only locally at nonzeros. In algorithmic terms, the character can be easily approximated by samples. Yeredor suggested joint diagonalization of the Hessian of the logarithmic character evaluated at several points in order to avoid the locality of the algorithm. Instead of joint diagonalization, we proposed to use a combined energy function based on the previously defined separator. This also takes into account global information, but does not have the drawback of being singular at zeros of the density.

Complex generalization

Comon [59] showed separability of linear real BSS using the Darmois-Skitovitch theorem (see theorem 4.3). He noted that his proof for the real case can also be extended to the complex setting. However, a complex version of the Darmois-Skitovitch theorem is needed. In [247], such a theorem was derived as a corollary of a multivariate extension of the Darmois-Skitovitch theorem, first noted by Skitovitch [234] and later shown in [93]:

THEOREM 4.4 COMPLEX S-D THEOREM: Let $s_1 = \sum_{i=1}^{n} \alpha_i x_i$ and $s_2 = \sum_{i=1}^{n} \beta_i x_i$ with $x_1, \ldots, x_n$ independent complex random variables and $\alpha_j, \beta_j \in \mathbb{C}$ for $j = 1, \ldots, n$. If s_1 and s_2 are independent, then all x_j with $\alpha_j \beta_j \neq 0$ are Gaussian.

This theorem can be used to prove separability of complex BSS and generalize this to the separation of dependent subspaces (see section 5.3). Note that a simple complex-valued uniqueness proof [248], which does not need the Darmois-Skitovitch theorem, can be derived similarly to the case of real-valued random variables from above. Recently, additional

relaxations of complex identifiability have been described [74].

4.5 ICA by Maximization of non-Gaussianity

In this and the following sections, we will present the most important "classical" ICA algorithms. We will follow the presentation in [123] in part. The following also serves as the script for a lecture presented by the author at the University of Regensburg in the summer of 2003.

First, we will develop the famous FastICA algorithm, which is among the most used current algorithms for ICA. It is based on componentwise minimization of the negentropy.

Basic Idea

Given the basic noiseless square linear BSS model

$$\mathbf{x} = \mathbf{A}\mathbf{s}$$

from section 4.3, we want to construct an ICA $\mathbf{W}$ of $\mathbf{x}$. Then ideally $\mathbf{W} = \mathbf{A}^{-1}$ (except for scaling and permutation). At first we do not want to recover all the sourcess but only one source component. We are searching among all linear combinations of the mixtures, which means we are looking for a coefficient vector $\mathbf{b} \in \mathbb{R}^n$ with

$$y = \sum_{i=1}^{n} b_i x_i = \mathbf{b}^\top \mathbf{x} = \mathbf{b}^\top \mathbf{A}\mathbf{s} =: \mathbf{q}^\top \mathbf{s}.$$

Ideally, $\mathbf{b}$ is a row of $\mathbf{A}^{-1}$, so $\mathbf{q}$ should have only one non- zero entry. But how to find $\mathbf{b}$?

The main idea of FastICA now is as follows. A heuristic usage of the central limit theorem (section 3.3) tells us that a sum of independent random variables lies closer to a Gaussian than the independent random variables themselves:

$$\text{Gaussianity}\left(\sum \text{indep. RVs}\right) > \text{Gaussianity}\,(\text{indep. RVs})$$

Of course later we will have to specify what Gaussianity means (i.e. how to measure how "Gaussian" a distribution is). So in general $y = \mathbf{q}^\top \mathbf{s}$ is more Gaussian than all source components s_i. But in ICA solutions y has the same distribution as one component s_i , hence solutions are least Gaussian.

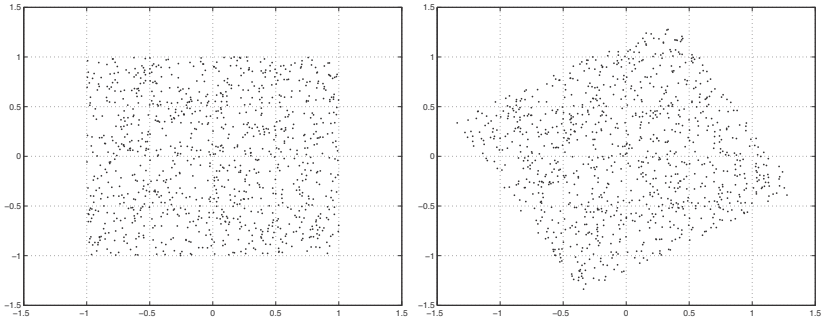


Figure 4.4
Kurtosis maximization: Source and mixture scatterplots. A two-dimensional in $[-1,1]^2$-uniform distribution with 20000 samples was chosen. The source random vector was linearly mixed by a rotation of 30 degrees. This mapping is multiplication by an orthogonal matrix, so the mixtures $\mathbf{z}$ are already white.

Algorithm: (*FastICA*) Find $\mathbf{b}$ with $\mathbf{b}^\top \mathbf{x}$ is maximal non Gaussian.

Indeed, as for PCA (section 3.4), we will see that we can restrict the search to unit-length vectors, that is to the $(n-1)$-sphere

$$S^{n-1} := \{\mathbf{x} \in \mathbb{R}^n | \, |\mathbf{x}| = 1\}.$$

And it turns out that such a cost function as above has $2n$ maxima on S^{n-1} corresponding to the solutions $\pm s_i$.

Figures 4.4 and 4.5 show an example of applying this ICA algorithm to a mixture of two uniform random variables, and figures 4.6 and 4.7 do the same for a Laplacian random vector. In both cases we see that the projections are maximally non-Gaussian in the separation directions.

Measuring non-Gaussianity using kurtosis

Given a random variable y, its kurtosis was defined as

$$\text{kurt}(y) := E(y^4) - 3(E(y^2))^2.$$

If y is Gaussian, then $E(y^4) = 3(E(y^2))^2$, so $\text{kurt}(y) = 0$. Hence, the kurtosis (or the squared kurtosis) gives a simple measure for the deviation from Gaussianity. Note that of course this measure is not definite, meaning that there also exist random variables with vanishing kurtosis that are not Gaussian.

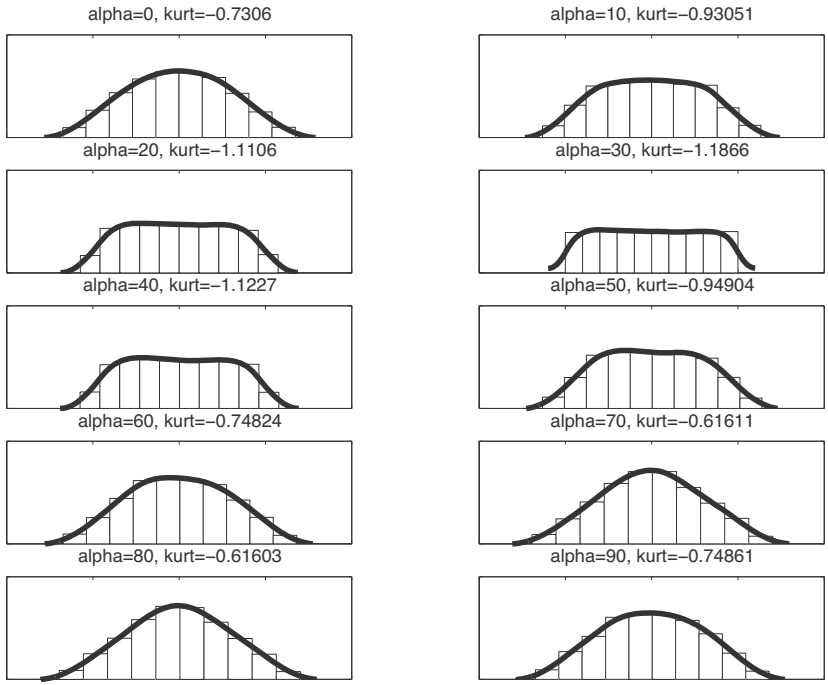


Figure 4.5
Kurtosis maximization: histograms. Plotted are the random variable $\mathbf{w}^\top \mathbf{z}$ for vectors $\mathbf{w} = (\cos(\alpha)\, \sin(\alpha))^\top$ and angle α between 0 and 90 degrees. The whitened mixtures $\mathbf{z}$ are shown in figure 4.4. Note that the projection is maximally non-Gaussian at the demixing angle 30 degrees; the absolute kurtosis is also maximal there(see also figure 4.4).

Under the assumption of unit variance, $E(y^2) = 1$, we get

$$\mathrm{kurt}(y) = E(y^4) - 3,$$

which is a sort of normalized fourth-order moment.

Let us consider a two-dimensional example first. Let

$$\mathbf{q} = \mathbf{A}^\top \mathbf{b} = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix}.$$

Then

$$y = \mathbf{b}^\top \mathbf{x} = \mathbf{q}^\top \mathbf{s} = q_1 s_1 + q_2 s_2.$$

Using linearity of kurtosis if the random variables are independent

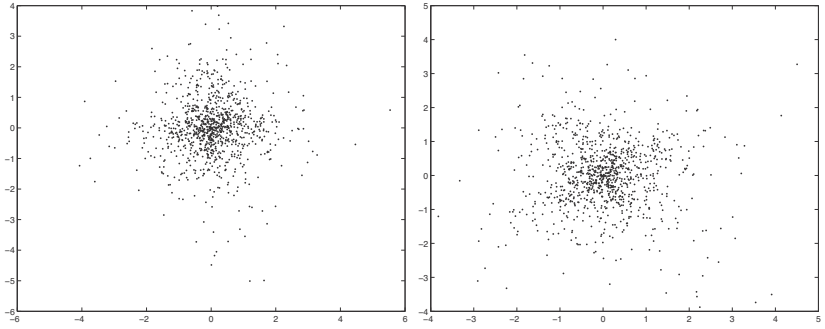


Figure 4.6
Kurtosis maximization, second example: Source and mixture scatterplots. A two-dimensional Laplacian distribution (super-Gaussian) with 20000 samples was chosen, again mixed by a rotation of 30 degrees.

(lemma 3.7), we therefore get

$$\operatorname{kurt}(y) = \operatorname{kurt}(q_1 s_1) + \operatorname{kurt}(q_2 s_2) = q_1^4 \operatorname{kurt}(s_1) + q_2^4 \operatorname{kurt}(s_2).$$

By normalization, we can assume $E(s_1^2) = E(s_2^2) = E(y^2) = 1$, so $q_1^2 + q_2^2 = 1$, which means that $\mathbf{q}$ lies on the circle $\mathbf{q} \in S^1$.

The question is: What are the maxima of

$$\begin{array}{ccc} S^1 & \longrightarrow & \mathbb{R} \\ \mathbf{q} & \mapsto & |q_1^4 \operatorname{kurt}(s_1) + q_2^4 \operatorname{kurt}(s_2)| \end{array}$$

This maximization on a smooth submanifold of $\mathbb{R}^2$ can be quickly solved using Lagrange multipliers. Using the function without absolute values, we can take derivatives and get two equations:

$$4q_i^3 \operatorname{kurt}(s_i) + 2\lambda q_i = 0$$

for $\lambda \in \mathbb{R}$, $i = 1, 2$. So

$$\lambda = -2q_1^2 \operatorname{kurt}(s_1) = -2q_2^2 \operatorname{kurt}(s_2)$$

or $q_1 = 0$ or $q_2 = 0$ (assuming that the kurtoses are not zero). Obviously only the latter two equations correspond to maxima, so from $\mathbf{q} \in S^1$ we get solutions

$$\mathbf{q} \in \{\pm\mathbf{e}_1, \pm\mathbf{e}_2\}$$

with the $\mathbf{e}_i$ denoting the unit vectors. And this is exactly what we

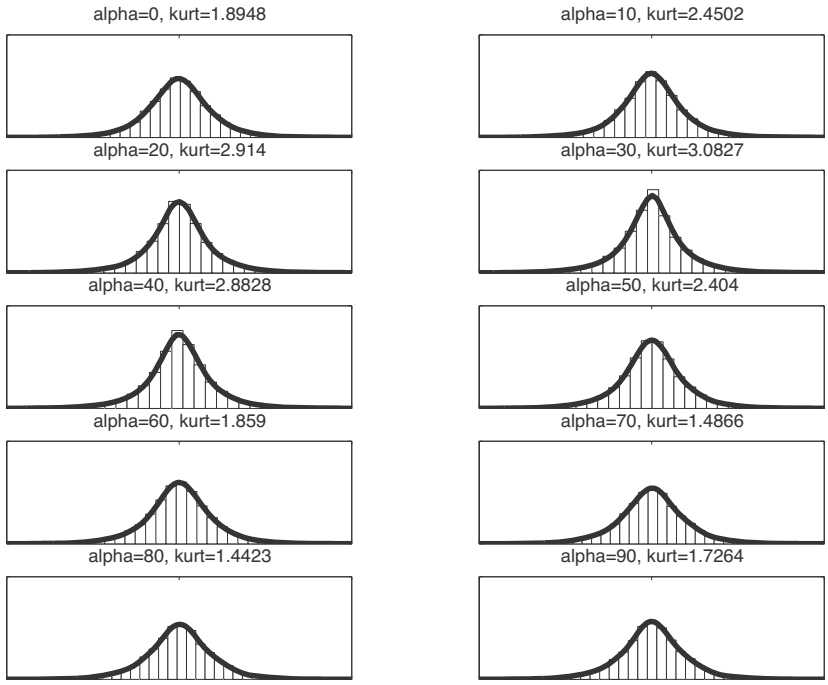


Figure 4.7
Kurtosis maximization, second example: histograms. For explanation, see figure 4.6. The data set is shown in figure 4.6. The kurtosis as function of the angle is also given in figure 4.6.

claimed: The points of maximal Gaussianity correspond to the ICA solutions.

Indeed, this can also be shown in higher dimensions (see [120]).

Algorithm

Of course, $\mathbf{s}$ is not known, so after whitening $\mathbf{z} = \mathbf{V}\mathbf{x}$ we have to search for $\mathbf{w} \in \mathbb{R}^n$ with $\mathbf{w}^\top \mathbf{z}$ maximal non-Gaussian. Because of $\mathbf{q} = (\mathbf{V}\mathbf{A})^\top \mathbf{w}$ we get

$$|\mathbf{q}|^2 = \mathbf{q}^\top \mathbf{q} = (\mathbf{w}^\top \mathbf{V}\mathbf{A})(\mathbf{A}^\top \mathbf{V}^\top \mathbf{w}) = |\mathbf{w}|^2$$

so if $\mathbf{q} \in S^{n-1}$, $\mathbf{w} \in S^{n-1}$ also. Hence, we get the following

Algorithm: (*kurtosis maximization*) Maximize $\mathbf{w} \mapsto |\operatorname{kurt}(\mathbf{w}^\top \mathbf{z})|$ on S^{n-1} after whitening.

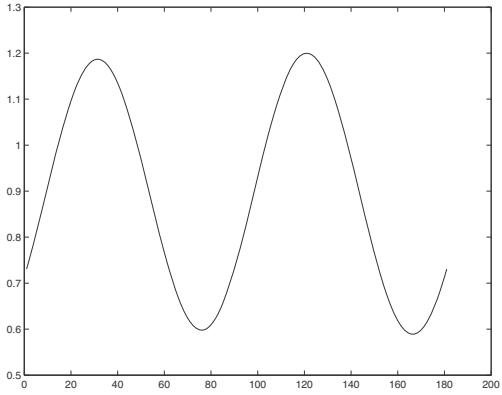


Figure 4.8
Kurtosis maximization: absolute kurtosis versus angle. The function $\alpha \mapsto |\operatorname{kurt}((\cos(\alpha)\,\sin(\alpha))\mathbf{z})|$ is plotted with the uniform $\mathbf{z}$ from figure 4.4.

We have seen that prewhitening (i.e. PCA) is essential for this algorithm — it reduces the search dimension by making the problem easily accessible. The above equation can be interpreted as finding the projection onto the line given by $\mathbf{w}$ such that $\mathbf{z}$ along this line is maximal non Gaussian.

In figures 4.8 and 4.9, the absolute kurtosis is plotted for the uniform-source example respectively the Laplacian example from above.

Gradient ascent kurtosis maximization

In practice local algorithms are often interesting. A differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ can be maximized by local updates in the direction of its gradient (which points to the direction of greatest ascent). Given a sufficiently small *learning rate* $\eta > 0$ and a starting point $\mathbf{x}(0) \in \mathbb{R}^n$, local maxima of f can be found by iterating

$$\mathbf{x}(t+1) = \mathbf{x}(t) + \eta \Delta \mathbf{x}(t)$$

with

$$\Delta \mathbf{x}(t) = (Df)(\mathbf{x}(t))^\top = \nabla f(\mathbf{x}(t)) = \frac{\partial f}{\partial \mathbf{x}}(\mathbf{x}(t))$$

being the gradient of f at $\mathbf{x}(t)$. This algorithm is called *gradient ascent*. Often, the learning rate η is chosen to be dependent on the time t, and

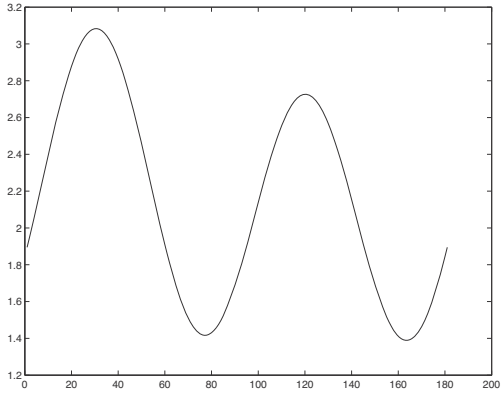


Figure 4.9
Kurtosis maximization, second example: absolute kurtosis versus angle. Again, we plot the function $\alpha \mapsto |\operatorname{kurt}((\cos(\alpha)\sin(\alpha))\mathbf{z})|$ with the super-Gaussian $\mathbf{z}$ from figure 4.6.

some suitable abort condition is defined. Furthermore, there are various ways of increasing the convergence speed of this type of algorithm.

In our case the gradient of $f(\mathbf{w}) := |\operatorname{kurt}(\mathbf{w}^\top \mathbf{z})|$ can be easily calculated as

$$\begin{aligned} \nabla|\operatorname{kurt}(\mathbf{w}^\top \mathbf{z})|(\mathbf{w}) &= \frac{\partial|\operatorname{kurt}(\mathbf{w}^\top \mathbf{z})|}{\partial \mathbf{w}} \\ &= 4\operatorname{sgn}(\operatorname{kurt}(\mathbf{w}^\top \mathbf{z}))\left(\mathbf{E}(\mathbf{z}(\mathbf{w}^\top \mathbf{z})^3) - 3|\mathbf{w}|^2\mathbf{w}\right) \quad (4.3) \end{aligned}$$

because by assumption $\operatorname{Cov}(\mathbf{z}) = \mathbf{I}$, so

$$E((\mathbf{w}^\top \mathbf{z})^2) = \mathbf{w}^\top \mathbf{E}(\mathbf{z}\mathbf{z}^\top)\mathbf{w} = |\mathbf{w}|^2.$$

By definition of the kurtosis, for white $\mathbf{z}$ we therefore get

$$\operatorname{kurt}(\mathbf{w}^\top \mathbf{z}) = E((\mathbf{w}^\top \mathbf{z})^4) - 3|\mathbf{w}|^4$$

hence

$$\frac{\partial \operatorname{kurt}(\mathbf{w}^\top \mathbf{z})}{\partial w_i} = 4E((\mathbf{w}^\top \mathbf{z})^3 Z_i) - 12|\mathbf{w}|^2 w_i$$

so

$$\frac{\partial \operatorname{kurt}(\mathbf{w}^\top \mathbf{z})}{\partial \mathbf{w}} = 4\left(\mathbf{E}((\mathbf{w}^\top \mathbf{z})^3 \mathbf{z}) - 3|\mathbf{w}|^2\mathbf{w}\right).$$

On S^1, the second part of the gradient can be neglected and we get

Algorithm: (*gradient ascent kurtosis maximization*) Choose $\eta > 0$ and $\mathbf{w}(0) \in S^{n-1}$. Then iterate

$$\begin{aligned} \Delta\mathbf{w}(t) &:= \operatorname{sgn}(\operatorname{kurt}(\mathbf{w}(t)^\top \mathbf{z}))\mathbf{E}(\mathbf{z}(\mathbf{w}(t)^\top \mathbf{z})^3) \\ \mathbf{v}(t+1) &:= \mathbf{w}(t) + \eta\Delta\mathbf{w}(t) \\ \mathbf{w}(t+1) &:= \frac{\mathbf{v}(t+1)}{|\mathbf{v}(t+1)|}. \end{aligned}$$

The third equation is needed in order for the algorithm to stay on the sphere S^{n-1}.

Fixed-point kurtosis maximization

The above local kurtosis maximization algorithm can be considerably improved by introducing the following fixed-point algorithm:

First, note that a continuously differentiable function f on S^{n-1} is extremal at $\mathbf{w}$ if its gradient $\nabla f(\mathbf{w})$ is proportional to $\mathbf{w}$ at this point. That is,

$$\mathbf{w} \propto \nabla f(\mathbf{w})$$

So here, using equation (4.5), we get

$$\mathbf{w} \propto \nabla f(\mathbf{w}) = E((\mathbf{w}^\top \mathbf{z})^3 \mathbf{z}) - 3|\mathbf{w}|^2 \mathbf{w}.$$

Algorithm: (*fixed-point kurtosis maximization*) Choose $\mathbf{w}(0) \in S^{n-1}$. Then iterate

$$\begin{aligned} \mathbf{v}(t+1) &:= \mathbf{E}((\mathbf{w}(t)^\top \mathbf{z})^3 \mathbf{z}) - 3\mathbf{w}(t) \\ \mathbf{w}(t+1) &:= \frac{\mathbf{v}(t+1)}{|\mathbf{v}(t+1)|}. \end{aligned}$$

The above iterative procedure has the separation vectors as fixed points. The advantage of using such a fixed-point algorithm lies in the facts that the convergence speed is greatly enhanced (cubic convergence in contrast to quadratic convergence of the gradient-ascent algorithm) and that other than the starting vector, the algorithm is parameter-free. For more details, refer to [124] [120].

Generalizations

Using kurtosis to measure non-Gaussianity can be problematic for non-Gaussian sources with very small or even vanishing kurtosis. In general it

often turns out that the algorithms can be improved by using a measure that takes even higher order moments into account. Such a measure can, for example, be the negentropy, defined in definition 3.19 to be

$$J(y) := H(y_{\text{gauss}}) - H(y).$$

As seen in section 3.3, the negentropy can indeed be used to measure deviation from the Gaussian. The smaller the negentropy, the "less Gaussian" the random variable.

Algorithm: (*negentropy minimization*) Minimize $\mathbf{w} \mapsto J(\mathbf{w}^\top \mathbf{z})$ on S^{n-1} after whitening.

We can assume that the random variable y has unit variance, so we get

$$J(y) := \frac{1}{2}(1 + \log 2\pi) - H(y).$$

Hence negentropy minimization equals entropy maximization.

In order to see a connection between the two Gaussianity measures kurtosis and negentropy, Taylor expansion of the negentropy can be used to get the approximation from equation (3.1):

$$J(y) = \frac{1}{12}E(y^3)^2 + \frac{1}{48}\operatorname{kurt}(y)^2 + \dots.$$

If we assume that the third-order moments of y vanish (for exampl,e for symmetric sources), we see that kurtosis maximization indeed corresponds to a first approximation of the more general negentropy minimization.

Other versions of gradient-ascent and fixed-point algorithms can now easily be developed by using more general approximations [120] of the negentropy.

Estimation of more than one component

So far we have estimated only one independent component (i.e. one row of $\mathbf{W}$). How can the above algorithm be used to estimate the whole matrix?

By prewhitening $\mathbf{W} \in O(n)$, so the rows of the whitened demixing mapping $\mathbf{W}$ are mutually orthogonal. The way to get the whole matrix $\mathbf{W}$ using the above non-Gaussianity maximization is to iteratively search components as follows.

Algorithm: (*deflation FastICA algorithm*) Perform fixed-point kurto-

sis maximization with additional Gram-Schmidt orthogonalization with respect to previously found ICs after each iteration.

This algorithm can be explicitly written down as follows:

Step 1 Set $p := 1$ (current IC).

Step 2 Choose $\mathbf{w}_p(0) \in S^{n-1}$.

Step 3 Perform a single kurtosis maximization step (here: fixed-point algorithm):

$$\mathbf{v}_p(t+1) := \mathbf{E}((\mathbf{w}_p(t)^\top \mathbf{z})^3 \mathbf{z}) - 3\mathbf{w}_p(t)$$

Step 4 Take only the part of $\mathbf{v}_p$ that is orthogonal to all previously found $\mathbf{w}_j$:

$$\mathbf{u}_p(t+1) := \mathbf{v}_p(t+1) - \sum_{j=1}^{p-1} (\mathbf{v}_p(t)\mathbf{w}_j)\mathbf{w}_j$$

Step 5 Normalize

$$\mathbf{w}_p(t+1) := \frac{\mathbf{u}_p(t+1)}{|\mathbf{u}_p(t+1)|}$$

Step 6 If the algorithm has not converged go to step 3.

Step 7 Increment p and continue with step 2 if p is less than the desired number of components.

Obviously any single-IC algorithm can be turned into a full ICA algorithm using this idea; this general principle is called the *deflation approach*. It is opposed to the *symmetric approach*, in which the single ICA update steps are performed simultaneously. The resulting matrix is then orthogonalized. Depending on the situation, the two methods perform differently. In the examples we will always use the deflation algorithm.

Example

We want to finish this section with an example application of FastICA. For this we use four speech signals, as shown in figure 4.10. They were

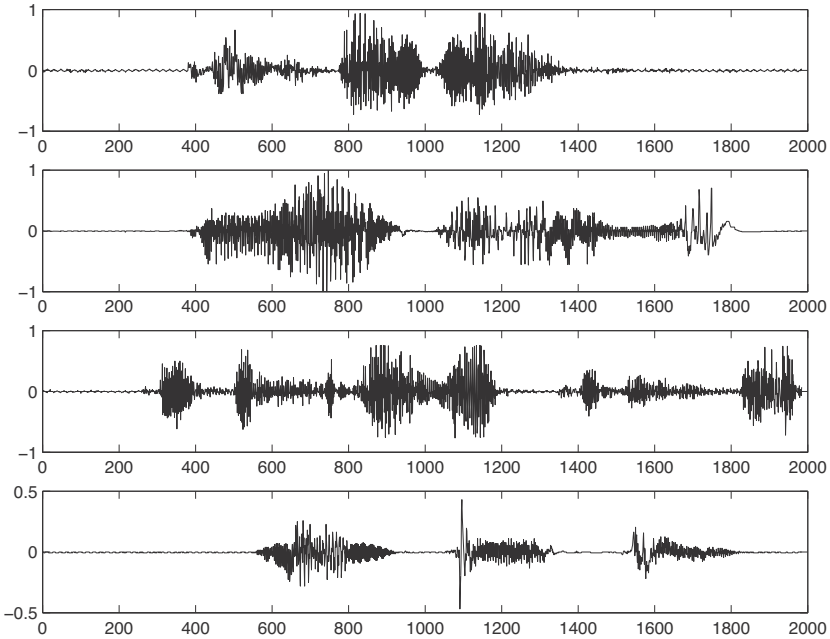


Figure 4.10
FastICA example: sources. In this figure, the four independent sources are shown — four speech signals (with time structure) were chosen. The texts of the signals are "peace and love", "hello how are you", "to be or not to be" and "one two three", all spoken by the same person except for "hello how are you". Distribution of speech signals tends to be super-Gaussian (here the kurtoses are $5.9, 4.8, 4.4$, and 14.0, respectively).

mixed by the matrix

$$\mathbf{A} := \begin{pmatrix} -0.59 & -0.60 & 0.86 & 0.05 \\ -0.60 & -0.97 & -0.068 & -0.59 \\ 0.21 & 0.49 & -0.16 & 0.34 \\ -0.46 & -0.11 & 0.69 & 0.68 \end{pmatrix}.$$

The mixtures are given in figure 4.11.

Applying the kurtosis-based FastICA algorithm with the deflation approach, we get recovered sources, as shown in figure 4.12, and a

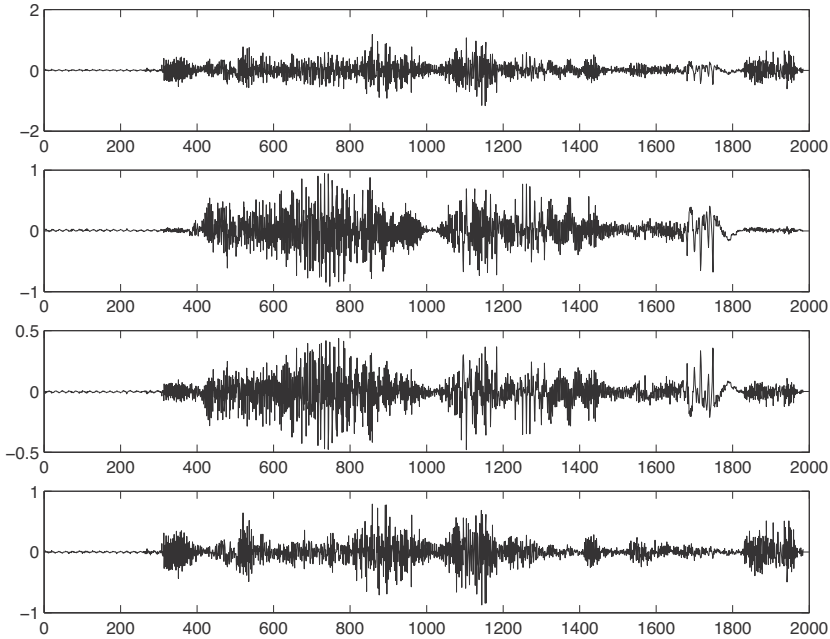


Figure 4.11
FastICA example: mixtures. The speech signals from figure 4.10 were linearly mixed by the mapping **A** given in the text. The four mixture signals are shown here.

demixing matrix

$$\mathbf{W} = \begin{pmatrix} 96 & 16 & 130 & -88 \\ 34 & 19 & 76 & -24 \\ 31 & 6 & 54 & -25 \\ 12 & -4.5 & 5.0 & -6.9 \end{pmatrix}.$$

In order to check whether the solution is good, we multiply **W** and **A**, and get

$$\mathbf{WA} = \begin{pmatrix} 0.036 & -0 & 0.0807 & -20 \\ -5.6 & 0.42 & -0.48 & 0.054 \\ 0.75 & 5.1 & -0.03 & -0.42 \\ -0.48 & 0.13 & 5.4 & 0.36 \end{pmatrix}.$$

We see that except for small perturbations this matrix is equivalent to the unit matrix (i.e. it is a scaling and a permutation.) To test this, we

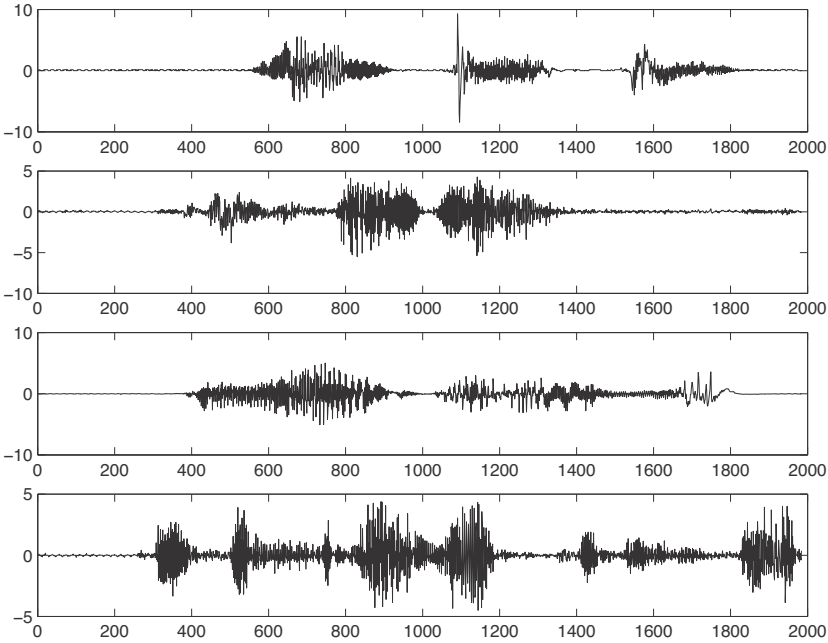


Figure 4.12
FastICA example: recovered sources. Application of kurtosis-based FastICA using the deflation approach to the mixtures from figure 4.11 gives the following recovered source signals. The first signal corresponds to the fourth source; the second, to the first source; the third, to the second source; and the fourth signal is the recovered third source. The cross-talking error between the mixture matrix $\mathbf{A}$ and the recovery matrix $\mathbf{W}$ is $E(\mathbf{A}, \mathbf{W}) = 1.1$, which is quite good in four dimensions.

can calculate the *cross-talking error*:

$$E(\mathbf{A}, \mathbf{W}) := E(\mathbf{W}^{-1}\mathbf{A}) = E(\mathbf{C}) = \sum_{i=1}^{n} \left(\sum_{j=1}^{n} \frac{|c_{ij}|}{\max_k |c_{ik}|} - 1 \right) + \sum_{j=1}^{n} \left(\sum_{i=1}^{n} \frac{|c_{ij}|}{\max_k |c_{kj}|} - 1 \right)$$

Note that $E(\mathbf{A}, \mathbf{W}) = 0$ if and only if $\mathbf{A}$ equals $\mathbf{W}^{-1}$ up to right-multiplication. We get $E(\mathbf{A}, \mathbf{W}) = 1.1$ as the measure of recovery quality, which is good in this four-dimensional example.

4.6 ICA Using Maximum-Likelihood Estimation

Maximum-likelihood estimation was introduced in section 3.2 in order to estimate the most probable parameters, given certain samples or observations in a parametric model. Here, we will use maximum likelihood estimation to estimate the mixing or separating matrix coefficients.

Likelihood of the ICA model

Consider the noiseless ICA model

$$\mathbf{x} = \mathbf{A}\mathbf{s}.$$

Let $\mathbf{B} := \mathbf{A}^{-1}$. Then, using the transformational properties of densities (theorem 3.1), we can write

$$p_{\mathbf{x}}(\mathbf{A}\mathbf{s}) = |\det \mathbf{B}| p_{\mathbf{s}}(\mathbf{s})$$

for $\mathbf{s} \in \mathbb{R}^n$. Using independence of the sources, we furthe get

$$p_{\mathbf{x}}(\mathbf{A}\mathbf{s}) = |\det \mathbf{B}| \prod_{i=1}^{n} p_i(\mathbf{s})$$

with $p_i := p_{s_i}$ the source component densities. Setting $\mathbf{x} := \mathbf{A}\mathbf{s}$ yields $\mathbf{s} = \mathbf{B}\mathbf{x}$. If we denote the rows of $\mathbf{B}$ with $\mathbf{b}_i^\top$, that is,

$$\mathbf{B} = (\mathbf{b}_1 | \ldots | \mathbf{b}_n)^\top$$

then $s_i = \mathbf{b}_i^\top \mathbf{x}$, and therefore

$$p_{\mathbf{x}}(\mathbf{x}) = |\det \mathbf{B}| \prod_{i=1}^{n} p_i(\mathbf{b}_i^\top \mathbf{x})$$

for fixed $\mathbf{A}$ (respectively) $\mathbf{B}$.

Thus according to section 3.2, we can calculate the likelihood function, given i.i.d. samples $\mathbf{x}(1), \ldots, \mathbf{x}(T)$, as

$$\begin{aligned} L(\mathbf{B}) &= \prod_{t=1}^{T} p_{\mathbf{x}|\mathbf{B}}(\mathbf{x}(t)|\mathbf{B}) \\ &= \prod_{t=1}^{T} |\det \mathbf{B}| \prod_{i=1}^{n} p_i(\mathbf{b}_i^\top \mathbf{x}(t)). \end{aligned}$$

The log likelihood then reads

$$\ln L(\mathbf{B}) = \sum_{t=1}^{T} \sum_{i=1}^{n} \ln p_i(\mathbf{b}_i^\top \mathbf{x}(t)) + T \ln |\det \mathbf{B}|$$

and, using the sample mean, we get

$$\frac{1}{T} \ln L(\mathbf{B}) = E\left(\sum_{i=1}^{n} \ln p_i(\mathbf{b}_i^\top \mathbf{x}(t)) \right) + \ln |\det \mathbf{B}|.$$

The main problem we are facing now is that in addition to the parametric model - the estimation of $\mathbf{B}$ - the unknown source densities have to be estimated; they cannot be directly described by a finite set of parameters. So we are dealing with so-called *semiparametric estimation*.

If we still want to use maximum likelihood estimation in order to find $\mathbf{B}$, two different solutions can be found, depending on prior information:

- Due to prior information, the source densities p_i are known. Then the likelihood of the whole model is described only by $L(\mathbf{B})$ because $\mathbf{B}$ is the only unknown parameter.
- If no additional information is given, the source densities p_i will have to be approximated using some sort of parameterized density families.

Indeed, the second route can be taken without too much difficulty, as is shown by theorem 4.5. It claims that for ICA estimation it is enough to locally describe each p_i by a simple binary density family (a family with only two elements) - this is quite astonishing, as the space of density families is obviously very large.

THEOREM 4.5: Let $\tilde{p}_i$ be the estimated IC densities, and assume $\tilde{p}_i > 0$. Let

$$g_i(s) := \frac{d}{ds} \ln \tilde{p}_i(s) = \frac{\tilde{p}_i'}{\tilde{p}_i}(s)$$

be the (negative) *score functions* and let $y_i := \mathbf{b}_i^\top \mathbf{x}$ be whitened. Then the maximum likelihood estimator is locally consistent if

$$E(s_i g_i(s_i) - g_i'(s_i)) > 0 \tag{4.4}$$

for $i = 1, \ldots, n$.

Here locally consistent means that locally the estimated matrix $\tilde{\mathbf{B}}$ converges to $\mathbf{B}$ in probability for $T \to \infty$.

For a proof of this theorem, see, for example theorem 9.1 from [123]

Note that equation 4.4 is invariant under small perturbations to the estimated densities $\tilde{p}_i$ because this equation depends only on the sign of $sg_i(s) - g_i'(s)$, so the local consistency of the maximum likelihood estimator is stable under small perturbations.

This idea enables us to use a simple binary density family. Define densities

$$\tilde{p}^+(s) := \frac{c_+}{\cosh^2(s)} \tag{4.5}$$

$$\tilde{p}^-(s) := \frac{c_- \cosh^2(s)}{\exp(s^2/2)} \tag{4.6}$$

with constant $c_\pm$ such that $\int \tilde{p}^\pm = 1$. Calculation shows that $c_+ = 0.5$ and $c_- \approx 0.0951$.

Taking logarithms, we note that

$$\begin{aligned} \ln \tilde{p}^+(s) &= \ln c_+ - 2 \ln \cosh(s) \\ \ln \tilde{p}^-(s) &= \ln c_- - \left(\frac{s^2}{2} - \ln \cosh^2(s) \right) \end{aligned}$$

so $\tilde{p}^+$ is super-Gaussian and $\tilde{p}^-$ is sub-Gaussian. This can also be seen in figure 4.13.

The score functions $g^\pm$ of these two densities are easily calculated as

$$g^+(s) = (-2 \ln \cosh s)' = -2 \tanh s$$

for $\tilde{p}^+$ and

$$g^-(s) = (-\frac{s^2}{2} + \ln \cosh s)' = -s + \tanh s$$

for $\tilde{p}^-$. Putting the score functions into (4.4) then yields

$$E(-s_i \tanh s_i + (1 - \tanh s_i)^2) > 0$$

and

$$E(s_i \tanh s_i - (1 - \tanh s_i)^2) > 0$$

respectively, (because $E(s_i^2) = 1$) for local consistency of the maximum likelihood estimator.

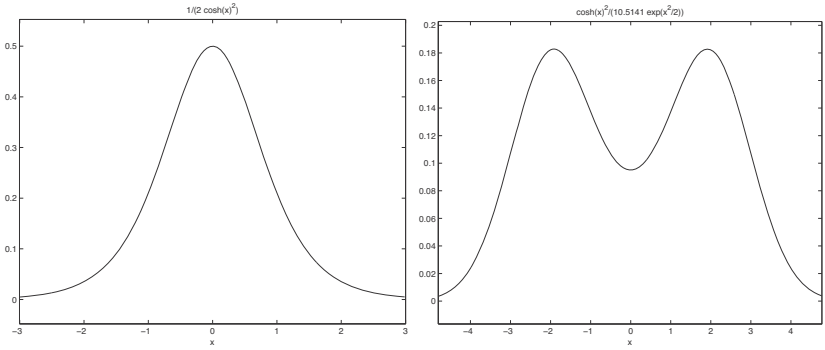


Figure 4.13
A binary density family. The left density is given by $\tilde{p}^+(s) := 0.5 \cosh^{-2}(s)$ (equation 4.5), and the right one by $\tilde{p}^-(s) := 0.0951 \cosh^2(s) \exp\left(-s^2/2\right)$ (equation 4.5).

If we assume that the source components fulfill $E(s_i \tanh s_i - (1 - \tanh s_i)^2) \neq$ (similar to the assumption $\text{kurt}(s_i) \neq 0$ in the kurtosis maximization algorithms), we have shown that either $\tilde{p}_i^+$ or $\tilde{p}_i^-$ fulfills equation (4.4). So, in order to guarantee local consistency of the estimator, for choosing the density of each source component we simply have to choose the correct $\tilde{p}_i^+$. Then theorem 4.5 guarantees that the maximum likelihood estimator with this approximated source density still gives the correct unmixing matrix $\mathbf{B}$ (as long as the mixtures have been whitened).

Note that if we put $g(s) = -s^3$ into equation (4.4), we get the condition $\text{kurt}(s_i) > 0$ for local consistency. So in some sense, the choice of $\tilde{p}_i^\pm$ corresponds to whether we minimize or maximize kurtosis, as we did in section 4.5.

Algorithms

Euclidean gradient and natural gradient

In the next section, we want to maximize the likelihood from above using gradient ascent. For this we have to calculate the gradient of a function defined on a manifold of matrices. The gradient of a function is defined as the dual of the differential of the function with respect to the scalar product. As the standard scalar product on $\mathbb{R}^n$ is $\mathbf{x}^\top \mathbf{y}$, the ordinary gradient is simply the transpose of the derivative of the

function. Here, we are interested in the gradient of a function defined on the open submanifold $\mathrm{Gl}(n)$ of $\mathbb{R}^{n^2}$. On $\mathrm{Gl}(n)$ we can either use the standard (Euclidean) scalar product (standard Riemannian metric) to get the *Euclidean gradient*

$$\nabla^{\mathrm{eukl}} f(\mathbf{W}) := \nabla f(\mathbf{W}) := (\mathbf{D} f(\mathbf{W}))^\top$$

or we can take a metric that is invariant under the group structure (multiplication) of $\mathrm{Gl}(n)$ to get the *natural gradient*

$$\nabla^{\mathrm{nat}} f(\mathbf{W}) := (\nabla^{\mathrm{eukl}} f(\mathbf{W}))\mathbf{W}^\top \mathbf{W}.$$

More details are given, for example, in chapter 2 of [244].

We also write for the Euclidean gradient

$$\frac{\partial}{\partial \mathbf{W}} f(\mathbf{W}) := \nabla^{\mathrm{eukl}} f(\mathbf{W}).$$

Lemma 4.1:

$$\frac{\partial}{\partial \mathbf{W}} \ln \det \mathbf{W} = \mathbf{W}^{-\top}$$

for $\mathbf{W} \in \mathrm{Gl}(n)$.

Proof We have to show that

$$\frac{\partial}{\partial w_{ij}} \ln \det \mathbf{W} = (\mathbf{W}^{-1})_{ji}$$

holds for $i, j = 1, \ldots, n$. Using the chain rule, we get

$$\frac{\partial}{\partial w_{ij}} \ln \det \mathbf{W} = \frac{1}{\det \mathbf{W}} \frac{\partial}{\partial w_{ij}} \det \mathbf{W}.$$

According to the Cramer rule for the inverse, we have

$$(\mathbf{W}^{-1})_{ji} = (-1)^{i+j} \frac{1}{\det \mathbf{W}} \det \mathbf{W}^{(ij)},$$

where $\mathbf{W}^{(ij)} \in \mathrm{Mat}((n-1) \times (n-1); \mathbb{R})$ denotes the matrix which comes from $\mathbf{W}$ by leaving out the i th row and the j th column. The proof is finished if we show

$$\frac{\partial}{\partial w_{ij}} \det \mathbf{W} = (-1)^{i+j} \det \mathbf{W}^{(ij)}.$$

For this, develop $\det \mathbf{W}$ by the i-th row to get

$$\det \mathbf{W} = \sum_{k=1}^{n} (-1)^{i+k} w_{ik} \det \mathbf{W}^{(ik)}.$$

Then, taking derivative by w_{ij} shows the claim. ■

LEMMA 4.2: For $\mathbf{W} \in \mathrm{Mat}(n \times n; \mathbb{R})$ and $p_i \in \mathbf{C}^\infty(\mathbb{R}, \mathbb{R})$, $i = 1, \ldots, k$

$$\frac{\partial}{\partial \mathbf{W}} \sum_{i=1}^{n} \ln p_i'(\mathbf{W}\mathbf{x})_i = \mathbf{g}(\mathbf{W}\mathbf{x})\mathbf{x}^\top,$$

for $\mathbf{x} \in \mathbb{R}^n$, where for $\mathbf{y} \in \mathbb{R}^n$,

$$\mathbf{g}(y) := \left(\frac{p_i''(y_i)}{p_i'(y_i)} \right)_{i=1}^{n} \in \mathbb{R}^n.$$

Proof We have to show that

$$\frac{\partial}{\partial w_{ij}} \sum_{k=1}^{n} \ln p_k'(\mathbf{W}\mathbf{x})_k = \frac{p_i''(y_i)}{p_i'(y_i)} x_j$$

This follows directly from the chain rule. ■

Bell-Sejnowski algorithm

With the following algorithm, Bell and Sejnowski gave one of the first easily applicable ICA algorithms [25]. It maximizes the likelihood from above by using gradient ascent.

The goal is to maximize the likelihood (or equivalently the log likelihood) of the parametric ICA model. If we assume that the source densities are differentiable, we can do this locally, using gradient ascent. The Euclidean gradient of the log likelihood can be calculated, using lemmata 4.1 and 4.2, to be

$$\frac{1}{T} \frac{\partial \ln L(\mathbf{B})}{\partial \mathbf{B}} = \mathbf{B}^{-\top} + \mathbf{E}(\mathbf{g}(\mathbf{B}\mathbf{x})\mathbf{x}^\top)$$

with the n-dimensional score function $\mathbf{g} = g_1 \times \ldots \times g_n$. Thus the local update algorithm goes as follows.

Algorithm: (*gradient ascent maximum likelihood*) Choose $\eta > 0$ and

$\mathbf{B}(0) \in \mathrm{Gl}(n)$. Then iterate for whitened mixtures $\mathbf{x}$

$$\begin{aligned} \Delta\mathbf{B}(t) &:= \mathbf{B}(t)^{-\top} + \mathbf{E}(\mathbf{g}(\mathbf{B}(t)\mathbf{x})\mathbf{x}^\top) \\ \mathbf{B}(t+1) &:= \mathbf{B}(t) + \eta\Delta\mathbf{B}(t). \end{aligned}$$

Instead of using this batch update, we can use a stochastical version by substituting expectation by samples to get

$$\Delta\mathbf{B}(t) := \mathbf{B}(t)^{-\top} + \mathbf{g}(\mathbf{B}(t)\mathbf{x}(t))\mathbf{x}(t)^\top)$$

for a sample $\mathbf{x}(t) \in \mathbb{R}^n$.

This algorithm was quite revolutionary in its early days, but it faces problems such as convergence speed and the numerically problematic matrix inversion in each update step.

Natural gradient algorithm

These problems were mostly fixed by Amari [8], who used the natural instead of the Euclidean gradient:

$$\frac{1}{T}\nabla^{\mathrm{nat}}L(\mathbf{B}) = \frac{1}{T}\left(\nabla^{\mathrm{eukl}}L(\mathbf{B})\right)\mathbf{B}^\top\mathbf{B} = (\mathbf{I} + \mathbf{E}(\mathbf{g}(\mathbf{y})\mathbf{y}^\top))\mathbf{B}$$

with $\mathbf{y} := \mathbf{B}\mathbf{x}$. Using

$$\Delta\mathbf{B}(t) := (\mathbf{I} + \mathbf{E}(\mathbf{g}(\mathbf{y})\mathbf{y}^\top))\mathbf{B}$$

gives both better convergence and numerical stability, as simulations confirm.

Score functions

Still, it is not clear which score functions are to be used. As we saw before, the score functions of the binary density family $\tilde{p}^{\pm}$ are

$$\begin{aligned} g^+(s) &= -2\tanh s \\ g^-(s) &= \tanh s - s. \end{aligned}$$

For the above two algorithms, the componentwise nonlinearities g_i are then chosen online according to equation (4.4): If

$$E(-s_i \tanh s_i + (1 - \tanh^2 s_i)) > 0$$

then we use g^+ for the i-th component, if not g^-. As said before, this is done online after prewhitening.

Infomax

Some of the first ICA algorithms, such as the Bell-Sejnowksi, algorithm were derived not from the maximum likelihood estimation principle as shown above, but from the *Infomax principle*. It states that in an input-output system, independence at the output is achieved by maximizing the *information flow* that is the mutual information between inputs and outputs. This makes sense only if some noise is introduced into the system:

$$\mathbf{x} = \mathbf{A}\mathbf{s} + \mathbf{N}$$

where $\mathbf{N}$ is an unknown white Gaussian random vector. One can show that in the noiseless limit ($|\mathbf{N}| \to 0$) Infomax corresponds to maximizing the output entropy.

Often input-output systems are modeled using neural networks. A single-layered *neural network* output function reads as

$$\mathbf{y} = \Phi(\mathbf{B}\mathbf{x}),$$

where $\Phi = \varphi_1 \times \varphi_n$ is a componentwise monotonously increasing nonlinearity and $\mathbf{B}$ is the weight matrix. In this case, using theorem 3.4, the entropy can be written as

$$H(\mathbf{y}) = H(\mathbf{x}) + E(\log|\det\frac{\partial\Phi}{\partial\mathbf{B}}|)$$

where $\mathbf{x}$ is the input random vector. Then

$$H(\mathbf{y}) = H(\mathbf{x}) + \sum_{i=1}^{n} E(\log\varphi_i(\mathbf{b}_i^\top\mathbf{x})) + \log|\det\mathbf{B}|.$$

Since $H(\mathbf{x})$ is fixed, comparing this with the logarithmic likelihood function shows that Infomax directly corresponds to maximum likelihood, if we assume that the componentwise nonlinearities are the cumulative densities of the source components (i.e. $\varphi_i' = p_i$).

4.7 Time-Structure Based ICA

So far we have considered only mixtures of random variables having no additional structure. In practice, this means that in each algorithm the order of the samples was arbitrary. Of course, in reality the signals often

have additional structure, such as time structure (e.g. speech signals) or higher-dimensional dependencies (e.g. images).

In the next section we will define what it means to have this additional time structure and how to build algorithms that specifically use this information. This means that the sample order of our signals is now relevant.

Stochastical processes

Definition 4.4 Stochastical process: A sequence of random vectors $\mathbf{x}(t), t = 1, 2, \ldots$ is called a *discrete stochastical process*. The process $(\mathbf{x}(t))_t$ is said to be i.i.d. if the $\mathbf{x}(t)$ are identically distributed and independent. A *realization* or *path* of $(\mathbf{x}(t))_t$ is given by the $\mathbb{R}^n$-sequence

$$\mathbf{x}(1)(\omega), \mathbf{x}(2)(\omega), \ldots$$

for any $\omega \in \Omega$.

The *expectation of the process* is simply the sequence of the expectations of the random vectors, and similarly for the *covariance of the process*, in particular for the variance:

$$\begin{aligned} \mathbf{E}\left((\mathbf{x}(t))_t\right) &:= (\mathbf{E}(\mathbf{x}(t)))_t \\ \mathrm{Cov}\left((\mathbf{x}(t))_t\right) &:= (\mathrm{Cov}(\mathbf{x}(t)))_t \end{aligned}$$

So far we have not yet used the time structure. Now we introduce a new term which makes sense only if this additional structure is present.

Given $\tau \in \mathbf{N}$, for $t > \tau$ we define the *autocovariance* of $(\mathbf{x}(t))_t$ to be the sequence of matrices

$$\mathbf{C}_\tau^{\mathbf{x}} := (\mathrm{Cov}(\mathbf{x}(t), \mathbf{x}(t-\tau)))_t$$

and the *autocorrelation* to be

$$\mathbf{R}_\tau^{\mathbf{x}} := (\mathrm{Cor}(\mathbf{x}(t), \mathbf{x}(t-\tau)))_t.$$

Consider the what we now call the instantaneous mixing model

$$\mathbf{x}(t) := \mathbf{A}\mathbf{s}(t)$$

for n-dimensional stochastic processes $\mathbf{s}$ and $\mathbf{x}$, and mixing matrix $\mathbf{A} \in \mathrm{Gl}(n)$. Now we do not need $\mathbf{s}(t)$ to be independent for every t,

but we require the autocovariance $\mathbf{C}^{\mathbf{s}}_{\tau}(t)$ to be diagonal for all t and τ. This second-order assumption holds for time signals which we would typically call "independent". Furthermore, note that we do not need the source distributions to be non-Gaussian.

In terms of algorithm, we will now use simple second-order statistics in the time domain instead of the higher-order statistics used before.

Without loss of generality, we can again assume $\mathbf{E}(\mathbf{x}(t)) = 0$ and $\mathbf{A} \in O(n)$. Then

$$\mathbf{C}^{\mathbf{x}}_{\tau}(t) := \mathbf{E}(\mathbf{x}(t)\mathbf{x}(t-\tau)^{\top}).$$

Time decorrelation

Let the offset $\tau \in \mathbf{N}$ be arbitrary, often $\tau = 1$. Define the *symmetrized autocovariance*

$$\bar{\mathbf{C}}^{\mathbf{x}}_{\tau} := \frac{1}{2}\left(\mathbf{C}^{\mathbf{x}}_{\tau} + (\mathbf{C}^{\mathbf{x}}_{\tau})^{\top}\right)$$

Using the usual properties of the covariance together with linearity, we get

$$\bar{\mathbf{C}}^{\mathbf{x}}_{\tau} = \mathbf{A}\bar{\mathbf{C}}^{\mathbf{s}}_{\tau}\mathbf{A}^{\top}. \tag{4.7}$$

By assumption $\bar{\mathbf{C}}^{\mathbf{s}}_{\tau}$ is diagonal, so equation 4.7 is an eigenvalue decomposition of $\bar{\mathbf{C}}^{\mathbf{x}}_{\tau}$. If we further assume that $\bar{\mathbf{C}}^{\mathbf{x}}_{\tau}$ has n different eigenvalues, then the above decomposition is uniquely determined by $\bar{\mathbf{C}}^{\mathbf{x}}_{\tau}$ except for orthogonal transformation in each eigenspace and permutation; since the eigenspaces are one-dimensionalm this means $\mathbf{A}$ is uniquely determined by equation 4.7 except for equivalence. Using this additional assumption, we have therefore shown the usual separability result, and we get an algorithm:

Algorithm: (*AMUSE*) Let $\mathbf{x}(t)$ be whitened and assume that for a given τ the matrix $\bar{\mathbf{C}}^{\mathbf{x}}_{\tau}$ has n different eigenvalues. Calculate an eigenvalue decomposition

$$\bar{\mathbf{C}}^{\mathbf{x}}_{\tau} = \mathbf{W}^{\top}\mathbf{D}\mathbf{W}$$

with $\mathbf{D}$ diagonal and $\mathbf{W} \in O(n)$. Then $\mathbf{W}$ is the separation matrix and $\mathbf{W}^{\top} \sim \mathbf{A}$.

Note that by equation 4.7, $\bar{\mathbf{C}}^{\mathbf{x}}_{\tau}$ and $\bar{\mathbf{C}}^{\mathbf{s}}_{\tau}$ have the same eigenvalues. Because $\bar{\mathbf{C}}^{\mathbf{s}}_{\tau}$ is diagonal, the eigenvalues are given by

$$E(s_i(t)s_i(t-\tau))$$

that is, the autocovariance of the component s_i. Thus the assumption reads that the source components are to have different autocovariances for given τ. In practice, if the eigenvalue decomposition is problematic, a different choice of τ often resolves this problem. However, the AMUSE algorithm is not applicable to sources with equal power spectra, meaning sources for which such a τ does not exist.

Another solution is instead of using simple diagonalization to choose more than one time lag and to do a simultaneous diagonalization of the corresponding autocovariances. Such algorithms turn out to be quite robust against noise, but of course also cannot overcome the problem of equal source power spectra.

For this, other time-based ICA algorithms also use higher-order moments in time, such as crosscumulants. A good overview of time-based ICA/BSS algorithms is given in [123].

EXERCISES

1. Define ICA and compare it with PCA.

2. After having found an ICA separating matrix of a linear noisy mixture $\mathbf{x} = \mathbf{A}\mathbf{s} + \mathbf{y}$ with white noise $\mathbf{y}$, how can the sources be estimated?

3. How can maximization of non-Gaussianity find independent components?

4. Study the central limit theorem experimentally. Consider T i.i.d. samples $x(t), t = 1, \ldots, T$ of a uniform random variable, and define

$$y := \frac{1}{T} \sum_{t=1}^{T} x(t).$$

Calculate 10^4 such realizations with corresponding y for $T = 2, 4, 10, 100$ and compare these with a Gaussian with mean 0 and variance $\operatorname{var} x$ by using histograms and kurtosis.

5. In exercise 9 from chapter 3, calculate determine also an ICA of the signals. Then compare the separated components with the principal components, visually using scatter plots and numerically by analyzing the mixing-separation-matrix products. For the ICA

algorithm, first implement the one-unit FastICA rule manually and then download and use the Matlab FastICA Package available at
`http://www.cis.hut.fi/projects/ica/fastica/code/FastICA_2.1.zip`

5 Dependent Component Analysis

In this chapter, we discuss the relaxation of the BSS model by taking into account additional structures in the data and dependencies between components. Many researchers have taken interest in this generalization, which is crucial for the application in real-world settings where such situations are to be expected.

Here, we will consider model indeterminacies as well as actual separation algorithms. For the latter, we will employ a technique that has been the basis of one of the first ICA algorithms [46], namely, *joint diagonalization (JD)*. It has become an important tool in ICA-based BSS and in BSS relying on second-order timedecorrelation [28]. Its task is, given a set of commuting symmetric $n \times n$ matrices $\mathbf{C}_i$, to find an orthogonal matrix $\mathbf{A}$ such that $\mathbf{A}^\top \mathbf{C}_i \mathbf{A}$ is diagonal for all i. This generalizes eigenvalue decomposition ($i = 1$) and the generalized eigenvalue problem ($i = 2$), in which perfect factorization is always possible.

Other extensions of the standard BSS model, such as including singular matrices [91] will be omitted from the discussion.

5.1 Algebraic BSS and Multidimensional Generalizations

Considering the BSS model from equation (4.1)—or a more general, noisy version $\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) + \mathbf{n}(t)$—the data can be separated only if we put additional conditions on the sources, such as the following:

- They are stochastically independent: $p_{\mathbf{s}}(s_1, \ldots, s_n) = p_{s_1}(s_1) \cdots p_{s_n}(s_n)$,
- Each source is sparse (i.e. it contains a certain number of zeros or has a low p-norm for small p and fixed 2-norm)
- $\mathbf{s}(t)$ is stationary, and for all τ, it has diagonal autocovariances $E(\mathbf{s}(t+\tau)\,\mathbf{s}(t)^\top)$; here zero-mean $\mathbf{s}(t)$ is assumed.

In the following, we will review BSS algorithms based on eigenvalue decomposition, JD, and generalizations. Thereby, one of the above conditions is denoted by the term *source condition*, because we do not want to specialize on a single model. The additive noise $\mathbf{n}(t)$ is modeled by a stationary, temporally and spatially white zero-mean process with variance σ^2. Moreover, we will not deal with the more complicated underdetermined case, so we assume that at most as many sources as sensors are

to be extracted (i.e. $n \leq m$).

The signals $\mathbf{x}(t)$ are observed, and the goal is to recover $\mathbf{A}$ and $\mathbf{s}(t)$. Having found $\mathbf{A}$, $\mathbf{s}(t)$ can be estimated by $\mathbf{A}^{\dagger}\mathbf{x}(t)$, which is optimal in the maximum-likelihood sense. Here † denotes the pseudo inverse of $\mathbf{A}$, which equals the inverse in the case of $m = n$. Thus the BSS task reduces to the estimation of the mixing matrix $\mathbf{A}$, and hence, the additive noise $\mathbf{n}$ is often neglected (after whitening). Note that in the following we will assume that all signals are real-valued. Extensions to the complex case are straightforward.

Approximate joint diagonalization

Many BSS algorithms employ joint diagonalization (JD) techniques on some source condition matrices to identify the mixing matrix. Given a set of symmetric matrices $\mathcal{C} := \{\mathbf{C}_1, \ldots, \mathbf{C}_K\}$, JD implies minimizing the squared sum of the off-diagonal elements of $\hat{\mathbf{A}}^{\top}\mathbf{C}_i\hat{\mathbf{A}}$, that is minimizing

$$f(\hat{\mathbf{A}}) := \sum_{i=1}^{K} \|\hat{\mathbf{A}}^{\top}\mathbf{C}_i\hat{\mathbf{A}} - \mathrm{diag}(\hat{\mathbf{A}}^{\top}\mathbf{C}_i\hat{\mathbf{A}})\|_F^2 \tag{5.1}$$

with respect to the orthogonal matrix $\hat{\mathbf{A}}$, where $\mathrm{diag}(\mathbf{C})$ produces a matrix, where all off-diagonal elements of $\mathbf{C}$ have been set to zero, and where $\|\mathbf{C}\|_F^2 := \mathrm{tr}(\mathbf{C}\mathbf{C}^{\top})$ denotes the squared Frobenius norm. A global minimum $\mathbf{A}$ of f is called a *joint diagonalizer* of $\mathcal{C}$. Such a joint diagonalizer exists if and only if all elements of $\mathcal{C}$ commute.

Algorithms for performing joint diagonalization include gradient descent on $f(\hat{\mathbf{A}})$, Jacobi-like iterative construction of $\mathbf{A}$ by Givens rotation in two coordinates [42], an extension minimizing a logarithmic version of equation (5.1) [202], an alternating optimization scheme switching between column and diagonal optimization [292], and, more recently, a linear least-squares algorithm for diagonalization [297]. The latter three algorithms can also search for non-orthogonal matrices $\mathbf{A}$. Note that in practice, minimization of the off-sums yields only an *approximate joint diagonalizer*—in the case of finite samples, the source condition matrices are estimates. Hence they only approximately share the same eigenstructure and do not fully commutate, so $f(\hat{\mathbf{A}})$ from equation (5.1) cannot be rendered zero precisely but only approximately.

Table 5.1
BSS algorithms based on joint diagonalization (centered sources are assumed)

algorithm	source model	condition matrices	optimization algorithm
FOBI [45]	independent i.i.d. sources	contracted quadricovariance matrix with $\mathbf{E}_{ij} = \mathbf{I}$	EVD after PCA (GEVD)
JADE [46]	independent i.i.d. sources	contracted quadricovariance matrices	orthogonal JD after PCA
eJADE [180]	independent i.i.d. sources	arbitrary-order cumulant matrices	orthogonal JD after PCA
HessianICA [246, 291]	independent i.i.d. sources	multiple Hessians $\mathbf{H}_{\log \hat{\mathbf{x}}}(\mathbf{x}^{(i)})$ or $\mathbf{H}_{\log p_{\mathbf{x}}}(\mathbf{x}^{(i)})$	orthogonal JD after PCA
AMUSE [178, 270]	wide-sense stationary $\mathbf{s}(t)$ with diagonal autocovariances	single autocovariance matrix $E(\mathbf{x}(t+\tau)\mathbf{x}(t)^\top)$	EVD after PCA (GEVD)
SOBI [28], TDSEP [298]	wide-sense stationary $\mathbf{s}(t)$ with diagonal autocovariances	multiple autocovariance matrices	orthogonal JD after PCA
mdAMUSE [262]	$\mathbf{s}(t_1, \ldots, t_M)$ with diagonal autocovariances	single multidimensional autocovariance matrix (5.3)	EVD after PCA (GEVD)
mdSOBI [228, 262]	$\mathbf{s}(t_1, \ldots, t_M)$ with diagonal autocovariances	multidimensional autocovariance matrices (5.3)	orthogonal JD after PCA
JADE$_{TD}$ [182]	independent $\mathbf{s}(t)$ with diagonal autocovariances	cumulant and autocovariance matrices	orthogonal JD after PCA

Source conditions

In order to get a well-defined source separation model, assumptions about the sources such as stochastic independence have to be formulated. In practice, the conditions are preferably given in terms of roots of some cost function that can easily be estimated. Here, we summarize some of the source conditions used in the literature; they are defined by a criterion specifying the diagonality of a set of matrices $\mathbf{C}(.) := \{\mathbf{C}_1(.), \ldots, \mathbf{C}_K(.)\}$, which can be estimated from the data. We require only that

$$\mathbf{C}_i(\mathbf{W}\mathbf{x}) = \mathbf{W}\mathbf{C}_i(\mathbf{x})\mathbf{W}^\top \tag{5.2}$$

for some matrix $\mathbf{W}$. Note that using the substitution $\bar{\mathbf{C}}_i(\mathbf{x}) := \mathbf{C}_i(\mathbf{x}) + \mathbf{C}_i(\mathbf{x})^\top$, we can assume $\mathbf{C}_i(\mathbf{x})$ to be symmetric. The actual source

Table 5.2
BSS algorithms based on joint diagonalization (continued)

algorithm	source model	condition matrices	optimization algorithm
SONS [52]	non-stationary $\mathbf{s}(t)$ with diagonal (auto-)covariances	(auto-)covariance matrices of windowed signals	orthogonal JD after PCA
ACDC [292], LSDIAG [297]	independent or auto-decorrelated $\mathbf{s}(t)$	covariance matrices and cumulant/autocovariance matrices	non-orthogonal JD
block-Gaussian likelihood [203]	block-Gaussian non-stationary $\mathbf{s}(t)$	(auto-)covariance matrices of windowed signals	non-orthogonal JD
TFS [27]	$\mathbf{s}(t)$ from Cohen's time-frequency distributions [58]	spatial time-frequency distribution matrices	orthogonal JD after PCA
FRT-based BSS [129]	non-stationary $\mathbf{s}(t)$ with diagonal block-spectra	autocovariance of FRT-transformed windowed signal	(non-)orthogonal JD
ACMA [273]	$\mathbf{s}(t)$ is of constant modulus (CM)	independent vectors in $\ker \hat{\mathbf{P}}$ of model-matrix $\hat{\mathbf{P}}$	generalized Schur QZ-decomp.
stBSS [254]	spatiotemporal sources $\mathbf{s} := \mathbf{s}(r, t)$	any of the above conditions for both $\mathbf{x}$ and $\mathbf{x}^\top$	non-orthogonal JD
group BSS [249]	group-dependent sources $\mathbf{s}(t)$	any of the above conditions	block orthogonal JD after PCA

model is then defined by requiring the sources to fulfill $\mathbf{C}_i(\mathbf{s}) = 0$ for all $i = 1, \ldots, K$. In table 5.1, we review some commonly used source conditions for an m-dimensional centered random vector $\mathbf{x}$ and a multivariate random process $\mathbf{x}(t)$.

Searching for sources $\mathbf{s} := \mathbf{W}\mathbf{x}$ fulfilling the source model requires finding matrices $\mathbf{W}$ such that $\mathbf{C}_i(\mathbf{W}\mathbf{x})$ is diagonal for all i. Depending on the algorithm, whitening by PCA is performed as preprocessing to allow for a reduced search on the orthogonal group $\mathbf{W} \in O(n)$. This is equivalent to setting all source second-order statistics to $\mathbf{I}$, and then searching only for rotations. In the case of $K = 1$, the search can be performed by eigenvalue decomposition of $\mathbf{C}_1(\tilde{\mathbf{x}})$ of the source condition of the whitened mixtures $\tilde{\mathbf{x}}$; this is equivalent to solving the *generalized eigenvalue decomposition (GEVD)* problem for the matrix pencil $(E(\mathbf{x}\mathbf{x}^\top), \mathbf{C}_1(\tilde{\mathbf{x}}))$. Usually, using more than one condition matrix

increases the robustness of the proposed algorithm, and in these cases the algorithm performs orthogonal JD of $\mathcal{C} := \{\mathbf{C}_i(\tilde{\mathbf{x}})\}$, for instance by a Jacobi-type algorithm [42].

In contrast to this *hard-whitening* technique, *soft-whitening* tries to avoid a bias toward second-order statistics and uses a nonorthogonal joint diagonalization algorithm [202, 292, 297] by jointly diagonalizing the source conditions $\mathbf{C}_i(\mathbf{x})$ together with the mixture covariance matrix $E(\mathbf{x}\mathbf{x}^\top)$. Then possible estimation errors in the second-order part do not influence the total error to a disproportional degree.

Depending on the source conditions, various algorithms have been proposed in the literature. Table 5.1 gives an overview of the algorithms together with the references, the source model, the condition matrices, and the optimization algorithm. For more details and references, see [258].

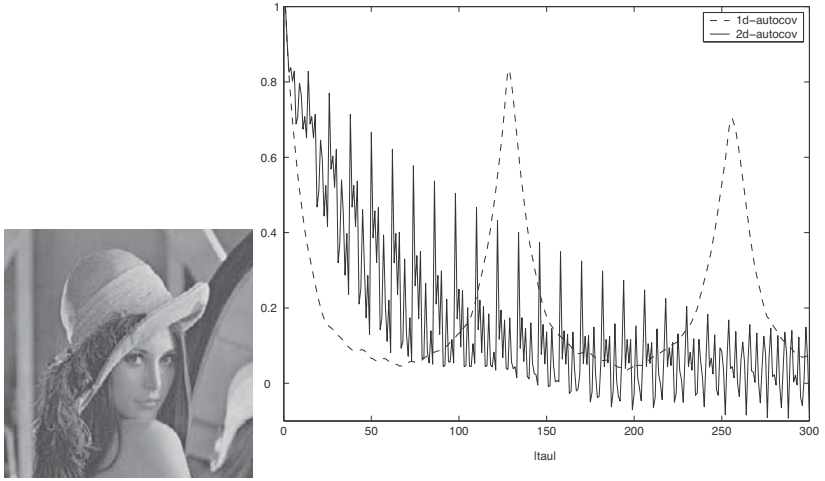
Multidimensional autodecorrelation

In [262], we considered BSS algorithms based on time decorrelation and the resulting source condition. Corresponding JD-based algorithms include AMUSE [270] and extensions such as SOBI [28] and TDSEP [298]. They rely on the fact that the data sets have non-trivial autocorrelations. We extended them to data sets having more than one direction in the parameterization such as images. For this, we replaced one-dimensional autocovariances with multidimensional autocovariances defined by

$$\mathbf{C}_{\tau_1,\ldots,\tau_M}(\mathbf{s}) := \mathbf{E}\left(\mathbf{s}(z_1+\tau_1,\ldots,z_M+\tau_M)\mathbf{s}(z_1,\ldots,z_M)^\top\right) \qquad (5.3)$$

where the $\mathbf{s}$ is centered and the expectation is taken over $(z_1,\ldots,z_M)$. $\mathbf{C}_{\tau_1,\ldots,\tau_M}(\mathbf{s})$ can be estimated given equidistant samples by replacing random variables with sample values and expectations with sums as usual.

A typical example of nontrivial multidimensional autocovariances is a source data set in which each component s_i represents an image of size $h \times w$. Then the data is of dimension $M = 2$, and samples of $\mathbf{s}$ are given at indices $z_1 = 1,\ldots,h$, $z_2 = 1,\ldots,w$. Classically, $\mathbf{s}(z_1, z_2)$ is transformed to $\mathbf{s}(t)$ by fixing a mapping from the two-dimensional parameter set to the one-dimensional time parameterization of $\mathbf{s}(t)$, for example, by concatenating columns or rows in the case of a finite number of samples (vectorization). If the time structure of $\mathbf{s}(t)$ is not used, as in all classical



(a) analyzed image (b) autocorrelation (1d/2d)

Figure 5.1
One- and two-dimensional autocovariance coefficients (b) of the gray-scale 128×128 Lena image (a) after normalization to variance 1. Clearly, using local structure in both directions (2-D autocov) guarantees that for small τ, higher powers of the autocorrelations are present than by rearranging the data into a vector (1-D autocov), thereby losing information about the second dimension.

ICA algorithms in which i.i.d. samples are assumed, this choice does not influence the result. However, in time-structure-based algorithms such as AMUSE and SOBI, results can vary greatly, depending on the choice of this mapping.

The advantage of using multidimensional autocovariances lies in the fact that now the multidimensional structure of the data set can be used more explicitly. For example, if row concatenation is used to construct $\mathbf{s}(t)$ from the images, horizontal lines in the image will make only trivial contributions to the autocovariances. Figure 5.1 shows the one- and two-dimensional autocovariance of the Lena image for varying τ (respectively (τ_1, τ_2)) after normalization of the image to variance 1. Clearly, the two-dimensional autocovariance does not decay as quickly with increasing radius as the one-dimensional covariance. Only at multiples of the image height is the one-dimensional autocovariance significantly high (i.e. captures image structure).

More details, as well as extended simulations and examples, are given in [228, 230, 262].

5.2 Spatiotemporal BSS

Real-world data sets such as recordings from functional magnetic resonance imaging often possess both spatial and temporal structure. In [253], we propose an algorithm including such spatiotemporal information into the analysis, and reduce the problem to the joint approximate diagonalization of a set of autocorrelation matrices.

Spatiotemporal BSS, in contrast to the more common spatial or temporal BSS, tries to achieve both spatial and temporal separation by optimizing a joint energy function. First proposed by Stone et al. [241], it is a promising method which has potential applications in areas where data contains an inherent spatiotemporal structure, such as data from biomedicine or geophysics (including oceanography and climate dynamics). Stone's algorithm is based on the Infomax ICA algorithm [25], which due to its online nature, involves some rather intricate choices of parameters, specifically in the spatiotemporal version, where online updates are being performed in both space and time. Commonly, the spatiotemporal data sets are recorded in advance, so we can easily replace spatiotemporal online learning with batch optimization. This has the advantage of greatly reducing the number of parameters in the system, and leads to more stable optimization algorithms. Stone's approach can be extended by generalizing the time-decorrelation algorithms to the spatiotemporal case, thereby allowing us to use the inherent spatiotemporal structures of the data [253].

For this, we considered data sets $x(\mathbf{r}, t)$ depending on two indices $\mathbf{r}$ and t, where $\mathbf{r} \in \mathbb{R}^n$ can be any multidimensional (spatial) index and t indexes the time axis. In order to be able to use matrixnotation, we contracted the spatial multidimensional index $\mathbf{r}$ into a one-dimensional index r by row concatenation. Then the data set $x(r, t) =: x_{rt}$ can be represented by a data matrix $\mathbf{x}$ of dimension ${}^{\mathbf{s}}m \times {}^{\mathbf{t}}m$, where the superscripts ${}^{\mathbf{s}}(.)$ and ${}^{\mathbf{t}}(.)$ denote spatial and temporal variables, respectively.

Temporal BSS implies the matrix factorization $\mathbf{x} = {}^{\mathbf{t}}\mathbf{A}{}^{\mathbf{t}}\mathbf{s}$, whereas spatial BSS implies the factorization $\mathbf{x}^\top = {}^{\mathbf{s}}\mathbf{A}{}^{\mathbf{s}}\mathbf{s}$ or equivalently $\mathbf{x} = {}^{\mathbf{s}}\mathbf{s}^\top {}^{\mathbf{s}}\mathbf{A}^\top$. Hence $\mathbf{x} = {}^{\mathbf{t}}\mathbf{A}{}^{\mathbf{t}}\mathbf{s} = {}^{\mathbf{s}}\mathbf{s}^\top {}^{\mathbf{s}}\mathbf{A}^\top$. Thus both source separation mod-

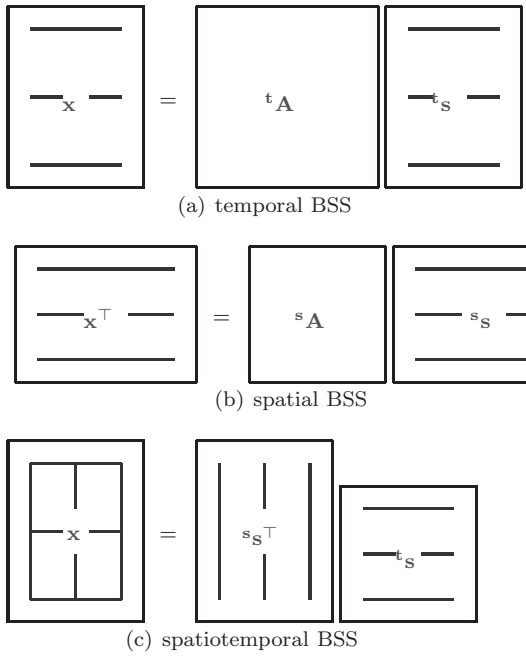


Figure 5.2
Temporal, spatial and spatiotemporal BSS models. The lines in the matrices ${}^*\mathbf{S}$ indicate the sample direction. Source conditions apply between adjacent such lines.

els can be interpreted as matrix factorization problems; in the temporal case, restrictions such as diagonal autocorrelations are determined by the second factor, and in the spatial case, by the first one. In order to achieve a spatiotemporal model, we required these conditions from both factors at the same time. Therefore, the *spatiotemporal BSS* model can be derived from the above as the factorization problem

$$\mathbf{x} = {}^{\mathbf{s}}\mathbf{s}^{\top}{}^{\mathbf{t}}\mathbf{s} \tag{5.4}$$

with spatial source matrix ${}^{\mathbf{s}}\mathbf{s}$ and temporal source matrix ${}^{\mathbf{t}}\mathbf{s}$, which both have (multidimensional) autocorrelations that are as diagonal as possible. The three models are illustrated in figure 5.2.

Concerning conditions for the sources, we interpreted $\mathbf{C}_i(\mathbf{x}) := \mathbf{C}_i({}^{\mathbf{t}}\mathbf{x}(t))$ as the i-th temporal autocovariance matrix, whereas $\mathbf{C}_i(\mathbf{x}^{\top}) := \mathbf{C}_i({}^{\mathbf{s}}\mathbf{x}(r))$ denoted the corresponding spatial autocovariance matrix.

Application of the spatiotemporal mixing model from equation (5.4) together with the transformation properties equation (5.2) of the source conditions yields

$$\mathbf{C}_i({}^{\mathbf{t}}\mathbf{s}) = {}^{\mathbf{s}}\mathbf{s}^{\dagger\top}\mathbf{C}_i(\mathbf{x}){}^{\mathbf{s}}\mathbf{s}^{\dagger} \quad \text{and} \quad \mathbf{C}_i({}^{\mathbf{s}}\mathbf{s}) = {}^{\mathbf{t}}\mathbf{s}^{\dagger\top}\mathbf{C}_i(\mathbf{x}^{\top}){}^{\mathbf{t}}\mathbf{s}^{\dagger} \tag{5.5}$$

because ${}^{*}m \geq n$ and hence ${}^{*}\mathbf{s}{}^{*}\mathbf{s}^{\dagger} = \mathbf{I}$. By assumption the matrices $\mathbf{C}_i({}^{*}\mathbf{s})$ are as diagonal as possible. In order to separate the data, we had to find diagonalizers for both $\mathbf{C}_i(\mathbf{x})$ and $\mathbf{C}_i(\mathbf{x}^{\top})$ such that they satisfy the spatiotemporal model equation (5.4). As the matrices derived from $\mathbf{X}$ had to be diagonalized in terms of both columns and rows, we denoted this by *double-sided approximate joint diagonalization*.

This process can be reduced to joint diagonalization [253, 254]. In order to get robust estimates of the source conditions, dimension reduction was essential. For this we considered the singular value decomposition $\mathbf{x}$, and formulated the algorithm in terms of the pseudo-orthogonal components of $\mathbf{X}$. Of course, instead of using autocovariance matrices, other source conditions $\mathbf{C}_i(.)$ from table 5.1 can be employed in order to adapt to the separation problem at hand.

We present an application of the spatiotemporal BSS algorithm to fMRI data using multidimensional autocovariances in chapter 8.

5.3 Independent Subspace Analysis

Another extension of the simple source separation model lies in extracting groups of sources that are independent of each other, but not within the group. Thus, multidimensional independent component analysis, or *independent subspace analysis (ISA)*, is the task of transforming a multivariate observed sensor signal such that groups of the transformed signal components are mutually independent—however, dependencies within the groups are still allowed. This allows for weakening the sometimes too strict assumption of independence in ICA, and has potential applications in fields such as ECG, fMRI analysis, and convolutive ICA.

Recently we were able to calculate the indeterminacies of group ICA for known and unknown group structures, which finally enabled us to guarantee successful application of group ICA to BSS problems. Here, we will review the identifiability result as well as the resulting algorithm for separating signals into groups of dependent signals. As before, the

algorithm is based on joint (block) diagonalization of sets of matrices generated using one or multiple source conditions.

Generalizations of the ICA model that are to include dependencies of multiple one-dimensional components have been studied for quite some time. ISA in the terminology of multidimensional ICA was first introduced by Cardoso [43] using geometrical motivations. His model, as well as the related but independently proposed factorization of multivariate function classes [155] are quite general. However, no identifiability results were presented, and applicability to an arbitrary random vector was unclear. Later, in the special case of equal group sizes k (in the following denoted as k-ISA), uniqueness results have been extended from the ICA theory [247]. Algorithmic enhancements in this setting have studied been recently [207]. Similar to [43], Akaho et al. [3] also proposed to employ a multidimensional-component, maximum-likelihood algorithm, but in the slightly different context of multimodal component analysis. Moreover, if the observations contain additional structures such as spatial or temporal structures, these may be used for the multidimensional separation [126, 276].

Hyvärinen and Hoyer [121] presented a special case of k-ISA by combining it with invariant feature subspace analysis. They model the dependence within a k-tuple explicitly, and are therefore able to propose more efficient algorithms without having to resort to the problematic multidimensional density estimation. A related relaxation of the ICA assumption is given by topographic ICA [122], where dependencies between all components are assumed and modeled along a topographic structure (e.g. a two-dimensional grid). However, these two approaches are not completely blind anymore. Bach and Jordan [13] formulate ISA as a component clustering problem, which necessitates a model for intercluster independence and intracluster dependence. For the latter, they propose to use a tree structure as employed by their tree-dependent component analysis [12]. Together with intercluster independence, this implies a search for a transformation of the mixtures into a forest (i.e. a set of disjoint trees). However, the above models are all semiparametric, and hence not fully blind. In the following, we will review two contributions, [247] and [251], where no additional structures were necessary for the separation.

Fixed group structure: k-ISA

A random vector $\mathbf{y}$ is called an *independent component* of the random vector $\mathbf{x}$ if there exist an invertible matrix $\mathbf{A}$ and a decomposition $\mathbf{x} = \mathbf{A}(\mathbf{y}, \mathbf{z})$ such that $\mathbf{y}$ and $\mathbf{z}$ are stochastically independent. Note that this is a more general notion of independent components in the sense of ICA, since we do not require them to be one-dimensional.

The goal of a general *independent subspace analysis (ISA)* or *multidimensional independent component analysis*, is the decomposition of an arbitrary random vector $\mathbf{x}$ into independent components. If $\mathbf{x}$ is to be decomposed into one-dimensional components, this coincides with ordinary ICA. Similarly, if the independent components are required to be of the same dimension k, then this is denoted by multidimensional ICA of fixed group size k, or simply k-ISA.

As we have seen before, an important structural aspect in the search for decompositions is the knowledge of the number of solutions (i.e. the indeterminacies of the problem). Clearly, given an ISA solution, invertible transforms in each component (scaling matrices $\mathbf{L}$), as well as permutations of components of the same dimension (permutation matrices $\mathbf{P}$), give an ISA of $\mathbf{x}$. This is of course known for 1-ISA (i.e. ICA, see section 4.2).

In [247], we were able to extend this result to k-ISA, given some additional restrictions to the model: We denoted $\mathbf{A}$ as *k-admissible* if for each $r, s = 1, \ldots, n/k$ the (r, s) sub-k-matrix of $\mathbf{A}$ is either invertible or zero. Then theorem 5.1 can be derived from the multivariate Darmois-Skitovitch theorem (see section 4.2) or using our previously discussed approach via differential equations [250].

THEOREM 5.1 SEPARABILITY OF k-ISA: Let $\mathbf{A} \in \mathrm{Gl}(n; \mathbb{R})$ be k-admissible, and let $\mathbf{s}$ be a k-independent, n-dimensional random vector having no Gaussian k-dimensional component. If $\mathbf{As}$ is again k-independent, then $\mathbf{A}$ is the product of a k-block-scaling and permutation matrix.

This shows that k-ISA solutions are unique except for trivial transformations, if the model has no Gaussians and is admissible, and can now be turned into a separation algorithm.

ISA with known group structure via joint block diagonalization

In order to solve ISA with fixed block size k or at least known block structure, we will use a generalization of joint diagonalization which searches for block structures instead of diagonality. We are not interested in the order of the blocks, so the block structure is uniquely specified by fixing a partition $n = m_1 + \ldots + m_r$ of n and setting $\mathbf{m} := (m_1, \ldots, m_r) \in \mathbb{N}^r$. An $n \times n$ matrix is said to be $\mathbf{m}$-*block diagonal* if it is of the form

$$\begin{pmatrix} \mathbf{M}_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mathbf{M}_r \end{pmatrix}$$

with arbitrary $m_i \times m_i$ matrices $\mathbf{M}_i$.

As with generalization of JD in the case of known block structure, the *joint* $\mathbf{m}$-*block diagonalization problem* is defined as the minimization of

$$f^{\mathbf{m}}(\hat{\mathbf{A}}) := \sum_{i=1}^{K} \|\hat{\mathbf{A}}^\top \mathbf{C}_i \hat{\mathbf{A}} - \mathrm{diag}^{\mathbf{m}}(\hat{\mathbf{A}}^\top \mathbf{C}_i \hat{\mathbf{A}})\|_F^2 \tag{5.6}$$

with respect to the orthogonal matrix $\hat{\mathbf{A}}$, where $\mathrm{diag}^{\mathbf{m}}(\mathbf{M})$ produces a $\mathbf{m}$-block diagonal matrix by setting all other elements of $\mathbf{M}$ to zero. Indeterminacies of any $\mathbf{m}$-JBD are $\mathbf{m}$-*scaling* (i.e. multiplication by an $\mathbf{m}$-block diagonal matrix from the right), and $\mathbf{m}$-*permutation*, which is defined by a permutation matrix that swaps only blocks of the same size.

Algorithms to actually perform JBD have been proposed [2, 80]. In the following we will simply perform joint diagonalization and then permute the columns of $\mathbf{A}$ to achieve block diagonality—in experiments this turns out to be an efficient solution to JBD, although other, more sophisticated pivot selection strategies for JBD are of interest [81]. The fact that JD induces JBD has been conjectured by Abed-Meraim and Belouchrani [2], and we were able to give a partial answer with theorem 5.2.

THEOREM 5.2 JBD VIA JD: Any block-optimal JBD of the $\mathbf{C}_i$'s (i.e., a zero of $f^{\mathbf{m}}$) is a local minimum of the JD cost function f from equation (5.1).

Clearly, not just any JBD minimizes f; only those such that in each

block of size m_k, $f(\hat{\mathbf{A}})$, when restricted to the block, is maximal over $\mathbf{A} \in O(m_k)$, which we denote as *block-optimal*. The proof is given in [251].

In the case of k-ISA, where $\mathbf{m} = (k, \ldots, k)$, we used this result to propose an explicit algorithm [249]. Consider the BSS model from equation (4.1). As usual, by preprocessing we may assume whitened observations $\mathbf{x}$, so $\mathbf{A}$ is orthogonal. For the density $p_\mathbf{s}$ of the sources, we therefore get $p_\mathbf{s}(\mathbf{s}_0) = p_\mathbf{x}(\mathbf{A}\mathbf{s}_0)$. Its Hessian transforms like a 2-tensor, which locally at $\mathbf{s}_0$ (see section 4.2) guarantees

$$\mathbf{H}_{\ln p_\mathbf{s}}(\mathbf{s}_0) = \mathbf{H}_{\ln p_\mathbf{x} \circ \mathbf{A}}(\mathbf{s}_0) = \mathbf{A}\mathbf{H}_{\ln p_\mathbf{x}}(\mathbf{A}\mathbf{s}_0)\mathbf{A}^\top. \tag{5.7}$$

The sources $\mathbf{s}(t)$ are assumed to be k-independent, so $p_\mathbf{s}$ factorizes into r groups each depending on k separate variables Thus $\ln p_\mathbf{s}$ is a sum of functions depending on k separate variables, and hence $\mathbf{H}_{\ln p_\mathbf{s}}(\mathbf{s}_0)$ is k-block diagonal. Hessian ISA now simply uses the block-diagonality structure from equation (5.7) and performs JBD of estimates of a set of Hessians $\mathbf{H}_{\ln p_\mathbf{s}}(\mathbf{s}_i)$ evaluated at different sampling points $\mathbf{s}_i$. This corresponds to using the HessianICA source condition from table 5.1. Other source conditions, such as contracted quadricovariance matrices [46] can also be used in this extended framework [251].

Unknown group structure: General ISA

A serious drawback of k-ISA (and hence of ICA) lies in the fact that the requirement of fixed group size k does not allow us to apply this analysis to an arbitrary random vector. Indeed, theoretically speaking, it may be applied only to random vectors following the k-ISA blind source separation model, which means that they have to be mixtures of a random vector that consists of independent groups of size k. If this is the case, uniqueness up to permutation and scaling holds according to theorem 5.1. However, if k-ISA is applied to any random vector, a decomposition into groups that are only "as independent as possible" cannot be unique, and depends on the contrast and the algorithm. In the literature, ICA is often applied to find representations fulfilling the independence condition only as well as possible. However, care has to be taken; the strong uniqueness result is not valid anymore, and the results may depend on the algorithm as illustrated in figure 5.3.

In contrast to ICA and k-ISA, we do not want to fix the size of the

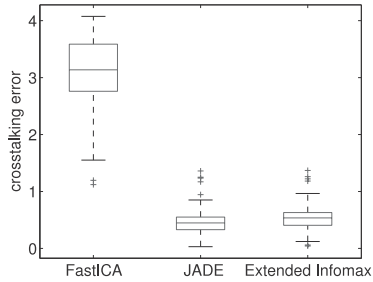


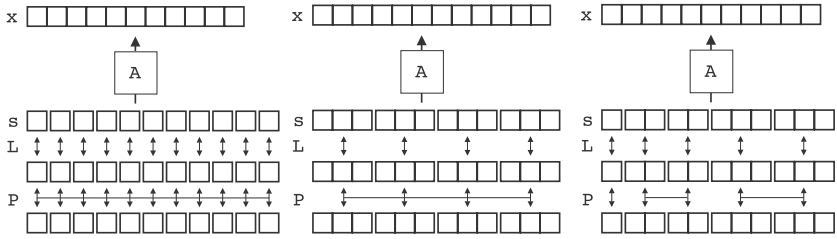
Figure 5.3
Applying ICA to a random vector $\mathbf{x} = \mathbf{As}$ that does not fulfill the ICA model; here $\mathbf{s}$ is chosen to consist of a two-dimensional and a one-dimensional irreducible component. Shown are the statistics over 100 runs of the Amari error of the random original and the reconstructed mixing matrix using the three ICA algorithms FastICA, JADE, and Extended Infomax. Clearly, the original mixing matrix could not be reconstructed in any of the experiments. However, interestingly, the latter two algorithms do indeed find an ISA up to permutation, which can be explained by theorem 5.2.

groups $\mathbf{S}_i$ in advance. Of course, some restriction is necessary; otherwise, no decomposition would be enforced at all. The key idea in [251], is to allow only irreducible components defined as random vectors without lower-dimensional independent components.

The advantage of this formulation is that it can clearly be applied to any random vector, although of course a trivial decomposition might be the result in the case of an irreducible random vector. Obvious indeterminacies of an ISA of $\mathbf{x}$ are scalings (i.e. invertible transformations within each $\mathbf{s}_i$) and permutation of $\mathbf{s}_i$ of the same dimension. These are already all indeterminacies, as shown by theorem 5.3.

Theorem 5.3 Existence and Uniqueness of ISA: Given a random vector $\mathbf{X}$ with existing covariance, an ISA of $\mathbf{X}$ exists and is unique except for permutation of components of the same dimension and invertible transformations within each independent component and within the Gaussian part.

Here, no Gaussians had to be excluded from $\mathbf{S}$ (as in the previous uniqueness theorems), because a dimension reduction results from [104, 251] can be used. The connection of the various factorization models and



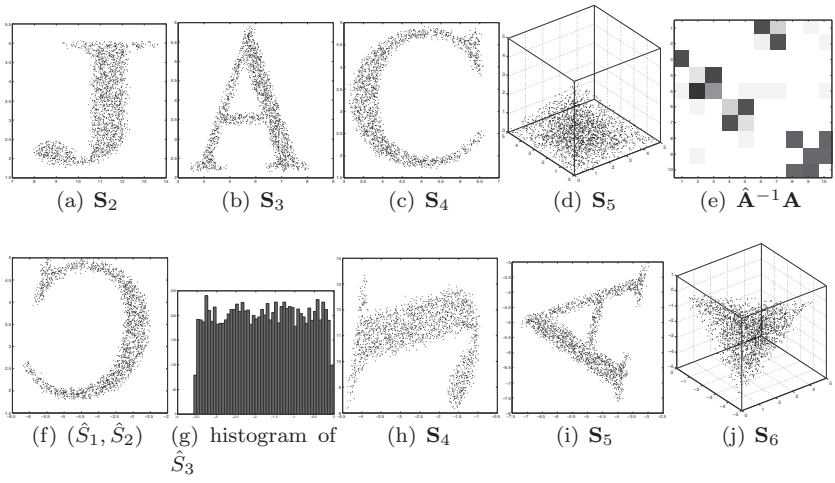
(a) ICA (b) ISA with fixed group-size (c) general ISA

Figure 5.4
Linear factorization models for a random vector $\mathbf{x} = \mathbf{As}$ and the resulting indeterminacies, where $\mathbf{L}$ denotes a one- or higher-dimensional invertible matrix (scaling), and $\mathbf{P}$ denotes a permutation, to be applied only along the horizontal line as indicated in the figures. The small horizontal gaps denote statistical independence. One of the key differences between the models is that general ISA may always be applied to *any* random vector $\mathbf{x}$, whereas ICA and its generalization, fixed-size ISA, yield unique results only if $\mathbf{x}$ follows the corresponding model.

the corresponding uniqueness results are illustrated in figure 5.4.

Again, we turned this uniqueness result into a separation algorithm, this time by considering the JADE source condition based on fourth-order cumulants. The key idea was to translate irreducibility into maximal block diagonality of the source condition matrices $\mathbf{C}_i(\mathbf{s})$. Algorithmically, JBD was performed using JD first using theorem 5.2, followed by permutation and block size identification, see [251].

As a short example, we consider a general ISA problem in dimension $n = 10$ with the unknown partition $\mathbf{m} = (1, 2, 2, 2, 3)$. In order to generate two- and three-dimensional irreducible random vectors, we decided to follow the nice visual ideas from [207] and to draw samples from a density following a known shape - in our case 2-D letters or 3-D geometrical shapes. The chosen source densities are shown in figure 5.5(a-d). Another 1-D source following a uniform distribution was constructed. Altogether, 10^4 samples were used. The sources $\mathbf{S}$ were mixed by a mixing matrix $\mathbf{A}$ with coefficients uniformly randomly sampled from $[-1, 1]$ to give mixtures $\mathbf{X} = \mathbf{AS}$. The recovered mixing matrix $\hat{\mathbf{A}}$ was then estimated, using the above block JADE algorithm with unknown block size; we observed that the method is quite sensitive to the choice of the threshold (here $\theta = 0.015$). Figure 5.5(e) shows the composed mixing-separating system $\hat{\mathbf{A}}^{-1}\mathbf{A}$; clearly the matrices are equal except

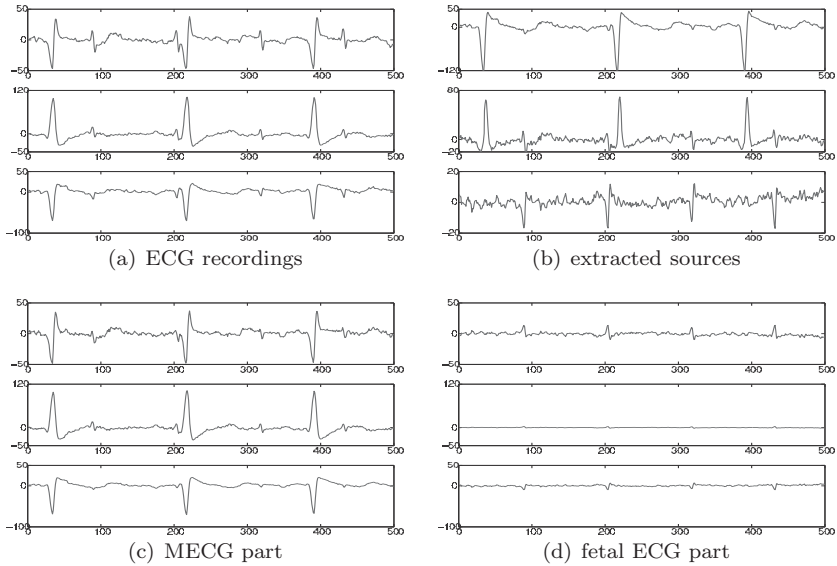


(a) $\mathbf{S}_2$ (b) $\mathbf{S}_3$ (c) $\mathbf{S}_4$ (d) $\mathbf{S}_5$ (e) $\hat{\mathbf{A}}^{-1}\mathbf{A}$

(f) $(\hat{S}_1, \hat{S}_2)$ (g) histogram of $\hat{S}_3$ (h) $\mathbf{S}_4$ (i) $\mathbf{S}_5$ (j) $\mathbf{S}_6$

Figure 5.5
Application of general ISA for unknown sizes $\mathbf{m} = (1, 2, 2, 2, 3)$. Shown are the scatter plots (i.e. densities of the source components) and the mixing-separating map $\hat{\mathbf{A}}^{-1}\mathbf{A}$.

for block permutation and scaling, which experimentally confirms theorem 5.3. The algorithm found a partition $\hat{\mathbf{m}} = (1, 1, 1, 2, 2, 3)$, so one 2-D source was misinterpreted as two 1-D sources, but by using previous knowledge combination of the correct two 1-D sources yields the original 2-D-source. The resulting recovered sources $\hat{\mathbf{S}} := \hat{\mathbf{A}}^{-1}\mathbf{X}$, figures 5.5(f-j), then equal the original sources except for permutation and scaling within the sources — which in the higher-dimensional cases implies transformations such as rotation of the underlying images or shapes. When applying ICA (1-ISA) to the above mixtures, we cannot expect to recover the original sources, as explained in figure 5.3. However, some algorithms might recover the sources up to permutation. Indeed, SJADE equals JADE with additional permutation recovery because the joint block diagonalization is performed using joint diagonalization. This explains why JADE retrieves meaningful components even in this non-ICA setting, as observed in [43].



(a) ECG recordings (b) extracted sources

(c) MECG part (d) fetal ECG part

Figure 5.6
Independent subspace analysis with known block structure $\mathbf{m} = (2, 1)$ is applied to fetal ECG. (a) shows the ECG recordings. The underlying FECG (4 heartbeats) is partially visible in the dominating MECG (3 heartbeats). (b) gives the extracted sources using ISA with the Hessian source condition from table 5.1 with 500 Hessian matrices. In (c) and (d) the projections of the mother sources (first two components from (b)) and the fetal source (third component from (b)) onto the mixture space (a) are plotted.

Application to ECG data

Finally, we report the example from [249] on how to apply the Hessian ISA algorithm to a real-world data set. Following [43], we show how to separate fetal ECG (FECG) recordings from the mother's ECG (MECG). Our goal is to extract an MECG component and an FECG component; however we cannot expect to find only a one-dimensional MECG due to the fact that projections of a three-dimensional vector (electric) field are measured. Hence, modeling the data by a multidimensional BSS problem with $k = 2$ (but allowing for an additional one-dimensional component) makes sense. Application of ISA extracts a two-dimensional MECG component and a one-dimensional FECG component. After block permutation we get estimated mixing matrix $\mathbf{A}$ and

sources $\mathbf{s}(t)$, as plotted in figure 5.6(b). A decomposition of the observed ECG data $\mathbf{x}(t)$ can be achieved by composing the extracted sources using only the relevant mixing columns. For example, for the MECG part this means applying the projection $\Pi_M := (\mathbf{a}_1, \mathbf{a}_2, 0)\mathbf{A}^{-1}$ to the observations. The results are plotted in figures 5.6 (c) and (d). The FECG is most active at sensor 1 (as visual inspection of the observation confirms). When comparing the projection matrices with the results from [43], we get quite high similarity of the ICA-based results, and a modest difference from the projections of the time-based algorithm.

EXERCISES

1. How does k-ISA for $k = 1$ compare with ICA, and how with complex ICA if $k = 2$?

2. *Autodecorrelation*

a) Implement a time-based ICA algorithm using autodecorrelation - how many calculations of an eigenvalue decomposition are needed?

b) Instead of only two autocorrelations, use a joint diagonalization method, such as Cardoso's [42] from
`http://www.tsi.enst.fr/~cardoso/Algo/Joint_Diag/`

c) Apply this algorithm to the separation of the artificial mixture of two natural images. For this, vectorize the images in order to get two "time series"'that can be mixed. Up to which noise level can you still separate the images?

d) Use the same algorithm to the separate the images, but now diagonalize not the one-dimensional autocorrelations but the multi dimensional ones. How does this perform with increasingly noise level?

3. *Multidimensional sources*

a) Generate two multi dimensional, independent sources by taking i.i.d. samples from nontrivial compact regions of $\mathbb{R}^n$, (e.g. letters or discs) as in figure 5.5.

b) Apply fastICA/JADE to separate the sources themselves and then a random mixture. Show that in general, the multi dimensional sources cannot be recovered.

c) Test all permutations of the recovered sources to show that after permutation, even the multi dimensional sources are typically restored.

6 Pattern Recognition Techniques

Modern classification paradigms such as neural networks, genetic algorithms, and neuro–fuzzy methods have become very popular tools in medical imaging. Whether diagnosis, therapeutics, or prognosis, artificial intelligence methods are leaders in these applications. In conjunction with computer vision, these methods have become extremely important for the development of computer-aided diagnosis systems which support the analysis and interpretation of the routine production of the vast numbers of medical images.

Artificial neural networks mimic the biological neural processing based on a group of information-processing units, called neurons, and a connectionist approach to computation. The neural architecture enables a highly parallel processing and an adaptive learning which changes the values of the interconnections between the neurons, called synapses, such that the system learns directly from the data. Like the brain, artificial neural networks are able to process incomplete, noise-corrupted, and inconsistent information.

This chapter gives an overview of the most important approaches in artificial neural networks and their application to biomedical imaging. Traditional architectures such as unsupervised or supervised architectures, and modern paradigms such as kernel methods, are presented in great detail. The chapter also reviews the classifier evaluation techniques in which the most relevant one represents the diagnostic accuracy of classification measured by ROC curves.

6.1 Learning Paradigms and Architecture Types

Neural networks are adaptive, interconnected nonlinear systems which are able to generalize and adapt to new environments by learning. Besides its architecture, the learning algorithm is the most important component for neural information processing. By learning, we mean an iterative updating algorithm, which changes the interconnections between the neurons according to input data. Learning, ideally inspired by connectionist principles, falls for artificial neural networks into two categories: supervised and unsupervised learning.

Supervised learning represents an error-correction learning which re-

quires that both the input data and the corresponding target answers are presented to the network. The error signal caused by the mismatch between known target outputs and actual outputs is employed to iteratively adapt the connection strength between the neurons. In unsupervised learning, on the other hand, a different paradigm is implemented: the training data of known labels are not available, and thus an error correction for all processing units or neurons does not take place. The neurons compete with each other, and the connections of the winner are adapted to the new input data. Learning is correlational and creates categories of neurons specialized to similar or correlated input data.

As previously mentioned, neural networks implement a nonlinear mapping between an input space and an output space by indirectly inferring the structure of the mapping from given data pairs.

There are three basic mapping neural networks known in the literature [110]:

1. Recurrent networks: The feedback structure determines the networks' temporal dynamics and thus enables the processing of sequential inputs. This dynamic system is highly nonlinear because of the nonlinear input-output mechanisms. This coupled with a sophisticated weights adjustment paradigm, poses many stability problems for the overall dynamic behavior. A form to control the dynamic behavior is based on choosing a stabilizing learning mechanism imposed by strict conditions on the "energy" function of this system. The most prominent representant is the Hopfield neural network [118]. Less known and previously used was the bidirectional associative memory (BAM) [143].
2. Multilayer feedforward neural networks: These are composed of a hierarchy of multiple units, organized in an input layer, an output layer and at least one hidden layer. Their neurons have nonlinear activations enabling the approximation of any nonlinear function or, equivalently, the classification of nonlinearly separable classes. The most important examples of these networks are the multilayer perceptron [159], the backpropagation–type neural network [61], and the radial–basis neural network [179].
3. Local interaction–based neural networks: These architectures implement the local information-processing mechanism in the brain. The learning mechanism is a competitive learning, and updates the weights based on the input patterns. In general, the winning neuron and those neurons in its close proximity are positively rewarded or reinforced while the others

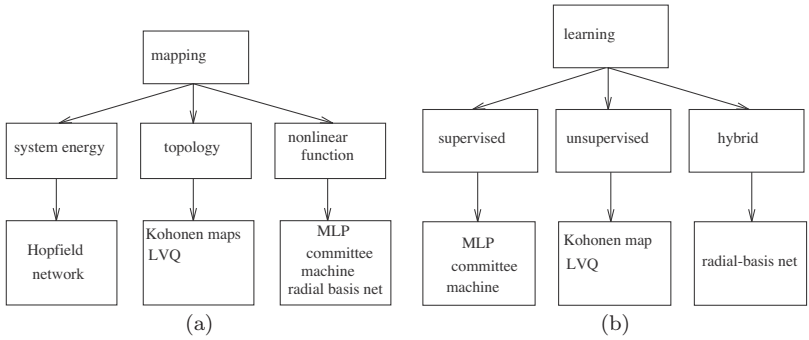


Figure 6.1
Classification of neural networks based on (a) architecture type and (b) learning algorithm.

are suppressed. This processing concept is called *lateral inhibition* and is mathematically described by the Mexican-hat function. The biologically closest network is the von der Malsburg model [277, 284]. Other networks are the Kohonen maps [139] and the ART maps [100, 101].

The previously introduced concepts regarding neural architecture and learning mechanisms are summarized in figure 6.1.

The theory and representation of the various network types are motivated by the functionality and representation of biological neural networks. In this sense, processing units are usually referred to as *neurons*, and interconnections are called *synaptic connections*.

Although different neural models are known, all have the following basic components in common:

1. A finite set of neurons $a(1), a(2), \ldots, a(n)$ with each neuron having a specific activity at time t, which is described by $a_t(i)$.
2. A finite set of neural connections $\mathbf{W} = (w_{ij})$, where w_{ij} describes the strength of the connection of neuron $a(i)$ with neuron $a(j)$.
3. A *propagation rule* $\tau_t(i) = \sum_{j=1}^{n} a_t(j) w_{ij}$.
4. An *activation function* f, which has τ as an input value and produces the next state of the neuron $a_{t+1}(i) = f(\tau_t(i) - \theta)$, where θ is a threshold and f is a nonlinear function such as a hard limiter, threshold logic, or sigmoid function.

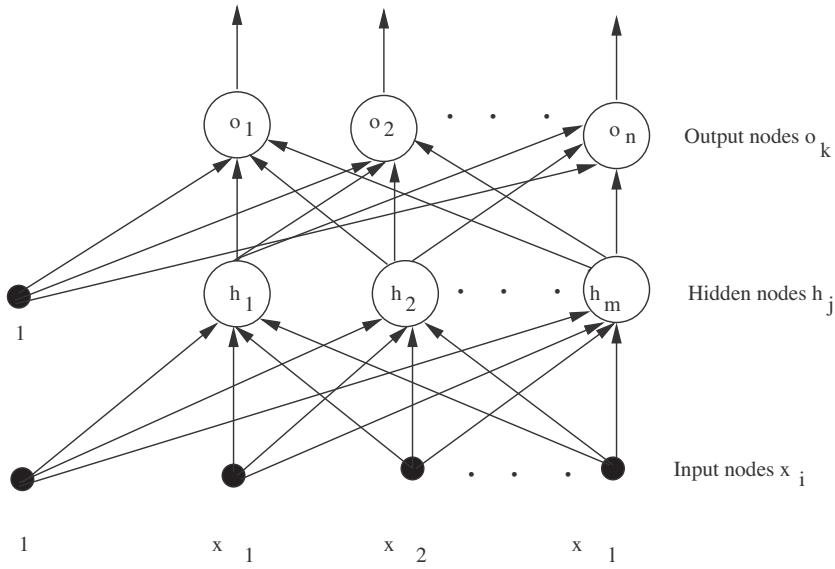


Figure 6.2
Two-layer perceptron.

6.2 Multilayer Perceptron (MLP)

Multilayer perceptrons are one of the most important neural architectures, with applications in both medical image processing and signal processing. They have a layered, feedforward structure with an error-based training algorithm. The architecture of the MLP is completely defined by an *input layer*, one or more *hidden layers*, and an *output layer*. Each layer consists of at least one neuron. The input vector is applied to the input layer and passes the network in a forward direction through all layers. Figure 6.2 illustrates the configuration of the MLP.

A neuron in a hidden layer is connected to every neuron in the layer above it and below it. In figure 6.2, weight w_{ij} connects input node x_i to hidden node h_j, and weight v_{jk} connects h_j to output node o_k. Classification starts by assigning the input nodes x_i, $1 \leq i \leq l$ equal to the corresponding data vector component. Then data propagates in a forward direction through the perceptron until the output nodes o_k, $1 \leq k \leq n$, are reached. The MLP is able to distinguish 2^n separate classes, given that its outputs are assigned to the binary values 0 and 1.

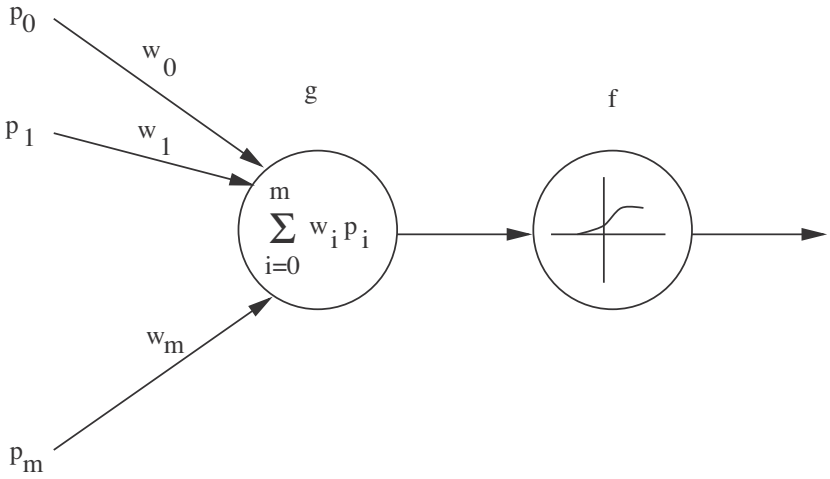


Figure 6.3
Propagation rule and activation function for the MLP network.

The input vector is usually the result of a preprocessing step of a measured sensor signal. This signal is denoised, and the most relevant information is obtained based on feature extraction and selection. The MLP acts as a classifier, estimates the necessary discriminant functions, and assigns each input vector to a given class. Mathematically, the MLP belongs to the group of universal approximators and performs a nonlinear approximation by using sigmoid kernel functions. The learning algorithm adapts the weights based on minimizing the error between given output and desired output.

The steps that govern the data flow through the perceptron during *classification* are the following [221]:

1. Present the pattern $\mathbf{p} = [p_1, p_2, \ldots, p_l] \in \mathbf{R^l}$ to the perceptron, that is, set $x_i = p_i$ for $1 \leq i \leq l$.
2. Compute the values of the hidden layer nodes as is illustrated in figure 6.3:

$$h_j = \frac{1}{1 + \exp\left[-\left(w_{0j} + \sum_{i=1}^{l} w_{ij} x_i\right)\right]} \quad 1 \leq j \leq m \tag{6.1}$$

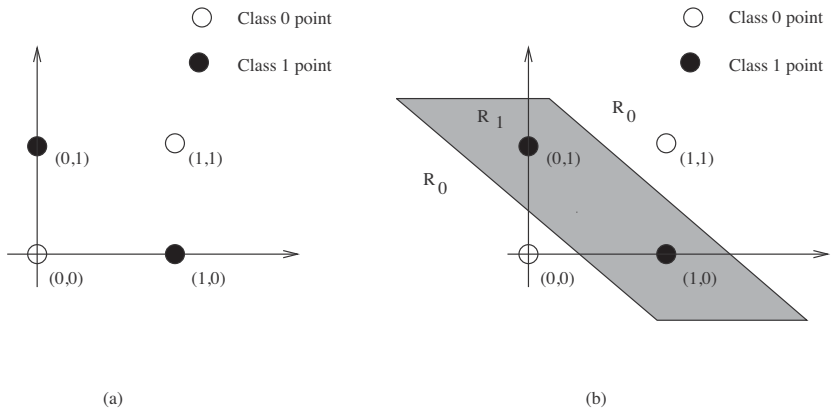


Figure 6.4
XOR-problem and solution strategy using the MLP.

The activation function of all units in the MLP is given by the sigmoid function $f(x) = \frac{1}{1+\exp(-x)}$ and is the standard activation function in feedforward neural networks. It is defined as a monotonically increasing function representing an approximation between nonlinear and linear behavior.

3. Calculate the values of the output nodes based on

$$o_k = \frac{1}{1 + \exp\left(v_{0k} + \sum_{j=1}^{m} v_{jk} h_j\right)} \quad 1 \leq k \leq n \tag{6.2}$$

4. The class $\mathbf{c} = [c_1, c_2, \ldots, c_n]$ that the perceptron assigns $\mathbf{p}$ must be a binary vector. Thus o_k must be the threshold of a certain class at some level τ and depends on the application.
5. Repeat steps 1 2 3 and 4 for each given input pattern.

MLPs are highly nonlinear interconnected systems and serve for both nonlinear function approximation and *nonlinear classification* tasks. A typical classification problem that can be solved only by the MLP is the *XOR problem*. Based on a linear classification rule, R^m can be partitioned into regions separated by a hyperplane. On the other hand, the MLP is able to construct very complex decision boundaries, as depicted in figure 6.4.

MLPs in medical signal processing operate based on either extracted

temporal or spectral features [5, 55, 56]. Key features for medical image processing are shape, texture, contours or size and in most cases describe the region of interest [66, 67].

Backpropagation-type neural networks

MLPs are trained based on the simple idea of the steepest descent method. The core part of the algorithm forms a recursive procedure for obtaining a gradient vector in which each element is defined as the derivative of a cost function (error function) with respect to a parameter. This learning algorithm, known as the error backpropagation algorithm, is bidirectional, consisting of a forward and a backward direction. The learning is accomplished in a supervised mode which requires the knowledge of the output for any given input. The learning is accomplished in two steps: the forward direction and the backward direction. In the forward direction, the output of the network in response to an input is computed, while in the backward direction, an updating of the weights is accomplished. The error terms of the output layer are a function of $\mathbf{c}^t$ and output of the perceptron $(o_1, o_2, \ldots, o_n)$.

The algorithmic description of the *backpropagation* is given below [61]:

1. **Initialization:** Initialize the weights of the perceptron randomly with numbers between –0.1 and 0.1; that is,

$$\begin{aligned} w_{ij} &= \text{random}([-0.1, 0.1]) \quad 0 \le i \le l, 1 \le j \le m \\ v_{jk} &= \text{random}([-0.1, 0.1]) \quad 0 \le j \le m, 1 \le k \le n \end{aligned} \tag{6.3}$$

2. **Presentation of training patterns:** Present $\mathbf{p}^t = [p_1^t, p_2^t, \ldots, p_l^t]$ from the training pair $(\mathbf{p}^t, \mathbf{c}^t)$ to the perceptron and apply steps 1, 2, and 3 from the perceptron classification algorithm described above.
3. **Forward computation (output layer):** Compute the errors $\delta_{ok}, 1 \le k \le n$ in the output layer using

$$\delta_{ok} = o_k(1 - o_k)(c_k^t - o_k), \tag{6.4}$$

where $\mathbf{c}^t = [c_1^t, c_2^t, \ldots, c_n^t]$ represents the correct class of $\mathbf{p}^t$. The vector $(o_1, o_2, \ldots, o_n)$ represents the output of the perceptron.

4. **Forward computation (hidden layer):** Compute the errors δ_{hj}, $1 \leq j \leq m$, in the hidden layers nodes based on

$$\delta_{hj} = h_j(1 - h_j) \sum_{k=1}^{n} \delta_{ok} v_{jk} \tag{6.5}$$

5. **Backward computation (output layer):** Let v_{jk} denote the value of weight v_{jk} after the tth training pattern has been presented to the perceptron. Adjust the weights between the output layer and the hidden layer based on

$$v_{jk}(t) = v_{jk}(t-1) + \eta \delta_{ok} h_j \tag{6.6}$$

   The parameter $0 \leq \eta \leq 1$ represents the learning rate.

6. **Backward computation (hidden layer):** Adjust the weights between the hidden layer and the input layer using

$$w_{ij}(t) = w_{ij}(t-1) + \eta \delta_{hj} p_i^t \tag{6.7}$$

7. **Iteration:** Repeat steps 2 through 6 for each pattern vector of the training data. One cycle through the training set is defined as an iteration.

Design considerations

MLPs represent global approximators by being able to implement any nonlinear mapping between the inputs and the outputs. The minimum requirement for the MLP to represent any function is fulfilled mathematically by imposing only one hidden layer [109]. In the beginning, the architecture of the network has to be carefully chosen since it remains fixed during the training and does not grow or prune like other networks having a hybrid or unsupervised learning scheme. As with all classification algorithms, the feature vector has to be chosen carefully, be representative of the all pattern classes, and provide a good generalization. Feature selection and extraction might be considered in order to remove redundancy of the data.

The number of neurons in the input layer equals the dimension of the training feature vector while those in the output layer are determined by the number of classes of feature vectors required to be distinguished. A

critical component of the training of the MLP is the number of neurons in the hidden layer. Too many neurons result in overlearning, and too few impair the generalization property of the MLP.

The complexity of the MLP is determined by the number of its adaptable parameters such as weights and biases. The goal of each classification problem is to achieve optimal complexity.

In general, complexity can be influenced by (1) data preprocessing such as feature selection/extraction or reduction, (2) training schemes such as cross validation and early stopping, and (3) network structure achieved through modular networks comprising multiple networks.

The cross validation technique is usually employed when we aim at a good generalization in terms of the optimal number of hidden neurons and when the training has to be stopped. Cross validation is achieved by dividing the training set into two disjoint sets. The first set is used for learning, and the latter is used for checking the classification error as long as there is an improvement of this error. Thus, cross validation becomes an effective procedure for detecting overfitting.

In general, the best generalization is achieved when three disjoint data sets are used: a training, a validation and a testing set. While the first two sets avoid overfitting, the latter is used to show a good classification.

Modular networks

Modular networks represent an important class of connectionist architectures and implement the principle of divide and conquer: a complex task (classification problem) is achieved collectively by a mixture of experts (hierarchy of neural networks). Mathematically, they belong to the group of universal approximators. Their architecture has two main components: expert networks and a gating network. The idea of the committee machine was first introduced by Nilsson [186]. The most important modular networks types are shown below.

- *Mixture of experts*: The architecture is based on experts and a single gating network that yields a nonlinear function of the individual responses of the experts.
- *Hierarchical mixture of experts*: This comprises several groups of mixture of experts whose responses are evaluated by a gating network. The architecture is a tree in which the gating networks sits at the

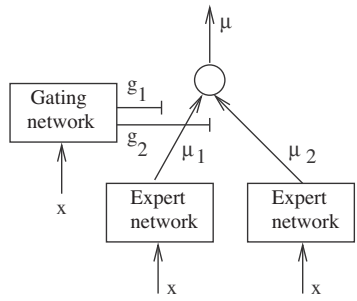


Figure 6.5
Mixture of two expert networks.

nonterminals of the tree.

Figure 6.5 shows the typical architecture of a mixture of experts. These networks receive the vector $\mathbf{x}$ as input and produce scalar outputs that are a partition of unity at each point in the input space. They are linear with the exception of a single output nonlinearity. Expert network i produces its output μ_i as a generalized function of the input vector $\mathbf{x}$ and a weight vector $\mathbf{u}_i$:

$$\mu_i = \mathbf{u}_i^T \mathbf{x} \tag{6.8}$$

The neurons of the gating networks are nonlinear.

Let ξ_i be an intermediate variable; then

$$\xi_i = \mathbf{v}_i^T \mathbf{x} \tag{6.9}$$

where $\mathbf{v}_i$ is a weight vector. Then the ith output is the "softmax" function of ξ_i given as

$$g_i = \frac{\exp(\xi_i)}{\sum_k \exp(\xi_k)}. \tag{6.10}$$

Note that $g_i > 0$ and $\sum_i g_i = 1$. The g_is can be interpreted as providing a "soft" partitioning of the input space.

The output vector of the mixture of experts is the weighted output of the experts, and becomes

$$\mu = \sum_i g_i \mu_i \tag{6.11}$$

Both g and μ depend on the input $\mathbf{x}$; thus, the output is a nonlinear function of the input.

6.3 Self–organizing Neural Networks

Self-organizing maps implement competition-based learning paradigms. They represent a nonlinear mapping from a higher-dimensional feature space onto a usually 1-D or 2-D lattice of neurons. This neural network has the closest resemblance to biological cortical maps. The training mechanism is based on competitive learning: similarity (dissimilarity) is selected as a measure, and the winning neuron is determined based on the largest activation. The output units are imposed on a neighborhood constraint such that similarity properties between input vectors are reflected in the output neurons' weights. If both the input and the neuron spaces (lattices) have the same dimension, then this self-organizing *feature map* [141] also becomes topology-preserving.

Self–organizing feature map

Mathematically, the self–organizing map (SOM) determines a transformation from a high–dimensional input space onto a one–dimensional or two–dimensional discrete map. The transformation takes place as an adaptive learning process such that when it converges, the lattice represents a topographic map of the input patterns. The training of the SOM is based on a random presentation of several input vectors, one at a time. Typically, each input vector produces the firing of one selected neighboring group of neurons whose weights are close to the input vector.

The most important features of such a network are the following:

1. A 1-D or 2-D *lattice of neurons* on which input patterns of arbitrary dimension are mapped, as visualized in figure 6.6a.
2. A measure that determines a *winner neuron* based on the similarity between the weight vector and the input vector.

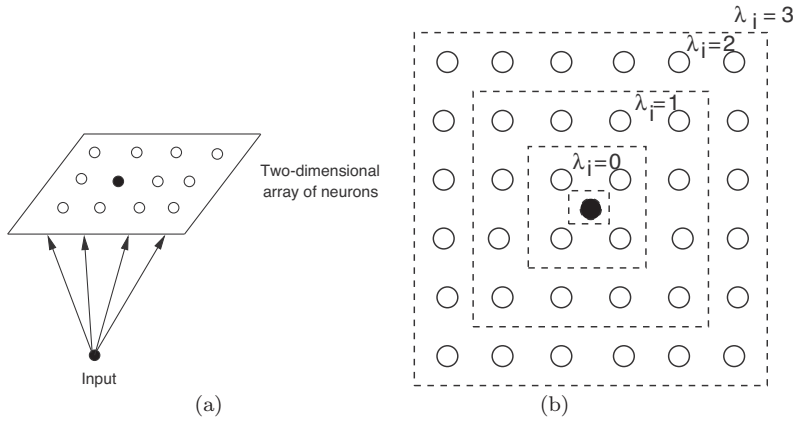


Figure 6.6
(a) Kohonen neural network and (b) neighborhood Λ_i, of varying size, around the "winning" neuron i, (the black circle).

3. A learning paradigm that chooses the winner and its neighbors simultaneously. A neighborhood $\Lambda_{i(\mathbf{x})}(n)$ is centered on the winning neuron and is adapted in its size over time n. Figure 6.6b illustrates such a neighborhood, which first includes the whole neural lattice and then shrinks gradually to only one "winning neuron" (the black circle).
4. An adaptive learning process that updates positively (reinforces) all neurons in the close neighborhood of the winning neuron, and updates negatively (inhibits) all those that are farther from the winner.

The learning algorithm of the self-organized map is simple and is described below.

1. **Initialization**: Choose random values for the initial weight vectors $\mathbf{w}_j(0)$ to be different for $j = 1, 2, \ldots, N$, where N is the number of neurons in the lattice. The magnitude of the weights should be small.
2. **Sampling**: Draw a sample $\mathbf{x}$ from the input data; the vector $\mathbf{x}$ represents the new pattern that is presented to the lattice.
3. **Similarity Matching**: Find the "winner neuron" $i(\mathbf{x})$ at time n based on the minimum distance Euclidean criterion:

$$i(\mathbf{x}) = \arg\min_j ||\mathbf{x}(n) - \mathbf{w}_j(n)||, \quad j = 1, 2, \ldots, N \tag{6.12}$$

4. **Adaptation**: Adjust the synaptic weight vectors of all neurons (winners or not), using the update equation

$$\mathbf{w}_j(n+1) = \begin{cases} \mathbf{w}_j(n) + \eta(n)[\mathbf{x}(n) - \mathbf{w}_j(n)], & j \in \Lambda_{i(\mathbf{x})}(n) \\ \mathbf{w}_j(n), & \text{else} \end{cases} \tag{6.13}$$

where $\eta(n)$ is the learning rate parameter and $\Lambda_{i(\mathbf{x})}(n)$ is the *neighborhood function* centered around the winning neuron $i(\mathbf{x})$; both $\eta(n)$ and $\Lambda_{i(\mathbf{x})}$ are functions of the discrete time n, and thus are continuously adapted for optimal learning.

5. **Continuation**: Go to step 2 until there are no noticeable changes in the feature map.

The presented learning algorithm has some interesting properties, which are described based on figure 6.7.

The feature map implements a nonlinear transformation Φ from a usually higher-dimensional continuous input space X to a spatially discrete output space A:

$$\Phi : X \to A. \tag{6.14}$$

In general, if the dimension between input and output space differs significantly, the map is performing a data compression between the higher-dimensional input space and the lower-dimensional output space. The map preserves the topological relationship that exists in the input space, if the input space has the same dimensionality as the output space. In all other cases, the map is said to be only neighborhood-preserving, in the sense that neighboring regions of the input space activate neighboring neurons on the lattice. In cases where an accurate topological representation of a high-dimensional input data manifold is required, the Kohonen feature map fails to provide perfectly topology-preserving maps.

Self-organizing maps have two fundamental properties:

- Approximation of the input space: The self-organizing feature map Φ, completely determined by the neural lattice, learns the input data distribution by adjusting its synaptic weight vectors $\{\mathbf{w}_j | j = 1, 2, \ldots, N\}$ to provide a good approximation to the input space X.
- Topological ordering achieved by the nonlinear feature map: There is

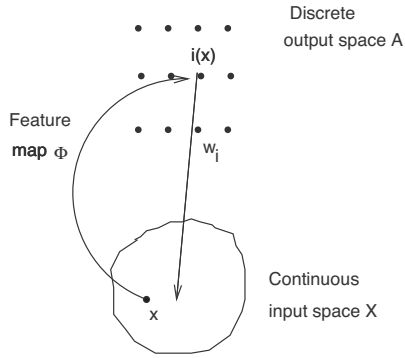


Figure 6.7
Mapping between input space X and output space A.

a correspondence between the location of a neuron on the lattice and a certain domain or distinctive feature of the input space.

Kohonen maps have been applied to a variety of problems in medical image processing [144, 148, 286].

Design considerations

The Kohonen map is mostly dependent on two parameters of the algorithm: the learning rate parameter η and the *neighborhood function* Λ_i. The choice of these parameters is critical for a successful application, and since there are no theoretical results, we have to rely on empirical considerations: the learning rate parameter $\eta(n)$ employed for adaptation of the synaptic vector $\mathbf{w}_j(n)$ should be time-varying. For the first 100 iterations $\eta(n)$ should stay close to unity and decrease thereafter slowly, but remain above 0.1. The neighborhood function Λ_i always has to include the winning neuron in the middle. The function is shrunk slowly and linearly with the time n, and usually reaches a small value of only a couple of neighboring neurons after about 1000 iterations.

Learning vector quantization

Vector quantization (VQ) [99, 156] is an adaptive data classification method which is used both to quantize input vectors into reference or code word values and to apply these values directly to the subsequent classification. VQ has its root in speech processing but has also been suc-

cessfully applied to medical image processing [60]. In image compression, VQ provides an efficient technique for data compression. Compression is achieved by transmitting the index of the code word instead of the vector itself.

VQ can be defined as a mapping that assigns each vector $\mathbf{x} = (x_0, x_1, \cdots, x_{n-1})^T$ in the n–dimensional space R^n to a code word from a finite subset of R^n. The subset $\mathbf{Y} = \{\mathbf{y}_i : i = 1, 2, \cdots, M\}$, representing the set of possible reconstruction vectors is called a *codebook* of size M. Its members are called the *code words*. Note that both the input space and the codebook have the same dimension and several $\mathbf{y}_i$ can be assigned to one class. In the encoding process, a distance measure, usually Euclidean, is evaluated to locate the closest code word for each input vector $\mathbf{x}$. Then the address corresponding to the code word is assigned to $\mathbf{x}$ and transmitted. The distortion between the input vector and its corresponding codeword $\mathbf{y}$ is defined by the distance $d(\mathbf{x}, \mathbf{y}) = ||\mathbf{x} - \mathbf{y}||$, where $||\mathbf{x}||$ represents the norm of $\mathbf{x}$.

A vector quantizer achieving a minimum encoding error is referred to as a *Voronoi quantizer*. Figure 6.8 shows an input data space partitioned into four regions, called Voronoi cells, and the corresponding Voronoi vectors. These regions represent all those input vectors that are very close to the respective Voronoi vector.

Recent developments in neural network architectures lead to a new unsupervised data-clustering technique, the *learning vector quantization* (LVQ). Its architecture is similar to that of a competitive learning network, with the only exception being that each output unit is associated with a class. The learning paradigm involves two steps. In the first step, the closest prototype (Voronoi vector) is located without using class information, while in the second step, the Voronoi vector is adapted. If the class of the input vector and the Voronoi vector match, the Voronoi vector is moved in the direction of the input vector $\mathbf{x}$. Otherwise, the Voronoi vector $\mathbf{w}$ is moved away from this vector $\mathbf{x}$.

The LVQ algorithm is simple and is described below.

1. **Initialization**: Initialize the weight vectors $\{\mathbf{w}_j(0)|j = 1, 2, \ldots, N\}$ by setting them equal to the first N exemplar input feature vectors $\{\mathbf{x_i}|i = 1, 2, \ldots, L\}$.
2. **Sampling**: Draw a sample $\mathbf{x}$ from the input data; the vector $\mathbf{x}$ represents

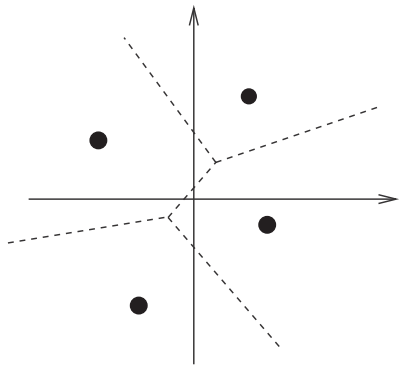


Figure 6.8
Voronoi diagram involving four cells. The circles indicate the Voronoi vectors and are the different region (class) representatives.

the new pattern that is presented to the LVQ.

3. **Similarity matching**: Find the best matching code word (Voronoi vector) $\mathbf{w}_j$ at time n, based on the minimum-distance Euclidean criterion:

$$\arg\min_j ||\mathbf{x}(n) - \mathbf{w}_j(n)||, \quad j = 1, 2, \ldots, N \tag{6.15}$$

4. **Adaptation**: Adjust only the best matching Voronoi vector, while the others remain unchanged. Assume that a Voronoi vector $\mathbf{w}_c$ is the closest to the input vector $\mathbf{x_i}$. We define the class associated with the Voronoi vector $\mathbf{w}_c$ y $C_{\mathbf{w_c}}$, and the class label associated with the input vector $\mathbf{x}_i$ by $C_{\mathbf{x_i}}$. The Voronoi vector $\mathbf{w}_c$ is adapted as follows:

$$\mathbf{w}_c(n+1) = \begin{cases} \mathbf{w}_c(n) + \alpha_n[\mathbf{x_i} - \mathbf{w}_c(n)], & C_{\mathbf{w_c}} = C_{\mathbf{x_i}} \\ \mathbf{w}_c(n) - \alpha_n[\mathbf{x_i} - \mathbf{w}_c(n)], & \text{otherwise} \end{cases} \tag{6.16}$$

where $0 < \alpha_n < 1$.

5. **Continuation**: Go to step 2 until there are no noticeable changes in the feature map.

The learning rate α_n is a positive, small constant; is is chosen as a function of the discrete time parameter n, and decreases monotonically.

The "neural-gas" Algorithm

The *"neural–gas" network* algorithm [166] is an efficient approach which, applied to the task of vector quantization, (1) converges quickly to low distortion errors, (2) reaches a distortion error E lower than that from Kohonen's feature map, and (3) at the same time obeys a gradient descent on an energy surface.

Instead of using the distance $||\mathbf{x} - \mathbf{w_j}||$ or the arrangement of the $||\mathbf{w_j}||$ within an external lattice, it utilizes a neighborhood ranking of the reference vectors $\mathbf{w_i}$ for the given data vector $\mathbf{x}$. The adaptation of the reference vectors is given by

$$\Delta \mathbf{w_i} = \varepsilon e^{-k_i(\mathbf{x}, \mathbf{w_i}/\lambda)}(\mathbf{x} - \mathbf{w_i}) \quad i = 1, \cdots, N \tag{6.17}$$

N is the number of units in the network. The step size $\varepsilon \in [0, 1]$ describes the overall extent of the modification, and k_i is the number of the closest neighbors of the reference vector $\mathbf{w_i}$. λ is a characteristic decay constant.

In [166] it was shown that the average change of the reference vectors can be interpreted as an overdamped motion of particles in a potential that is given by the negative data point density. Added to the gradient of this potential is a "force" which points in the direction of the space, where the particle density is low. The results of this "force" are based on a repulsive coupling between the particles (reference vectors). In its form it's similar to an entropic force and tends to distribute the particles (reference vectors) uniformly over the input space, as is the case with a diffusing gas. Therefore the name "neural-gas" algorithm. Interestingly the reference vectors are slowly adapted, and therefore, pointers that are spatially close at an early stage of the adaptation procedure might not be spatially close later. Connections that have not been updated for a while die out and are removed.

Another important feature of the algorithm compared to the Kohonen algorithm is that it doesn't require a prespecified graph (network). In addition, it can produce topologically preserving maps, which is possible only if the topological structure of the graph matches the topological structure of the data manifold. However, in cases where an appropriate graph cannot be determined from the beginning, for example, in cases where the topological structure of the data manifold is not known in advance or is too complex to be specified, Kohonen's algorithm always fails to provide perfectly topology-preserving maps.

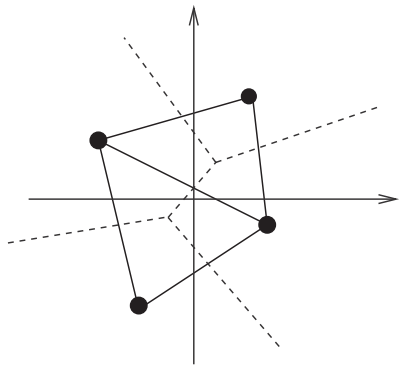


Figure 6.9
Delaunay triangulation.

To obtain perfectly topology-preserving maps, we employ a powerful structure from computational geometry: the *Delaunay triangulation*, which is the dual of the Voronoi diagram [212]. In a plane, the Delaunay triangulation is obtained if we connect all pairs $\mathbf{w_j}$ by an edge if and only if their Voronoi polyhedra are adjacent. Figure 6.9 shows an example of a Delaunay triangulation. The Delaunay triangulation arises as a graph matching the given pattern manifold.

The "neural-gas" algorithm is simple and is described below.

1. **Initialization**: Randomly initialize the weight vectors $\{\mathbf{w}_j | j = 1, 2, \ldots, N\}$ and the training parameters $(\lambda_i, \lambda_f, \varepsilon_i, \varepsilon_f)$, where λ_i, ε_i are initial values of $\lambda(t)$ and $\varepsilon(t)$ and λ_f, ε_f are the corresponding final values.
2. **Sampling**: Draw a sample $\mathbf{x}$ from the input data; the vector $\mathbf{x}$ represents the new pattern that is presented to the "neural-gas" network.
3. **Distortion**: Determine the distortion set $D_{\mathbf{x}}$ between the input vector $\mathbf{x}$ and the weights $\mathbf{w}_j$ at time n, based on the minimum-distance Euclidean criterion:

$$D_{\mathbf{x}} = ||\mathbf{x}(n) - \mathbf{w}_j(n)||, \quad j = 1, 2, \ldots, N \tag{6.18}$$

   Then order the distortion set in ascending order.
4. **Adaptation**: Adjust the weight vectors according to

$$\Delta \mathbf{w_i} = \varepsilon e^{-k_i(\mathbf{x},\mathbf{w_i}/\lambda)}(\mathbf{x} - \mathbf{w_i}) \quad i = 1, \cdots, N, \tag{6.19}$$

where $i = 1, \cdots, N$. The parameters have the time dependencies $\lambda(t) = \lambda_i(\lambda_f/\lambda_i)^{\frac{t}{t_{max}}}$ and $\varepsilon(t) = \varepsilon_i(\varepsilon_f/\varepsilon_i)^{\frac{t}{t_{max}}}$
Increment the time parameter t by 1.

5. **Continuation**: Go to step 2 until the maximum iteration number t_{max} is reached.

6.4 Radial-Basis Neural Networks (RBNN)

Radial-basis neural networks implement a hybrid learning mechanism. They are feedforward neural networks with only one hidden layer; their neurons in the hidden layer are locally tuned; and their responses to an input vector are the outputs of radial-basis functions. The radial-basis functions process the distance between the input vector (activation) and its center (location). The hybrid learning mechanism describes a combination of an unsupervised adaptation of the radial-basis functions' parameter and a supervised adaptation of the output weights using a gradient-based descent method.

The design of a neural network based on radial-basis functions is equivalent to model nonlinear relationships, and implement an interpolation problem in a high-dimensional space. Thus, learning is equivalent to determining an interpolating surface which provides a best match to the training data. To be specific, let us consider a system with n inputs and m outputs, and let $\{x_1, \cdots, x_n\}$ be an input vector and $\{y_1, \cdots, y_m\}$ the corresponding output vector describing the system's answer to that specific input. During the training, the system learns the input and output data distribution, and when this is completed, it is able to find the correct output for any input. Learning can be described as finding the "best" approximation function $\widehat{f}(x_1, \cdots, x_n)$ of the actual input–output mapping function [70, 208].

In the following, we will describe the mathematical framework for solving the approximation problem based on radial-basis neural networks. In this context, we will present the concept of interpolation networks and how any function can be approximated arbitrarily well, based on radial-basis functions under some restrictive conditions.

Interpolation networks

Both the *interpolation network* problem and the *approximation network* problem can be very elegantly solved by a three-layer feedforward neural network. The architecture is quite simple, and has the structure of a feedforward neural network with one hidden layer. The input layer has branching neurons equal in number to the dimension of the input vector. The hidden layer has locally tuned neuron sand performs a nonlinear transformation, while the output layer performs a linear transformation.

The mathematical formulation of the simplified interpolation problem, assuming that there is no noise in the training data, is given below.

Let's assume that to N different points $\{\mathbf{m}_i \in \mathcal{R}^n | i = 1, \cdots N\}$ there correspond N real numbers $\{d_i \in \mathcal{R} | i = 1, \cdots, N\}$. Then find a function $F : \mathcal{R}^n \rightarrow \mathcal{R}$ that satisfies the interpolation condition such that it yields exact desired outputs for all training data:

$$F(\mathbf{m}_i) = d_i \quad \text{for } i = 1, \cdots, N. \tag{6.20}$$

The simplified interpolation network based on radial-basis functions has to determine a simplified representation of the function F that has the form [208]

$$F(\mathbf{x}) = \sum_{i=1}^{N} c_i h(||\mathbf{x} - \mathbf{m}_i||) \tag{6.21}$$

where h is a smooth function, known as a radial–basis function. $||.||$ is the Euclidean norm in $\mathcal{R}^n$ and c_i are weight coefficients. It is assumed that the radial-basis function $h(r)$ is continuous on $[0, \infty)$ and its derivatives on $[0, \infty)$ are strictly monotonic.

The above equation represents a superposition of locally tuned neurons and can be easily represented as a three-layer neural network, as shown in figure 6.10. The figure shows a network with a single output which can be easily generalized.

As previously stated, the presented architecture implements any nonlinear function of the input data. Interpolation networks with radial-basis functions have three key features:

1. This interpolation network with an infinite number of radial-basis neu-

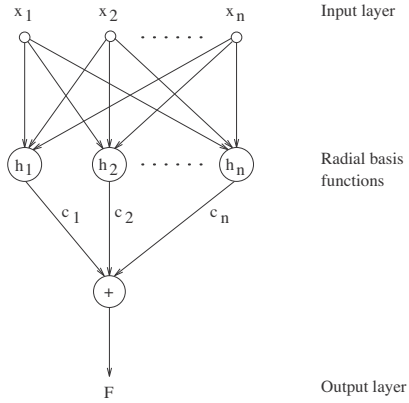


Figure 6.10
Approximation network.

rons represents a *universal approximator* based on the Stone-Weierstrass theorem [209]. In essence, every multivariate, nonlinear, and continuous function can be approximated.

2. The interpolation network with radial-basis functions has the *best approximation* property compared to other neural networks, such as the three-layer perceptron. The sigmoid function does not represent a translation and rotation-invariant function, as the radial-basis function does. Thus, every unknown nonlinear function f is better approximated by a choice of coefficients than any other choice.

3. The interpolation problem can be solved even more simply by choosing radial-basis functions of the same width $\sigma_i = \sigma$, as shown in [197]:

$$F(\mathbf{x}) = \sum_{i=1}^{N} c_i g\left(\frac{||\mathbf{x} - \mathbf{m}_i||}{\sigma}\right) \tag{6.22}$$

In other words, Gaussian functions of the same width can approximate any given function.

Data processing in radial-basis function networks

Radial-basis neural networks implement a hybrid learning algorithm. They have a combined learning scheme of supervised learning for the output weights and unsupervised learning for radial-basis neurons. The ac-

tivation function of the hidden-layer neurons mathematically represents a kernel function but also has an equivalent in neurobiology: it represents the *receptive field*. The unsupervised learning mechanism emulates the "winner takes all" principle found in biological neural networks, and the MLP's backpropagation algorithm is an optimization method, known in statistics as *stochastic approximation*. The theoretical basis of interpolation and regularization networks based on radial-basis functions can be found in [179] and [210].

The RBF network has a feedforward architecture with three distinct layers. Let's assume that the network has N hidden neurons, where the output of the ith output node $f_i(\mathbf{x})$ when the n-dimensional input vector $\mathbf{x}$ is given by

$$f_i(\mathbf{x}) = \sum_{j=1}^{N} w_{ij} \Psi_j(\mathbf{x}) \tag{6.23}$$

$\Psi_j(\mathbf{x}) = \Psi(||\mathbf{x}-\mathbf{m}_j||/\sigma_j)$ represents a suitable rotational and translation-invariant kernel function that defines the output of the jth hidden node. For most RBF networks, $\Psi(.)$ is chosen to be the Gaussian function where the width parameter σ_j is the standard deviation and $\mathbf{m}_j$ is its center. w_{ij} is the weight connecting the jth kernel/hidden node to the ith output node. Figure 6.11a illustrates the architecture of the network.

The steps of a simple learning algorithm for am RBF neural network are presented below.

1. **Initialization**: Choose random values for the initial weights of the RBF network. The magnitude of the weights should be small. Choose the centers $\mathbf{m}_i$ and the shape matrices $\mathbf{K}_i$ of the N given radial-basis functions.
2. **Sampling:** Randomly draw a pattern $\mathbf{x}$ from the input data. This pattern represents the input to the neural network.
3. **Forward computation of hidden layer's activations:** Compute the values of the hidden-layer nodes as is illustrated in figure 6.11b:

$$\psi_i = \exp\left(-d(\mathbf{x}, \mathbf{m}_i, \mathbf{K}_i)/2\right) \tag{6.24}$$

$d(\mathbf{x}, \mathbf{m}_i) = (\mathbf{x} - \mathbf{m}_i)^T \mathbf{K}_i (\mathbf{x} - \mathbf{m}_i)$ is a metric norm and is known as the Mahalanobis distance. The *shape matrix* $\mathbf{K}_i$ is positive definite, and its

elements K_{jk}^{i},

$$K_{jk}^{i} = \frac{h_{jk}}{\sigma_j * \sigma_k} \tag{6.25}$$

are the correlation coefficients h_{jk} and σ_j the standard deviation of the ith shape matrix.
For h_{jk} we choose: $h_{jk} = 1$ for $j = k$, and $|h_{jk}| \leq 1$ otherwise.

4. **Forward computation of output layer's activations:** Calculate the values of the output nodes according to

$$f_{oj} = \varphi_j = \sum_i w_{ji}\psi_i \tag{6.26}$$

5. **Updating:** Adjust weights of all neurons in the output layer based on a steepest descent rule.
6. **Continuation:** Continue with step 2 until no noticeable changes in the error function are observed.

The above algorithm assumes that the locations and the shape of a fixed number of radial-basis functions are known a priori. RBF networks have been applied to a variety of problems in medical diagnosis [301].

Design considerations

The RBF network has only one hidden layer, and the number of basis functions and their shape are problem-oriented and can be determined online during the learning process [151, 206]. The number of neurons in the input layer equals the dimension of the feature vector. Likewise, the number of nodes in the output layer corresponds to the number of classes.

The success of RBF networks as local approximators of nonlinear mappings is highly dependent on the number of radial-basis functions, their widths, and their locations in the feature space. We are free to determine the kernel functions of the RBF networks: they can be fixed or adjusted through either supervised or unsupervised learning during the training phase.

Unsupervised methods determine the locations of the kernel functions based on clustering or learning vector quantization. The best-known techniques are hard c-means algorithm, fuzzy c-means algorithm

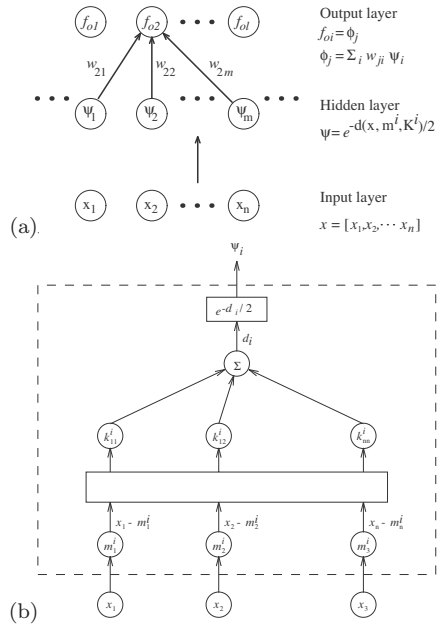


Figure 6.11
RBF network: (a) three-layer model; (b) the connection between input layer and hidden layer neuron.

and fuzzy algorithms for LVQ.

The supervised methods for selection of the locations of the kernels is based on an error-correcting learning. It starts with defining a cost function

$$E = \frac{1}{2}\sum_{j=1}^{P} e_j^2 \tag{6.27}$$

where P is the size of the training sample and e_j is the error defined by

$$e_j = d_j - \sum_{i=1}^{M} w_i G(||\mathbf{x_j} - \mathbf{m}_i||_{C_i}) \tag{6.28}$$

The goal is to find the widths, centers, and weights such that the error E is minimized.

The results of this minimization [110] are summarized in table 6.1.

From that table, we can see that the update equations for w_i, $\mathbf{x_i}$, and Σ_i^{-1} have different learning rates thus visualizing the different timescales. The presented procedure is different from the backpropagation of the MLP.

Table 6.1
Adaptation formulas for the linear weights and the position and widths of centers for an RBF network [110].

1.	**Linear weights of the output layer**
	$\frac{\partial \mathcal{E}(n)}{\partial w_i(n)} = \sum_{j=1}^{N} e_j(n) G(\|\|\mathbf{x_j} - \mathbf{m}_i(n)\|\|)$
	$w_i(n+1) = w_i(n) - \eta_1 \frac{\partial \mathcal{E}(n)}{\partial w_i(n)}, \quad i = 1, \cdots, M$
2.	**Position of the centers of the hidden layer**
	$\frac{\partial \mathcal{E}(n)}{\partial \mathbf{m}_i(n)} = 2 w_i(n) \sum_{j=1}^{N} e_j(n) G^{'}(\|\|\mathbf{x_j} - \mathbf{m}_i(n)\|\|) \mathbf{K^i} [\mathbf{x_j} - \mathbf{m}_i(n)]$
	$\mathbf{m}_i(n+1) = \mathbf{m}_i(n) - \eta_2 \frac{\partial \mathcal{E}(n)}{\partial \mathbf{m}_i(n)}, \quad i = 1, \cdots, M$
3.	**Widths of the centers of the hidden layer**
	$\frac{\partial \mathcal{E}(n)}{\partial \mathbf{k^i}(n)} = -w_i(n) \sum_{j=1}^{N} e_j(n) G^{'}(\|\|\mathbf{x_j} - \mathbf{m}_i(n)\|\|) \mathbf{Q}_{ji}(n)$
	$\mathbf{Q}_{ji}(n) = [\mathbf{x_j} - \mathbf{m}_i(n)][\mathbf{x_j} - \mathbf{m}_i(n)]^T$
	$\mathbf{K^i}(n+1) = \mathbf{K^i}(n) - \eta_3 \frac{\partial \mathcal{E}(n)}{\partial \mathbf{K^i}(n)}$

6.5 Transformation Radial-Basis Networks (TRBNN)

The selection of appropriate features is an important precursor to most statistical pattern recognition methods. A good feature selection mechanism helps to facilitate classification by eliminating noisy or nonrepresentative features that can impede recognition. Even features that provide some useful information can reduce the accuracy of a classifier when the amount of training data is limited. This *curse of dimensionality*, along with the expense of measuring and including features, demonstrates the utility of obtaining a minimum-sized set of features that allow a classifier to discern pattern classes well. Well-known methods in the literature that are applied to feature selection are floating search methods [214] and genetic algorithms [232].

Radial-basis neural networks are excellent candidates for feature selection. It is necessary to add an additional layer to the traditional architecture to obtain a representation of relevant features. The new paradigm is based on an explicit definition of the relevance of a feature

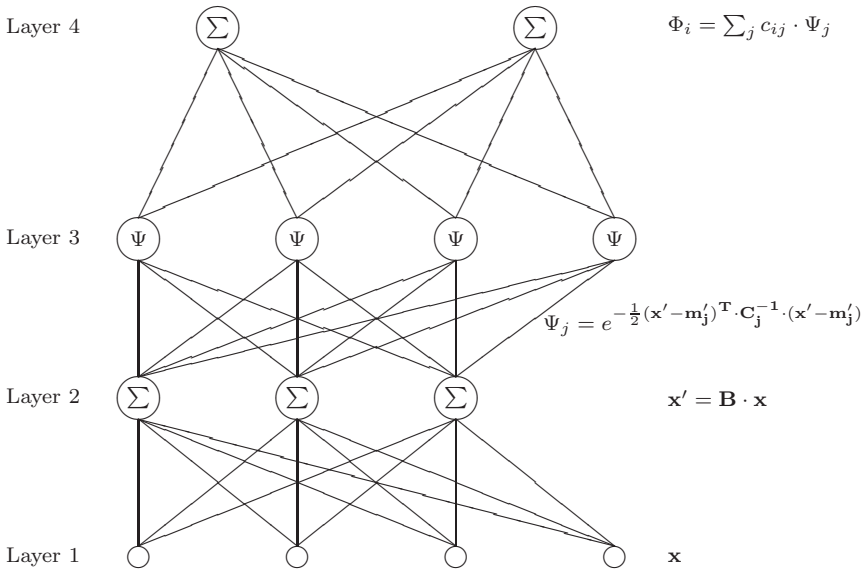


Figure 6.12
Linear transformation of a radial-basis neural network.

and realizes a linear transformation of the feature space.

Figure 6.12 shows the structure of a radial-basis neural network with the additional layer 2, which transforms the feature space linearly by multiplying the input vector and the center of the nodes by the matrix **B**. The covariance matrices of the input vector remain unmodified.

$$\mathbf{x}^{'} = \mathbf{Bx}, \quad \mathbf{m}^{'} = \mathbf{Bm}, \quad \mathbf{C}^{'} = \mathbf{C} \tag{6.29}$$

The neurons in layer 3 evaluate a kernel function for the incoming input and the neurons in the output layer perform a weighted linear summation of the kernel functions:

$$\mathbf{y}(\mathbf{x}) = \sum_{\mathbf{i=1}}^{\mathbf{N}} \mathbf{w_i} \exp\left(-\mathbf{d}(\mathbf{x}^{'}, \mathbf{m}_\mathbf{i}^{'})/\mathbf{2}\right) \tag{6.30}$$

with

$$d(\mathbf{x}^{'}, \mathbf{m}_\mathbf{i}^{'}) = (\mathbf{x}^{'} - \mathbf{m}_\mathbf{i}^{'})^T \mathbf{C}_\mathbf{i}^{-\mathbf{1}} (\mathbf{x}^{'} - \mathbf{m}_\mathbf{i}^{'}). \tag{6.31}$$

Here, N is the number of neurons in the second hidden layer, $\mathbf{x}$ is the n-dimensional input pattern vector, $\mathbf{x}^{'}$ is the transformed input pattern vector, $\mathbf{m_i^{'}}$ is the center of a node, w_i are the output weights, and $\mathbf{y}$ is the m-dimensional output of the network. The $n \times n$ covariance matrix $\mathbf{C_i}$ is of the form

$$C_{jk}^{i} = \begin{cases} \frac{1}{\sigma_{jk}^2} & \text{if} \quad m = n \\ 0 & \text{otherwise} \end{cases} \tag{6.32}$$

where σ_{jk} is the standard deviation. Because the centers of the Gaussian potential function units (GPFU) are defined in the feature space, they will be subject to transformation by $\mathbf{B}$ as well. Therefore, the exponent of a GPFU can be rewritten as

$$d(\mathbf{x}, \mathbf{m_i^{'}}) = (\mathbf{x} - \mathbf{m_i})^{\mathbf{T}} \mathbf{B}^T \mathbf{C_i^{-1}} \mathbf{B} (\mathbf{x} - \mathbf{m_i}) \tag{6.33}$$

and is in this form similar to equation (6.31).

For the moment, we will regard $\mathbf{B}$ as the identity matrix. The network models the distribution of input vectors in the feature space by the weighted summation of Gaussian normal distributions, which are provided by the GPFU Ψ_j. To measure the difference between these distributions, we define the relevance ρ_n for each feature x_n:

$$\rho_n = \frac{1}{PJ} \sum_p \sum_j \frac{(x_{pn} - m_{jn})^2}{2\sigma_{jn}^2} \tag{6.34}$$

where P is the size of the training set and J is the number of the GPFUs. If ρ_n falls below the threshold ρ_{th}, one will decide to discard feature x_n. This criterion will not identify every irrelevant feature. If two features are correlated, one of them will be irrelevant, but this cannot be indicated by the criterion.

Learning paradigm for the transformation radial-basis neural network

We follow [151] for the implementation of the neuron allocation and learning rules for the TRBNN. The network generation process starts without any neuron.

The mutual dependency of correlated features can often be approximated by a linear function, which means that a linear transformation

of the input space can render features irrelevant.

First we assume that layers 3 and 4 have been trained so that they comprise a model of the pattern-generating process, and $\mathbf{B}$ is the identity matrix. Then the coefficients B_{nr} can be adapted by gradient descent with the relevance $\rho_n^{'}$ of the transformed feature $x_n^{'}$ as the target function. Modifying B_{nr} means changing the relevance of x_n by adding x_r to it with some weight B_{nr}. This can be done online, that is, for every training vector $\mathbf{x_p}$, without storing the whole training set. The diagonal elements B_{nn} are constrained to be constant 1, because a feature must not be rendered irrelevant by scaling itself. This in turn guarantees that no information will be lost. B_{nr} will be adapted only under the condition that $\rho_n < \rho_p$, so that the relevance of a feature can be decreased only by some more relevant feature. The coefficients are adapted by the learning rule:

$$B_{nr}^{new} = B_{nr}^{old} - \mu \frac{\partial \rho_n}{\partial B_{nr}} \tag{6.35}$$

with the learning rate μ and the partial derivative

$$\frac{\partial \rho_n}{\partial B_{nr}} = \frac{1}{PJ} \sum_p \sum_j \frac{(x_{pn}^{'} - m_{jn}^{'})}{\sigma_{jn}^2} (x_{pr}^{'} - m_{jr}^{'}). \tag{6.36}$$

In the learning procedure, which is based on, for example, [151], we minimize, according to the LMS criterion, the target function

$$E = \frac{1}{2} \sum_{p=0}^{P} |y(\mathbf{x}) - \Phi(\mathbf{x})|^2. \tag{6.37}$$

where P is the size of the training set. The neural network has some useful features, such as automatic allocation of neurons, discarding of degenerated and inactive neurons, and variation of the learning rate depending on the number of allocated neurons.

The relevance of a feature is optimized by gradient descent:

$$\rho_i^{new} = \rho_i^{old} - \eta \frac{\partial E}{\partial \rho_i} \tag{6.38}$$

Based on the new introduced relevance measure and the change in the architecture, we get the following correction equations for the neural

network:

$$\begin{aligned}
\frac{\partial E}{\partial w_{ij}} &= -(y_i - \Phi_i)\Psi_j \\
\frac{\partial E}{\partial m_{jn}} &= -\sum_i (y_i - \Phi_i) w_{ij} \Psi_j \sum_k (x'_k - m'_{jk}) \frac{B_{kn}}{\sigma^2_{jk}} \\
\frac{\partial E}{\partial \sigma_{jn}} &= -\sum_i (y_i - \Phi_i) w_{ij} \Psi_j \frac{(x'_n - m'_{jn})^2}{\sigma^3_{jn}}.
\end{aligned} \tag{6.39}$$

In the transformed space the hyperellipses have the same orientation as in the original feature space. Hence they do not represent the same distribution as before. To overcome this problem, layers 3 and 4 will be adapted at the same time as $\mathbf{B}$. Converge these layers fast enough, and they can be adapted to represent the transformed training data, thus providing a model on which the adaptation of $\mathbf{B}$ can be based. The adaptation with two different target functions (E and ρ) may become unstable if $\mathbf{B}$ is adapted too fast, because layers 3 and 4 must follow the transformation of the input space. Thus μ must be chosen $\ll \eta$. A large gradient has been observed to cause instability when a feature of extreme high relevance is added to another. This effect can be avoided by dividing the learning rate by the relevance, that is, $\mu = \mu_0/\rho_r$.

6.6 Hopfield Neural Networks

An important concept in neural networks theory is dynamic recurrent neural systems. The Hopfield neural network implements the operation of auto associative (content-addressable) memory by connecting new input vectors with the corresponding reference vectors stored in the memory.

A pattern, in the parlance of an N-node *Hopfield neural network*, is an N-dimensional vector $\mathbf{p} = [p_1, p_2, \ldots, p_N]$ from the space $\mathbf{P} = \{-1, 1\}^N$. A special subset of $\mathbf{P}$ represents the set of stored or reference patterns $\mathbf{E} = \{\mathbf{e^k} : 1 \leq k \leq K\}$, where $\mathbf{e^k} = [e^k_1, e^k_2, \ldots, e^k_N]$. The Hopfield network associates a vector from $\mathbf{P}$ with a certain reference pattern in $\mathbf{E}$. The neural network partitions $\mathbf{P}$ into classes whose members are in some way similar to the stored pattern that represents the class. The Hopfield network finds a broad application area in image restoration and segmentation.

Like the other neural networks, the Hopfield network has the following four components:

Neurons: The Hopfield network has a finite set of neurons $\mathbf{x}(i), 1 \leq i \leq N$ which serve as processing units. Each neuron has a value (or state) at time t, described by $\mathbf{x}_t(i)$. A neuron in the Hopfield network has one of the two states, either -1 or +1; that is, $\mathbf{x}_t(i) \in \{-1, +1\}$.

Synaptic connections: The learned information of a neural network resides within the interconnections between its neurons. For each pair of neurons $\mathbf{x}(i)$ and $\mathbf{x}(j)$, there is a connection w_{ij}, called the synapse, between them. The design of the Hopfield network requires that $w_{ij} = w_{ji}$ and $w_{ii} = 0$. Figure 6.13a illustrates a three-node network.

Propagation rule: It defines how states and synapses influence the input of a neuron. The propagation rule $\tau_t(i)$ is defined by

$$\tau_t(i) = \sum_{j=1}^{N} \mathbf{x}_t(j) w_{ij} + b_i \tag{6.40}$$

b_i is the externally applied bias to the neuron.

Activation function: The activation function f determines the next state of the neuron $\mathbf{x}_{t+1}(i)$ based on the value $\tau_t(i)$ computed by the propagation rule and the current value $\mathbf{x}_t(i)$. Figure 6.13b illustrates this. The activation function for the Hopfield network, is the hard limiter defined here:

$$\mathbf{x}_{t+1}(i) = f(\tau_t(i), \mathbf{x}_t(i)) = \begin{cases} 1, & \text{if} \quad \tau_t(i) > 0 \\ -1, & \text{if} \quad \tau_t(i) < 0 \end{cases} \tag{6.41}$$

The network learns patterns that are N-dimensional vectors from the space $\mathbf{P} = \{-1, 1\}^N$. Let $\mathbf{e^k} = [e_1^k, e_2^k, \ldots, e_n^k]$ define the kth exemplar pattern where $1 \leq k \leq K$. The dimensionality of the pattern space is reflected in the number of nodes in the network, such that the latter will have N nodes $\mathbf{x}(1), \mathbf{x}(2), \ldots, \mathbf{x}(N)$.

The training algorithm of the Hopfield neural network is simple and outlined below.

1. **Learning:** Assign weights w_{ij} to the synaptic connections:

$$w_{ij} = \begin{cases} \sum_{k=1}^{K} e_i^k e_j^k, & \text{if} \quad i \neq j \\ 0, & \text{if} \quad i = j \end{cases} \tag{6.42}$$

Keep in mind that $w_{ij} = w_{ji}$, so it is necessary to perform the preceding

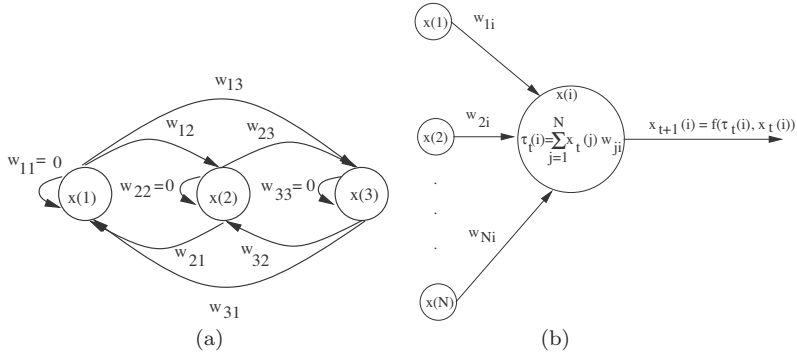


Figure 6.13
(a) Hopfield neural network; (b) propagation rule and activation function for the Hopfield network.

computation only for $i < j$.

2. **Initialization:** Draw an unknown pattern. The pattern to be learned is now presented to the network. If $\mathbf{p} = [p_1, p_2, \ldots, p_N]$ is the unknown pattern, write

$$\mathbf{x}_0(i) = p_i, \quad 1 \leq i \leq N \tag{6.43}$$

3. **Adaptation:** Iterate until convergence. Using the propagation rule and the activation function for the next state we get

$$\mathbf{x}_{t+1}(i) = f\left(\sum_{j=1}^{N} \mathbf{x}_t(j) w_{ij}, \mathbf{x}_t(i)\right). \tag{6.44}$$

This process should be continued until any further iteration will produce no state change at any node.

4. **Continuation:** For learning a new pattern, repeat steps 2 and 3.

There are two types of Hopfield neural networks: binary and continuous. The differences between the two of them are shown in table 6.2.

In dynamic systems parlance, the input vectors describe an arbitrary initial state, and the reference vectors describe attractors or stable states. The input patterns cannot leave a region around an attractor, which is called the basin of attraction.

Table 6.2
Comparisons between binary and continuous Hopfield neural networks

Network type	Binary	Continuous-valued
Updating	Asynchronous	Continuous
Neuron function	Hard limiter	Sigmoid function
Description	Update only one random neuron's output	Update continuously and and simultaneously all neurons' outputs

The network's dynamics minimizes an energy function, and those attractors represent possible local energy minima. Additionally, these networks are able to process noise-corrupted patterns, a feature that is relevant for performing the important task of content-addressable memory.

The convergence property of Hopfield's network depends on the structure of $\mathbf{W}$ (the matrix with elements w_{ij}) and the updating mode. An important property of the Hopfield model is that if it operates in a sequential mode and $\mathbf{W}$ is symmetric with non negative diagonal elements, then the energy function

$$\begin{aligned} E_{hs}(t) &= \tfrac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}x_i(t)x_j(t) - \sum_{i=1}^{n} b_i x_i(t) \\ &= -\tfrac{1}{2}\mathbf{x}^T(t)\mathbf{W}\mathbf{x}(t) - \mathbf{b}^T\mathbf{x}(t) \end{aligned} \tag{6.45}$$

is nonincreasing [117]. The network always converges to a fixed point.

Hopfield neural networks are applied to solve many optimization problems. In medical image processing, they are applied in the continuous mode to image restoration, and in the binary mode to image segmentation and boundary detection.

6.7 Performance Evaluation of Clustering Techniques

Determining the optimal number of clusters is one of the most crucial classification problems. This task is known as cluster validity. The chosen validity function enables the validation of an accurate structural representation of the partition obtained by a clustering method. While a visual visualization of the validity is relatively simple for two-dimensional data, in the case of multidimensional data sets this becomes very tedious.

In this sense, the main objective of cluster validity is to determine the optimal number of clusters that provide the best characterization of a given multidimensional data set. An incorrect assignment of values to the parameter of a clustering algorithm results in a data-partitioning scheme that is not optimal, and thus leads to wrong decisions.

In this section, we evaluate the performance of the clustering techniques in conjunction with three cluster validity indices: Kim's index, the Calinski-Harabasz (CH) index, and the intraclass index. These indices were successfully applied earlier in biomedical time-series analysis [97]. In the following, we describe the above-mentioned indices.

Calinski-Harabasz index: [39]: This index is computed for m data points and K clusters as

$$CH = \frac{[\text{trace}B/(K-1)]}{[\text{trace}W/(m-K)]} \tag{6.46}$$

where B and W represent the between- and within-cluster scatter matrices.

The maximum hierarchy level is used to indicate the correct number of partitions in the data.

Intraclass index [97]: This index is given as

$$I_W = \frac{1}{n}\sum_{k=1}^{K}\sum_{i=1}^{n_k} ||\mathbf{x}_i - \mathbf{w}_k||^2 \tag{6.47}$$

where n_k is the number of points in cluster k and $\mathbf{w}_k$ is a prototype associated with the kth cluster. I_W is computed for different cluster numbers. The maximum value of the second derivative of I_W as a function of cluster number is taken as an estimate for the optimal partition. This index provides a possible way of assessing the quality of a partition of K clusters.

Kim's index [138]: This index equals the sum of the overpartition $v_o(K, \mathbf{X}, \mathbf{W})$, and the underpartition $v_u(K, \mathbf{X}, \mathbf{W})$ function measure

$$I_{Kim} = \frac{v_u(K) - v_{umin}}{v_{umax} - v_{umin}} + \frac{v_o(K) - v_{omin}}{v_{omax} - v_{omin}}. \tag{6.48}$$

where $v_u(K)$ is the underpartitioned average over the cluster number of the mean intracluster distance, and measures the structural compactness

of each class, v_{umin} is its minimum and v_{umax} is the maximum value. $v_u(K, \mathbf{X}, \mathbf{W})$ is given by the average of the mean intracluster distance over the cluster number K, and measures the structural compactness of each and every class. $v_o(K, \mathbf{X}, \mathbf{W})$ is given by the ratio between the cluster number K and the minimum distance between cluster centers, describing intercluster separation. $\mathbf{X}$ is the matrix of the data points and $\mathbf{W}$ is the matrix of the prototype vectors. Similarly, $v_o(K)$ is the overpartitioned measure defined as the ratio between the cluster number and the minimum distance between cluster centers that measures the intercluster separation. v_{omin} is its minimum and v_{omax} is the maximum value. The goal is to find the optimal cluster number with the smallest value of I_{Kim} for a cluster number $K = 2$ to K_{max}.

6.8 Classifier Evaluation Techniques

The evaluation of the classification accuracy of the pattern recognition paradigms and the comparisons among them are accomplished based on well-known tools such as the confusion matrix, the ranking order curves, and ROC curves.

Confusion matrix

For a classification system, it's important to determine the percentage of correctly and incorrectly classified data.

A convenient visualization tool when analyzing results in an error-prone classification system in general is the *confusion matrix*, which is a two–dimensional matrix containing information about the actual and predicted classes. The dimension of the matrix corresponds to the number of classes. Entries on the diagonal of the matrix are the correct classes and those off–diagonal are the misclassifications. The columns are the actual classes and the rows are the predicted classes. The ideal error–free classification case is a diagonal confusion matrix. Table 6.3 shows a sample confusion matrix. The confusion matrix allows us to keep track of all possible outcomes of a classification process. In summary, each element of the confusion matrix indicates the chances that the row element is confused with the column element.

Table 6.3
Confusion matrix for a classification of three classes: A_1, A_2, A_3.

	Output		
Input	A_1	A_2	A_3
A_1	92%	3%	5%
A_2	0%	94%	6%
A_3	12%	88%	0%

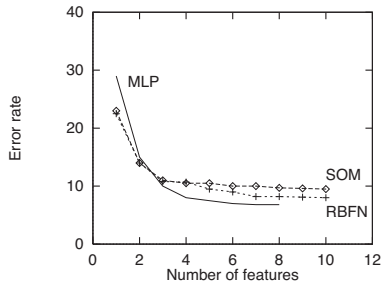


Figure 6.14
Example of ranking order curves showing feature selection results using three different classifiers (MLP, SOM, RBFN).

Ranking order curves

Ranking order curves are a useful method that provides a feature set that can be used to train a classifier to have very good generalization capability.

The importance of the set of the most relevant features is well-known in pattern recognition. In general, by adding additional features, we may improve the classification performance. However, we observe that after considering additional features, this may deteriorate or lead to overtraining. This situation varies across the different types of classifiers. To avoid this problem, several simulations are required to determine the optimal feature set. As a result, the ranking order curves provide a clear picture of the feature dependence and, at the same time, a comparison of the classification performance of different classifiers. Figure 6.14 visualizes three feature ranking order curves for supervised, unsupervised, and hybrid classifiers.

Table 6.4
Results of a test in two populations, one of them with a disease.

	Disease present	Disease absent	Sum
Test positive	true positive (TP)	false positive (FP)	(TP + FP)
Test negative	false negative (FN)	true positive (TN)	(FN + TN)
Sum	(TP + FN)	(FP + RN)	

6.9 Diagnostic Accuracy of Classification Measured by ROC curves

Receiver operating characteristics (ROC) curves were discovered in connection with signal detection theory, as a graphical plot to discriminate between hits and false alarms. It is a graphical representation of the false positive (false alarm) rate versus the true positive rate that is plotted while a threshold parameter is varied.

Recently, ROC analysis has become an important tool in medical decision-making by enabling the discrimination of diseased cases from normal cases [172]. For example, in cancer research, the false positive (FP) rate represents the probability of incorrectly classifying a normal tissue region as a tumor region. On the other hand, the true positive (TP) rate gives the probability of correctly classifying a tumor region as such. Both the TP and the FP rates take values on the interval from 0.0 to 1.0, inclusive. In medical imaging the TP rate is commonly referred to as *sensitivity*, and (1.0 - FP rate) is called *specificity*.

The schematic outcome of a particular test in two populations, one with a disease and the other without the disease, is summarized in table 6.4.

In the following it is shown how ROC curves are generated given the two pdfs of healthy and tumor tissue [287]. A decision threshold T is set, such that if the ratio is larger than T, the unknown outcome is classified as abnormal, otherwise as normal. By changing T, the sensitivity/specificity trade–off of the test can be altered. A larger T will result in lower TP and FP rates, while a smaller T will result in higher TP and FP rates. The procedure described in [287] is illustrated in figure 6.15.

The sensitivity is a performance measure of how well a test can

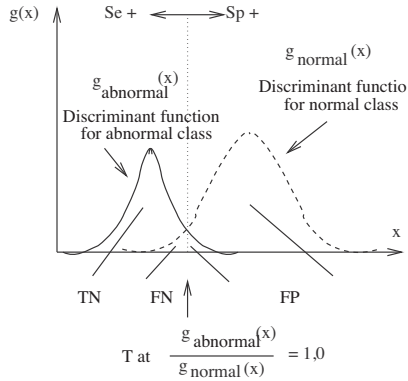


Figure 6.15
Discriminant functions for two populations, one with a disease and the other without the disease. A perfect separation between the two groups is rarely given; an overlap is mostly observed. The FN, FP, TP, and TN areas are indicated.

determine the patients with disease, and the specificity shows the ability of the test to determine the patients who do NOT have the disease.

In general, the sensitivity S_e and the specificity S_p of a particular test can be mathematically determined.

Sensitivity S_e reveals that the test result will be positive when disease is present (true positive rate, expressed as a percentage):

$$S_e = \frac{TP}{FN + TP} \tag{6.49}$$

Specificity S_p is the probability that a test result will be negative when the disease is not present (true negative rate, expressed as a percentage):

$$S_p = \frac{TN}{TN + FP} \tag{6.50}$$

Sensitivity and specificity are functions of each other and also counterrelated. The x-axis describes the specificity and the ROC curve expresses 1-specificity. Thus, the x and y coordinates are given as

$$x = 1 - \frac{TN}{TN + FP} \tag{6.51}$$

$$y = \frac{TP}{FN + TP} \tag{6.52}$$

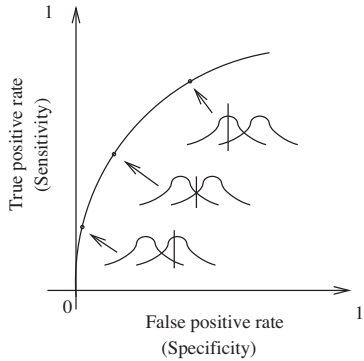


Figure 6.16
Typical ROC curve.

Another important parameter in connection with ROC curves is the discriminability index d', which captures both the separation and the spread of the disease and disease-free curves. Thus, it's an estimate of the signal strength and does not depend on interpretation criteria and is therefore a measure of the internal response. The discriminability index d' is defined as

$$d' = \frac{\text{separation}}{\text{spread}} \tag{6.53}$$

For $d' = 0$, we have the 45° diagonal line.

A typical ROC curve is shown in figure 6.16. High values of sensitivity and specificity (i.e., high y-axis values at low x-axis values) demonstrate a good classification result. The area under the curve (AUC) is an accepted modality of comparing classifier performance, where an area of 1.0 signifies near perfect accuracy, and an area of less than 0.5 indicates random guessing.

A given classifier has a flexibility, in terms of chosen parameter values, to change the FP and TP rates and to determine a different operating point (TP, FP pair). Furthermore, it may thus obtain a lower (higher) FP rate at the expense of a higher (lower) TP detection.

Another important aspect in the context of ROC curves is the degree of overlapping between the two pdfs. The more they overlap, the smaller the AUC becomes. When the overlap is complete, the resulting

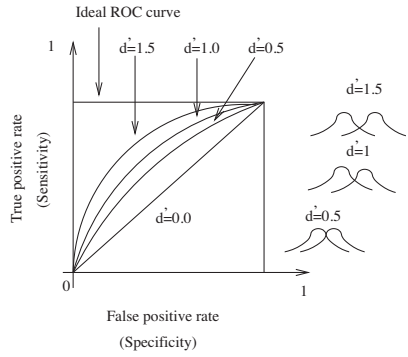


Figure 6.17
ROC curves for different discriminability index d'. When the overlap is minimal and d' is large, the ROC curve becomes more bowed.

ROC curve becomes a diagonal line connecting the points (0,0) and (1,1). Figure 6.17 illustrates the dependence of the ROC curve on the discriminability index d'. ROC curves for a higher d' (not much overlap) bow out further than ROC curves for lower d' (lots of overlap).

An ROC curve for a given two-group population problem (disease/non-disease) is easily plotted based on the following steps:

1. We run a test for the disease and rank the test results in order of increasing magnitude. We start at the origin of the axis where both false positive and true positive are zero.
2. We set the threshold just below the largest result. If this first result belongs to a patient with the disease, we obtain a true positive and read from the overlapping pdfs the values of the true positive and false negative, and plot the first point of the ROC curve.
3. We lower the threshold just below the second largest result and repeat step 2.
4. We continue this process until we have moved the threshold below the lowest value.

In summary, this procedure is very simple: the ranked values are labeled as either true or false positive and then the curve is constructed. The main requirement in connection with ROC curves is that the values have to be ranked.

Some important aspects in the context of the ROC curve are of

special interest:

- In the parlance of pattern recognition, it shows the performance of a classifier as a trade-off between selectivity and sensitivity.
- The curve always connects the two coordinates $(0, 0)$ (finds no positives) and $(1, 1)$ (finds no negatives), and for the perfect classifier has an AUC=1.
- The area under the ROC curve is similar to the Mann-Whitney statistics.
- In the context of ROC curves we speak of the "gold standard" which confirms the absence or presence of a disease.
- In the specific case of randomly paired normal and abnormal radiological images, the area under the ROC curve represents a measure of the probability that the perceived abnormality of the two images will allow correct identification.
- Similar AUC values do not prove that ROC curves are also similar. Deciding if similar AUC values belong to similar ROC curves requires the application of bivariate statistical analysis.

6.10 Example: Adaptive Signal Analysis of Immunological Data

This section aims to illustrate how both supervised and unsupervised signal analysis can contribute to the interpretation of immunological data. For this purpose a data base was set up containing cellular data from bronchoalveolar lavage fluid which was obtained from 37 children with pulmonary diseases. The children were dichotomized into two groups: 20 children suffered from chronic bronchitis and 17 children had an interstitial lung disease. A self-organizing map (SOM) (see section 6.3) and linear independent component analysis were utilized to test higher-order correlations between cellular subsets and the patient groups. Furthermore, a supervised approach with a perceptron trained to the patients' diagnosis was applied. The SOM confirmed the results that were expected from previous statistical analyses. The results of the ICA were rather weak, presumably because a linear mixing model of independent sources does not hold; nevertheless, we could find parameters of high diagnosis influence that were confirmed by the perceptron. The super-

vised perceptron learning after principal component analysis for dimension reduction turned out to be highly successful by linearly separating the patients into two groups with different diagnoses. The simplicity of the perceptron made it easy to extract diagnosis rules, which partly were known already and could now readily be tested on larger data sets.

The neural network signal analysis of this immunological data set has been performed in [257] and extended using ICA in [256].

Medical background

Immunological approaches have gained increasing importance in modern biochemical research. Within the last few years a broad array of sophisticated experimental tools has been developed, and ultimately has led to the generation of an immense quantity of new and complex information. Since the interpretation of these results is often not trivial, there is a need for novel data analysis instruments that allow evaluation of large databases. For this purpose three different algorithms were applied to immunological data that were generated as outlined below.

In inflammatory airway diseases, lymphocytes accumulate in the pulmonary tissue. Since the lung is perfused by two different arterial systems that feed the bronchi and the alveoli, lymphocytes can enter the pulmonary tissue by two separate vascular routes. Therefore, a selective recruitment of distinct effector T cells into the two pulmonary compartments may occur. Controlled trafficking of T cells to peripheral sites occurs through adhesion molecules and the interaction of chemokines with their counterpart receptors. Accordingly, a number of chemokine receptors are differentially expressed on lymphocytes in an organ- or disease-specific manner [92]. Chemokines are classified into four families (CC, CXC, CX3, C) based on the positioning of amino acids between the two N-terminal cysteine residues (see also [224]). CX3- and C-chemokines are each represented by single members, whereas the other two groups have multiple members. While the group of CXC-chemokines acts preferentially on neutrophils, the CC-chemokine group is mainly involved in the attraction of lymphocytes [224]. However, these distinctions are not absolute.

To test whether a selective recruitment of T cells into the lung occurs, 37 children suffering from various pulmonary diseases were selected for the study. Based on clinical and radiological findings, the children were further subdivided into two groups which mirrored the two pul-

monary compartments. Seventeen children (f=10; mean age 5.3 years; range 0.3-17.3 years) had *chronic bronchitis (CB)*. Twenty children (f=7; mean age 6.8 years, range 2 months - 18.8 years) had *interstitial lung diseases (ILD)*. In all children a bronchoalveolar lavage was performed for diagnostic and/or therapeutic indications. Cells were obtained from bronchoalveolar lavage fluid (BALF), and the frequency of lymphocytes expressing different chemokine receptors (CXCR3+, CCR5+, CCR4+, and CCR3+) which control lymphocyte migration was analyzed by four-color flow cytometry on CD4+ and CD8+ T cell subsets. To evaluate the contribution of the corresponding chemokines to the local effector cell recruitment, the ligands for CXCR3 and CCR5, termed IP-10 (Interferon-γ inducible Protein of 10 kDa), and RANTES (Regulated upon Activation Normal T cell Expressed and Secreted) were quantified in BALF with a commercial enzyme-linked immunosorbent assay (R&D Systems, Minneapolis, Minnesota USA).

Signal analysis

We analyzed the following parameters in BALF (visualization in figure 6.18): RANTES relative to the cell number in BALF (RANTESZZ), IP-10, CD4+ T cells, CD8+ T cells, the ratio of CD4+ to CD8+ T cells (CD4/CD8), CD19+ B cells, CCR5+CD4+ cells, CXCR3+CD4+ cells, CXCR3+CD8+ cells, macrophages (M), lymphocytes (L), neutrophile granulocytes (NG), eosinophile granulocytes (EG), the total cell count in BALF (ZZ), systemic corticosteroid therapy (CORTISONE), and C-reactive protein (CRP).

Altogether, we had a data set of 30 parameters; however, some parameters were missing for some of the patients. In the following we will use preselected subsets of these parameters as specified in the corresponding section.

Self-organizing maps

SOMs approximate nonlinear statistical relationships between high-dimensional data items by easier geometric relationships on a low-dimensional display. They also perform abstraction by reducing the information while preserving the most important topological and metric relationships of the primary data. These two aspects, visualization and abstraction, can be utilized in a number of ways in complex tasks such

as process analysis, machine perception, control, and communication.

In the following we will use SOMs as unsupervised analysis tools mainly to visualize the complex data set from above and to find clusters in the data set which might belong to separate diagnoses.

Results

Calculations were performed on a P4-2000 PC with Windows and Matlab, using the "SOM Toolbox" from the Helsinki group[1]. In figure 6.18, we show a SOM generated on the described data set.

The information obtained from the visualized data agreed with previous statistical analyses [108]. The parameter ZZ showed distinct clusters on map units which represented samples of patients with ILD; a weaker clustering was observed for RANTESZZ and CRP. Patients with CB were characterized by map unit clusters of CD8 and CXCR3C+D8. Furthermore, the SOM indicated relationships between immunological parameters and patient groups which had not been identified by conventional statistical approaches. NG showed a positive relationship to CRP on map units which represented a subgroup of ILD samples (correlation 0.32 after normalization). M were predominately clustered on map units of CB samples. Interestingly, the SOM separated three ILD samples on map units from the ILD main cluster. These ILD samples showed distinct parameter characteristics in comparison to the ILD main cluster group, both a higher density on the cluster map and agreater neighborhood correlation than the other ILDs. The parameters CD4, CD4/CD8, CD19, CR5CD4, and CX3CD4 showed a clear relationship (correlations with respect to CD4 of CD4/CD8, CD19, CR5CD4, and CX3CD4 are 0.76, 0.47, 0.86, and 0.67). This is not surprising because these are parameter subgroups of cells from the same group, so they must correlate.

Independent component analysis

Algorithm

Principal component analysis (PCA), also called the Karhunen-Loève transformation, is one of the most common multivariate data analysis tools based on early works of Pearson [198]. PCA is a well-known technique often used for data preprocessing in order to whiten the data

1 Available online at http://www.cis.hut.fi/projects/somtoolbox/.

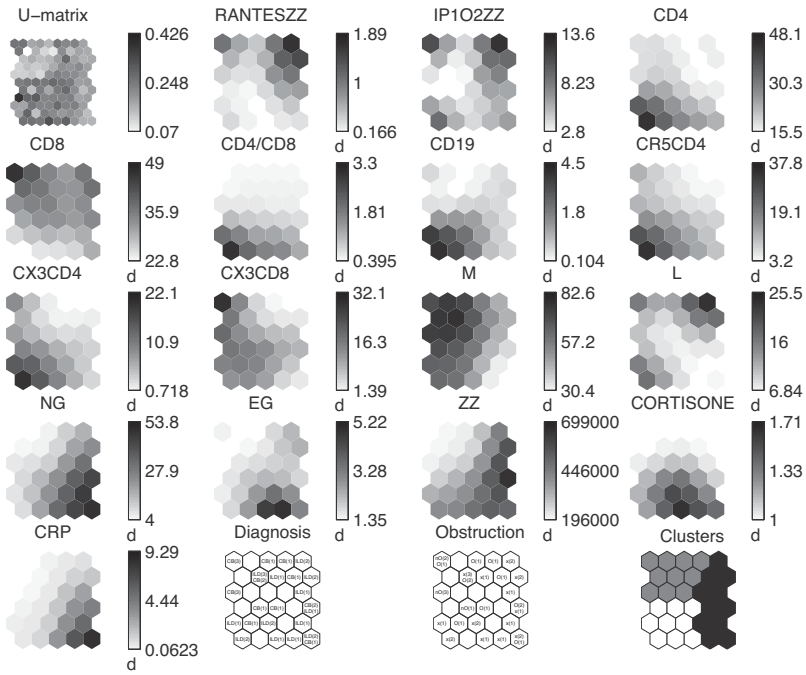


Figure 6.18
Self-organizing map generated on the 16-dimensional immunology data set. In addition, the upper left image gives a visualization of the distance matrix between hexagons – darker areas are larger distances – and the lower two images with labels show how diagnosis and obstruction of each patient are mapped onto the 2-dimensional grid. The bottom right figure shows a plot of k-means clustering applied to the distance matrix using 3 clusters.

and reduce its dimensionality, see chapter 3.

Given a random vector, the goal of ICA is to find its statistically independent components, see chapter 4. This can be used to solve the blind source separation (BSS) problem, which is, given only the mixtures of some underlying independent sources, to separate the mixed signals and thus recover the original sources. In contrast to correlation-based transformations such as PCA, ICA renders the output signals as statistically independent as possible by evaluating higher-order statistics. The idea of ICA was first expressed by Herault and Jutten [112] [127] and the term ICA was later coined by Comon [59]. However, the field became popular only with the seminal paper by Bell and Sejnowski [25],

who elaborated upon the Infomax principle first advocated by Linsker [157] [158].

In the calculations we used the well-known and well-studied FastICA algorithm [124] of Hyvärinen and Oja, which separates the signals using negentropy, and therefore non-Gaussianity, as a measure of the separation signal quality.

Results

We used only 29 of the 39 samples because the number of missing parameters was too high in the other samples. As preprocessing, we applied PCA in order to whiten the data and to project the 16-dimensional data vector to the five dimensions of highest eigenvalues.

Figure 6.19 gives a plot of the linearly separated signals together with the comparison patient diagnosis - the first 14 samples were CB (diagnosis 0) and the last 15 were ILD (diagnosis 1). Since we were trying to associate immunological parameters with a given diagnosis in our data set, we calculated the correlation of the separated signals with this diagnosis signal. In figure 6.18, the signal with the highest diagnosis correlation is signal 5, with a correlation of 0.43 (which is still quite low).

The rows of the inverse mixing matrix contain the information on how to construct the corresponding independent components from the sample data. After normalization to unit signal variance, ICA signal 5 is constructed by multiplication of

$$\hat{\mathbf{w}}^{\top} = 10^4(\begin{array}{lllll} -9.5 & -10.1 & 1.6 & -10.1 & 4.7 \\ 0.40 & -1.6 & -8.5 & -21 & 3.6 \\ -6.2 & -1.8 & 3.5 & 0 & -0.037 \\ 3.6 &) & & & \end{array}$$

with the signal data. We see that parameter 1 (RANTES), parameter 2 (IP10), parameter 4 (CD8), parameter 8 (CXCR3CD4), and parameter 9 (CXCR3+CD8) are those with the highest absolute values. This indicates that those parameters have the greatest influence on the classification of the patients into one of the two diagnostic groups. The perceptron learning results from the next section will confirm that high values of RANTESZZ (which is positively correlated with RANTES related to lymphocytes in BALF (RANBALLY), which is analyzed using the neural network) and CX3CD8 are indicators for CB; of course this

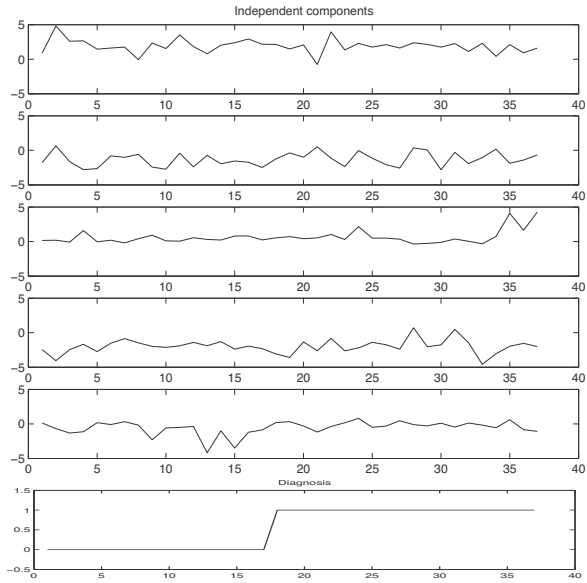


Figure 6.19
ICA components using FastICA with symmetric approach and pow3-nonlinearity after whitening and PCA dimension reduction to 5 dimensions. Below the components, the diagnoses (0 or 1) of the patients are plotted for comparison. The covariances of each signal with the diagnoses are −0.16, 0.27, 0.25, 0.04 and 0.43, and visual comparison already confirms bad correspondence of one of the ICs with the diagnosis signal.

holds true only with other values being small.

All in all, however, we note that the linear ICA model applied to the given immunology data did not hold very well when trying to find diagnosis patterns. Of course we did not have such nice linear models as EEG data; altogether, not many medical models describing connections of these immunology parameters have been found. Therefore we will try to model the parameter-diagnosis relationship using supervised learning in the next section.

Neural network learning

Having used the two unsupervised learning algorithms from above, we now use supervised learning in order to approximate the parameter-diagnosis function. We will show that the measured parameters are

indeed sufficient to determine the patient diagnosis quite well.

Algorithm

Supervised learning algorithms try to approximate a given function $\mathbf{f} : \mathbb{R}^n \to A \subset \mathbb{R}^m$ by using a number of given sample-observation pairs $(\mathbf{x}_\lambda, \mathbf{f}(\mathbf{x}_\lambda)) \in \mathbb{R}^n \times A$. If A is finite, we speak of a classification problem. Typical examples of supervised learning algorithms are polynomial and spline interpolation or artificial neural network (ANN) learning. In many practical situations, ANNs have the advantage of higher generalization capability than other approximation algorithms, especially when only few samples are available.

McCulloch and Pitts [167] were the first to describe the abstract concept of an artificial neuron base on the biological picture of a real neuron. A single neuron takes a number of input signals, sums these and plugs the result into a specific activation function (for example a (translated) Heaviside function or an arc tangent). The *neural network* itself consists of a directed graph with an edge labeling of real numbers called weights. At each graph node we have a neuron that takes the weighted input and transmits it to all following neurons. Using ANNs has the advantage that in neural networks, which are adaptive systems, we know for a given energy function how to algorithmically minimize this function (for example, using the standard accelerated gradient descent method). When trying to learn the function $\mathbf{f}$, we use as the energy function the summed square error $\sum_\lambda |\mathbf{f}(\mathbf{x}_\lambda) - \mathbf{y}(\mathbf{x}_\lambda)|^2$, where $\mathbf{y}$ denotes the neural network output function. Moreover, more general functions can then be approximately learned using the fact that sufficiently complex neural networks are so called universal approximators [119]. For more details about ANNs, see some of the many available textbooks (e.g. [9] [110] [113]).

We will restrict ourselves to feed forward layered neural networks. Furthermore, we found that simple single-layered neural networks (perceptrons) already sufficed to learn the diagnosis data well. In addition, they have the advantage of easier rule extraction and interpretation.

A *perceptron* with output dimension 1 consists of only a single neuron, so the output function y can be written as

$$y(\mathbf{x}) = \theta(\mathbf{w}^\top \mathbf{x} + w_0)$$

with weight $\mathbf{w} \in \mathbb{R}^n$, n input dimension, $w_0 \in \mathbb{R}$ the bias, and as activation function θ, the Heaviside function ($\theta(x) = 0$ for $x < 0$ and $\theta(x) = 1$ for $x \geq 0$). Often, the bias w_0 is added as additional weight to $\mathbf{w}$ with fixed input 1.

Learning in a perceptron means minimizing the error energy function shown above. This can be done, for example, by gradient descent with respect to $\mathbf{w}$ and w_0. This induces the well-known *delta rule* for the weight update,

$$\Delta\mathbf{w} = \eta(y(\mathbf{x}) - t)^\top \mathbf{x},$$

where η is a chosen learning rate parameter, $y(\mathbf{x})$ is the output of the neural network at sample $\mathbf{x}$, and t is the observation of input $\mathbf{x}$. It is easy to see that a perceptron separates the data linearly, with the boundary hyperplane given by $\{\mathbf{x} \in \mathbb{R}^n | \mathbf{w}^\top \mathbf{x} + w_0 = 0\}$.

Results

We wanted to approximate the diagnosis function $\bar{d} : \mathbb{R}^{30} \to \{0, 1\}$ that classifies each parameter set to one of the two diagnoses. It turned out that we achieved best results in terms of approximation quality by using the 13-dimensional column subset with parameters RANTESRO, RANTESZZ, RANBALLY, IP101RO, IP101ZZ, IP102RO, IP1O2ZZ, CD8, CD4/CD8, CX3CD8, NG, ZZ and CORTISONE, as explained earlier in this section. The diagnosis of each patient in this sample set was known; so we really wanted to approximate the now 13-dimensional diagnosis function $d : \mathbb{R}^{13} \to \{0, 1\}$.

We had to omit 10 of the original 39 samples because too many parameters of those samples were missing. Of the remaining 29 samples, one parameter of one sample was unknown, so we replaced it with the mean value of this parameter of the other samples.

After centering the data, further preprocessing was performed by applying a PCA to the 13-D data set in order to normalize and whiten the data and to reduce their dimension. With only this small number of samples, learning in a 13-D neural network can easily result in very low generalization quality of the network. In figure 6.20, we give a plot of reduction dimension versus the output error of a perceptron trained with all 29 samples after reduction to the given dimension. We see that dimension reduction as low as five dimensions still yields quite good

results: only three samples were not correctly classified. Note that we use the same sample set for training and for testing the net; this is due to the fact of the low number of samples did not allow testing techniques like jackknifing or splitting the sample set into training and testing samples. Therefore, we also used a simple perceptron and not a more complex multi layered perceptron; its simple structure resulted in a linear separation of the given sample set.

The perceptron used had a Heaviside activation function and an additional bias for threshold shifting. We trained the network using 1000 epochs, although convergence was achieved after less than 50 epochs. We got a reconstruction error of only three samples.

The weight matrix of the learned perceptron converged to

$$\mathbf{w}^\top = (\begin{array}{lllll} 0.047 & -0.66 & -3.1 & 0.010 & -0.010 \\ 0.010 & 0.029 & -0.010 & 1.0 & -0.32 \\ -0.059 & < 10^4 & 4.1 &)^\top. & \end{array}$$

with bias $w_0 = -2.1$, where we had already multiplied $\mathbf{w}$ by the dewhitening PCA matrix. If we normalize the signals to unit variance, we get normalized weights

$$\hat{\mathbf{w}}^\top = (\begin{array}{lllll} 2.7 & -0.69 & -4.4 & 5.7 & -0.17 \\ 5.6 & 0.40 & -0.19 & 3.1 & -6.0 \\ -1.7 & 1.81.6 &)^\top & & \end{array}$$

and $\hat{w}_0 = 6.0$. These entries in $\hat{\mathbf{w}}$ can be used to detect parameters that have significant influence on the separation of the perceptron; these are mainly parameters 1 (RANTESRO), 3 (RANBALLY), 4 (IP101RO), 6 (IP102RO), 9 (CD4/CD8), 10 (CX3CD8). By setting the other parameters to zero, we constructed a new perceptron

$$\bar{\mathbf{w}}^\top = (\begin{array}{lllll} 0.047 & 0 & -3.2 & 0.010 & 0 \\ 0.010 & 0 & 0 & 1.04 & -0.32 \\ 0 & 0 & 0 &)^\top & \end{array}$$

and $\bar{w}_0 = -2.0$, again given for the non normalized source data. If we apply the data to this new reduced perceptron, we get a reconstruction error of five samples, which means that even this low number of parameters seems to distinguish the diagnosis quite well.

Further information can be obtained from the nets if we look at the sample classification without applying the signum function. We get

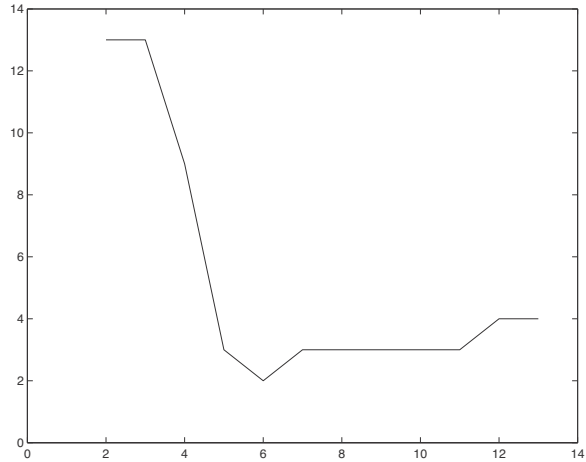


Figure 6.20
PCA Dimension reduction versus perceptron reconstruction error.

values for the original network $\mathbf{w}, w_0$, as shown in figure 6.21.

The single-layer neural network was trained with our measured immunological parameters to reveal the diagnoses of our patients. Since the bearing and the interdependencies of our measured parameters are not fully understood, it is difficult to ascribe importance to certain parameters. Six measured parameters were found to be essential for the ANN learning process to assign the diagnosis CB or ILD to the individual data samples.

A point of interest is the distances of the patient samples from the ANN separation boundary line (figure 6.21). The ANN showed three outliers in the assignment of the samples to the diagnoses CB and ILD, leading to wrong diagnosis assignments. Under these three outliers, two turned out to be CB patients with bronchial asthma, representing a distinct subgroup of the CB patient group. Two CB patients had the greatest distance to the separation boundary; those were identified as patients with a severe clinical course of CB. Similarly, three patients with ILD showed a distinct separation distance. These patients were identified as those with a severe course of the disease. Thus the ANN showed a graduated discrimination specificity for the diagnoses CB and ILD.

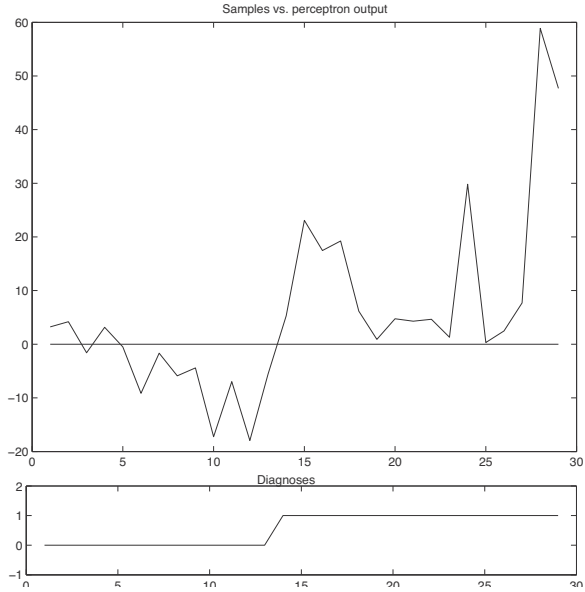


Figure 6.21
The upper figure shows a plot of the sample number versus the perceptron output $\mathbf{w}^\top \mathbf{x} + w_0$. Samples 1, 2, and 4 are not correctly classified (should be below zero). For comparison, the lower figure shows a plot of the correct diagnosis of the sample.

Discussion

We applied supervised and unsupervised signal analysis methods to study lymphocyte subsets in BALF of children with different pulmonary diseases. The self-organizing map read outs matched very well with the results of perviously performed statistical analyses. Therefore, the SOM clusters confirmed the expected differences in the frequency of distinct lymphocyte subsets in both patient groups. In addition, the SOM revealed possible relationships of immunological parameters, which were not identified by conventional non parametric statistical methods. Since the number of samples used for this analysis was limited, generalizations cannot be made at this point. However, the analysis of larger sample numbers will further help to evaluate the importance of SOM and advanced clustering methods in the description of immunological contiguities.

With a linear separation, the perceptron learned a diagnosis differentiation in 90% of the analyzed samples. The network showed a graduated discrimination specificity for the diagnoses CB and ILD. The application of the ANN to a larger number of samples and higher-dimensional data sets, could prove the benefit of this artificial intelligence tool.

In conclusion, the combination of these artificial intelligence approaches could be a very helpful tool to facilitate diagnosis assignment from immunological patient data where no diagnosis can be given or the discrimination between diagnoses is difficult.

6.11 Overview of Statistical, Syntactic, and Neural Pattern Recognition

The artificial neural networks techniques are an important part of the field of pattern recognition. In general, there are many classification paradigms which lead to a reasonable solution of a classification problem: syntactic, statistical, or neural. The delimitations between statistical, syntactic and neural pattern recognition approaches are fuzzy since all share common features and are geared toward obtaining a correct classification result.

The decision to choose a particular approach over another is based on analysis of underlying statistical components, or grammatical structure, or on the suitability of a neural network solution [173].

Table 6.5 and figure 6.22 elucidate the similarities and differences between the three pattern recognition approaches [227].

Both neural and statistical classification techniques require that the information be given as a numerical-valued feature vector. In some cases, information is available as a structural relation between the components of a vector. The important aspect of structural information forms the basis of the structural and syntactic classification concepts. Thus, structural pattern recognition can be employed for both classification and description.

Each method has its strengths, but at the same time there are also some drawbacks: the statistical method does not operate with syntactic information; the syntactic method does not operate based on adaptive learning rules; and the neural network approach does not contain any semantic information in its architecture [173].

Table 6.5
Comparing statistical, Syntactical and neural pattern recognition approaches.

	Statistical	**Syntactic**	**Neural**
Pattern Generation Basis	Probabilistic Models	Formal Grammars	Stable State or Weight Matrix
Pattern Classification Basis	Estimation or Decision Theory	Parsing	Neural Network Properties
Feature Organization	Input Vector	Structural Relations	Input Vector
Training Mechanism			
Supervised	Density Estimation	Forming Grammars	Determining Neural Network Parameters
Unsupervised	Clustering	Clustering	Clustering
Limitations	Structural Information	Learning Structural Rules	Semantic Information

EXERCISES

1. Consider a biased input of the form

$$\tau_t(i) = \sum_k a_t(i) w_{ik} + b$$

and a logistic activation function. What bias b is necessary for $f(0) = 0$? Does this also hold for the algebraic sigmoid function?

Hint: The logistic function is defined as $f(x) = \frac{1}{1+\exp -\alpha x}$ with α being a slope parameter. The algebraic sigmoid function is given as $f(x) = \frac{x}{\sqrt{1+v^2}}$.

2. For $f(\tau_j)$ given as

$$f(\tau_j) = \frac{1}{1 + \exp -\{\frac{\tau_j - \theta_j}{\theta_0}\}},$$

a) Determine and plot $f^{'}(\tau_j)$ for $\tau_j = 0$ and $\theta_0 = 10$.

b) Repeat this for $\tau_j = 0$, $\theta_0 = 100$, and $\theta_0 = 0.1$.

3. Show that if the output activation is given by

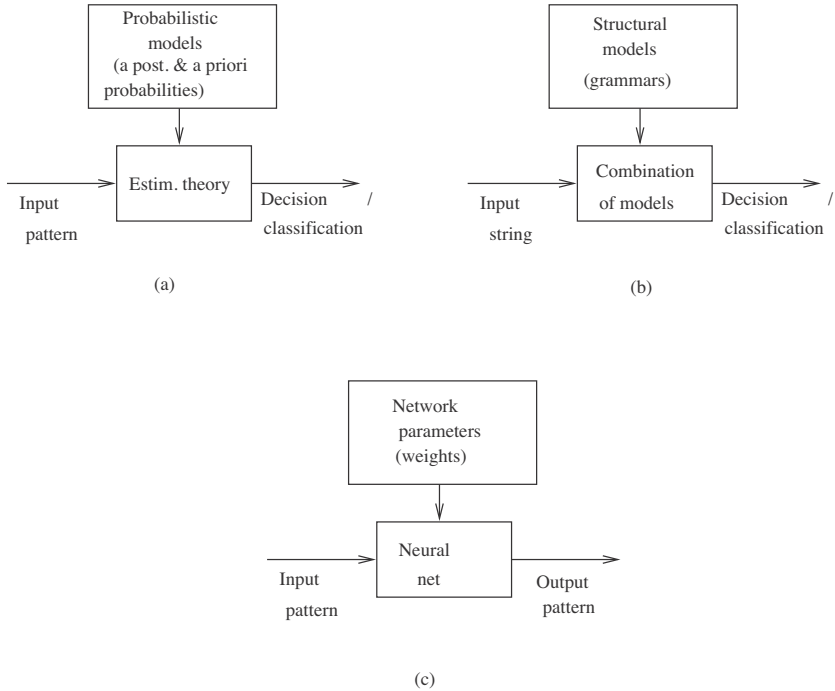


Figure 6.22
Pattern recognition approaches: (a) statistical approach (b) syntactic approach (c) neural approach.

$$o_j = f(\tau_j) = \frac{\tau_j}{\sqrt{1+\tau_j^2}}$$

then we obtain for its derivative

$$\frac{\partial f(\tau_j)}{\partial \tau_j} = \frac{f^3(\tau_j)}{\tau_j^3}$$

Is it possible to have a τ_j such that we obtain $f(\tau_j) = 0$?

4. Explain why an MLP does not learn if the initial weights and biases are all zeros.
5. A method to increase the rate of learning, yet to avoid the instability, is to modify the weight updating rule

$$w_{ij}(n) = w_{ij}(n-1) + \eta\delta_{hj}p_i^t \tag{6.54}$$

by including a momentum term as described in [61]

$$\Delta w_{ij}(n) = \alpha\Delta w_{ij}(n-1) + \eta\delta_{hj}p_i^t \tag{6.55}$$

where α is a positive constant called the momentum constant. Describe how this affects the weights and also explain how a normalized weight updating can be used for speeding the MLP backpropagation training.

6. The momentum constant is in most cases a small number with $0 \leq \alpha < 1$. Discuss the effect of choosing a small negative constant with $-1 < \alpha \leq 0$ for the modified weight updating rule from equation (6.55).

7. Create two data sets, one for training an MLP and the other for testing the MLP. Use a single-layer MLP and train it with the given data set. Use two possible nonlinearities: $f(x) = \frac{x}{\sqrt{1+v^2}}$ and $f(x) = \frac{2}{\pi}\tan x^{-1}$. Determine for each of the given nonlinearities

a) The computational accuracy of the network by using the test data.

b) The effect on the network performance by varying the size of the hidden layer.

8. Comment on the differences and similarities between the Kohonen map and the LVQ.

9. Which unsupervised learning neural networks are "topology-preserving" and which are "neighborhood-preserving"?

10. Consider a Kohonen map performing a mapping from a 3-D input onto a 1-D neural lattice of 100 neurons. The input data are random points uniformly distributed inside a sphere of radius 1 centered at the origin. Compute the map produced by the neural network after 100, 1000, and 10,000 iterations.

11. Write a program to show how the Kohonen map can be used for image compression. Choose blocks of $4\times$ representing gray values from the image as input vectors for the feature map.

12. When does the radial-basis neural network become a "fuzzy" neu-

ral network? Comment on the architecture of such a network and design strategies.

13. Show that the Gaussian function representing a radia-basis function is invariant under the product operator. In other words, prove that the product of two Gaussian functions is still a Gaussian function.

14. Find a solution for the XOR problem using an RBF network with four hidden units where four two-radial–basis function centers are given by $\mathbf{m}_1 = [1, 1]^T$, $\mathbf{m}_2 = [1, 0]^T$, $\mathbf{m}_3 = [0, 1]^T$, and $\mathbf{m}_4 = [0, 0]^T$. Determine the output weight matrix $\mathbf{W}$.

15. How does the choice of the weights of the Hopfield neural network affect the energy function in equation (6.45)?

16. Assume we switch the signs of the weights in the Hopfield algorithm. How does this affect the convergence?

7 Fuzzy Clustering and Genetic Algorithms

Besides artificial neural networks, fuzzy clustering and genetic algorithms represent an important class of processing algorithms for biosignals.

Biosignals are characterized by uncertainties resulting from incomplete or imprecise input information, ambiguity, ill–defined or overlapping boundaries among the disease classes or regions, and indefiniteness in extracting features and relations among them. Any decision taken at a particular point will heavily influence the following stages. Therefore, an automatic diagnosis system must have sufficient possibilities to capture the uncertainties involved at every stage, such that the system's output results should reflect minimal uncertainty. In other words, a pattern can belong to more than one class. Translated to clinical diagnosis, this means that a patient can exhibit multiple symptoms belonging to several disease categories. The symptoms do not have to be strictly numerical. Thus, fuzzy variables can be both linguistic and/or set variables. An example of a fuzzy variable is the heart-beat of a person ranging from 40 to 150 beats per minute, which can be described as slow, normal, or fast. The main difference between fuzzy and neural paradigms is that neural networks have the ability to learn from data, while fuzzy systems (1) quantify linguistic inputs and (2) provide an approximation of unknown and complex input-output rules.

Genetic algorithms are usually employed as optimization procedures in biosignal processing, such as determining the optimal weights for neural networks when applied, for example, to the segmentation of ultrasound images or to the classification of voxels.

This chapter reviews the basics of fuzzy clustering and of genetic algorithms. Several well-known fuzzy clustering algorithms and fuzzy learning vector quantization are presented.

7.1 Fuzzy Sets

Fuzzy sets are an important tool for the description of imprecision and uncertainty.

A classical set is usually represented as a set with a crisp boundary. For example,

$$X = \{x | x > 8\} \tag{7.1}$$

where 8 represents an unambiguous boundary. On the other hand, a fuzzy set does not have a crisp boundary. To represent this fact, a new concept is introduced, that of a membership function describing the smooth transition from the fact "belongs to a set" to "does not belong to a set". Fuzzyness stems not from the randomness of the members of the set but from the uncertain nature of concepts.

This chapter will review some of the basic notions and results in fuzzy set theory.

Fuzzy systems are described by fuzzy sets and operations on fuzzy sets. Fuzzy logic approximates human reasoning by using linguistic variables and introduces rules based on combinations of fuzzy sets by these operations. The notion of fuzzy set way introduced by Zadeh [295].

Crisp sets

DEFINITION 7.1: *Crisp set*

Let X be a non empty set considered to be the *universe of discourse*. A *crisp set* A is defined by enumerating all elements $x \in X$,

$$A = \{x_1, x_2, \cdots, x_n\} \tag{7.2}$$

that belong to A.

The universe of discourse consists of ordered or nonordered discrete objects or of the continuous space.

DEFINITION 7.2: *Membership function*

The *membership function* can be expressed by a function u_A, that maps X on a binary value described by the set $I = \{0, 1\}$:

$$u_A : X \to I, \quad u_A(x) = \begin{cases} 1 & \text{if} \quad x \in A \\ 0 & \text{if} \quad x \quad \notin A. \end{cases} \tag{7.3}$$

Here, $u_A(x)$ represents the *membership degree* of x to A.

Thus, an arbitrary x either belongs to A or it does not; partial member-

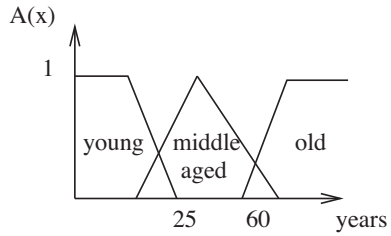


Figure 7.1
A membership function of temperature.

ship is not allowed.

For two sets A and B, combinations can be defined by the following operations:

$$A \cup B = \{x | x \in A \quad \text{or} \quad x \in B\} \tag{7.4}$$

$$A \cap B = \{x | x \in A \quad \text{and} \quad x \in B\} \tag{7.5}$$

$$\bar{A} = \{x | x \notin A, x \in X\}. \tag{7.6}$$

Additionally, the following rules have to be satisfied:

$$A \cup \bar{A} = \varnothing, \text{and} \quad A \cap \bar{A} = X \tag{7.7}$$

Fuzzy sets

DEFINITION 7.3: *Fuzzy set*

Let X be a non–empty set considered to be the universe of discourse. A *fuzzy set* is a pair (X, A), where $u_A : X \to I$ and $I = [0, 1]$.

Figure 7.1 is an example of a possible membership function.

The family of all fuzzy sets on the universe x will be denoted by $L(X)$. Thus

$$L(X) = \{u_A | u_A : X \to I\} \tag{7.8}$$

and $u_A(x)$ is the membership degree of x to A. For $u_A(x) = 0$, x does not belong to A, and for $u_A(x) = 1$, x does belong to A. All other cases are considered fuzzy.

Definition 7.4: *Membership function of a crisp set*
The fuzzy set A is called non ambiguous, or crisp, if $u_A(x) \in \{0, 1\}$.

Definition 7.5: *Complement of a fuzzy set*
If A is from $L(X)$, the *complement* of A is the fuzzy set $\bar{A}$, defined as

$$u_{\bar{A}}(x) = 1 - u_A(x), \forall x \in X \tag{7.9}$$

In the following, we define fuzzy operations which allow us to work with fuzzy sets defined by membership functions.

For two fuzzy sets A and B on X, the following operations can be defined.

Definition 7.6: *Equality*
Fuzzy set A is equal to fuzzy set B if and only if $u_A(x) = u_B(x)$ for all X. In symbols,

$$A = B \Longleftrightarrow u_A(x) = u_B(x), \forall x \in X \tag{7.10}$$

The next two definitions are for the inclusion and the product of two fuzzy sets.

Definition 7.7: *Inclusion*
Fuzzy set A is contained in fuzzy set B if and only if $u_A(x) \leq u_B(x)$ for all X. In symbols,

$$A \subseteqq B \Longleftrightarrow u_A(x) \leq u_B(x), \forall x \in X \tag{7.11}$$

Definition 7.8: *Product*
The product AB of fuzzy set A with fuzzy set B has a membership function that is the product of the two separate membership functions. In symbols,

$$u_{(AB)}(x) = u_A(x) \cdot u_B(x), \forall x \in X \tag{7.12}$$

The next two definitions pertain to intersection and union of two fuzzy sets.

Definition 7.9: *Intersection*

The intersection of two fuzzy sets A and B has as a membership function the minimum value of the two membership functions. In symbols,

$$u_{(A\cap B)}(x) = \min(u_A(x), u_B(x)), \forall x \in X \tag{7.13}$$

Definition 7.10: *Union*

The union of two fuzzy sets A and B has as a membership function the maximum value of the two membership functions. In symbols,

$$u_{(A\cup B)}(x) = \max(u_A(x), u_B(x)), \forall x \in X \tag{7.14}$$

Besides these classical set theory definitions, there are additional fuzzy operations possible, as shown in [71].

Definition 7.11: *Fuzzy partition*

The family $A_1, \cdots, A_n, n \geq 2$, of fuzzy sets is a *fuzzy partition* of the universe X if and only if the condition

$$\sum_{i=1}^{n} u_{A_i}(x) = 1 \tag{7.15}$$

holds for every x from X.

The above condition can be generalized for a fuzzy partition of a fuzzy set. By C we define a fuzzy set on X. We may require that the family $A_1, \cdots, A_n$ of fuzzy sets is a fuzzy partition of C if and only if the condition

$$\sum_{i=1}^{n} u_{A_i}(x) = u_C(x) \tag{7.16}$$

is satisfied for every x from X.

7.2 Mathematical Formulation of a Fuzzy Neural Network

Fuzzy neural networks represent an important extension of the traditional neural network. They are able to process "vague" information instead of crisp. The fuzziness can be found at different levels in the process: as a fuzzy input, weights, or logic equations.

We attempt to give a concise mathematical formulation of the fuzzy neural network as introduced by [194]. The fuzzy input is defined with $\mathbf{x}$ and is the fuzzy output vector is defined with $\mathbf{y}$, both being fuzzy numbers or intervals. The connection weight vector is denoted with $\mathbf{W}$.

The fuzzy neural network achieves a mapping from the n–dimensional input space to the l–dimensional space:

$$\mathbf{x}(t) \in \mathbf{R^n} \rightarrow \mathbf{y}(t) \in \mathbf{R^l}. \tag{7.17}$$

A confluence operation $\otimes$ determines the similarity between the fuzzy input vector $\mathbf{x}(t)$ and the connection weight vector $\mathbf{W}(t)$. For neural networks, the confluence operation represents a summation or product operation, while for the fuzzy neural network it describes an arithmetic operation such as fuzzy addition and fuzzy multiplication.

The output neurons implement the nonlinear operation

$$\mathbf{y}(t) = \psi[\mathbf{W}(t) \otimes \mathbf{x}(t)], \tag{7.18}$$

Based on the given training data $\{(\mathbf{x}(t), \mathbf{d}(t)), \mathbf{x}(t) \in \mathbf{R^n}, \mathbf{d}(t) \in \mathbf{R^l}, t = 1, \cdots, N\}$, the cost function can be optimized:

$$E_N = \sum_{t=1}^{N} d(\mathbf{y}(t), \mathbf{d}(t)), \tag{7.19}$$

where $d(\cdot)$ defines a distance in $\mathbf{R^l}$.

The learning algorithm of the fuzzy neural network is given by

$$\mathbf{W}(t+1) = \mathbf{W}(t) + \varepsilon \Delta \mathbf{W}(t), \tag{7.20}$$

and thus adjusts $N^{\mathbf{W}}$ connection weights of the fuzzy neural network.

Figure 7.2
Different cluster shapes: (a) compact, and (b) spherical.

7.3 Fuzzy Clustering Concepts

Clustering partitions a data set in groups of similar pattern, each group having a representant that is characteristic of the considered feature class. Within each group or cluster, patterns have the largest similarity to each other. In pattern recognition, we distinguish between crisp and fuzzy clustering. Fuzzy clustering has a major advantage in real-world application where the belonging of a pattern to a certain class is ambiguous. To obtain such a fuzzy partitioning, the membership function is allowed to have elements with values between 0 and 1, as shown in the previous section, In other words, in fuzzy clustering a pattern belongs *simultaneously* to more than one cluster, with the degree of belonging specified by membership grades between 0 and 1, whereas in traditional statistical approaches it belongs *exclusively* to only one cluster.

Clustering is based on minimizing a cost or objective function J of dissimilarity (or distance) measure. This predefined measure J is a function of the input data and of an unknown parameter vector set $\mathbf{L}$. The number of clusters n is assumed in the following to be predefined and fixed. Algorithms with growing or pruning cluster numbers and geometries are more sophisticated and are described in [264].

An optimal clustering is achieved by determining the parameter $\mathbf{L}$ such that the cluster structure of the input data is as captured as well as possible. It is plausible that this parameter depends on the type of geometry of the cluster: compact or spherical as visualized in figure 7.2.

While compact clusters can be accurately described by a set of n points $\mathbf{L}_i \in \mathbf{L}$ representing these clusters, spherical clusters are described by the centers of the cluster $\mathbf{V}$ and by the radii $\mathbf{R}$ of the clusters.

In the following, we will review the most important fuzzy clustering

techniques, and show their relationship to nonfuzzy approaches.

Metric concepts for fuzzy classes

Let $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_p\}, \mathbf{x}_j \in \mathbf{R}^s$, be a data set. Suppose the optimal number of clusters in $\mathbf{X}$ is given and that the cluster structure of $\mathbf{X}$ may be described by disjunct fuzzy sets which, when combined, yield $\mathbf{X}$.

Also, let C be a fuzzy set associated with a class of objects from $\mathbf{X}$ and $F_n(C)$ be the family of all n–member fuzzy partitions of C. Let n be the given number of subclusters in C. The cluster family of C can be appropriately described by a fuzzy partition P from $F_n(C)$, $P = \{A_1, \cdots, A_n\}$.

Every class A_i is described by a cluster prototype $\mathbf{L}_i$ which represents a point in an s–dimensional Euclidean space $\mathbf{R}^s$. The clusters' form can be either spherical or ellipsoidal. $\mathbf{L}_i$ represents the mean vector of the fuzzy class A_i.

The fuzzy partition is typically described by an $n \times p$ *membership matrix* $\mathbf{U} = [u_{ij}]_{n\times p}$ which has binary values for crisp partitions and continuous values between 0 and 1 for fuzzy partitions. Thus, the membership u_{ij} represents the degree of assignment of the pattern $\mathbf{x}_j$ to the ith class. The contrast between fuzzy and crisp partition is the following: Given a fuzzy partition, a given data point $\mathbf{x}_j$ can belong to several classes as assigned by the membership matrix $\mathbf{U} = [u_{ij}]_{n\times p}$, while for a crisp partition, this data point belongs to exactly one class. In the following we will use the notation $u_{ij} = u_i(\mathbf{x}_j)$.

We also will give the definition of a weighted Euclidean distance.

DEFINITION 7.12: The *norm–induced distance* d between two data $\mathbf{x}$ and $\mathbf{y}$ from $\mathbf{R}^s$ is given by

$$d^2(\mathbf{x}, \mathbf{y}) = ||\mathbf{x} - \mathbf{y}|| = (\mathbf{x} - \mathbf{y})^T \mathbf{M} (\mathbf{x} - \mathbf{y}) \tag{7.21}$$

where $\mathbf{M}$ is a symmetric positive definite matrix.

The distance with respect to a fuzzy class is given by definition.

DEFINITION 7.13: The distance d_i between $\mathbf{x}$ and $\mathbf{y}$ with respect to

the fuzzy class A_i is given by

$$d_i(\mathbf{x}, \mathbf{y}) = \min(u_{A_i}(\mathbf{x}), u_{A_i}(\mathbf{y}))d(\mathbf{x}, \mathbf{y}), \quad \forall \mathbf{x}, \mathbf{y} \in \mathbf{X} \tag{7.22}$$

Alternating optimization technique

The minimization of the objective function for fuzzy clustering depends on variables such as cluster geometry as well as the membership matrix. The standard approach used in most analytical optimization-based cluster algorithms where coupled parameters are optimized alternatively, is the alternating optimization technique. In each iteration, a set of variables is optimized while fixing all others. In general, the cluster algorithm attempts to minimize an objective function which is based n either an intra class similarity measure or a dissimilarity measure.

Let the cluster substructure of the fuzzy class C be described by the fuzzy partition $P = \{A_1, \cdots, A_n\}$ of C being equivalent to

$$\sum_{j=1}^{p} u_{ij} = u_C(\mathbf{x}_j), \quad j = 1, \cdots, p. \tag{7.23}$$

Further, let $\mathbf{L}_i \in \mathbf{R}^s$ be the prototype of the fuzzy class A_i, and a point from the data set $\mathbf{X}$. We then obtain

$$u_{A_i}(\mathbf{L}_i) = \max_j u_{ij}. \tag{7.24}$$

The *dissimilarity* between a data point and a prototype $\mathbf{L}_i$ is given by:

$$D_i(\mathbf{x}_j, \mathbf{L}_i) = u_{ij}^2 d^2(\mathbf{x}_j, \mathbf{L}_i). \tag{7.25}$$

The *inadequacy* $I(A_i, \mathbf{L}_i)$ between the fuzzy class A_i and its prototype is defined as

$$I(A_i, \mathbf{L}_i) = \sum_{j=1}^{p} D_i(\mathbf{x}_j, \mathbf{L}_i) \tag{7.26}$$

Assume $\mathbf{L} = (\mathbf{L}_1, \cdots, \mathbf{L}_n)$ is the set of cluster centers and describes a representation of the fuzzy partition P.

The inadequacy $J(P, \mathbf{L})$ between the partition P and its representation $\mathbf{L}$ is defined as

$$J(P, \mathbf{L}) = \sum_{i=1}^{n} I(A_i, \mathbf{L}_i) \tag{7.27}$$

Thus the objective function $J : F_n(C) \times \mathbf{R}^{sn} \to R$ is obtained:

$$J(P, \mathbf{L}) = \sum_{i=1}^{n} \sum_{j=1}^{p} u_{ij}^2 d^2(\mathbf{x}_j, \mathbf{L}_i) = \sum_{i=1}^{n} \sum_{j=1}^{p} u_{ij}^2 ||\mathbf{x}_j - \mathbf{L}_i||^2 \tag{7.28}$$

It can be seen that the objective function is of the least-squares error type, and a local solution of this minimization problem gives the optimal fuzzy partition and its representation:

$$\begin{cases} \text{minimize} \quad J(P, \mathbf{L}) \\ P \in F_n(C) \\ \mathbf{L} \in \mathbf{R}^{sn} \end{cases} \tag{7.29}$$

We obtain an approximate solution of the above problem based on an iterative method, the *alternating optimization technique* [33], by minimizing the functions $J(P, \cdot)$ and $J(\cdot, \mathbf{L})$.

In other words, the minimization problem from equation (7.29) is replaced by two separate problems:

$$\begin{cases} \text{minimize} \quad J(P, \mathbf{L}) \to \min \\ P \in F_n(C) \\ \mathbf{L} \quad \text{is fixed} \end{cases} \tag{7.30}$$

and

$$\begin{cases} \text{minimize} \quad J(P, \mathbf{L}) \to \min \\ \mathbf{L} \in \mathbf{R}^{sn} \\ P \quad \text{is fixed} \end{cases} \tag{7.31}$$

To solve the first optimization problem, we introduce the notation

$$I_j = \{i | 1 \leq i \leq n, \quad d(\mathbf{x}_j, \mathbf{L}_i) = 0\} \tag{7.32}$$

and

$$\bar{I}_j = \{1, 2, \cdots, n\} - I_j. \tag{7.33}$$

Two theorems without proof are given regarding the minimization of the function $J(P, \cdot)$ or $J(\cdot, \mathbf{L})$ in equations (7.30) and (7.31).

Theorem 7.1:

$P \in F_n(C)$ represents a minimum of the function $J(\cdot, \mathbf{L})$ only if

$$I_j = \varnothing \Rightarrow u_{ij} = \frac{u_C(\mathbf{x}_j)}{\sum\limits_{k=1}^{n} \frac{d^2(\mathbf{x}_j, \mathbf{L}_i)}{d^2(\mathbf{x}_j, \mathbf{L}_k)}}, \quad \forall 1 \leq i \leq n; \quad 1 \leq j \leq p \tag{7.34}$$

and

$$I_j \neq \varnothing \Rightarrow u_{ij} = 0, \forall i \in I_j \tag{7.35}$$

and arbitrarily $\sum\limits_{i \in I_j} u_{ij} = u_C(\mathbf{x}_j)$.

THEOREM 7.2:

If $\mathbf{L} \in \mathbf{R}^{sn}$ is a local minimum of the function $J(P, \cdot)$, then $\mathbf{L}_i$ is the cluster center (mean vector) of the fuzzy class A_i for every $i = 1, \cdots, n$:

$$\mathbf{L}_i = \frac{1}{\sum\limits_{j=1}^{p} u_{ij}^2} \sum_{j=1}^{p} u_{ij}^2 \mathbf{x}_j \tag{7.36}$$

The alternating optimization (AO) technique is based on the Picard iteration of equations (7.34), (7.35), and (7.36).

It is worth mentioning that a more general objective function can be considered:

$$J_m(P, \mathbf{L}) = \sum_{i=1}^{n} \sum_{j=1}^{p} u_{ij}^m d^2(\mathbf{x}_j, \mathbf{L}_i) \tag{7.37}$$

with $m > 1$ being a weighting exponent, sometimes known as a *fuzzifier*, and d the norm–induced distance.

Similar to the case $m = 2$ shown in equation (7.28), we have two solutions for the optimization problem regarding both the prototypes and the fuzzy partition. Since the parameter m can take infinite values, an infinite family of fuzzy clustering algorithms is obtained. In the case $m \to 1$, the fuzzy n-means algorithm converges to a hard n-means solution. As m becomes larger, more data with small degrees of membership are neglected, and thus more noise is eliminated.

7.4 Fuzzy Clustering Algorithms

This section describes several well-known fuzzy clustering algorithms, such as the generalized adaptive fuzzy n-means algorithm, the generalized adaptive fuzzy n-shells algorithm, the Gath-Geva algorithms, and fuzzy learning vector quantization algorithms.

Let $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_p\}$ define the data set, and C a fuzzy set, on $\mathbf{X}$. The following assumptions are made:

- C represents a cluster of points from $\mathbf{X}$.
- C has a cluster substructure described by the fuzzy partition $P = \{A_1, \cdots, A_n\}$.
- n is the number of known subclusters in C.

The algorithms require a random initialization of the fuzzy partition. In order to monitor the convergence of the algorithm, the $n \times p$ *partition matrix* $\mathbf{Q}^i$ is introduced to describe each fuzzy partition P^i at the ith iteration, and is used to determine the distance between two fuzzy partitions. The matrix $\mathbf{Q}^i$ is defined as

$$\mathbf{Q}^i = \mathbf{U} \quad \text{at} \quad \text{iteration} \quad \text{i.} \tag{7.38}$$

The termination criterion for iteration m is given by

$$d(P^m, P^{m-1}) = ||\mathbf{Q}^m - \mathbf{Q}^{m-1}|| < \varepsilon. \tag{7.39}$$

where ε defines the admissible error and $||\cdot||$ is any vector norm.

Generalized Adaptive Fuzzy n-Means Algorithm

This adaptive fuzzy technique employs different distance metrics such that several cluster shapes, ranging from spherical to ellipsoidal, can be detected.

To achieve this, an adaptive metric is used. We define a new distance metric $d(\mathbf{x}_j, \mathbf{L}_i)$, from the data point $\mathbf{x}_j$ to the cluster prototype $\mathbf{L}_i$, as

$$d^2(\mathbf{x}_j, \mathbf{L}_i) = (\mathbf{x}_j - \mathbf{L}_i)^T \mathbf{M}_i (\mathbf{x}_j - \mathbf{L}_i), \tag{7.40}$$

where $\mathbf{M}_i$ is a symmetric and positive definite shape matrix and adapts to the clusters' shape variations. The growth of the shape matrix is

monitored by the bound

$$|\mathbf{M}_i| = \rho_i, \quad \rho_i > 0, \quad i = 1, \cdots, n \tag{7.41}$$

Let $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_p\}$, $\mathbf{x}_j \in \mathbf{R}^s$ be a data set. Let C be a fuzzy set on $\mathbf{X}$ describing a fuzzy cluster of points in $\mathbf{X}$, and having a cluster substructure which is described by a fuzzy partition $P = \{A_1, \cdots, A_n\}$ of C. Each fuzzy class A_i is described by the point prototype $\mathbf{L}_i \in \mathbf{R}^s$. The local distance with respect to A_i is given by

$$d_i^2(\mathbf{x}_j, \mathbf{L}_i) = u_{ij}^2 (\mathbf{x}_j - \mathbf{L}_i)^T \mathbf{M}_i (\mathbf{x}_j - \mathbf{L}_i) \tag{7.42}$$

As an objective function we choose

$$J(P, \mathbf{L}, M) = \sum_{i=1}^{n} \sum_{j=1}^{p} d^2(\mathbf{x}_j, \mathbf{L}_i) = \sum_{i=1}^{n} \sum_{j=1}^{p} u_{ij}^2 (\mathbf{x}_j - \mathbf{L}_i)^T \mathbf{M}_i (\mathbf{x}_j - \mathbf{L}_i) \tag{7.43}$$

where $M = (\mathbf{M}_1, \cdots, \mathbf{M}_n)$.

The objective function chosen is again of the least-squares error type. We can find the optimal fuzzy partition and its representation as the local solution of the minimization problem:

$$\begin{cases} \text{minimize} \quad J(P, \mathbf{L}, M) \\ \sum\limits_{i=1}^{n} u_{ij} = u_C(\mathbf{x}_j), \quad j = 1, \cdots, p \\ |\mathbf{M}_i| = \rho_i, \quad \rho_i > 0, \quad i = 1, \cdots, n \\ \mathbf{L} \in \mathbf{R}^{sn} \end{cases} \tag{7.44}$$

Without proof theorem 7.3 which regards the minimization of the functions $J(P, \mathbf{L}, \cdot)$, is given. It is known as the adaptive norm theorem.

THEOREM 7.3: Assuming that the point prototype $\mathbf{L}_i$ of the fuzzy class A_i equals the cluster center of this class, $\mathbf{L}_i = \mathbf{m}_i$, and the determinant of the shape matrix $\mathbf{M}_i$ is bounded, $|\mathbf{M}_i| = \rho_i, \rho_i > 0, i = 1, \cdots, n$, then $\mathbf{M}_i$ is a local minimum of the function $J(P, \mathbf{L}, \cdot)$ only if

$$\mathbf{M}_i = [\rho_i |\mathbf{S}_i|]^{\frac{1}{s}} \mathbf{S}_i^{-1} \tag{7.45}$$

where $\mathbf{S}_i$ is the within-class scatter matrix of the fuzzy class A_i:

$$\mathbf{S}_i = \sum_{j=1}^{p} u_{ij}^2 (\mathbf{x}_j - \mathbf{m}_i)(\mathbf{x}_j - \mathbf{m}_i)^T. \tag{7.46}$$

Theorem 7.3 can be employed as part of an alternating optimization technique. The resulting iterative procedure is known as the *generalized adaptive fuzzy n–means* (GAFNM) algorithm.

An algorithmic description of the GAFNM is given below.

1. **Initialization:** Choose the number n of subclusters in C and the termination criterion ε. P^1 is selected as a random fuzzy partition of C having n atoms. Set the iteration counter $l = 1$.
2. **Adaptation, part I:** Determine the cluster prototypes $\mathbf{L}_i = \mathbf{m}_i, i = 1, \cdots, n$ using

$$\mathbf{L}_i = \frac{1}{\sum_{j=1}^{p} u_{ij}^2} \sum_{j=1}^{p} u_{ij}^2 \mathbf{x}_j. \tag{7.47}$$

3. **Adaptation, part II:** Determine the within-class scatter matrix $\mathbf{S}_i$ using

$$\mathbf{S}_i = \sum_{j=1}^{p} u_{ij}^2 (\mathbf{x}_j - \mathbf{m}_i)(\mathbf{x}_j - \mathbf{m}_i)^T. \tag{7.48}$$

Determine the shape matrix $\mathbf{M}_i$ using

$$\mathbf{M}_i = [\rho_i |\mathbf{S}_i|]^{\frac{1}{s}} \mathbf{S}_i^{-1} \tag{7.49}$$

and compute the distance $d^2(\mathbf{x}_j, \mathbf{m}_i)$ using

$$d^2(\mathbf{x}_j, \mathbf{m}_i) = (\mathbf{x}_j - \mathbf{m}_i)^T \mathbf{M}_i (\mathbf{x}_j - \mathbf{m}_i). \tag{7.50}$$

4. **Adaptation, part III:** Compute a new fuzzy partition P^l of C using the rules

$$I_j = \varnothing \Rightarrow u_{ij} = \frac{u_C(\mathbf{x}_j)}{\sum\limits_{k=1}^{n} \frac{d^2(\mathbf{x}_j, \mathbf{m}_i)}{d^2(\mathbf{x}_j, \mathbf{m}_k)}}, \quad \forall 1 \leq i \leq n; \quad 1 \leq j \leq p \tag{7.51}$$

and

$$I_j \neq \varnothing \Rightarrow u_{ij} = 0, \forall i \in I_j \tag{7.52}$$

and arbitrarily $\sum\limits_{i \in I_j} u_{ij} = u_C(\mathbf{x}_j)$.
The standard notation is used:

$$I_j = \{i | 1 \leq i \leq n, \quad d(\mathbf{x}_j, \mathbf{L}_i) = 0\} \tag{7.53}$$

and

$$\bar{I}_j = \{1, 2, \cdots, n\} - I_j \tag{7.54}$$

5. **Continuation:** If the difference between two successive partitions is smaller than a predefined threshold, $||P^l - P^{l-1}|| < \varepsilon$, then stop. Otherwise, go to step 2.

An important issue for the GAFNM algorithm is the selection of the bounds of the shape matrix $\mathbf{M}_i$. They can be chosen as

$$\rho_i = 1, \quad i = 1, \cdots, n \tag{7.55}$$

If we choose $C = \mathbf{X}$, we obtain $u_C(\mathbf{x}_j) = 1$ and thus get the membership degrees

$$u_{ij} = \frac{1}{\sum\limits_{k=1}^{n} \frac{d^2(\mathbf{x}_j, \mathbf{m}_i)}{d^2(\mathbf{x}_j, \mathbf{m}_k)}}, \quad \forall 1 \leq i \leq n; \quad 1 \leq j \leq p \tag{7.56}$$

The resulting iterative procedure is known as the adaptive fuzzy n-means (AFNM) algorithm.

Generalized adaptive fuzzy n-shells algorithm

So far, we have considered clustering algorithms that use point prototypes as cluster prototypes. Therefore, the previous algorithms cannot

detect clusters that can be described by shells, hyperspheres, or hyperellipsoids. The *generalized adaptive fuzzy n-shells* algorithm [63, 64] is able to detect such clusters. The cluster prototypes that are used are s-dimensional hyperellipsoidal shells, and the distances of data points are measured from the hyperellipsoidal surfaces. Since the prototypes contain no interiors, they are referred to as shells.

The hyperellipsoidal shell prototype $\mathbf{L}_i(\mathbf{v}_i, r_i, \mathbf{M}_i)$ of the fuzzy class A_i is given by the set

$$\mathbf{L}_i(\mathbf{v}_i, r_i, \mathbf{M}_i) = \{\mathbf{x} \in \mathbf{R}^s | (\mathbf{x} - \mathbf{v}_i)^T \mathbf{M}_i (\mathbf{x} - \mathbf{v}_i) = r_i^2\} \tag{7.57}$$

with $\mathbf{M}_i$ representing a symmetric and positive definite matrix.
The distance d_{ij} between the point $\mathbf{x}_j$ and the cluster center $\mathbf{v}^i$ is defined as

$$d_{ij}^2 = d^2(\mathbf{x}_j, \mathbf{v}_i) = [(\mathbf{x} - \mathbf{v}_i)^T \mathbf{M}_i (\mathbf{x} - \mathbf{v}_i)]^{\frac{1}{2}} - r_i \tag{7.58}$$

Thus a slightly changed objective function is obtained:

$$J(P, V, R, M) = \sum_{i=1}^{n} \sum_{j=1}^{p} u_{ij}^2 d_{ij}^2 = \sum_{i=1}^{n} \sum_{j=1}^{p} u_{ij}^2 [(\mathbf{x} - \mathbf{v}_i)^T \mathbf{M}_i (\mathbf{x} - \mathbf{v}_i)]^{\frac{1}{2}} - r_i]^2. \tag{7.59}$$

For optimization purposes, we need to determine the minimum of the functions $J(\cdot, V, R, M)$, $J(P, \cdot, R, M)$, and $J(P, V, \cdot, M)$. It can be shown that they are given by propositions 7.1 and 7.2 [71].

Proposition 7.1 is the proposition for optimal partition.

Proposition 7.1: The fuzzy partition P represents the minimum of the function $J(\cdot, \mathbf{V}, \mathbf{R}, \mathbf{M})$ only if

$$I_j = \varnothing \Rightarrow u_{ij}) = \frac{u_C(\mathbf{x}_j)}{\sum\limits_{k=1}^{n} \frac{d_{ij}^2}{d_{kj}^2}} \tag{7.60}$$

and

$$I_j \neq \varnothing \Rightarrow u_{ij} = 0, \forall i \in I_j \tag{7.61}$$

and arbitrarily $\sum\limits_{i \in I_j} u_{ij} = u_C(\mathbf{x}_j)$.

Proposition 7.2 is the proposition for optimal prototype centers.

PROPOSITION 7.2: The optimal value of $\mathbf{V}$ with respect to the function $J(P, \cdot, R, M)$ is given by

$$\sum_{j=1}^{p} u_{ij}^2 \frac{d_{ij}}{q_{ij}} (\mathbf{x}_j - \mathbf{v}_i) = 0, \quad i = 1, \cdots, n, \tag{7.62}$$

where q_{ij} is given by

$$q_{ij} = (\mathbf{x}_j - \mathbf{v}_i)^T \mathbf{M}_i (\mathbf{x}_j - \mathbf{v}_i) \tag{7.63}$$

Proposition 7.3 is the proposition for optimal prototype radii.

PROPOSITION 7.3: The optimal value of R with respect to the function $J(P, V, \cdot, M)$ is given by

$$\sum_{j=1}^{p} u_{ij}^2 d_{ij} = 0, \quad i = 1, \cdots, n. \tag{7.64}$$

To ensure that the adaptive norm is bounded, we impose the constraint

$$|\mathbf{M}_i| = \rho_i, \quad \text{where} \quad \rho_i > 0, \quad i = 1, \cdots, n \tag{7.65}$$

The norm is given by theorem 7.4, the adaptive norm theorem [71].

THEOREM 7.4:

Let $\mathbf{X} \subset \mathbf{R^s}$. Suppose the objective function J already contains the optimal P, V, and R. If the determinant of the shape matrix $\mathbf{M}_i$ is bounded, $|\mathbf{M}_i| = \rho_i, \quad \rho_i > 0, \quad i = 1, \cdots, n$, then $\mathbf{M}_i$ is a local minimum of the function $J(P, V, R, \cdot)$ only if

$$\mathbf{M}_i = [\rho_i |\mathbf{S}_{si}|]^{\frac{1}{s}} \mathbf{S}_{si}^{-1}, \tag{7.66}$$

where $\mathbf{S}_{si}$ represents the nonsingular shell scatter matrix of the fuzzy class A_i:

$$\mathbf{S}_{si} = \sum_{j=1}^{p} u_{ij}^2 \frac{d_{ij}}{q_{ij}} (\mathbf{x}_j - \mathbf{v}_i)(\mathbf{x}_j - \mathbf{v}_i)^T. \tag{7.67}$$

In praxis, the bound is chosen as $\rho_i = 1$, $i = 1, \cdots, n$.

The above theorems can be used as the basis of an alternating optimization technique. The resulting iterative procedure is known under as the *generalized adaptive fuzzy n-shells* (GAFNS) algorithm.

An algorithmic description of the GAFNS is given below:

1. **Initialization:** Choose the number n of subclusters in C and the termination criterion ε. P^1 is selected as a random fuzzy partition of C having n atoms. Initialize $\mathbf{M}_i = \mathbf{I}, i = 1, \cdots, n$ where $\mathbf{I}$ is a $s \times s$ unity matrix. Set the iteration counter $l = 1$.
2. **Adaptation, part I:** Determine the centers $\mathbf{v}_i$ and radii r_i by solving the system of equations

$$\begin{cases} \sum_{j=1}^{p} u_{ij}^2 \frac{d_{ij}}{q_{ij}} (\mathbf{x}_j - \mathbf{v}_i) = 0 \\ \sum_{j=1}^{p} u_{ij}^2 d_{ij} = 0 \end{cases} \tag{7.68}$$

   where $i = 1, \cdots, n$ and $q_{ij} = (\mathbf{x}_j - \mathbf{v}_i)^T \mathbf{M}_i (\mathbf{x}_j - \mathbf{v}_i$.
3. **Adaptation, part II:** Determine the shell scatter matrix $\mathbf{S}_{si}$ of the fuzzy class A_i,

$$\mathbf{S}_{si} = \sum_{j=1}^{p} u_{ij}^2 \frac{d_{ij}}{q_{ij}} (\mathbf{x}_j - \mathbf{v}_i)(\mathbf{x}_j - \mathbf{v}_i)^T. \tag{7.69}$$

   where the distance d_{ij} is given by

$$d_{ij}^2 = [(\mathbf{x}_j - \mathbf{v}_i)^T \mathbf{M}_i (\mathbf{x}_j - \mathbf{v}_i)]^{1/2} - r_i \tag{7.70}$$

4. **Adaptation, part III:** Determine the approximate value of $\mathbf{M}_i$:

$$\mathbf{M}_i = [\rho_i |\mathbf{S}_{si}|]^{\frac{1}{s}} \mathbf{S}_{si}^{-1}, \quad i = 1, \cdots, n \tag{7.71}$$

   where $\rho_i = 1$ or ρ_i is equal to the determinant of the previous $\mathbf{M}_i$.
5. **Adaptation, part IV:** Compute a new fuzzy partition P^l of C using the following rules:

$$I_j = \varnothing \Rightarrow u_{ij} = \frac{u_C(\mathbf{x}_j)}{\sum\limits_{k=1}^{n} \frac{d_{ij}^2}{d_{kj}^2}} \tag{7.72}$$

and

$$I_j \neq \varnothing \Rightarrow u_{ij} = 0, \quad \forall i \in I_j \tag{7.73}$$

and arbitrarily $\sum\limits_{i \in I_j} u_{ij} = u_C(\mathbf{x}_j)$.

Set $l = l + 1$.

6. **Continuation:** If the difference between two successive partitions is smaller than a predefined threshold,$||P^l - P^{l-1}|| < \varepsilon$, then stop. Else go to step 2.

If we choose $u_C = \mathbf{X}$, we obtain $u_C(\mathbf{x}_j) = 1$, and thus we get the following fuzzy partition:

$$I_j = \varnothing \Rightarrow u_{ij} = \frac{1}{\sum\limits_{k=1}^{n} \frac{d_{ij}^2}{d_{kj}^2}} \tag{7.74}$$

and

$$I_j \neq \varnothing \Rightarrow u_{ij} = 0, \forall i \in I_j \tag{7.75}$$

and arbitrarily $\sum\limits_{i \in I_j} u_{ij} = 1$.

The resulting iterative procedure is known a *adaptive fuzzy n-shells* (AFNS) algorithm. This technique enables us to identify the elliptical data substructure, and even to detect overlapping between clusters to some degree.

The Gath–Geva algorithm

A major problem arises when fuzzy clustering is performed in real–world tasks: the necessary cluster number, their locations, their shapes, and their densities are usually not known beforehand. The Gath-Geva algorithm [89] represents an important development of existing fuzzy clustering algorithms. The cluster sizes are not restricted as im other algorithms, and the cluster densities are also considered.

To allow the detection of cluster shapes ranging from spherical to ellipsoidal, different metrics have to be used. Usually, an adaptive metric is used. In general a distance metric $d(\mathbf{x}_j, \mathbf{L}_i)$ from the data point $\mathbf{x}_j$ to the cluster prototype $\mathbf{L}_i$ is defined as

$$d^2(\mathbf{x}_j, \mathbf{L}_i) = (\mathbf{x}_j - \mathbf{L}_i)^T \mathbf{F}_i^{-1} (\mathbf{x}_j - \mathbf{L}_i), \tag{7.76}$$

where $\mathbf{F}_i$ is a symmetric and positive definite shape matrix, and adapts to the clusters' shape variations.

Due to this exponential distance, the Gath–Geva algorithm seeks an optimum in a narrow local region. Its major advantage is obtaining good partition results in cases of unequally variable features and densities, but only when the starting cluster prototypes are properly chosen.

An algorithmic description of the Gath–Geva algorithm is given below [89]:

1. **Initialization and adaptation, part I:** These are similar to the fuzzy n-means algorithm.
3. **Adaptation, part II:** Determine the fuzzy covariance matrix $F_i, i = 1, \cdots, c$ by using

$$F_i = \frac{\sum_{k=1}^{N} u_{ik}^2 (\mathbf{x}_k - \mathbf{L}_i)(\mathbf{x}_k - \mathbf{L}_i)^T}{\sum_{k=1}^{N} u_{ik}^2} \tag{7.77}$$

4. **Adaptation, part III:** Compute the exponential distance d_e:

$$d_e^2(\mathbf{x}_j, \mathbf{L}_i) = \frac{\sqrt{|\mathbf{F}_i|}}{\alpha_i} e^{[(\mathbf{x}_j - \mathbf{L}_i)^T \mathbf{F}_i^{-1} (\mathbf{x}_j - \mathbf{L}_i)/2]}, \tag{7.78}$$

with the a priori probability $\alpha_i = \frac{1}{N} \sum_{k=1}^{N} u_{ik}^{l-1}$, where $l-1$ is the previous iteration.

5. **Adaptation, part IV:** Update the membership degrees according to

$$u_{ij} = \frac{1}{\sum_{k=1}^{c} \frac{d_e^2(\mathbf{x}_j, \mathbf{L}_i)}{d_e^2(\mathbf{x}_j, \mathbf{L}_k)}}, \forall 1 \leq i \leq c; \quad 1 \leq j \leq N. \tag{7.79}$$

6. **Continuation:** If the difference between two successive partitions is smaller than a predefined threshold $||\mathbf{U}^l - \mathbf{U}^{l-1}|| < \varepsilon$, then stop. Else go to step 2.

Fuzzy algorithms for learning vector quantization

The idea of combining the advantages of fuzzy logic with learning vector quantization is reflected in the concept of fuzzy learning vector quantization (FALVQ) [15, 130], where FALVQ stands for fuzzy algorithms for learning vector quantization. Thus, fusing the concepts of approximate reasoning and imprecision with unsupervised learning acquires the benefits of both paradigms.

Let us consider the set $\mathbf{X}$ of samples from an n–dimensional Euclidean space and let $f(\mathbf{x})$ be the probability distribution function of $\mathbf{x} \in \mathbf{X} \in \mathbf{R}^n$. Learning vector quantization is based on the minimization of the functional [193]

$$D(\mathbf{L}_1, \cdots, \mathbf{L}_c) = \int \cdots \int_{\mathbf{R}^n} \sum_{r=1}^{c} u_r(\mathbf{x}) ||\mathbf{x} - \mathbf{L}_r||^2 f(\mathbf{x}) d\mathbf{x} \tag{7.80}$$

with $D_{\mathbf{x}} = D_{\mathbf{x}}(\mathbf{L}_1, \cdots, \mathbf{L}_c)$ being the expectation of the loss function, defined as

$$D_{\mathbf{x}}(\mathbf{L}_1, \cdots, \mathbf{L}_c) = \sum_{r=1}^{c} u_r(\mathbf{x}) ||\mathbf{x} - \mathbf{L}_r||^2 \tag{7.81}$$

$u_r = u_r(\mathbf{x}), 1 \leq r \leq c$, all membership functions that describe competitions between the prototypes for the input $\mathbf{x}$. Supposing that $\mathbf{L}_i$ is the winning prototype that belongs to the input vector $\mathbf{x}$, that is, the closest prototype to $\mathbf{x}$ in the Euclidean sense, the memberships $u_{ir} = u_r(\mathbf{x}), 1 \leq r \leq c$ are given by

$$u_{ir} = \begin{cases} 1, & \text{if} \quad r = i, \\ u(\frac{||\mathbf{x}-\mathbf{L}_i||^2}{||\mathbf{x}-\mathbf{L}_r||^2}), & \text{if} \quad r \neq i \end{cases} \tag{7.82}$$

The role of the loss function is to evaluate the error of each input vector locally with respect to the winning reference vector.

FALVQ considers both the very important winning prototype and also the global non winning information. Several FALVQ algorithms can

Table 7.1
Membership functions and interference functions for the FALVQ1, FALVQ2, and FALVQ3 families of algorithms

Algorithm	u(z)	w(z)	n(z)
FALVQ1 $(0 < \alpha < \infty)$	$z(1+\alpha z)^{-1}$	$(1+\alpha z)^{-2}$	$\alpha z^2(1+\alpha z)^{-2}$
FALVQ2 $(0 < \beta < \infty$	$z\exp(-\beta z)$	$(1-\beta z)\exp(-\beta z)$	$\beta z^2\exp(-\beta z)$
FALVQ3 $(0 < \gamma < 1)$	$z(1-\gamma z)$	$1-2\gamma z$	γz^2

be determined based on minimizing the loss function.

The winning prototype $\mathbf{L}_i$ is adapted iteratively, based on the following rule:

$$\Delta\mathbf{L}_i = -\eta^{'}\frac{\partial D_{\mathbf{x}}}{\partial \mathbf{L}_i} = \eta(\mathbf{x}-\mathbf{L}_i)\left(1+\sum_{i\neq r}^{c} w_{ir}\right), \tag{7.83}$$

where

$$w_{ir} = u^{'}\left(\frac{||\mathbf{x}-\mathbf{L}_i||^2}{||\mathbf{x}-\mathbf{L}_r||^2}\right) = w\left(\frac{||\mathbf{x}-\mathbf{L}_i||^2}{||\mathbf{x}-\mathbf{L}_r||^2}\right). \tag{7.84}$$

The nonwinning prototype $\mathbf{L}_j \neq \mathbf{L}_i$ is also adapted iteratively, based on the following rule:

$$\Delta\mathbf{L}_j = -\eta^{'}\frac{\partial D_{\mathbf{x}}}{\partial \mathbf{L}_j} = \eta(\mathbf{x}-\mathbf{L}_j)n_{ij} \tag{7.85}$$

where

$$n_{ij} = n\left(\frac{||\mathbf{x}-\mathbf{L}_i||^2}{||\mathbf{x}-\mathbf{L}_j||^2}\right) = u_{ij} - \frac{||\mathbf{x}-\mathbf{L}_i||^2}{||\mathbf{x}-\mathbf{L}_j||^2}w_{ij}$$

It is very important to mention that the fuzzyness in FALVQ is employed in the learning rate and update strategies, and is not used for creating fuzzy outputs.

The above-presented mathematical framework forms the basis of the three fuzzy learning vector quantization algorithms presented in [131]. Table 7.1 shows the membership functions and interference functions $w(\cdot)$ and $n(\cdot)$ that generated the three distinct fuzzy LVQ algorithms.

An algorithmic description of the FALVQ is given below.

1. **Initialization:** Choose the number c of prototypes and a fixed learning

rate η_0 and the maximum number of iterations N. Set the iteration counter equal to zero, $\nu = 0$. Randomly generate an initial codebook $\mathbf{L} = \{\mathbf{L}_{1,0}, \cdots, \mathbf{L}_{c,0}\}$.

2. **Adaptation, part I:** Compute the updated learning rate $\eta = \eta_0 \left(1 - \frac{\nu}{N}\right)$. Also set $\nu = \nu + 1$.

3. **Adaptation, part II:** For each input vector $\mathbf{x}$ find the winning prototype based on the equation

$$||\mathbf{x} - \mathbf{L}_{i,\nu-1}||^2 < ||\mathbf{x} - \mathbf{L}_{j,\nu-1}||^2, \quad \forall j \neq i \tag{7.86}$$

Determine the membership functions $u_{ir,\nu}$ using

$$u_{ir,\nu} = u\left(\frac{||\mathbf{x} - \mathbf{L}_{i,\nu-1}||^2}{||\mathbf{x} - \mathbf{L}_{r,\nu-1}||^2}\right), \quad \forall r \neq i. \tag{7.87}$$

Determine $w_{ir,\nu}$ using

$$w_{ir,\nu} = u'\left(\frac{||\mathbf{x} - \mathbf{L}_{i,\nu-1}||^2}{||\mathbf{x} - \mathbf{L}_{r,\nu-1}||^2}\right), \quad \forall r \neq i. \tag{7.88}$$

Determine $n_{ir,\nu}$ using

$$n_{ir,\nu} = u_{ir,\nu} - \left(\frac{||\mathbf{x} - \mathbf{L}_{i,\nu-1}||^2}{||\mathbf{x} - \mathbf{L}_{r,\nu-1}||^2}\right) w_{ir,\nu}, \quad \forall r \neq i. \tag{7.89}$$

4. **Adaptation part III:** Determine the update of the winning prototype $\mathbf{L}_i$ using

$$\mathbf{L}_{i,\nu} = \mathbf{L}_{i,\nu-1} + \eta(\mathbf{x} - \mathbf{L}_{i,\nu-1})(1 + \sum_{r \neq i}^{c} w_{ir,\nu}) \tag{7.90}$$

Determine the update of the nonwinning prototype $\mathbf{L}_j \neq \mathbf{L}_i$ using

$$\mathbf{L}_{j,\nu} = \mathbf{L}_{j,\nu-1} + \eta(\mathbf{x} - \mathbf{L}_{j,\nu-1}) n_{ij,\nu}. \tag{7.91}$$

5. **Continuation:** If $\nu = N$, stop; else go to step 2.

7.5 Genetic Algorithms

Basic aspects and operations

Genetic algorithms (GA) are simple heuristic optimization tools for both continuous and discrete variables. These tools provide near-global optimal values even for poorly behaved functions. Compared to traditional optimization techniques, GAs have softer mathematical requirements by removing the restrictions on allowable models and error laws. In return, "softer" solutions to the optimization problem are provided that nevertheless are very good.

Their most important characteristics are the following:

- Parallel-search procedures: implementation on parallel-processing computers, ensuring fast computations.
- Stochastic nature: avoid local minima, and thus desirable for practical optimization problems.
- Applications: continuous and discrete optimization problems.

Genetic algorithms are, like neural networks, biologically inspired and are based on the application of the principles of "Darwinian natural selection" to a population of numerical representations of the solution domain. The natural evolution is emulated by allowing solutions to reproduce, creating offsprings of them, and allowing only the fittest to survive. Average fitness improves over generations, although some offsprings may not be improved compared to the previous generation, such that the best (fittest) solution is close to the global optimum.

Let's look again at the definition of a GA. In a strict sense, the classical GA is based on the original work of John Holland in 1975 [116]. This novel evolution-inspired paradigm - known also as the canonical genetic algorithm - is still a relevant research topic. In a more detailed sense, the GA represents a solution (population)–based model which employs selection, mutation, and recombination operators to generate new data points (offsprings) in a search space [282]. There are several GA models known in the literature, most of them designed as optimization tools for several applications in medical imaging. A very important one - the edge detection - will be reviewed in this chapter.

In summary, GAs differ from classical optimization and search procedures by (1) direct manipulation of a coding, (2) search from a pop-

Table 7.2
Definition analogies

Pattern recognition	Biology/genetics
vector, string	chromosome
feature, character	gene
feature, value	allele
set of all vectors	population

ulation of points and not a single solution, (3) search via sampling, a so–called blind search, and (4) search using stochastic operators, not deterministic rules.

Most of the definitions used in context with GAs have their roots in genetics but also have an equivalent in pattern recognition. For a better understanding, we can find those correspondents in table 7.2.

In the next section, we will review the basics of GAs such as encoding and mathematical operators, and describe edge detection in medical images based on GAs, as one of the most important applications of GAs.

Problem encoding and operators in genetic algorithms

The application of a GA as an optimization tool has three important parts: representation of solutions, operations that manipulate these solutions, and fitness selection.

If real solutions are required, these are represented as binary integers, which are mapped onto the real number axis. For example, for encoding solutions on the real interval $[-l, l]$, we will choose 0000...000 for $-l$ and 1111...111 for l. Adding a binary "1" to an existing number increases its value by $\frac{l}{2^{D-1}}$, where D is the length (number of digits) of the binary representation. Thus, an efficient coding is obtained, which enables bitwise operations.

In the beginning, a large initial population of random possible solutions is produced. The solution pool is continuously altered based on genetic operations such as selection and crossover. The selection is favorable to good solutions and punishes poor ones. To overcome convergence based on homogeneity resulting from excessive selection, and thus a local optimum, operations such as inversion and mutations are employed. They introduce diversity in the solution pool and prevent a local convergence.

These important and most common operators are the following [282]:

- **Encoding scheme:**
Transforms pattern vectors into bit string representations. Each coordinate value of a feature vector can be encoded as a binary string. Through an efficient encoding scheme, problem-specific knowledge is translated directly into the GA framework and implicitly influences the GA's performance.
- **Fitness evaluation:**
After the creation of a generation, fitness evaluation becomes important in order to provide the correct ranking information necessary for perpetuation. Usually, fitness of a member is related to the evaluation of the objective function of the point representing this member.
- **Selection:**
Based on *selection*, population members are chosen based on their fitness (the value of the objective function for that solution). The strings in the current population are copied in proportion to their fitness and placed in an intermediate generation. Selection enables the fittest genes to perpetuate, and guarantees the convergence of the population toward the desired solution.
- **Crossover:**
Crossover describes the swapping of fragments between two binary strings at a random position and combines the head of one with the tail of the other, and vice versa. Thus, two new offsprings are created and are inserted into the next population. In summary, new sample points are generated by recombining two parent strings. Consider the two strings 000101000 and 111010111. Using a single randomly chosen crossover point, recombination occurs as follows:

000|101000
111|010111.

The following offsprings are produced by swapping the fragments between the two parents:

000010111 and 111101000

This operator also guarantees the convergence of the population.

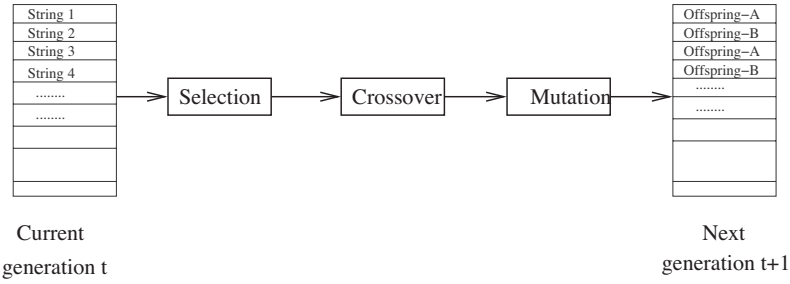


Figure 7.3
Splitting of a generation into a selection phase and a recombination phase.

• **Mutation:**
This operator does not represent a critical operation for GAs since many authors employ only selection and crossover. *Mutation* transforms the population by randomly changing the state (0 or 1) of individual bits. It prevents both an early convergence and a local optimum by creating divergence and inhomogeneity in the solution pool. In addition, new combinations are produced which lead to better solutions. Mutation is often performed after crossover has been applied, and should be employed with care. At most, one out of 1000 copied bits should undergo a mutation.

Apart from these very simple operations, many others emulating genetic reproduction have been proposed in the literature [176].

The application of a GA as an optimization techniques involves two steps: selection (duplication) and recombination (crossover). Initially, a large random population of random candidate solutions is generated. These solutions are continuously transformed by operations that model genetic reproduction: based on selection we obtain an intermediate population, and afterward based on recombination and mutation, we obtain the next population. The procedure of generating the next population from the current population represents one generation in the execution of a GA. Figure 7.3 visualizes this procedure [282].

An intermediate population is generated from the current population. In the beginning, the current population is given by the initial population. Then, every single string is evaluated and its *fitness value* is de-

termined. There is an important difference between the fitness function and the evaluation function in context with GAs: the *evaluation function* represents a performance measure for a particular set of parameters, while the *fitness function* gives the chance of reproductive opportunities based on the measured performance. Thus, the fitness function defines the criterion for ranking potential hypotheses and for probabilistically selecting them for inclusion in the population of the next generation. While the evaluation of a string describing a particular set of parameters is not related to any other string evaluation, the fitness of that string is related to the other strings of the current population. Thus, the probability that a hypothesis is chosen is directly proportional to its own fitness, and inversely proportional to the rest of the competing hypotheses for the given population.

For canonical GAs the definition of the fitness is given by $f_i/\bar{f}$, where f_i is the evaluation associated with string i and $\bar{f}$ is the average evaluation of all strings in the population.

$$\bar{f} = \frac{1}{n} \sum_{i=1}^{n} f_i. \tag{7.92}$$

As stated before, after generating the initial population, the fitness $f_i/\bar{f}$ for all members of the current population is evaluated, and then the selection operator is employed. Members of the population are copied or duplicated proportional to their fitness and then entered in the intermediate generation. If for a string i, we obtain $f_i/\bar{f} > 1.0$, then the integer portion of fitness determines the number of copies of this string that enter directly int the intermediate population. A string with a fitness of $f_i/\bar{f} = 0.69$ has a 0.69 chance of placing one string in the intermediate population, and a string with a fitness of $f_i/\bar{f} = 1.38$ places one copy in the intermediate population. The selection process continues until the intermediate population is generated.

Next the recombination operator is carried out as a process of generating the next population from the intermediate population. Then crossover is applied and models the exchange of genetic material between a pair of strings. These strings are recombined with a probability of p_c, and the newly generated strings are included in the next population. The mutation is the last operator needed for producing the next population. Its goal is to maintain diversity and to introduce new alleles

into the generation. The mutation probability of a bit p_m is very small, usually $p_m \ll 1\%$. For practical applications, we normally choose p_m close to 0.01. Mutation changes the bit values, and produces a nearly identical copy with some components of the string altered. Selection, recombination, and mutation operators are applied to each population in each generation. The GA stops either when a satisfactory solution is found or after a predefined number of iterations.

The algorithmic description of a GA is given below.

Generate the initial population randomly for the strings a_i:
$\Pi = \{a_i\}$, $i = 1, \cdots, n$.
for $i \leftarrow 1$ *to Numberofgenerations* **do**
  Initialize mating set $M \leftarrow \varnothing$ and Offspring O
  for $j \leftarrow 1$ *to* n **do**
    Add $f(a_i)/\bar{f}$ copies from a_i to M.
  end
  for $j \leftarrow 1$ *to* $n/2$ **do**
    Choose two parents a_j and a_k from M and perform with the probability p_c $O = O \cup Crossover(a_j, a_k)$.
  end
  for $i \leftarrow 1$ *to* n **do**
    for $j \leftarrow 1$ *to* d **do**
      Mutate with the probability p_m the j-th bit from $a_i \in O$
    end
  end
  Update the population $\Pi \leftarrow \text{combine}(\Pi, O)$.
end

It is extremely important to mention that the theoretical basis for convergence of the GA toward the global maximum is based on the *schema theorem*. The formation and preservation of a schema which is a local optimal pattern should happen at rates acceptable for solving problems in practice. While we have been dealing so far with only binary strings, schemas represent bit patterns based on a ternary alphabet: 0, 1, and $*$ (do not care). Thus, a crossover operation enables information sharing between two optimal schemas such that new and better solutions are generated.

Finally, we would like to point out the analogy between the traditional optimization approach and GAs. The binary strings correspond to an orthogonal direction system, crossovers to moving randomly at the same time in multiple directions from one point to another of the surface, and mutation to searching along a single, randomly chosen direction.

Optimization of a simple function

A GA represents a general-purpose optimization method that searches irregular, poorly characterized function spaces and is easily implemented on parallel computers. The performance of the solutions is continuously tested based on a fitness function. It's not always guaranteed that an optimal candidate is found, but in most cases GAs do find a candidate with high fitness. An important application area for GAs is pattern recognition: the highly nonlinear problem of estimating the weights in a neural network.

This section will apply the most important basic operations of a GA to an example of function optimization [50].

The following function is considered:

$$g(x) = x^2 - 42x + 152$$

where x is an integer. The goal is to find, based on a GA, the minimum of this function in the interval $[0 \cdots 63]$:

$$g(x_0) \leq g(x), \quad \text{for all} \quad x \in [0 \cdots 63].$$

To solve this optimization problem, some typical GA operators are employed.

Number representation

The integer-valued x have to be transformed into a binary vector (chromosome). Since $2^6 = 64$, we will use six-bit binary numbers to represent the solutions. This means that six bits are needed to represent a binary vector (chromosome).

The transformation of a binary number $< b_5 \cdots b_0 >$ into an integer number x is done by the following rule:

Transform the binary number $< b_5 \cdots b_0 >$ from basis 2 into basis 10:

$$(< b_5 \cdots b_0 >)_2 = (\sum_{i=0}^{5} b_i \cdot 2^i)_{10} = x$$

Initial population

The initial population is randomly generated. Each chromosome represents a six-bit binary vector.

Evaluation function

The evaluation function f of the binary vector $\mathbf{v}$ is equivalent to the function $g(x)$:

$$f(\mathbf{v}) = g(x).$$

The five given x-values $x_1 = 37$, $x_2 = 13$, $x_3 = 35$, $x_4 = 44$, and $x_5 = 6$ correspond to the following five chromosomes:

$\mathbf{v}_1 = (100110)$,
$\mathbf{v}_2 = (001101)$,
$\mathbf{v}_3 = (100011)$,
$\mathbf{v}_4 = (101110)$,
$\mathbf{v}_5 = (000110)$

The evaluation function provides the following values:

$$\begin{aligned} f(\mathbf{v}_1) = g(x_1) &= 0 \\ f(\mathbf{v}_2) = g(x_2) &= -225 \\ f(\mathbf{v}_3) = g(x_3) &= -93 \\ f(\mathbf{v}_4) = g(x_4) &= 336 \\ f(\mathbf{v}_4) = g(x_5) &= -64. \end{aligned}$$

We immediately see that $\mathbf{v}_2$ is the fittest chromosome since its evaluation function provides the minimal value.

Genetic operators

While the GA is executed, three distinct operators are employed to change the chromosomes: selection, mutation, and crossover.

We randomly choose the first and fourth chromosomes for selection. Since $f(\mathbf{v}_1) < f(\mathbf{v}_4)$, chromosome 4 will be replaced by chromosome 1. After five other random selections, we obtain the following values:

$$\begin{aligned} f(\mathbf{v}_1) &= g(x_1) = 0 \\ f(\mathbf{v}_2) &= g(x_2) = -225 \\ f(\mathbf{v}_3) &= g(x_3) = -93 \\ f(\mathbf{v}_4) &= g(x_4) = -225 \\ f(\mathbf{v}_5) &= g(x_5) = -93. \end{aligned}$$

As we see, no new solutions were produced and the fittest solution was perpetuated.

Next, we randomly choose chromosome 1 and 4 for crossover at the fourth gene and obtain the following solutions:

$$\begin{aligned} f(\mathbf{v}_1) &= g(x_1) = 287 \\ f(\mathbf{v}_2) &= g(x_2) = -225 \\ f(\mathbf{v}_3) &= g(x_3) = -93 \\ f(\mathbf{v}_4) &= g(x_4) = -64 \\ f(\mathbf{v}_5) &= g(x_5) = -93. \end{aligned}$$

After undergoing four pairs of crossing, we obtain:

$$
\begin{aligned}
f(\mathbf{v}_1) &= g(x_1) = 285 \\
f(\mathbf{v}_2) &= g(x_2) = -273 \\
f(\mathbf{v}_3) &= g(x_3) = -288 \\
f(\mathbf{v}_4) &= g(x_4) = -285 \\
f(\mathbf{v}_5) &= g(x_5) = -33.
\end{aligned}
$$

Next, we apply mutation and randomly suppose chromosome 3 and bit 6 are chosen. Thus, the mutated chromosome is 010101 and gives a further improvement of f to -288.

Simulation parameters

To determine the solution of the given optimization problem, we will choose the following parameters: the population consists of 100 distinct chromosomes, and we choose 5950 random pairs for selection.

Simulation results

The results achieved after one cycle, including the above-mentioned operators, are the following:

$$
\begin{aligned}
f(\mathbf{v}_1) &= g(x_1) = 285 \\
f(\mathbf{v}_2) &= g(x_2) = -289 \\
f(\mathbf{v}_3) &= g(x_3) = -288 \\
f(\mathbf{v}_4) &= g(x_4) = -285 \\
f(\mathbf{v}_5) &= g(x_5) = -33.
\end{aligned}
$$

The best value is $x_{min} = 21$. We can show that the GA converges toward the minimum of the given function. The fact that this solution is reached is more a coincidence than a property of the GA. It's important to emphasize that a GA may not find an exact optimal solution, but most often finds solutions close to the neighborhood of the global optimum.

As a final remark, it's very important to mention that GAs can be very well applied in combinatorial optimization where the decision vari-

ables are integer or mixed. We have seen that problems with integer variables can be reduced to those of 0 and 1 binary variables. Thus, we are left with problems with 0 and 1 binary variables. Many optimization problems, such as the traveling salesman problem, are NP-complete problems, and there are both heuristics and exact solutions available, although they are considered to be unsolved problems in their generality. The variable selection problem thus becomes very interesting, not only for theoretical reasons. In life sciences, such situations occur very frequently: there is a large number of candidate variables and a known condition (y=1 or 0) where the data may be not completely known. For example, the problem of locating homologies in the human genome represents an important discrete choice problem.

Edge detection using a genetic algorithm

Most edge detection algorithms applied to medical images perform satisfactorily when applied for a certain anatomical structure, but cannot be generalized to other modalities or anatomical structures. This motivates the search for an efficient algorithm to overcome these drawbacks. GAs are optimal and robust candidates since they are not affected by spurious local optima in the solution space.

A GA can be used to detect well-localized, unfragmented, thin edges in medical images based on optimization of edge configurations [103].

An edge structure is defined within a 3×3 neighborhood $W_{ij}(S)$ around a single center pixel $l = s(i, j)$ in $S \in \mathbf{S}$, where $\mathbf{S}$ represents the set of all possible edge configurations in an image I.

The total cost for an edge configuration $S \in \mathbf{S}$ is the sum of the point costs at every pixel in an image:

$$F(S) = \sum_{l \in I} F(S, l) = \sum_{l \in I} \sum_{j} w_j c_j(S, l). \tag{7.93}$$

c_j consists of the five cost factors: the dissimilarity cost C_d, the curvature cost C_c, the edge pixel cost C_e, the fragmentation cost C_f, and the cost for thick edges C_t. w_j represents the corresponding weights w_d, w_c, w_e, w_f, w_t employed for optimizing the shape of the edges.

Edge detectors can be imagined as edges in binary images where edge pixels are assigned the value of 1 and nonedge pixels have the value of 0. Thus, there is an orientation and adjacency-preservation map between

Table 7.3
Approximation of the size of the search space, assuming independent subregions [103].

Size	No. of Regions	No. of Combinations	Search Space
256×256	1	2^{65536}	$> 10^{19728}$
128×128	4	2^{16364}	$> 4 \cdot 10^{4932}$
64×64	16	2^{4096}	$> 10^{1234}$
32×32	64	2^{1024}	$> 10^{310}$
16×16	256	2^{256}	$> 10^{79}$
8×8	1024	2^{64}	$> 10^{22}$
4×4	40960	2^{16}	$> 10^{8}$

the binary edge image and the original one.

The search and solution space for the edge-detection problem is huge as shown in table 7.3. The table shows, for different image sizes, the number of combinations and the corresponding search space. In order to reduce the sample space and simplify the optimization problem, the original image has to be split into linked regions.

It has been shown in [103] that the GA for edge detection works best for regions sized 4×4 and larger. Thus, for each subregion we have a single independent GA which tries to optimize the edge configuration within the subregion.

Pratt's figure of merit [211] provides a quantitative comparison of the results of different edge detectors by measuring the deviation of the output edge from a known ideal edge:

$$P = \frac{1}{max(I_A, I_I)} \sum_{i=1}^{I_A} \frac{1}{1 + \alpha d^2(i)} \tag{7.94}$$

with I_A being the number of detected edge points, I_I the edge points in the ideal image, α a scaling factor, and $d(i)$ the distance of the detected edge pixel from the nearest ideal edge position. Thus, Pratt's figure of merit represents a rough indicator of edge quality in the sense that a higher value denotes a better edge image.

The results shown in [103] demonstrate that GA improved Pratt's figure of merit from 0.77 to 0.85 for ideal images and detected most of the basic edge features (thin, continuous, and well-localized) for MR, CT, and US images.

EXERCISES

1. Suppose that fuzzy set A is described by the membership function $u_A(x)$,

$$u_A(x) = bell(x; a, b, c) = \frac{1}{1 + |\frac{x-c}{x}|^{2b}}, \tag{7.95}$$

where the parameter b is usually positive. Show that the classical complement of a is given as $u_{\bar{A}}(x) = bell(x; a, -b, c)$.

2. Derive the Gath-Geva algorithm based on the distance metric.

3. Derive the update of the winning and nonwinning prototypes for the FALVQ algorithm.

4. Consider the function

$$g(x) = 31.5 + x|sin(4\pi x)|.$$

Find the maximum of this function in the interval $[-4 \cdots 22.1]$ by employing a GA.

5. Apply the GA to determine an appropriate set of weights for a $4 \times 2 \times 1$ multilayer perceptron. Encode the weights as bit strings, and apply the required genetic operators. Discuss how the backpropagation algorithm differs from a GA regarding the weights' learning.

6. Consider the function

$$J = \sum_{i=1}^{N} d^2(\mathbf{x}, C_{\mathbf{v_i}})$$

where $d(\mathbf{x}, C_{\mathbf{v_i}})$ describes the distance between an input vector $\mathbf{x}$ and a set using no representatives for the set. Propose a coding of the solutions for a GA that uses this function. Discuss the advantages and disadvantages of this coding.

II APPLICATIONS

8 Exploratory Data Analysis Methods for fMRI

Functional magnetic resonance imaging (fMRI) has been shown to be an effective imaging technique in human brain research [188]. By blood oxygen level- dependent contrast (BOLD), local changes in the magnetic field are coupled to activity in brain areas. These magnetic changes are measured using MRI. The high spatial and temporal resolution of fMRI combined with its noninvasive nature makes it an important tool for discovering functional areas in the human brain and their interactions. However, its low signal-to-noise ratio and the high number of activities in the passive brain require a sophisticated analysis method. These methods either (1) are based on models and regression, but require prior knowledge of the time course of the activations, or (2) employ model-free approaches such as BSS by separating the recorded activation into different classes according to statistical specifications without prior knowledge of the activation.

The blind approach (2) was first studied by McKeown et al. [169]. According to the principle of functional organization of the brain, they suggested that the multifocal brain areas activated by performance of a visual task should be unrelated to the brain areas whose signals are affected by artifacts of a physiological nature, head movements, or scanner noise related to fMRI experiments. Every single process can be described by one or more spatially independent components, each associated with a single time course of a voxel and a component map. It is assumed that the component maps, each described by a spatial distribution of fixed values, represent overlapping, multifocal brain areas of statistically independent fMRI signals. This is visualized in figure 8.1.

In addition, McKeown et al. [169] considered the distributions of the component maps to be spatially independent and in this sense uniquely specified (see section 4.2). They showed that these maps are independent if the active voxels in the maps are sparse and mostly nonoverlapping. Additionally, they assumed that the observed fMRI signals are the superpositions of the individual component processes at each voxel. Based on these assumptions, ICA can be applied to fMRI time series to spatially localize and temporally characterize the sources of BOLD activation. Considerable research has been devoted to this area since the late 1990s.

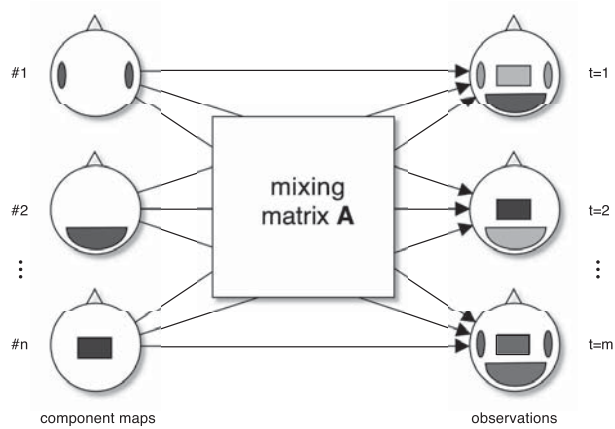
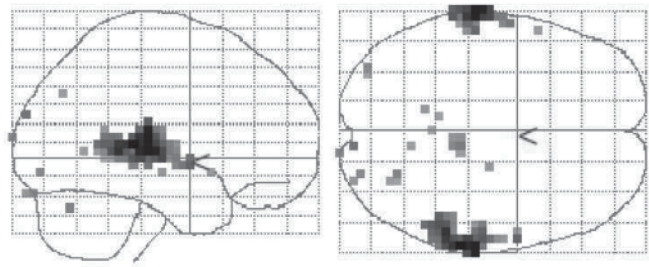


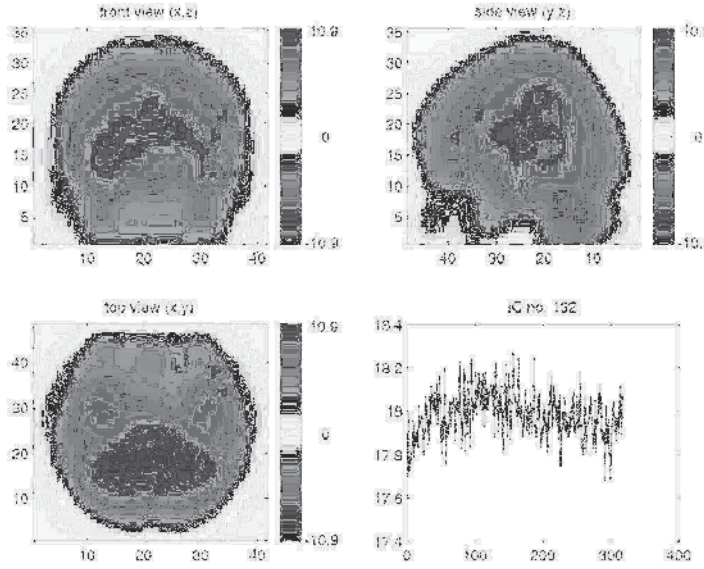
Figure 8.1
Visualization of the spatial fMRI separation model. The n-dimensional source vector is represented as component maps, which are interpreted as contributing linearly in different concentrations to the fMRI observations at the time points $t \in \{1, \ldots, m\}$. See plate 2 for the color version of this figure.

8.1 Model-based Versus Model-free Analysis

However, the use of blind signal-processing techniques for the effective analysis of fMRI data has often been questioned, and in many applications, neurologists and psychologists prefer to use the computationally simpler regression models. In [135], these two approaches are compared using a sufficiently complex task of a combined word perception and motor activity. The event-based experiment was part of a study to investigate the network of neurons involved in the perception of speech and the decoding of auditory speech stimuli. One- and two-syllable words were divided into several frequency bands and then rearranged randomly to obtain a set of auditory stimuli. Only a single band was perceivable as words. During the functional imaging session these stimuli were presented pseudo–randomized to five subjects, according to the rules of a stochastic event-related paradigm. The task of the subjects was to press a button as soon as they were sure that they had just recognized a word in the sound presented. It was expected that in the case of the single perceptible frequency band, these four types of stimuli activate different areas of the auditory system as well as the superior temporal sulcus in the left hemisphere [236].



(a) general linear model analysis



(b) one independent component

Figure 8.2
Comparison of model-based and model-free analyses of a word-perception fMRI experiment. (a) illustrates the result of a regression-based analysis, which shows activity mostly in the auditory cortex. (b) is a single component extracted by ICA and corresponds to a word-detection network. See plate 3 for the color version of this figure.

The regression-based analysis using a general linear model was performed using SPM2. This was compared with components extracted using ICA, namely fastICA [124].

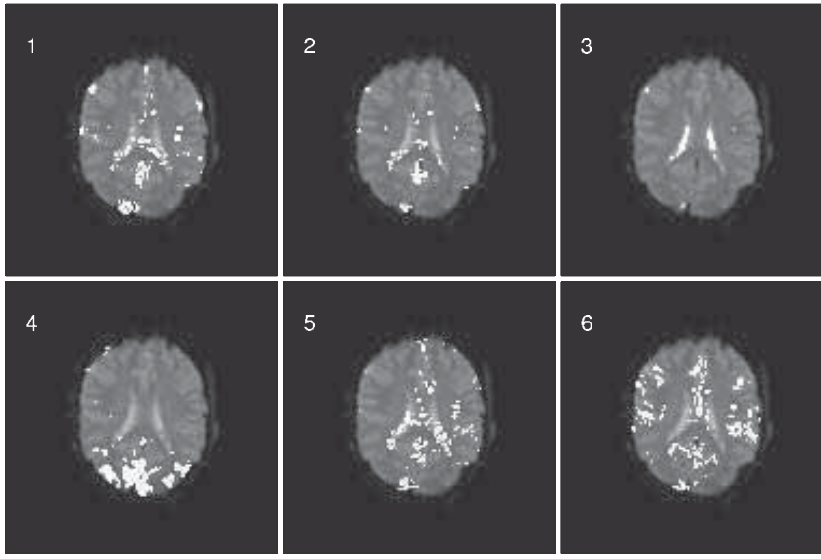
The results are illustrated in figure 8.2, and are explained in more detail in [135]. Indeed, one independent component represented a network of three simultaneously active areas in the inferior frontal gyrus, which was previously proposed to be a center for the perception of speech [236]. Altogether, we were able to show that ICA detects hidden or suspected links and activity in the brain that cannot be found using the classical, model-based approach.

8.2 Spatial and Spatiotemporal Separation

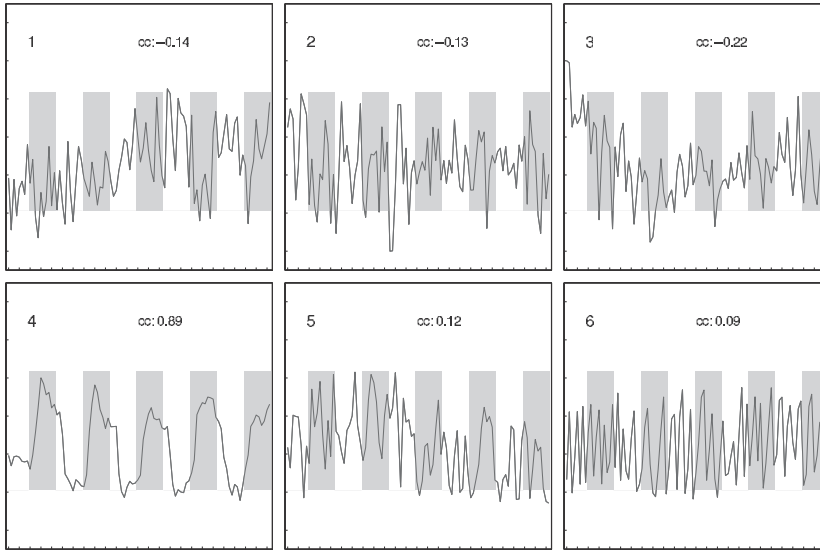
As short example of spatial and spatiotemporal BSS, we present the analysis of an experiment using visual stimuli. fMRI data were recorded from 10 healthy subjects performing a visual task. One hundred scans were acquired from each subject with five periods of rest and five photic stimulation periods, and a resolution of $3 \times 3 \times 4$ mm. A single 2-D slice, which is oriented parallel to the calcarine fissure, is analyzed. Photic stimulation was performed using an 8 Hz alternating checkerboard stimulus with a central fixation point and a dark background.

First, we show an example result using spatial ICA. We performed a dimension reduction using PCA to $n =$ six dimensions, which still contained 99.77% of the eigenvalues. Then we applied HessianICA with $K = 100$ Hessians evaluated at randomly chosen samples (see section 4.2 and [246]). The resulting six-dimensional sources are interpreted as the six component maps that encode the data set. The columns of the mixing matrix contain the relative contribution of each component map to the mixtures at the given time point, so they represent the components' time courses. The maps and the corresponding time courses are shown in figure 8.3. A single highly task-related component (#4) is found, which after a shift of 4s has a high crosscorrelation with the block-based stimulus ($cc = 0.89$). Other component maps encode artifacts (e.g., in the interstitial brain region) and other background activity.

We then tested the usefulness of taking into account additional information contained in the data set such as the spatiotemporal dependencies. For this, we analyzed the data using spatiotemporal BSS as described in chapter 5 (see [253, 255]). In order to make things more challenging, only four components were to be extracted from the data, with preprocessing either by PCA only or by the slightly more gen-



(a) recovered component maps



(b) time courses

Figure 8.3
Extracted ICA components of fMRI recordings. (a) shows the spatial, and (b) the corresponding temporal, activation patterns, where in (b) the gray bars indicate stimulus activity. Component 4 contains the (independent) visual task, active in the visual cortex (white points in (a)). It correlates well with the stimulus activity (b).

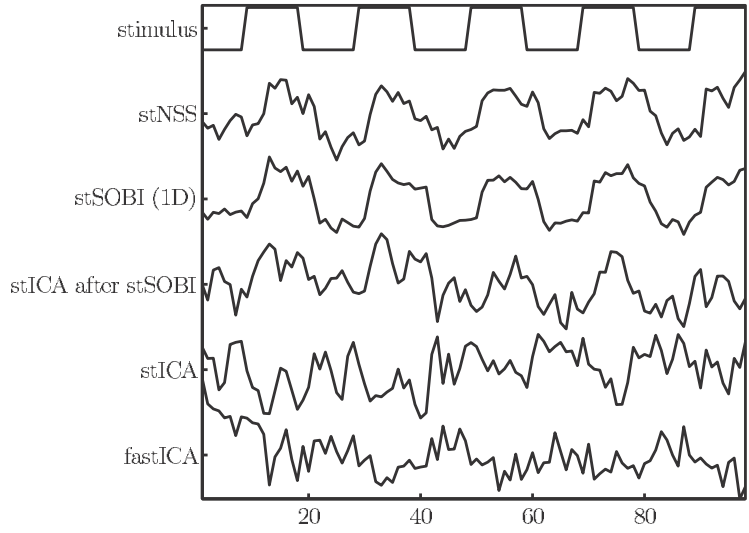


Figure 8.4
Comparison of the recovered component that is maximally auto-crosscorrelated with the stimulus task (top) for various BSS algorithms, after dimension reduction to four components.

eral singular value decomposition, a necessary preprocessing for spatiotemporal BSS. We based the algorithms on joint diagonalization, for which $K = 10$ autocorrelation matrices were used, for both spatial and temporal decorrelation, weighted equally ($\alpha = 0.5$). Although the data were reduced to only four components, stSOBI was able to extract the stimulus component very well, with a equally high crosscorrelation of $cc = 0.89$. We compared this result with some established algorithms for blind fMRI analysis by discussing the single component that is maximally autocorrelated with the known stimulus task (see figure 8.4). The absolute corresponding autocorrelations are 0.84 (stNSS), 0.91 (stSOBI with one-dimensional autocorrelations), 0.58 (stICA applied to separation provided by stSOBI), 0.53 (stICA), and 0.51 (fastICA). The observation that neither Stone's spatiotemporal ICA algorithm [241] nor the popular fastICA algorithm [124] could recover the sources showed that spatiotemporal models can use the additional data structure efficiently, in contrast to spatial-only models, and that the parameter-free joint-diagonalization-based algorithms are robust against convergence issues.

8.3 Other Analysis Models

Before continuing to other biomedical applications, we briefly want to review other recent work of the authors in this field.

The concept of window ICA can be used for the analysis of fMRI data[133]. The basic idea is to apply spatial ICA in sliding time windows; this approach avoids the problems related to the high number of signals and the resulting issues with dimension reduction methods. Moreover, it gives some insight into small changes during the experiment which are otherwise not encoded in changes in the component maps. We demonstrated the usefulness of the proposed approach in an experiment where a subject listened to auditory stimuli consisting of sinusoidal sounds (beeps) and words in varying proportions. Here, the window ICA algorithm was able to find different auditory activation patterns related to the beeps (respectively, the words).

An interesting model for activity maps in the brain is given by sparse coding; after all, the component maps are always implicitly assumed to show only strongly focused regions of activation. Hence we asked whether specific sparse modeling approaches could be applied to fMRI data. We showed a successful application to the above visual-stimulus experiment in [90]. Again, we were able to show that with only five components, the stimulus-related activity in the visual cortex could be nicely reconstructed.

A similar question of model generalization was posed in [263]. There we proposed to study the post-nonlinear mixing model in the context of fMRI data. We derived an algorithm for blindly estimating the sensor characteristics of such a multisensor network. From the observed sensor outputs, the nonlinearities are recovered using a well-known Gaussianization procedure. The underlying sources are then reconstructed using spatial decorrelation as proposed by Ziehe et al. [296]. Application of this robust algorithm to data sets acquired through fMRI leads to the detection of a distinctive bump of the BOLD effect at larger activations, which may be interpreted as an inherent BOLD-related nonlinearity.

The concept of dependent component analysis (see chapter 5) in the context of fMRI data analysis is discussed in [174], [175]. It can be shown that dependencies can be detected by finding clusters of dependent components; algorithmically, it is interesting to compare this with tree-dependent [12] and topographic ICA [122]. For the fMRI data, a

comparative quantitative evaluation of tree–dependent and topographic ICA was performed. We observed that topographic ICA outperforms other ordinary ICA methods and tree–dependent ICA when extracting only a few independent components. This resulted in a postprocessing algorithm based on clustering of ICA components resulting from different source-component dimensions [134].

The above algorithms have been included in our MFBOX (Model-free Toolbox) package [102], a Matlab toolbox for data-driven analysis of biomedical data, which may also be used as an SPM plug-in. Its main focus is on the analysis of functional nuclear magnetic resonance imaging (fMRI) data sets with various model-free or data-driven techniques. The toolbox includes BSS algorithms based on various source models including ICA, spatiotemporal ICA, autodecorrelation, and NMF. They can all be easily combined with higher-level analysis methods such as reliability analysis using projective clustering of the components, sliding time window analysis, and hierarchical decomposition.

The time-series analysis employed for fMRI signal processing also forms also the basis for a general MRI signal processing as described in chapters 9, 10, and 11. There, exploratory data analysis techniques are applied to resting-state fMRI data, the diagnosis of dynamic breast MR data and the detection of cerebral infarctions based on perfusion MRI.

9 Low-frequency Functional Connectivity in fMRI

Low-frequency fluctuations (< 0.08 Hz) temporally correlated between functionally related areas have been reported for the motor, auditory, and visual cortices and other structures [35]. The detection and quantification of these patterns without user bias poses a current challenge in fMRI research. Many recent studies have shown decreased low-frequency correlations for subjects in pathological states or in the case of cocaine use [199], which can potentially indicate normal neuronal activity within the brain.

The standard technique for detecting low-frequency fluctuations has been the crosscorrelation method. However, it has several drawbacks, such as sensitivity to data drifts and choosing the reference waveform when no external paradigm is present. The use of prespecified regions of interest (ROI) or "seed clusters" has been the method of choice in functional connectivity studies [35], [199]. The main limitation of this method is that it is user-biased.

Model-free methods that have recently been applied to fMRI data analysis include projection-based and clustering-based. The first method, PCA [14, 242] and ICA [10, 77, 168, 170] extracts several high-dimensional components from original data to separate functional response and various noise sources from each other. The second method, fuzzy clustering analysis [24, 53, 226, 285] or the self-organizing map [84, 185, 285], attempts to classify time signals of the brain into patterns according to temporal similarity among these signals.

Recently, self-organizing maps (SOM) have been applied to the detection of resting-state functional connectivity [199]. It has been shown that the SOM represents an adequate model-free analysis method for detecting functional connectivity.

The present chapter elaborates this interesting idea and introduces several unsupervised clustering methods implementing arbitrary distance metrics for the detection of low-frequency connectivity of the resting human brain. These techniques allow the detection of time courses of low-frequency fluctuations in the resting brain that exhibit functional connectivity with time courses in several other regions which are related to motor function. The results achieved by these approaches are compared to standard model-based techniques.

9.1 Imaging Protocol

fMRI data were recorded on a 1.5 T scanner (Magnetom Vision, Siemens, Erlangen, Germany) from four subjects (three males and one female, between the ages of 25 and 28) with no history of neurological disease. The sequence acquired 512 images (TR/TE=500/40 msec). Two 10.0-mm-thick axial slices were acquired in each TR, with an in-plane resolution of 1.37×1.37 mm.

The four subjects were studied under conditions of activation and rest. Two separate data sets, one a task-activation set and one a resting-state set, were acquired for each subject. During the resting-state collection, the subjects were told to refrain from any cognitive, language, or motor task. For the task-activation set, a sequential finger-tapping motor paradigm (20.8-sec fixation, 20.8-sec task, 6 repeats) was performed. The slices were oriented parallel to the calcarine fissure.

9.2 Postprocessing and Exploratory Data Analysis Methods

Motion artifacts were compensated for by automatic image registration (AIR, [288]). To remove the effect of signal drifts stemming from either the scanner and/or physiological changes in the subjects, linear detrending was employed. In addition, for the resting-state data, the time courses were filtered with a low-pass filter having a cutoff frequency of 0.08 Hz. Thus, the influence of respiratory and cardiovascular oscillations was avoided while preserving the frequency spectrum pertaining to functional connectivity [35]. The time courses were further normalized in order to focus on signal dynamics rather than amplitude. See discussion in [285] on this issue.

The following unsupervised clustering techniques are presented and evaluated: topographic mapping of proximity, minimum free energy neural network, fuzzy clustering, and Kohonen's self-organizing map. These techniques have in common that they group pixels together based on the similarity of their intensity profile in time (i.e., their time courses).

Let n denote the number of sequential scans in an fMRI study, and let K be the number of pixels in each scan. The dynamics of each pixel $\mu \in \{1, \ldots, K\}$ can be interpreted as a vector $\mathbf{x}^\mu \in \mathbf{R}^n$ in the n-dimensional feature space of possible signal time series. In the following,

the pixel-dependent vector $\mathbf{x}^{\mu}$ will be called a pixel time course (PTC).

Here, several vector quantization (VQ) approaches are employed as a method for unsupervised time series analysis. VQ clustering identifies several groups of pixels with similar PTCs, and these groups or clusters are represented by prototypical time series called codebook vectors (CV) located at the center of their corresponding cluster. The CVs represent prototypical PTCs sharing similar temporal characteristics. Thus, each PTC can be assigned in the crisp clustering scheme to one specific CV according to a minimal distance criterion, and in the fuzzy scheme according to membership to several CVs. Accordingly, the outcomes of VQ approaches for fMRI data analysis can be plotted as "crisp" or "fuzzy" cluster assignment maps.

Besides the more traditional VQ approaches, a soft topographic vector quantization algorithm is employed here which supports the topographic mapping of proximity (TMP) data [98]. This algorithm can be seen as an extension of Kohonen's self-organizing map to arbitrary distance measures. The TMP processes the data based on a dissimilarity matrix, and the topographic neighborhood by a matrix of transition probabilities. A detailed mathematical derivation can be found in [98]. This algorithm is employed in connection with two different distance measures, the linear crosscorrelation between the time courses, which is refered to as TMP^{corr}, and also in connection with the nonlinear prediction error between time courses, which is refered to as TMP^{pred}. The nonlinear prediction error between time courses is determined by a generalized radial-basis function (GRBF) neural network [179, 208]. For the fuzzy c-means vector quantization, two different implementations are employed: fuzzy c-means with unsupervised codebook initialization (FSM), and the fuzzy c-means algorithm (FVQ) with random codebook initialization.

9.3 Cluster Analysis of fMRI Data Sets Under Motor Stimulation

This section describes the simulation results obtained with unsupervised clustering methods during the activation state of the finger-tapping motor paradigm.

The first objective is to demonstrate the applicability of the TMP

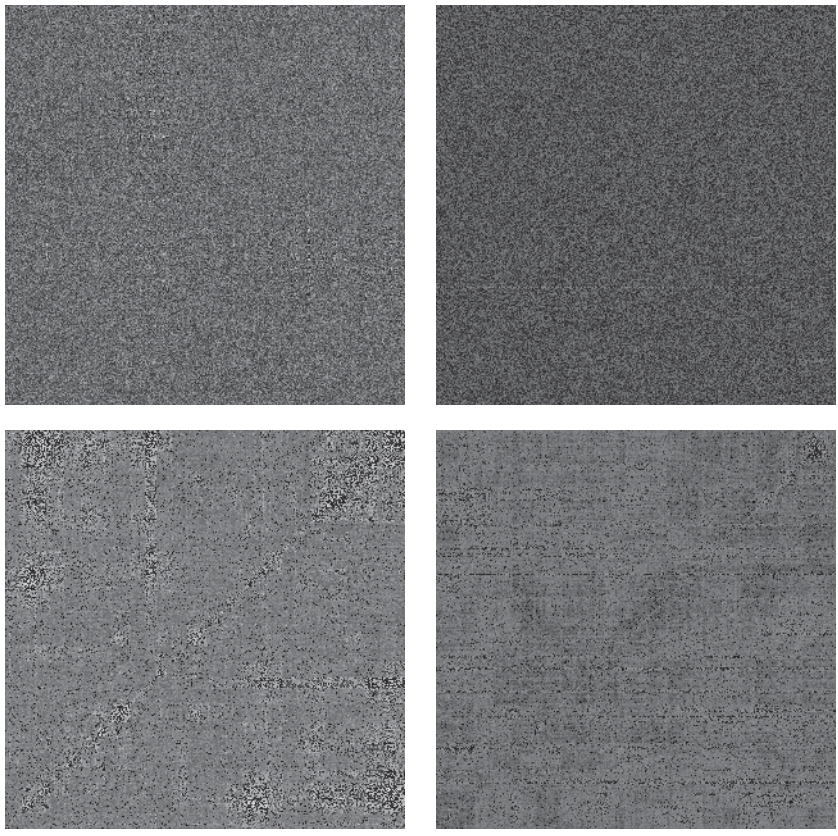
algorithm to the partitioning of fMRI data. In a following step, a comparison between the unsupervised algorithms implementing different distance metrics is performed.

The TMP algorithm determines the mutual pairwise similarity between the PTCs, which leads to an important issue in fMRI data analysis: What is the underlying basic similarity measure between the PTCs? Two approaches described in the exploratory data analysis part are employed: the TMP^{corr} considering the correlation between the PTCs and the TMP^{pred} considering the prediction error.

Figure 9.1 visualizes the computed distance matrices for subject #1 and for $N = 25$ clusters based on both the correlation and the prediction error methods. The first row shows the unsorted distance matrices and the second row shows the results obtained after application of the TMP algorithm, resulting in a display of the distance matrix, where the rows and columns appear in an ordered fashion. The emerging block-diagonal structure reflects the characteristic of the TMP algorithm to cluster PTCs based on their mutual dependency (i.e., their pairwise distance).

By taking the average value of all PTCs belonging to a certain cluster, a cluster-representative PTC is obtained. Figure 9.2 shows a comparison of the segmentation results obtained by the unsupervised clustering methods for subject #1. The cc-cluster describes a method based on the threshold segmentation of the correlation map. This map assigns to each pixel the Pearson correlation coefficient between the PTC and the stimulus function. The threshold was chosen as $\Delta = 0.6$, and thus every pixel with a correlation of its PTC exceeding 0.6 is considered to be activated and is white on the map. For the clustering methods, all the clusters with an average correlation of PTCs above the threshold of $\Delta = 0.6$ are collected and their pixels are plotted white on the map. The average value of all PTCs belonging to a certain segmentation determines a segmentation-specific PTC shown under the assignment maps. A high correlation of these representative PTCs with the stimulus function $cc = 0.75$ is found exceeding for all methods.

It is important to perform a quantitative analysis of the relative performance of the introduced exploratory data analysis techniques for all four subjects. To do so, the proposed algorithms are compared for 9, 16, and 25 clusters in terms of ROC analysis using a correlation map with a chosen threshold of 0.6 as the reference. The ROC performances for the four subjects are shown in figure 9.3. The figure illustrates the average



(a) (b)

Figure 9.1
Distance matrices with distances represented by gray values, with $N = 25$ clusters used for the analysis of the motor stimulation data set of subject #1. Distances are determined based on the correlation method (a) and the prediction error method (b). The upper and lower rows show the matrices before and after applying the TMP algorithm, respectively. The dissimilarity matrices were plotted such that the rows from bottom to top and the columns from left to right correspond to increasing indices of the PTCs. The block-diagonal structure of the ordered distance matrices becomes evident. The dark lines represent the cluster borders and are overlaid onto the distance matrices. Small distances are plotted dark, representing close proximity.

area under the curve and its deviations for 20 different ROC runs for each algorithm, using the same parameters but different initializations.

From this figure, it can be seen that all clustering methods achieve

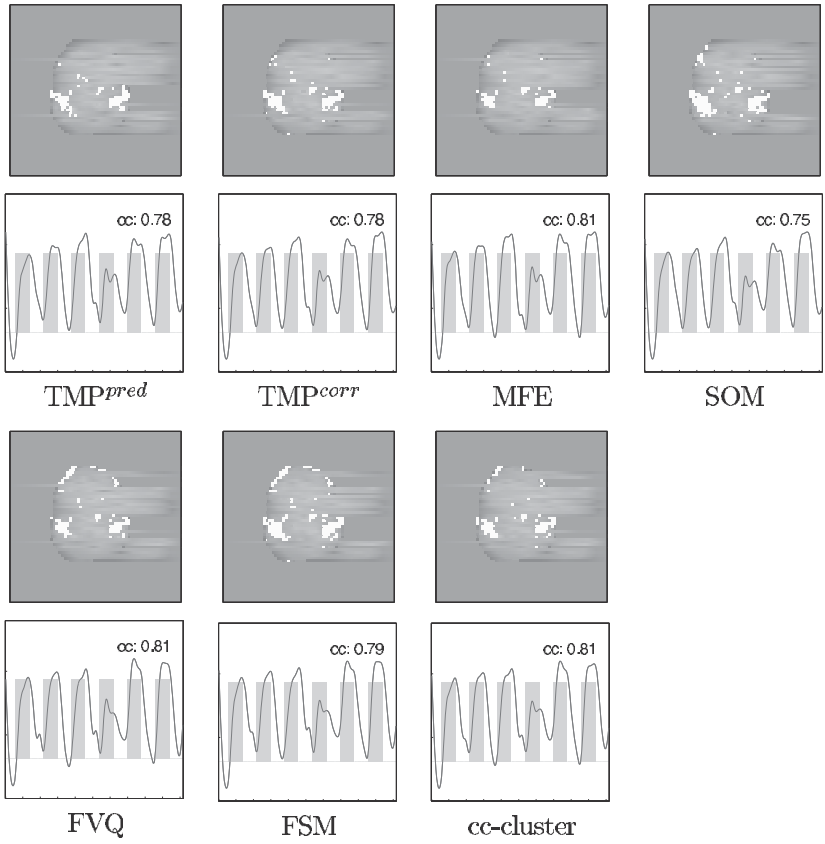


Figure 9.2
Segmentation results in the motor areas of subject #1 in the motor stimulation experiment. The obtained task activation maps are shown for all unsupervised methods. For comparison, the cc-cluster describes a method based on the threshold segmentation of a pixel-specific correlation map. This map assigns to each pixel the Pearson correlation coefficient between the PTC and the stimulus function. The threshold was chosen as $\Delta = 0.6$ and thus every pixel correlation exceeding 0.6 is considered as activated and is colored white on the map. For the clustering methods, all the clusters with an average correlation of PTCs above the threshold of $\Delta = 0.6$ are collected and their pixels are plotted white on the map. The average value of all PTCs belonging to a certain segmentation determines a segmentation-specific PTC shown under the assignment maps. The motor task reference waveform is given as a square wave and overlaid on the average PTC.

good results expressed by an area A under the curve of $A > 0.8$. For a smaller number of clusters, for all subjects SOM is outperformed by the

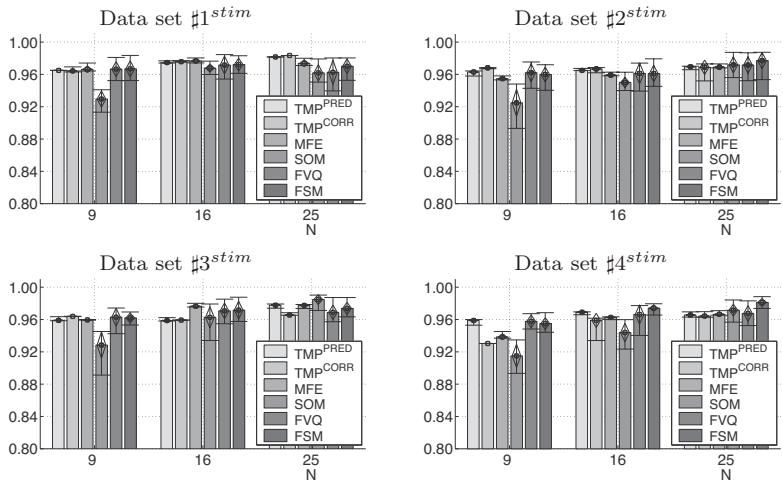


Figure 9.3
Results of the comparison between the different exploratory data analysis methods on motor stimulation fMRI data. Spatial accuracy of the maps is assessed by ROC analysis using the pixel-specific correlation map with a threshold of 0.6 as the reference segmentation. The figure illustrates the average area under the ROC curve and its deviations for 20 runs of each algorithm, using the same parameters but different initializations. The number of clusters for all techniques is equal to 9, 16, and 25, and results are plotted for all four subjects.

other methods, while for $N = 25$ this difference cannot be observed, an important result is that the TMP algorithm, for both distance measures (i.e. the nonlinear prediction error and cross-correlation), yields competitive results when compared to the established clustering methods.

9.4 Functional Connectivity Under Resting Conditions

This section describes results obtained with the unsupervised clustering methods for the analysis of the resting-state fMRI data. The partitioning results are compared with regard to the segmentation of the motor cortex.

Figure 9.4 visualizes the computed distance matrices for the resting-state data set of subject #1 for $N = 25$ clusters, based on both the correlation and the prediction error methods. The first row shows the unsorted distance matrices, and the second row shows the results obtained after application of the TMP algorithm, resulting in a display of the dis-

tance matrix where the rows and columns appear in an ordered fashion. The emerging block-diagonal structure reflects the characteristic of the TMP algorithm to cluster PTCs based on their mutual dependency (i.e. their pairwise distance).

For each resting-state fMRI data set, the position of the motor cortex is determined based on the segmentation provided by the pixel-specific stimulus-correlation map obtained in the motor task fMRI experiment of the same subject. That is, a PTC whose correlation coefficient in the motor stimulation experiment is above a defined threshold of Δ (e.g., $\Delta = 0.6$) is considered as belonging to the motor cortex. This segmentation approach is referred to as the cc-cluster method.

For the clustering methods, the segmentation of the motor cortex is obtained by merging single clusters. The identification of such clusters is determined by the similarity index (SI) [300]. The SI index is defined as

$$SI = 2\frac{|A_1 \cap A_2|}{|A_1| + |A_2|} \tag{9.1}$$

and gives a measure of the agreement of the two binary segmentations A_1 and A_2. It is defined as the ratio of twice the common area to the sum of the individual areas. An excellent agreement is given for $SI > 0.7$, according to [300]. Although the absolute value of SI is difficult to interpret, it gives a quantitative comparison between measurement pairs.

The cluster identification works as follows. First, the cluster showing the largest SI value with the reference segmentation is selected. Then this cluster is combined with the remaining cluster, if the SI value of the two merged clusters is increased. This procedure continues until no increase in the SI value is observed.

Figure 9.5 shows a comparison between the segmentation results obtained by the unsupervised clustering methods for subject #1 in the resting-state. By taking the average value of all PTCs belonging to a certain determined segmentation, a representative PTC for each segmentation is obtained. The figure shows that both the topographic mapping of proximity data and the classical clustering techniques are able to detect low-frequency connectivity associated with the motor cortex.

The resulting values for the SI index for the proposed methods

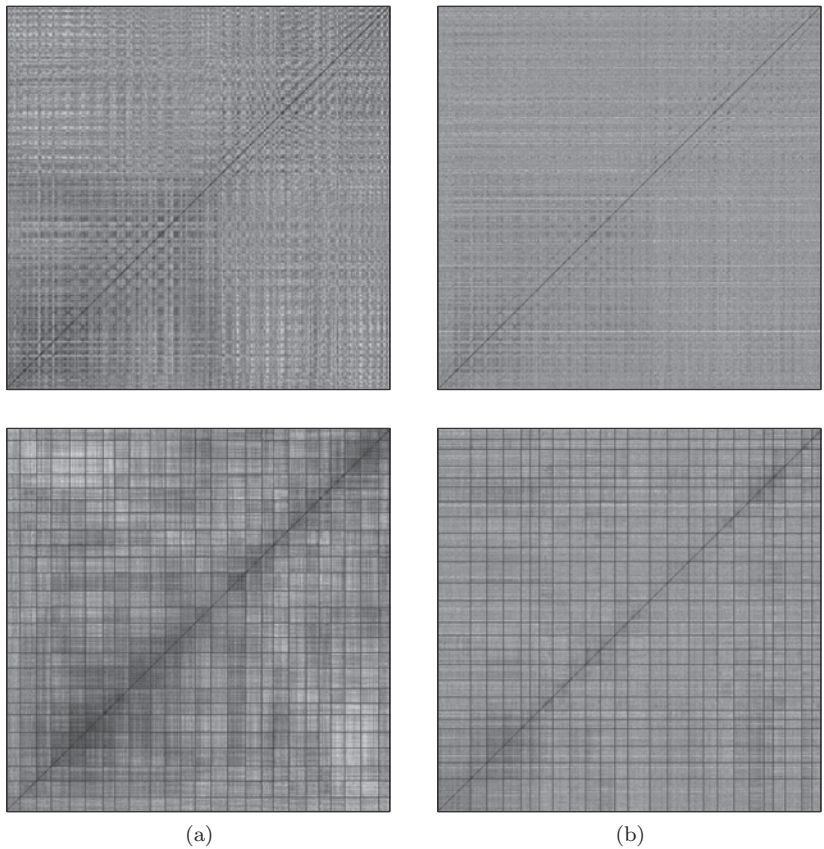


Figure 9.4
Distance matrices with distances represented by gray values, if $N = 25$ clusters is used for the analysis of subject #1 in the resting state experiment. Distances are determined based on the correlation method (a) and the prediction error method (b). The upper and lower rows show the matrices before and after applying the TMP algorithm, respectively. The dissimilarity matrices were plotted such that the rows from bottom to top and the columns from left to right correspond to increasing indices of the PTCs. The block-diagonal structure of the ordered distance matrices becomes evident. The dark lines represent the cluster borders and are overlaid on the distance matrices. Small distances are plotted dark, representing close proximity.

represent a quantitative evaluation of this observation and are shown in table 9.1. For all applied methods, they range within the interval [0.5, 0.6], showing a fair agreement. It should be noted that the novel TMP

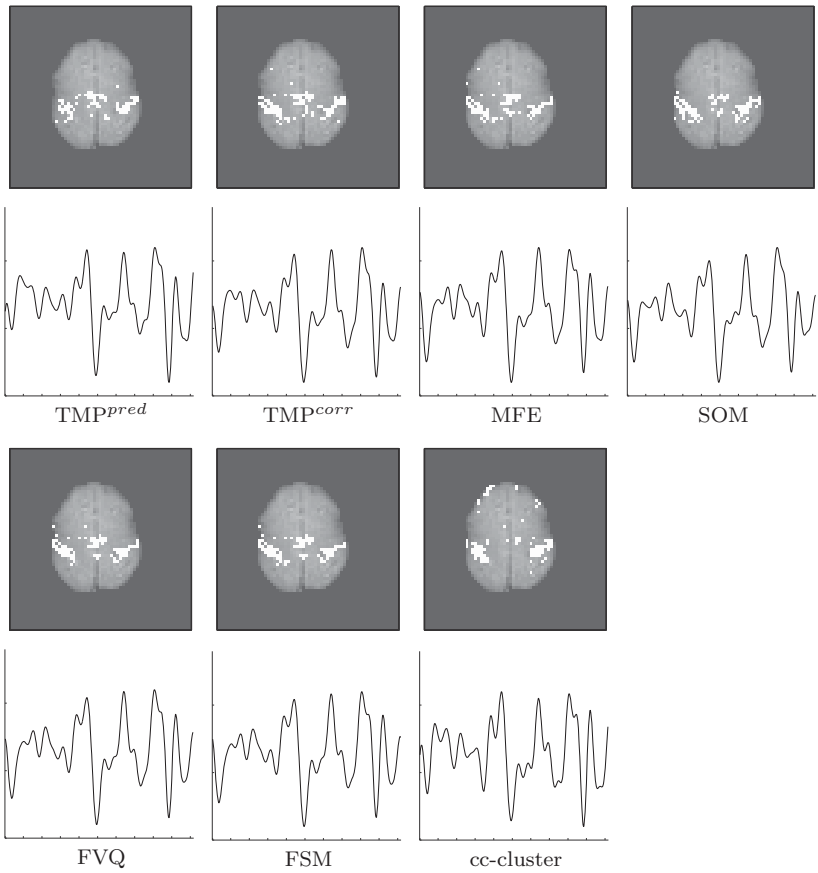


Figure 9.5
Segmentation results in the motor areas of subject #1 in the resting-state. The obtained functional connectivity maps are shown for all unsupervised methods. The cc-cluster describes a method based on the threshold segmentation of the pixel-specific correlation map of the *motor stimulation* fMRI experiment. This map assigns to each pixel the Pearson correlation coefficient between the PTC and the time-delayed stimulus function. The threshold was chosen as $\Delta = 0.6$, and thus every pixel correlation exceeding 0.6 is considered as activated and is white on the *cc*-cluster map. The procedure used in order to obtain the segmentation for clustering of the resting-state data is explained in the text. The average value of all PTCs belonging to segmented areas determines a segmentation representative PTC shown under the respective assignment map.

method in both variants yields acceptable results compared to the other

Table 9.1
SI-index as a quantitative measure of the agreement of the segmentation between the motor cortex areas in figure 9.5 and the reference segmentation *cc*-cluster.

TMP^{pred}	TMP^{corr}	MFE	SOM	FVQ	FSM
0.5409	0.5169	0.5476	0.5294	0.5663	0.5509

established clustering methods.

A comparison of the task activation maps with the functional connectivity maps reveals some very interesting observations regarding the resting-state data set: (a) the segmented motor areas in both hemispheres are less predominant for the resting-state data set; (b) the segmentation results for this data set does not show any pixels belonging to the frontal lobes; and (c) the segmentations of the resting-state data set include an increased number of pixels in the region of the supplementary motor cortex when compared to the cluster segmentation of the motor stimulation data set in figure 9.2. Looking at these differences, it becomes clear why an excellent agreement of $SI > 0.7$ for the cluster segmentations and the reference cannot be observed. Whether these differences are induced by physiological changes of the resting-state connectivity in comparison to the situation found in motor activity, remains speculative at this point.

9.5 Summary

This chapter has demonstrated the applicability of various unsupervised clustering methods using different distance metrics to the analysis of motor stimulation and resting-state functional MRI data. Two different strategies were compared: a Euclidian distance metric as the basis of the classical unsupervised clustering techniques and a topographic mapping of proximities determined by the correlation coefficient and the prediction error. Both strategies were successfully applied to segmentation tasks for both motor activation and resting-state fMRI data to capture spatiotemporal features of functional connectivity.

The most important results are summarized as follows: (1) both unsupervised clustering approaches show comparable results in connection with model-based evaluation methods in task-related fMRI experiments; and (2) they allow for the construction of connectivity maps of the motor

cortex that unveil dependencies between anatomically separated parts of the motor system at rest. It can be conjectured that the presented methods may be helpful for further investigation of functional connectivity in the resting human brain.

10 Classification of Dynamic Breast MR Image Data

Breast cancer is the most common cancer among women. Magnetic resonance (MR) is an emerging and promising new modality for detection and further evaluation of clinically, mammographically, and sonographically occult cancers [115, 293]. However, film and soft-copy reading and manual evaluation of breast MRI data are still critical, time–consuming and inefficient, leading to a decreased sensitivity [204]. Furthermore, the limited specificity of breast MR imaging continues to be problematic. Two different approaches are mentioned in literature [145] aiming to improve the specificity: (1) single–breast imaging protocols with high spatial resolution offer a meticulous analysis of the lesion's structure and internal architecture, and are able to distinguish between benign and malignant lesions; (2) lesion differential diagnosis in dynamic protocols is based on the assumption that benign and malignant lesions exhibit different enhancement kinetics. In [145], it was shown that the shape of the time-signal intensity curve is an important criterion in differentiating benign and malignant enhancing lesions in dynamic breast MR imaging. The results indicate that the enhancement kinetics, as shown by the time-signal intensity curves visualized in figure 10.1, differ significantly for benign and malignant enhancing lesions and thus represent a basis for differential diagnosis. In breast cancers, plateau or washout time courses (type II or III) prevail. Steadily progressive signal intensity time courses (type I) are exhibited by benign enhancing lesions. Also, these enhancement kinetics are shared not only by benign tumors but also by fibrocystic changes [145].

Concurrently, computer–aided diagnosis (CAD) systems in conventional X–ray mammography are being developed to expedite diagnostic and screening activities. The success of CAD in conventional X–ray mammography motivated the research of similar automated diagnosis techniques in breast MRI. Although, they are an issue of enormous clinical importance with obvious implications for health care politics, research initiatives in this field concentrate only on pattern recognition methods based on traditional artificial neural networks [161] ,[1, 162, 271].

A standard multilayer perceptron (MLP) was applied to the classification of signal–time curves from dynamic breast MRI in [161]. The

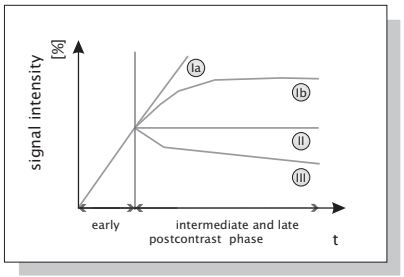


Figure 10.1
Schematic drawing of the time-signal intensity curve types [145]. Type I corresponds to a straight (Ia) or curved (Ib) line; enhancement continues over the entire dynamic study. Type II is a plateau curve with a sharp bend after the initial upstroke. Type III is a washout time course $\frac{SI_c - SI}{SI}$ where SI is the precontrast signal intensity and SI_c is the postcontrast signal intensity. In breast cancers, plateau or washout time courses (type II or III) prevail. Steadily progressive signal intensity time courses (type I) are exhibited by benign enhancing lesions.

major disadvantage of the MLP approach and also of any other supervised technique is the fixed number of input nodes, which imposes the constraint of a fixed imaging protocol. Delayed administration of the contrast agent or a different temporal resolution has a negative effect on the classification and segmentation capabilities. Thus, a change in the MR imaging protocol requires a new training of the CAD system. In addition, the system fails in most cases to diagnose small breast masses with a diameter of only a few millimeters. It must be mentioned that during the training phase of a classifier, a histopathologically classified lesion represents only a single input pattern. There is an urgent need, based on the limited number of existing training data, to efficiently extract information from a mostly inhomogeneous available data pool. While supervised classification techniques often fail to accomplish this task, the proposed biomimetic neural networks, in the long run, represent the best training approaches leading to advanced CAD systems.

When applied to segmentation of MR images, traditional pattern recognition techniques such as the MLP have shown unsatisfactory detection results and limited application capabilities [1, 162]. Furthermore, the underlying supervised nonbiological learning strategy leads to the inability to capture the feature structure of the breast lesion in the neural architecture. One recent paper demonstrated examples of the segmentation of dynamic breast MRI data sets by unsupervised neural networks.

Trough use of a Kohonen neural network, areas with similar signal time courses in mammographic image series were detected, making possible a clear detection of carcinoma [85].

In Summary, the major disadvantages associated with standard techniques in breast MRI are (1) requirement of a fixed MR imaging protocol, (2) lack of increase in sensitivity and/or specificity, (3) inability to capture the lesion structure, and (4) training limitations due to an inhomogeneous lesion data pool.

To overcome the above-mentioned problems, a minimal free energy vector quantization neural network is employed that focuses strictly on the observed complete MRI signal time series and enables a self–organized, data–driven segmentation of dynamic contrast–enhanced breast MRI time series with regard to fine-grained differences of signal amplitude, and dynamics, such as focal enhancement in patients with indeterminate breast lesions. This method is developed, tested, and evaluated for functional and structural segmentation, visualization, and classification of dynamic contrast-enhanced breast MRI data. Thus, it is a contribution toward the construction and evaluation of a flexible and reusable software system for CAD in breast MRI.

The results show that new method reveals regional properties of contrast–agent uptake characterized by subtle differences of signal amplitude and dynamics. As a result, one obtains both a set of prototypical time series and a corresponding set of cluster assignment maps which further provide a segmentation with regard to identification and regional subclassification of pathological breast tissue lesions. The inspection of these clustering results is a unique practical tool for radiologists, enabling a fast scan of the data set for regional differences or abnormalities of contrast-agent uptake. The proposed technique contributes to the diagnosis of indeterminate breast lesions by noninvasive imaging.

10.1 Materials and Methods

Patients

A total of 13 patients, all female and ranging in age from 48 to 61, with solid breast tumors, were examined. All patients had histopathologically confirmed diagnosis from needle aspiration/excision biopsy and surgical removal. Breast cancer was diagnosed in 8 of the 13 cases.

MR imaging

MRI was performed with a 1.5 T system (Magnetom Vision, Siemens, Erlangen, Germany) equipped with a dedicated surface coil to enable simultaneous imaging of both breasts. The patients were placed in a prone position. First, transversal images were acquired with a STIR (short TI inversion recovery) sequence (TR=5600 ms, TE=60 ms, FA=90°, IT=150 ms, matrix size 256×256 pixels, slice thickness 4 mm). Then a dynamic T1 weighted gradient echo sequence (3-D fast, low, angle-shot sequence) was performed (TR=12 ms, TE=5 ms, FA=25°) in transversal slice orientation with a matrix size of 256×256 pixels and an effective slice thickness of 4 mm.

The dynamic study consisted of six measurements with an interval of 83 sec. The first frame was acquired before injection of paramagnetic contrast agent (gadopentatate dimeglumine, 0.1 mmol/kg body weight; Magnevist^{TM}, Schering, Berlin, Germany) and immediately followed by the five other measurements. Rigid image registration by the AIR method [288] as a preprocessing step was used. As this did not correct for nonlinear deformations, only data sets without relevant motion artifacts were included. The initial localization of suspicious breast lesions was performed by computing difference images (i.e., subtracting the image data of the first acquisition from the fourth acquisition). As a preprocessing step to clustering, each raw gray-level time series $S(\tau), \tau \in \{1, \cdots, 6\}$ was transformed into a pixel time course (PTC) of relative signal reduction $x(\tau)$ for each voxel, the precontrast scan at $\tau = 1$ serving as reference. Based on this implicit normalization, no significant effect of magnetic field inhomogeneities on the segmentation results was observed.

Data clustering

The employed classifier (the minimal free energy vector quantization neural network) is according to grouping image pixels together based on the similarity of their intensity profiles in time (i.e., their time courses).

Let n denote the number of subsequent scans in a dynamic contrast-enhanced breast MRI study, and let K be the number of pixels in each scan. $\mu \in \{1, \cdots, K\}$, that is, the sequence of signal values $\{\mathbf{x}^{\mu}(1), \cdots, \mathbf{x}^{\mu}(n)\}$, can be interpreted as a vector $\mathbf{x}^{\mu}(i) \in \mathbf{R}^n$ in the n–dimensional feature of possible PTCs at each pixel.

Cluster analysis groups image pixels together based on the similarity of their intensity profiles in time. In the clustering process, a time course with n points is represented by one point in an n–dimensional Euclidean space which is subsequently partitioned into clusters based on the proximity of the input data. These groups or clusters are represented by prototypical time series called codebook vectors (CV), located at the centers of the corresponding clusters. The CVs represent prototypical PTCs sharing similar temporal characteristics.

Segmentation methods

In the following, three segmentation methods for the evaluation of signal intensity time courses for the differential diagnosis of enhancing lesions in breast MRI are presented. The results obtained by these methods are shown exemplarily on data set #1.

Segmentation method I

This segmentation method is based on carefully choosing a circular ROI defined by taking into account the voxels whose intensity curves are above a radiologist-defined threshold ($> 50\%$) in the early postcontrast phase. The specific choice of this threshold is motivated by the relevant literature (e.g., [82], where the probability of missing malignant lesions by excluding regions with a relative signal increase of less than 50% is considered negligible). For all voxels belonging to this ROI, an average time-signal intensity curve is computed. This averaged value is then rated. This very simple method corresponds to the radiologists' conventional way of analyzing dynamic MRI mammography data. Figure 10.2 illustrates the described segmentation method. White pixels have an above–threshold signal increase. The contrast–enhanced pixels are shown in figure 10.2b. Based on a region–growing method [95], the suspicious lesion area can be easily determined (see figure 10.8).

Figure 10.3 shows the result of the segmentation when it is applied to data set #1. Slices #14 to #17 contain the lesion. The average contrast–enhanced dynamics over all pixels is shown in the right image of this figure. It is a plateau curve after an initial medium upstroke.

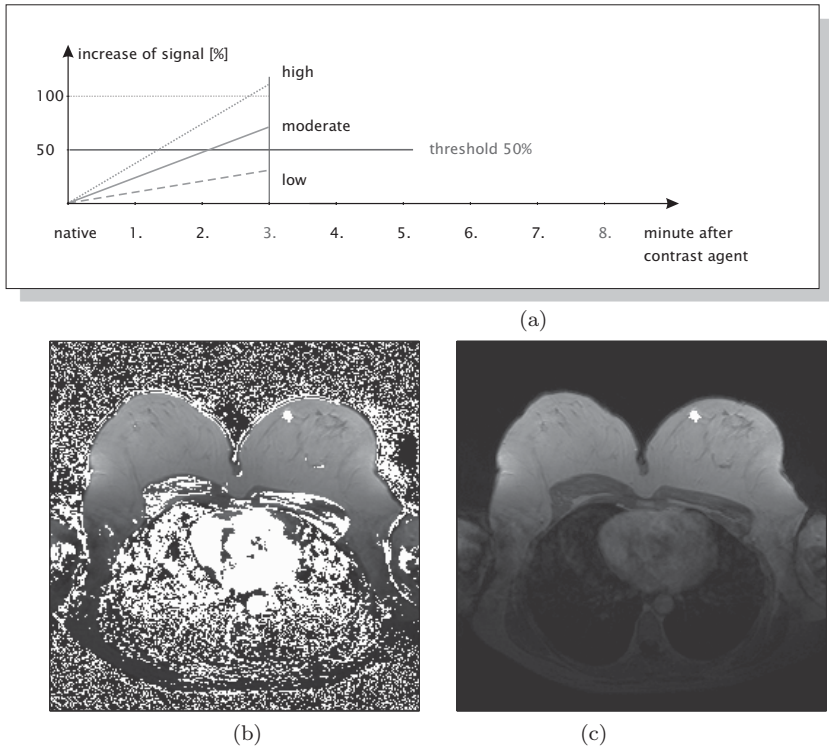


Figure 10.2
Segmentation method I. (a) Threshold segmentation. (b) Classification based on threshold segmentation: pixels exhibiting time signal intensity curves above a given threshold are white. (c) The lesion is determined based on region growing.

Segmentation method II

The ROI contains a slice through the whole breast, and all the voxels within the ROI are subject to cluster analysis. Results on data set #1 are presented in figures 10.4 and 10.5 for the clustering technique employing nine clusters. They are numbered consecutively from 1 to 9. The figures show cluster assignment maps and corresponding codebook vectors of breast MRI data covering a supramamillar transversal slice of the left breast containing a suspicious lesion that has been proven to be malignant by subsequent histological examination.

The procedure is able to segment the lesion from the surrounding breast tissue, as can be seen from cluster #6 of figure 10.4. The rapid

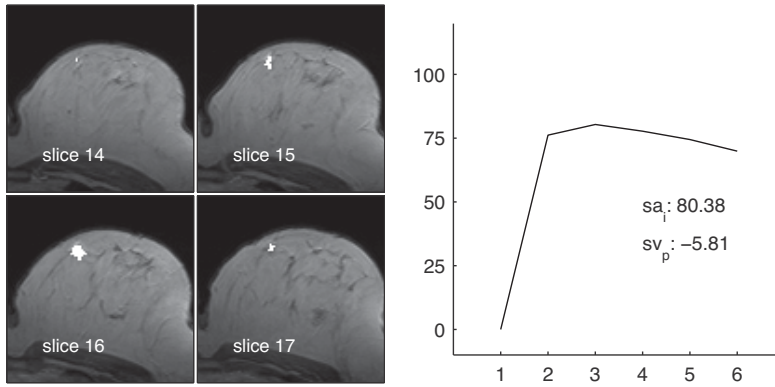


Figure 10.3
Segmentation method I applied to data set #1 (scirrhous carcinoma). The left image shows the lesion extent over slices #14 to #17. The right image shows the average time-signal intensity curve of all pixels belonging to this lesion.

and strong contrast-agent uptake is followed by subsequent plateau and washout phases in the round central region of the lesion, as indicated by the corresponding CV of cluster #6 in figure 10.5.

Furthermore, clustering results enable a subclassification within this lesion with regard to regions characterized by different MRI signal time courses: The central cluster #6 is surrounded by the peripheral circular clusters #7, 8, and 9, which primarily can be separated from both the central region and the surrounding tissue by the amplitude of their contrast-agent uptake ranging between CV #6 and all the other CVs.

Segmentation method III

This segmentation method combines method I with method II. Method I is chosen for determining the lesions with a super-threshold contrast-agent uptake, while method II performs a cluster analysis of the identified lesion.

Figure 10.6 shows the segmentation results for data set #1.

10.2 Results

The computation time for vector quantization depends on the number of PTCs included in the procedure. The computation time per data set

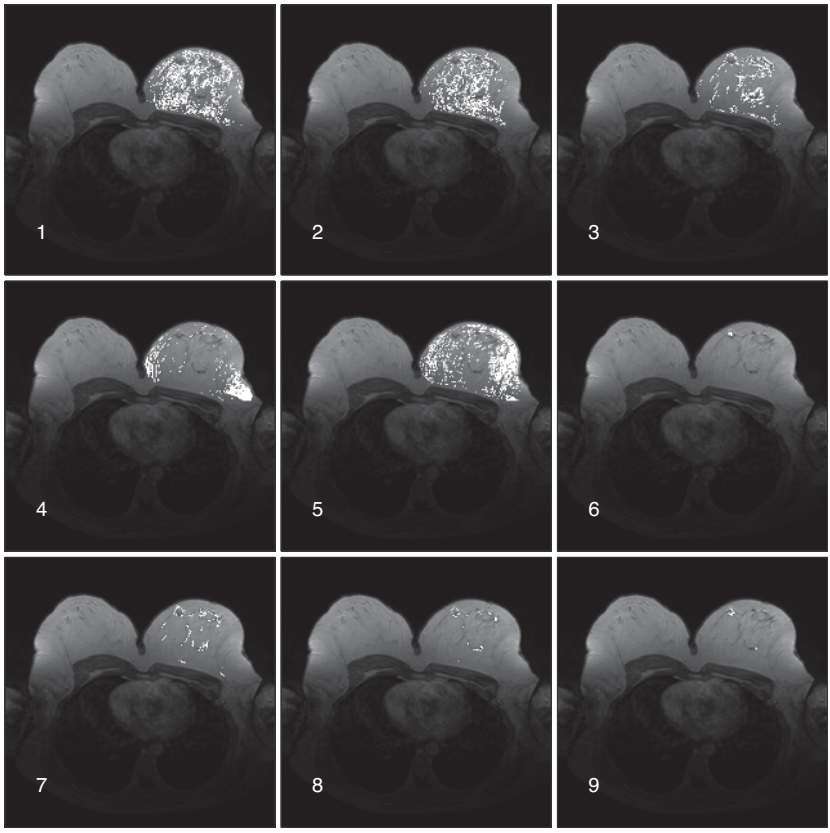


Figure 10.4
Segmentation method II: Cluster assignment maps for cluster analysis using the fuzzy clustering technique based on deterministic annealing of the dynamic breast MRI study (data set #1).

was 285 ± 110 s and 3.1 ± 2.5 sec for segmentation methods II and III, respectively, using an ordinary PC (Intel Pentium 4 CPU, 1.6 GHz, 512 MB RAM).

In the following, a comparison of three different lesion segmentation methods is presented when applied to a study involving 13 subjects. Segmentations I and III and a slightly changed version of segmentation method I which is called * are considered. Only the slice where the lesion has its largest circumference is chosen as an ROI, and then the process proceeds as described in method I. The results achieved by segmentation

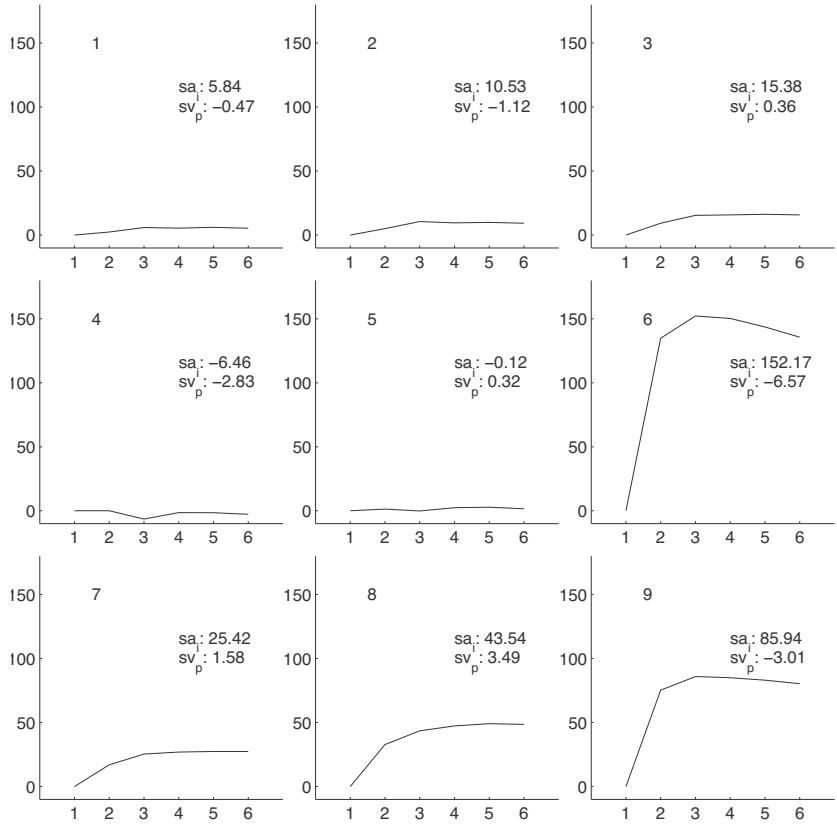


Figure 10.5
Segmentation method II: Codebook vectors for fuzzy clustering technique based on deterministic annealing of the dynamic breast MRI study according to figure 10.4. sa_i represents the initial, and sv_p the postinitial, time-signal intensity.

method II are not included, since it involves the whole breast and will be less accurate than method III.

The obtained time-signal intensity curves of enhancing lesions were plotted and presented to two experienced radiologists who were blinded to any clinical or mammographic information of the patients. The radiologists were asked to rate the time courses as having a steady, plateau, or washout shape type I, II, or III, respectively [145]. Their ratings are the column entries in table 10.1.

The classification of the lesions on the basis of the time-course

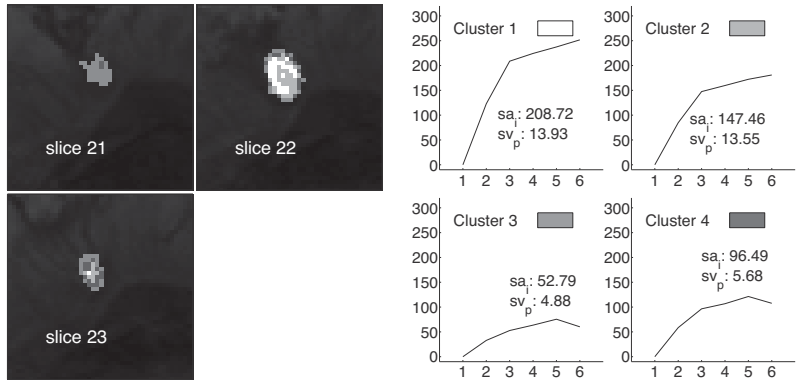


Figure 10.6
Segmentation method III applied to data set #3 (benign lesion, fibroadenoma), and resulting in four clusters. The left image shows the cluster distribution for slices 21 through 23. The right image visualizes the representative time-signal intensity time curves for each cluster. See plate 4 for the color version of this figure.

analysis was then compared for all three segmentation methods and with the lesions' definitive diagnoses. The definitive diagnosis was obtained histologically by means of excisional biopsy or of follow–up of the cases that, on the basis of history, clinical, mammographic, ultrasound, and breast MR imaging findings, were rated to be probably benign.

The results show an increase in sensitivity of breast MRI with regard to malignant tissue changes for 4 out of 13 cases. Also, the data sets #4 and 10 are incorrectly classified by method I and I as a benign lesion. Only method III, which includes cluster analysis as well as the conventional method of thresholding, correctly distinguishes between the two lesion types.

The mismatch between the three segmentation methods is shown in figures 10.11 to 10.18.

Figure 10.14 illustrates the result of this segmentation method when it is applied to a malignant lesion (ductal carcinoma in situ). Cluster 1 shows the central body of the lesion while and 2, 3, and 4 mark the periphery, surrounding the central part like a shell. The time-signal intensity curve for cluster 1 is of type III, while those for clusters 2, 3, and 4 are of type Ib.

Segmentation method I, which is based on the average time-signal intensity curve of the pixels, shows only a type Ib curve, which is

Table 10.1
Comparison of different data-driven segmentation methods of dynamic contrast-enhanced breast MRI time series. The differentiation between benign and malignant lesions is based on the method described in [145]. m is a malignant lesion and b a benign lesion.

Data set	Method I	Method I*	Method III	Lesion	Description
# 1	III	III	III	m	Scirrhous carcinoma
# 2	II	II	III	m	Tubulo–lobular carcinoma
# 3	Ib	Ib	Ib	b	Fibroadenoma
# 4	**Ib**	**Ib**	**III**	**m**	**Ductal carcinoma in situ**
# 5	Ia	Ia	Ia	b	Fibrous mastopathy
# 6	III	III	III	m	Papilloma
# 7	II	II	II	m	Ductal carcinoma in situ
# 8	Ib	Ib	Ib	b	Inflammatory granuloma
# 9	Ib	Ib	Ib	b	Scar, no relapse
# 10	**Ib**	**Ib**	**II**	**m**	**Ductal carcinoma in situ**
# 11	II	II	III	m	Invasive, ductal carcinoma
# 12	Ib	Ib	Ib	b	Fibroadenoma
# 13	III	III	III	m	Medullary carcinoma

characteristic of benign lesions. This fact is visualized in figure 10.11. The resulting mismatch between these two segmentation methods shows the main advantage of segmentation method III: based on a differentiated examination of tissue changes, we obtain an increase in sensitivity of breast MRI with respect to malignant lesions.

The examined data sets show that the relevance of the minimal free energy vector quantization neural network for MRI breast examination lies in the potential to increase the diagnostic accuracy for MRI mammography by improving the sensitivity without reduction of specificity. In order to document this improvement induced by segmentation method III, the results are included of all three segmentation methods on all the "critical" data sets (i.e., those where such a mismatch between segmentation methods I and III could be observed: data sets #2, 4, 10, and 11), see figures 10.7-10.22.

In this chapter, three different segmentation methods have been presented for the evaluation of signal-intensity time-courses for the differential diagnosis of enhancing lesions in breast MRI. Starting from the conventional methodology, the concepts of threshold segmentation and cluster analysis were introduced and in the last step those two concepts were combined.

The introduction of new techniques was motivated by the conceptual

weaknesses of the conventional technique. A manually predefined ROI substantially impacts the differential diagnosis in breast MRI. However, cluster analysis is almost independent of manual intervention, yet is computationally intensive. Threshold-based segmentation allows a differentiation between contrast–enhancing lesions and surrounding tissue. However, a subdifferentiation within the lesion is not provided. A fusion of the techniques of threshold segmentation and cluster analysis combines the advantages of these single methods. Thus, a fast segmentation method is obtained which carefully discriminates between regions with different lesion enhancement kinetics. Additionally, the third segmentation method, when compared to the method based only on cluster analysis, provides a subdifferentiation of the enhancement kinetics within a lesion, and is mostly independent of user intervention.

However, the most important advantage lies in the potential to increase the diagnostic accuracy of MRI mammography by improving the sensitivity without reduction of specificity for the data sets examined.

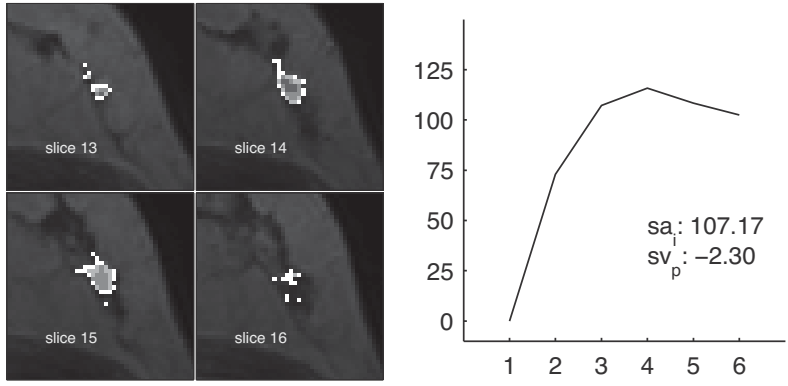


Figure 10.7
Segmentation method I applied to data set #2 (tubulo-lobular carcinoma). The left image shows the lesion extent over slices 13 to 16. The right image shows the average time-signal intensity curve of all pixels belonging to this lesion.

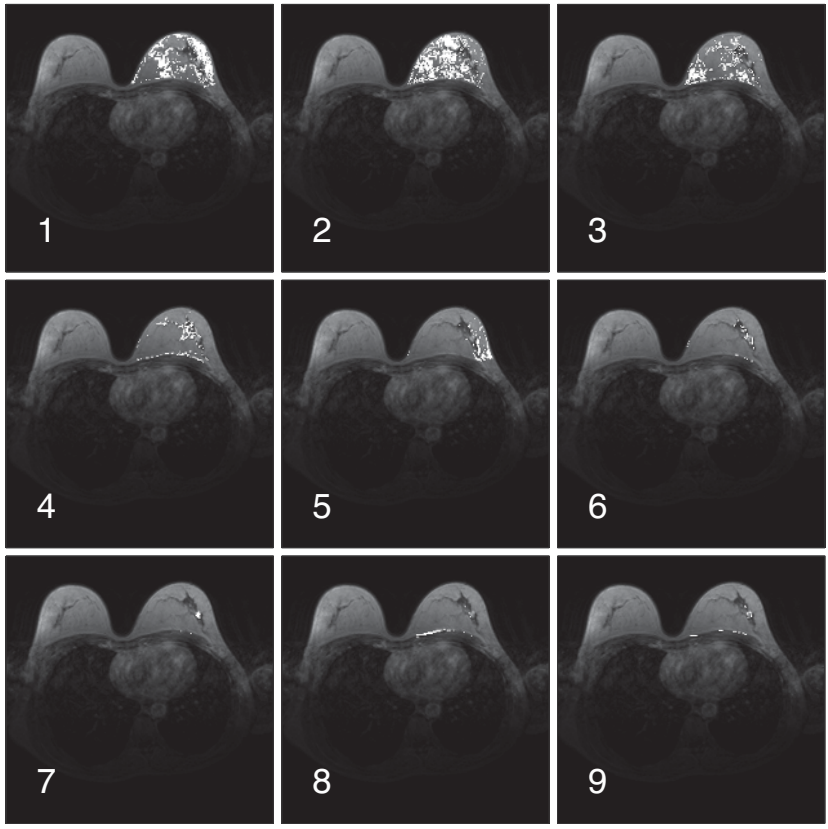


Figure 10.8
Segmentation method II: Cluster assignment maps for cluster analysis using on the fuzzy clustering technique based on deterministic annealing of the dynamic breast MRI study (data set #2).

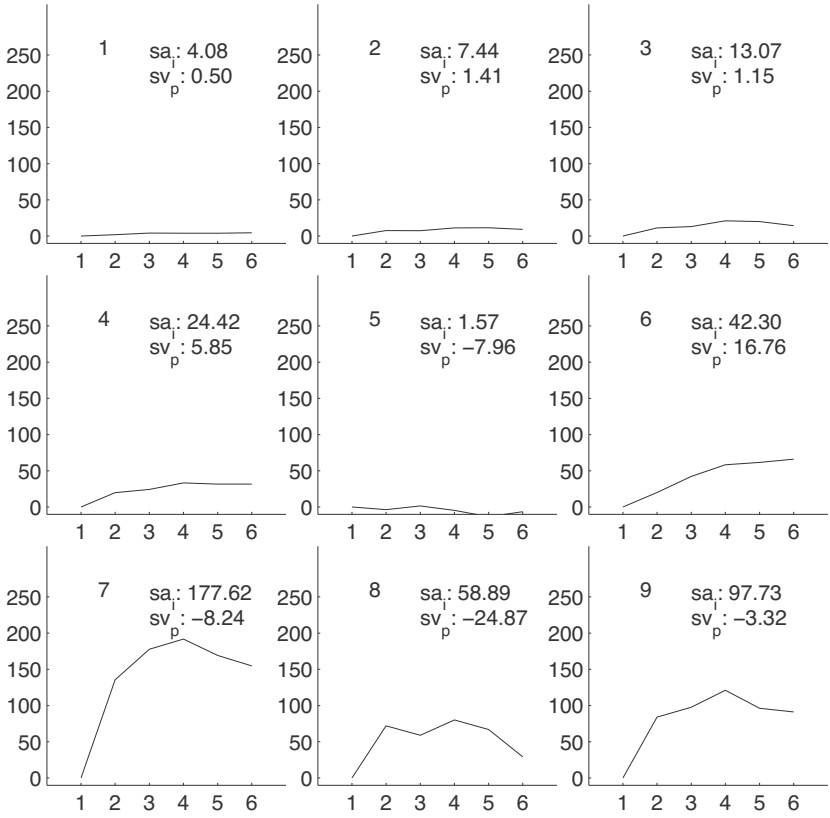


Figure 10.9
Segmentation method II: Codebook vectors for fuzzy clustering technique based on deterministic annealing of the dynamic breast MRI study according to figure 10.8. sa_i represents the initial, and sv_p the postinitial, time-signal intensity.

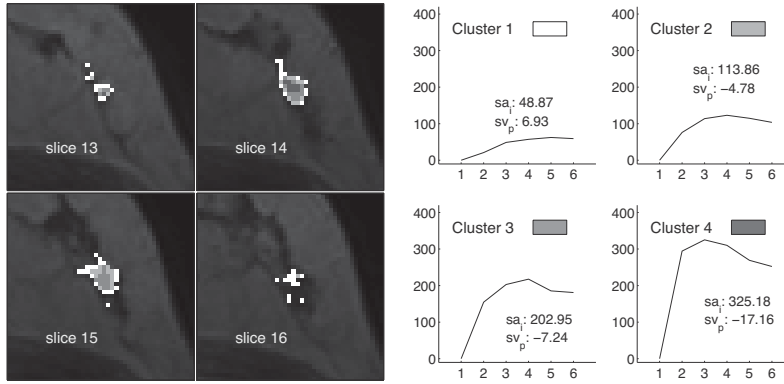


Figure 10.10
Segmentation method III applied to data set #1 (malignant lesion, tubulo-lobular carcinoma) with four clusters. The left image shows the cluster distribution for slices 13 through 16. The right image visualizes the representative time-signal intensity curves for each cluster. See plate 5 for the color version of this figure.

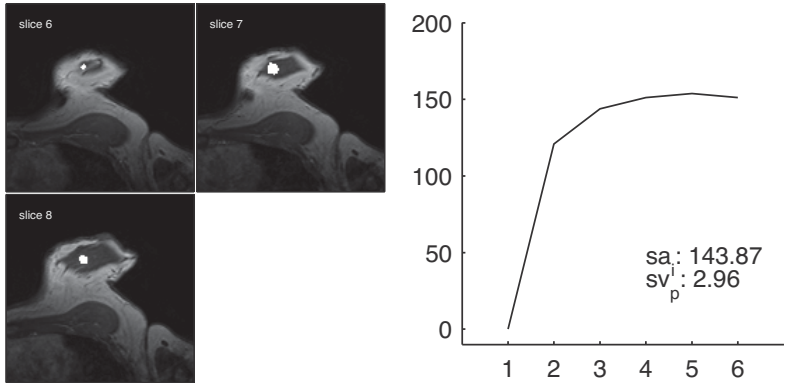


Figure 10.11
Segmentation method I applied to data set #4. The left image shows the lesion's extent over slices 6 to 8. The right image shows the average time-signal intensity curve of all pixels belonging to this lesion.

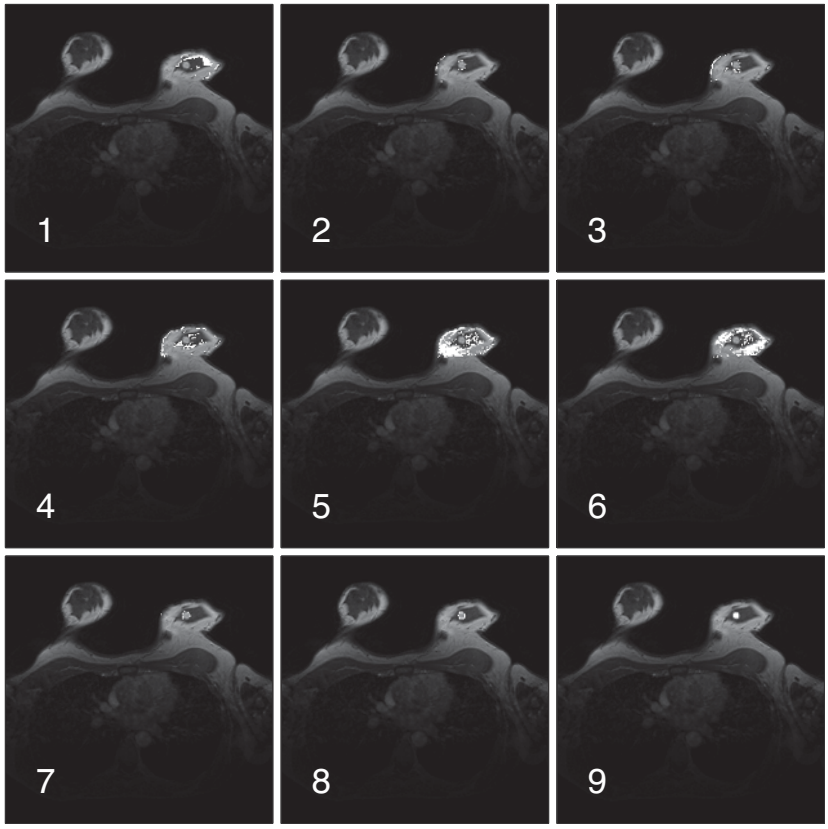


Figure 10.12
Segmentation method II: Cluster assignment maps for cluster analysis using the fuzzy clustering technique using deterministic annealing of the dynamic breast MRI study (data set #4).

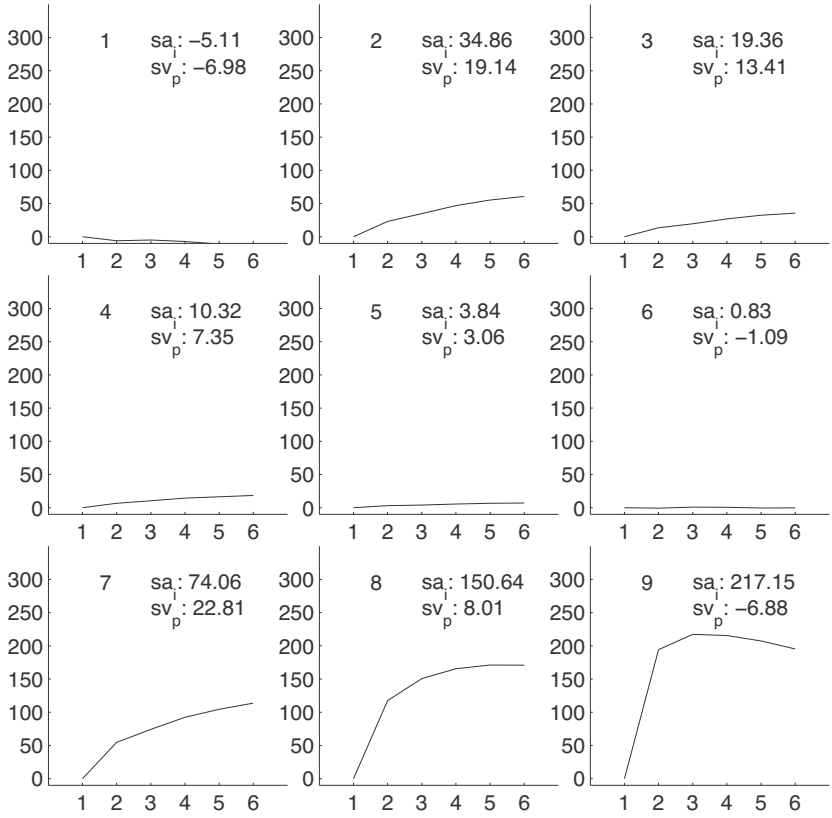


Figure 10.13
Segmentation method II: Codebook vectors for fuzzy clustering technique using deterministic annealing of the dynamic breast MRI study according to figure 10.12. sa_i represents the initial, and sv_p the postinitial, time-signal intensity.

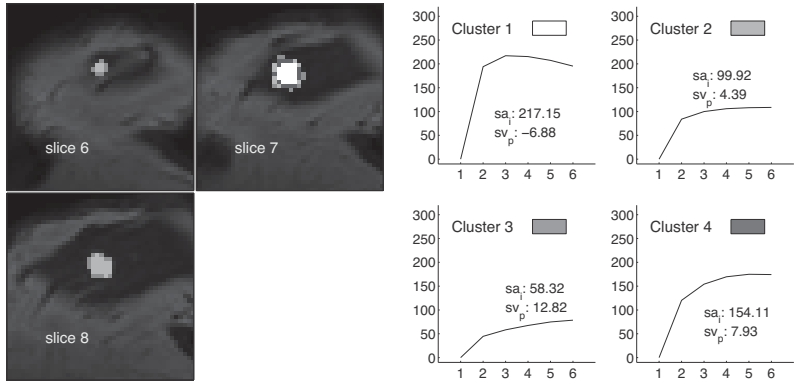


Figure 10.14
Segmentation method III applied to data set #4 (malignant lesion, ductal carcinoma in situ) and resulting in four clusters. The left image shows the cluster distribution for slices 6 through 8. The right image visualizes the representative time-signal intensity time curve for each cluster. See plate 6 for the color version of this figure.

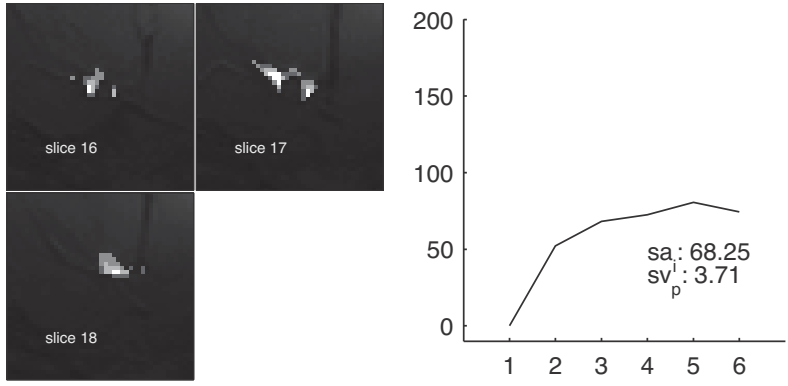


Figure 10.15
Segmentation method I applied to data set #10 (ductal carcinoma in situ). The left image shows the lesion's extent over slices 16 to 18. The right image shows the average time-signal intensity curve of all pixels belonging to this lesion.

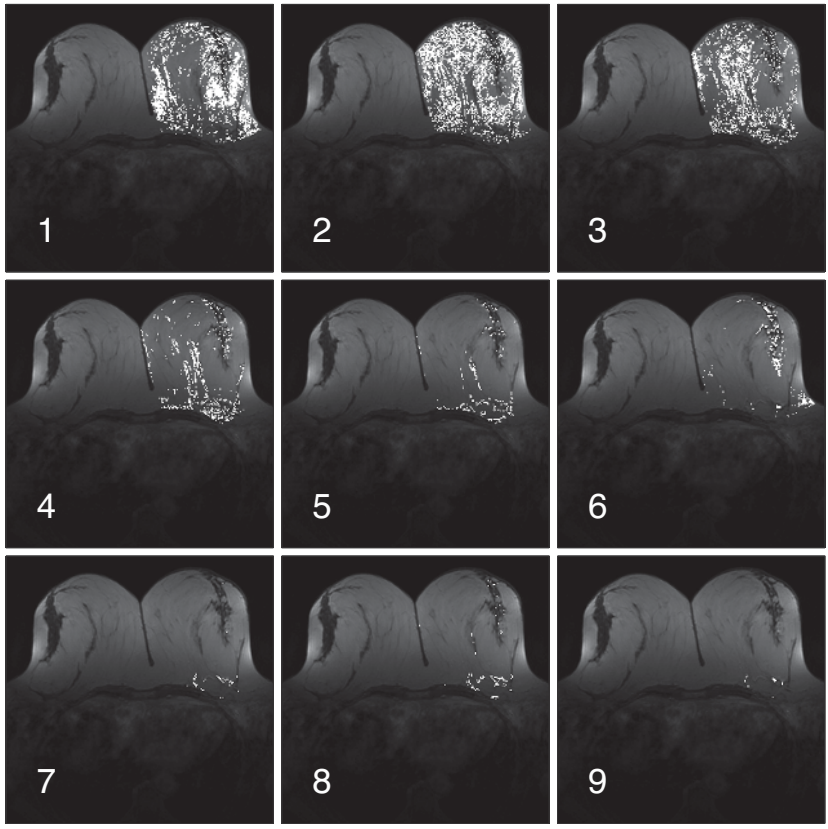


Figure 10.16
Segmentation method II: Cluster assignment maps for cluster analysis using the fuzzy clustering technique using deterministic annealing of the dynamic breast MRI study (data set #10).

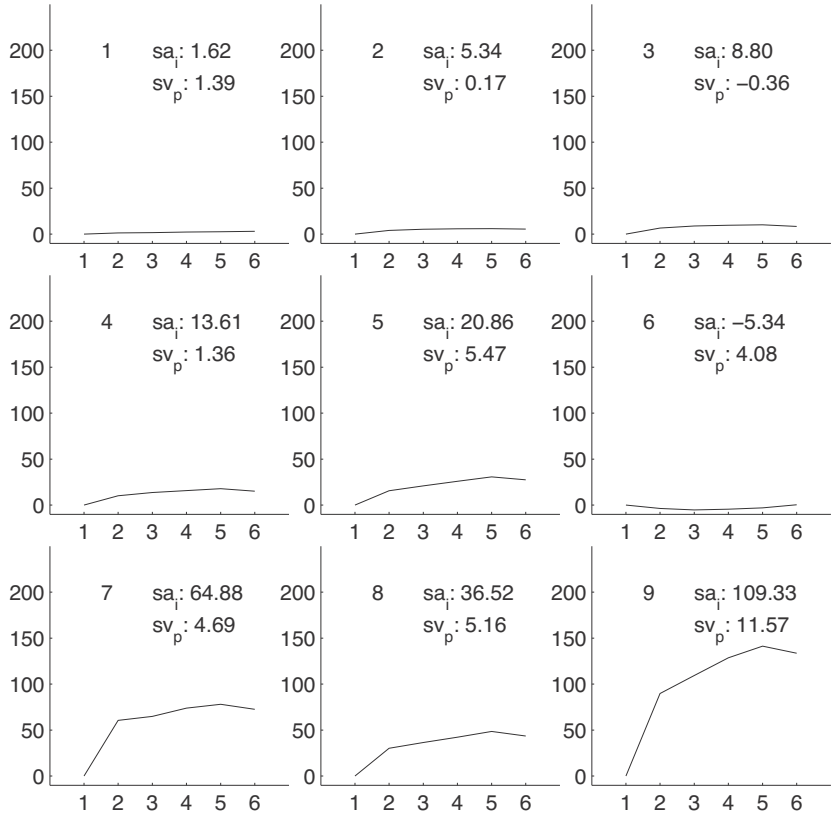


Figure 10.17
Segmentation method II: Codebook vectors for fuzzy clustering technique using deterministic annealing of the dynamic breast MRI study according to figure 10.16. sa_i represents the initial, and sv_p the postinitial, time-signal intensity.

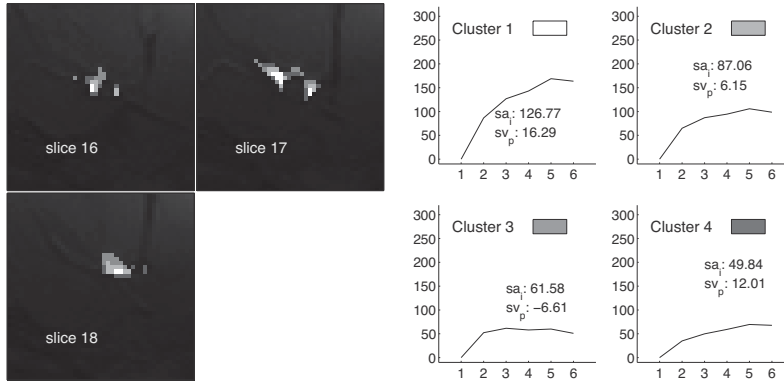


Figure 10.18
Segmentation method III applied to data set #10 (malignant lesion, ductal carcinoma in situ) with four clusters. The left image shows the cluster distribution for slices 16 through 18. The right image visualizes the representative time-signal intensity curve for each cluster. See plate 7 for the color version of this figure.

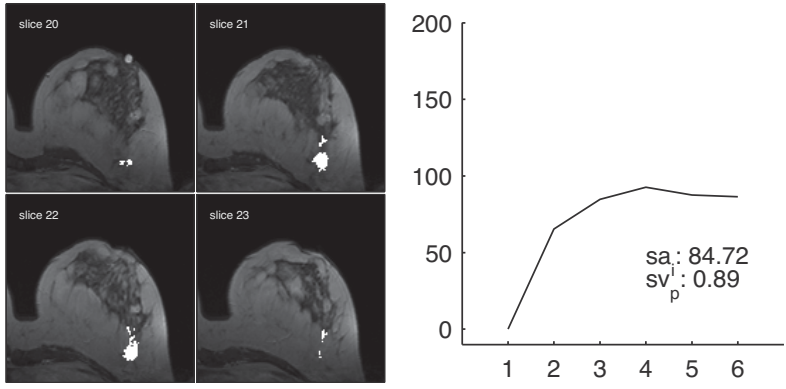


Figure 10.19
Segmentation method I applied to data set #11. The left image shows the lesion extent over slices 20 to 23. The right image shows the average time-signal intensity curve of all pixels belonging to this lesion.

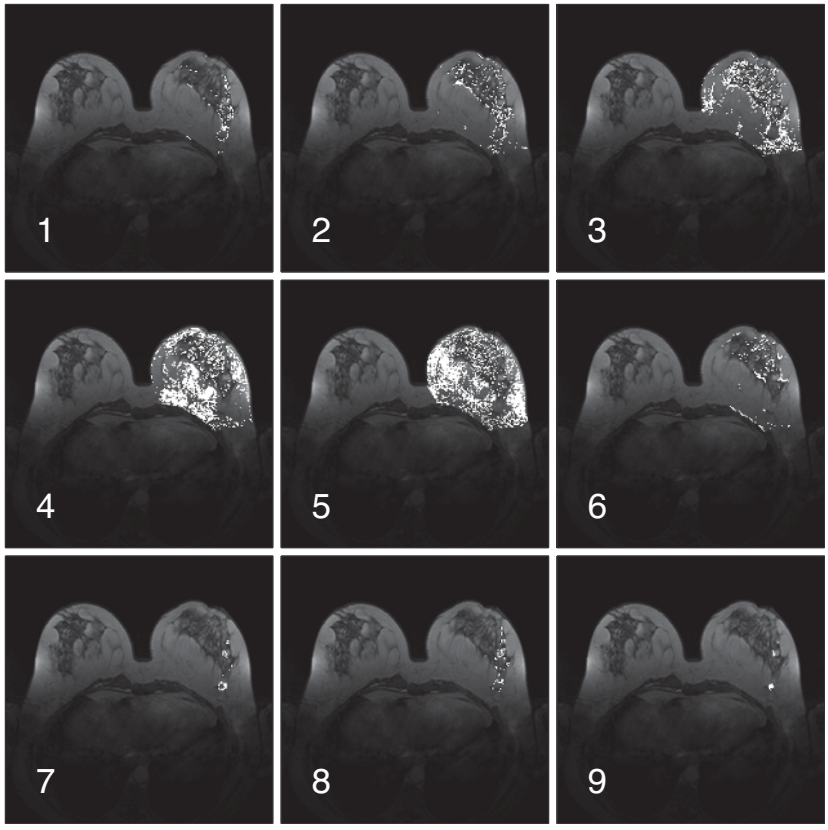


Figure 10.20
Segmentation method II: Cluster assignment maps for cluster analysis using the fuzzy clustering technique using deterministic annealing of the dynamic breast MRI study (data set #11).

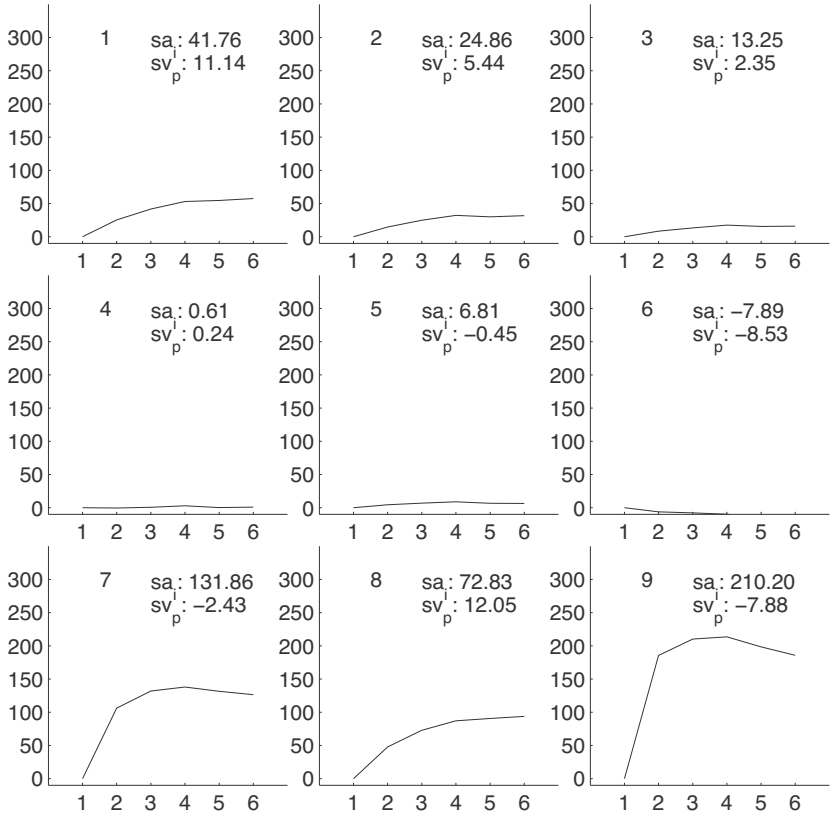


Figure 10.21
Segmentation method II: Codebook vectors for fuzzy clustering technique using deterministic annealing of the dynamic breast MRI study according to figure 10.20. sa_i represents the initial, and sv_p the postinitial, time-signal intensity.

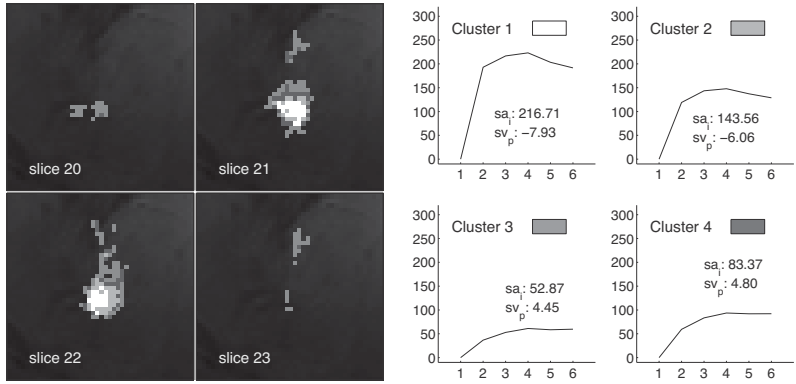


Figure 10.22
Segmentation method III applied to data set #11 (malignant lesion, invasive ductal carcinoma) with four clusters. The left image shows the cluster distribution for slices 20 through 23. The right image visualizes the representative time-signal intensity curve for each cluster. See plate 8 for the color version of this figure.

11 Dynamic Cerebral Contrast-enhanced Perfusion MRI

Cerebrovascular stroke is the third leading cause of mortality in industrial countries after cardiovascular disease and malignant tumors [86]. Therefore, the analysis of cerebral circulation has become an issue of enormous clinical importance.

Novel magnetic resonance imaging (MRI) techniques have emerged since the 1990s that allow for rapid assessment of normal brain function as well as cerebral pathophysiology. Both diffusion-weighted imaging and perfusion-weighted imaging have already been used extensively for the evaluation of patients with cerebrovascular disease [65]. They are promising research tools that provide data about infarct evolution as well as mechanisms of stroke recovery. Combining these two techniques with high-speed MR angiography leads to improvements in the clinical management of acute stroke subjects [192].

Measurement of tissue perfusion yields important information about organ viability and function. Dynamic susceptibility contrast MR imaging, also known as contrast-agent bolus tracking represents a noninvasive method for cerebrovascular perfusion analysis [275]. In contrast to other methods to determine cerebral circulation, such as iodinated contrast media in combination with dynamic X-ray computed tomography (CT) [11] and the administration of radioactive tracers for positron emission tomography (PET) blood-flow quantification studies [114], it allows high spatial and temporal resolution and avoids the disadvantage of patient exposure to ionizing radiation.

MR imaging allows assessment of regional cerebral blood-flow (rCBF), regional cerebral blood volume (rCBV), and mean transit time (MTT) (for definitions, see, e.g. [220]).

In clinical praxis, the computation of rCBV, rCBF, and MTT values from the MRI signal dynamics has been demonstrated to be relevant, even if its underlying theoretical basis may be weak under pathological conditions [65]. The conceptual difficulties with regard to the parameters MTT, rCBV, and rCBF arise from four basic constraints: (1) homogeneous mixture of the contrast-agent and blood pool, (2) negligible contrast-agent injection volume, (3) hemodynamic indifference of the contrast-agent, and (4) strict intravascular presence of the indicator substance. Conditions (1)-(3) are usually satisfied in dynamic susceptibility

contrast MRI using intravenous bolus administration of gadolinium compounds. Condition (4), however, requires an intact blood-brain barrier. This prerequisite is fulfilled in examinations of healthy subjects. These limitations for the application of the indicator dilution theory have been extensively discussed in the literature on MRI [200, 220] and nuclear medicine [149]. If, absolute flow quantification by perfusion MRI should be performed, the additional measurement of the arterial input function is needed, which is difficult to obtain in clinical routine diagnosis.

However, clinicians agree that determining parameter images based on the MRI signal dynamics, is a key issue in clinical decision-making, bearing a huge potential for diagnosis and therapy.

The analysis of perfusion MRI data by unsupervised clustering methods provides the advantage that it does not imply speculative presumptive knowledge on contrast-agent dilution models, but strictly focuses on the observed complete MRI signal time series. In this chapter, the applicability of clustering techniques as tools for the analysis of dynamic susceptibility contrast MRI time series is demonstrated and the performance of five different clustering methods is compared for this purpose.

11.1 Materials and Methods

Imaging protocol

The study group consisted of four subjects: (1) two men aged 26 and 37 years without any neurological deficit, history of intracranial abnormality, or previous radiation therapy. They were referred to clinical radiology to rule out intracranial abnormality. (2) two subjects (one man and one woman, aged 61 and 76 years, respectively) with subacute stroke (symptoms two and four days, respectively) who underwent MRI examination as a routine clinical diagnostic procedure. All four subjects gave their written consent. Dynamic susceptibility contrast MRI was performed on a 1.5 T system (Magnetom Vision, Siemens, Erlangen, Germany) using a standard circularly polarized head coil for radio frequency transmission and detection. First, fluid-attenuated inversion recovery, T2-weighted spin echo, and diffusion-weighted MRI sequences were obtained in transversal slice orientation, enabling initial localization and evaluation of the cerebrovascular insult in the subjects with stroke. Then dynamic susceptibility contrast MRI was performed us-

ing a 2-D gradient echo echoplanar imaging (EPI) sequence employing 10 transversal slices with a matrix size of 128×128 pixels, pixel size 1.88×1.88 mm, and a slice thickness of 3.0 mm (TR = 1.5 sec, TE = 0.54 sec, FA = 90°). The dynamic study consisted of 38 scans with an interval of 1.5 sec, between each scan. The perfusion sequence and an antecubital vein bolus injection (injection flow 3 ml/sec) of gadopentetate dimeglumine (0.15 mmol/kg body weight, Magnevist^{TM}, Schering, Berlin, Germany) were started simultaneously in order to obtain several (more than six) scans before cerebral first pass of the contrast-agent. The registration of the images was performed based on the automatic image alignment (AIR) algorithm [288].

Data analysis

In an initial step, a radiologist excluded by manual contour tracing the extracerebral parts of the given data sets. Manual presegmentation was used for simplicity, as this study was designed to examine only a few MRI data sets in order to demonstrate the applicability of the perfusion analysis method.

For each voxel, the raw gray-level time series $S(\tau)$, $\tau \in \{1, \ldots, 38\}$ was transformed into a pixel time course (PTC) of relative signal reduction $x(\tau)$ by

$$x(\tau) = \left(\frac{S(\tau)}{S_0} \right)^{\alpha}, \tag{11.1}$$

where S_0 is the precontrast gray level and $\alpha > 0$ a is distortion exponent. The effect of the native signal intensity was eliminated prior to contrast-agent application. If time-concentration curves are not computed according to the above equation (i.e., avoiding division of the raw time series data by the pre-contrast gray level before clustering), implicit use is made of additional tissue-specific MR imaging properties that do not directly relate to perfusion characteristics alone.

In the study, S_0 was computed as the average gray level at scan times $\tau \in \{3, 4, 5\}$, excluding the first two scans. There exists an exponential relationship between the relative signal reduction $x(\tau)$ and the local contrast-agent tissue concentration $c(\tau)$ [223], [181], [83], [137]:

$$c(\tau) = -\ln x(\tau) = -\alpha \ln \left(\frac{S(\tau)}{S_0} \right), \tag{11.2}$$

where $\alpha > 0$ is an unknown proportionality constant. Based on equation (11.2), the concentration-time curves (CTCs) are obtained from the signal PTCs.

Conventional data analysis was performed by computing MTT, rCBV, and rCBF parameter maps employing the relations (e.g. [299], [11], [240])

$$\mathrm{MTT} = \frac{\int \tau \cdot c(\tau)\, d\tau}{\int c(\tau)\, d\tau}, \quad \mathrm{rCBV} = \int c(\tau)\, d\tau, \quad \mathrm{rCBF} = \frac{\mathrm{rCBV}}{\mathrm{MTT}}. \quad (11.3)$$

Methods for analyzing perfusion MRI data require presumptive knowledge of contrast-agent dynamics based on theoretical ideas of contrast-agent distribution that cannot be confirmed by experiment (e.g., determination of relative CBF, relative CBV, or MTT computation from MRI signal dynamics). Although these quantities have been shown to be very useful for practical clinical purposes, their theoretical foundation is weak, as the essential input parameters of the model cannot be observed directly. On the other hand, methods for absolute quantification of perfusion MRI parameters do not suffer from these limitations [200]. However, they are conceptually sophisticated with regard to theoretical assumptions and require additional measurement of arterial input characteristics, which sometimes may be difficult to perform in clinical routine diagnosis. At the same time, these methods require computationally expensive data postprocessing by deconvolution and filtering. For example, deconvolution in the frequency domain is very sensitive to noise. Therefore, additional filtering has to be performed, and heuristic constraints with regard to smoothness of the contrast-agent residual function have to be introduced. Although other methods, such as singular value decomposition (SVD), could be applied, a gamma variate fit [213, 265] was used in this context.

The limitations with regard to perfusion parameter computation-based equations (11.3) are addressed in the literature (e.g., [281], [220]).

Evaluation of the clustering methods

This section is dedicated to presenting the algorithms and evaluating the discriminatory power of unsupervised clustering techniques. These are Kohonen's self-organizing map (SOM), fuzzy clustering based on deterministic annealing, the "neural gas" network, and the fuzzy c-means algorithm. These techniques are according to grouping image

pixels together based on the similarity of their intensity profile in time (i.e., their time courses).

Let n denote the number of scans in a perfusion MRI study, and let K be the number of pixels in each scan. The dynamics of each pixel $\mu \in \{1, \ldots, K\}$ (i.e., the sequence of signal values $\{\mathbf{x}^{\mu}(1), \ldots, \mathbf{x}^{\mu}(n)\}$) can be interpreted as a vector $\mathbf{x}^{\mu}(i) \in \mathbf{R}^n$ in the n-dimensional feature space of possible signal time series at each pixel (PTC). For perfusion MRI, the feature vector represents the PTC.

The chosen parameters for each technique are the following. For SOM [142] is chosen: (1) a one-dimensional lattice and (2) the maximal number of iterations. For the fuzzy clustering based on deterministic annealing, a batch expectation maximization (EM) version [173] of fuzzy clustering based on deterministic annealing is used in which the computation of CVs $\mathbf{w}_j$ (M-step) and assignment probabilities a_j (E-step) is decoupled and iterated until convergence at each annealing step characterized by a given "temperature" $T = 2\rho^2$. Clustering was performed employing 200 annealing steps corresponding to approximately 8×10^3 EM iterations within an exponential annealing schedule for ρ. The constant α in equation (11.1) was set at to $\alpha = 3$. For "neural gas" network we chose: (1) the learning parameters $\varepsilon_i = 0.5$ and $\varepsilon_f = 0.005$, and (2) the lattice parameters λ_i equal to half the number of classes and $\lambda_f = 0.01$, and (3) the maximal number of iterations equal to 1000. For the fuzzy algorithms [33], the fuzzy factor=1.05, and the maximal number of iterations equal to 120 is chosen.

The performance of the clustering techniques was evaluated by

(1) qualitative visual inspection of cluster assignment maps (i. e. cluster membership maps) according to a minimal distance criterion in the metric of the PTC feature space shown exemplarily only for the "neural gas" network; (2) qualitative visual inspection of corresponding cluster-specific CTCs for the "neural gas" network; (3) quantitative analysis of cluster-specific CTCs by computing cluster-specific relative perfusion parameters (rCBV, rCBF, MTT); (4) comparison of the best-matching cluster representing the infarct region from the cluster assignment maps for all presented clustering techniques with conventional pixel-specific relative perfusion parameter maps; (5) quantitative assessment of asymmetry between the affected and a corresponding non-affected contralateral brain region based on clustering results for a subject with stroke in the right basal ganglia; (6) cluster validity indices, and (7) receiver

operating characteristic (ROC) analysis;

The implementation of a quantitative ROC analysis demonstrating the performance of the presented clustering paradigms is reported in the following. Besides the four clustering techniques - "neural gas" network, Kohonen's self-organizing map (SOM), fuzzy clustering based on deterministic annealing, and fuzzy c-means vector quantization - for the last, two different implementations are employed: fuzzy c-means with unsupervised codebook initialization (FSM) and the fuzzy c-means algorithm (FVQ) with random codebook initialization. The two relevant parameters in an ROC study, sensitivity and specificity, are explained in the following for evaluating the dynamic perfusion MRI data. In the study, sensitivity is the proportion of the activation site identified correctly, and specificity is the proportion of the inactive region identified correctly. Both sensitivity and specificity are functions of the two threshold values Δ_1 and Δ_2, representing the thresholds for the reference and compared partitions, respectively. Δ_2 is varied over its whole range while Δ_1 is kept constant. By plotting the trajectory of these two parameters (sensitivity and specificity), the ROC curve is obtained. In the ideal case, sensitivity and specificity are both 1, and thus any curve corresponding to a certain method closest to the uppermost left corner of the ROC plot will be the method of choice. The results of quantitative ROC analysis presented in figure 11.14 show large values of the areas under the ROC curves as a quantitative criterion of diagnostic validity (i.e. agreement between clustering results and parametric maps).

The threshold value Δ_1 in table 11.1 was carefully determined for both performance metrics, regional cerebral blood volume (rCBV; left column), and mean transit time (MTT): Δ_1 was chosen as the one that maximizes the AUC of the ROC curves of experimental series. The optimal threshold value Δ_1 is given individually for each data set (see table 11.1) and corresponds to the maximum of the sum over all ROC areas for each possible threshold value.

The ground truth used for the ROC analysis is given by the segmentation obtained for the parameter values of the time series of each individual pixel (i.e. the conventional analysis). The implemented procedure is as follows: (a) Select a threshold Δ_1. (b) Then, determine the ground truth: for the time series of each individual pixel, compare the MTT value to Δ_1. If the MTT value of this specific pixel is less than Δ_1, assign this pixel to the active ground truth region; otherwise, assign it

Table 11.1
Optimal threshold value Δ_1 for the data sets #1 to #4 based on rCBV and MTT.

	rCBV	MTT
#1	0.30	21.0
#2	0.30	28.0
#3	0.30	18.7
#4	0.20	21.5

to the inactive one. (c) Select a threshold Δ_2 independently of Δ_1. Determine all the clusters whose cluster-specific concentration time-curve reveals an MTT less than Δ_2. Assign all the pixels belonging to these clusters to the active region found by the method. Plot the (sensitivity, specificity) point for the chosen value of Δ_2 by comparing with the ground truth. (d) Repeat (c) for different values of Δ_2.

Thus, for each Δ_2, a single (sensitivity, specificity) point is obtained. For each Δ_1, however, a complete ROC curve is obtained by variation of Δ_2, where Δ_1 remains fixed. This means that for different values of Δ_1, different ROC curves in general are obtained. Δ_1 is chosen for each data set in such a way that the area under the ROC curve (generated by variation of Δ_2) is maximal. The corresponding values for Δ_1 are given in table 7.2.

11.2 Results

In this section, the clustering results of the pixel time courses based on the presented methods are presented.

To elucidate the clustering process in general, and thus to obtain a better understanding of the techniques, the cluster assignment maps and the corresponding cluster-specific concentration-time curves belonging to the clusters exemplarily only for the "neural gas" network are shown.

Clustering results for a 38-scan dynamic susceptibility MRI study in a subject with a subacute stroke affecting the right basal ganglia are presented in figures 11.1 and 11.2. After discarding the first two scans, a relative signal reduction time series $x(\tau), \tau \in \{1, \ldots, n\}, n = 36$ can be computed for each voxel according to equation (11.1). Similar PTCs form a cluster. Figure 11.1 shows the "cluster assignment maps" overlaid onto an EPI scan of the perfusion sequence. In these maps, all the pixels that belong to a specific cluster are highlighted. The decision on assigning

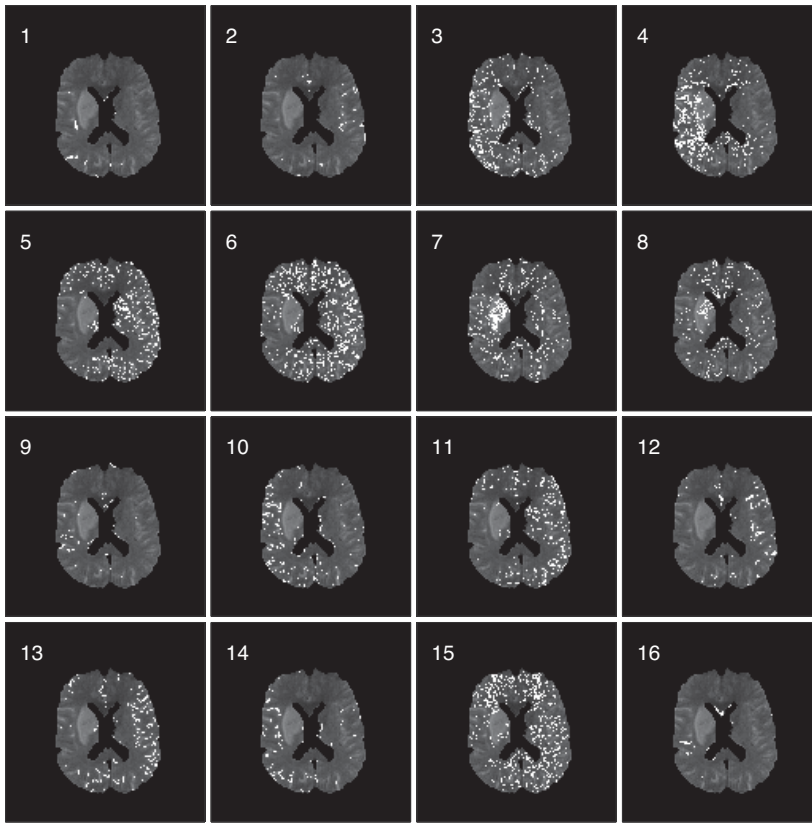


Figure 11.1
Cluster assignment maps for the "neural gas" network of a dynamic perfusion MRI study in a subject with a stroke in the right basal ganglia. Self-controlled hierarchical neural network clustering of PTCs $x(\tau)$ was performed by the "neural gas" network employing 16 CVs (i.e., a maximal number of 16 separate clusters at the end of the hierarchical VQ procedure). For a better orientation, an anatomic EPI scan of the analyzed slice is underlaid.

a pixel ν characterized by the PTC $\mathbf{x}_\nu = (x_\nu(\tau)), \tau \in \{1, \ldots, n\}$ to a specific cluster j is based on a minimal distance criterion in the n-dimensional time series feature space (i.e., ν is assigned to cluster j), if the distance $\|\mathbf{x}_\nu - \mathbf{w}_j\|$ is minimal, where $\mathbf{w}_j$ denotes the CV belonging to cluster j. Each CV represents the weighted mean value of all the PTCs belonging to this cluster.

Self-controlled hierarchical neural network clustering of PTCs $x(\tau)$

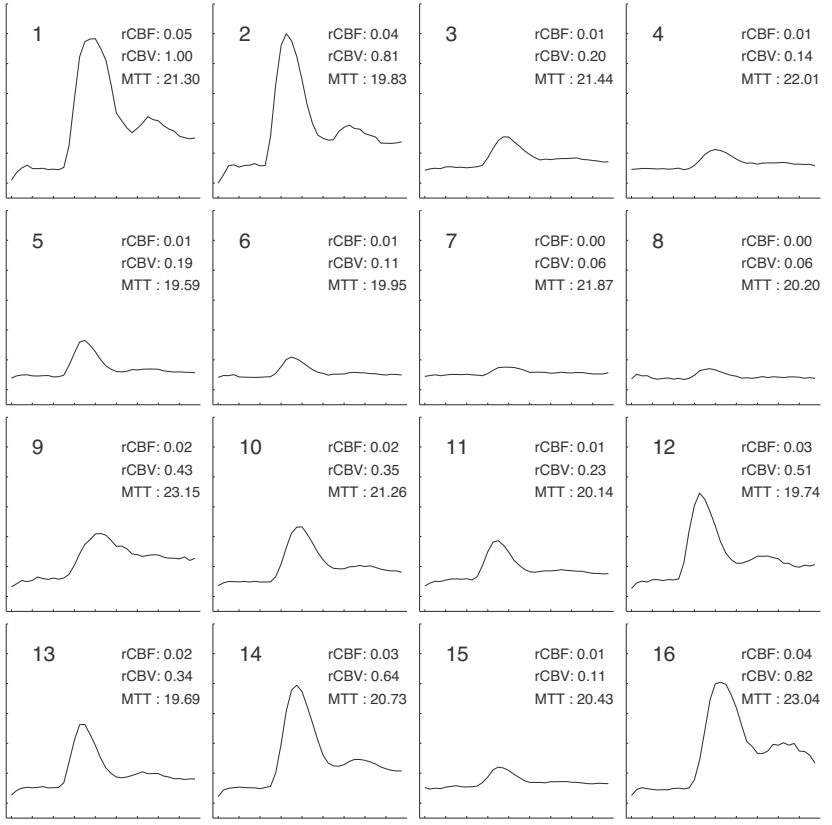


Figure 11.2
Cluster-specific concentration-time curves for the "neural gas" network of a dynamic perfusion MRI study in a subject with a stroke in the right basal ganglia. Cluster numbers correspond to figure 11.1. MTT values are indicated as multiples of the scan interval (1.5 sec), rCBV values are normalized with regard to the maximal value (cluster #1). rCBF values are computed from MTT and rCBV by equation (11.3). The X-axis represents the scan number, and the Y-axis is arbitrary.

was performed by a "neural gas" network employing 16 CVs (i.e. a maximal number of 16 separate clusters at the end of the hierarchical VQ procedure, as shown in figure 11.1).

Figure 11.2 shows the prototypical cluster-specific CTCs belonging to the pixel clusters of figure 11.1. These can be computed from equation (11.2), where the pixel-specific PTC $x(\tau)$ is replaced by the cluster-specific CV.

The area of the cerebrovascular insult in the right basal ganglia for subject 1 is clearly represented mainly by cluster #7 and also by cluster #8, which contains other essential areas. The small CTC amplitude is evident (i.e., the small cluster-specific rCBV, the rCBF, and the large MTT). Cluster #3 and #4 contain peripheral and adjacent regions. Clusters #1, #2, #12, #14, and #16 can be attributed to larger vessels located in the sulci. Figure 11.2 shows the large amplitudes and apparent recirculation peaks in the corresponding cluster-specific CTCs .

Further, clusters #2, #12, and #11 represent large, intermediate, and small parenchymal vessels respectively of the nonaffected left side showing subsequently increasing rCBV and smaller recirculation peaks. The clustering technique unveils even subtle differences of contrast agent first-pass times: small time-to-peak differences of clusters #1, #2, #12, #14, and #16 enable discrimination between left- and right-side perfusion. Pixels corresponding to regions supplied by a different arterial input tend to be collected into separate clusters: For example, clusters #6 and #11 contain many pixels that can be attributed to the supply region of the left middle cerebral artery, whereas clusters #3 and #4 include regions supplied by the right middle cerebral artery. Contralateral clusters #6 and #11 versus #3 and #4 show different cluster-specific MTTs as evidence for an apparent perfusion deficit at the expense of the right-hand side.

The diffusion-weighted image in figure 11.3a visualizes the structural lesion. Figs. 11.3b, c, and d represent the conventional pixel-based MTT, rCBF, and rCBV maps at the same slice position in the region of the right basal ganglia. A visual inspection of the clustering results in Figs. 11.1 and 11.2 (clusters #7 and #8) shows a close correspondence with the findings of these parameter maps. In addition, the unsupervised and self-organized clustering of pixels with similar signal dynamics allows a deeper insight in the spatiotemporal perfusion properties .

Figure 11.4 visualizes a method for comparative analysis of clustering results with regard to side differences of brain perfusion. The best-matching cluster #7, with the diffusion-weighted image corresponding to the infarct region in figure 11.1 is shown in figure 11.4a.

To better visualize the perfusion asymmetry between the affected and the nonaffected sides, a spatially connected region of interest (ROI) can be obtained from the clustering results by spatial low-pass filtering and thresholding of the given pixel cluster. The resulting ROI is

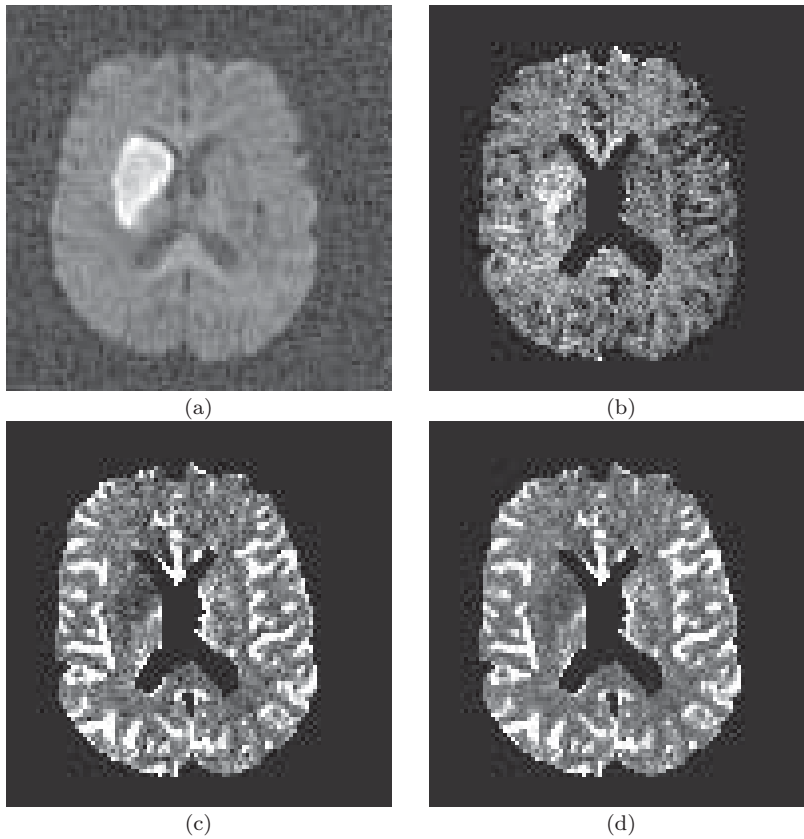


Figure 11.3
Diffusion-weighted MR image and conventional perfusion parameter maps of the same patient as in figures 11.1 and 11.2. (a) Diffusion weighted MR image; (b) MTT map; (c) rCBV map; (d) rCBF map.

shown in figure 11.4b (white region). In addition, a symmetrical contralateral ROI can be determined (light gray region). Then, the mean CTC values of all the pixels in the ROIs are determined and visualized in figure 11.4d, together with the corresponding quantitative perfusion parameters: the difference between the affected (figure 11.4c) and the nonaffected (figure 11.4d) sides with regard to CTC amplitude and dynamics is visualized, in agreement with highly differing corresponding quantitative perfusion parameters. Comparative quantitative analyses for fuzzy clustering based on deterministic annealing, the self-organizing

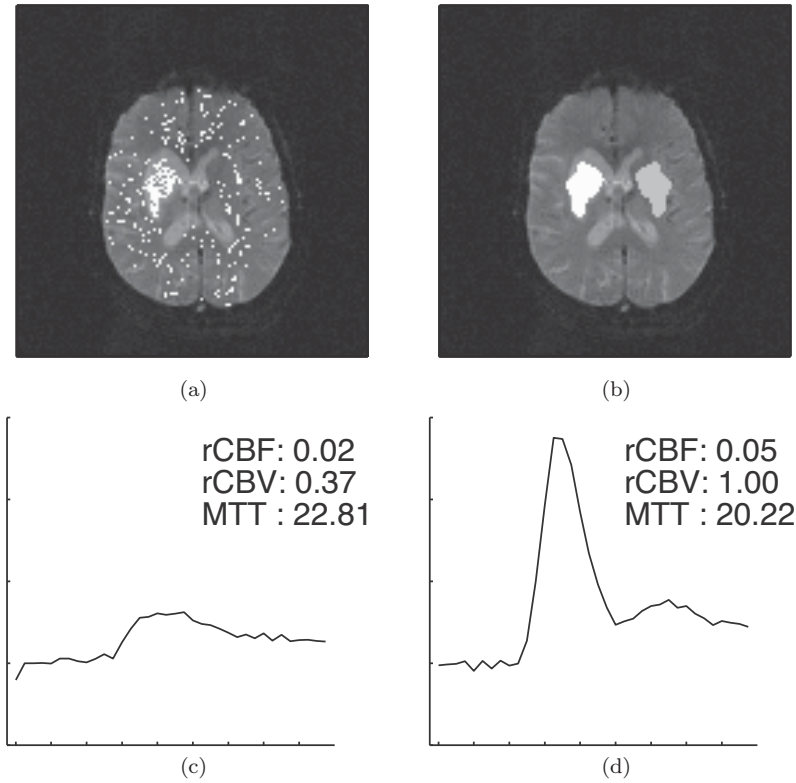


Figure 11.4
Quantitative analysis of the results for the "neural gas" network in figure 11.1 with regard to side asymmetry of brain perfusion. (a) Best-matching cluster #7 of figure 11.1 representing the infarct region; (b) contiguous ROI constructed from (a) by spatial low-pass filtering and thresholding (white), and a symmetrical ROI at an equivalent contralateral position (light gray); (c) average concentration-time curve of the pixels in the ROI of the affected side, (d) average concentration-time curve of the pixels in the ROI of the nonaffected side. For a better orientation, an anatomic EPI scan of the analyzed slice is underlaid in (a) and (b). The X-axis represents the scan number, and the Y-axis is arbitrary for (c) and (d).

map, and the fuzzy *c*-means vector quantization are shown in figures 11.5, 11.6, and 11.7, respectively.

The power of the clustering techniques is also demonstrated for a perfusion study in a control subject without evidence of cerebrovascular disease (see figures 11.8 and 11.9). The conventional perfusion parameter maps, together with a transversal T2-weighted scan at a corresponding

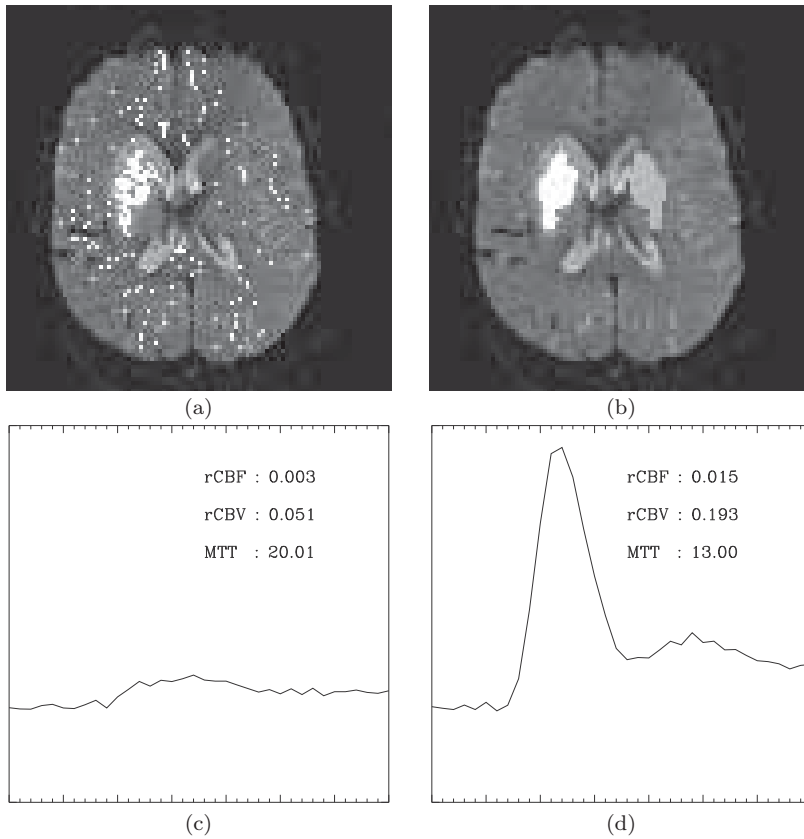


Figure 11.5
Quantitative analysis of clustering results with regard to side asymmetry of brain perfusion in analogy to figure 11.4 for vector quantization by fuzzy clustering based on deterministic annealing. For a better orientation, an anatomic EPI scan of the analyzed slice is underlaid in (a) and (b). The X-axis represents the scan number while the Y-axis is arbitrary for (c) and (d).

slice position, are presented in figure 11.10. Clusters #1, #3, #4, and #15 represent larger vessels located primarily in the cerebral sulci, while most of the other clusters seem to correspond to parenchymal vascularization. The important difference from the results of the stroke subject data in figures 11.1, 11.2, 11.3, and 11.5 is evident: the side-asymmetry with regard to both the temporal pattern and the amplitude of brain perfusion is here nonexistent. This fact becomes obvious since

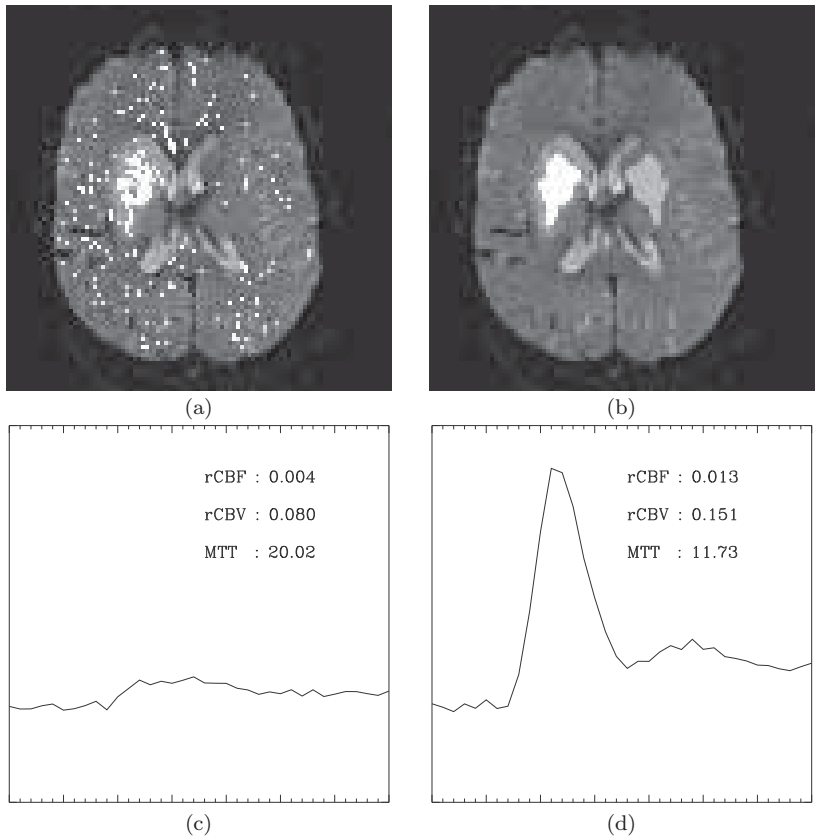


Figure 11.6
Quantitative analysis of clustering results with regard to side asymmetry of brain perfusion in analogy to figure 11.4 for vector quantization by a self-organizing map. For a better orientation, an anatomic EPI scan of the analyzed slice is underlaid in (a) and (b). The X-axis represents the scan number, and the Y-axis is arbitrary for (c) and (d).

each cluster in figure 11.1 contains pixels in roughly symmetrical regions of both hemispheres, different from the situation visualized in figure 11.1. In addition, no localized perfusion deficit results from the clustering. The clustering results of figures 11.8 and 11.9 match the information derived from the conventional perfusion parameter maps in figures 11.10b, c, and d.

The effectiveness of the different cluster validity indices and clus-

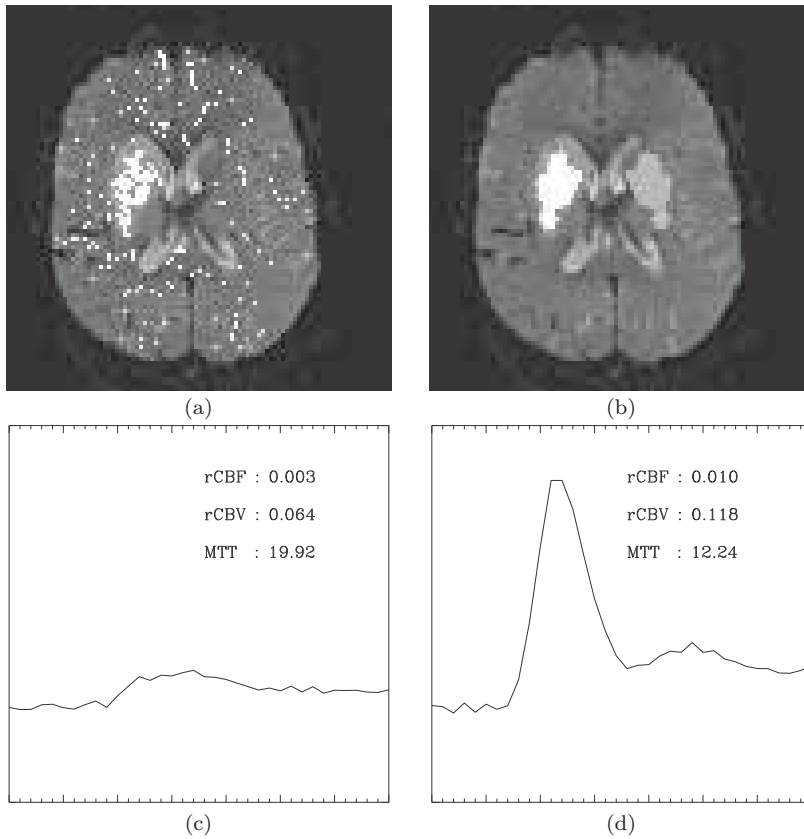


Figure 11.7
Quantitative analysis of clustering results with regard to side asymmetry of brain perfusion in analogy to figure 11.4 for fuzzy c-means vector quantization. For a better orientation, an anatomic EPI scan of the analyzed slice is underlaid in (a) and (b). The X-axis represents the scan number, and the Y-axis is arbitrary for (c) and (d).

tering methods in automatically evolving the appropriate number of clusters is demonstrated experimentally in the form of cluster assignment maps for the perfusion MRI data sets, with the number of clusters varying from 2 to 36.

Table 11.2 shows the optimal cluster number K^* obtained for each perfusion MRI data set, based on the different cluster validity indices.

Figures 11.11 and 11.12 show results for cluster-validity analysis for

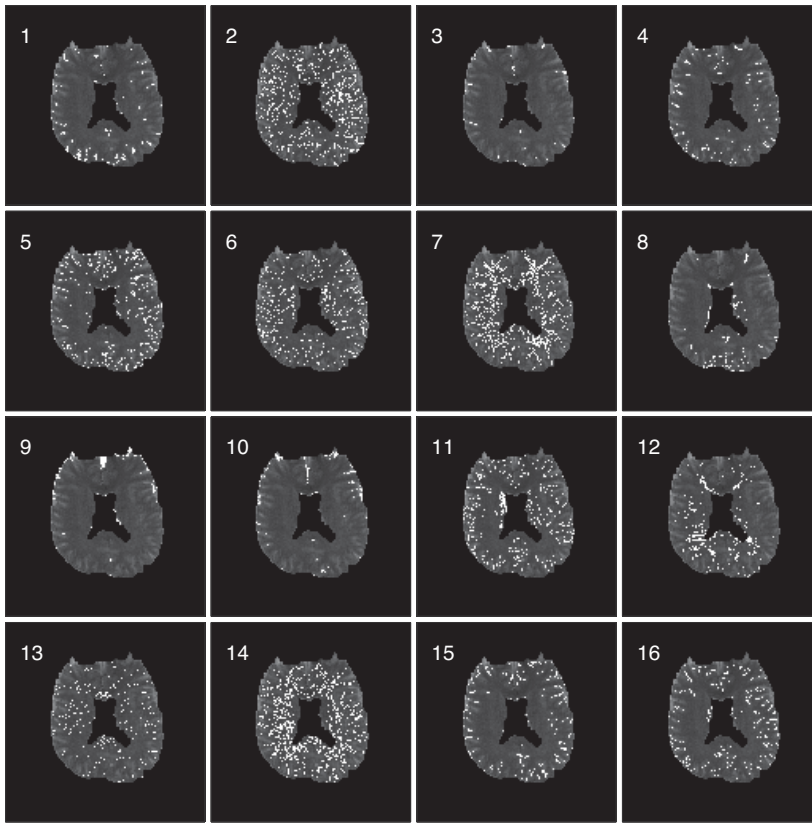


Figure 11.8
Cluster assignment maps for the "neural gas" network of a dynamic perfusion MRI study in a control subject without evidence of cerebrovascular disease. For a better orientation, an anatomic EPI scan of the analyzed slice is underlaid.

Table 11.2
Optimal cluster number K^* for the data sets #1 to #4, based on different cluster validity indices. The detailed curve for the cluster validity indices for data set #1 is shown in figures 11.11 and 11.12.

Index	#1	#2	#3	#4
K^*_{Kim}	18	6	10	12
K^*_{CH}	24	4	19	21
$K^*_{intraclass}$	3	3	3	3

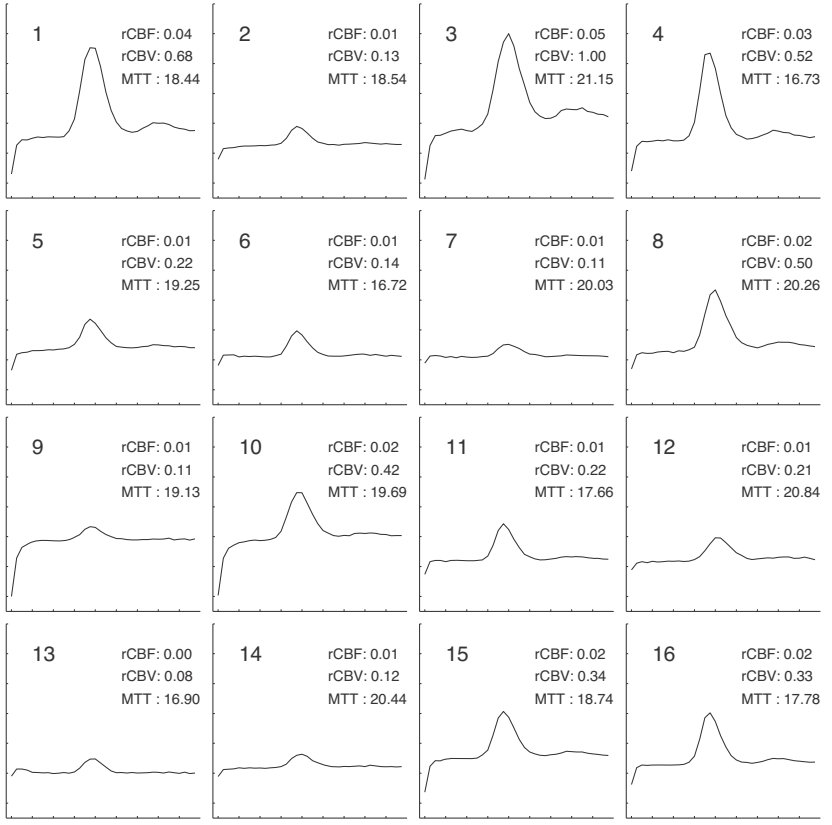


Figure 11.9
Cluster-specific concentration-time curves for the "neural gas" network of a dynamic perfusion MRI study in a control subject without evidence of cerebrovascular disease. Cluster numbers correspond to figure 11.8. The X-axis is the scan number, and the Y-axis is arbitrary.

data set #1, representing the minimal rCBV obtained by the minimal free energy VQ, and the values of the three cluster validity indices depending on cluster number. The cluster-dependent curve for the rCBVs was determined based on the minimal obtained rCBV value as a result of the clustering technique for fixed cluster numbers. For each of the twenty runs of the partitioning algorithms, the minimal codebook-specific rCBV was computed separately. The cluster whose CTC showed the minimal rCBV was selected for the plot. The MTT of this CTC is

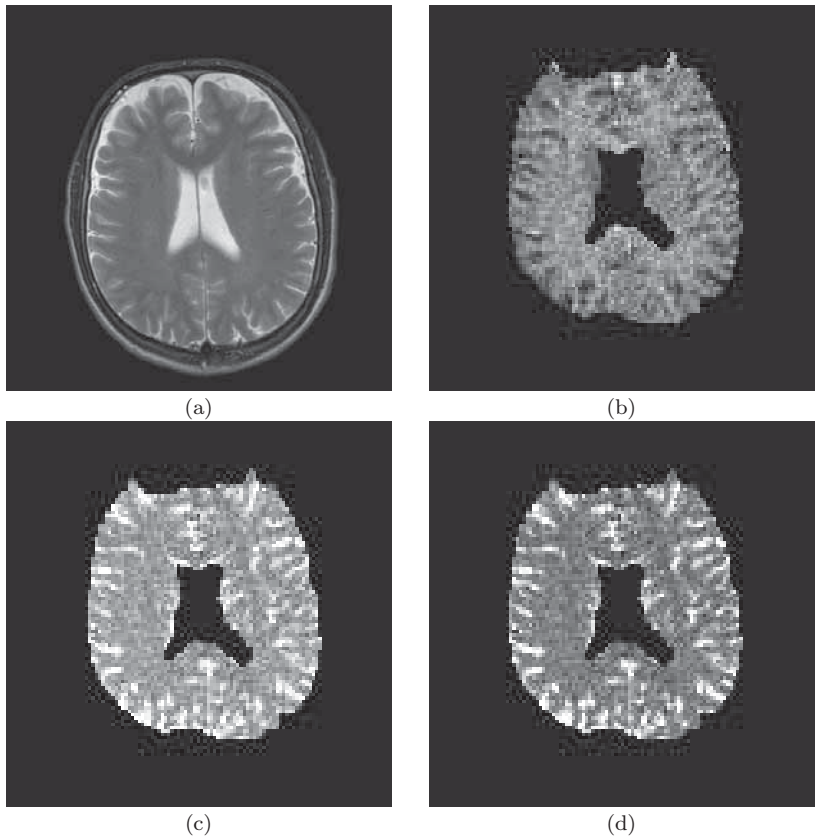


Figure 11.10
T2-weighted MR image and conventional perfusion parameter maps of the same subject as in figures 11.8 and 11.9. (a) T2-weighted MR image; (b) MTT map; (c) rCBV map; (d) rCBF map.

indicated in the plot as well. The bottom part of the figure shows the cluster assignment maps for cluster numbers corresponding to the optimal cluster number K^* and $K = K^* \pm 1$. The cluster assignment maps correspond to the cluster-specific concentration-time curves exhibiting the minimum rCBV.

The results show that based on the indices K_{Kim} and $K_{Intraclass}$, a larger number of clusters is needed to represent the data sets #1, #3, and #4.

In the following, the results of the quantitative ROC analysis are

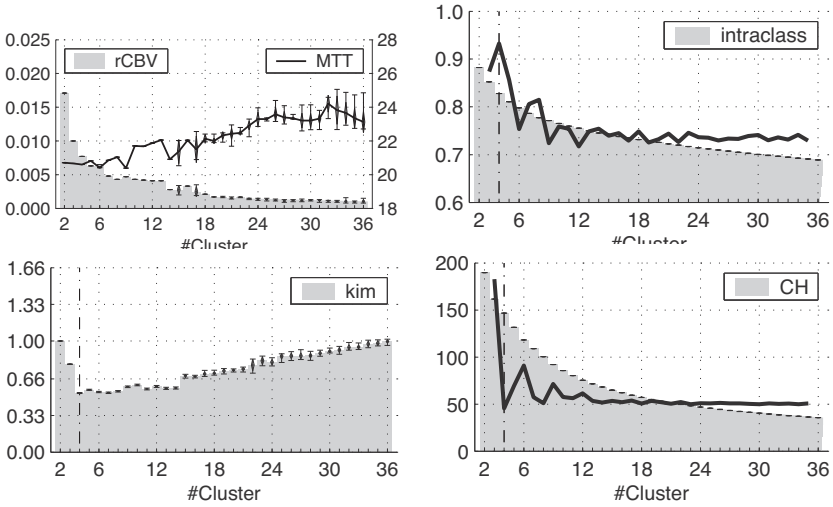


Figure 11.11
Visualization of the minimal rCBV curve and the curves for the three cluster validity indices – Kim's index, the Calinski-Harabasz (CH) index, and the intraclass index for data set #1 – and as a result of classification based on the minimal free energy VQ. The cluster number varies from 2 to 36. The average, minimal and maximal values of 20 different runs using the same parameters but different algorithms' initializations are plotted as vertical bars. For the intraclass and Calinski-Harabasz validity indices, the second derivative of the curve is plotted as a solid line.

presented. An ROC curve for subject 1 in figure 11.13, using the "neural gas" network with $N = 16$ codebook vectors as the clustering algorithm, is shown.

The clustering results are given for four subjects: subject 1 (stroke in the right basal ganglia), subject 2 (large stroke in the supply region of the middle cerebral artery, left hemisphere, and subjects 3 and 4 (both with no evidence of cerebrovascular disease). The codebook vectors from 3 to 36 for the proposed algorithms were varied, and an ROC analysis using two different performance metrics was performed: the classification outcome regarding the discrimination of the concentration-time curves based on the rCBV value and the discrimination capability of the codebook vectors based on their MTT value. The ROC performances for the four subjects are shown in figure 11.14. The figure illustrates the average area under the curve and its deviations for 20 different ROC runs using the same parameters but different algorithms' initializations. The

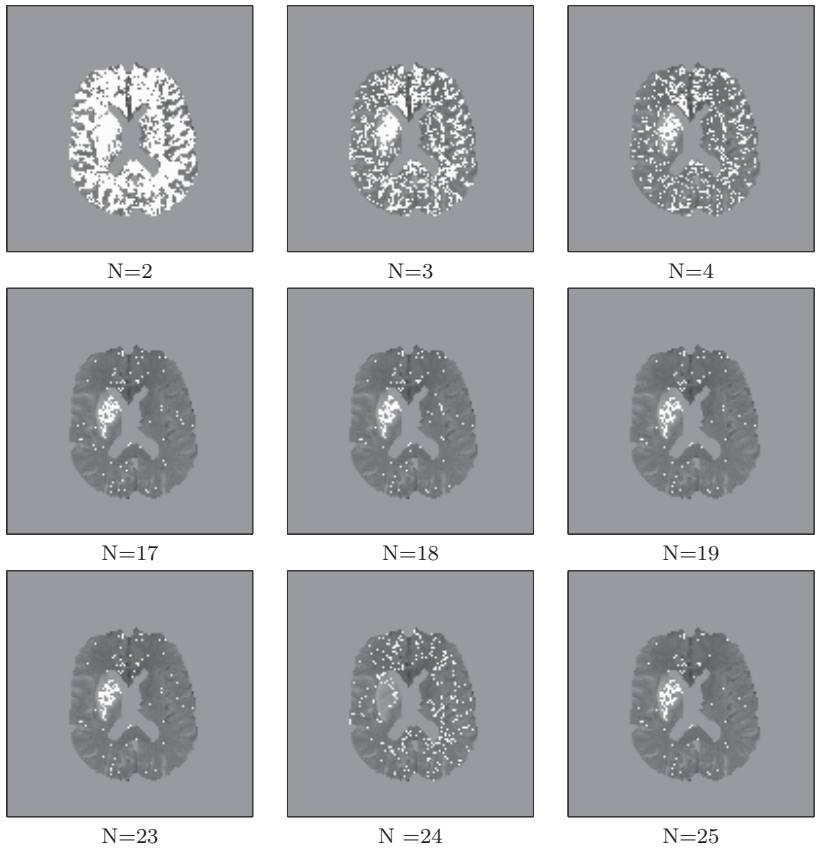


Figure 11.12
Cluster assignment maps for cluster numbers corresponding to the optimal cluster number K^* and $K = K^* \pm 1$. The cluster assignment maps correspond to the cluster-specific concentration-time curves exhibiting the minimum rCBV.

ROC analysis shows that rCBV outperforms MTT with regard to its diagnostic validity when compared to the conventional analysis serving as the gold standard in this study, as can be seen from the larger area under the ROC curve for rCBV.

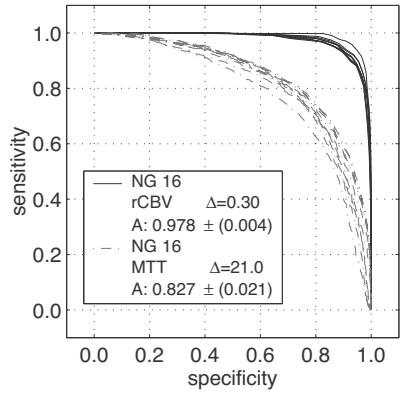


Figure 11.13
ROC curve of the cluster analysis of data set for subject 1 analyzed with the "neural gas" network for N=16 codebook vectors. "A" represents the area under the ROC curve, and Δ the threshold for rCBV/MTT.

11.3 General Aspects of Time Series Analysis Based on Unsupervised Clustering in Dynamic Cerebral Contrast-enhanced Perfusion MRI

The advantages of unsupervised self-organized clustering over the conventional and single extraction of perfusion parameters are the following:

1. Relevant information given by the signal dynamics of MRI time series is not discarded.
2. A nonbiased interpretation that results from the indicator-dilution theory of nondiffusible tracers only for an intact blood-brain barrier.

Nevertheless, clustering results support the findings from the indicator-dilution theory, since conventional perfusion parameters like MTT, rCBV, and rCBF values can be derived directly from the resulting prototypical cluster-specific CTCs.

The proposed clustering techniques were able to unveil regional differences of brain perfusion characterized by subtle differences of signal amplitude and dynamics. They could provide a rough segmentation with regard to vessel size, detect side asymmetries of contrast-agent first pass, and identify regions of perfusion deficit in subjects with stroke.

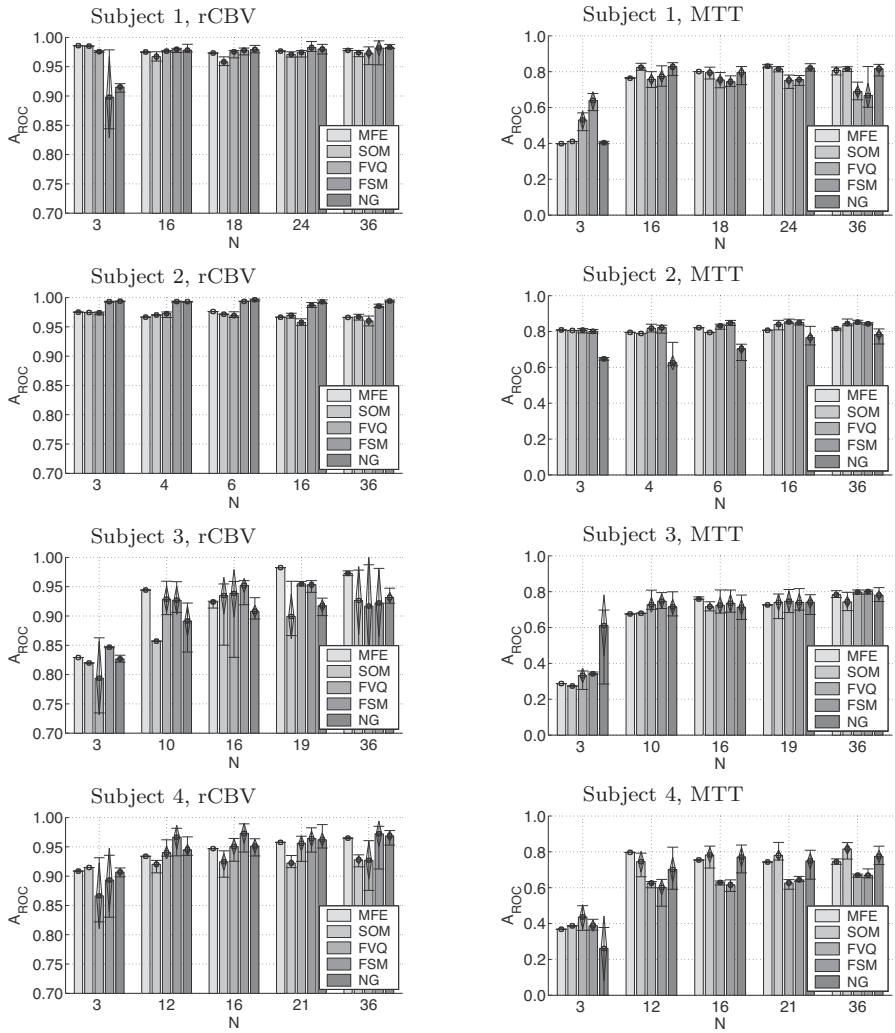


Figure 11.14
Results of the comparison between the different clustering analysis methods on perfusion MRI data. These methods are minimal free energy VQ (MFE), Kohonen's map (SOM), the "neural gas" network (NG), fuzzy clustering based on deterministic annealing, fuzzy *c*-means with unsupervised codebook initialization (FSM), and the fuzzy *c*-means algorithm (FVQ) with random codebook initialization. The average area under the curve and its deviations are illustrated for 20 different ROC runs using the same parameters but different algorithms' initializations. The number of chosen codebook vectors for all techniques is between 3 and 36, and results are plotted for four subjects. Subjects 1 and 2 had a subacute stroke, while subjects 3 and 4 gave no evidence of cerebrovascular disease. The ROC analysis is based on two performance metrics: regional cerebral blood volume (rCBV) (left column) and mean transit time (MTT) (right column). See plate 9.

In general, a minimal number of clusters is necessary to obtain a good partition quality of the underlying data set, which leads to a higher area under the ROC curve. This effect can clearly be seen for subjects 3 and 4. For the data sets of subjects 1 and 2, the cluster number doesn't seem to play a key role. A possible explanation of this aspect is the large extent of the infarct area. Thus, even with a smaller number of codebook vectors, it becomes possible to obtain a good separation of the stroke areas from the rest of the brain. Any further partitioning, obtained by increasing the number of codebook vectors, is not of crucial importance - the area under the curve does not change substantially. Also, for the patients without evidence of a cerebrovascular disease, the area under the ROC curve is smaller than that for the subjects with stroke.

Three important aspects remain to be discussed: the interpretation of the codebook vector, the normalization of the signal time curves, and the relatively high MTT values.

A codebook vector can be specified as a time series representing the center (i.e., average) of all the time series belonging to a cluster. Here, a cluster represents a set of pixels whose corresponding time series are characterized by similar signal dynamics. Thus, "codebook vectors" as well as "clusters" are defined in an operational way that - at a first glance - does not refer to any physiological implications. However, it is common practice in the literature to conjecture [84] that similar signal characteristics may be induced by similar physiological processes or properties, although this cannot be proven definitely. It is very interesting to observe that the average values for the areas under the ROC curves seem to be higher for the patients with stroke in comparison to the patients without stroke. So far, no explanation can be given for this, but it may be an important subject for further examination in future work. The different numbers of codebook vectors used for different subjects can be explained as follows: 16 and 36 codebook vectors were used for clustering in all data sets. In addition, the optimal number of clusters was determined by a detailed analysis using several "cluster-validity criteria": Kim [138], Calinski, and Harabazs (CH) [39], and intraclass [97].

In biomedical MRI time series analysis considered here, a similar problem is faced: It is certainly not possible to interpret all details of the signal characteristics of the time series belonging to each pixel of the data set as known physiological processes. Nevertheless, it may be a use-

ful hypothesis to interpret the time series of at least some clusters in the light of physiological meta knowledge, although a definite proof of such an interpretation will be missing. Hence, such an approach is certainly biased by subjective interpretation on the part of the human expert performing this interpretation of the resulting clusters, and thus, may be subject to error. In summary, it is not claimed that a specific cluster is well-correlated with physiological phenomena related to changes of brain perfusion, although one cannot exclude that a subjective interpretation of some of these clusters by human experts may be useful to generate hypotheses on underlying physiological processes in the sense of exploratory data analysis. These remarks are in full agreement with the whole body of literature dealing with unsupervised learning in MRI time series analysis, such as [84] and [53].

The normalization of signal time-curves represents an important issue where the concrete choice depends on the observer's focus of interest. If cluster analysis is to be performed with respect to signal dynamics rather than amplitude, clustering should be preceded by time series normalization. While normalization may lead to noise amplification in low-amplitude CTCs, in cluster analysis of signal time series, preceding normalization is an option. However, CTC amplitude unveils important clinical and physiological information, and therefore it forms the basis of the reasoning for not normalizing the signal time-curves before they undergo clustering.

In order to provide a possible explanation of the relatively high MTT values obtained in the results, the following should be mentioned. The rationale for using equation (11.3) for computing MTT is that the arterial input function, which is difficult to obtain in routine clinical diagnosis, was not determined. The limitations of such an MTT computation have been addressed in detail in the theoretical literature on this topic (e.g., [299]). In particular, it has been pointed out that the signal intensity changes measured with dynamic MR imaging are related to the amount of contrast material remaining in the tissue, not to the efflux concentration of contrast material. Therefore, if a deconvolution approach using the experimentally acquired arterial input function (e.g., according to [149, 281]), is not performed, equation (11.3) can be used only as an approximation for MTT. However, this approximation has been widely used in the literature on both myocardial and cerebral MRI perfusion studies (e.g., [106, 219, 283]).

In summary, the study shows that unsupervised clustering results are in good agreement with the information obtained from conventional perfusion parameter maps, but may sometimes unveil additional hidden information (e.g., disentangle signals with regard to different vessel sizes). In this sense, clustering is not a competitive, but a complementary, additional method that may extend the information extracted from conventional perfusion parameter maps by taking into account fine-grained differences of MRI signal dynamics in perfusion studies. Thus, the presented techniques can contribute to exploratory visual analysis of perfusion MRI data by human experts as a complementary approach to conventional perfusion parameter maps. They provide computer-aided support to appropriate data processing in order to assist the neuroradiolgist, and not to replace his/her interpretation. In addition, following further pilot studies on larger samples, the nature of additional information can be better clarified, as the proposed techniques should be applicable in a larger group to assess validity and reliability. In conclusion, clustering is a useful extension to conventional perfusion parameter maps.

12 Skin Lesion Classification

This chapter describes an application of biomedical image analysis: the detection of malignant and benign skin lesions by employing local information rather than global features. For this we will build a neural network model in order to classify these different skin lesions by means of ALA-induced fluorescence images. After various image preprocessing steps, eigenimages and independent base images are extracted using PCA and ICA. In order to use local information in the images rather than global features, we first add self-organizing maps (SOM) to cluster patches of the images and then extract local features by means of ICA (local ICA). These components are used to distinguish skin cancer from benign lesions. An average classification rate of 70% is achieved, which considerably exceeds the rate obtained by an experienced physician. These PCA- and ICA-based tumor classification ideas have been published in [21] and extend previous work presented in [19].

12.1 Biomedical Image Analysis

Many kinds of biomedical data, such as fMRI, EEG, and optical imaging data, form a challenge to any data-processing software due to their high dimensionality. Low-dimensional representations of these signals are key to solving many of the computational problems. Therefore, principal component analysis (PCA) commonly was used in the past to provide practically useful and compact representations. Furthermore, PCA was successfully applied to the classification of images [272]. One major deficiency of PCA is its global, orthogonal representation, which often cannot extract the intrinsic information of high-dimensional data.

Independent component analysis (ICA) is a generalization of principal component analysis which decorrelates the higher-order moments of the input in addition to the second-order moments. In a task such as image recognition, much of the important information is contained in the higher-order statistics of the image. Hence ICA should be able to extract local feature like structures of objects, such as fluorescence images of skin lesions. Bartlett demonstrated that ICA outperformed the face recognition performance of PCA [18]. Finally, local ICA was

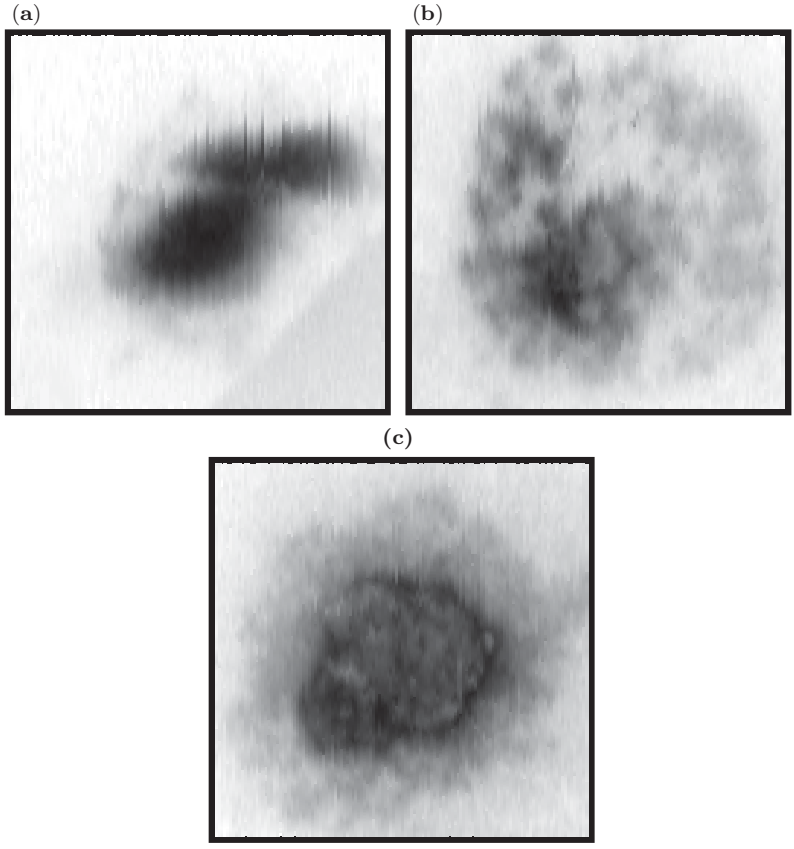


Figure 12.1
Typical fluorescence images of psoriasis (a), actinic keratosis (b), and a basal cell carcinoma (c).

developed by Karhunen and Malaroiu to take advantage of the localized features in high-dimensional data [132]. Using Kohonen's self-organizing maps [140], multivariate data are first split into clusters and then local features are extracted using ICA within these clusters.

Here, we intend to classify *skin lesions* (*basal cell carcinoma*, *actinic keratosis*, and *psoriasis* plaques) through their fluorescence images (see figures 12.1 and 12.2).

Even an experienced physician is unable to distinguish malignant from the benign lesions when fluorescence images are taken. For the

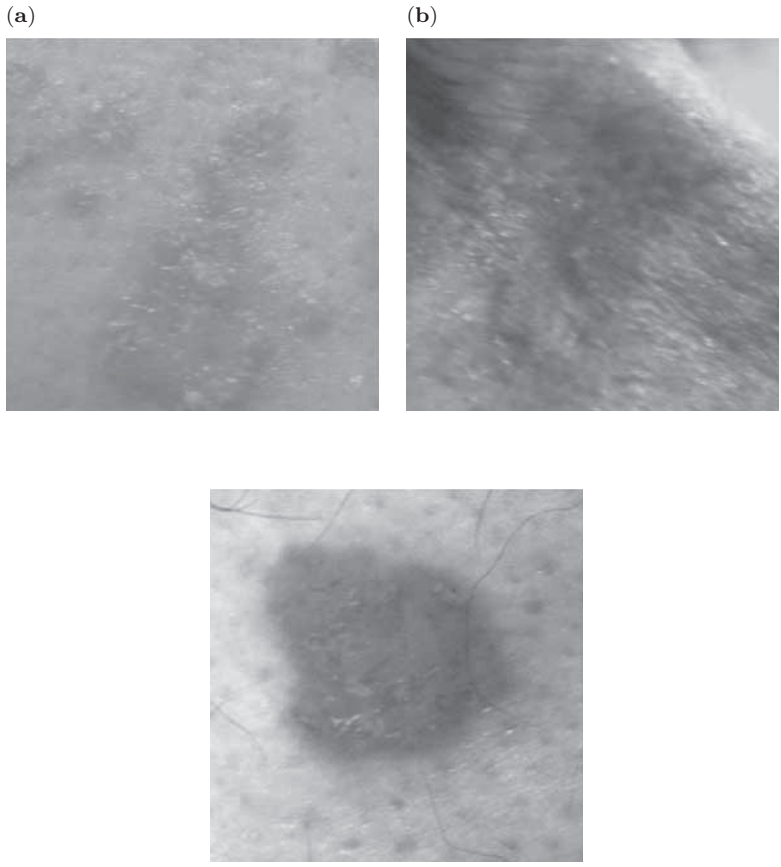


Figure 12.2
Nonfluorescence images of psoriasis (a), actinic keratosis (b), and basal cell carcinoma.

sake of simplicity, we will just denote the diseases as malignant, since basal cell carcinoma is a skin cancer and actinic keratosis is considered a premalignant condition.

12.2 Classification Based on Eigenimages

PCA is a well-known method for feature extraction and was successfully applied to face recognition tasks by Turk and Pentland [272], Bartlett

et al. [17, 18] and others. Thereby images are decomposed into a set of orthogonal feature images called *eigenimages*, which can then be used for classification. A new image is first projected into the PCA subspace spanned by the eigenimages. Then image recognition is performed by comparing the position of the test image with the position of known images, using the reconstruction error as the recognition criterion. For a statistical analysis of the obtained results, hypothesis testing is used for a reliable classification.

Calculation of the eigenimages

Consider a set of m images $\mathbf{x}_1, \ldots, \mathbf{x}_m$ with each image vector

$$\mathbf{x}_i = [x_i(1), \ldots, x_i(N^2)]^\top$$

comprising N^2 pixel values of the $N \times N$ image i. Merge the whole set of images into an $N^2 \times m$ matrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_m]$ and assume the expectation value $E\{\mathbf{x}_i\}$ of each image vector to be zero.
Then the covariance matrix can be calculated according to

$$\mathrm{Cov}(\mathbf{X}) = \frac{1}{m} \sum_{i=1}^{m} \mathbf{x}_i \mathbf{x}_i^\top = \mathbf{X}\mathbf{X}^\top.$$

A set of N^2 orthogonal eigenimages $\mathbf{u}_i$ can now be determined by solving the following eigenvalue problem:

$$\mathbf{X}\mathbf{X}^\top \mathbf{u}_i = \mathbf{\Sigma} \mathbf{u}_i, \tag{12.1}$$

where $\mathbf{\Sigma} = \mathrm{diag}\,[\sigma_1, \ldots, \sigma_{N^2}]$ denotes the diagonal matrix with the variances σ_i of the projections $r_i = \mathbf{x}_i^\top \mathbf{u}_i = \mathbf{u}_i^\top \mathbf{x}_i$.

As solving the eigenvalue problem for large matrices (i. e. , for the reduced fluorescence images we still deal with a $128^2 \times 128^2$ covariance matrix) proves computationally very demanding, Turk and Pentland introduced the following dimension reduction technique [272] :
Consider the eigenvalue system

$$\mathbf{X}^\top \mathbf{X} \mathbf{v}_i = \lambda_i \mathbf{v}_i, \tag{12.2}$$

where $\mathbf{v}_i$ denotes an eigenvector with its corresponding eigenvalue λ_i.

Premultiplying equation (12.2) with $\mathbf{X}$ results in

$$\begin{aligned} \mathbf{X}\mathbf{X}^\top\mathbf{X}\mathbf{v}_i &= \mathbf{X}\lambda_i\mathbf{v}_i \\ \mathrm{Cov}(\mathbf{X})\mathbf{X}\mathbf{v}_i &= \lambda_i\mathbf{X}\mathbf{v}_i, \end{aligned} \tag{12.3}$$

thus indicating that $\mathbf{X}\mathbf{v}_i$ also is also eigenvector of the covariance matrix $\mathrm{Cov}(\mathbf{X})$. Define an $m \times m$ matrix

$$\mathbf{L} = (l_{ij})_{0<i\leq m, 0<j\leq m} = \mathbf{X}^\top\mathbf{X}$$

and its corresponding eigenvectors $\mathbf{v}_l$. Then, using equation (12.3), the calculation of the eigenvectors of the covariance matrix can be accomplished by the linear combination

$$\mathbf{u}_l = \sum_{i=1}^{m} \mathbf{v}_l^{(i)}\mathbf{x}_i,$$

where $\mathbf{v}_l^{(i)}$ denotes the ith component of the lth eigenvector $\mathbf{v}_l$. Thus the number of calculations is greatly reduced from the order of the number of pixels N^2 in the images to the order of the number of images m in the training ensemble.
Note that the associated eigenvalues imply a ranking of the eigenvectors according to their usefulness in characterizing the variation among the images. The first four eigenimages of an ensemble of psoriasis are displayed in figure 12.3.

Classification based on the reconstruction error

Using eigenimages, a classification criterion can be defined, based on the reconstruction error of images. Therefore consider a set of eigenimages $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_m]$ computed as in the previous section. Furthermore, calculate the projections r_i of the original images $\mathbf{x}_i$ into the PCA space, following

$$\mathbf{r}_i = \mathbf{U}^\top\mathbf{x}_i. \tag{12.4}$$

Now a dimension reduction can be accomplished, transforming the projections r_i back into the input space, using only a subset of the $m' < m$ eigenimages $u_1, \ldots, u_{m'}$ with the largest eigenvalues. Thus the images $\mathbf{x}_i$ can be reconstructed, thereby generating the reconstruction error

$$\varepsilon_i = \|\mathbf{x}_i - \mathbf{x}_i^{\mathrm{rec}}\|. \tag{12.5}$$

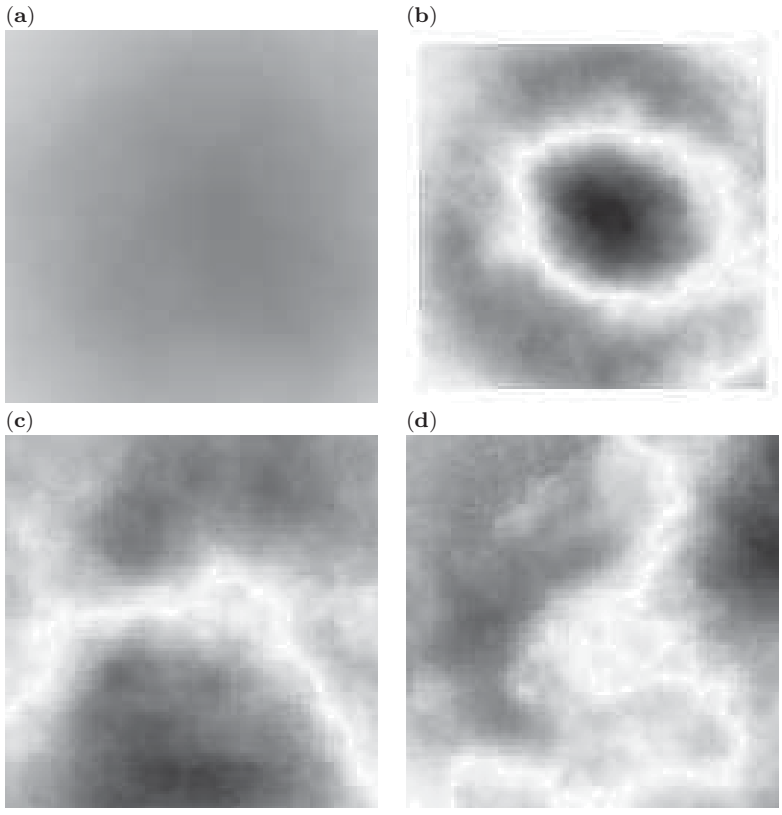


Figure 12.3
The first four eigenimages of a psoriasis ensemble, displayed according to their variances. Note the increasing localized structures in the eigenimages.

For an ensemble of images, a *relative reconstruction error* $\varepsilon_i^{\text{rel}}$ can be defined for an image $\mathbf{x}_i$ according to

$$\varepsilon_i^{\text{rel}} = -\frac{\varepsilon_i - \varepsilon_{\max}}{\varepsilon_{\max} - \varepsilon_{\min}}, \tag{12.6}$$

where $\varepsilon_{\max} = \max\{\varepsilon_i\}$ and $\varepsilon_{\min} = \min\{\varepsilon_i\}$, respectively.

12.3 Classification Using Independent Base Images

In tasks such as image classification, much of the important information may be contained in the higher-order correlations among the image pixels. As PCA is based on second-order statistics only, it does not take into account higher-order statistical dependencies which can be addressed by ICA. Thus not only decorrelation, but also statistical independence of the signals, can be achieved, thereby allowing us to extract relevant information which is coded in higher-order statistics.

By analogy to the eigenimages of the previous section, here we separate images across space and thus extract a set of statistically independent base images which may capture some independent features of the corresponding ensemble.

Statistically independent base images

By analogy to the eigenimages in the previous section, assume all fluorescence images $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_m]^\top$ with $\mathbf{x}_i = [x_i(1), \ldots, x_i(N^2)]$ to be a linear combination (*mixture*) of m source images $\mathbf{S}$ according to the image synthesis model in figure 12.4. As the mixing matrix $\mathbf{A}$ is unknown, these source images, have to be recovered by a matrix $\mathbf{W}_I$ which produces statistically independent output according to $\mathbf{Y} = \mathbf{W}_I\mathbf{X}$. As already mentioned, these base images $\mathbf{Y}$ can be considered an ensemble of independent (localized) features in the images and the coefficients for the linear combinations of the independent base images $\mathbf{Y}$, which comprise each image $\mathbf{x}_i$, are represented by the matrix $\mathbf{A} = \mathbf{W}_I^{-1}$.

In order to be able to control the number of recovered source images extracted by an ICA algorithm, learning is performed on the first m principal component eigenimages, which are calculated as in the previous section. Thus let $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_{m'}]$ denote the $N^2 \times m'$ matrix containing the first m' eigenimages $\mathbf{u}_i = [u_i(1), \ldots, u_i(N^2)]^\top$ in its columns.
After a random initialization of the weight matrix $\mathbf{W}$, the input data are sphered, using a sphering matrix $\mathbf{W}_z$; thus the unmixing matrix is given by $\mathbf{W}_I = \mathbf{W}\mathbf{W}_z$. ICA is then performed on $\mathbf{U}^\top$ (i. e. , at each step m' pixels at the same location in the different eigenimages are presented to the network). Thereby the Infomax learning rule with the natural

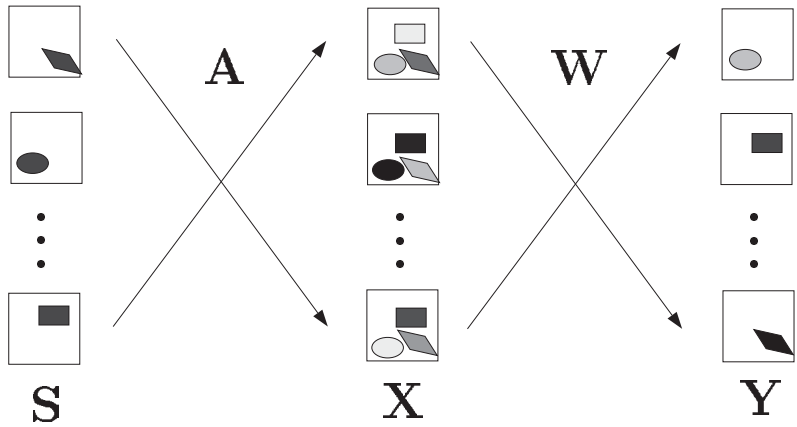


Figure 12.4
Image synthesis model. The recorded fluorescence images $\mathbf{X}$ are considered to be a set of linearly mixed source images $\mathbf{S}$ according to $\mathbf{X} = \mathbf{AS}$. The underlying independent base images $\mathbf{Y}$ can be estimated by determining the unmixing matrix $\mathbf{W}$, where the indetermination of ICA with regard to scaling and permutation remains.

gradient extension is used, according to [8] and [26]:

$$\mathbf{W}(t+1) = \mathbf{W}(t) + \eta \left(\mathbf{I}_{m'} - \boldsymbol{\varphi}(\mathbf{y})\mathbf{y}^\top \right) \mathbf{W},$$

where η controls the learning rate and $\mathbf{I}_{m'}$ denotes the $m' \times m'$ identity matrix. Due to the computational complexity only a fixed sigmoidal score function $\boldsymbol{\varphi}(\mathbf{y})$ is used, thus reducing the required calculation to a large extent. Learning is stopped when

$$\frac{1}{m'^2} \sum_{i,j}^{m'} |w_{ij}(t+1) - w_{ij}(t)| < \varepsilon,$$

where ε was commonly chosen as $\varepsilon = 0.0001$. Therefore, $50,000$ iterations proved necessary at a learning rate of $\eta = 0.01$ for the convergence of the network.

Classification via base coefficients

By analogy to the reconstruction using eigenimages, the coefficients

$$\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_m]^\top$$

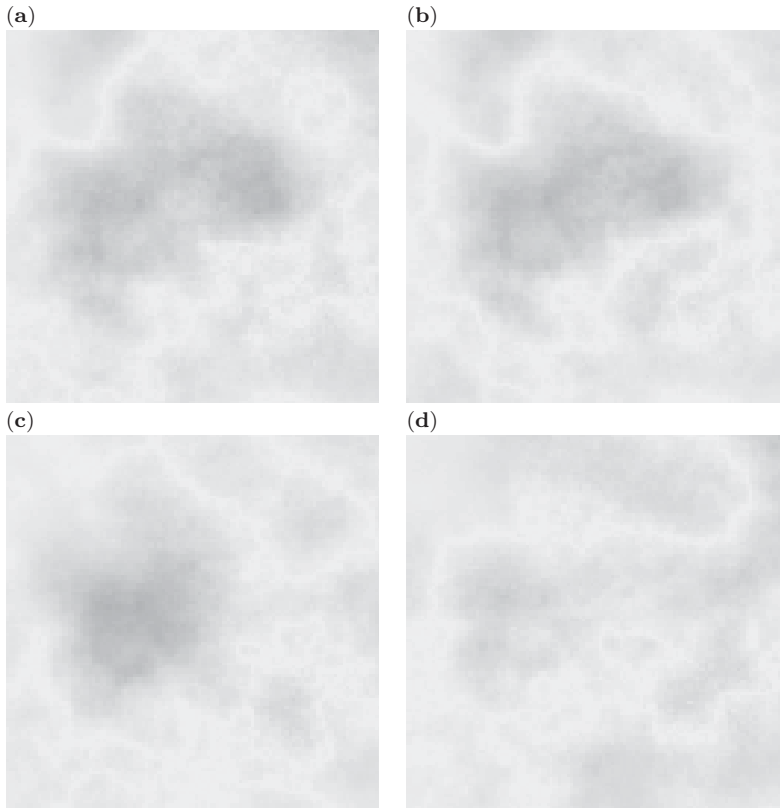


Figure 12.5
Four independent base images of the psoriasis ensemble. By analogy to the PCA eigenimages, these base images are considered the underlying images of the recorded fluorescence images as clarified by the image synthesis model shown in figure 12.4. However, note that none of the independent base images shows large structures like the first PCA eigenimages in figure 12.3.

for the linear combinations of the independent source images $\mathbf{Y}$ that comprise the original fluorescence images $\mathbf{X}$ have to be determined next. Therefore, consider that instead of the original fluorescence images $\mathbf{X}$, their corresponding eigenimages $\mathbf{U}$ are used to train the network, and hence

$$\mathbf{W}_I \mathbf{U}^\top = \mathbf{Y}. \tag{12.7}$$

As demonstrated in the previous section, PCA reconstruction is obtained by first calculating the projections onto the eigenimages according to $\mathbf{r}_i = \mathbf{U}^\top \mathbf{x}_i$, or in matrix notation

$$\mathbf{R} = \mathbf{X}\mathbf{U},$$

and then performing a subsequent backtransformation into the original system according to

$$\mathbf{X}_{\text{rec}} = \mathbf{R}\mathbf{U}^\top. \tag{12.8}$$

Solving equation (12.7) for $\mathbf{U}^\top$ and plugging into equation (12.8) leads to

$$\begin{aligned} \mathbf{X}^{\text{rec}} &= \mathbf{R}\mathbf{W}_I^{-1}\mathbf{Y} = \mathbf{B}\mathbf{Y} \\ \mathbf{x}_i^{\text{rec}} &= \mathbf{b}_i\mathbf{Y}, \end{aligned}$$

where the rows of $\mathbf{B}$ contain the coefficients for the linear combinations of the statistically independent sources $\mathbf{Y}$. The resulting basis images are shown in figure 12.5.

Image classification can now be performed by evaluating the coefficient vectors $\mathbf{b}_i$ for different image ensembles. Therefore the projections onto the eigenimages $\mathbf{U}$ are calculated for a training set and a test set according to

$$\mathbf{R}^{\text{test}} = \mathbf{X}^{\text{test}}\mathbf{U} \quad \text{and} \quad \mathbf{R}^{\text{train}} = \mathbf{X}^{\text{train}}\mathbf{U} \tag{12.9}$$

by analogy to the previous section. Then the coefficient matrices $\mathbf{B}^{\text{train}}$ and $\mathbf{B}^{\text{test}}$ are determined using the learned unmixing matrix $\mathbf{W}$, following

$$\mathbf{B}^{\text{train}} = \mathbf{R}^{\text{train}}\mathbf{W}^{-1} \quad \text{and} \quad \mathbf{B}^{\text{test}} = \mathbf{R}^{\text{test}}\mathbf{W}^{-1}.$$

Image recognition performance can now be computed by first calculating the similarity of the coefficient vectors as evaluated by the cosine of the angle between them, according to

$$d_{ij} = \frac{\mathbf{b}_i^{\text{test}}\mathbf{b}_j^{\text{train}}}{\|\mathbf{b}_i^{\text{test}}\|\,\|\mathbf{b}_j^{\text{train}}\|}, \tag{12.10}$$

and finally assigning each test image i' the class label of the training image j' with $d_{i'j'} = \max\{d_{ij}\}$.

12.4 Classification Using Local Features Extracted by ICA

Local ICA was proposed by Karhunen and Malaroiu to take advantage of localized features in high dimensional data [132]. Although standard ICA yields meaningful results in many cases, it can only provide a crude approximation for nonlinear data distributions. Therefore, self organizing maps [140] are used to split multivariate data into various clusters, followed by a local feature extraction using ICA within these clusters.

Clustering the data applying SOM

First, the images are split up into n_0 square patches such that the patches overlap with at least two pixels. Thus the statistical structure in the image data can be conserved. However, sometimes parts of the images are no longer covered by the patches, a fact which can be neglected due to the little information contained in the outer image areas.

For the SOM, a one-dimensional chain with n_0 neurons is used, whereby each neuron is supposed to finally learn one certain type of image patch. After a random initialization of the synaptic weights $\mathbf{w}^{(j)}, 0 < j \leq n_0$, the patch vectors are presented to the network. Learning is accomplished following Kohonen's algorithm, according to

$$\Delta w_i^{(j)}(t) = \eta(t)\,\Lambda\left(\|\mathbf{w}^{(j)}(t) - \mathbf{w}^{(j^*)}(t)\|, t\right) \cdot \left[x_i(t) - w_i^{(j)}(t)\right],$$

with $\eta(t)$ time-dependent learning rate and neighborhood function $\Lambda\left(\|\mathbf{w}^{(j)}(t) - \mathbf{w}^{(j^*)}(t)\|, t\right)$ with the following properties:

$$\lim_{\|\mathbf{w}^{(j)}(t) - \mathbf{w}^{(j^*)}(t)\| \to \infty} \Lambda\left(\|\mathbf{w}^{(j)}(t) - \mathbf{w}^{(j^*)}(t)\|, t\right) = 0$$

$$\lim_{t \to \infty} \Lambda\left(\|\mathbf{w}^{(j)}(t) - \mathbf{w}^{(j^*)}(t)\|, t\right) = 0.$$

Thereby, typically $T = 1000$ iterations proves necessary for convergence. In order to allow a fast spreading of the Kohonen chain at the beginning of the simulations, the neighborhood function Λ is set to 1 and hence all neurons are updated at each iteration. While learning proceeds, the influence of the neighborhood function is reduced and the

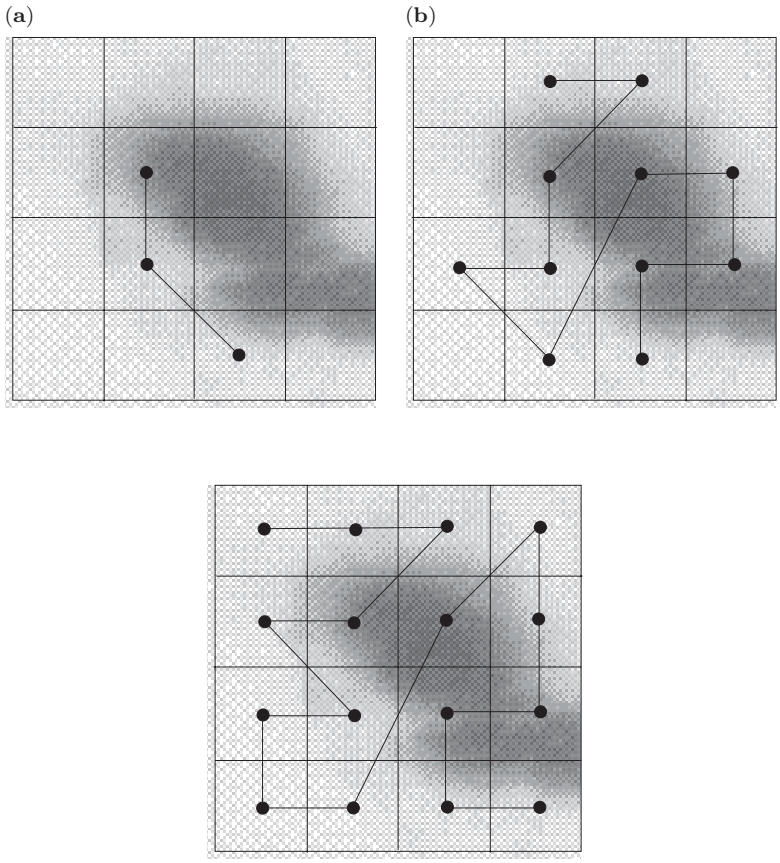


Figure 12.6
Typical spreading of a Kohonen chain over 16 patches of a psoriasis image. Only patches which can already be represented adequately are indicated by a corresponding neuron. After convergence of the network, all patches can be described by the weight vectors $\mathbf{w}^{(j)}$ of the Kohonen chain.

learning rate $\eta(t)$ is decreased. Thus the one-dimensional Kohonen chain quickly spreads over the image data at the beginning of the simulation and subsequently is fine-tuned for an accurate representation of the corresponding patch cluster in the end.

This procedure is illustrated in fig 12.6, where the spreading of a Kohonen chain over 16 patches of a psoriasis image is displayed: Note that for the sake of clarity, only patches which can already be

represented sufficiently exactly are indicated by a corresponding neuron. Although the synaptic weights are randomly initialized, three patches can already be appropriately represented by neurons. After a rapid spreading of the chain over the image at a high learning rate $\eta(t)$, all patches are covered and the neurons are finally adapted such that their corresponding clusters are best represented.

Classification via averaged base coefficients

Once the algorithm has converged, ICA is applied within the sets of similar patches. The similarity coefficients are evaluated for every patch individually with regard to the different clusters and image ensembles, following

$$d_{ij}^{(l)} = \frac{(\mathbf{b}_i^{(l)})_{\text{train}}(\mathbf{b}_j^{(l)})_{\text{test}}}{\|(\mathbf{b}_i^{(l)})_{\text{train}}\|\|(\mathbf{b}_j^{(l)})_{\text{test}}\|}.$$

Finally, classification can be performed by averaging over all similarity coefficients of the different patches per image according to

$$d_{ij} = \frac{1}{n_0}\sum_{l=1}^{n_0} d_{ij}^{(l)},$$

thus obtaining a single similarity coefficient per image, as in the previous sections.

12.5 Results

Data material

The raw data material consists of 50 images of each type of skin lesion, which were recorded with a conventional CCD camera at a size of 786×572 pixels and 256 shades of gray ($0 \ldots 255$, where 0 symbolizes white and 255 symbolizes black). In order to reduce the computational load, the fluorescence images were first centered and subsequently reduced by coarse graining to a size of 128×128 pixels [225].

After reducing the dimensions of the data by coarse graining, the images were transformed via several transfer functions which are standard

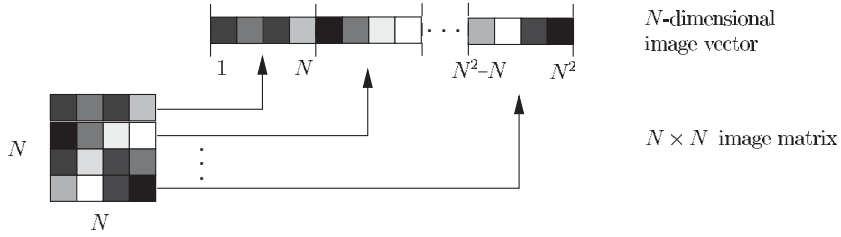


Figure 12.7
Basic procedure for converting the $N \times N$ image matrix into an N^2-dimensional image vector. The rows of the matrix are subsequently combined, resulting in an N^2-dimensional vector.

methods in image processing. The goal was to stress either contrast or smoothness of the images. Thus, we generated different ensembles of images for each cell type.

Presentation of the samples

For any analysis of the fluorescence images, a rearrangement of the image matrices proved necessary either to calculate the covariance matrix needed for PCA or to train the neural network when performing ICA. Therefore consider an image $\mathbf{X}$, the N^2 pixels of which are stored in an $N \times N$ matrix according to $\mathbf{X} = (x_{ij})_{0<i\leq N, 0<j\leq N}$. An image vector $\mathbf{x} = [x(1), \ldots, x(N^2)]$ can then be created by subsequently concatenating the rows of the matrix $\mathbf{X}$, thus obtaining an N^2-dimensional image vector, as illustrated in figure 12.7.

Reconstruction error-based classification using PCA

First the PCA eigenimages are computed for the different ensembles of skin lesions and the corresponding relative reconstruction error $\varepsilon_i^{\text{rel}}$ is determined, following equation (12.6). In figure 12.8, $\varepsilon_i^{\text{rel}}$ for 20 fluorescence images of each type of skin lesion based on a reconstruction by psoriasis eigenimages is shown. An image is then classified as belonging to class i when the minimum ε_i is below some fixed threshold Θ. For a reliable classification, this threshold Θ has to be adapted such that the ratio between the relative portion of the chosen class below and above the threshold Θ is maximum.

Applying this technique to all three types of skin lesions, three different sub-parts are defined by the corresponding thresholds, as also

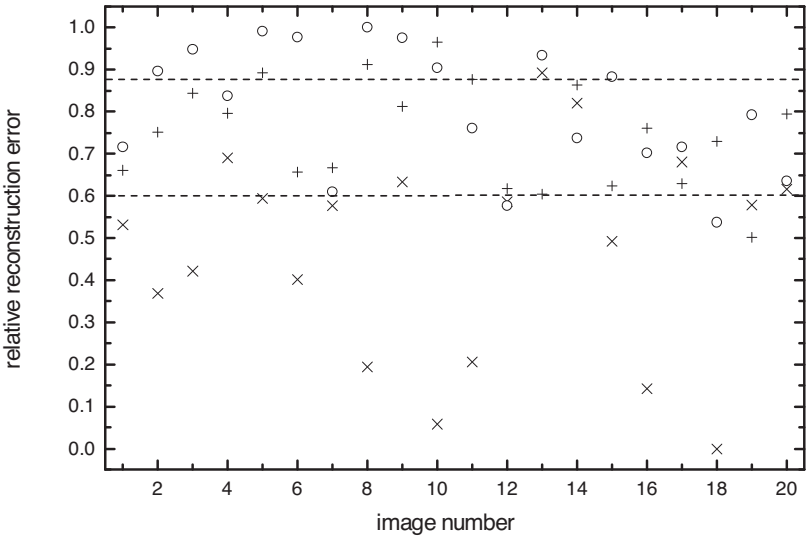


Figure 12.8
The relative reconstruction error for 20 images of each type of skin lesion, based on psoriasis eigenimages. The thresholds $\Theta_1 = 0.111$ and $\Theta_2 = 0.165$ divide the image into three sub-parts. Actinic keratosis is symbolized by ($\circ$), basal cell carcinoma by ($+$), and psoriasis by ($\times$), respectively.

depicted in figure 12.8, leading to $\Theta_1 = 0.111$ and $\Theta_2 = 0.165$ as plotted. Every image can now be assigned the class label of the respective sub-part.

However, this technique involves an error which has to be evaluated next: Assume that image i has binomial distribution (i. e. , it either belongs or does not belong to class $\mathcal{A}$). Furthermore, note that all values depicted in figure 12.8 are only some samples in the whole sample space. Due to the large number of available fluorescence images, it is possible to verify the assumption concerning the relative portion with regard to the whole ensemble several times. Therefore, for each experiment 10 samples are taken, and thus the reliability of the classification can be evaluated using hypothesis testing. As several simulations with the same training set can be accomplished, the overall relative portion of the relevant image ensemble is chosen close to the mean value of the corresponding relative portions of the experiments.

The results are summarized in the following table, based on the reconstruction using the different ensembles for the calculation of the principal axis (pa).

	pa: akt. kera.	pa: bcc	pa: psori.
akt. kera.	65%	63%	68%
bcc	55%	55%	?
psori.	70%	71%	82%

It must to be noted that classifying basal cell carcinoma on the base of psoriasis eigenimages leads to a rate which is smaller than 54% at an α-error of $\alpha = 0.05$. Thus a reliable classification is not possible, as 50% corresponds to guessing.

Classification by ICA

By analogy to the evaluation of the classification rate based on PCA eigenimages, here first the coefficient matrices $\mathbf{B}^{\text{train}}$ and $\mathbf{B}^{\text{test}}$ are computed, and subsequently the similarity coefficients d_{ij} are calculated. Then the thresholds Θ_1 and Θ_2 are determined as explained in the previous section, using 10 different ensembles of the same class for validation. Figure 12.9 shows the similarity coefficients d_{ij} for 20 test images of each skin lesion class, based on a set of psoriasis training images. The thresholds are fixed at $\Theta_1 = 0.99$ and $\Theta_2 = 0.999$. A profound analysis of the recognition rate is accomplished based on hypothesis testing as demonstrated above, resulting in 72% for actinic keratosis, 55% for basal cell carcinoma and 87% for psoriasis images, respectively, at an α-error of 5%.

Using various training ensembles, no significant differences in the results are obtained. This is in contrast to the outcomes using PCA, where the overall classification rates depended on the ensemble used for the calculation of the principal axes. Note however, that the test images which are of the same ensemble as the training images always show the largest similarity coefficients, as expected.

Although various image preprocessing steps were used, no significant classification enhancement could be obtained, as summarized in the table

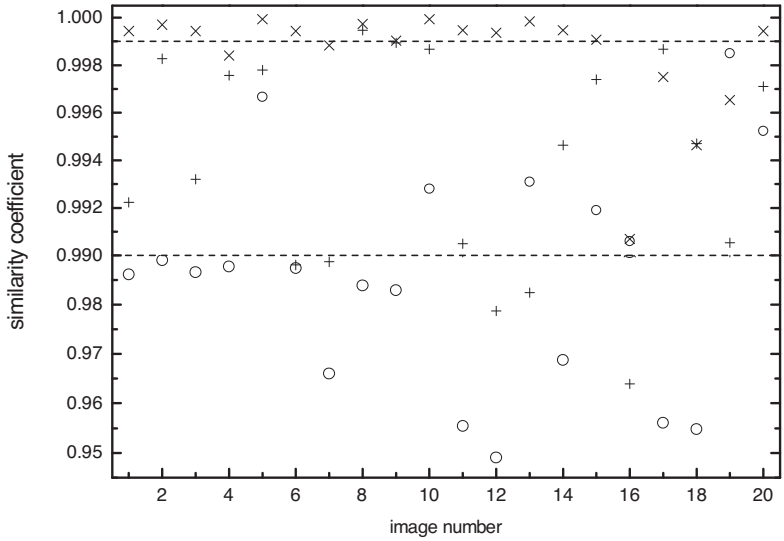


Figure 12.9
The similarity coefficients d_{ij} for 20 images of actinic keratosis (○), basal cell carcinoma (+), and psoriasis (×), based on a set of psoriasis training images. The thresholds $\Theta_1 = 0.99$ and $\Theta_2 = 0.999$ are evaluated, averaging over 10 different ensembles of the same class. For the sake of clarity, the Y-axis is differently scaled below $\Theta_1 = 0.99$.

below.

	orig.	$y_i = x_i^2$	$y_i = \sqrt{x_i}$	hist. equal.
akt. kera.	72%	72%	72%	70%
bcc.	55%	56%	56%	61%
psori.	87%	87%	86%	84%

Based on a training ensemble of psoriasis images which are preprocessed by the corresponding contrast manipulations and histogram equalization, respectively, only a slight performance increase for basal cell carcinoma can be noted at the cost of psoriasis images for histogram equalization.

Local ICA

By analogy to the evaluation demonstrated above, 10 different training ensembles of the same type of skin lesion are used for the simulations,

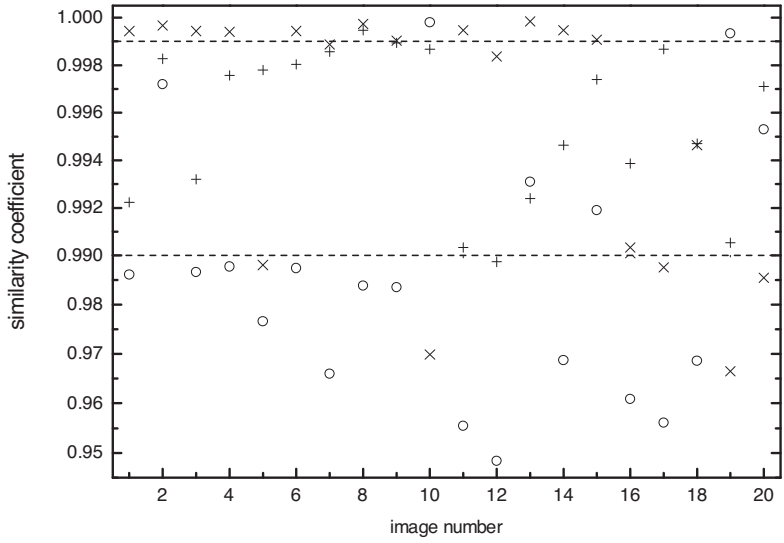


Figure 12.10
Based on the training ensemble of psoriasis images, the similarity coefficients d_{ij} are displayed for 20 fluorescence images of aktinic keratosis ($\circ$), basal cell carcinoma ($+$), and psoriasis ($\times$). For a better representation, the Y-axis is scaled differently for $d_{ij} < \Theta_1 = 0.99$.

thus allowing hypothesis testing for a reliable classification. Here the results using 16×16 patches of psoriasis images as the training ensemble are exemplified:

Reviewing figure 12.10, the thresholds Θ_1 and Θ_2 are calculated as in the PCA case, obtaining $\Theta_1 = 0.99$ and $\Theta_2 = 0.999$. As expected, the highest similarity coefficients are achieved for psoriasis images, resulting in a classification rate of more than 80% at an α-error of 5%. In the sub part $\Theta_1 \leq d_{ij} < \Theta_2$, basal cell carcinoma shows the highest relative portion, with an average value of $\overline{p} = 0.69$. As the hypothesis $H_0 : \pi \leq 0.69$ can be rejected, a basal cell carcinoma image can be correctly identified at a rate of more than 69%. Finally, for actinic keratosis an average classification rate of 70% is achieved.

Neither different training ensembles nor various preprocessing steps significantly influence the quality of the obtained results. However, due to the high classification rate of basal cell carcinoma, a relatively homo-

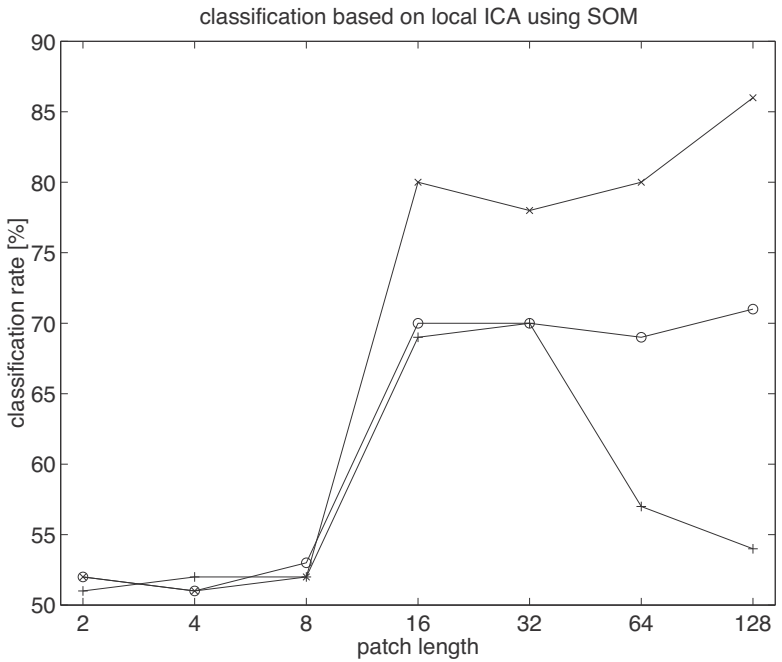


Figure 12.11
The classification rates for the three skin lesions strongly depends on the patch size. A patch size of at least 16×16 is needed to obtain a reliable classification rate. While for large patch sizes the classification rate for basal cell carcinoma (+) strongly decreases, for psoriasis ($\times$) a further enhancement can be noted, leading to the ICA results.

geneous identification of the different skin lesion ensembles is achieved.

As the clustering step is added to capture nonlinear trends in the multivariate data, various patch sizes have to be analyzed as generally nothing is known about the inherent structure of the fluorescence images. In figure 12.11 the classification rates for the three different skin lesion ensembles, based on a training set of psoriasis images, are displayed, depending on the patch sizes: At least a size of $16 \times 16 = 256$ pixels is needed to obtain a reasonable recognition rate. For patch sizes between 16×16 and 32×32 pixels per patch, on the average a relatively good recognition rate is achieved. Thereby the classification rate of psoriasis images (80%) still considerably exceeds the values obtained for basal cell carcinoma and actinic keratosis (70%). A further increase of the patch

Table 12.1
Comparison of the classification rates for actinic keratosis, basal cell carcinoma and psoriasis, depending on different algorithms. PCA corresponds to the results based on the original PCA-classification criterion. In the rows of *SOM-ICA* the results based on local ICA are summarized. The mean values and corresponding standard deviations are calculated by substituting $(p-1)\%$ when classification rates $< p\%$ were obtained. Thus the average classification rate per algorithm (row wise) and per skin lesion (column wise) can be calculated.

algorithm	act. kera.	bcc.	psor.	mean	std.
PCA	72%	60%	80%	71%	10%
ICA 1	72%	55%	87%	71%	16%
SOM-ICA	70%	70%	81%	74%	6%
mean	70%	59%	79%		
std.	2%	8%	7%		

size leads to a notable effect: while the recognition rate for psoriasis images is further ameliorated until the ICA results are obtained (87%), for basal cell carcinoma a strong decrease is noted. This may be due to its inherent structure, which cannot be represented adequately by (global) ICA.

12.6 Performance Comparison

The main goal of this chapter was to develop PCA- and ICA-based classification techniques which allow a reliable identification of psoriasis, basal cell carcinoma, and actinic keratosis. While an experienced physician still needs a biopsy for the distinction of basal cell carcinoma and actinic keratosis, in this study only fluorescence images of the relevant skin lesions were used. The results discussed in the following are shortly summarized in table 12.1.

PCA is a well-established method for classification and recognition tasks. Thereby a transformation of the base system allows a more efficient representation of important features in the multivariate data, see chapter 3. These structures (eigenimages) are ordered by their decreasing variances, so that most information can be transferred via the first principal components. For an exact examination of the possible enhancement by ICA, first PCA eigenimages of the skin lesion ensembles were computed, and classification was subsequently based on the reconstruction error. Obviously the principal axes of the image ensembles differ

sufficiently strongly that ensemble-specific features and structures are stressed by the projection, and thus a relatively high recognition rate is achieved. However, when reconstructing the images, one major reason may be that information is lost when reconstructing the images with a reduced number of eigenimages, which leads to a rather inexact representation of the fluorescence images by the corresponding reconstruction error.

ICA is an extension of PCA and was introduced to solve the *blind source separation* problem, see chapter 4. Thereby some source signals which were linearly mixed by an unknown process are reconstructed based on the statistical properties of the mixtures only. Here the Infomax algorithm of Bell and Sejnowski with the natural gradient extension was applied to the classification of the skin lesions, using the following images synthesis model. The recorded images were considered linear combinations of independent base images (by analogy to the PCA eigenimages). Therefore weight vectors were found in the directions of statistical dependencies in the ensemble of fluorescence images. Independent base images were subsequently calculated by projecting the fluorescence images onto the weights, and thus each fluorescence image was represented by the coefficients for the linear combinations of the independent base images. Classification was then based on the similarity of the coefficients for a set of training and test images.
However, no significant enhancement could be stated when using independent base images in comparison to the results obtained for PCA eigenimages. Although the recognition rate for psoriasis could be improved considerably (87%), basal cell carcinoma could hardly be identified (55%).
It seems that the ICA algorithm could not extract relevant features, so that the additional information coded in higher-order moments allowed higher classification rates. The main problem when dealing with ICA models is the indetermination of the number of sources. Unlike in PCA where the significance of the eigenimages is determined by their variances, a ranking of the ICA base images is not available. As the number of underlying independent base images is unknown, any set of base images—independent of their significance—could have been extracted. Especially when dealing with overcomplete systems, a subset of randomly found independent base images does not necessarily code

the most relevant information, and thus no further enhancement of the classification rate might be understandable.
And there is a second reason, while PCA provides a statistical tool based on an exact algebraic solution and independent of the probability distribution, the neural implementation of the Bell-Sejnowski ICA algorithm strongly depends on the assumed source density. Due to the computational load of the high-dimensional data, no adaptive techniques could be applied for an accurate modeling of the source densities. Although the distributions of the fluorescence images are super-Gaussian, and consequently the assumption of a Laplacian source distribution proves reasonable, slight deviations may involve errors which automatically lower the level of accuracy, and therefore the classification rate.

Using local ICA based on Kohonen's SOMs, the cluster size proved most essential for a reliable classification. Patches with sizes up to 8×8 pixels did not contain sufficient spatially structured information to allow a further increase of the classification rates. However, using larger patches (16×16 and 32×32 pixels), the obtained results clearly outperformed the classification rates achieved by PCA and ICA. It must to be noted that a strong simultaneous increase of the rates for all ensembles was obtained when evaluating patches of 16×16 pixels. Obviously, corresponding spatial structures in the fluorescence images allow high classification rates. However, a Fast Fourier Transformation (FFT) could not prove this hypothesis.
For larger patch sizes (64×64 and 128×128 pixels), a further increase of the classification rates for psoriasis and actinic keratosis could be noted. However, at the same time, the results for basal cell carcinoma deteriorated considerably until the values for (global) ICA were obtained. This might be due to the inherent structure of basal cell carcinoma, again only an assumption which could not be evidenced by an FFT analysis.

For answering the question of *when* to apply *which* classification method, two circumstances have to be taken into account: In the case where a high classification for a single skin lesion is needed (*Does image i belong to class $\mathcal{A}$?*), the applied method depends on the desired classification class: while for actinic keratosis PCA and ICA showed equally high classification rates (72%), basal cell carcinoma can be identified best by using local ICA based on SOM, resulting in an average classifi-

cation rate of 70%. For the identification of psoriasis, ICA proves most reliable, to 87%.
However, when dealing with an unknown set of fluorescence images and trying to classify the images (*What class does image i belong to?*), local ICA based on SOM showed the best overall recognition rate at a very low standard deviation ($74\% \pm 6\%$).

It is interesting to note that the dermatologists' identification problem between actinic keratosis and basal cell carcinoma is mirrored in the average recognition rate of the skin lesions. While psoriasis can be identified at a very high rate of 79%, actinic keratosis and basal cell carcinoma show much lower recognition rates (70% and 59%, respectively), both of which are still considerably higher than those achieved by an experienced physician.

Nevertheless, various improvements are necessary until a reliable assisting tool for the diagnosis based on the fluorescence images is available. Some interesting aspects include the following approaches.

- An independent analysis of the three channels of the corresponding RGB color image (see figure 12.2) may reveal additional information about the lesions. Furthermore, taking into account diameter, shape, or volume of the lesions, the evaluation of supplementary knowledge may contribute to an increased classification rate.
- All fluorescence images had to be reduced from an original size of 768×572 pixels to 128×128 pixels through coarse graining in order to reduce the computational load. Thereby, important information may have been lost, particularly when taking into account many-pixel correlations. With the next computer generations providing faster processors and significantly more memory, a further increase of the classification rates when analyzing entire fluorescence images is expected.
- Adaptive source density estimators may be applied for a more accurate extraction of the independent base images. Again, computational complexity prohibited the possible application of KBDE or the neural adaptation of the score function.
- A calibration of the fluorescence recordings with regard to size, location, and contrast of the lesion is most desirable, as similar experiments for face recognition resulted in much higher classification rates [17]. The

main differences lay only laid in the specified orientations and sizes of the faces, allowing an exact matching of characteristic features such as eyes, nose, and lips. Additionally, the tissue surrounding the lesions may differ strongly, depending on the affected region of the body. In order to cancel out its influence on the analysis, sophisticated preprocessing tools may be applied.

In Conclusion, various PCA- and ICA-based classification methods to identify actinic keratosis, basal cell carcinoma, and psoriasis have been evaluated in a comparative study. The results underline the importance of higher-order statistics in recognition tasks, as much information seems to be coded in higher correlations. The average classification rates considerably exceed the rates achieved by an experienced dermatologist, and therefore raise hope for a cheap and reliable diagnostic tool.

13 Microscopic Slice Image Processing and Automatic Labeling

A supervised interpretation of the initial data analysis model from section 4.1 leads to a classification problem: given a set of input-output samples, find a map that interpolates these samples, and, hopefully generalizes well to new input samples. Such a map thus serves as classifier if the output consists of discrete labels. Classification based on support vector machines [36, 37, 229] or neural networks [111] has prominent applications in biomedical data analysis. Here we review an application to biomedical image processing [260].

While many different tissues of the mammalian organism are capable of renewing themselves after damage, it was long believed that the nervous system is not able to regenerate at all. Nevertheless, the first data showing, that the generation of new nerve cells in the adult brain could happen were presented in the 1960s [7], showing new neurons in the brain of adult rats. In order to quantify neurogenesis in animals, newborn cells are labeled with specific markers such as BrdU; in brain sections these cells can later be analyzed and counted through the use of a confocal microscope. However, so far this counting process had been performed manually.

The goal of this chapter is to automate the task of counting labeled cells, which is currently done manually in many laboratories. Our novel algorithm contributes to a substantial speed-up in experimental settings. Furthermore, when comparing manual counts, differences in the counts are often noticed; hence, with an automated counting algorithm we hope to achieve an objective counter with known error bounds.

The chapter is organized as follows: section 13.1 presents the necessary neurobiological background of the analyzed section images. We then give an overview of the ZANE cell-counting algorithm in section 13.2. Section 13.3 presents an efficient algorithm for image stitching used in ZANE to allow for counting larger brain sections. The neural-network cell classifier is constructed in section 13.4, and is then used to analyze cell images in section 13.5. Comparisons with other methods are presented in section 13.6, and our main results are shown in section 13.7, comparing ZANE with manually counted section images. We finish with a discussion of further applications and future work in section 13.8.

13.1 Biological background

While many different tissues of the mammalian organism are capable of renewing themselves after damage, it was long believed that the nervous system is not able to regenerate at all. The first data showing that new nerve cells could be generated in the adult brain could happen were presented by Altman and Das in the 1960s. They published histological data showing new neurons in the brain of adult rats [7]. To identify those cells, they used the audioradiographic method by labeling newly emerged cells with 3H-thymidine. As there were no tools available for proving that these cells were adult nerve cells, their findings remained relatively unnoticed. In the early 1980s S. Goldman and F. Nottebohm found newly developed neurons in the dorsomedial striatum of adult songbirds [94]. But adult neurogenesis did not come into focus until the 1990s [16, 40, 146], when new techniques to analyze the newborn neurons were established. In particular, the introduction of *thymidine-analogon bromodeoxyuridine (BrdU)* as a nonradioactive marker for dividing cells gave rise to many new studies concerning adult neurogenesis. On the other hand, by establishing confocal microscopy it became possible to identify the characteristics of the newborn cells more clearly.

After that it could be shown that adult neurogenesis occurred in rodents (rats and mice), but also was found in primates and even in humans [40, 72, 75]. But neuroscientists also found that under physiological conditions adult neurogenesis is restricted to two brain regions. One is the lateral wall of the lateral ventricle, which is called the *subventricular zone*. The cells generated there migrate through the rostral migratory stream to the olfactory bulb, where they differentiate into mature neurons. The other "neurogenic" region in the adult brain is the granular cell layer of the dentate gyrus in the hippocampal formation of the temporal lobe. There, new cells are born in a thin zone right below the granular cell layer. During differentiation the cells integrate into the granular cell layer and become mature neurons with all functions of a granular cell [274]. "Neurogenesis" does not mean proliferation of cells alone; these newborn cells have to differentiate into mature nerve cells and be integrated into the existing network of neurons.

After these important findings much research was performed on the possible factors influencing adult neurogenesis. It was be shown that adult neurogenesis can be regulated by administering growth factors,

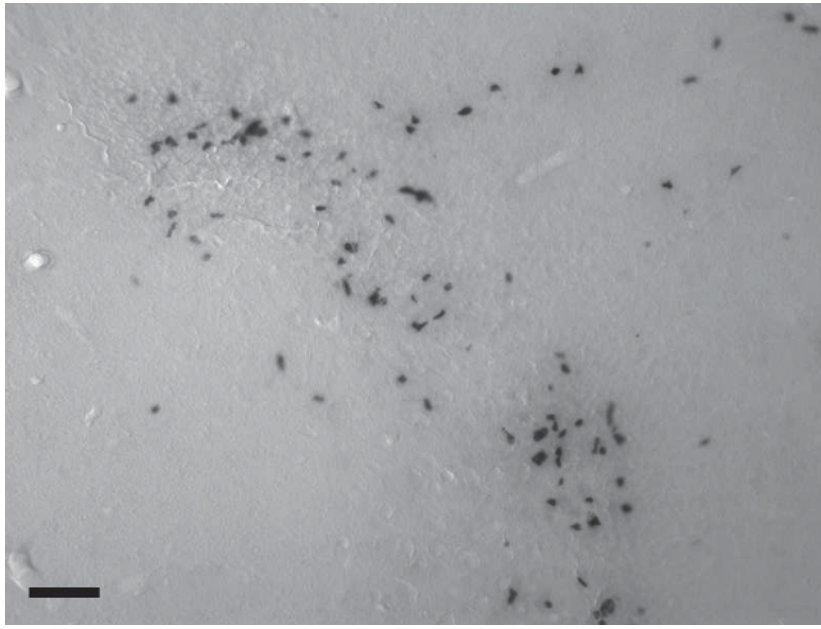


Figure 13.1
Brain section image of the dentate gyrus of a mouse. The black scale bar is $50\mu m$ long. The number of cells counted within the boundary (region of interest) following the dentate gyrus is 84; the number of cells in the whole image is 116.

neurotransmitters and several drugs. Further, pathological influences such as ischemia, seizures, or radiation affect the number of newly generated nerve cells. Also, general effects such as age, genetic modifications, and the amount of physical activity influence neurogenesis (review in [147]). Animals living in an enriched environment compared to standard laboratory conditions showed an increase of neurogenesis under distinct conditions [136].

13.2 Automated Counting

Figure 13.1 shows a brain section image of the dentate gyrus of a mouse in which the cells are to be counted. Classical approaches such as thresholding and erosion after image normalization could not successfully count the cells, mainly because cell clusters in the image cannot be properly

detected and counted using this method. In sections 13.6 and 13.7 we give a more detailed comparison with other methods.

ZANE

We propose the following adaptive counting algorithm, which we call *ZANE* (zell[1] analysis and evaluation). In the first step ZANE performs image stitching of the various microscope images and manual ROI selection to acquire the analysis image. The main counting step is based on a method proposed by Nattkemper et al. [183, 184] for evaluating fluorescence micrographs of lymphocytes invading human tissue; here, however, it is applied to light microscope images, and classifier preprocessing and training, as well as application, are different. The main idea is first to construct a function mapping an image patch to a *confidence value* in $[0, 1]$, indicating how probable it is that a cell lies in this patch or not – we call this function the *cell classifier*. In the second step this function is applied as a local filter onto the whole image; its application gives a probability distribution over the image with local maxima at cell positions. Nattkemper et al. call this distribution a *confidence map*. Maxima analysis of the confidence map reveals the number and the position of the cells. A flow chart of the ZANE algorithm is shown in figure 13.2.

Regions of interest

In practice, the cell counting is to be performed not within the whole image but only within a restricted region of the image called *region of interest (ROI)*. For example, in the presented experiments we want to count only cells from the dentate gyrus in the hippocampal formation of the temporal lobe. So far, the selection of the ROI is done manually, but we hope to automate this process in the future. However, precise criteria for the ROI detection seem to be difficult to extract – we assume a joint criterion taking both shape and image background texture into account is needed.

The impact of manual ROI selection is rather low in our experiments – the brain region of interest can be roughly identified manually by brightness and, especially, shape. Small deviations in this identification (given, for example, when comparing two experts who use implicit

1 German for “cell”.

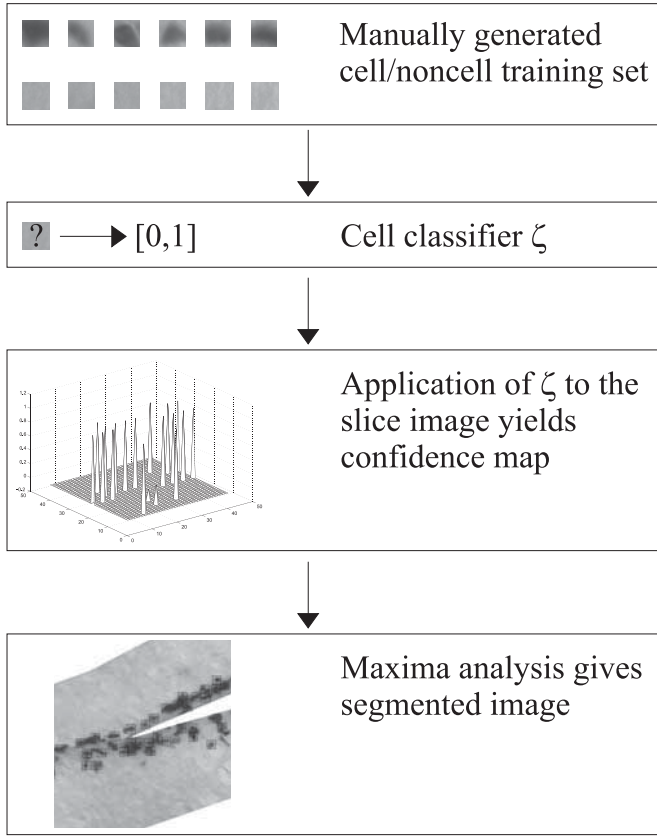


Figure 13.2
Flow chart of the main counting steps during ZANE image analysis.

ROI detection when focusing through the slice) do not matter because newborn cells sit in the subgranular layer, which is always included by protocol. Hence a certain cell type is analyzed which is present only in the given region. Other labeled cells in other regions are of another type, which is not to be counted. These cells typically lie far enough from the ROI region, which yields the low variability in ROI selection.

13.3 Image Stitching

Typically, brain sections are too large to be digitized as a single image by the camera at fixed resolution. In this case, multiple pictures are taken of the section with horizontally and vertically translated origin. In a first preprocessing step, these translated image patches have to be stitched together. This task is called by *image stitching*, and has been widely studied in the image-processing community (see e.g. [41, 96, 154] and references therein). Mathematical properties, together with the more general *geometric pattern-matching problem*, are nicely discussed in [96].

Measuring differences between masked images

With ZANE, we take a quite direct approach to the image-stitching problem. First, we need to define a measure for comparing two image patches $I_1, I_2 \in \mathbb{R}^{w \times h}$ of size $w \times h$. As mentioned in section 13.2, we consider patches with marked regions of interest. For the sake of simplicity, we assume that the given image patch has pixel entries only within a given interval C (say $[0, 255]$). Pixels not belonging to the ROI are to be set to a fixed value outside of C, say -1 (in the figures, we show those pixels as white). An image patch comparison measure can then simply be defined by

$$\bar{d}_p(I_1, I_2) := \left(\sum_{\substack{(x,y), I_1(x,y) \neq -1 \\ I_2(x,y) \neq -1}} |I_1(x,y) - I_2(x,y)|^p \right)^{1/p}, \qquad (13.1)$$

with $p > 0$. This is obviously equivalent to taking the p-norm of the vector of pixels lying in both ROIs. Typical choices of p are $p = 1, 2$. In order to be able to compare image patches of a varying sizes in the ROI, we further normalize this measure (for images with nonempty ROI) as follows:

$$d_p(I_1, I_2) := \frac{\bar{d}_p(I_1, I_2)}{|\{(x,y) | I_1(x,y) \neq -1 \wedge I_2(x,y) \neq -1\}|}. \qquad (13.2)$$

Then $d_p(I_1, I_2) \in C$, so overlapping image patches of different sizes with different ROI sizes can be compared. In practice, we set $d_p(I_1, I_2)$ to some large value if the overlap of the image patches and their ROIs is too small.

Since we have to consider only translations (the scale as well as the rotation of the image patches can be assumed to be the same due to the experimental setup), image-stitching of two patches I_1 and I_2 is performed by minimizing $d_p(I_1, \tau_{\boldsymbol{\delta}}(I_2))$, where $\tau_{\boldsymbol{\delta}}(I)(x,y) := I(x - \delta_1, y - \delta_2)$ denotes the translation of the image patch I by the vector $\boldsymbol{\delta} \in \mathbb{R}^2$ (possible additional zero-padding of the images assumed):

$$\boldsymbol{\delta}_0 := \operatorname{argmin}_{\boldsymbol{\delta}} d_p(I_1, \tau_{\boldsymbol{\delta}}(I_2)). \tag{13.3}$$

Various minimization algorithms can be employed to find or approximate $\boldsymbol{\delta}_0$. A simple solution is, for example, given by (discrete) *gradient descent* to determine local minima: the update rule is defined by

$$\boldsymbol{\delta}_{\text{new}} = \boldsymbol{\delta}_{\text{old}} - \eta \nabla d_p(I_1, \tau_{\boldsymbol{\delta}}(I_2))^p, \tag{13.4}$$

where η denotes a fixed or adaptive learning rate and ∇ is the discretized gradient of the cost function (taken to the power p to avoid roots) with respect to $\boldsymbol{\delta}$. The latter can easily be calculated as

$$p \sum_{\substack{(x,y), I_1(x,y) \neq -1 \\ \tau_{\boldsymbol{\delta}} I_2(x,y) \neq -1}} \operatorname{sgn}(\tau_{\boldsymbol{\delta}} I_2(x,y) - I_1(x,y)) |\tau_{\boldsymbol{\delta}} I_2(x,y) - I_1(x,y)|^{p-1} \cdot \begin{pmatrix} \tau_{(\delta_1+1,\delta_2)}(I_2)(x,y) - \tau_{\boldsymbol{\delta}}(I_2)(x,y) \\ \tau_{(\delta_1,\delta_2+1)}(I_2)(x,y) - \tau_{\boldsymbol{\delta}}(I_2)(x,y) \end{pmatrix}. \tag{13.5}$$

FFT-based speed-up

In practice, we use a previously selected feature from each image to restrict the search space spanned by $\boldsymbol{\delta}$ in equation (13.3), and then search for translations $\boldsymbol{\delta}$ within this restricted region. This is necessary because evaluation of the image distance equation (13.2) is computationally expensive; an exhaustive search for all possible $4hw$ translations is not feasible, and local update algorithms such as equation (13.4) need good starting values in order to avoid local minima.

In the following, we will describe an easy-to-calculate approximation of image similarity which allows us to estimate a fusion parameter $\hat{\boldsymbol{\delta}}_0$, from which we can start the above algorithm. The idea is to determine a $\boldsymbol{\delta}$ with maximal crosscorrelation between I_1 and $\tau_{\boldsymbol{\delta}}(I_2)$; in other words, we want to maximize the autocorrelation between the images. This can be interpreted as a second-order approximation of the distance equation (13.2), ignoring scaling and especially ROI parameters. In order to

account for the latter, we perform preprocessing in each image I by first subtracting the ROI-mean

$$\bar{I} := I - \frac{\sum_{(x,y),I(x,y)\neq -1} I(x,y)}{|\{(x,y)|I(x,y) \neq -1\}|}, \tag{13.6}$$

and then setting all non-ROI pixels to zero:

$$\tilde{I}(x,y) := \begin{cases} \bar{I}(x,y) & I(x,y) \neq -1 \\ 0 & I(x,y) = -1 \end{cases} . \tag{13.7}$$

Thus $\tilde{I}$ also has mean zero; hence, in the *masked autocovariance*

$$R_{\boldsymbol{\delta}}(I_1, I_2) := \sum_{x,y} \tilde{I}_1(x,y)\tilde{I}_2(x-\delta_1, y-\delta_2) \tag{13.8}$$

we do not have to subtract the means, and non-ROI regions in any of the two patches do not contribute to the sum. The desired initial starting parameter $\hat{\boldsymbol{\delta}}_0$ is now simply estimated by maximizing the masked correlation between the translated patches:

$$\hat{\boldsymbol{\delta}}_0 := \text{argmax}_{\boldsymbol{\delta}}\, R_{\boldsymbol{\delta}}(I_1, I_2) \tag{13.9}$$

We introduced this additional measure because although calculating equation (13.8) for all $\boldsymbol{\delta}$ is just as expensive as (13.2), we can now use the multiplicative structure to derive an immensely faster calculation of equation (13.8). For this we use the well-known trick [29] of rewriting the autocorrelation as a 2-D convolution:

$$R_{\boldsymbol{\delta}}(I_1, I_2) = \sum_{x,y} \tilde{I}_1(x,y)\tilde{I}_2'(\delta_1 - x, \delta_2 - y) = \tilde{I}_1 * \tilde{I}_2', \tag{13.10}$$

where $I'(x,y) := I(-x,-y)$ (with additional zero-padding). But the convolution reduces to a simple multiplication in the Fourier spaces, so, using the 2-D Fourier transformation [48] $\mathcal{F}$, we get:

$$R_{\boldsymbol{\delta}}(I_1, I_2) = \tilde{I}_1 * \tilde{I}_2' = \mathcal{F}^{-1}\left(\mathcal{F}(\tilde{I}_1)\mathcal{F}(\tilde{I}_2')\right) \tag{13.11}$$

The discrete Fourier transforms and the inverse are calculated using the 2-D FFT algorithm, which costs $O(wh \log wh)$ operations. The multiplication in the frequency domain itself needs only $O(wh)$ operations, so the total cost of using equation (13.11) is $O(wh \log wh)$, which, especially for large images, is much cheaper than the $O(w^2h^2)$ operations needed for directly convolving the images in equation (13.8).

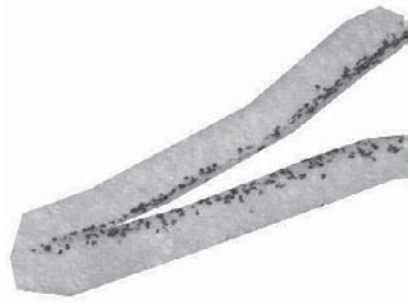
Due to the limited precision of the FFT-based intermediate results, equation (13.11) does contain small numerical errors. Moreover, some caution concerning the above theoretical gain seems to be appropriate: FFT routines generally have a larger overhead and use more memory than the direct convolution. Also, the convolution is real-valued, whereas the multiplication in the frequency domain and the FFTs need complex operations, so the speed-up factor should be decreased by a factor of 4 to 6.

Example

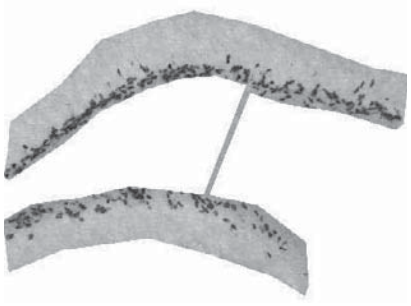
Figure 13.3 shows an experiment, in which two images of sizes $w = 1600$, $h = 1200$ with marked ROIs (see 13.3(a) and 13.3(b)), are stitched together using the above algorithm. The 2-D autocorrelation $R_{\boldsymbol{\delta}}$ is calculated, using FFTs, by equation (13.11). The result is a complex-valued matrix; however, the sum over all complex components is about 10^{15} lower than the sum over the real ones. Hence it comes from numerical errors and is discarded. The autocorrelation is displayed in figure 13.3(c). Clearly a dominant maximum at $\hat{\boldsymbol{\delta}}_0 = (1159, -237)$ is present. The more precise distance measure from (13.2) is then applied, using an exhaustive search within the square $\hat{\boldsymbol{\delta}} + (\pm 5, \pm 5)$, which yields the final $\boldsymbol{\delta}_0 = (1159, -238)$. Obviously the FFT had already localization yielded a nice match, though in some situations (e.g., given complicated masking borders) fine-tuning by equation (13.2) is more important. In our MATLAB realization the exhaustive search within the small square took about twice as long as the FFT-based calculation of the full-size autocorrelation (though the latter consumed considerably more memory), so the speed-up factor was around 38000.

13.4 Cell Classifier

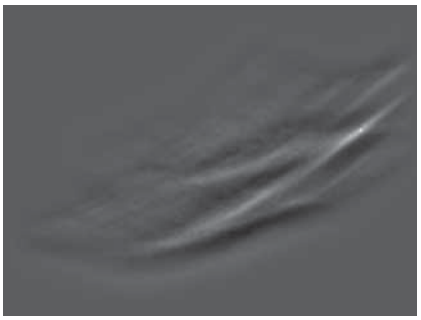
In this section, we will explain how to generate a cell classifier that is a function mapping image patches to cell confidence values. For this a sample set of cells and non cells is generated; then an artificial neural network is trained using this sample set.



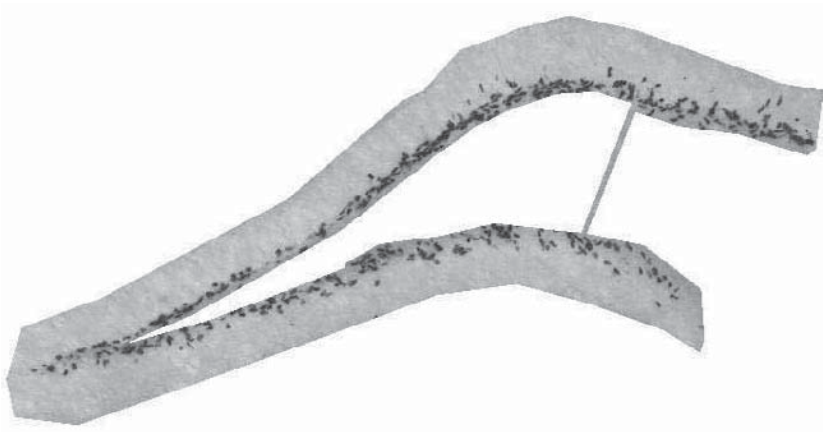
(a) Left image patch



(b) Right image patch



(c) Rescaled and translated autocorrelation calculated using 2-D FFT



(d) Stitched image

Figure 13.3
Image stitching. The two image patches (a) and (b) with marked ROIs are stitched together using translation of the two patches against each other. (c) shows the calculated autocorrelation (white = higher values), (d) is the final result.

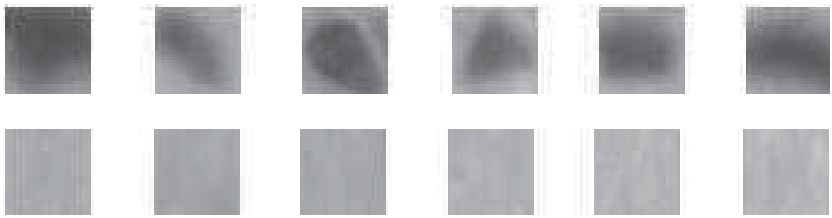


Figure 13.4
Part of the training set. The first row consists of 20x20-pixel image patches that contain cells; the lower row consists of non cell image patches.

Sample set

After fixing the patch size – in the following we will use 20×20 pixel gray-level image patches – a training set of cell and non cell patches has to be manually generated by the expert. The image set is enlarged by adding rotated, flipped copies of the patches. The image patches are then to be classified by a neural network. Figure 13.4 shows some cell and non cell patches.

Interpreting each 20×20 image patch as a 400-dimensional vector, we get a set of L training vectors

$$T := \{(\mathbf{x}_1, t_1), \ldots, (\mathbf{x}_L, t_L)\} \tag{13.12}$$

with $\mathbf{x}_i \in \mathbb{R}^n$ – here $n = 20^2$ – representing the image patch and $t_i \in \{0, 1\}$ either 0 or 1, depending on whether x_i is a non cell or a cell. This can easily be generalized to classify different types of cells. The goal is to find a mapping that correctly classifies this data set that is a mapping a $\zeta : \mathbb{R}^n \to [0, 1]$ with $\zeta(\mathbf{x}_i) = t_i$ for $i = 1, \ldots, L$. We call such a mapping *cell classifier*. Of course ζ is not uniquely defined by the above property, so some regularization has to be introduced. Any interpolation technique, such as a Fourier or a Taylor approximation, can be used to find ζ; we will use single-layer and multilayer perceptrons, as explained in the following sections.

Preprocessing

Before we apply neural network learning, we preprocess the data as follows. Denote $\mathbf{x}$ as the underlying n-dimensional random vector from which the samples $\mathbf{x}$ have been drawn.

In a first normalization step, we scale and translate $\mathbf{x}$ such that the two density maxima of $\mathbf{x}$ – corresponding to the gray background and the dark cell color – are mapped onto two fixed values. Subtracting the mean $\mathbf{x} \mapsto \mathbf{x} - \mathbf{E}(\mathbf{x})$ then ensures that the data set is centered. In order to reduce dimension as well as to decorrelate the data in a first separation step, we apply *principal component analysis (PCA)*, i.e. linearly transform the random vector $\mathbf{x}$ in order to decorrelate it and also to reduce its dimension by projecting along the largest eigenvectors (principal axes) of the correlation matrix of $\mathbf{x}$, see chapter 3.

When analyzing the eigenvalue structure of the training set covariance, we note that by taking only the first five eigenvalues, projection along those first five principal axes still retains 95% of the data. Thus, the 400-dimensional data space is reduced to a whitened five-dimensional data set. A visualization of the 120-sample data set is given in figure 13.5, after projection to three dimensions. One can easily see the cell and non cell components can be linearly separated – thus using a perceptron (see later) can indeed already learn the cell classifier. Furthermore, a k-means clustering algorithm has been applied with $k = 2$ in order to find the two data clusters. They correspond directly to the cell/non cell components (see figure 13.5).

The above result also indicates that unsupervised learning algorithms can produce a meaningful approximation of a cell classifier. We will confirm this by successful application of *independent component analysis (ICA)*. In ICA, given an (observed) random vector, the goal is to find its statistically independent components. This can be used to solve the *blind source separation (BSS)* problem, which is, given only the mixtures of some underlying independent sources, to separate the mixed signals and thus recovering the original sources. In contrast to correlation-based transformations such as PCA, ICA renders the output signals as statistically independent as possible by evaluating higher-order statistics. The idea of ICA was first expressed by Hérault and Jutten [112] while the term ICA was later coined of Comon in [59]. In the calculations we used the well-known FastICA algorithm [123] by Hyvärinen and Oja, which separates the signals by using negentropy, and therefore non-Gaussianity, as a measure of the signal separation quality.

Figure 13.6 is a plot of the linearly separated signals together with the cell/non cell function for comparison. The fifth component is highly correlated ($cc = 0.9$) with the desired output function, so instead of

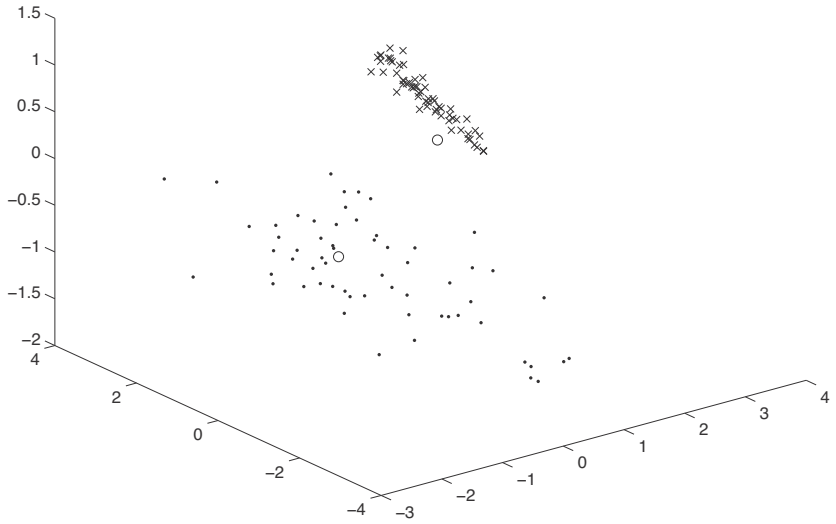


Figure 13.5
Data set with 120 samples after three-dimensional PCA projection (91% of the data was retained). The dots mark the 60 samples representing cells; the crosses mark the 60 non cell data points. The two circles indicate clusters of a k-means application with a search for two clusters. Obviously, k-means nicely differentiates between the cell and the non cell components.

using a linear perceptron, projection along the direction of the fifth IC, together with a sign function, can be used in order to separate cells and non cells. This is quite interesting because in comparison to the supervised perceptron learning approach above, the ICA is completely unsupervised. Only later, when comparing the ICs, do we use the prior information on cells and non cells in order to differentiate between the source components. The fact that the data set contains a cell/non cell independent component was already indicated by the k-means cluster analysis from figure 13.5, where we saw that the data set clusters into the cell and non cell components. If we perform PCA to decorrelate the data, we can also identify a cell/non cell component; however, its crosscorrelation with the correct classification function is 5% lower than the ICA result. This confirms that higher-order correlations improve

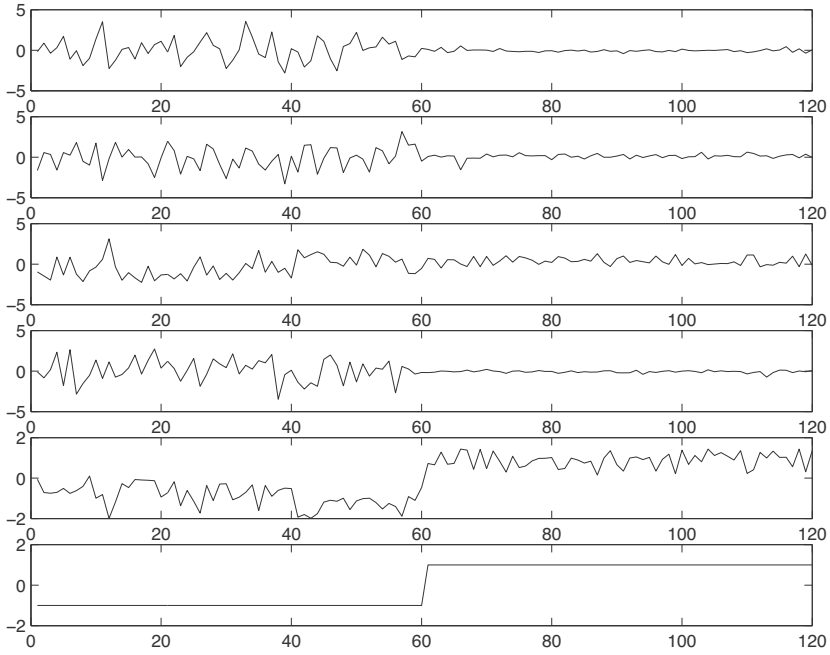


Figure 13.6
The five independent components of the data set calculated using FastICA with deflation and pow3-nonlinearity after whitening and PCA dimension reduction to five dimensions. Below the five components, the cell/non cell functions (-1 or 1) of the samples are plotted for comparison. The crosscorrelations of each signal with these functions are -0.055, 0.11, 0.37, -0.081, and 0.90. Visual comparison already confirms good correspondence of the fifth IC with the cell/non cell function.

data separation, albeit by a rather small factor in this case.

Neural network learning

In the previous text, we saw that even unsupervised methods were sufficient to detect an acceptable cell classifier in the given BrdU-labeled cell experiment. However, performance is somewhat weak, because the knowledge of cell/non cell labels was exploited only afterward. Also, if more complicated structures are to be analyzed, nonlinear classifiers turn out to be preferable. In order to allow for training as well as flexibility regarding the strength of possible nonlinearities, we will use neural networks to learn such a cell classifier. Depending on the applications

presented later, either simple linear, single-layered, or more complicated network structures have to be used.

Supervised learning algorithms try to approximate a given function $\mathbf{f} : \mathbb{R}^n \to A \subset \mathbb{R}^m$ by using a number of given sample-observation pairs $(\mathbf{x}_\lambda, f(\mathbf{x}_\lambda)) \in \mathbb{R}^n \times A$. If A is finite, we speak of a classification problem. Typical examples of supervised learning algorithms are polynomial or spline interpolation and artificial neural network (ANN) learning. In many practical situations, ANNs have the advantage of higher generalization capability than other approximation algorithms, especially when only a few samples are available, see chapter 6.

We will restrict ourselves to feed forward layered neural networks. Furthermore, we found that in comparison to multi-layered perceptrons (MLP), simple single-layered neural networks (perceptrons) already sufficed to learn the data set well – and they have the advantage of easier rule extraction and interpretation.

In order to learn the cell classifier, we use a single-unit perceptron with a linear activation function to get a measure for the certainty of cell/non cell classification. Application of the delta learning rule to the five-dimensional data set from above gives excellent performance after four epochs of batch learning. The final performance error (variance of perceptron estimation error of the training set) after 55 epochs was 0.0038, which confirms the good performance as well as the linearity of the classification problem. This was confirmed by training a two-layered network with five hidden neurons in order to test for nonlinearities in the data set. Indeed, the MLP could not significantly enhance the result: after only 10 epochs, the classification error was already very small (10^{-4}), and it could finally be diminished to $3 \cdot 10^{-19}$; the latter, however, did not enhance classification noticeably.

Directional neural networks

The above approach works well in the case of cell types that are more or less circular, where mainly texture identification is important. However, if we have to deal with more complicated classification problems, we lose cell classification specificity – this follows from the fact that we do not know the orientation of the cells in both the training data set and the test image. Above, we accounted for this by adding rotated and mirrored versions of the cells to the data set; however, this leads to an approximate radial symmetry in our classifier.

Depending on the image patch type – for example, patches containing higher-order structures such as axon-dendrite networks of the neurons – a more elaborate network structure has to be found in order to avoid the symmetry. In the following, we introduce a preprocessing method that allows us to orient the image patches in a default way, thus enabling the shape classification of arbitrarily oriented image patches. We will denote trained neural classifiers employing this preprocessing as *directional neural networks*.

The idea, similar to PCA, is to orient an image I along its principal axis, where the image pixels themselves are interpreted as samples of a two-dimensional random vector $\mathbf{x}_I$. Hence I is a two-dimensional histogram of $\mathbf{x}_I$, and the density of $\mathbf{x}_I$ at the pixel (x, y) can be estimated by $p_{\mathbf{x}_I}(x, y) \approx I(x, y)/T$ with $T := \sum_{x,y} I(x, y)$. This yields estimates for the mean

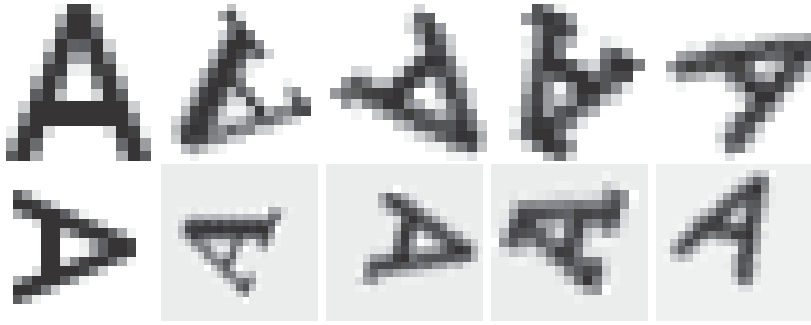
$$\mathbf{E}(\mathbf{x}_I) \approx \boldsymbol{\mu}_I := \sum_{x,y} \begin{pmatrix} x \\ y \end{pmatrix} \frac{I(x, y)}{T} \tag{13.13}$$

and the covariance

$$\mathrm{Cov}(\mathbf{x}_I) \approx \mathbf{C}_I := \left(\sum_{x,y} \begin{pmatrix} x^2 & xy \\ xy & y^2 \end{pmatrix} \frac{I(x, y)}{T} \right) - \boldsymbol{\mu}_I \boldsymbol{\mu}_I^\top . \tag{13.14}$$

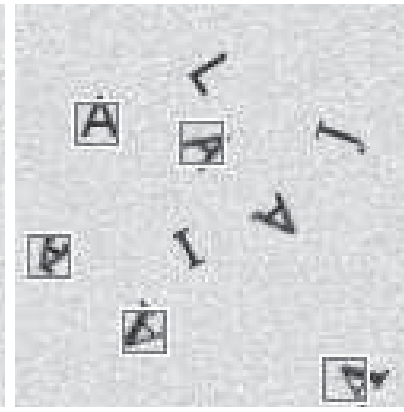
For a given image I, let $\rho(I)$ be the rotated image I of the same size such that the eigenvector of $\mathbf{C}_{\rho(I)}$ corresponding to the largest eigenvalue is parallel to the x-axis $(1, 0)$. Applying the neural network training from section 13.4 to the "normalized" training set $(\rho(\mathbf{x}_\lambda), f(\mathbf{x}_\lambda))$ (after adding possible reflections of the patches at the x-axis) yields the desired directional neural network, which now is directionally selective. Any possibly rotated input image patch is applied to the composed classifier $f \circ \rho$.

Figure 13.7(a) is an example of the application of the eigenvector-based rotation. In the top row, five 15×15 input patches displaying the character A in various rotations and typed in various fonts are displayed. The corresponding normalized images are given in the second row; clearly all characters, except for the last one, were oriented such that their main axis is parallel to the x-axis. Apparently due to aliasing effects in the original image, in the last character the horizontal bar of the A contributed most to the covariance, and hence could not be rotated correctly.



(a) directional normalization

(b) training data set



(c) classification result

Figure 13.7
Directional neural networks. (a) shows five rotated As together with their normalized image patches (below).

Figures 13.7(b) and 13.7(c) briefly demonstrate the successful application of the ZANE classification scheme based on directional neural networks to character detection. A character classifier ζ is trained using a directional network based on a generalized regression neural network [279], a specialized radial basis function network allowing for more complicated nonlinear function approximations. The classifier is trained using the seven noisy "A" training samples from 13.7(b) together with 70 randomly selected non-A patches. After directional normalization, PCA

is performed to reduce the dimension to 10, and a generalized regression network is trained with five hidden neurons. The classifier is applied (see section 13.5 for algorithmic details) to identify rotated characters in figure 13.7(c). Five characters were correctly identified, a single one was not, and the algorithm produced no misclassifications. This result is good, considering the small training set.

13.5 Confidence Map

Generation

The cell classifier has to be trained only once. Given such a cell classifier, section pictures can now be analyzed as follows.

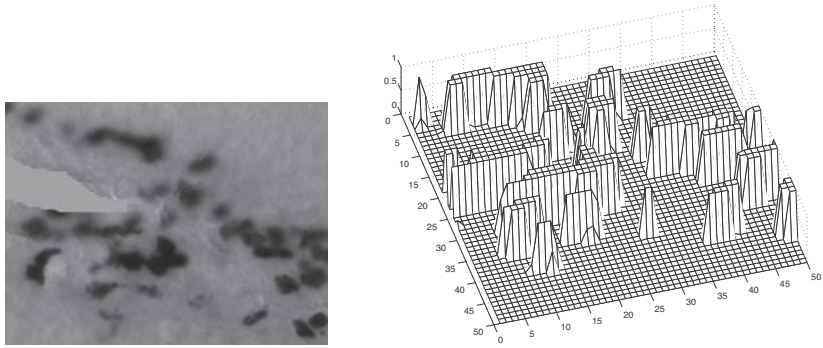
A pixelwise scan of the image yields an image patch with center location at the scan point; the cell classifier is then applied onto this image patch in order to give a probability determining whether a cell is located at the given position or not. This yields a probability distribution over the whole image which is called a *confidence map*. Each point of the confidence map is a value in $[0, 1]$ stating how probable it is that a cell is depicted at the specified location.

In practice, a pixelwise scan can be too expensive in terms of calculation time, so a grid value γ can be introduced, and the picture is scanned only every γ-th pixel. This yields a rasterization of the original confidence map, which for small γ can still be fine enough to detect cells. Figure 13.8 shows the rasterized confidence map of a section part. The maxima of the confidence map correspond to the cell locations; small but non zero values in the confidence map typically depict misclassifications that can be avoided by thresholding.

Depending on the type of cell classifier, a method to increase performance similar to that in section 13.3 can be applied. In the simplest case, the classifier is a linear separator (for example, learned by a perceptron or one of the above unsupervised techniques). Then

$$\zeta : \mathbb{R}^{n\times n} \to [0,1], I_1 \mapsto \sigma\left(\sum_{x,y} I_1(x,y)W(x,y)\right), \tag{13.15}$$

where I_1 is the $n \times n$ image patch to be tested, $W \in \mathbb{R}^{n\times n}$ the trained weight matrix, and $\sigma : \mathbb{R} \to \mathbb{R}$ is an increasing nonlinearity (already



(a) source image (b) confidence map

Figure 13.8
The plot shows the confidence map (b) generated with grid value $\gamma = 5$ from the source image (a).

containing a possible bias). The inherent decision rule $\zeta(I_1) > 0$ then translates to the inner product of I_1 and W being greater than $\sigma^{-1}(0)$.

By definition, the confidence map $\kappa \in \mathbb{R}^{h \times w}$ of an image I of size $h \times w$ is given by

$$\begin{aligned} \kappa(u,v) &= \zeta(I(u \dots u+n, v \dots v+n)) && (13.16)\\ &= \sigma\left(\sum_{x,y} I(u+x, v+y) W(x,y)\right) && (13.17)\\ &= \sigma\left(\sum_{x,y} I(x,y) W'(u-x, v-y)\right) && (13.18)\\ &= \sigma\left(I * W'(u,v)\right) && (13.19) \end{aligned}$$

after sufficient zero-padding of W and again with W' denoting the reflected image. Application of the filter W' can now be easily speeded up by using multiplication in the Fourier space (see section 13.3):

$$\kappa = \sigma\left(\mathcal{F}^{-1}\left(\mathcal{F}(I)\mathcal{F}(W')\right)\right). \qquad (13.20)$$

Here σ is implicitly applied for each pixel; in addition $\mathcal{F}(W')$ can be calculated beforehand, so that only two 2-D FFTs have to be calculated. Similar to the case of image-stitching, this results in a consid-

erable performance increase, especially for larger images. Furthermore, this approach can be readily extended to the case of MLPs by replacing each neuronal activity calculation with the FFT-perceptron weight calculation [29].

Evaluation

After the confidence map has been generated, it can be evaluated by simple maxima analysis. However, as seen in figure 13.8, due to noise and non cell objects in the images, maxima do not always correspond to cell positions, so thresholding in the confidence map has to be applied first. Values of 0.5 to 0.8 yield good results in experiments, and if a neural network-based approach is taken, the threshold values are already implicitly given by the bias value at the output neuron. Furthermore, the cell classifier yields high values corresponding to a single cell when applied to image patches with large overlap. Therefore, after a maximum has been detected, adjacent points in the confidence map are also set to zero within a given radius (15 to 18 were good values for 20×20 image patches). Iterative application of this algorithm then gives the final cell positions, and hence the image segmentation and the cell count.

13.6 Relation to Other Methods

Although feature *counting* per se has not been studied very intensely (see, e.g., [183, 184]), it can of course be interpreted as a secondary problem in the larger field of *image segmentation*. Its goal is to decompose one or multiple images into their "natural" parts, those being specified by similarities such as color, shape, texture, or some higher-level semantic meaning. Our problem of cell counting can then be solved by counting those image segments that represent cells – in the case of a perfect segmentation, these should consist of all components except for the large background component (which itself could contain multiple segments).

Nowadays common algorithms for image segmentation, apart from neural network based approaches like the above, include segmentation using morphological operations or linear decomposition algorithms such as *non negative matrix factorization* [150]. A very common technique belonging to the first category is the so-called *watershed transform*. Its

intuitive idea can be visualized in geographical terms: in a landscape flooded by water, watersheds divide the domains of attraction of rain falling over the region. If image properties are measured by a single variable specified at each pixel, the watershed algorithm finds connected components belonging to separate local minima. This algorithm was first proposed by Digabel and Lantuéjoul [68] and later improved by Beucher and Lantuéjoul [32]. A nice up-to-date overview can, for example, be found in [222]. More elaborate frameworks and extensions have been proposed, such as the combination of watershedding with region merging in a hierarchical structure [107] or graph-theoretic segmentation known as the n-cut method [231].

An application of the watershed transform to cell image segmentation and recognition is presented in [191]; however, the cell images are acquired from bone marrow smear, which substantially reduces background noise. Furthermore, no focusing problems are involved, so all cells are of similar shape, texture, and especially size.

A more direct method, which has recently been suggested for slice image segmentation in [30], uses thresholding for noise removal, and afterward simply counts the number of connected components. Clusters of appropriate pixel size are then interpreted as a single cell and counted.

We have also considered the use of these more classical approaches to cell counting, but apart from some computational issues (and choice of the various involved thresholding parameters), the main disadvantage in contrast to the proposed algorithm lies in the fact that the above algorithms do not take the actual cell shapes into account. Essentially they are indifferent to shape, and count any object of appropriate color and pixel size. A related problem can be seen in figure 13.9. There compare ZANE with the two most common methods by applying the watershed transform both to the confidence map and to a distance map (containing distance values of pixels to cell boundaries given by a threshold). In both cases the result strongly depends on image preprocessing and thresholding to avoid too many local minima. Apart from some misclassified regions due to thresholding problems, watershedding of the confidence map partially separates cell clusters, but also introduces additional segments at intersections. If water shedding was applied to the distance map, cell clusters could not be separated at all. We believe that this is due to the somewhat problematic conditions of image acquisition using a confocal light microscope, which cannot give cell boundaries as clearly

as other experimental settings such as [191]. ZANE resolves this problem by using the previously learned shapes; the better counting performance can be observed in figure 13.9(c). Finally, the advantage of ZANE lies in the fact that after training, image analysis is basically performed by a combination of filtering (to generate the confidence map) and maxima analysis, which is enough for counting but not for segmentation. Hence, ZANE can be expected to outperform the above methods.

13.7 Results

Brain section image acquisition

The analyzed brain sections were taken from the dentate gyrus of mice. BrdU given systemically is integrated into the dividing DNA instead of thymidine during the S-phase of the mitosis [69]. Using a specific antibody against BrdU, labeled cells can be detected by an immunohistochemical staining procedure. The nuclei of labeled cells on $40\mu m$-thick brain sections are dense dark brown or black. To determine the number of BrdU-positive cells in the granular cell layer of the dentate gyrus, they were counted on a light microscope (Olympus IX 70; Hamburg, Germany) with a $20\times$ objective. Digital images with a resolution of 1600×1200 pixels were taken by a color video camera adapted to the analySIS software system (Soft Imaging System, Münster, Germany).

Manual counting

Before analyzing ZANE counting performance, it should be noted that the number of labeled cells counted by an expert varies considerably between observers. This is not due only to inaccuracies of the experts, but mainly depends on the interpretations of whether a labeled cell is present at a location or not (i.e., on the personal cell classifiers of the experts). In order to quantitatively compare the proposed algorithm with manual counts, we therefore need an estimate of the expert cell count deviation. Such a deviation has been mentioned in the literature [30], but finding actual values is difficult.

From our knowledge, a typical value of the inter observer variability in expert cell counts ranges from 10% to 15%. In order to confirm this, we compare the cell counts of two experts given 18 different animals with five to seven slices each. Each slice is counted using the microscope, so

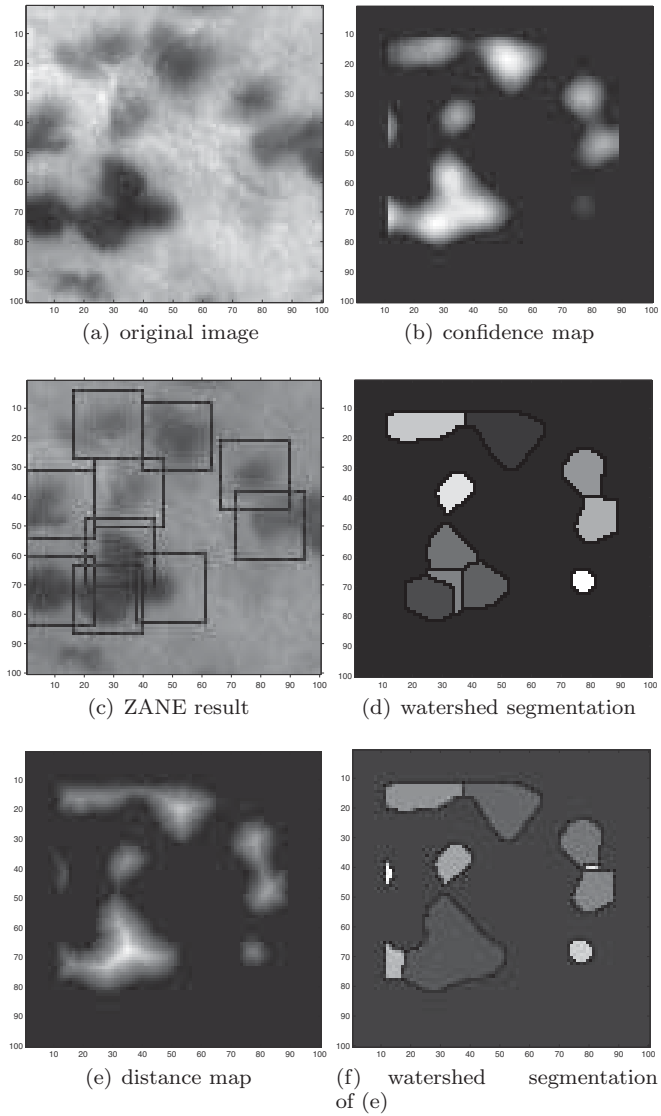


Figure 13.9
Section image segmentation using the watershed transform. Using the cell-classifier ζ, an example 100×100 section image (a) is transformed to give the confidence map (b). For comparison the ZANE counting/cell localization result is presented in (c). Direct application of the watershed transform (after thresholding) yields (d); distance map generation of the thresholded source image (e) gives a different watershed result (f).

additional three-dimensional information was available to the experts for confirmation of the cell/non cell status. Figure 13.11(a) shows the resulting numbers; clearly the experts differ considerably, with the first expert typically counting more than the second one. The standard deviation $\sigma_i := \sqrt{\operatorname{var} X_i}$ in each counting result X_i ranges from 14.1 to 171.8. We normalize the deviation by dividing by the estimated mean $\mu_i := E(X_i)$ of each experiment, and get a total relative deviation of $E(\sigma_i/\mu_i) = 0.1494$. Hence the mean deviation between the two experts lies at 15%.

Finally, even though the problem of differences in the number of counted cells is well known, a possible solution such as deliberate slight overcounting in order not to miss important features is not feasible. This is because over- and undercounting are equally bad. The former results in a too high variability, whereas the latter does not allow for sufficient differentiation.

ZANE counting

When training the cell classifier in practice, we use perceptron learning after preprocessing with both and ICA in order to increase the performance of the learning algorithm with linearly separated data. Using prior knowledge about the sizes of cells and the zoom factor of the images, the patch size is chosen to be 20×20. Optimal values for threshold and cut out radii have been obtained using optimization on the training images. A thresholding of 0.8 is applied in the confidence map, and the cut out radius for cell detection in the confidence map is chosen to be 18 pixels. In figure 13.1, an automatically segmented picture is shown. Figure 13.10 presents the segmentation of the stitched image from figure 13.3. ZANE counted 281 cells, versus 267 counted cells by an expert using focusing in the full three-dimensional slice; this confirms the good performance of the counting algorithm.

A more detailed analysis of the ZANE cell counting algorithm is shown in figure 13.11. In (c), five stitched brain section images of a single mouse are counted using ZANE, manual counting, and two other segmentation algorithms, based on clustering [30] and the watershed transform [68] respectively. Note that manual counting can be performed either by using the digital image only or by directly using the microscope (this is usually done in experiments). Then the counting person can change the focus plane, and hence detect and count cells that lie below or

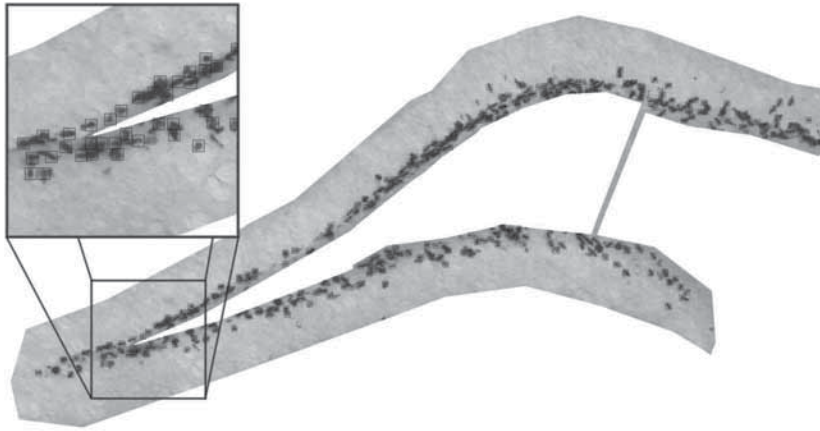


Figure 13.10
Automatically segmented image from figure 13.3. Here, the number of counted cells (marked by black boxes) is 281. The expert counted 267 cells (focusing through the section).

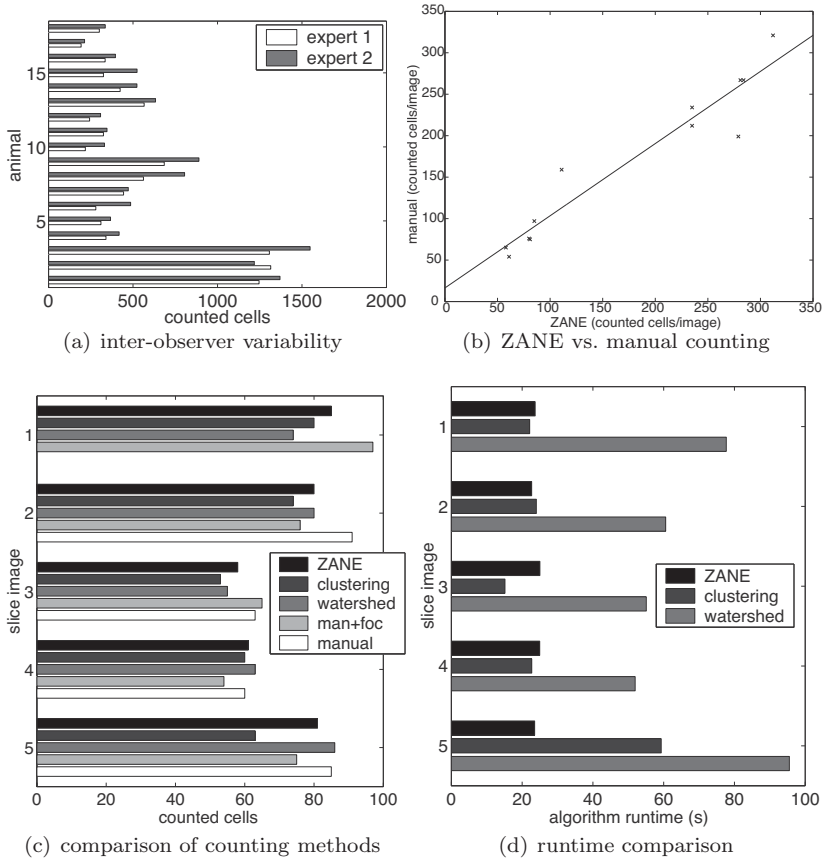
above the fixed focus plane of the digital image (in the digital image those cells are visible only as slightly darker shadows). We compare ZANE with Benali's clustering method [30] and a more elaborate segmentation scheme using the watershed transform for segmentation and afterward counting only sufficiently large clusters (see section 13.6).

Comparing mean manual counts using focusing versus ZANE, we get a standard deviation of ± 8.6 cells, and comparing it against the counts from the digital images, we get a deviation of ± 4.9 cells. In both cases the deviation is acceptable, taking into account that counts of two experts often vary by 5 to 10 cells or even more.

When comparing the manually segmented the digital images with the ZANE segmentations of images 2–5 from figure 13.11(c), we get the following average confusion matrix:

$$\begin{pmatrix} 90\% \pm 3.8\% & 4.1\% \pm 2.6\% \\ 9.9\% \pm 3.8\% & \end{pmatrix}. \tag{13.21}$$

This means that in the average ZANE correctly labeled 90% of all (manually detected) cells, additionally labeled 4.1% of all cells, and forgot to label 9.9% of all cells. Figure 13.11(b) shows a comparison of ZANE versus manual counting with varying focus for a larger data



(a) inter-observer variability (b) ZANE vs. manual counting

(c) comparison of counting methods (d) runtime comparison

Figure 13.11
Cell-counting results. (a) gives the counting results of two experts using a microscope (and changing the focus plane). (b) presents a comparison of ZANE versus manual microscope-based counting within various mice. (c) compares the ZANE counting performance with other proposed counting method as well as manual counting by microscope, counting only cells in the digital image within one mouse. (d) lists the corresponding algorithm runtimes. In (b) and (c), a single expert made all manual counts.

set (from two mice). The slope of the fitted line is about 0.9, which implies a counting error of around 10%.

The comparisons of ZANE with other counting algorithms also favor our proposed algorithm; indeed, the more advanced shape selectivity is

Table 13.1
Mean square error of automatic counting methods applied to the five section images from figure 13.11.

Algorithm	Counting error (vs. manual+focusing)	Counting error (vs. manual)
ZANE	3.4	3.5
clustering	5.0	6.8
watershed	5.8	5.4

preferable: using the data from figure 13.11(c), we get mean square errors of the three counting algorithms as shown in table 13.1. Clearly ZANE considerably outperforms the other two algorithms, especially in terms of counting performance given only the digital image data. A comparison of the computing times T (figure 13.11)(d), shows that ZANE (mean $\bar{T} = 24$sec) and Benali's clustering method (mean $\bar{T} = 29$sec) perform similarly, with the watershed taking roughly twice as long (mean $\bar{T} = 68$sec).

Finally, we wanted to test ZANE on images with different numbers of labeled cells and to give some neuro biologically interesting results. We therefore analyzed neurogenesis in the dentate gyrus of mice, and compared a control group of animals with mice that had been treated with pilocarpin to induce a status epilepticus. It is known that this condition raises the number of proliferating cells in the hippocampus one week after the treatment, which can be shown by an increase of BrdU-labeled cells [31, 196]. Our experiment confirmed these findings by showing that proliferation of cells in the dentate gyrus was 340% stronger in the epilepticmice than in the control group (see figure 13.12).

Counting images with multiple markers

The advantage of the presented method lies in the fact that it can be readily extended to the detection, localization, and identification of other kinds of cells in microscopic images. For example, images marked by multiple markers such as *neuronal nuclei antigen (NeuN)*, BrdU, *doublecortin (DCX)*, or *S100β* allow the differentiation of various types of cells. By adapting the cell classifier we can identify the desired type of cell, and multiple cell classifiers can then identify the various cell classes.

In the following we will demonstrate this by applying ZANE to a slice

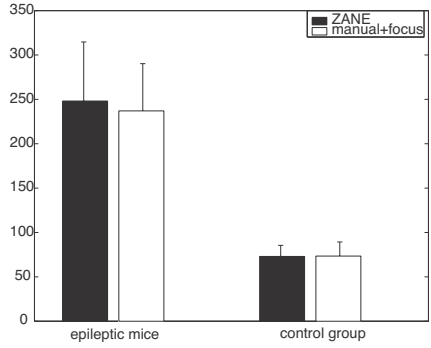


Figure 13.12
Number of cells in brain sections of a epileptic mouse versus number of cells of a mouse in a control group (counted both manually and using ZANE).

image that contains BrdU-marked cells in the green channel and S100β-labeled cells in the blue channel. Here, the polyclonal rabbit antibody S100β (SWant, Bellinzona, Switzerland) was used to detect astrocytes and glial cells. Newborn BrdU-labeled cells are located mainly in the subgranular layer (others are cut out due to ROI selection), and are not expected to be labeled by S100β. BrdU labels only newborn cells, and hence is not specific with respect to neurons or glial cells. The double labeling allows to us classify this (i.e., to exclude glial cells, which are not to be counted).

The BrdU-labeled cells, which in the multicolored images turned out to be larger, are counted using a cell classifier based on a two-layer MLP with two hidden neurons (see figure 13.13). The S100β-marked cells, the astrocytes, do not possess such a simple radial structure, and, moreover, axons and dendrites are also colored. Hence we apply a directional neural network based on a generalized regression network to construct a cell classifier. Due to the additional preparation time, essentially only three usable multi labeled scans were available for testing. The training data set was acquired from the first two images; 35 cell patches of size 34×34 were identified by an expert, and 350 non cell patches were randomly generated. The patches together with their x-mirrored equivalents were added after directional normalization (see section 13.4). Training of the RBF was fast, and after 30 epochs of batch training, the performance error was negligible, $3 \cdot 10^{-25}$. The cell classifier was applied to the third

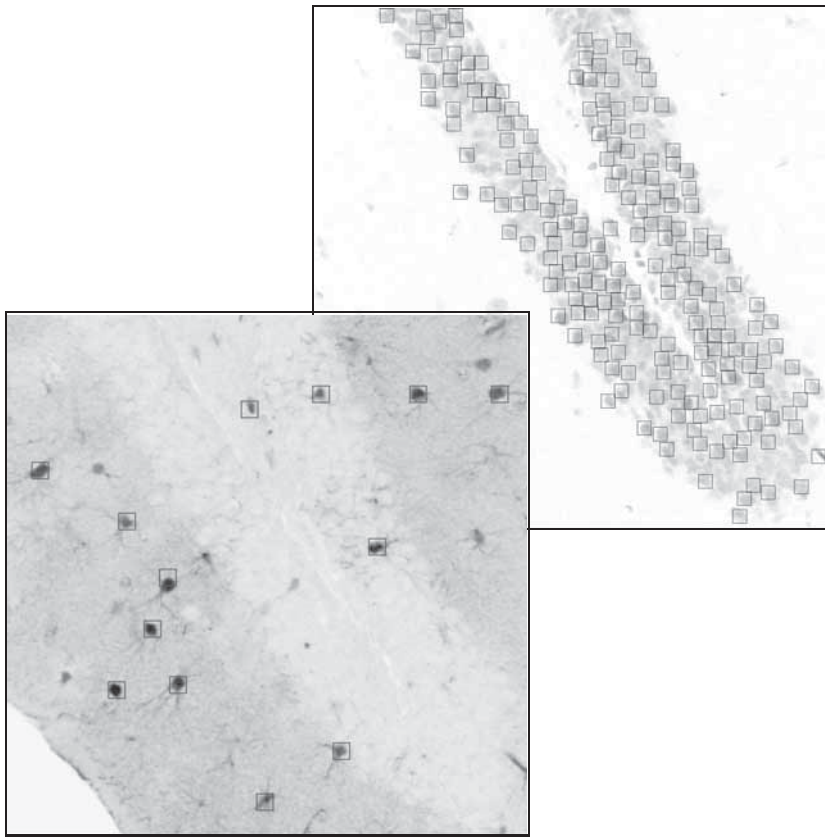


Figure 13.13
Multi colored image – counting in the green channel using MLP-based ZANE (top right image) and in the blue channel using a directional neural network (bottom left).

sample image with a high threshold of 0.9999, step size $\gamma = 10$ and cutout radius of 60. The resulting labeled image is shown in figure 13.13. Altogether, 13 neurons were counted and no overcounting was observed, but depending on the expert two or three additional possible neuron locations could have been added.

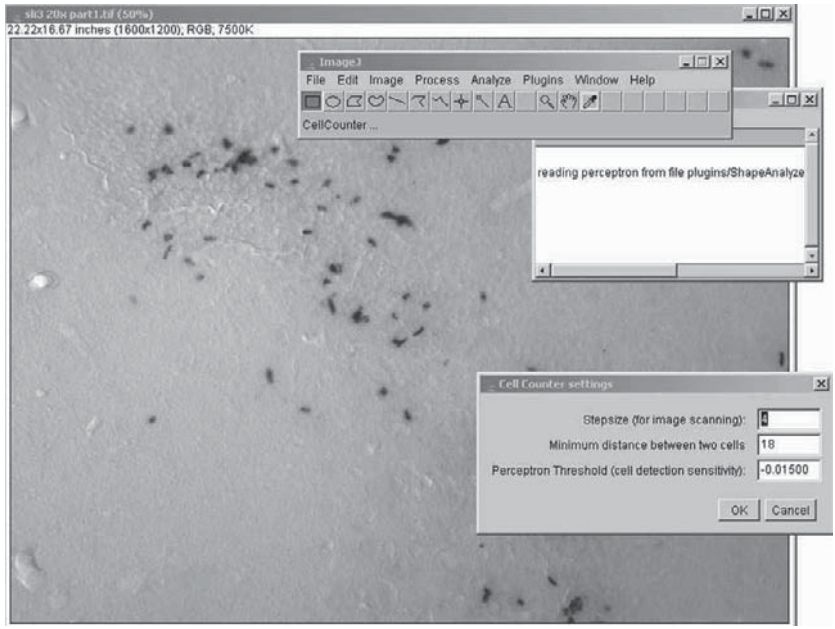


Figure 13.14
ZANE front end. A plug-in for the Java image editor ImageJ has been developed. It loads a perceptron text file and applies the ZANE image segmentation algorithm to give a segmented section as well as the cell count.

User interface

A visual front end has been developed in order to simplify the use of ZANE in the laboratory. It has been realized as plug-in for the Java image editor "ImageJ". An arbitrary linear cell classifier can be specified by a text file. Then the ZANE image segmentation algorithm is applied to calculate a segmented section and the cell count. A screen shot is shown in figure 13.14.

13.8 Conclusion

We have presented a framework for brain section image segmentation and analysis. The feature detector, here the cell classifier, was first trained onto a given sample set using neural networks or ICA. This

detector was then applied onto the image to get a confidence map, and maxima analysis yielded the cell locations. Experiments also showed good performance of the classifier when compared against more traditional segmentation techniques.

In future work, one goal is to face the problem that in typical brain section images, some cells not lying directly in the focus plane are blurred. In order to count those without counting them twice in two section images with different focus planes, a three-dimensional cell classifier can be trained for fixed focus plane distances. A different approach for accounting for non focused cells is simply to allow "overcounting", and then to reduce doubles in the segmented images according to location. This seems suitable, given the fact that cells do not vary greatly in size. We also plan on automating ROI selection in the future by classifying separating features of these regions.

14 NMR Water Artifact Removal

Multidimensional proton nuclear magnetic resonance (NMR) spectra of biomolecules dissolved in aqueous solutions are usually contaminated by an intense water artifact. In this chapter, we will discuss the application of the generalized eigenvalue decomposition method using a matrix pencil to solve the blind source separation (BSS) problem of separating out the water artifacts. BSS methods explore either the time structure of the free induction decay (*FID*) signals or their corresponding spectral power densities, using second-order correlations only. We analyze 2D spectra acquired from Nuclear Overhauser and Exchange Spectroscopy (*2-D NOESY*). The spectra of simple solutes as well as dissolved proteins are studied. Results are compared to those obtained with the FastICA algorithm, which explores higher-order statistical dependencies as well.

The ICA analyis of NMR spectra was first presented by Stadlthanner et al. [237] and extended in two conference proceedings [238, 239].

14.1 Use time-structure-based BSS

Blind source separation addresses the problem of finding which signals contribute to any given sensor signals recorded. It is of interest if little or nothing is known about the source signals and the mixing process, hence the term "blind". This is a widespread problem in signal processing, so BSS techniques have many applications in speech and image processing, biomedical signal processing, and communications, see chapter 4 for a review of BSS and related algorithms.

In general the problem is very ill-posed and needs to be regularized to become solvable. Two means have been considered in the past. Either one assumes statistically independent source signals and exploits higher-order correlations in the data – this has been done in the previous chapters – or one exploits time correlations in the data, relying on second-order statistics only. In any case, a linear mixing model is mostly considered. Most solutions consider a two-step procedure. During a whitening step the sensor signals are linearly transformed (via PCA, for example, see chapter 3) such that the covariance matrix becomes the identity matrix. During this step the dimensionality of the sensor signal vector can be reduced to the source vector dimensionality. The

problem is then reduced to finding an orthogonal (or unitary, in the case of complex-valued signals) separating matrix using higher-order or time-decorrelation algorithms. higher-order decorrelation techniques have been intensely studied, and many algorithms have been proposed, among which are the popular *Infomax* algorithm [25] or its natural gradient version [289], the JADE algorithm [47] that exploits fourth-order correlations and the very efficient FastICA algorithm, as well as geometric approaches such as the fastGeo algorithm [259] (see section 4.5).

Second-order techniques exploit the temporal structure of the source signals. The blind identification of the mixing model can be converted to standard (EVD) or generalized (GEVD) eigenvalue decomposition and simultaneous or joint diagonalization (SD) problems. In algorithms such as AMUSE and EFOBI [269], also see section 4.7 of the ICA chapter, a standard EVD is performed on a matrix derived from fourth-order cumulants or time-delayed correlations. Algorithms such as SOBI instead try to jointly diagonalize a set of delayed covariance matrices of whitened data to extract their average eigenstructure. Recently GEVD solutions have been presented which comprise the simultaneous diagonalization of a matrix pencil formed with the sensor signals. The matrices forming the pencil can be computed in different ways: Souloumiac [235] considers two segments of time-dependent signals with distinct energies, Lo et al. [160] consider different embedding spaces of chaotic signals, Molgedey and Schuster [178] and Chang et al. [49] compute time-delayed correlation matrices, and Tomé [266] considers filtered versions of the sensor signals. Later, Tomé [267] also presented an algebraic formulation of the GEVD problem using the notion of congruent matrix pencils and block matrix operations when the mixing matrix has more rows than columns. Iterative as well as *online* methods to compute the eigendecomposition of a symmetric, positive, definite pencil have also been presented [79, 268]. We will follow this latter approach and apply it to the separation of water artifacts from 2-D NOESY NMR proteins spectra.

14.2 The General Eigendecomposition Approach

For convenience we briefly review the *general eigendecomposition* approach using congruent matrix pencils. Consider the *matrix pen-*

cil $(\mathbf{R}_{s1}.\mathbf{R}_{s2})$ formed with the source signals and the matrix pencil $(\mathbf{R}_{x1}, \mathbf{R}_{x2})$ formed with the sensor signals. Both pencils are considered *congruent* if there exists an invertible matrix $\mathbf{A} \in \mathrm{Gl}(n)$ such that

$$\begin{aligned} \mathbf{R}_{x1} &= \mathbf{A}\mathbf{R}_{s1}\mathbf{A}^T \\ \mathbf{R}_{x2} &= \mathbf{A}\mathbf{R}_{s2}\mathbf{A}^T. \end{aligned} \tag{14.1}$$

In BSS problems $\mathbf{A} = \{a_{ij}\}, i = 1, ..., m, \quad j = 1, ..., n$ represents the instantaneous mixing matrix. It has been shown that the inverse or pseudo inverse of the mixing matrix can be estimated from the sensor signal pencil if the eigenvector matrix of the source signal pencil is diagonal. In fact, congruent pencils possess the same eigenvalues which form the roots of the characteristic polynomials

$$\begin{aligned} \chi_x(\lambda) &= \det(\mathbf{R}_{x1} - \mathbf{R}_{x2}) = 0 \\ \chi_s(\lambda) &= \det(\mathbf{R}_{s1} - \mathbf{R}_{s2}) = 0 \end{aligned} \tag{14.2}$$

With $\mathbf{A}$ a rectangular matrix $(m > n)$, if $\mathbf{A}^T\mathbf{A}$ is an invertible matrix, the congruent source signal pencil $(\mathbf{A}^T\mathbf{A}\mathbf{R}_{s1}\mathbf{A}^T\mathbf{A}, \mathbf{A}^T\mathbf{A}\mathbf{R}_{s2}\mathbf{A}^T\mathbf{A})$ also possesses the same eigenvalues. Hence the sensor signal pencil formed with $(m \times m)$ matrices shows n eigenvalues equal to the eigenvalues of the source signal pencil.

The generalized eigendecomposition of the sensor signal pencil now reads

$$\mathbf{R}_{x1}\mathbf{E} = \mathbf{R}_{x2}\mathbf{E}\mathbf{\Lambda} \tag{14.3}$$

where $\mathbf{E}$ represents the unique eigenvector matrix if the diagonal matrix $\mathbf{\Lambda}$ has distinct eigenvalues λ_i. The corresponding eigendecomposition statement concerning the source signal pencil can be obtained easily by substituting equation(14.1) into equation(14.3), yielding

$$\mathbf{A}\mathbf{R}_{s1}\mathbf{A}^T\mathbf{E} = \mathbf{A}\mathbf{R}_{s2}\mathbf{A}^T\mathbf{E}\mathbf{\Lambda} \tag{14.4}$$

Multiplying both sides of equation(14.4) by $\mathbf{A}^{-1}$ and using

$$\mathbf{E}_s = \mathbf{A}^T\mathbf{E} \tag{14.5}$$

as the corresponding eigendecomposition statement of the source signal pencil results in

$$\mathbf{R}_{s1}\mathbf{E}_s = \mathbf{R}_{s2}\mathbf{E}_s\mathbf{\Lambda}, \tag{14.6}$$

where $\mathbf{E}_s$ represents its eigenvector matrix, and the normalized eigenvectors corresponding to a particular eigenvalue are

$$\vec{e}_s = \alpha \mathbf{A}^T \vec{e} \tag{14.7}$$

with α a normalizing constant.

Concerning square ($m = n$) BSS problems, it can be seen from equation(14.5) that the eigenvector matrix $\mathbf{E}$ forms an estimate of the inverse of the mixing matrix $\mathbf{A}$ if the matrix $\mathbf{E}_s$ corresponds to the identity matrix or a simple permutation matrix. This occurs if the source signal pencils are both diagonal.

With nonsquare mixing matrices, equation (14.6) can be rewritten in block matrix notation if $\mathbf{A}$ and $\mathbf{E}$ are both divided into two blocks: $\mathbf{A}$ into $\mathbf{A}_H, (n \times n)$ and $\mathbf{A}_L, ((m-n) \times n)$, and $\mathbf{E}$ into $\mathbf{E}_H, (n \times m)$ and $\mathbf{E}_L, ((m-n) \times m)$. Then the eigendecomposition statement can be reformulated as

$$\begin{aligned} \mathbf{A}_H \mathbf{R}_{s1} \mathbf{\Phi} &= \mathbf{A}_H \mathbf{R}_{s2} \mathbf{E}_s \mathbf{\Lambda} \\ \mathbf{A}_L \mathbf{R}_{s1} \mathbf{\Phi} &= \mathbf{A}_L \mathbf{R}_{s2} \mathbf{E}_s \mathbf{\Lambda} \qquad (14.8) \\ \mathbf{E}_s &= \mathbf{A}_H^T \mathbf{E}_H + \mathbf{A}_L^T \mathbf{E}_L = \mathbf{A}^T \mathbf{E} \qquad (14.9) \end{aligned}$$

and $\mathbf{E}_s$ is now an $(n \times m)$ matrix representing the eigenvector matrix of the source signal pencil having $(m-n)$ columns of zeros paired with the corresponding eigenvalues in $\mathbf{\Lambda}$ that do not belong to the eigenvalue decomposition of the source signal pencil $(\mathbf{R}_{s1}, \mathbf{R}_{s2})$.

Since after the separation $(m-n)$ signals have vanishing amplitudes this approach also allows one to estimate the number of source signals. If the latter is known, then a subset of n sensor signals can be used to compute the corresponding matrix pencil, and identical results will be obtained. In summary, the GEVD approach to BSS problems is feasible if the congruent source signal pencils are formed with statistically independent source signals yielding the identity matrix or a permutation matrix only.

14.3 Computing the Eigendecomposition of Symmetric Pencils

The matrix pencil $(\mathbf{R}_{\vec{x},1}, \mathbf{R}_{\vec{x},2})$ of zero mean data comprises two correlation matrices of the data. The first matrix is computed as follows:

$$\mathbf{R}_{\vec{x},1} = \frac{1}{N}\mathbf{S}(\omega_2, t_1)\mathbf{S}^H(\omega_2, t_1), \tag{14.10}$$

with $N = 2048$ representing the number of samples in the ω_2 domain and $\mathbf{S}^H$ the conjugate transpose of the matrix $\mathbf{S}$. The second correlation matrix $\mathbf{R}_{\vec{x},2}$ of the pencil has been computed after filtering each single spectrum (each row of $\mathbf{S}(\omega_2, t_1)$) with a bandpass filter of Gaussian shape centered in the spectrum and having a variance in the range of $1 \leq \sigma^2 \leq 4$. Both matrices of the pencil are of dimension 128×128, since we assume as many sources as there are sensor signals.

A very common approach to computing the eigenvalues and eigenvectors of a matrix pencil is to reduce the GEVD statement

$$\mathbf{R}_{x2}\mathbf{E} = \mathbf{R}_{x1}\mathbf{E}\mathbf{\Lambda}$$

to the standard *eigenvalue decomposition* (EVD) problem, which is of the form

$$\mathbf{C}\mathbf{Z} = \mathbf{Z}\mathbf{\Lambda}.$$

The strategy that we will follow is first to solve the eigendecomposition of the matrix $\mathbf{R}_{x1}$, giving

$$\mathbf{R}_{x1} = \mathbf{S}\mathbf{D}\mathbf{S}^T = \mathbf{S}^{1/2}\mathbf{D}^{1/2}\mathbf{S}^T\mathbf{S}\mathbf{D}^{1/2}\mathbf{S}^T = \mathbf{W}\mathbf{W}.$$

Substituting this result into the GEVD statement and defining $\mathbf{Z} = \mathbf{W}\mathbf{E}$ yields the transformed equation

$$\mathbf{W}^{-1}\mathbf{R}_{x2}\mathbf{W}^{-1}\mathbf{Z} = \mathbf{Z}\mathbf{\Lambda},$$

which is the standard EVD form of a real symmetric matrix $\mathbf{C} = \mathbf{W}^{-1}\mathbf{R}_{x2}\mathbf{W}^{-1}$ if the matrix $\mathbf{R}_{x2}$ is also symmetric positive definite and the transformation matrix $\mathbf{W}^{-1}$ is obtained as

$$\mathbf{W}^{-1} = \mathbf{S}\mathbf{D}^{-1/2}\mathbf{S}^T.$$

While the eigenvalues of the matrix pencil are available from the solution of the EVD of the matrix $\mathbf{C}$, the corresponding eigenvectors are obtained via $\mathbf{E} = \mathbf{W}^{-1}\mathbf{Z}$.

14.4 NMR Spectra

Modern multidimensional *NMR spectroscopy* [76] is a versatile tool for the determination of the native 3-D structure of biomolecules in their natural aqueous environment. Proton NMR (i.e. the observation of the magnetization of the ^{1}H nuclei in the probe), is an indispensable contribution to this structure determination process but is hampered by the presence of the very intense water (H_2O) proton signal. Since it is the most intense signal in two-dimensional spectra, it causes the most trouble with baseline distortions and t_1 noise, and it can obscure weak signals lying under its edges. Because of its intensity it also causes severe dynamic range problems; hence sophisticated experimental protocols have been developed to suppress the water signal as far as possible. All these procedures introduce spectral distortions that can be neither avoided nor removed, and prevent the analysis of the spectral region close to the water resonance. Hence equivalent spectra of the molecules dissolved in heavy water (D_2O) also have to be taken which raises additional problems not the least being that heavy water differs sufficiently in its physical-chemical properties from light water to cast a doubt on a direct comparison of both spectra. Hence it is interesting to consider whether (BSS) techniques can contribute to the removal of the water artifact in such spectra without regard to any sophisticated water suppression pulse protocols except a simple presaturation to reduce the dynamic range problem. However, even a long, weak pulse on the water resonance can bleach nearby solute proton resonances and can also affect other signals through crossrelaxation or chemical exchange.

Concerning structure determination, homonuclear 2-D NOESY spectra are a must. They rely on the nuclear Overhauser effect, the change in the intensity of the resonance of one spin species upon saturation of an adjacent spin with which it has an appreciable dipole-dipole interaction. They provide information about crossrelaxation rates, which for protons depend mainly on magnetic dipolar interactions. The latter vary with distance as r^{-6}, and hence allow distances to neighboring nuclei to be determined. Loosely speaking, one can consider the NOE effect an atomic ruler, which allows the 3-D structure to be determined if enough NOEs are available experimentally. A two-dimensional NMR time domain signal, called free induction decay (FID), is modeled by a sum of

damped complex harmonic functions

$$S(t_1, t_2) = \sum_i M_i \exp(-i\Omega_{1i} t_1) \exp(-\lambda_{1i} t_1) \exp(-i\Omega_{2i} t_2) \exp(-\lambda_{2i} t_2)$$

on to which Gaussian noise is superimposed. Signal processing is performed by Fourier analysis, resulting in spectra consisting of sums of Lorentzian-shaped resonance lines [76] given by

$$f(\omega_1, \omega_2) = \sum_i M_i \left(\frac{1}{i\Delta\Omega_{1i} + \lambda_{1i}} \right) \left(\frac{1}{i\Delta\Omega_{2i} + \lambda_{2i}} \right).$$

Statistical independence of two signals requires their scalar product to be zero both in the time domain and in the frequency domain. Therefore nonoverlapping resonance lines should be reasonably independent. But because the limited range of chemical shifts (i.e. the spread of the proton resonances on the frequency scale) is rather limited compared to individual resonance line widths, statistical independence is hard to assure in general. second-order techniques like the GEVD using matrix pencils discussed above, as well as many others, exploit some weaker conditions for the separation of sources, assuming that they have a temporal structure with different autocorrelation functions or, equivalently, different power spectra.

14.5 Results and Discussion

EDTA spectra

First, 2-D NOESY spectra of simple solute molecules such as EDTA were analyzed. Presaturation of the water resonance was applied in all cases. FIDs $S(t_{1,j}, t_2)$ recorded at fixed evolution times $t_{1,j}, \quad j = 1, ..., m$ were sampled over time spans t_2 and have been Fourier transformed with respect to both time domains to obtain corresponding spectra $S(\omega_1, \omega_2)$ which could be corrected for any phase distortions. Data matrices $\mathbf{X} = \vec{x}_1, ..., \vec{x}_N$ were then formed with one row representing a single spectrum $S(\omega_2, t_{1,j})$ corresponding to a fixed evolution time $t_{1,j}$. The final $m \times N$ matrix $\mathbf{X}$ then contained as many rows as there were different evolution times $t_{1,j}$ according to the experimental protocol. Typically $m = 128$ evolution periods were considered and $N = 2048$ data points were sampled from each spectrum in the t_2 domain. Due to

phase cycling every fourth spectrum has been considered yielding only data matrices of size ($m \times N = 32 \times 2048$).

A matrix pencil ($\mathbf{C}_1, \mathbf{C}_2$) comprised two covariance matrices $\mathbf{C}$ of the data where the second covariance matrix $\mathbf{C}_2$ represented a delayed or filtered version of $\mathbf{R}_1$. With zero mean data the covariance matrices $\mathbf{C}$ of the data equaled their correlation matrices $\mathbf{C} = \mathbf{R}$. The latter were of dimension 32×32, and the expectations were estimated according to

$$< x_i x_j >= \frac{1}{N} \sum_{n=1}^{N} x_i(n) x_j(n) \tag{14.11}$$

with $N = 2048$ representing the number of samples in the ω_1 domain in the case of $\mathbf{R}_1$. The second correlation matrix $\mathbf{R}_2$ of the pencil was obtained in two different ways:

- First, by collecting spectral data at frequencies below the water resonance (i.e., only data points between 1285 and 2048) were used to calculate the expectations in the covariance matrix R_2 of the pencil. That amounts to low-pass filtering the whole spectrum. Any smaller frequency shifts did not yield reasonable results (i.e., a successful separation of the water and the EDTA resonances could not be obtained).
- A second procedure consisted in bandpass filtering the water resonance in the frequency domain with a narrow-band filter which removed only the water resonance. The spectra were then converted to the time domain with an inverse Fourier transform, and corresponding correlation matrices were calculated with time domain data for both correlation matrices of the pencil. Even in the case of $\mathbf{R}_1$ the data had to be Fourier-transformed first to be able to effect a phase correction to the spectra, which then were subjected to an inverse Fourier transform to obtain suitably corrected time domain data.

The matrix pencil thus obtained was treated in the manner given above to estimate the independent components of the EDTA spectra and the corresponding demixing matrix. Independent components showing spectral energy only in the frequency range of the water resonance were related to the water artifact. To effect a separation of the water artifact and the EDTA spectra, these water-related independent components were deliberately set to zero. Then the whole EDTA spectrum could be reconstructed with the estimated inverse of the demixing matrix and the

corrected matrix of estimated source signals.

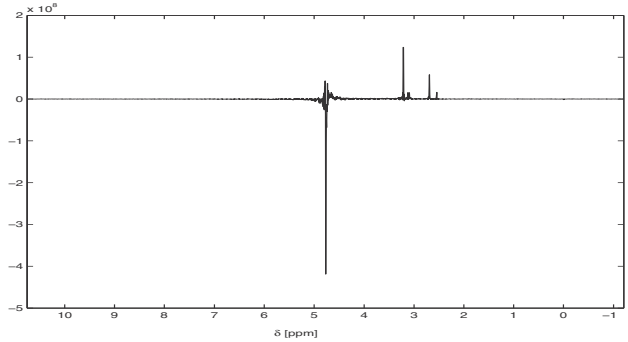
A typical 1-D EDTA spectrum is shown in figure 14.1(a). It illustrates the still intense water artifact around sample point 1050, corresponding to a frequency shift of 4.8 ppm relative to the resonance frequency of the standard. Figure 14.1(b) presents the reconstructed spectrum with the water artifact removed. The small distortions remaining are due to baseline artifacts caused by truncating the FID due to limited sampling times.

To see whether the use of higher-order statistics could perform better the data set has also been analyzed with the FastICA algorithm [124]. As the latter does not use any time structure, all 128 data points in each column of the (128×2048)-dimensional data matrix $\mathbf{X}$ were used. Again, independent components related to the water artifact were nulled in the reconstruction procedure. The result is shown in figure 14.1(c). Visual inspection shows a comparable separation quality of both methods in the case of 2-D NOESY EDTA spectra.

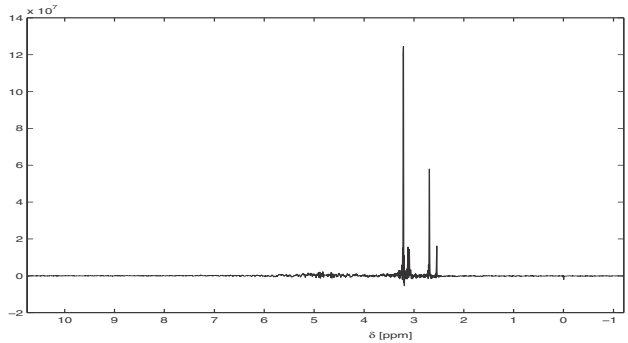
Simulated protein spectra

We then analyzed simulated noise- and artifact-free 2-D NOESY spectra of the cold-shock protein (CSP) of *thermotoga maritima*, comprising 66 amino acids, were overlaid with experimental NOESY spectra of pure water taken with presaturation of the water resonance to simulate conditions corresponding to experimental protein NOESY spectra to be analyzed later on.

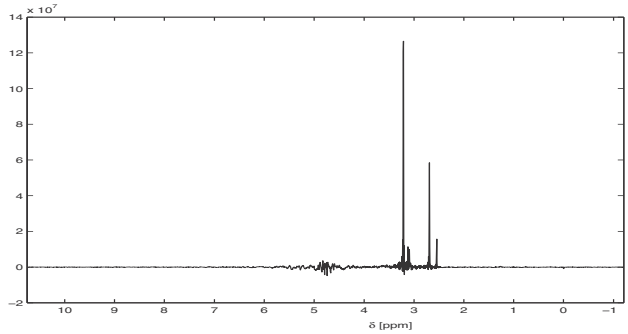
A 1-D CSP spectrum backcalculated with the RELAX algorithm overlaid with the experimental water spectrum is shown in figure 14.2(a), illustrating the realistically scaled, rather intense water artifact around sample point 1050. The matrix pencil calculated from these data was treated in the manner given above to estimate the independent components (ICs) of the artificial CSP spectra and the corresponding demixing matrix. Figure 14.2(b,c) present the reconstructed spectra with the water artifact removed using the matric pencial algorithm and the fastICA algorithm. The small distortions remaining are due to a limited number of ICs components estimated. Attempts to overlay water spectra that have been taken without presaturation, and hence show an undistorted water resonance, indicated that a 3×3 mixing matrix then suffices to reach an equally good separation. This is due to the fact that the presat-



(a) 1-D slice of 2-D NOESY data



(b) reconstruction with removed water artifact using matrix pencil



(c) reconstruction with removed water artifact using ICA

Figure 14.1
(a) 1-D slice of a 2-D NOESY spectrum of EDTA in aqueous solution corresponding to the shortest evolution period t_2. The chemical shift ranges from -1.206 ppm to 10.759 ppm. (b) Reconstructed EDTA spectrum (a) with the water artifact removed using frequency structure by applying the proposed matrix pencil algorithm. (c) Reconstructed spectrum using statistical independence (fastICA).

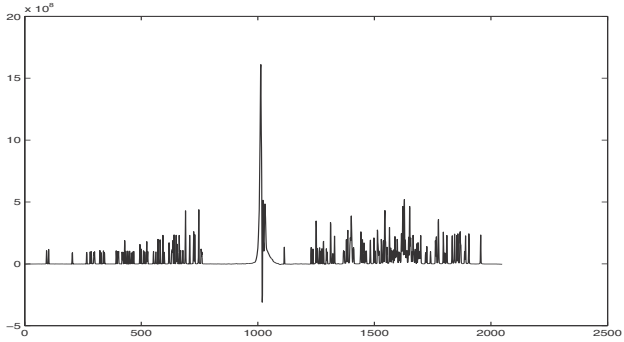
uration pulse introduces many phase distortions, which then cause the algorithm to decompose the water resonance into many ICs instead of just one.

The fastICA results are somewhat less convincing; indeed the algorithm introduced spectral distortions such as inverted multiplets, hardly visible on the figures presented, that not observed in the analysis with the GEVD method using a matrix pencil. This is of course an important issue concerning an automated water artifact separation procedure, as any spectral distortions might result in false structure determinations using these 2-D NOESY data.

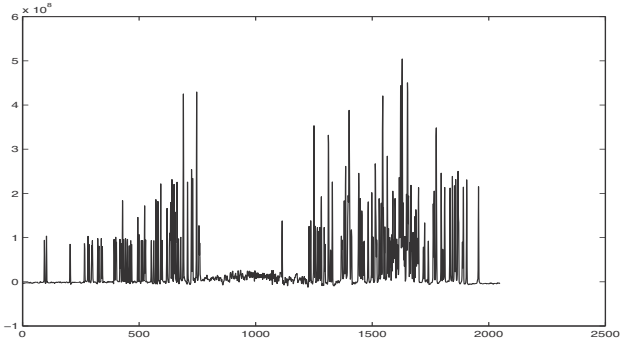
Spectra of the protein RALGEF

As a second data set 2-D NOESY spectra of the protein RALH814 were analyzed as well. The data were analyzed with the matrix pencil method as described above. This time both correlation matrices had the dimension (128×128) and all 2048 data points were used to estimate the expectations within the correlation matrices. Again the second correlation matrix $\mathbf{R}_2$ of the matrix pencil corresponded to a bandpass-filtered version of the correlation matrix $\mathbf{R}_1$. Figure 14.3 shows an original protein spectrum with the prominent water artifact, its reconstructed version with the water artifact separated out, and a spectrum difference between original and reconstructed spectra.

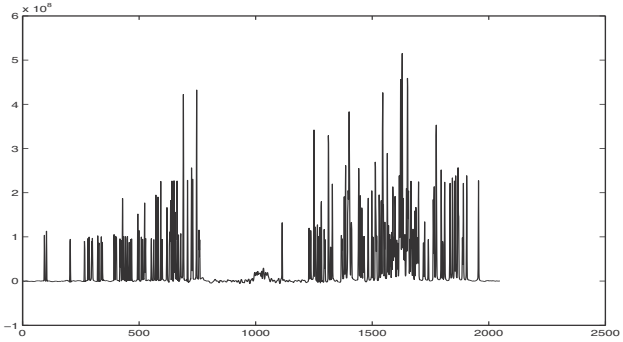
An equally good separation of the water artifact could have been obtained if the correlation matrix $\mathbf{R}_2$ had been calculated by estimating the corresponding expectations with the low-frequency samples, those with shifts below the water resonance, of the spectrum only (see figure 14.4(a)). Again the data were analyzed with the FastICA algorithm as well yielding comparable results (see figure 14.4(b)). However, though hardly visible on the figures presented, the FastICA algorithm introduced some spectral distortions that had not been observed in the analysis with the GEVD method using a matrix pencil. This is of course an important issue concerning an automated water artifact separation procedure, as any spectral distortions might result in false structure determinations using these 2-D NOESY data.



(a) simulated CSP and water spectrum

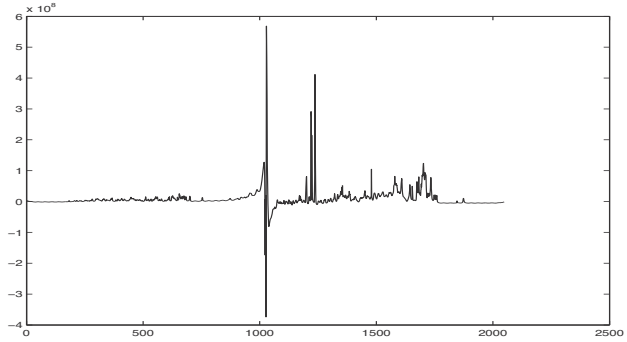


(b) reconstruction with removed water artifact using GEVD

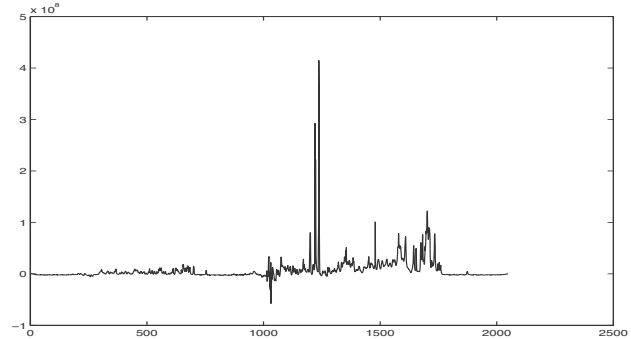


(c) reconstruction with removed water artifact using ICA

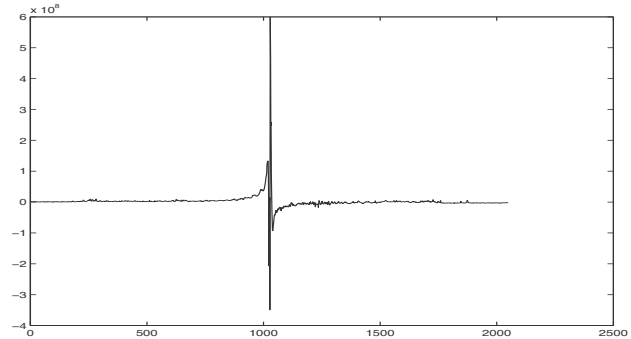
Figure 14.2
(a) 1-D slice of a simulated 2-D NOESY spectrum of CSP overlaid with an experimental water spectrum corresponding to the shortest evolution period t_2. The chemical shift ranges from 10.771 ppm (left) to -1.241 ppm (right). Only the real part of the complex quantity $S(\omega_2, t_1)$ is shown. Reconstructed CSP spectra with the water artifact removed by solving the BSS problem using a congruent matrix pencil (b) and the fastICA algorithm (c).



(a) simulated CSP and water spectrum

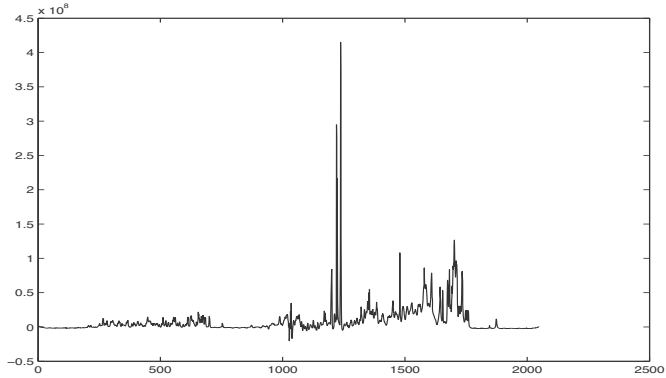


(b) reconstruction with removed water artifact using GEVD

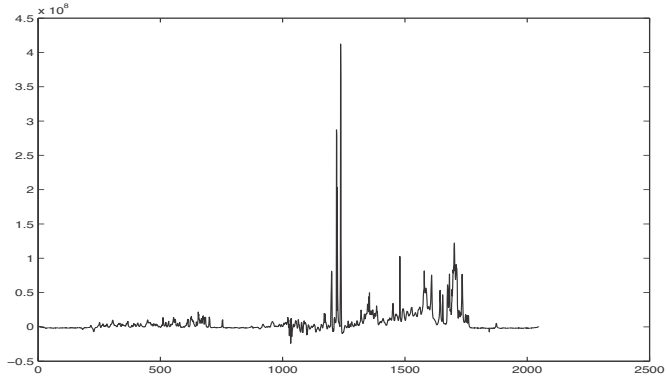


(c) reconstruction with removed water artifact using GEVD

Figure 14.3
(a) 1-D slice of a 2-D NOESY spectrum of the protein RALH814 in aqueous solution corresponding to the shortest evolution period t_2. The chemical shift ranges from −1.189 ppm to 10.822 ppm, i.e. one digit corresponds to a shift of 5.864E-3 ppm. (b) Reconstructed protein spectrum with the water artifact removed with the GEVD using a matrix pencil. (c) Difference between original and reconstructed protein spectra.



(a) modified reconstruction with removed water artifact using GEVD



(b) reconstruction with removed water artifact using ICA

Figure 14.4
Reconstructed protein spectrum obtained with the GEVD algorithm using a matrix pencil(a) and fastICA (b). In (a), the expectations within the second covariance matrix were calculated using low-frequency sample points only.

14.6 Conclusions

Proton 2-D NOESY spectra are an indispensable part of any determination of the three-dimensional conformation of native proteins, which forms the basis for understanding their function in living cells. Water is the most abundant molecule in biological systems, hence proton protein spectra are generally contaminated by large water resonances that cause

severe dynamic range problems. We have shown that ICA methods can be useful to separate these water artifacts out and obtain largely undistorted, pure protein spectra. Generalized eigenvalue decompositions using a matrix pencil are an exact and easily applied second-order technique to effect such arefact removal from the spectra. We have tested this method with simple EDTA spectra where no solute resonances appear close to the water resonance. Application of the method to protein spectra with resonances hidden in part by the water resonance showed a good separation quality with few remaining spectral distortions in the frequency range of the removed water resonance. It is important to note that no noticeable spectral distortions were introduced farther away from the water artifact, in contrast to the FastICA algorithm, which introduced distortions in other parts of the spectrum. Further, baseline artifacts due to the intense water resonance can also be cured to a large extent with this procedure. Further investigations will have to improve the separation quality even further and to determine whether solute resonances hidden underneath the water resonance can be made visible with these or related methods.

References

[1]P. Abdolmaleki, L. Buadu, and H. Naderimansh. Feature extraction and classification of breast cancer on dynamic magnetic resonance imaging using artificial neural network. *Cancer Letters*, 171(8):183–191, 2001.

[2]K. Abed-Meraim and A. Belouchrani. Algorithms for joint block diagonalization. In *Proc. EUSIPCO 2004*, pages 209–212, 2004.

[3]S. Akaho, Y. Kiuchi, and S. Umeyama. MICA: Multimodal independent component analysis. In *Proc. IJCNN 1999*, pages 927–932, 1999.

[4]A. N. Akansu and R. A. Haddad. *Multiresolution Signal Decomposition*. Academic Press, 1992.

[5]M. Akay. *Time-Frequency and Wavelets in Biomedical Signal Processing*. IEEE Press, 1997.

[6]E. Alpaydin. *Introduction to Machine Learning*. MIT Press, 2004.

[7]J. Altman and G. Das. Autoradiographic and histological evidence of postnatal hippocampal neurogenesis in rats. *J. Comp. Neurol.*, 124(3):319–335, 1965.

[8]S. Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10:251–276, 1998.

[9]M. Anthony and P. Bartlett. Neural Network Learning: Theoretical Foundations. *Cambridge University Press*, 1999.

[10]K. Arfanakis, D. Cordes, V. Haughton, M. Moritz, M. Quigley, and M. Meyerand. Combining independent component analysis and correlation analysis to probe interregional connectivity in fMRI task activation datasets. *Magnetic Resonance Imaging*, 18(8):921–930, 2000.

[11]L. Axel. Cerebral blood flow determination by rapid-sequence computed tomography. *Radiology*, 137(10):679–686, 1980.

[12]F. Bach and M. Jordan. Beyond independent components: trees and clusters. *Journal of Machine Learning Research*, 4:1205–1233, 2003.

[13]F. Bach and M. Jordan. Finding clusters in independent component analysis. In *Proc. ICA 2003*, pages 891–896, 2003.

[14]W. Backfrieder, R. Baumgartner, M. Samal, E. Moser, and H. Bergmann. Quantification of intensity variations in functional mr images using rotated principal components. *Phys. Med. Biol.*, 41(8):1425–1438, 1996.

[15]A. Baraldi and P. Blonda. A survey of fuzzy clustering algorithms for pattern recognition-part ii. *IEEE Transactions on Systems, Man and Cybernetics, part B*, 29(6):786–801, 1999.

[16]A. Barnea and F. Nottebohm. Seasonal recruitment of hippocampal neurons in adult free-ranging black-capped chickadees. *Proc. Natl. Acad. Sci. USA*, 91(23):11217–11221, 1994.

[17]M. Bartlett. *Face Image Analysis by Unsupervised Learning and Redundancy Reduction*. PhD thesis, University of California at San Diego, 1998.

[18]M. Bartlett and T. Sejnowski. Independent components of face images: A representation for face recognition. In *Proceedings of the 4th Annual Joint Symposium on Neural Computation*, 1997.

[19]C. Bauer. *Independent Component Analysis of Biomedical Signals*. Logos Verlag Berlin, 2001.

[20]C. Bauer, C. Puntonet, M. Rodriguez-Alvarez, and E. Lang. Separation of EEG signals with geometric procedures. *C. Fyfe, ed., Engineering of Intelligent Systems (Proc. EIS 2000)*, pages 104–108, 2000.

[21]C. Bauer, F. Theis, W. Bumler, and E. Lang. Local features in biomedical image clusters extracted with independent component analysis. In *Proc. IJCNN 2003*, pages 81–84, 2003.

[22]H. Bauer. *Mass- und Integrationstheorie*. Walter de Gruyter, Berlin and New York, 1990.

[23]H. Bauer. *Wahrscheinlichkeitstheorie*. 4th ed. Walter de Gruyter, Berlin and New York, 1990.

[24]R. Baumgartner, L. Ryder, W. Richter, R. Summers, M. Jarmasz, and R. Somorjai. Comparison of two exploratory data analysis methods for fMRI: fuzzy clustering versus principal component analysis. *Magnetic Resonance Imaging*, 18(8):89–94, 2000.

[25]A. Bell and T. Sejnowski. An information-maximisation approach to blind separation and blind deconvolution. *Neural Computation*, 7:1129–1159, 1995.

[26]A. J. Bell and T. J. Sejnowski. An information-maximisation approach to blind separation and blind deconvolution. *Neural Computation*, 7:1129–1159, 1995.

[27]A. Belouchrani and M. Amin. Blind source separation based on time-frequency signal representations. *IEEE Trans. Signal Processing*, 46(11):2888–2897, 1998.

[28]A. Belouchrani, K. A. Meraim, J.-F. Cardoso, and E. Moulines. A blind source separation technique based on second order statistics. *IEEE Transactions on Signal Processing*, 45(2):434–444, 1997.

[29]S. Ben-Yacoub. Fast object detection using MLP and FFT. IDIAP-RR 11, IDIAP, 1997.

[30]A. Benali, I. Leefken, U. Eysel, and E. Weiler. A computerized image analysis system for quantitative analysis of cells in histological brain sections. *Journal of Neuroscience Methods*, 125:33–43, 2003.

[31]J. Bengzon, Z. Kokaia, E. Elmer, A. Nanobashvili, M. Kokaia, and O. Lindvall. Apoptosis and proliferation of dentate gyrus neurons after single and intermittent limbic seizures. *Proc. Natl. Acad. Sci. USA*, 94:10432–10437, 1997.

[32]S. Beucher and C. Lantuéjoul. Use of watersheds in contour detection. In *International Workshop on Image Processing, Real-Time Edge and Motion Detection/Estimation. IRISA Report, Vol. 132*, page 132, Rennes, France, 1979.

[33]J. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, 1981.

[34]C. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[35]B. Biswal, Z. Yetkin, V. Haughton, and J. Hyde. Functional connectivity in the motor cortex of resting human brain using echoplanar MRI. *Magnetic Resonance in Medicine*, 34(8):537–541, 1995.

[36]B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Proc. COLT 1992*, pages 144–152, 1992.

[37]C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[38]C. Burrus, R. A. Gopinath, and H. Guo. *Introduction to Wavelets and Wavelet Transform*. Prentice Hall, 1997.

[39]R. B. Calinski and J. Harabasz. A dendrite method for cluster analysis. *Communications in Statistics – Theory and Methods*, 3(9):1–27, 1974.

[40]H. Cameron, C. Woolley, B. McEwen, and E. Gould. Differentiation of newly born neurons and glia in the dentate gyrus of the adult rat. *Neuroscience*, 56(2):337–344, 1993.

[41]M. Capek, R. Wegenkittl, and P. Felkel. A fully automatic stitching of 2D medical datasets. In J. Jan, J. Kozumplik, and I. Provaznik, editors, *BIOSIGNAL 2002: The 16th international EURASIP Conference*, pages 326–328, 2002.

[42]J. Cardoso and A. Souloumiac. Jacobi angles for simultaneous diagonalization. *SIAM J. Mat. Anal. Appl.*, 17(1):161–164, 1995.

[43]J.-F. Cardoso. Multidimensional independent component analysis. In *Proc. of ICASSP '98*, 1998.

[44]J.-F. Cardoso and B. Laheld. Equivariant adaptive source separation. *IEEE Transactions on Signal Processing*, 44(12):3017–3030, 1996.

[45]J.-F. Cardoso and A. Souloumiac. Localization and identification with the quadricovariance. *Traitement du Signal*, 7(5):397–406, 1990.

[46]J.-F. Cardoso and A. Souloumiac. Blind beamforming for non-Gaussian signals. *IEE Proceedings-F*, 140(6):362–370, 1993.

[47]J.-F. Cardoso and A. Souloumiac. Jacobi angles for simultaneous diagonalization. *SIAM Journal on Matrix Analysis and Applications*, 17:161–164, 1995.

[48]K. Castleman. *Digital Image Processing*. Prentice Hall, 1996.

[49]C. Chang, Z. Ding, S. Yau, and F. Chan. A matrix pencil approach to blind source separation of colored nonstationary signals. *IEEE Transactions on Signal Processing*, 48:900–907, 2000.

[50]S. Chatterjee, M. Laudato, and L. Lynch. Genetic algorithms and their statistical applications: An introduction. *Computational Statistics and Data Analysis*, 22(11):633–651, 11 1996.

[51]Z. Cho, J. Jones, and M. Singh. *Foundations of Medical Imaging*. J. Wiley Interscience, 1993.

[52]S. Choi and A. Cichocki. Blind separation of nonstationary sources in noisy mixtures. *Electronics Letters*, 36(848-849), 2000.

[53]K. Chuang, M. Chiu, C. Lin, and J. Chen. Model-free functional MRI analysis using Kohonen clustering neural network and fuzzy c-means. *IEEE Transactions on Medical Imaging*, 18(12):1117–1128, 1999.

[54]E. Ciaccio, S. Dunn, and M. Akay. Biosignal pattern recognition and interpretation systems: Part I. *IEEE Engineering in Medicine and Biology*, 13(9):89–97, 1993.

[55]E. Ciaccio, S. Dunn, and M. Akay. Biosignal pattern recognition and interpretation systems: Part III. *IEEE Engineering in Medicine and Biology*, 14(9):129–135, 1994.

[56]E. Ciaccio, S. Dunn, and M. Akay. Biosignal pattern recognition and interpretation systems: Part IV. *IEEE Engineering in Medicine and Biology*, 14(5):269–283, 1994.

[57]A. Cichocki and S. Amari. *Adaptive blind signal and image processing*. John Wiley, 2002.

[58]L. Cohen. *Time-Frequency Analysis*. Prentice Hall, Englewood Cliffs, NJ, 1995.

[59]P. Comon. Independent component analysis-a new concept? *Signal Processing*, 36:287–314, 1994.

[60]P. Cosman, R. Gray, and R. Olshen. Evaluating quality of compressed medical images: SNR subjective rating, and diagnostic accuracy. *Proc. IEEE*, 82(6):919–932, 1994.

[61]G. H. D. Rumelhart and J. McClelland. *A general framework for parallel distributed processing*. Cambridge Press, 1986.

[62]G. Darmois. Analyse générale des liaisons stochastiques. *Rev. Inst. Internationale Statist.*, 21:2–8, 1953.

[63]R. Dave. Fuzzy shell clustering and applications to circle detection in digital images. *International Journal of General Systems*, 16(4):343–355, 1990.

[64]R. Dave and K. Bhaswan. Adaptive fuzzy c–shells clustering and detection of

ellipses. *IEEE Transactions on Neural Networks*, 3(5):643–662, 1992.

[65]S. Davis, M. Fisher, and S. Warach. *Magnetic Resonance Imaging in Stroke*. Cambridge University Press, Cambridge, 2003.

[66]A. Dhawan, Y. Chitre, C. Kaiser-Bonasso, and M. Moskowitz. Analysis of mammographic microcalcifications using gray-level image structure features. *IEEE Transaction on Medical Imaging*, 15(3):246–259, 1996.

[67]A. Dhawan and E. LeRoyer. Mammographic feature enhancement by computerized image processing. *Computer Methods and Programs in Biomedicine*, 27(1):23–33, 1988.

[68]H. Digabel and C. Lantuéjoul. Iterative algorithms. In *Actes du Second Symposium Européen d'Analyse Quantitative des Microstructures en Sciences des Matériaux, Biologie et Médecine*, pages 85–99. Riederer Verlag, Stuttgart, 1977.

[69]F. Dolbeare. Bromodeoxyuridine: A diagnostic tool in biology and medicine, part I: Historical perspectives, histochemical methods and cell kinetics. *Histochem. J.*, 27(5):339–369, 1995.

[70]R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. Wiley, 1973.

[71]D. Dumitrescu, B. Lazzerini, and L. Jain. *Fuzzy Sets and Their Application to Clustering and Training*. CRC Press, 2000.

[72]e. a. E. Gould, P. Tanapat. Proliferation of granule cell precursors in the dentate gyrus of adult monkeys is diminished by stress. *Proc. Natl. Acad. Sci. USA*, 95(6):3168–3171, 1998.

[73]J. Eriksson and V. Koivunen. Identifiability and separability of linear ica models revisited. In *Proc. of ICA 2003*, pages 23–27, 2003.

[74]J. Eriksson and V. Koivunen. Complex random vectors and ICA models: Identifiability, uniqueness, and separability. *IEEE Transactions on Information Theory*, 52(3):1017–1029, 2006.

[75]P. Eriksson, E. Perfilieva, T. Bjork-Eriksson, A. Alborn, C. Nordborg, D. Peterson, and F. Gage. Neurogenesis in the adult human hippocampus. *Nat. Med.*, 4(11):1313–1317, 1998.

[76]R. Ernst, G. Bodenhausen, and A. Wokaun. *Principles of nuclear magnetic resonance in one and two dimensions*. Oxford University Press, 1987.

[77]F. Esposito, E. Formisano, E. Seifritz, R. Goebel, R. Morrone, G. Tedeschi, and F. D. Salle. Spatial independent component analysis of functional MRI time–series: to what extent do results depend on the algorithm used? *Human Brain Mapping*, 16(8):146–157, 2002.

[78]J. Fan. *Overcomplete Wavelet Representations with Applications in Image Processing*. PhD thesis, University of Florida, 1997.

[79]N. Ferreira and A. Tomé. Blind source separation of temporally correlated signals. In *Proc. RECPAD 02*, 2002.

[80]C. Févotte and F. Theis. Orthonormal approximate joint block-diagonalization. Technical report, GET/Télécom, Paris, 2007.

[81]C. Févotte and F. Theis. Pivot selection strategies in Jacobi joint block-diagonalization. In *Proc. ICA 2007*, volume 4666 of *LNCS*, pages 177–184. Springer, London, 2007.

[82]U. Fischer, V. Heyden, I. Vosshenrich, I. Vieweg, and E. Grabbe. Signal characteristics of malignant and benign lesions in dynamic 2D-MRI of the breast. *RoFo*, 158(8):287–292, 1993.

[83]C. Fisel, J. Ackerman, R. Bruxton, L. Garrido, J. Belliveau, B. Rson, and T. Brady. MR contrast due to microscopically heterogeneous magnetic susceptibility: Numerical simulations and applications to cerebral physiology.

Magn. Reson. Med., (6):336–347, 1991.

[84]H. Fisher and J. Hennig. Clustering of functional MR data. *Proc. ISMRM 4th Ann. Meeting*, 96(8):1179–1183, 1996.

[85]H. Fisher and J. Hennig. Neural network-based analysis of MR time series. *Magnetic Resonance in Medicine*, 41(8):124–131, 1999.

[86]N. C. for Health Statistics. National Vital Statistics Reports. vol. 6, 1999.

[87]J. Friedman and J. Tukey. A projection pursuit algorithm for exploratory data analysis. *IEEE Transactions on Computers*, 23(9):881–890, 1975.

[88]D. Gabor. Theory of communication. *Journal of Applied Physiology of the IEE*, 93(10):429–457, 1946.

[89]I. Gath and A. Geva. Unsupervised optimal fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(3):773–781, 1989.

[90]P. Georgiev, P. Pardalos, F. Theis, A. Cichocki, and H. Bakardjian. *Data Mining in Biomedicine*, chapter Sparse component analysis: a new tool for data mining. Springer, in print, 2005.

[91]P. Georgiev and F. Theis. Blind source separation of linear mixtures with singular matrices. In *Proc. ICA 2004*, volume 3195 of *LNCS*, pages 121–128. Springer, 2004.

[92]C. Gerard and B. Rollins. Chemokines and disease. *Nat. Immunol.*, 2:108–115, 2001.

[93]S. Ghurye and I. Olkin. A characterization of the multivariate normal distribution. *Ann. Math. Statist.*, 33:533–541, 1962.

[94]S. Goldman and F. Nottebohm. Neuronal production, migration, and differentiation in a vocal control nucleus of the adult female canary brain. *Proc. Natl. Acad. Sci. USA*, 80(8):2390–2394, 1983.

[95]R. C. Gonzalez and R. Woods. *Digital Image Processing*. Prentice Hall, 2002.

[96]M. Goodrich, J. Mitchell, and M. Orletsky. Approximate geometric pattern matching under rigid motions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(4):371–379, 1999.

[97]C. Goutte, P. Toft, E. Rostrup, F. Nielsen, and L. Hansen. On clustering fmri series. *NeuroImage*, 9(3):298–310, 1999.

[98]T. Graepel and K. Obermayer. A stochastic self–organizing map for proximity data. *Neural Computation*, 11(7):139–155, 1999.

[99]R. Gray. Vector quantization. *IEEE ASSP Magazine*, 1(1):4–29, 1984.

[100]S. Grossberg. Adaptive pattern classification and universal recording. *Biological Cybernetics*, 23(7):121–134, 1976.

[101]S. Grossberg. Competition, decision and consensus. *Journal of Mathematical Analysis and Applications*, 66:470–493, 7 1978.

[102]P. Gruber, C. Kohler, and F. Theis. A toolbox for model-free analysis of fMRI data. In *Proc. ICA 2007*, volume 4666 of *LNCS*, pages 209–217. Springer, London, 2007.

[103]M. Gudmundsson, E. El-Kwae, and M. Kabuka. Edge detection in medical iamges uisng a genetic algorithm. *IEEE Transactions on Medical Imaging*, 17(3):469–474, 1998.

[104]H. Gutch and F. Theis. Independent subspace analysis is unique, given irreducibility. In *Proc. ICA 2007*, volume 4666 of *LNCS*, pages 49–56. Springer, London, 2007.

[105]M. Habl. Nichtlineare Analyseverfahren zur Extraction statistisch unabhängiger Komponenten aus multisensorischen EEG-Datensätzen. *Diploma*

Thesis, Institute of Biophysics, University of Regensburg, Germany, 2000.

[106]O. Haraldseth, R. Jones, T. Muller, A. Fahlvik, and A. Oksendal. Comparison of DTPA, BMA and superparamagnetic iron oxide particles as susceptibility contrast agents for perfusion imaging of regional cerebral ischemia in the rat. *J. Magn. Reson. Imaging*, (8):714–717, 1996.

[107]K. Haris, S. N. Efstratiadis, N. Maglaveras, and A. Katsaggelos. Hybrid image segmentation using watershed and fast region merging. *IEEE Trans. Img. Proc.*, 7(12):1684–1699, 1998.

[108]D. Hartl, M. Griese, R. Gruber, D. Reinhardt, D. Schendel, and S. Krauss-Etschmann. Expression of chemokine receptors ccr5 and cxcr3 on t cells in bronchoalveolar lavage and peripheral blood in pediatric pulmonary diseases. *Immunobiology*, 206(1 - 3):224–225, 2002.

[109]E. J. Hartman, J. D. Keeler, and J. M. Kowalski. Layered neural networks with Gaussian hidden units as universal approximations. *Neural Computation*, 2(2):210–215, 1990.

[110]S. Haykin. *Neural Networks*. Macmillan College Publishing, 1994.

[111]S. Haykin. Neural networks. *Macmillan College Publishing Company*, 1994.

[112]J. Hérault and C. Jutten. Space or time adaptive signal processing by neural network models. In J. Denker, editor, *Neural Networks for Computing: Proceedings of the AIP Conference*, pages 206–211, New York, 1986. American Institute of Physics.

[113]J. Hertz, A. Krogh, and R. Palmer. *Introduction to the Theory of Neural Computation*. Addison-Wesley Publishing Company, Redwood City, 1991.

[114]H. Herzog. Basic ideas and principles for quantifying regional blood flow with nuclear medical techniques. *Nuklearmedizin*, (5):181–185, 1996.

[115]S. Heywang, A. Wolf, and E. Pruss. MRI imaging of the breast: Fast imaging sequences with and without gd-DTPA. *Radiology*, 170(2):95–103, 1989.

[116]J. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.

[117]J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Science, USA*, 79(8):2554–2558, 1982.

[118]J. Hopfield and D. Tank. Computing with neural circuits: A model. *Science*, 233(4764):625–633, 1986.

[119]K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:359–366, 1989.

[120]A. Hyvärinen. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks*, 10(3):626–634, 1999.

[121]A. Hyvärinen and P. Hoyer. Emergence of phase and shift invariant features by decomposition of natural images into independent feature subspaces. *Neural Computation*, 12(7):1705–1720, 2000.

[122]A. Hyvärinen, P. Hoyer, and M. Inki. Topographic independent component analysis. *Neural Computation*, 13(7):1525–1558, 2001.

[123]A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wiley Interscience, 2001.

[124]A. Hyvärinen and E. Oja. A fast fixed-point algorithm for independent component analysis. *Neural Computation*, 9:1483–1492, 1997.

[125]A. Hyvärinen and P. Pajunen. On existence and uniqueness of solutions in nonlinear independent component analysis. *In Proceedings of the 1998 IEEE International Joint Conference on Neural Networks (IJCNN '98)*, vol. 2:1350–1355,

1998.

[126]A. Ilin. Independent dynamics subspace analysis. In *Proc. ESANN 2006*, pages 345–350, 2006.

[127]C. Jutten, J. Hérault, P. Comon, and E. Sorouchiary. Blind separation of sources, parts I, II and III. *Signal Processing*, 24:1–29, 1991.

[128]A. Kagan, Y. Linnik, and C. Rao. *Characterization Problems in Mathematical Statistics*. Wiley, New York, 1973.

[129]S. Karako-Eilon, A. Yeredor, and D. Mendlovic. Blind Source Separation Based on the Fractional Fourier Transform. In *Proc. ICA 2003*, pages 615–620, 2003.

[130]N. Karayiannis. A methodology for constructing fuzzy algorithms for learning vector quantization. *IEEE Transactions on Neural Networks*, 8(3):505–518, 1997.

[131]N. Karayiannis and P. Pai. Fuzzy algorithms for learning vector quantization. *IEEE Transactions on Neural Networks*, 7(5):1196–1211, 1996.

[132]J. Karhunen and S. Malaroiu. Local independent component analysis using clustering. In *Proc. First Int. Workshop on Independent Component Analysis and Blind Signal Separation(ICA99)*, pages 43–49, 1999.

[133]J. Karvanen and F. Theis. Spatial ICA of fMRI data in time windows. In *Proc. MaxEnt 2004*, volume 735 of *AIP Conference Proceedings*, pages 312–319, 2004.

[134]I. Keck, F. Theis, P. Gruber, E. Lang, K. Specht, G. Fink, A. Tomé, and C. Puntonet. Automated clustering of ICA results for fMRI data analysis. In *Proc. CIMED 2005*, pages 211–216, Lisbon, Portugal, 2005.

[135]I. Keck, F. Theis, P. Gruber, E. Lang, K. Specht, and C. Puntonet. 3D spatial analysis of fMRI data on a word perception task. In *Proc. ICA 2004*, volume 3195 of *LNCS*, pages 977–984. Springer, 2004.

[136]G. Kempermann, H. Kuhn, and F. Gage. More hippocampal neurons in adult mice living in an enriched environment. *Nature*, 386(6624):493–495, 1997.

[137]R. Kennan, J. Zhong, and J. Gore. Intravascular susceptibility contrast mechanism in tissues. *Magn. Reson. Med.*, pages 9–21, 6 1994.

[138]D. J. Kim, Y. W. Park, and D. J. Park. A novel validity index for determination of the optimal number of clusters. *IEICE Transactions on Inf. and Syst.*, E84-D(2):281–285, 2001.

[139]T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43(1):59–69, 1982.

[140]T. Kohonen. Self-organized formation of topologically correct feature maps. *Biol. Cybern.*, 43:59–69, 1982.

[141]T. Kohonen. *Self–Organization and Associative Memory*. Springer Verlag, 1988.

[142]T. Kohonen, J. Hynninen, J. Kangas, and J. Laaksonen. Som pak: The self-organizing map program package. *Helsinki University of Technology, Technical Report A31*, 1996.

[143]B. Kosko. Adaptive bidirectional associative memory. *Applied Optics*, 26(9):4947–4960, 1987.

[144]C. Kotropoulos, X. Magnisalis, I. Pitas, and M. Strintzis. Nonlinear ultrasonic image processing based on signal-adaptive filters and self-organizing neural networks. *IEEE Transaction on Image Processing*, 3(1):65–77, 1994.

[145]C. K. Kuhl, P. Mielcareck, S. Klaschik, C. Leutner, E. Wardelmann, J. Gieseke, and H. Schild. Dynamic breast MR imaging: Are signal intensity time course data useful for differential diagnosis of enhancing lesions? *Radiology*,

211(1):101–110, 1999.

[146]H. Kuhn, H. Dickinson-Anson, and F. Gage. Neurogenesis in the dentate gyrus of the adult rat: Age-related decrease of neuronal progenitor proliferation. *J. Neurosci.*, 16(6):2027–2033, 1996.

[147]H. Kuhn, T. Palmer, and E. Fuchs. Adult neurogenesis: A compensatory mechanism for neuronal damage. *Eur. Arch. Psychiatry Clin. Neurosci.*, 251(4):152–158, 2001.

[148]O. Lange, A. Meyer-Baese, M. Hurdal, and S. Foo. A comparison between neural and fuzzy cluster analysis techniques for functional MRI. *Biomedical Signal Processing and Control*, 1(3):243–252, 2006.

[149]N. Lassen and W. Perl. *Tracer Kinetic Methods in Medical Physiology*. Raven Press, New York, 1979.

[150]D. Lee and H. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.

[151]S. Lee and R. M. Kil. A Gaussian Potential Function Network with Hierarchically Self–Organizing Learning. *Neural Networks*, 4(9):207–224, 1991.

[152]T. Lee, M. Girolami, and T. Sejnowski. Independent component analysis using an extended infomax algorithm for mixed sub-Gaussian and super-Gaussian sources. *Neural Computation*, 11:417–441, 1999.

[153]C. Leondes. *Image Processing and Pattern Recognition*. Academic Press, 1998.

[154]A. Levin, A. Zomet, S. Peleg, and Y. Weiss. Seamless Image Stitching in the Gradient Domain. Technical Report 2003-82, Leibniz Center, Hebrew University, Jerusalem, 2003.

[155]J. Lin. Factorizing multivariate function classes. In *Advances in Neural Information Processing Systems*, volume 10, pages 563–569. MIT Press, 1998.

[156]Y. Linde, A. Buzo, and R. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, 28(3):84–95, 1980.

[157]R. Linsker. An application of the principle of maximum information preservation to linear systems. *Advances in Neural Information Processing Systems*, 1, MIT Press, 1989.

[158]R. Linsker. Local synaptic learning rules suffice to maximize mutual information in a linear network. *Neural Computation*, 4:691–702, 1992.

[159]R. P. Lippman. An introduction to computing with neural networks. *IEEE ASSP Magazine*, 4(4):4–22, 1987.

[160]Lo, Leung, and Litva. Separation of a mixture of chaotic signals. In *Proc. Int. Conf. Accustics, Speech and Signal Processing*, pages 1798–1801, 1996.

[161]E. Lucht, S. Delorme, and G. Brix. Neural network-based segmentation of dynamic (MR) mammography images. *Magnetic Resonance Imaging*, 20(8):89–94, 2002.

[162]E. Lucht, M. Knopp, and G. Brix. Classification of signal-time curves from dynamic (MR) mammography by neural networks. *Magnetic Resonance Imaging*, 19(8):51–57, 2001.

[163]D. MacKay. *Information Theory, Inference, and Learning Algorithms*. 6th ed. Cambridge University Press, 2003.

[164]A. Macovski. *Medical Imaging Systems*. Prentice Hall, 1983.

[165]S. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, 1997.

[166]T. Martinetz, S. Berkovich, and K. Schulten. Neural gas network for vector quantization and its application to time-series prediction. *IEEE Transactions on*

Neural Networks, 4(4):558–569, 1993.

[167]W. McCulloch and W. Pitts. A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:115–133, 1943.

[168]M. McKeown, T. Jung, S. Makeig, G. Brown, S. Kindermann, T. Lee, A. Bell, and T. Sejnowski. Spatially independent activity patterns in functional magnetic resonance imaging data during the stroop color-naming task. *Proc. Natl. Acad. Sci. USA*, 95(8):803–810, 1998.

[169]M. McKeown, S. Makeig, G. Brown, T. Jung, S. Kindermann, A. Bell, and T. Sejnowski. Analysis of fMRI data by blind separation into independent spatial components. *Human Brain Mapping*, 6:160–188, 1998.

[170]M. McKeown, S. Makeig, G. Brown, T. Jung, S. Kindermann, A. Bell, and T. Sejnowski. Analysis of fMRI data by blind separation into independent spatial components. *Human Brain Mapping*, 6(8):160–188, 1998.

[171]L. A. Meinel, A. Stolpen, K. Berbaum, L. Fajardo, and J. Reinhardt. Breast MRI lesion classification: Improved performance of human readers with a backpropagation network computer-aided diagnosis (CAD) system. *Journal of Magnetic Resonance Imaging*, 25(1):89–95, 2007.

[172]C. Metz. ROC methodology in radiologic imaging. *Invest. Radiol.*, 21(6):720–733, 1986.

[173]A. Meyer-Bäse. *Pattern Recognition for Medical Imaging*. Elsevier Science/Academic Press, 2003.

[174]A. Meyer-Bäse, F. Theis, O. Lange, and C. Puntonet. Tree-dependent and topographic-independent component analysis for fMRI analysis. In *Proc. ICA 2004*, volume 3195 of *LNCS*, pages 782–789. Springer, 2004.

[175]A. Meyer-Bäse, F. Theis, O. Lange, and A. Wismüller. Clustering of dependent components: A new paradigm for fMRI signal detection. In *Proc. IJCNN 2004*, pages 1947–1952, 2004.

[176]Z. Michalewicz. *Genetic Algorithms*. Springer Verlag, 1995.

[177]T. Mitchell. *Machine Learning*. McGraw Hill, 1997.

[178]L. Molgedey and H. Schuster. Separation of a mixture of independent signals using time-delayed correlations. *Physical Review Letters*, 72(23):3634–3637, 1994.

[179]J. Moody and C. Darken. Fast learning in networks of locally-tuned processing units. *Neural Computation*, 1(2):281–295, 1989.

[180]E. Moreau. A generalization of joint-diagonalization criteria for source separation. *IEEE Transactions on Signal Processing*, 49(3):530–541, 2001.

[181]M. Moseley, Z. Vexler, and H. Asgari. Comparison of Gd- and Dy-chelates for T2* contrast-enhanced imaging. *Magn. Reson. Med.*, 22(6):259–264, 1991.

[182]K.-R. Müller, P. Philips, and A. Ziehe. JADETD: Combining higher-order statistics and temporal information for blind source separation (with noise). In *Proc. of ICA 1999*, pages 87–92, 1999.

[183]T. Nattkemper, H. Ritter, and W. Schubert. A neural classifier enabling high-throughput topological analysis of lymphocytes in tissue sections. *IEEE Trans. ITB*, 5:138–149, 2001.

[184]T. Nattkemper, T. Twellmann, H. Ritter, and W. Schubert. Human vs. machine: Evaluation of fluorescence micrographs. *Computers in Biology and Medicine*, 33:31–43, 2003.

[185]S. Ngan and X. Hu. Analysis of fMRI imaging data using self-organizing mapping with spatial connectivity. *Magn. Reson. Med.*, 41:939–946, 8 1999.

[186]N. Nilsson. *Learning Machines: Foundations of Trainable Pattern-Classifying Systems*. McGraw-Hill, 1965.

[187]S. Ogawa, T. Lee, and B. Barrere. The sensitivity of magnetic resonance image signals of a rat brain to changes in the cerebral venous blood oxygenation activation. *Magn. Reson. Med.*, 29(8):205–210, 1993.

[188]S. Ogawa, T. Lee, A. Kay, and D. Tank. Brain magnetic-resonance-imaging with contrast dependent on blood oxygenation. *Proc. Nat. Acad. Sci. USA*, 87:9868–9872, 1990.

[189]S. Ogawa, D. Tank, R. Menon, and et. al. Intrinsic signal changes accompanying sensory stimulation: Functional brain mapping with magnetic resonance imaging. *Proceedings of the National Academy of Sciences*, 89(8):5951–5955, 1992.

[190]A. Oppenheim and R. Schafer. *Digital Signal Processing*. Prentice Hall, 1975.

[191]S. Osowski, T. Markiewicz, B. Marianska, and L. Moszczyński. Feature generation for the cell image recognition of myelogenous leukemia. In *Proc. EUSICPO 2004*, pages 753–756, 2004.

[192]L. Østergaard, A. Sorensen, K. Kwong, R. Weisskopf, C. Gyldensted, and B. Rosen. High resolution measurement of cerebral blood flow using intravascular tracer bolus passages. Part II: Experimental comparison and preliminary results. *Magnetic Resonance in Medicine*, 36(10):726–736, 1996.

[193]N. Pal, J. Bezdek, and E. Tsao. Generalized clsutering networks and Kohonen's self-organizing scheme. *IEEE Transactions on Neural Networks*, 4(9):549–557, 1993.

[194]S. Pal and S. Mitra. *Neuro–Fuzzy Pattern Recognition*. JWiley, 1999.

[195]A. Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, 1986.

[196]J. Parent, T. Yu, R. Leibowitz, D. Geschwind, R. Sloviter, and D. Lowenstein. Dentate granule cell neurogenesis is increased by seizures and contributes to aberrant network reorganization in the adult rat hippocampus. *J. Neursci.*, 17:3727–3738, 1997.

[197]J. Park and I. Sandberg. Universal approximation using radial-basis-function networks. *Neural Computation*, 3(6):247–257, 1991.

[198]K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 6th ser., 2:559–572, 1901.

[199]S. Peltier, T. Polk, and D. Noll. Detecting low-frequency functional connectivity in fMRI using a self–organizing map (SOM) algorithm. *Human Brain Mapping*, 20(4):220–226, 2003.

[200]H. Penzkofer. *Entwicklung von Methoden zur magnetresonanztomographischen Bestimmung der myokardialen und zerebralen Perfusion*. PhD thesis, LMU Munich, 1998.

[201]N. Petrick, H. Chan, B. Sahiner, M. Helvie, M. Goodsitt, and D. Adler. Computer-aided breast mass detection: False positive reducing using breast tissue composition. *Excerpta Medica*, 1119(6):373–378, 1996.

[202]D.-T. Pham. Joint approximate diagonalization of positive definite matrices. *SIAM Journal on Matrix Anal. and Appl.*, 22(4):1136–1152, 2001.

[203]D.-T. Pham and J.-F. Cardoso. Blind separation of instantaneous mixtures of nonstationary sources. *IEEE Transactions on Signal Processing*, 49(9):1837–1848, 2001.

[204]C. Piccoli. Contrast-enhanced breast MRI: Factors affecting sensitivity and specificity. *European Radiology*, 7(2):281–288, 1997.

[205]E. Pietka, A. Gertych, and K. Witko. Informatics infrastructure of CAD system. *Computerized Medical Imaging and Graphics*, 29:157–169, 10 2005.

[206]J. Platt. A resource-allocating network for function interpolation. *Neural Computation*, 3:213–225, 6 1991.

[207]B. Poczos and A. Lörincz. Independent subspace analysis using k-nearest neighborhood distances. In *Proc. ICANN 2005*, volume 3696 of *LNCS*, pages 163–168. Springer, 2005.

[208]T. Poggio and F. Girosi. Extensions of a theory of networks for approximations and learning: Outliers and negative examples. *Touretky's Connectionist Summer School*, 3(6):750–756, 1990.

[209]T. Poggio and F. Girosi. Networks and the best approximation property. *Biological Cybernetics*, 63(2):169–176, 1990.

[210]T. Poggio and F. Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78(9):1481–1497, 1990.

[211]W. Pratt. *Digital Image Processing*. Wiley, 1978.

[212]F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer, 1988.

[213]W. Press, B. Flannery, S. Teukolsky, and W. Vetterling. *Numerical Recipes in C*. Cambridge University Press, Cambridge, 1992.

[214]P. Pudil, J. Novovicova, and J. Kittler. Floating search methods in feature selection. *Pattern Recognition Letters*, 15(11):1119–1125, 1994.

[215]C. Puntonet, M. Alvarez, A. Prieto, and B. Prieto. Separation of speech signals for nonlinear mixtures. *vol. 1607 (II) of LNCS*, 1607(II):665–673, 1999.

[216]C. Puntonet, C. Bauer, E. Lang, M. Alvarez, and B. Prieto. Adaptive-geometric methods: Application to the separation of EEG signals. *P.Pajunen and J.Karhunen, eds., Independent Component Analysis and Blind Signal Separation (Proc. ICA'2000)*, pages 273–278, 2000.

[217]C. Puntonet and A. Prieto. An adaptive geometrical procedure for blind separation of sources. *Neural Processing Letters*, 2:23–27, 1995.

[218]C. Puntonet and A. Prieto. Neural net approach for blind separation of sources based on geometric properties. *Neurocomputing*, 18:141–164, 1998.

[219]W. Reith, S. Heiland, G. Erb, T. Brenner, M. Forsting, and K. Sartor. Dynamic contrast-enhanced T2*-weighted MRI in patients with cerebrovascular disease. *Neuroradiology*, 30(6):250–257, 1997.

[220]K. Rempp, G. Brix, F. Wenz, C. Becker, F. Gückel, and W. Lorenz. Quantification of regional cerebral blood flow and volume with dynamic susceptibility contrast-enhanced MR imaging. *Radiology*, 193(10):637–641, 1994.

[221]G. Ritter and J. Wilson. *Handbook of Computer Vision Algorithms in Image Algebra*. CRC Press, 1996.

[222]J. Roerdink and A. Meijster. The watershed transform: Definitions, algorithms and parallelization strategies. *Fundamenta Informaticae*, 41(1):187–228, 2001.

[223]B. Rosen, J. Belliveau, J. Vevea, and T. Brady. Perfusion imaging with NMR contrast agents. *Magnetic Resonance in Medicine*, 14(10):249–265, 1990.

[224]D. Rossi and A. Zlotnik. The biology of chemokines and their receptors. *Annu. Rev. Immunol.*, 18:217–242, 2000.

[225]D. L. Ruderman. The statistics of natural images. *Network*, 5:517–548, 1994.

[226]G. Scarth, M. McIntrye, B. Wowk, and R. Samorjai. Detection of novelty in functional imaging using fuzzy clustering. *Proc. SMR 3rd Annu. Meeting*, 95:238–242, 8 1995.

[227]R. Schalkoff. *Pattern Recognition*. Wiley, 1992.

[228]I. Schießl, H. Schöner, M. Stetter, A. Dima, and K. Obermayer. Regularized

second order source separation. In *Proc. ICA 2000*, volume 2, pages 111–116, 2000.

[229]B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, Mass.,, 2002.

[230]H. Schöner, M. Stetter, I. Schießl, J. Mayhew, J. Lund, N. McLoughlin, and K. Obermayer. Application of blind separation of sources to optical recording of brain activity. In *Advances in Neural Information Procession Systems*, volume 12, pages 949–955. MIT Press, 2000.

[231]J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Machine Intell.*, 22(8):888–905, 2000.

[232]W. Siedlecki and J. Sklansky. A note on genetic algorithms for large-scale feature selection. *Pattern Recognition Letters*, 10(11):335–347, 1989.

[233]V. Skitovitch. On a property of the normal distribution. *DAN SSSR*, 89:217–219, 1953.

[234]V. Skitovitch. Linear forms in independent random variables and the normal distribution law. *Izvestiia AN SSSR, ser. matem.*, 18:185–200, 1954.

[235]A. Souloumiac. Blind source detection using second order non-stationarity. In *Proc. Int. Conf. Acoustics, Speech and Signal Processing*, pages 1912–1916, 1995.

[236]K. Specht and J. Reul. Function segregation of the temporal lobes into highly differentiated subsystems for auditory perception: An auditory rapid event-related fMRI-task. *NeuroImage*, 20:1944–1954, 2003.

[237]K. Stadlthanner, A. Tomé, F. Theis, W. Gronwald, H.-R. Kalbitzer, and E. Lang. Blind source separation of water artifacts in NMR spectra using a matrix pencil. In *Proc. ICA 2003*, pages 167–172, 2003.

[238]K. Stadlthanner, A. Tomé, F. Theis, W. Gronwald, H.-R. Kalbitzer, and E. Lang. Removing water artefacts from 2D protein NMR spectra using GEVD with congruent matrix pencils. In *Proc. ISSPA 2003*, volume 2, pages 85–88, 2003.

[239]K. Stadlthanner, A. Tomé, F. Theis, and E. Lang. A generalized eigendecomposition approach using matrix pencils to remove artifacts from 2d NMR spectra. In *Proc. IWANN 2003*, volume 2687 of *LNCS*, pages 575–582. Springer, 2003.

[240]G. Stewart. Researches on the circulation time in organs and on the influences which affect it. *J. Physiol.*, 6:1–89, 1894.

[241]J. Stone, J. Porrill, N. Porter, and I. Wilkinson. Spatiotemporal independent component analysis of event-related fMRI data using skewed probability density functions. *NeuroImage*, 15(2):407–421, 2002.

[242]J. Sychra, P. Bandettini, N. Bhattacharya, and Q. Lin. Synthetic images by subspace transforms I. Principal components images and related filters. *Med. Phys.*, 21(8):193–201, 1994.

[243]M. Tervaniemi and T. van Zuijen. Methodologies of brain research in cognitive musicology. *Journal of New Music Research*, 28(3):200–208, 1999.

[244]F. Theis. Nichtlineare ICA mit Musterabstossung. Master's thesis, Institute of Biophysics, University of Regensburg, Germany, 2000.

[245]F. Theis. *Mathematics in Independent Component Analysis*. Logos Verlag, Berlin, 2002.

[246]F. Theis. A new concept for separability problems in blind source separation. *Neural Computation*, 16:1827–1850, 2004.

[247]F. Theis. Uniqueness of complex and multidimensional independent component analysis. *Signal Processing*, 84(5):951–956, 2004.

[248]F. Theis. Uniqueness of real and complex linear independent component analysis revisited. In *Proc. EUSIPCO 2004*, pages 1705–1708, 2004.

[249]F. Theis. Blind signal separation into groups of dependent signals using joint block diagonalization. *Proc. ISCAS 2005*, pages 5878–5881, 2005.

[250]F. Theis. Multidimensional independent component analysis using characteristic functions. In *Proc. EUSIPCO 2005*, 2005.

[251]F. Theis. Towards a general independent subspace analysis. *Proc. NIPS 2006*, 2007.

[252]F. Theis, C. Bauer, and E. Lang. Comparison of maximum entropy and minimal mutual information in a nonlinear setting. *Signal Processing*, 82:971–980, 2002.

[253]F. Theis, P. Gruber, I. Keck, and E. Lang. A robust model for spatiotemporal dependencies. *Neurocomputing*, 71(10 - 12):2209–2216, 2008.

[254]F. Theis, P. Gruber, I. Keck, A. Meyer-Bäse, and E. Lang. Spatiotemporal blind source separation using double-sided approximate joint diagonalization. *Proc. EUSIPCO 2005*, 2005.

[255]F. Theis, P. Gruber, I. Keck, A. Tomé, and E. Lang. A spatiotemporal second-order algorithm for fMRI data analysis. *Proc. CIMED 2005*, pages 194–201, 2005.

[256]F. Theis, D. Hartl, S. Krauss-Etschmann, and E. Lang. Adaptive signal analysis of immunological data. In *Proc. Int. Conf. Information. Fusion 2003*, pages 1063–1069, 2003.

[257]F. Theis, D. Hartl, S. Krauss-Etschmann, and E. Lang. Neural network signal analysis in immunology. In *Proc. ISSPA 2003*, volume 2, pages 235–238, 2003.

[258]F. Theis and Y. Inouye. On the use of joint diagonalization in blind signal processing. In *Proc. ISCAS 2006*, 2006.

[259]F. Theis, A. Jung, C. Puntonet, and E. Lang. Linear geometric ICA: Fundamentals and algorithms. *Neural Computation*, 15:419–439, 2003.

[260]F. Theis, Z. Kohl, H. Kuhn, H. Stockmeier, and E. Lang. Automated counting of labelled cells in rodent brain section images. *Proc. BioMED 2004*, pages 209–212, 2004.

[261]F. Theis and E. Lang. Maximum entropy and minimal mutual information in a nonlinear model. In *Proc. ICA 2001*, pages 669–674, 2001.

[262]F. Theis, A. Meyer-Bäse, and E. Lang. Second-order blind source separation based on multi-dimensional autocovariances. In *Proc. ICA 2004*, volume 3195 of *LNCS*, pages 726–733. Springer, 2004.

[263]F. Theis and T. Tanaka. A fast and efficient method for compressing fMRI data sets. In *Proc. ICANN 2005, part 2*, volume 3697 of *LNCS*, pages 769–777. Springer, 2005.

[264]S. Theodoridis and K. Koutroumbas. *Pattern Recognition*. Academic Press, 1998.

[265]H. Thompson, C. Starmer, R. Whalen, and D.McIntosh. Indicator transit time considered as a gamma variate. *Circ. Res.*, 14(6):502–515, 1964.

[266]A. Tomé. Blind source separation using a matrix pencil. In *Int. Joint Conf. on Neural Networks (IJCNN)*, Como, Italy, 2000.

[267]A. Tomé. An iterative eigendecomposition approach to blind source separation. In *Proc. 3rd Int. Conf. on Independent Component Analysis and Signal Separation*, pages 424–428, 2001.

[268]A. Tomé and N. Ferreira. On-line source separation of temporally correlated signals. In *Proc. EUSIPCO' 02*, Toulouse, France, 2002.

[269]L. Tong, Y. Inouye, V. Soon, and Y.-F. Huang. Indeterminacy and identifiability of blind identification. *IEEE Trans. on Circuits and Systems*,

38:499–509, 1991.

[270]L. Tong, R.-W. Liu, V. Soon, and Y.-F. Huang. Indeterminacy and identifiability of blind identification. *IEEE Transactions on Circuits and Systems*, 38:499–509, 1991.

[271]G. Torheim, F. Godtliebsen, D. Axelson, K. Kvistad, O. Haraldseth, and P. Rinck. Feature extraction and classification of dynamic contrast-enhanced T2-weighted breast image data. *IEEE Transactions on Medical Imaging*, 20(12):1293–1301, 2001.

[272]M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3:71–86, 1991.

[273]A. van der Veen and A. Paulraj. An analytical constant modulus algorithm. *IEEE Trans. Signal Processing*, 44(5):1–19, 1996.

[274]H. van Praag, A. Schinder, B. Christie, N. Toni, T. Palmer, and F. Gage. Functional neurogenesis in the adult hippocampus. *Nature*, 415(6875):1030–1034, 2002.

[275]A. Villringer, B. Rosen, J. Belliveau, J. Ackerman, R. Lauffer, R. Buxton, Y.-S. Chao, V. Wedeen, and T. B. TJ. Dynamic imaging of lanthanide chelates in normal brain: Changes in signal intensity due to susceptibility effects. *Magn. Reson. Med.*, 6:164–174, 1988.

[276]R. Vollgraf and K. Obermayer. Multi-dimensional ICA to separate correlated sources. In *Proc. Advances in Neural INformation Processing Systems2001*, pages 993–1000. MIT Press, 2001.

[277]C. von der Malsburg. Self-organization of orientation sensitive cells in striata cortex. *Kybernetik 14*, (7):85–100, 1973.

[278]D. Walnut. *An Introduction to Wavelet Analysis*. Birkhäuser, 2002.

[279]P. Wasserman. *Advanced Methods in Neural Computing*. Van Nostrand Reinhold, New York, 1993.

[280]S. Webb. *The Physics of Medical Imaging*. Adam Hilger, 1990.

[281]R. Weisskoff, D. Chesler, J. Boxerman, and B. Rosen. Pitfalls in MR measurement of tissue blood flow with intravascular tracers: Which mean transit time? *Magnetic Resonance in Medicine*, 29(10):553–558, 1993.

[282]D. Whitley. Genetic algorithm tutorial. *Statistics and Computing*, 4(11):65–85, 1994.

[283]N. Wilke, C. Simm, J. Zhang, J. Ellermann, X. Ya, H. Merkle, G. Path, H. Lüdemann, R. Bache, and K. Ugurbil. Contrast-enhanced first-pass myocardial perfusion imaging: Correlation between myocardial blood flow in dogs at rest and during hyperemia. *Magn. Reson. Med.*, 29(6):485–497, 1993.

[284]D. Willshaw and C. von der Malsburg. How patterned neural connections can be set up by self–organization. *Proc. Royal Society London, ser. B*, 194:431–445, 1976.

[285]A. Wismüller, O. Lange, D. Dersch, G. Leinsinger, K. Hahn, B. Pütz, and D. Auer. Cluster analysis of biomedical image time–series. *International Journal on Computer Vision*, 46:102–128, 2 2002.

[286]A. Wismüller, A. Meyer-Bäse, O. Lange, D. Auer, M. Reiser, and D. Sumners. Model-free fMRI analysis based on unsupervised clustering. *Journal of Biomedical Informatics*, 37(9):13–21, 2004.

[287]K. Woods. *Automated Image Analysis Techniques for Digital Mammography*. PhD thesis, University of South Florida, 1994.

[288]R. Woods, S. Cherry, and J. Mazziotta. Rapid automated algorithm for aligning and reslicing PET images. *Journal of Computer Assisted Tomography*,

16:620–633, 8 1992.

[289]H. Yang and S. Amari. A stochastic natural gradient descent algorithm for blind signal separation. In S. S.Usui, Y. Tohkura and E.Wilson, editors, *Proc. IEEE Signal Processing Society Workshop*, Neural Networks for Signal Processing VI, pages 433–442, 1996.

[290]H. Yang and S. Amari. Adaptive on-line learning algorithms for blind separation - maximum entropy and minimum mutual information. *Neural Computation*, 9:1457–1482, 1997.

[291]A. Yeredor. Blind source separation via the second characteristic function. *Signal Processing*, 80(5):897–902, 2000.

[292]A. Yeredor. Non-orthogonal joint diagonalization in the leastsquares sense with application in blind source separation. *IEEE Trans. on Signal Processing*, 50(7):1545–1553, 2002.

[293]E. Yousef, R. Duchesneau, and R. Alfidi. Magnetic resonance imaging of the breast. *Radiology*, 150(2):761–766, 1984.

[294]S. Yu and L. Guan. A CAD system for the automatic detection of clustered microcalcifications in digitized mammogram films. *IEEE Transactions on Medical Imaging*, 19(8):115–126, 2000.

[295]L. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.

[296]A. Ziehe, M. Kawanabe, S. Harmeling, and K.-R. Müller. Blind separation of post-nonlinear mixtures using linearizing transformations and temporal decorrelation. *Journal of Machine Learning Research*, 4:1319–1338, 2003.

[297]A. Ziehe, P. Laskov, K.-R. Müller, and G. Nolte. A linear least-squares algorithm for joint diagonalization. In *Proc. of ICA 2003*, pages 469–474, 2003.

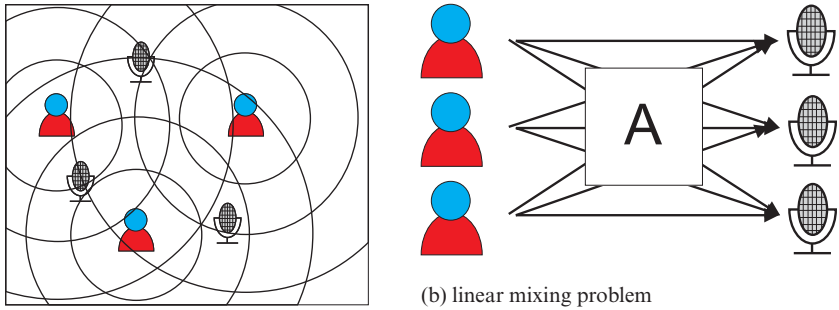
[298]A. Ziehe and K.-R. Müller. TDSEP : An efficient algorithm for blind separation using time structure. In L. Niklasson, M. Bodén, and T. Ziemke, editors, *Proc. of ICANN 98*, pages 675–680. Springer Verlag, Berlin, 1998.

[299]K. Zierler. Theoretical basis of indicator-dilution methods for measuring flow and volume. *Circ. Res.*, 10(6):393–407, 1965.

[300]A. Zijdenbos, B. Dawant, R. Margolin, and A. Palmer. Morphometric analysis of white matter lesions in MR images: Method and validation. *IEEE Transactions on Medical Imaging*, 13(12):716–724, 1994.

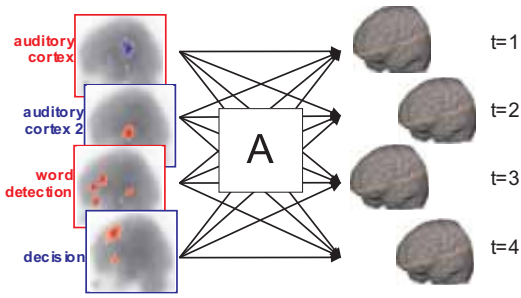
[301]X. Zong, A. Meyer-Bäse, and A. Laine. Multiscale segmentation through a radial basis neural network. *IEEE Int. Conf. on Image Processing*, 3(8):400–403, 1997.

Index



(a) cocktail party problem

(b) linear mixing problem



(c) neural cocktail party

Plate 1
Cocktail party problem. (a) A linear superposition of the speakers is recorded at each microphone. This can be written as the mixing model $\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t)$ equation (4.1) with speaker voices $\mathbf{s}(t)$ and activity $\mathbf{x}(t)$ at the microphones (b). Possible applications lie in neuroscience: given multiple activity recordings of the human brain, the goal is to identify the underlying hidden sources that make up the total activity (c).

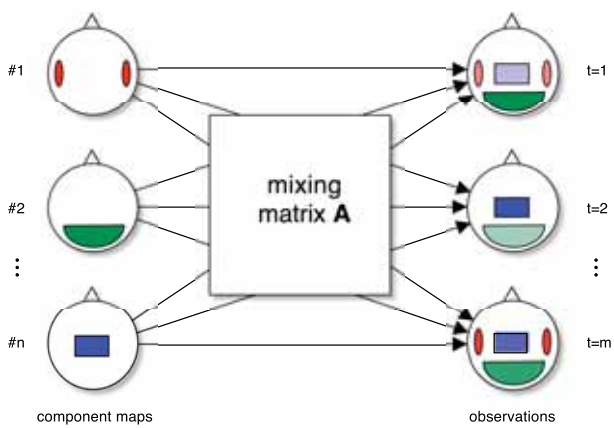
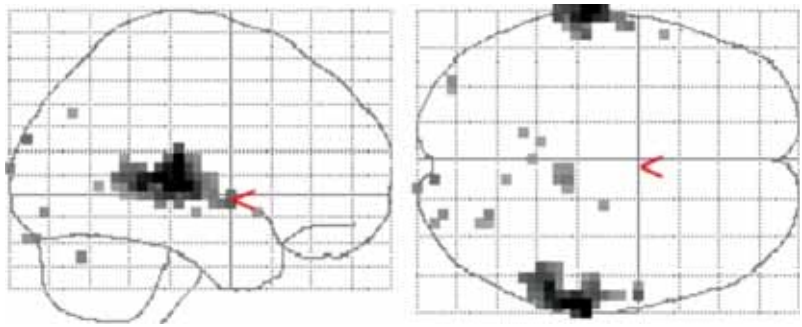
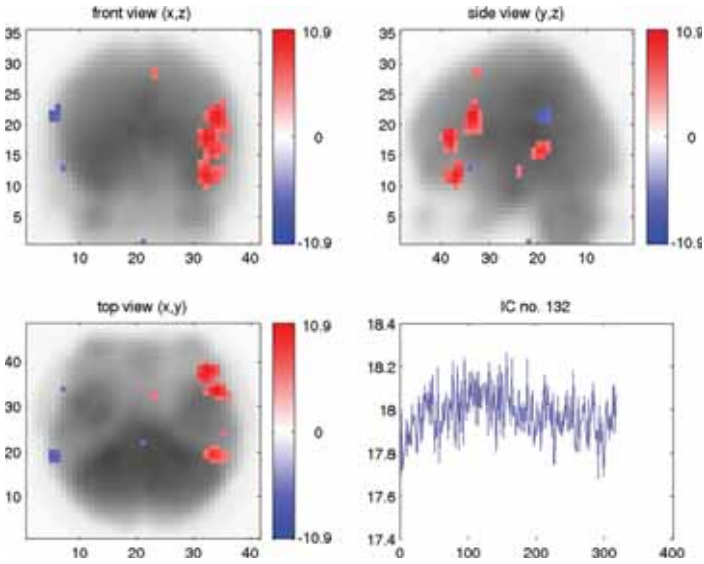


Plate 2
Visualization of the spatial fMRI separation model. The n-dimensional source vector is represented as component maps, which are interpreted as contributing linearly in different concentrations to the fMRI observations at the time points $t \in \{1, \ldots, m\}$.



(a) general linear model analysis



(b) one independent component

Plate 3
Comparison of model-based and model-free analyses of a word-perception fMRI experiment. (a) illustrates the result of a regression-based analysis, which shows activity mostly in the auditory cortex. (b) is a single component extracted by ICA which corresponds to a word-detection network.

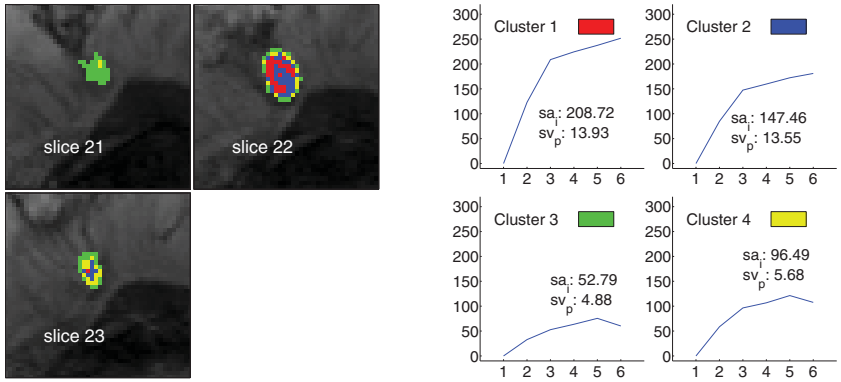


Plate 4
Segmentation method III applied to data set #3 (benign lesion, fibroadenoma), resulting in four clusters. The left image shows the cluster distribution for slices 21 through 23. The right image visualizes the representative time-signal intensity time curves for each cluster.

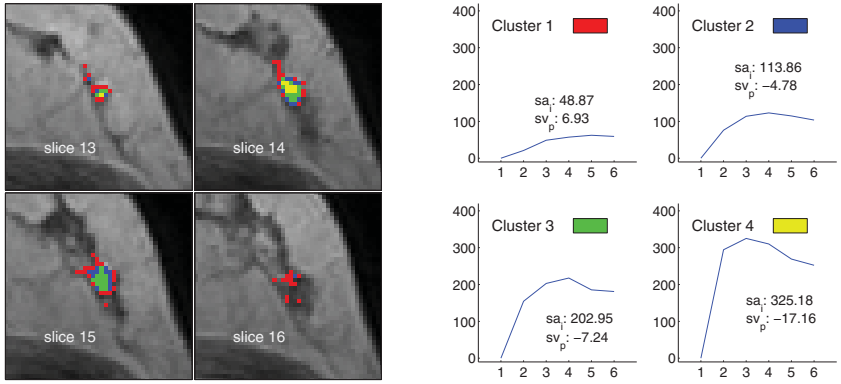


Plate 5
Segmentation method III applied to data set #1 (malignant lesion, tubulo-lobular carcinoma) with four clusters. The left image shows the cluster distribution for slices 13 through 16. The right image visualizes the representative time-signal intensity curves for each cluster.

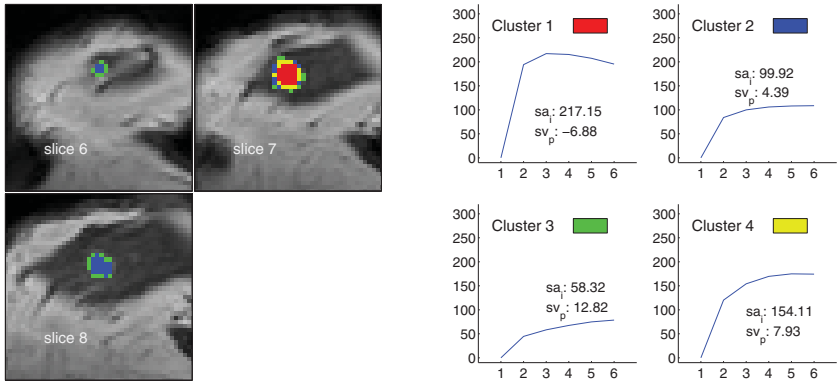


Plate 6
Segmentation method III applied to data set #4 (malignant lesion, ductal carcinoma in situ) and resulting in four clusters. The left image shows the cluster distribution for slices 6 through 8. The right image visualizes the representative time-signal intensity time curve for each cluster.

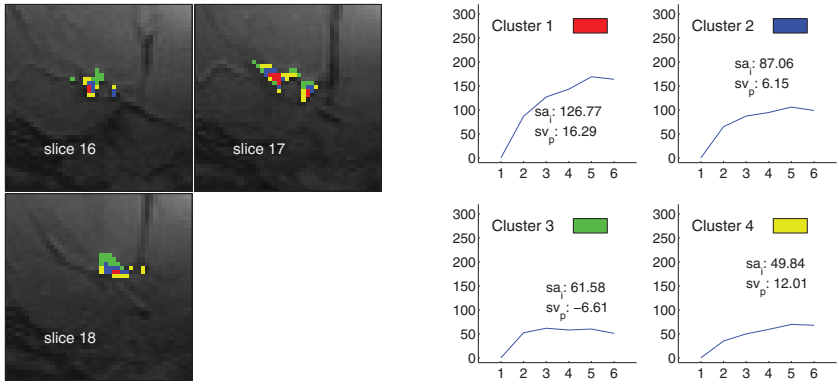


Plate 7
Segmentation method III applied to data set #10 (malignant lesion, ductal carcinoma in situ) with four clusters. The left image shows the cluster distribution for slices 16 through 18. The right image visualizes the representative time-signal intensity curve for each cluster.

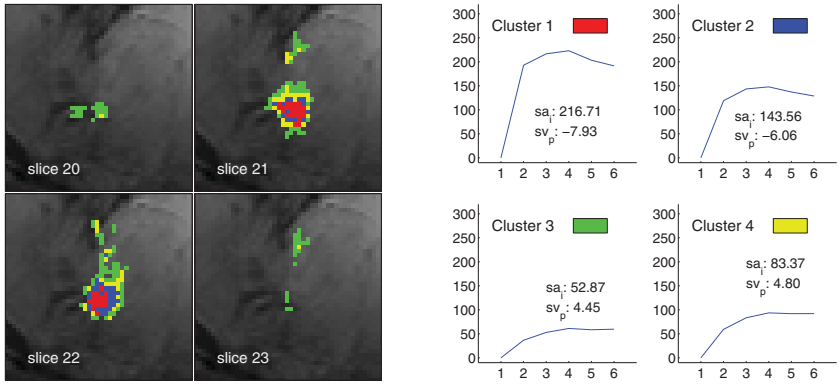


Plate 8
Segmentation method III applied to data set #11 (malignant lesion, invasive ductal carcinoma) with four clusters. The left image shows the cluster distribution for slices 20 through 23. The right image visualizes the representative time-signal intensity curve for each cluster.

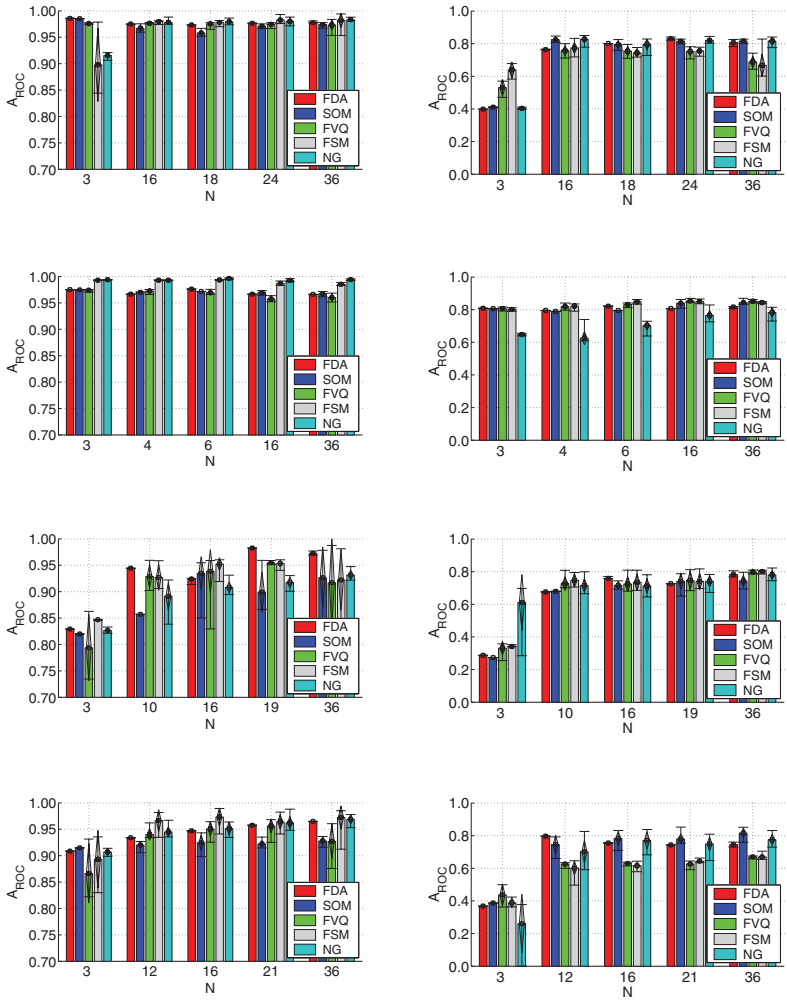


Plate 9
Results of the comparison between the different clustering analysis methods on perfusion MRI data. These methods are Kohonen's map (SOM), the "neural gas" network (NG), fuzzy clustering based on deterministic annealing, fuzzy c-means with unsupervised codebook initialization (FSM), and the fuzzy c-means algorithm (FVQ) with random codebook initialization. The average area under the curve and its deviations are illustrated for 20 different ROC runs using the same parameters but different algorithms' initializations. The number of chosen codebook vectors for all techniques is between 3 and 36, and results are plotted for four subjects. Subjects 1 and 2 had a subacute stroke, while subjects 3 and 4 gave no evidence of cerebrovascular disease. The ROC analysis is based on two performance metrics: regional cerebral blood volume (rCBV) (left column) and mean transit time (MTT) (right column).