STOCHASTIC APPROACHES FOR CORRELATION-BASED LEARNING

By

ZHE CHEN, B.SC., M.S.E.E.

A Thesis

Submitted to the School of Graduate Studies

in Partial Fulfilment of the Requirements

for the Degree

Doctor of Philosophy

McMaster University

Your file Votre référence
ISBN: 0-494-04227-3
Our file Notre référence
ISBN: 0-494-04227-3

Stochastic Approaches for Correlation-Based Learning

DOCTOR OF PHILOSOPHY (2004) McMaster University

(Electrical and Computer Engineering) Hamilton, Ontario

TITLE: Stochastic Approaches for Correlation-Based Learning

AUTHOR: Zhe Chen, B.Sc., M.S.E.E.

SUPERVISOR: Prof. Simon Haykin

NUMBER OF PAGES: xxiv, 170

Stochastic Approaches For Correlation-Based Learning

By **Zhe Chen**

To my parents and the ones I love.

Abstract

The goal of this thesis is to exploit stochastic (noise-driven or Monte Carlo) approaches for generic derivative-free optimization, and to apply these methods to various machine learning problems, including unsupervised learning for perceptual systems and supervised learning for training neural networks.

We review the root of correlation-based learning and introduce a correlation-based gradient-free optimization procedure, which is known as ALOPEX (ALgorithm Of Pattern EXtraction). As a generic optimization framework, the ALOPEX-type algorithms have certain advantageous features: (i) gradient-free; (ii) network architecture independence; (iii) synchronous learning; and (iv) using noise to help escape local minima or maxima. These appealing features make the ALOPEX be a useful tool for many non-convex optimization and machine learning problems. We have successfully applied the ALOPEX-type algorithms for many perceptual learning tasks. We have, for the first time, applied the algorithm to several classic figure-ground segregation perceptual tasks in sensory perception and reported some novel findings. We also pioneer to apply the ALOPEX algorithm to learn a Neurocompensator for hearing compensation as an ingredient of the hearing-aid design.

We have provided a systematical overview of Bayesian estimation, Monte Carlo sampling and optimization. In particular, we have applied sequential Monte Carlo sampling methods, within the Bayesian framework, to both state and parameter estimation problems. In sequential state estimation, we have applied particle filtering, with several proposed improvement schemes, to the tracking problems, including a real-life multiple-input-multiple-output (MIMO) wireless channel estimation problem. In parameter estimation, we have proposed two novel Monte Carlo sampling-based ALOPEX algorithms for optimization and training neural networks. Experiments on various learning tasks, including pattern recognition, on-line financial data prediction, on-line system identification, and chaotic time series prediction, have demonstrated the efficacy and strengths of our proposed algorithms.

In summary, we have addressed a unified philosophical and technical theme in this thesis; we have presented some new theoretical propositions, several novel algorithmic developments, as well as many successful (including some novel) applications.

Abbreviations and Symbols

ABBREVIATIONS

a.k.a.	also known as
ALOPEX	algorithm of pattern extraction
APF	auxiliary particle filter
AR	auto-regressive
ARMA	auto-regressive-moving-average
BIC	Bayesian information criterion
BLAST	Bell laboratories layered space-time
BPTT	back-propagation through time
BSS	blind source separation
CNN	convolution neural network
CT	coordinated turn
DSTBC	differential space-time block codes
e.g.	*exempli gratia*
EKF	extended Kalman filter
EM	expectation-maximization
FA	factor analysis
FFT	fast Fourier transform
FPE	Fokker-Planck equation
HMC	hybrid Monte Carlo
HMM	hidden Markov model
ICA	independent component anlaysis
i.e.	*id est*
i.i.d.	independent, identically distributed
IPS	interacting particle systems
JADE	joint approximate diagonalization of eigen-matrices
KL	Kullback-Leibler (divergence)
LGN	lateral geniculate nucleus
LMS	least-mean-square
MAP	maximum a posteriori
MCMC	Markov chain Monte Carlo

MDL	minimum description length
MIMO	multiple-input, multiple-output
MISO	multiple-input, single-output
MDP	Markov decision process
MKF	mixture Kalman filters
MLE	maximum-likelihood estimate
MLP	multilayer perceptron
MMI	minimum mutual information
MMSE	minimum mean-squared-error
NMSE	normalized mean-squared-error
MSE	mean-squared error
NP	non-deterministic polynomial-time
PCA	principal component analysis
pdf	probability density function
pmf	probability mass function
PoEs	products of experts
PSK	phase-shift keying
OE	output error
RBF	radial basis function
RLS	recursive least-squares
RMLP	recurrent multilayer perceptron
RTRL	real-time recurrent learning
SDE	stochastic differential equation
SER	symbol error rate
SIR	sampling-importance-resampling
SIS	sequential importance sampling
SISO	siingle-input, single-output
SNR	signal-to-noise ratio
SOM	self-organizing map
s.t.	subject to, such that
STBD	space-time block decoder
STFT	short-term Fourier transform
TD	temporal difference
TPF	Turbo-particle filter
UKF	unscented Kalman filter
UPF	unscented particle filter
w.r.t.	with respect to

IMPORTANT SYMBOLS

$\mathcal{F}$	free energy

$\mathcal{H}$	Hamiltonian energy
$\mathcal{K}$	kinetic energy
E	error metric, objective function; potential energy
$\mathbb{E}[\cdot]$	mathematical expectation
f	generic nonlinear function
H	Shannon (differential) entropy
H_2	Renyi quadratic entropy
J	negentropy
K	kernel function; number of mixtures or clusters
ℓ	the number of data (observations)
$\mathcal{L}$	log-likelihood
$\mathcal{L}_{av}$	average log-likelihood
$\mathcal{M}$	model
N_p	number of particles
N_r	number of receivers
N_t	number of transmitters
$\hat{N}_{eff}$	estimated effective particle number
$\hat{N}_{\mathrm{KL}}$	sample impoverishment measure
$\mathcal{O}(\cdot)$	order of
$p(x), q(x)$	probability density function
$\mathcal{P}, \mathcal{Q}$	probability metric space; probability set
$\mathbb{R}$	real number set
T	temperature; maturity time
$\mathbf{s}(t)$	source signal
$\mathbf{x}(t)$	mixed signal
$\mathbf{y}(t)$	demixed signal
$\mathbf{d}$	dynamical noise
$\mathbf{v}$	measurement noise
$\mathbf{x}$	input vector; hidden state vector
$\mathbf{y}$	output vector; observation vector
$\mathbf{A}$	mixing matrix
$\mathbf{H}$	Hessian matrix, input-output mapping
$\mathbf{I}$	identity matrix
$\mathbf{J}$	Jacobian matrix
$\mathbf{R}$	rotation matrix
$\mathbf{X}$	channel matrix
$\mathbf{W}$	demxing matrix
$\mathbf{w}$	synaptic weights
W	importance weight
$\tilde{W}$	normalized importance weight
Z	normalizing constant
$\mathrm{sgn}(\cdot)$	signum function
t, n	continous/discrete time index

tanh	hyperbolic tangent function
$\mathrm{tr}(\mathbf{A})$	trace of matrix $\mathbf{A}$
$\det(\mathbf{A})$	determinant of matrix $\mathbf{A}$
$\mathrm{adj}(\mathbf{A})$	adjoint matrix of matrix $\mathbf{A}$
$cond(\mathbf{A})$	condition number of matrix $\mathbf{A}$
$rank(\mathbf{A})$	rank of matrix $\mathbf{A}$
$\mathrm{diag}\{\}$	diagonal matrix
$\mathrm{Var}[\cdot]$	variance
$\mathrm{Cov}[\cdot]$	covariance
$kurt(\cdot)$	kurtosis
$\lVert\mathbf{x}\rVert$	Euclidean (L_2) norm of vector $\mathbf{x}$
$\lvert\mathbf{x}\rvert_\infty$	L_∞ norm of vector $\mathbf{x}$
$\boldsymbol{A}$	linear operator
$\delta\theta$	infinitesimal change of θ
$\Delta\boldsymbol{\theta}$	change of vector $\boldsymbol{\theta}$
∇	gradient operator
θ	parameter
β	momentum coefficient; AR coefficient
η	learning rate, step-size
γ	step-size
$\boldsymbol{\mu}$	mean vector
$\boldsymbol{\rho}$	momentum vector (in kinetic energy)
λ	forgetting factor; eigenvalue
σ	singular value; standard deviation; regularization coefficient
$\alpha(\mathbf{x},\mathbf{x}')$	probability of move
Σ	covariance matrix
ξ	noise; index parameter
$\mathcal{U}(a,b)$	uniform distribution in region (a,b)
$\mathcal{N}(\boldsymbol{\mu},\Sigma)$	Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance Σ
$\mathbb{C}(\boldsymbol{\mu},\Sigma)$	complex Gaussian distribution with (complex-valued) mean $\boldsymbol{\mu}$ and covariance Σ
$\pi(\cdot)$	invariant probability distribution
$p(\cdot)$	probability density function
$q(\cdot)$	proposal distribution
$D(p\lVert q)$	Kullback-Leibler divergence between distributions p and q
$\mathrm{sgn}(\cdot)$	signum function
$\phi(\cdot)$	logistic sigmoid function
$\Psi(\cdot)$	activation function
$\langle\cdot\rangle$	average
F_s	sampling frequency
^{T}	transpose operator
$\sim$	sample from or distributed according to

Contents

List of Figures

List of Tables

Acknowledgments

First and foremost, I am greatly indebted to my supervisor, Prof. Simon Haykin, for giving me the freedom and support to pursue my research interests; for his confidence and encouragement of my work; for his inspiring discussions and helpful criticism in the past four years. In retrospect, I was greatly influenced by his endless enthusiasm in the pursuit of scientific knowledge, as well as his numerous contributions on variety of academic topics. The privilege working with Simon has been an enjoyable journey during my Ph.D. research at McMaster University.

I would like to express my deep gratitude to Drs. Sue Becker and Thia Kirubarajan for serving my Ph.D. supervision committee. Their insightful discussions and critical comments throughout the supervision are extremely helpful and appreciated. At the initial stage of my research, Sue introduced me the ALOPEX algorithm that builds the foundation of the thesis. I shall also thank Sue's valuable input and guidance throughout several coauthored contributions, for which I was also impressed by Sue's careful thoughts and endless effort in writing or polishing a paper. Dr. Kirubarajan has been very helpful in guiding me to purse independent research and tackle technical problems among his busy schedule. Being always supportive and confident about what I did, Kiruba has been an encouragement source to me in the last two years, for which I am tremendously grateful; I also benefit from taking Kiruba's course and thank him for the valuable support in my difficult times.

To others from whom I have directly or indirectly benefited their knowledge. Dr. Zhi-Quan Luo has served my supervision committee at the early stage and offered some helpful suggestions. I wish to take a chance here to thank Drs. Steven L. Grant and Jacob Benesty, who were the former research scientists in the multimedia communications research lab, Bell Laboratories, Lucent Technologies, for support and many insightful discussions during my summer internship. I really cherished the friendship with many friends previously affiliated with the Adaptive Systems Lab, with whom my life became so colorful. Particularly, I thank Drs. Rembrandt Bakker, Antonio Cezar de Castro Lima, and Gustavo López-Risueño for their generous help and numerous insightful discussions. Gustavo, *muchas gracias* for your hospitality during my fantastic trip in Spain! Cezar, *Obrigado*, I miss the wonderful journey with you in Boston, New York city, and Brazil! I am indebted to Dr. Mark Morelande for many illuminating discussions on particle filters, in particular some ideas related to Turbo-particle filtering and the target tracking with coordinate turn experiment. I am grateful to Dr. Deniz Erdogmus for simulating email discussions on entropy minimization, which gave rise to some results after the ICASSP-Montreal trip. I need to thank Dave Landrigan and Ian Bruce for providing me data and codes used in some experiments. I shall thank Kris Huber for collaboration on the project of wireless channel estimation using particle filtering (reported in Chapter 5), with whom I also had a good time in the swim pool; I also thank Jeff Bondy for collaboration on the project of Neurocompensator (reported in Chapter 4). Nelson Costa has been enormously helpful in teaching me my unknowns on circuits and communications. Many thanks also go to Vai Sellathurai, Kevin Kan, and many former or current members in the Adaptive Systems Laboratory. I also share many delightful moments with other ECE fellow students, in particular, Xuan Wang, Mike Daly, Derek Yee, Amin and Shirin. To the friendship beyond McMaster I shall mention Ji Zhu and Feng Liang, whom I met and had fun in Vancouver and NYC. Specifically, I wish to thank my dear friend and soul mate, Spring Sun, for sharing my joys and griefs whenever possible.

The presentation of part of the thesis has benefited from many valuable feedbacks and discussions from a number of persons, including some anonymous reviewers through the peer review process. In particular, Drs. Tony Bell, Peter Dayan, Geoffrey Hinton, David MacKay, and Terrence Sejnowski, have influenced some of my thoughts through their talks, conversations, and correspondence; Dr. Nando de Freitas offered some assistance in my early investigation on particle filtering; Prof. Stephen Grossberg also provided valuable feedback regarding the cocktail party problem, who is also always an inspiration and personal hero to me. I should also be grateful to the knowledge obtained from the speakers (to name a few, Jont Allen, Shun-ichi Amari, Robert Hecht-Nielson, Michael Jordan, Erkki Oja, Jose Principe, Shihab Shamma) of many workshops and conferences that I attended across Canada and USA.

I would like to thank Cosmin Coroiu, Brian Currie, and Terry Greenlay, for the assistance in the computer facility and others. Special thanks must go to Lola Brooks, Cheryl Gies, and Helen Jachna, for their countless helps and supports throughout my doctoral study.

The last but never the least, I owe my deepest gratitude and appreciation to my family especially my parents, without them I definitely cannot write here and accomplish this thesis. I thank them for their endless and unselfish support, love, patience, and understanding. This dissertation, being the culmination of my more than 20-year student career, is dedicated to them.

The research reported herein was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada. During the Ph.D. program, the author was also supported financially by the Clifton W. Sherman Scholarship.

Chapter 1

Introduction

"I'm a machine and you're a machine, and we both think, don't we?"
— Claude E. Shannon

1.1 Learning

Learning is an essential component of intelligent behavior of human beings in daily life since the day of their births. By *intelligence*, we mean "the capacity to learn or to profit by experience" and "a biological mechanism by which the effects of a complexity of stimuli are brought together and given a somewhat unified effect in behavior" (Pfeifer and Scheier, 1999, Chap. 1). The notion of intelligence is omnipresent in almost every human activity involved, such as perception, action, thinking, memory recall, recognition, and etc.

As an interactive, interdisciplinary subject between cognitive science and artificial intelligence, machine learning is devoted to discovering, exploring, and applying the general principles of intelligence. The main goal of machine learning is to mimic, augment, or outperform (if possible) human learning to perform certain intelligent tasks.

Mathematically, the problem of learning consists of three aspects of endeavor: *estimation*, *approximation*, and *computation.*

Estimation is a statistical problem. Given the observed data, the statistical estimation is to search for the most likely hypothesis within a large class of hypothesis spaces. There might exist several different hypotheses that all yield the same accuracy on the observed data. The learning algorithm is supposed to pick one of them that generalizes well on the out-of-the-sample data; this is often achieved by imposing certain constraints (e.g., smoothness) on the hypothesis space (e.g., functional space), or by pursuing some inductive principles (e.g., minimum description length, Bayesian inference, structural risk minimization). Roughly speaking, estimation is concerned with reducing the *variance* of the estimator.

Approximation is a representation problem. The representation problem arises when the chosen hypothesis space might not contain any hypothesis that yields a good approximation to the true function to be estimated. Then the learning algorithm aims to search a representation using an approximate hypothesis (or a weighted sum of hypotheses). The approximation problem also arises when the complexity of the model is concerned. Roughly speaking, approximation is concerned with reducing the *bias* of the estimator. Choosing a suitable model with a certain level of complexity is essentially finding a good tradeoff for the *bias vs. variance* dilemma.

Computation is an optimization problem. Given the data and the representation model, computation aims to search a set of optimal or suboptimal parameters in the parameter space, subject to certain constraints (e.g., criterion of optimality, computation power or time, etc.). The procedure that leads to the parameter update is called the optimization or learning rule.

In the literature, learning can be categorized into three major types according to the nature of the task: *supervised learning* (learning with teachers), *unsupervised learning* (learning without teachers), and *reinforcement learning* (learning with critics).

Supervised learning can be understood as a multivariate function approximation problem; in the statistical jargon, it amounts to regression for a specific parametric or nonparameteric model.[1]

Unsupervised learning is aimed at learning the structure or regularity of the data (Hinton and Sejnowski, 1999); unsupervised learning exploits the basic information-processing principles (e.g., self-organization, maximum entropy) via either *bottom-up* or *top-down* approaches.

Reinforcement learning can be understood as a Markov decision process (MDP) aimed at learning which actions lead to optimal outcomes; it is aimed to solve a *temporal credit assignment* problem (Sutton and Barto, 1998; Bertsekas and Tsitsiklis, 1996). Rooted in dynamic programming, reinforcement learning has been extended for varieties of prediction and control problems.

The current thesis focuses on supervised and unsupervised learning.

1.1.1 Learning and Inference

Learning can be regarded as a statistical inference problem. Using statistical inference principles (e.g., maximum likelihood, Bayesian inference), learning attempts to characterize the statistical dependence underlying the analyzed data. On the other hand, learning also differs from inference in a number of ways: (i) Inference focuses on finding the most

[1]Classification can be viewed as a special example of regression.

likely explanation of the observed data; but inference itself often ignores (or assumes) the statistical model to be used; it does not take account into the issue of model selection, thereby running into the risk of overfitting, whereas a learning task consists of learning the model as well as learning the data. (ii) In Bayesian inference, inference is pursued by using Bayes' rule by assuming priors, whereas Bayesian learning often superimposes learning the priors as well as other parameters (e.g., regularization parameter, hyperparameter). (iii) For many adaptive systems, learning has to precede the inference procedure. For instance, a complete inference appears possible only when the hidden Markov model (HMM) is first learned by the forward-backward Baum-Welch algorithm, and then followed by the inference conducted through the Viterbi algorithm to obtain a *maximum a posteriori* (MAP) estimate.

1.1.2 Learning and Optimization

Learning can be regarded as an optimization problem. Given a predefined objective function, the common goal of learning is to find an optimization procedure to minimize (or maximize) the objective function.

As far as the optimization problem itself is concerned, optimization can be classified into several categories:

- *constrained* vs. *unconstrained* optimization. In some cases, constrained optimization can be transformed into an unconstrained optimization problem via the Lagrange multiplier.

- *convex* vs. *nonconvex* optimization. Convex optimization has many nice properties such as global optimality; the solution induced from convex optimization is often global or approximately global. Many nonconvex optimization problems are NP-complete or NP-hard, thereby involving the necessity of an approximate solution or Monte Carlo simulation.

- *exact* vs. *approximate* optimization. Exact optimization searches for the exact solution in all the hypothesis space. In contrast, approximate optimization attempts to find a reasonably good solution given certain accuracy or computation time constraints. For instance, bound optimization and variational optimization are typical approximate methods. Many solutions to NP-hard combinatorial optimization problems are approximate.

- *deterministic* vs. *stochastic* optimization. Deterministic optimization treats the parameters to be optimized as deterministic variables, and the underlying system is also deterministic; or, the optimization procedure is just purely deterministic. In contrast, stochastic optimization involves some factors of uncertainty, such as noise or random perturbation; the parameters are often treated as random variables. Bayesian-type optimization strategies also belong to stochastic optimization.

- *gradient-based* vs. *gradient-free* optimization. Gradient-based algorithms are the most popular optimization schemes, which are simply based on the hill-climbing heuristic but do not guarantee to reach the global solution in the non-convex system. Gradient descent/ascent can get stuck in stationary points (local minima/maxima); gradient-type algorithms (e.g., Bertsekas, 1999) can be either first-order (such as the gradient-descent and conjugate gradient), or second-order (such as Levenberg-Marquardt), or approximate second-order (such as the quasi-Newton type). In contrast, gradient-free optimization often involves a gradient approximation and uses it to guide the search direction; examples of this type include line search, simplex algorithm, and Powell's method (Press et al., 1992; Bertsekas, 1999).

1.1.3 Volition and Free Will

In studying the human mind, philosophers have suggested the concepts of *volition* and *free will*. The notion of volition characterizes the deterministic nature of the mind, whereas free will accounts for the *probabilistic* or *stochastic* strategy that is used for decisions. Understanding and reconciling the *volition-and-freewill* paradox is an ever-going research subject.

It should be emphasized here that randomness is ubiquitous in every aspect of human activities. In a macroscopic level, the environment outside each human entity is surrounded by countless random, unpredictable events. In a microscopic level inside the human brain, it is well known that the firing rate of the neurons fluctuates significantly, as a clear evidence of randomness.[2]

In physics, Newton mechanics is deterministic, whereas quantum mechanics introduces uncertainty to the microscopic world; the motion of a particle following the wave equation is *probabilistic*. Physicists have used the *volition-and-freewill* analogy to illustrate many paradoxical physical phenomena.

In neuroscience, the *volition-and-freewill* analogy has also been brought into the discussions of the uncertainty of human brain and behavior (Glimcher, 2003).

In cell biology and genetics, the *volition-and-freewill* analogy is also used to explain the mystery of the genes in DNA.

In economics, economists have attempted to use the *volition-and-freewill* analogy to explain the rationality and uncertainty of the market.

Here, we attempt to abuse such an analogy, and use this metaphor for the learning and optimization problems. To illustrate the optimization problem, let us use an analogy of a hill-climbing task. If a person intends to approach the highest position among the mountains, while the hills are separated by many steep valleys. It is known that the simple,

[2]Note that however, it is not the mean firing rate, but its correlation that are vital to information processing. Neurophysiological studies has evidenced that higher-order correlations and fluctuations play an important role in the nervous systems.

deterministic hill-climbing strategy cannot assure one to reach the zenith; on the other hand, exhaustive random search is theoretically plausible but not realistic. Simply put, deterministic optimization strategy is beneficial in providing a good hint for the search direction; stochastic optimization strategy exploits the randomness to escape from the local valley in a bumpy potential well. Therefore, using a mixed strategy by integrating the features of the deterministic and stochastic optimization, it is likely to yield a better solution.

1.2 Bayesianism

Bayesian theory was originally discovered by Thomas Bayes in a posthumous publication in 1763 (Bayes, 1763). The well-known Bayes theorem describes the fundamental probability law governing the process of logical inference. However, Bayesian theory has not gained its deserved attention in the early days until its modern form was rediscovered by the French mathematician Pierre-Simon de Laplace in *Théorie analytique des probailités.*[3] Bayesian inference (Bernardo and Smith, 1998; Robert, 2001), devoted to applying Bayesian statistics to statistical inference, has become one of the important branches in statistics, and has been applied successfully in statistical decision, detection and estimation, expert systems, machine learning, and even used for understanding the probabilistic functions of the brain (Knills and Richard, 1995; Rao, 1999; Rao et al., 2002). A philosophical question even arises in computational neuroscience: *Do people perform a Bayesian-like thinking in the daily life?* Specifically, the November 19 issue of 1999 SCIENCE magazine has given the Bayesian research boom a four-page special attention. In many scenarios, the solutions gained through Bayesian inference are viewed as "optimal", in that the Bayesian solution combines prior knowledge and takes quantitative account of the uncertainty and evidence. In this thesis, we will focus attention on a sequential Bayesian estimation technique called *particle filtering*, rooted in Bayesian theory and Monte Carlo simulation; specifically, we will apply this powerful tool to various inference (state estimation) and learning (parameter estimation) problems. Quite interestingly, it is noted that there have been recent efforts to suggest particle filtering might serve a functional role in human visual and motor cortices (Lee and Mumford, 2003; Brockwell et al., 2004).

1.3 Roadmap

The two main themes of the thesis are *stochastic correlative learning* and *Monte Carlo optimization and inference.*

The notion of correlation-based learning can be traced back to the Greek philosopher Aristotle. The early formulation of correlative learning related to the brain process was

[3]An interesting history of Thomas Bayes and its famous essay is found in (Dale, 1991).

due to William James (1890; see also (Anderson and Rosenfeld, 1988)). Specifically, it was stated that (James, 1890, Chap. XVI):

> "When two elementary brain-processes have been active together or in immediate succession, one of them, on re-occurring, tends to propagate its excitement into the other."

However, the formal establishment of correlation-based learning was credited to Donald Hebb, whose postulate is thus named as Hebbian learning (Hebb, 1949). Describing a correlative synaptic mechanism, Hebbian learning is a *local* rule, therefore it is physiologically (Stent, 1973) and biologically plausible (see Bi and Poo, 2001, for a review). More specifically, the modification of the synaptic strength depends on the pre- and postsynaptic firing rates and the present value of the synapse. Many variants of correlation-based learning and rate-based learning rules have been developed (Willshaw et al., 1969; Grossberg, 1976; Bienenstock et al., 1982). See (Sejnowski and Tesauro, 1989) for some reviews.

The idea of using Monte Carlo simulation for optimization is not new; genetic algorithms (e.g., Goldberg, 1989) and simulated annealing (Kirkpatrick et al., 1983) are two representative examples. Recently, many sophisticated sampling-based algorithms have been developed for machine learning, such as the Gibbs sampling (Geman and Geman, 1984), Monte Carlo expectation-maximization (EM) algorithm (McLachlan and Krishnan, 1997), dynamic weighting (Wong and Liang, 1997), Fisher scoring (Briegel and Tresp, 1999), and the HySIR algorithm (deFreitas, 1999; deFreitas et al., 2000). See (Andrieu et al., 2003) for an excellent review.

In the thesis, we apply stochastic optimization procedures, rooted in correlation-based learning, to a variety of learning and optimization problems. A particular gradient-free optimization procedure, called ALOPEX (ALgorithm Of Pattern EXtraction), is extensively studied. One of the distinct features of the ALOPEX is to use the noise for optimization. In addition, motivated by sequential Monte Carlo sampling and Bayesian estimation, we develop two novel sequential Monte Carlo sampling-based ALOPEX algorithms for optimization and training neural networks. The idea behind that is to boost the convergence performance of conventional ALOPEX by introducing Monte Carlo simulations. Surrounding the main thesis theme, we also propose several improved schemes for particle filtering and investigate several problems related to machine learning.

1.4 Organization

The thesis contains seven chapters, and the organization is as follows. After the introductory chapter, correlation-based learning paradigms are briefly reviewed in Chapter 2, in which the ALOPEX algorithm and its variants are introduced. In Chapter 3, a stochastic correlative learning rule, as a variant of the ALOPEX, is applied to three classic figure-ground segregation perceptual tasks. In Chapter 4, we describe the perceptual learning procedure for hearing compensation, and apply the developed ALOPEX algorithm for gradient-free optimization. In Chapter 5, we present the underlying theory of sequential Monte Carlo and MCMC methods for Bayesian estimation; we also propose two improved schemes and apply them to two synthetic and one real-life tracking problems within the Bayesian sequential state estimation framework. In Chapter 6, we present two novel sampling-based ALOPEX algorithms, rooted in particle filtering and ALOPEX, for sequential parameter estimation. In particular, we apply the algorithms to train neural networks for a variety of tasks, including pattern recognition, financial data prediction, and system identification. Finally, we summarize and conclude the thesis in Chapter 7.

1.5 Contributions

This dissertation was accomplished by my own intellectual work, the contributed work due to the other researchers or collaborators is acknowledged at the appropriate position throughout the text. The contributions closely related to this dissertation are summarized here:

- Z. Chen, M. Morelande, T. Kirubarajan, and S. Haykin. Improved particle filtering and applications in target tracking and wireless communication. *IEEE Transactions on Signal Processing*, submitted. (Chap. 5)
- Z. Chen, T. Kirubarajan, and M. Morelande (2004). Improved particle filtering schemes for target tracking. To appear in *Proc. ICASSP2005*, Philadelphia, USA. (Chap. 5)
- Z. Chen, S. Becker, J. Bondy, I. Bruce, and S. Haykin (2004). A novel, gradient-free optimization method for model-based hearing compensation. *Neural Computation*, under review. (Chap. 4)
- Z. Chen and S. Haykin (2004). Figure-ground segregation in sensory perception using a stochastic correlative learning rule. to be submitted. (Chap. 3)
- S. Haykin and Z. Chen (2004). The cocktail party problem. *Neural Computation*, under revision. (Chap. 3)
- Z. Chen (2004). Stochastic correlative firing for figure-ground segregation. *Biological Cybernetics*, under revision. (Chap. 3)
- Z. Chen, S. Haykin, and S. Becker (2004). Theory of Monte Carlo sampling-based ALOPEX algorithms for neural networks. *Proc. ICASSP2004*, V-501–504, Montreal, Canada. (Chap. 6)
- S. Haykin, Z. Chen, and S. Becker (2004). Stochastic correlative learning algorithms. *IEEE Transactions on Signal Processing* (Special Issue on Machine Learning Methods in Signal Processing), vol. 52, no. 8, pp. 2200–2209. (Chaps 2, 3, 6)

- S. Haykin, K. Huber, and Z. Chen. (2004). Bayesian sequential state estimation for MIMO wireless communications. *Proceedings of the IEEE* (Special Issue on Sequential State Estimation), vol. 92, no. 3, pp. 439–454. (Chap. 5)
- Z. Chen (2003). Bayesian filtering: From Kalman filters to particle filters, and beyond. Tech. Rep., Adaptive Systems Lab, McMaster University. (Chap. 5)
- Z. Chen (2003). An odyssey of the cocktail party problem. Tech. Rep., Adaptive Systems Lab, McMaster University. (Chap. 3)
- S. Haykin, Z. Chen, and S. Becker (2003). Correlation: Novel basis for statistical learning algorithms. Tech. Rep., Adaptive Systems Lab, McMaster University. (Chap. 2)

Besides the thesis, some other publications during the Ph.D. program also include:

- Z. Chen and A. C. de C. Lima (2004). A new neural equalizer for decision-feedback equalization. *Proc. IEEE Workshop on Machine Learning for Signal Processing*, São Luis, Brazil.
- Z. Chen, S. L. Gay, and S. Haykin (2003). Proportionate adaptation: New paradigms in adaptive filters. In S. Haykin and B. Widrow (Eds.) *Least-Mean-Square Adaptive Filters*, chap. 8, pp. 293–334, New York: Wiley.
- Z. Chen and S. Haykin (2002). On different facets of regularization theory. *Neural Computation*, vol. 14, no. 12, pp. 2791–2846.
- Z. Chen and S. Haykin (2001). A new view of regularization theory. *Proc. IEEE Conference on Systems, Man and Cybernetics*, pp. 1642–1647, Tucson, Arizona, USA.

Chapter 2

Correlation-Based Learning

"A learning machine is any device whose actions are influenced by past experiences."
— Nils Nilsson, *Learning Machines*

Correlation, be it in regards to the autocorrelation function of a prescribed signal or the cross-correlation function between a pair of somewhat similar signals, is basic to statistical signal processing. Its origin, in the context of random processes, may be traced to a series of papers on Brownian motion and spectral analysis of random processes, which began around 1925 and culminated in the publication of the classic paper "Generalized Harmonic Analysis" by Norbert Wiener in 1930, and with it the discipline of statistical signal processing was established (Khinchin, 1933; Wiener, 1930, 1949; Lange, 1967).

Correlation features just as prominently in adaptive filtering. The correction terms in the time-updates of the ubiquitous least-mean-square (LMS) algorithm involve multiplication of the underlying estimate error by tap-input signals in a linear combiner (Widrow and Hoff, 1960; Widrow and Stearns, 1985). In the course of time, the correction terms become proportional to the cross-correlation function between the estimation error and tap-input signals.[1]

Equally interesting is the fact that correlation constitutes a basic mechanism of the human brain. Specifically, the brain explores the sensory environment in a multitude of ways and uses the information so gathered to control behavior. More specifically, correlation is used in the formation of topographic maps, the detection of events in the outside world, and the performing of functions such as learning, association, pattern recognition, and memory recall (Eggermont, 1990). In particular, correlation theory has been applied to model the brain functions (von der Malsburg, 1981), for visual (von der Malsburg, 1981; Harth et al., 1987) and auditory systems (von der Malsburg and Schneider, 1986; Wang and Brown, 1999); and correlated activities are believed to play a critical role in the every timescale in the central nervous system: from the short-term experiences of mem-

[1]Correlation techniques were also dominant in communications and spectral analysis. The autocorrelation or cross-correlation schemes can be developed as feature detectors (Lange, 1967).

ory recall, coincidence detection, novelty detection, perception, learning, to the long-term evolution, all of which reflect the ubiquitous nature of correlation in human intelligent behavior (Cook, 1991).

In what follows, we present a brief survey of correlation-based learning algorithms in the literature.[2] We argue that correlation is an important concept and a mathematical basis in developing statistical learning algorithms.

2.1 The Postulate of Hebbian Learning

Correlation-based theories of learning have a long history in psychology and neurobiology (James, 1890, Chap. XVI). It has been said that correlation is the brain's "... main mechanism to evaluate, to control, and to learn" and "... the basis of learning, association, pattern recognition, and memory recall in the human nervous system" (Eggermont, 1990). Undoubtedly the most influential proponent of learning as correlative process was Donald Hebb, who postulated that (Hebb, 1949, p. 62):

> "When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic changes take place in one or both cells such that A's efficiency as one of the cells firing B, is increased."

Stated mathematically, Hebb's postulate can be formulated as:

$$\Delta\theta_{AB}(t) = \eta x_A(t) y_B(t), \tag{2.1}$$

where x_A and y_B represent the pre- and post-synaptic activities, respectively, between the synapse connecting neurons A and B; $\Delta\theta_{AB}$ denotes the change of synaptic strength; and η is a small step-size parameter. Namely, the change of the synaptic weight $\theta_{AB}(t)$ is proportional to the product of $x_A(t)$ (input) and $y_B(t)$ (output). Averaged over many time steps, the synaptic weight becomes proportional to the correlation between pre- and post-synaptic firing. The important features of Hebbian rule include (Haykin, 1999):

- time-dependence mechanism;
- local mechanism;
- associative mechanism; and
- correlational mechanism, for which the Hebbian synapse is often referred to as a *correlational synapse*.

[2]This part is excerpted and modified from a technical report (Haykin, Chen, and Becker, 2003).

Supported by subsequent electrophysiological data (Bliss and Lomo, 1973), Hebb's profoundly influential idea has not only withstood the test of time in neurobiological circles, but has become the foundation of a wide range of statistical learning theories. A variety of learning algorithms and models, including Oja's rule for maximum eigenfiltering (Oja, 1982), the correlation matrix memory (Willshaw et al., 1969; Anderson, 1969, 1972; Nakano, 1972; Kohonen, 1972; Cooper, 1973), Hopfield network (Hopfield, 1982), and Boltzmann machine (Hinton and Sejnowski, 1986), were indeed motivated by Hebb's postulate of learning.

Note that Hebb's original postulate only allows for an increase in synaptic weight between *synchronously* firing neurons. To prevent unlimited growth, it is necessary to extend Hebb's rule to allow for weight decreases when neurons fire asynchronously. To take care of this matter, Sejnowski (1977a,b) proposed a covariance-based learning rule:

$$\Delta\theta_{ij} = \eta(x_i - \langle x_i \rangle)(y_j - \langle y_j \rangle). \tag{2.2}$$

The rule dictates that when neurons fire synchronously in a correlated manner, their connection strength should increase; whereas when their firing patterns are anti-correlated the weights should decrease. Willshaw and Dayan (1990) showed the optimality of the covariance rule and similar Hebb-like rules for storing patterns in correlation matrix memories. More recent data on spike-time-dependent plasticity (see Bi and Poo, 2001, for a review) has led to computational models that apply a temporally asymmetric time window on Hebbian learning; that is, if a pre-synaptic neuron fires a short time before the post-synaptic neuron, positive Hebbian learning occurs, whereas if the post-synaptic neuron fires a short time before the pre-synaptic neuron, anti-Hebbian learning occurs (see e.g., Gerstner, 2001). This form of learning is, in fact, truer to Hebb's original postulate, because it captures the causal relationship that exists between pre-synaptic and post-synaptic firing.

2.2 Correlation as A Mathematical Basis for Learning

2.2.1 Error-Correcting Learning Rule

A general and powerful form of correlative learning is the error-correcting LMS learning rule (Widrow and Hoff, 1960):

$$\boldsymbol{\theta}(t+1) = \boldsymbol{\theta}(t) + \eta\mathbf{x}(t)e(t), \tag{2.3}$$

where $e(t)$ denotes the estimation error $e(t) = d(t) - \mathbf{x}^T(t)\boldsymbol{\theta}(t)$, and $d(t)$ denotes the desired output signal. According to (2.3), the correction term is proportional to the product of the tap-input vector $\mathbf{x}(t)$ and the estimation error $e(t)$. In the limit as t approaches infinity, the correction term approaches the time-average cross-correlation function between $\mathbf{x}(t)$

and $e(t)$, which, in turn, approaches zero in accordance with the principle of orthogonality, whereupon the weight vector $\boldsymbol{\theta}(t)$ converges to the Wiener solution (Haykin, 2002). Thus, the LMS rule is similar to Hebbian learning, with the correlation between input and output being replaced by the correlation between tap-delay inputs and estimation error. It can also be extended to a recursive least-square (RLS) filter by incorporating the computation of the time-varying second-order correlation matrix of the tap-delay input signals in the learning rule (Haykin, 2002). In addition, although the LMS rule was initially developed for operating with a linear neuron, a generalized "delta rule" of (2.3) can be extended to nonlinear neurons and multilayer networks using back-propagation (Rumelhart et al., 1986).

2.2.2 Competitive Learning Rule

Another example of correlative learning is the competitive learning rule, which has either a supervised or unsupervised form, as respectively given by (e.g., Kohonen, 2001):

$$\boldsymbol{\theta}(t+1) = \boldsymbol{\theta}(t) + \eta[\mathbf{x}(t) - \boldsymbol{\theta}(t)]y(t), \tag{2.4a}$$
$$\boldsymbol{\theta}(t+1) = \boldsymbol{\theta}(t) + \eta[\mathbf{x}(t) - \boldsymbol{\theta}(t)], \tag{2.4b}$$

where where $\mathbf{x}(t)$ represents the input vector that is often normalized, $\|\mathbf{x}(t)\| = 1$, and $y(t)$ denotes the (bipolar) label of the associated input data, $y(t) = \pm 1$. Equation (2.4a) can be decomposed into two terms, the first term $\eta\mathbf{x}(t)y(t)$ being a Hebbian term, and the second term $-\eta\boldsymbol{\theta}(t)y(t)$ being viewed as a weight decay. When $y(t) \equiv 1$, (2.4a) reduces to the unsupervised form (2.4b). Note, however, that the unsupervised rule is only applied to the neuron that wins the competition (i.e., its output activation function equals unity). Hence competitive self-organization can be seen as a Hebbian learning with a decay term that guarantees normalization. This property can be interpreted as a conservation of metabolic resources, thus the sum of synaptic strengths cannot exceed a certain value which is governed by physical characteristics of the cell to support pre- and post- synaptic activities.

Competitive learning can also be used for the formation of topographic maps (Willshaw and von der Malsburg, 1976; Grossberg, 1976; Kohonen, 1984). In a particular form, the self-organizing map is formulated as (Kohonen, 2001):

$$\boldsymbol{\theta}_j(t+1) = \boldsymbol{\theta}_j(t) + \eta h_{j,i(\mathbf{x})}[\mathbf{x}(t) - \boldsymbol{\theta}_j(t)], \tag{2.5}$$

where $h_{j,i(\mathbf{x})}$ is a neighborhood function that defines the "winner-take-all" region; only the winning neuron j is allowed to update (i.e. excitatory), whereas the others are inhibitory.

2.2.3 Maximum Eigenfiltering

Oja (1982) proposed a self-organizing learning rule that keeps the Euclidean norm of a neuron's incoming weight vector (of dimensionality n) as unity. Specifically, the weight feeding the input signal $x_j(t)$ at node j to produce the output signal $y_i(t)$ at node i, is updated using the formula

$$\theta_{ij}(t) = \frac{\theta_{ij}(t-1) + \eta y_j(t) x_i(t)}{\left\{\sum_{k=1}^{n} [\theta_{kj}(t-1) + \eta y_j(t) x_k(t)]^2\right\}^{\frac{1}{2}}}, \tag{2.6}$$

which, for a sufficiently small learning-rate parameter η $(0 < \eta < 1)$, can be approximated by a simpler rule with a decay term:

$$\theta_{ij}(t) = \theta_{ij}(t-1) + \eta y_j(t)[x_i(t) - y_j(t)\theta_{ij}(t-1)]. \tag{2.7}$$

The learning rule (2.7) has the important property that the weight vector converges to the predominant eigenvector of the covariance matrix of the input vector, or in other words, the first principal component of the input distribution. Oja's learning rule has been further extended to multiple output neurons with orthogonal weight vectors, to extract multiple principal components (Oja, 1989; Sanger, 1989).

2.2.4 Boltzmann Learning Rule

Boltzmann machine (Hinton and Sejnowski, 1986; Ackley et al., 1985) uses a *contrastive* Hebbian learning rule, combining a correlative learning term in the "positive" phase (*clamped* state) of learning and an anti-correlative term in the "negative" phase (*free* state):

$$\Delta\theta_{ij} = \eta(\langle y_i \, y_j \rangle_+ - \langle y_i \, y_j \rangle_-), \tag{2.8}$$

where $\langle \cdot \rangle$ denotes sample expectation, averaged over a sample of noisy values. In the positive phase, the activities of the so-called "visible neurons" are fully constrained by the training patterns, whereas in the negative phase, depending on the model, some or all of the visible neurons' states are generated by a constrained sampling procedure.

Further developments in multilayer generative models that can approximate the probability distribution of the data include the Helmholtz machine (Hinton et al., 1995; Dayan et al., 1995) and products of experts (PoEs) (Hinton, 2000).

2.2.5 Temporal Difference Learning

Reinforcement learning (Sutton and Barto, 1998), another basic form of learning, can also be viewed as a correlative learning process. The overall goal is to learn a good approximation of a value function, which indicates the expected sum of future rewards, where a reward received τ steps into the future is often discounted by an exponential factor γ^τ. In temporal-difference (TD) learning, a special form of reinforcement learning, the value prediction error at two successive time steps (so-called TD error) is used to update the value function $V(t)$. The TD learning is analogous to the LMS learning rule (2.3) in that the weights are updated in proportion to the correlation between the input stimuli and the error in predicting a reinforcement signal (Sutton, 1988):

$$\begin{aligned}\boldsymbol{\theta}(t+1) &= \boldsymbol{\theta}(t) + \eta[r(t+1) + \gamma V(\boldsymbol{s}(t+1)) - V(\boldsymbol{s}(t))]\nabla_{\boldsymbol{\theta}} V(\boldsymbol{s}(t)) \\ &= \boldsymbol{\theta}(t) + \eta \mathbf{x}(t)\epsilon(t), \end{aligned} \tag{2.9}$$

where $\mathbf{x}(t) = \nabla_{\boldsymbol{\theta}} V(\boldsymbol{s}(t))$ is the gradient vector of the *linear* value function: $V(\boldsymbol{s}(t)) = \mathbf{x}^T(t)\boldsymbol{\theta}(t)$.[3] The term $\epsilon(t)$ is called the TD error, defined by

$$\epsilon(t) = r(t+1) + \gamma V(t+1) - V(t), \tag{2.10}$$

where $0 < \gamma \leq 1$ is the discount factor, and $r(t)$ denotes the reward. The learning rule (2.9) has the effect that when more reward is received than expected ($\epsilon(t) > 0$), $\boldsymbol{\theta}$ is incremented in proportion to the correlation between unexpected reward (positive TD error) and input states; on the other hand, if less reward is received than expected, $\boldsymbol{\theta}$ is incremented in proportion to the penalty (negative TD error) and input states. The neurobiological plausibility of TD learning as an example of classical conditioning is discussed in (Sutton and Barto, 1998; Dayan and Abbott, 2001). Specifically, it is suggested that the value function $V(t)$ provides a plausible mechanism by which animals may use prediction to optimize the behavior when rewards are delayed (i.e. the so-called *temporal credit assignment problem*), and explains a wide range of psychological and neurobiological data.

2.2.6 Anti-Hebbian Learning Rule

The parameter update rules discussed thus far cover a wide range of learning mechanisms, starting from self-organizing Hebbian learning, going onto supervised error-correction learning, supervised and unsupervised competitive learning, and reinforcement learning. Despite the fact that these learning mechanisms originated from entirely different principles, they all do share the common *correlative* property in one form or another. In a related context, we may identify another class of learning rules that operate by virtue of the *decorrelative* property, examples of which are found in blind source separation and independent component analysis (Bell and Sejnowski, 1995; Hyvärinen et al., 2001). In particular, for blind source separation,[4] the update rule for the demixing matrix, $\mathbf{W}$, in the natural gradient form is described by (Amari et al., 1996):

$$\Delta \mathbf{W}(t+1) = \eta[\mathbf{I} - \boldsymbol{\Psi}(\mathbf{y}(t))\mathbf{y}^T(t)]\mathbf{W}(t), \tag{2.11}$$

where $\boldsymbol{\Psi}(\cdot)$ is a predefined vector-valued activation function, and $\mathbf{I}$ denotes an identity matrix. Equation (2.11) is essentially a decorrelative learning rule. Specifically, the outer

[3]In the classic conditioning experiment of animal learning, $\mathbf{x}(t)$ can be viewed as a vector of binary variables, with each of its components representing the presence or absence of a given stimulus.

[4]We will revisit this problem in detail in Chapter 3.

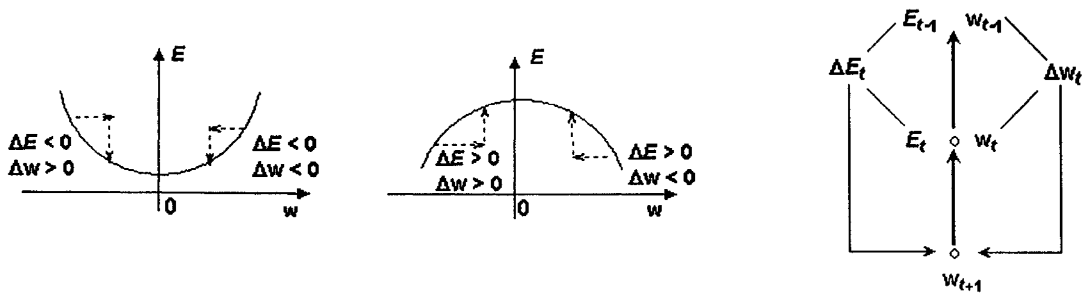

Figure 2.1: Graphical illustration of the correlative learning rule in a one-dimensional domain. *Left:* minimize an objective function. *Middle:* maximize an objective function. Note that the objective function to be minimized/maximized can be also concave/convex. *Right:* the signal-flow graph.

product $\Psi(\mathbf{y}(t))\mathbf{y}^T(t)$ is the cross-correlation matrix between the output signals $\mathbf{y}(t)$ and its nonlinearly transformed version $\Psi(\mathbf{y}(t))$. After a sufficiently large number of the time steps, the correlation matrix approximates the identity matrix, whereupon the incremental change in the demixing matrix $\Delta\mathbf{W}(t+1)$ is reduced to zero and the algorithm converges.

2.3 The ALOPEX Algorithm and Its Variants

2.3.1 Heuristics

Before presenting a rigorous mathematical derivation, we give some heuristic illustration and explanation underlying the development of the ALOPEX procedure.

Without loss of generality, let us consider a one-dimensional example. Suppose that the procedure is to minimize or maximize an objective function $E(w)$, where w is the parameter to be optimized; according to the definition, the gradient of $E(w)$ is given by:

$$\frac{\partial E(w)}{\partial w} = \lim_{\delta w \to 0} \frac{E(w+\delta w) - E(w)}{\delta w} = \lim_{\delta w \to 0} \frac{\delta E}{\delta w} \approx \frac{\Delta E}{\Delta w},$$

where the approximation is valid when Δw is sufficiently small and therefore approximates the infinitesimal perturbation δw.[5] Note that the algebraic sign of gradient remains unchanged if we substitute $\Delta E/\Delta w$ with the product form $\Delta w \Delta E$;[6] the approximation

[5]This is known as the *finite forward-difference* approximation in optimization theory (Fletcher, 2000). Although in principle one can replace the "forward-difference" with "central-difference":

$$\frac{\partial E(w)}{\partial w} \approx \frac{E(w+\delta w) - E(w-\delta w)}{2\delta w} \quad (|\delta w| \to 0)$$

for more appealing approximation accuracy; the "forward-difference" approximation is simpler from the implementation perspective.

[6]Alternatively, $\Delta w \Delta E$ can be replaced by its average value $\langle \Delta w \Delta E \rangle$.

can be easily justified from a geometrical picture illustrated in Figure 2.1. When the unknown parameter is multidimensional (i.e., the scalar w is replaced by a vector $\boldsymbol{\theta}$), purely using $\Delta\boldsymbol{\theta}\Delta E$ as a gradient estimate merely permits a fixed (positive or negative) search direction for each element;[7] in order to circumvent this limitation, we need to introduce the noise to allow multidimensional search. How to control the amount of the noise is the key of the ALOPEX procedure. In the sequel, we will discuss this issue in detail and summarize the appealing features of the algorithm at the end of the chapter.

2.3.2 Mathematical Derivation

By analogy to the correlative form of Hebbian learning, we will derive a simple correlative form of learning rule. We do so by relating an incremental *continuous-time* perturbation in the weight vector, $\delta\boldsymbol{\theta}$, to the correlation between a *discrete-time* change in the weight vector, $\Delta\boldsymbol{\theta}$, and the corresponding incremental continuous-time perturbation in the objective function $\delta E = E(\boldsymbol{\theta} + \Delta\boldsymbol{\theta}) - E(\boldsymbol{\theta})$, defined as (Fujita, 1993):

$$\delta\boldsymbol{\theta} \propto \langle \Delta\boldsymbol{\theta}, \delta E \rangle, \tag{2.12}$$

where the time-average operator $\langle x, y \rangle$ accounts for temporally local correlations between two variables x and y. Moreover, invoking the first-order Taylor series, we may approximate δE due to discrete-time changes in the individual elements of the N-dimensional weight vector $\boldsymbol{\theta}$ as

$$\delta E \approx \sum_{j=1}^{N} \frac{\partial E}{\partial \theta_j}\bigg|_{\boldsymbol{\theta}} \Delta\theta_j.$$

Correspondingly, we may write

$$\langle \Delta\theta_i, \delta E \rangle \approx \sum_{j=1}^{N} \frac{\partial E}{\partial \theta_j}\bigg|_{\boldsymbol{\theta}} \langle \Delta\theta_i, \Delta\theta_j \rangle, \quad i = 1, \cdots, N. \tag{2.13}$$

Assuming that the Euclidean norm $\|\Delta\boldsymbol{\theta}\| \ll 1$ and that individual element changes $\Delta\theta_i$ ($i = 1, \cdots, N$) are independent of each other (in time average), we may approximate the cross-correlation term in the right-hand-side of (2.13) as

$$\langle \Delta\theta_i, \Delta\theta_j \rangle \approx \eta \Delta\theta_i^2 \delta_{ij},$$

where η is a small-valued positive constant, and

$$\delta_{ij} = \begin{cases} 0, & i \neq j \\ 1, & i = j \end{cases}$$

[7]Note that the multiplication of a vector with a scalar does not change the direction of the vector.

is the Kronecker delta. Accordingly, we may further approximate (2.13) as

$$\begin{aligned}\langle \Delta\theta_i, \delta E\rangle &\approx \eta \frac{\partial E}{\partial \theta_i}\bigg|_{\boldsymbol{\theta}} \Delta\theta_i^2 \\ &\approx \eta \Delta E \Delta\theta_i, \quad i = 1, \cdots, N.\end{aligned}$$

In vector form, we thus have the compact relation

$$\Delta\boldsymbol{\theta}(t+1) \propto \eta \Delta\boldsymbol{\theta}(t)\Delta E(t), \tag{2.14}$$

where

$$\Delta\boldsymbol{\theta}(t) = \boldsymbol{\theta}(t) - \boldsymbol{\theta}(t-1), \tag{2.15}$$

$$\Delta E(t) = E(t) - E(t-1). \tag{2.16}$$

Stated in words, the correction in the update formula (2.14) is proportional to the *instantaneous* cross-correlation between the weight modification $\Delta\boldsymbol{\theta}(t)$ in two consecutive time steps and the corresponding objective function change $\Delta E(t)$, where the algebraic sign in the right-hand-side of (2.14) depends on the form of the objective function to be maximized or minimized. The algorithm for weight changes given by (2.14) forms the basis for the ALOPEX algorithm as discussed below, which also incorporates a stochastic decision rule for determining the direction of weight change.

2.3.3 The ALOPEX Algorithm

There exist several forms of the ALOPEX algorithm. A popular form of the algorithm developed for neural network training (Unnikrishnan and Venugopal, 1994) is summarized below:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \eta\boldsymbol{\xi}_t, \tag{2.17}$$

where η is the learning-rate parameter. The vector $\boldsymbol{\xi}_t$ is a random vector with its jth entry determined element-wise by

$$\xi_j(t) = \operatorname{sgn}(u_j - p_j(t)), \quad u_j \sim \mathcal{U}(0,1), \tag{2.18}$$

$$p_j(t) = \phi(c_j(t)/T(t)) = \frac{1}{1+\exp(-c_j(t)/T(t))}, \tag{2.19}$$

$$c_j(t) = \Delta\theta_j(t)\Delta E(t), \tag{2.20}$$

where u is a uniformly distributed random variable; $\operatorname{sgn}(\cdot)$ is the signum function; and $\phi(\cdot)$ is the logistic sigmoid function. The key term is $c_j(t)$, which correlates changes in the cost function with parameter vector changes; it is the scalar version of (2.14). The $T(t)$ is a time-varying annealing parameter that plays a similar role to "temperature" in simulated annealing:

$$T(t) = \eta\frac{\sum_{k=t-T_0}^{t-1}|\Delta E(k)|}{T_0}, \tag{2.21}$$

where $T_0 > 1$ is an integer. The algorithm starts with a randomly initialized parameter $\boldsymbol{\theta}_0$ and stops when the cost function $E(t)$ is sufficiently small. The stochastic component $\boldsymbol{\xi}_t$, being a random force with certain acceptance probability, is included to help the algorithm escape from local minima.

2.3.4 The ALOPEX-B Algorithm

Recently, Bia (2001) has developed a quasi-deterministic version of the ALOPEX algorithm, which he called ALOPEX-B. Unlike the ALOPEX described in (2.17) through (2.21), ALOPEX-B does not employ any annealing scheme and uses fewer parameters, with reported simpler implementation and faster convergence. Consistent with the preceding notations, the ALOPEX-B algorithm proceeds as follows:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \eta\boldsymbol{\xi}_t, \tag{2.22}$$

$$\xi_j(t) = \mathrm{sgn}(u_j - p_j(t)), \quad u_j \sim \mathcal{U}(0,1), \tag{2.23}$$

$$p_j(t) = \phi(C_j(t)), \tag{2.24}$$

$$C_j(t) = \frac{\mathrm{sgn}(\Delta\theta_j(t))\Delta E(t)}{\sum_{k=2}^{t}\lambda(\lambda-1)^{t-k}|\Delta E(k-1)|}, \tag{2.25}$$

where $0 < \lambda < 1$ is a forgetting parameter. An optimal forgetting parameter is often problem-specific, a common value is often chosen within the range $[0.35, 0.7]$.

2.3.5 An Improved Version of the ALOPEX Algorithm

In the course of pragmatic experiments performed on the two versions of the ALOPEX algorithm discussed above, we have formulated an improved version of the ALOPEX algorithm in terms of convergence behavior. As a result of these experiments, we have found that it is more efficient to combine (2.22) and (2.14) in a hybrid learning form, which leads to the modified ALOPEX-B algorithm:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \eta\boldsymbol{\xi}_t + \gamma\Delta\boldsymbol{\theta}_t\Delta E(t), \tag{2.26}$$

where γ is a step-size parameter, $\boldsymbol{\xi}_t$ corresponds to the same stochastic term in (2.22) without invoking the temperature annealing, and $\Delta\boldsymbol{\theta}_t\Delta E(t)$ corresponds to the product term on the right-hand-side of (2.14). The motivation for inclusion of the noise term $\boldsymbol{\xi}_t$ is to introduce a small amount of randomness in the direction of weight change, thereby helping the algorithm escape from local minima. Our extensive experiments have confirmed that (2.26) often converges faster than (2.22) or (2.14) working alone.

The modified ALOPEX-B algorithm has two types of correlation:

- The first kind of correlation takes a form of instantaneous cross-correlation described by the product term $\Delta\boldsymbol{\theta}_t\Delta E(t)$.

- The second kind of correlation appears in the computation of $\boldsymbol{\xi}_t$ as in (2.22) through (2.24), which determines the acceptance probability of random perturbation force $\boldsymbol{\xi}_t$.

We note that when the term $\boldsymbol{\xi}_t$ takes a simplified form of noise, (2.26) reduces to the special form described in (Tzanakou, 2000):

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \eta \Delta\boldsymbol{\theta}_t \Delta E(t) + \boldsymbol{u}_t, \tag{2.27}$$

where $\boldsymbol{u}_t$ is a Gaussian noise vector. The additive noise term $\boldsymbol{u}_t$ differs from $\boldsymbol{\xi}_t$ in that it ignores the correlation information that is used to determine the noise amount in either (2.19) and (2.20) or (2.24) and (2.25).

It is noteworthy that the original ALOPEX algorithm (Harth and Tzanakou, 1974; Tzanakou et al., 1979) assumes that the objective function is to be maximized. If, however, the goal is to minimize $E(t)$, then the algebraic sign of the cross-correlation product term in (2.26) and (2.27) should be changed to the *minus* sign.

2.3.6 Two-Timescale ALOPEX

Lately, Sastry et al. (2002) proposed a two-timescale version of ALOPEX algorithm, which they called 2t-ALOPEX. The key feature of the 2t-ALOPEX is to recursively update the acceptance probability $p_j(t)$:

$$p_j(t) = (1-\lambda)p_j(t-1) + \lambda\zeta_j(t), \tag{2.28}$$

where

$$\zeta_j(t) = \phi\left(\xi_j(t-1)\frac{E(\boldsymbol{\theta}_t) - E(\boldsymbol{\theta}_t - \eta\boldsymbol{\xi}_{t-1})}{\eta T(t)}\right), \tag{2.29}$$

and $\phi(\cdot)$ is the logistic sigmoid function. The motivation of this modification (to the ALOPEX procedure described in Section 2.3.3) is to force the heuristic idea via an approximation of the first-order Taylor series, which is also amenable for theoretical analysis.

2.3.7 Other Types of Correlation Mechanisms

- *Time-averaged correlation*:

$$\begin{aligned} \theta_j(t+1) &= \theta_j(t) + \gamma R_j(t) + u_j, & (2.30)\\ R_j(t) &= \lambda R_j(t-1) + \Delta E(t)\Delta\theta_j(t), & (2.31) \end{aligned}$$

where the instantaneous correlation is substituted by a window-averaged correlation estimate. Note that by this change, the current parameter is influenced by the errors in previous steps (i.e., penalizing previous trajectories), the learning rule is forced to search a locally smooth solution in the parameter space.

- *Inverse correlation*:

$$\theta_j(t+1) = \theta_j(t) + \gamma \Delta E(t)/\Delta\theta_j(t) + u_j, \tag{2.32}$$

where the instantaneous value $\Delta E(t)/\Delta\theta_j(t)$ replaces its local (time or spatial) average. The inverse correlation, however, has the disadvantage that the crosstalk noise increases as $\Delta\theta_j(t)$ becomes small in comparison with $\Delta E(t)$, since $\Delta E(t)$ might include the change caused by other $\theta_k(t)$, $k \neq j$ (Fujita, 1993). In addition, the inverse correlation often suffers from a poor numerical problem in practice: if $\Delta\theta_j(t)$ is very small, it can cause an overflow problem in computer simulations.

- *Gain-and-loss discriminated correlation* (Fujita, 1993):

$$\theta_j(t+1) = \begin{cases} \theta_j(t) + \gamma\Delta E(t)\Delta\theta_j(t) + u_j, & \text{if } \Delta E(t) < 0 \\ \theta_j(t) + \gamma\Delta E(t)/\Delta\theta_j(t) + u_j, & \text{if } \Delta E(t) > 0 \end{cases} \tag{2.33}$$

which is a form of either *gain-emphasized* correlation (when $\Delta E(t) < 0$) or *loss-suppressed* correlation (when $\Delta E(t) > 0$): When $\Delta\theta_j$ gives rise to a gain ($\Delta E(t) < 0$), $\Delta E(t)$ is multiplied by $\Delta\theta_j(t)$, the gain is further used to bring in a bigger change of θ_j, thus a lower potential of E at a farther point is an attractor; when $\Delta\theta_j$ results in a loss ($\Delta E(t) > 0$), $\Delta E(t)$ is divided by $\Delta\theta_j(t)$, the loss moves θ_j according to the local gradient. The motivation of such discriminated correlations is to change the parameters via the *attractive force* of the global minimum and the *repulsive force* of the local gradient (Fujita, 1993).

2.4 Summary

Thus far, we have discussed several different versions of the ALOPEX. Although distinct from each other, they do share many common attractive features, as summarized below:

- The ALOPEX optimization procedure is gradient-free, and is independent of the objective function and network (model) architecture.
- The optimization is synchronous in the sense that all parameters are updated in parallel, thereby sharing the features of algorithmic simplicity and ease of hardware (parallel) implementation.
- The optimization relies on noise, which is used to control the search direction and escape from the local minimum or maximum.
- The basic principle of ALOPEX is a *trial-and-error* process; it is similar in spirit to the "weight perturbation" method (a.k.a. "MIT rule") in the control literature.
- Synaptic plasticity introduces a feedback mechanism; it is temporally asymmetric and characterizes a reinforcement-reward like casuality, which establishes itself as a similar form of the temporal-difference (TD) learning.

Chapter 3

Perceptual Learning for Figure-Ground Segregation in Sensory Perception

"What we see is the solution to a computational problem, our brains compute the most likely causes from the photon absorptions within our eyes."

— H. Helmholtz

3.1 Perception: Learning Cyclopean Sensory Data

Learning cyclopean sensory information provides a human the main sources of the surrounding real-life world. Among many, segregation and streaming are important tasks for sensory perception in visual and auditory systems. How to understand and imitate the brain's amazing capability to segregate a perceptual object of interest in a complex visual or auditory scene, nevertheless, remains a major challenge. Sensory perception is often viewed as an unsupervised learning problem in computational neuroscience (Harth, 1976; Harth et al., 1986; Hinton and Sejnowski, 1983). Biologically speaking, the process of synapse adaptation is called learning, and the procedure for adjusting the synapse is referred to as the learning rule. In mathematical terms, learning is an optimization process in that we need to design a procedure to find an optimal solution that achieves a minimum or maximum value of a specified objective function. Bearing in mind the human's remarkable perception ability, we may infer that perception should not cast itself as an NP-hard optimization problem. To understand how the brain efficiently solves such optimization problems is of theoretical and technical importance. This conundrum, despite many efforts, is still profoundly mysterious and elusive. It is our firm belief that the candidate learning rule to the solutions of the self-organized perceptual tasks should be simple, robust, universal, biologically plausible, and amenable for parallel and distributed computation, as we have observed from its biological counterpart in the brain.

In a neurobiological context, the idea of correlation-based learning was formally established by Hebb (1949). As reviewed in Chapter 1, Hebbian-like learning is temporally local since it only depends on pre- and postsynaptic firing rates and present state of the synapse (namely, the information that is locally available at the location of the synapse). Correlative learning can be viewed as a special case of the Hebbian rule, depending on different interpretations of pre- and postsynaptic activities. In the computational neuroscience and connnectionist literature, correlation theory has found its great successes in associative memory (see e.g., Anderson, 1972; Kohonen, 1972; Hopfield, 1982), vision (see e.g., von der Malsburg, 1981; Miller, 1990, 1998), audition (Eggermont, 1993), novelty detection and learning (Eggermont, 1990; Singer, 1993; Arbib, 2003), and sensory segmentation (see e.g., von der Malsburg and Schneider, 1986; von der Malsburg, 1999; König and Engel, 1995). It has also been argued (Cook, 1991) that the correlated activities play a significant role in the central nervous system on every time scale.

In this chapter (and the chapter that follows), we will study perceptual learning in visual and auditory systems, using the developed stochastic correlative learning algorithm. Specifically, in the current chapter, we look at an important perceptual phenomenon, the so-called figure-ground segregation; experimental results on various perceptual tasks and discussions will be presented in detail.

3.2 Figure-Ground Segregation

The perception goal in figure-ground segregation is to discriminate a "figure" (the object of interest) from a "ground" (the background), given some sensory stimuli (i.e. the auditory or visual scene). The "figure" and "ground" are relative concepts in a sense that their positions can be switched by selective attention. In certain scenarios, "figure" is obvious and it often pops out from the background; sometimes, "figure" and "ground" are ambiguous and bistable (for instance, a famous example is Rubin's *vase-face* illusion). Numerous psychophysical effects, such as ***boundary, size, orientation, contrast, symmetry, convexity***, and *meaningfulness*, influence the figure-ground organization and perception. In the literature many computational models have been proposed for various figure-ground segregation tasks (e.g., Kienker et al., 1986; Sejnowski and Hinton, 1987; Sporns et al., 1991; Grossberg and Wyse, 1991, 1992; Hérault and Horaud, 1993). Here, we have focused the attention on stereo observations in sensory perception tasks. Note that, however, figure-ground segregation is by no means limited by binocular or binaural observations; instead, monocular or monaural input is sufficient for many figure-ground discrimination tasks.

In the subsequent chapter, we illustrate that a simple stochastic correlative learning rule, as a variant form of the ALOPEX (Harth and Tzanakou, 1974; Tzanakou et al., 1979), is sufficient to accomplish several classic figure-ground segregation and perceptual vision tasks. Though not being its debut, we do demonstrate, for the first time, some novel applications of this learning rule for solving several unsupervised learning problems. The effort for attacking these problems is not the first trial, but we present some fresh thinking regarding individual perceptual problems and produce the solutions in a rather different way (compared to available computational modeling approaches).

3.2.1 Stochastic Correlative Learning Rule

Let $\mathbf{w}$ denote a vector of synaptic weights or some unknown parameters, and let $E(\mathbf{w})$ (abbreviated as E from now on) denote a predefined objective function related to $\mathbf{w}$. The proposed stochastic correlative learning rule, as a variant of the ALOPEX discussed in Chapter 2, is rewritten here:

$$\mathbf{w}_{t+1} = \mathbf{w}_t \pm \eta \Delta \mathbf{w}_t \Delta E_t + \gamma \mathbf{r}_t, \tag{3.1}$$

where $\Delta \mathbf{w}_t = \mathbf{w}_t - \mathbf{w}_{t-1}$, $\Delta E_t = E_t - E_{t-1}$, $\mathbf{r}$ is an additive random noise vector,[1] η is a learning rate scalar, and γ is another scalar that controls the amount of additive noise. Equation (3.1) states that the current synaptic adaptation $\Delta \mathbf{w}_{t+1}$ depends on the *instantaneous* cross-correlation term $\Delta \mathbf{w}_t \Delta E_t$, a product of previous synaptic modulation and the resulting performance error change. The algebraic (positive or negative) sign depends on the form of the objective function to be maximized or minimized.[2] The main role of the noise is to introduce randomness into the optimization process to help to escape local maxima or minima. Sharing with many other stochastic optimization procedures such as stochastic relaxation (Ballard et al., 1983) and simulated annealing (Kirkpatrick et al., 1983; Geman and Geman, 1984), an interesting aspect of this correlative learning rule is that it relies heavily on the effect of noise. We have observed that the influence of the noise is critical to the optimization process in all of tasks reported later. We regard this effect as a putative role for high degree of stochastic variability in the firing pattern of neurons in the sensory cortices (Sejnowski and Hinton, 1987). Equation (3.1) characterizes a temporally asymmetric synaptic plasticity and a reinforcement-reward like causality (in the sense that the action $\Delta \mathbf{w}$ yields either a reward or penalty measured by ΔE). In fact, it establishes a *trial-and-error* relationship. In addition, the freedom of designing an objective function E allows much space to incorporate the higher-order correlation between stimulus and the unknown parameters via the cross-correlation term $\Delta \mathbf{w}_t \Delta E_t$. Another intriguing feature of (3.1) is that it is gradient-free and model-independent, thus being potentially applicable for various postulated neural architectures that have modular, hierarchical, or feedback structure. This is even more appealing when it is expensive or impossible to calculate the exact gradient of the objective function (especially those involving second- and higher-order statistics), or the objective function is non-differentiable (due to some discontinuity) or non-convex. Besides, the learning rule (3.1) inherently incorporates a feedback mechanism (recalling Figure 2.1). From a practical implementation

[1]In our experiments reported here it is chosen to be uniformly distributed; however, it can be specifically designed to accelerate the optimization process, as discussed earlier in Chapter 2.

[2]Mathematically, minimizing an objective function E is functionally equivalent to maximizing its negative value, $-E$.

viewpoint, the simplicity of (3.1) readily lends itself to parallel optimization and possible VLSI implementation.

Here, E is a global function with respect to the weight vector $\mathbf{w}$. The synaptic plasticity with a global function might seem biologically implausible (Becker, 1995).[3] However, E can also be regarded as a local E contributed partially by the local $\mathbf{w}$. For instance, E can be measured by the number of firing spikes of the cells; or E can be a certain reward signal released by chemical substance that is induced by a top-down projection of cognition message (either success or failure). In general, synapses at different locations of the cortex can be associated with different costs, and the learning rule is readily extended to multiple objective functions case. Equation (3.1) characterizes a synaptic plasticity of "*think globally, act locally and synchronously*".

It is noteworthy to emphasize the causal synaptic plasticity of (3.1) induced by a Hebbian term. The correlative mechanism reinforces individual pre- and postsynaptic activities to strengthen or attenuate the synapses. Another way to view it is to think the neuron as a *coincidence detector*: a combination of presynaptic events (Δw_i) leads to a common postsynaptic action potential (ΔE), namely, coincident spiking. This perspective is in contrast to the conventional view of an *integrate-and-fire* neuron, as noticed by a number of researchers (Singer, 1993; Shadlen and Newsome, 1994; König and Engel, 1995; König et al., 1996).

3.2.2 Pattern Extraction

Pattern extraction is an important task in unsupervised learning. ALOPEX (ALgorithm Of Pattern EXtraction), as its name suggests, is a procedure designed for (visual) pattern extraction (Harth and Tzanakou, 1974; Tzanakou et al., 1979). In perceptual learning, a central task is to extract the interesting (often non-Gaussian) features behind the high-dimensional sensory data. In its original appearance, ALOPEX was developed for visual research, in which the *response feedback* was used to construct visual patterns that optimize the neurons' responses. The underlying assumption in the ALOPEX process is that, apart from noise fluctuations, the response of a neuron in the visual pathway increases as the stimulus approaches some optimal pattern, or the so-called *receptive field*.[4] It was also suggested (Harth and Tzanakou, 1974) that in principle, any visual event or sequence of events displayed on the retina may be the receptive field of a neuron (or population of neurons), and that such neurons function as *detectors* of the specific sensory *trigger features* defined by their receptive fields. In particular, when the detectors' generated patterns (starting with a random pattern) match the receptive field (i.e., they are highly correlated), the cell is likely to produce a high response (i.e., high firing rate). The ALOPEX process takes the feedback of the neurons' responses (monitored by mi-

[3]It is often difficult, but not impossible, to transform a learning rule driven by a global objective function to a purely local learning rule (see Linsker, 1997; Xie and Seung, 2003, for special examples).

[4]In (Harth and Tzanakou, 1974), the authors defined the receptive field as that spatio-temporal stimulus pattern, which maximally affects the firing rate of a given neuron.

croelectrodes) and further optimizes its produced patterns, until the correlations between the ALOPEX patterns and the receptive field patterns are very high; by then a coincident detection is accomplished. Such a trial-and-error stimulus pattern-matching process essentially reflects the key idea of *feedback* in cybernetics (Wiener, 1948). The concept of the coincident/correlation detector is also widely used in communications and auditory perception, as reviewed in (Lange, 1967).

In addition, being a generic stochastic optimization procedure, it is not surprising that the ALOPEX has also found numerous *unique* applications in modeling perceptual systems (e.g., Tzanakou, 2000; Anderson and Tzanakou, 2002). As a matter of fact, the ALOPEX process (used as an optimization procedure) was also suggested to play a critical role in the thalamus via the thalamocortical (feedforward) and corticothalamic (feedback) loops (Harth et al., 1987; Mumford, 1995).

3.2.3 Objective Function

If we view perception as an unsupervised learning process, designing a suitable objective function (that is often a global measure) is deemed to be crucial to the perception. Information theory provides a good guide for sensory perception. In the literature, information-theoretic criteria have been suggested for various kinds of unsupervised learning (e.g., Barlow, 1989; Linsker, 1990; Becker and Hinton, 1992; Atick, 1992; Intrator, 1992; Blais et al., 1998). Among many, independence component analysis, or ICA (see e.g., Bell and Sejnowski, 1995; Haykin, 2000; Hyvärinen et al., 2001), is an important class of unsupervised learning for source separation and feature extraction. In a conventional batch learning setup, let $\mathbf{S}$ denote a source matrix that contains the original signals, in which each source is represented by a row vector. The sensors receive the signals subject to a linear mixing process described by $\mathbf{X} = \mathbf{AS}$, where $\mathbf{A}$ is called the mixing matrix (usually, $\mathbf{A}$ is a square matrix with a full rank). The recovered signals are written in a matrix form $\mathbf{Y} = \mathbf{WX}$, where $\mathbf{W}$ is called the demixing matrix. The goal of ICA is to separate out the sources, given one or more of the ensuing assumptions about the sources: (i) non-Gaussian; (ii) statistically independent or uncorrelated; and (iii) non-stationary. In terms of objective function, there are many criteria for ICA, such as the maximum entropy (Infomax), maximum likelihood, maximum a posteriori, and higher-order cumulant. Most of them attempt to search for the extreme of the contrast function (e.g., kurtosis); see (Hyvärinen et al., 2001) for more discussion. We briefly mention a few of them below.

Kurtosis statistics. Assuming that the source signals are non-Gaussian (leptokurtic or platykurtic), a common measure for non-Gaussianity is the minimum or maximum kurtosis criterion (Hyvärinen et al., 2001). The kurtosis function, being a higher-order statistic, has previously been used for feature extraction (e.g., Blais et al., 1998). In the literature, the kurtosis-based ICA approaches include the kurtosis-based gradient adaptation rule (Girolami and Fyfe, 1997; Hyvärinen et al., 2001):

$$\begin{aligned}\Delta \mathbf{w} &\propto \operatorname{sign}\big(kurt(\mathbf{w}^T\mathbf{z})\big)\big\langle \mathbf{z}(\mathbf{w}^T\mathbf{z})^3\big\rangle,\\ \mathbf{w} &\leftarrow \mathbf{w}/\|\mathbf{w}\|,\end{aligned}$$

where $\mathbf{z}$ is the whitened data from $\mathbf{x}$, and *kurt* represents a kurtosis function (a.k.a. the fourth-order cumulant) defined as

$$kurt(z) = \langle z^4 \rangle - 3(\langle z^2 \rangle)^2$$

with $\langle \cdot \rangle$ denoting the sample average; random variable z is assumed to be zero-mean. One can also derive the kurtosis-based fixed-point FastICA algorithm (Hyvärinen and Oja, 1997; Hyvärinen et al., 2001).

In the literature, it has been suggested to use the *normalized* kurtosis function for source separation or feature extraction (Girolami and Fyfe, 1997; Blais et al., 1998):

$$kurt(z) = \frac{\langle z^4 \rangle}{(\langle z^2 \rangle)^2} - 3.$$

Specifically, we can maximize the absolute value of the kurtosis of the output signals by assuming that the recovered "figure" has a nonzero-mean non-Gaussian feature:[5]

$$E = \left| \sum_{i=1}^{n} \frac{\langle (\mathbf{y}_i - \langle \mathbf{y}_i \rangle)^4 \rangle}{(\langle \mathbf{y}_i^2 \rangle - \langle \mathbf{y}_i \rangle^2)^2} - 3 \right|, \tag{3.2}$$

where $\mathbf{y}_i$ denotes the ith output vector, and n denotes the number of sensors. Note that in using the proposed learning rule (3.1), we make no assumption regarding the distributions of the sources; in general, the distributions can be super- or sub-Gaussian or a mixture of both.[6]

Negentropy. Assuming that the sources are non-Gaussian, a natural objective criterion is to maximize the discrepancy of transformed outputs from the normality (Comon, 1994). In particular, we can maximize the sum of marginal negentropies:

$$E = \sum_{i=1}^{n} J(\mathbf{y}_i), \tag{3.3}$$

[5]Exploration projection pursuit theory (Freidman, 1987) states that search for the interesting structure in data space can be achieved by searching for the deviation from the Gaussian distribution in the projected space. Here, we assume that the kurtosis of Gaussian, super-Gaussian, and sub-Gaussian signals are equal, greater, and smaller than 3, respectively.

[6]Also note that optimizing equation (3.2) has a tendency to produce duplicate sources at the outputs, unless an extra (independent or uncorrelated) constraint is imposed between the different outputs $\mathbf{y}_i$ and $\mathbf{y}_j$ $(i \neq j)$.

where $J(\mathbf{y}_i)$ represents the negentropy of the ith output vector $\mathbf{y}_i$; the negentropy function can be roughly approximated by (Hyvärinen et al., 2001):

$$J(\mathbf{y}_i) \approx \frac{1}{12}\langle \mathbf{y}_i^3 \rangle^2 + \frac{1}{48} kurt(\mathbf{y}_i)^2. \tag{3.4}$$

Note that the second-term of the right-hand-side of (3.4) consists of the squared kurtosis, thus maximizing the sum of marginal negentropies (3.3) has a similar effect as maximizing the absolute value of the kurtosis function as in (3.2).

Mutual information. Assuming that the source signals have zero mean and are statistically independent, one can also use the minimum mutual information (MMI) criterion (Comon, 1994; Amari et al., 1996; Yang and Amari, 1997) to minimize the dependency among the outputs. This is equivalent to minimizing the Kullback-Leibler (KL) divergence between the joint and the product of the marginal distributions of the output components:

$$\begin{aligned} D(\mathbf{W};\mathbf{y}) &= \int p(\mathbf{y}) \log \frac{p(\mathbf{y})}{\prod_{i=1}^n p_i(y_i)} d\mathbf{y} \\ &= -H(\mathbf{y}) + \sum_{i=1}^n H(y_i), \end{aligned}$$

where $\mathbf{y} = [y_1, \cdots, y_n]^T$ denotes a vector output; n denotes the number of the sensors; $H(\mathbf{y})$ denotes the joint entropy of the outputs, and $H(y_i)$ denotes the marginal entropy of the ith output. From $\mathbf{y} = \mathbf{W}\mathbf{x}$, it follows that (Cover and Thomas, 1991):

$$H(\mathbf{y}) = H(\mathbf{x}) + \log|\det(\mathbf{W})|, \tag{3.5}$$

where $\det(\mathbf{W})$ denotes the determinant of the *square* matrix $\mathbf{W}$,[7] and $H(\mathbf{x})$ denotes the entropy of $\mathbf{x}$ that is independent of $\mathbf{W}$.

Since the $p_i(y_i)$ are often unknown, we may use the *Gram-Charlier expansion* to approximate the marginal probability density and the associated marginal entropy $H(y_i)$; in doing so, the mutual information can be rewritten as (Amari et al., 1996):

$$\begin{aligned} D(\mathbf{W};\mathbf{y}) &\approx -H(\mathbf{x}) - \log|\det(\mathbf{W})| + \frac{n}{2}\log(2\pi e) \\ &\quad - \sum_{i=1}^n \left[\frac{(\kappa_{3,i})^2}{12} + \frac{(\kappa_{4,i})^2}{48} - \frac{5}{8}(\kappa_{3,i})^2\kappa_{4,i} - \frac{(\kappa_{4,i})^3}{16}\right], \end{aligned} \tag{3.6}$$

where $\kappa_{3,i}$ and $\kappa_{4,i}$ denote the third and fourth order cumulants of y_i.[8] In practice, we often use sample statistics to approximate $\kappa_{3,i}$ and $\kappa_{4,i}$; the first and the third terms of the right-hand-side of (3.6) are constants that are irrelevant to the optimization.

[7]If $\mathbf{W}$ is a non-square (degenerate) mixing matrix, $H(\mathbf{y}) = H(\mathbf{x}) + \log\sqrt{\det(\mathbf{W}\mathbf{W}^T)}$; more generally, if $\mathbf{y} = \mathbf{f}(\mathbf{W}\mathbf{x})$, then $H(\mathbf{y}) = -\int d\mathbf{x} p(\mathbf{x}) \log \frac{p(\mathbf{x})}{\sqrt{\det(\mathbf{J}\mathbf{J}^T)}}$, where $\mathbf{J} \equiv \mathbf{J}(\mathbf{x}) = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}$ is the Jacobian matrix.

[8]For a zero-mean random variable x, the first four cumulants are $\kappa_1 = 0$, $\kappa_2 = \mathbb{E}[x^2]$, $\kappa_3 = \mathbb{E}[x^3]$, and $\kappa_4 = \mathbb{E}[x^4] - 3(\mathbb{E}[x^2])^2$; for general definitions, refer to (Hyvärinen et al., 2001).

In what follows, we will apply the kurtosis objective function (3.2) for perceptual learning in three figure-ground segregation tasks, in which we are mainly concerned with sensory feature extraction (given two sensors) rather than source separation (though the problem formulation is motivated from ICA). Our motivation and contribution here, however, is substantially different from the available kurtosis maximization/minimization-based approaches in the ICA or blind separation literature (e.g., Girolami and Fyfe, 1997; Hyvärinen and Oja, 1997; Cardoso, 1998; Hyvärinen et al., 2001), where they are often the gradient-driven optimization procedures for *square-mixing* source separation; rather, we are interested in extracting the coherent (non-Gaussian) features from the binocular or binaural observations available from the normal human being.

In implementing the ALOPEX algorithm, there are two particular issues that need to be considered: a stopping rule for terminating the computation, and a criterion for assessing the quality of the outputs provided by the algorithm. In all the applications of the ALOPEX algorithm that are pursued in this chapter, we have followed the ensuring procedures:

- Based on experimentation, a number of iterations is determined for terminating the computation.
- Visual inspection of the pictures or waveforms, pertaining to the experiment in question, was used to assess the quality of the algorithm's outputs.

It is recognized that these two procedures are rather ad hoc in nature. Nevertheless, for the purpose of this thesis, they are considered to be adequate.

3.2.4 Binocular Fusion of Stereo Images

The spatial difference between two stereo images seen by the two eyes is called *binocular disparity*, which is an important cue for depth perception in stereovision (Julesz, 1971).[9] The computation of disparity often requires correct identification of corresponding features in the stereo images, hence it is essentially a correspondence problem. Here, we first consider a classic disparity perception task in the random-dot stereogram (Julesz, 1960, 1971). As shown in Figure 3.1, when the stereo images are presented to the eyes, the binocular fusion gives the percept of a central square floating in depth in front of the surround; although the object of the square does not physically exist in each image. The stereovision assumes that the eyes observe two slightly different scenes, and the depth perception is accomplished by detecting disparity of the object in the retinae. The first computational solution of stereo disparity was found by Marr and Poggio using a cooperative algorithm (Marr and Poggio, 1976; Marr et al., 1978). Other computational paradigms have also been proposed (e.g., Qian and Sejnowski, 1989; Qian, 1994). Here we

[9]The other cues such as the color, brightness, shading, or motion, can also help depth perception even with monocular vision.

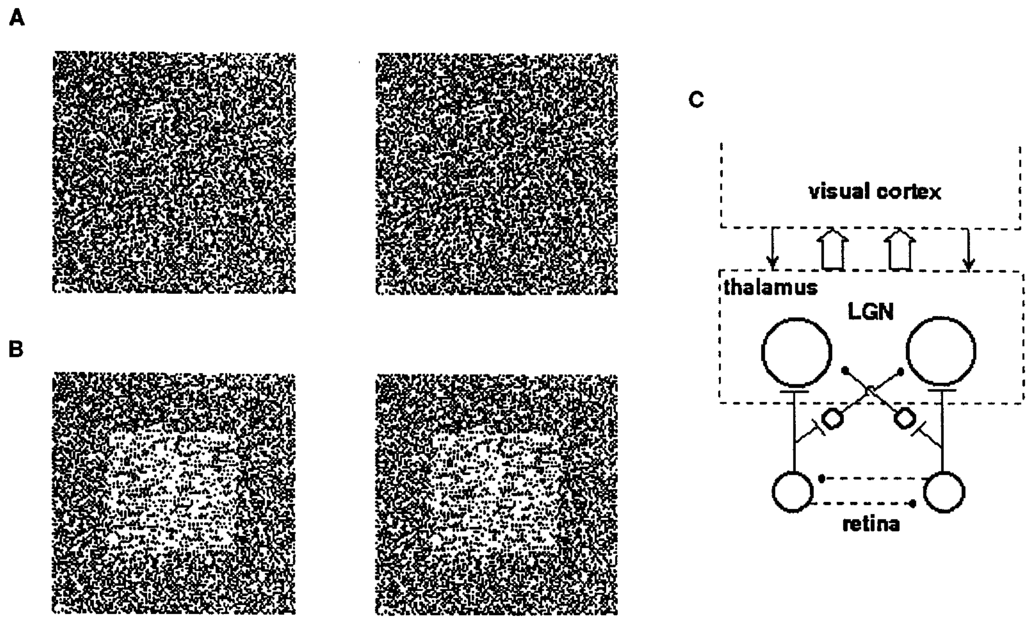

Figure 3.1: (A) Two 128 × 128 pixel random-dot stereograms used as input by analogy to stereovision. (B) The produced outputs after 5 iterations of learning rule (3.1); the squares appear detected floating above the grounds. (C) The functional wiring diagram, where the circles represent the populations of cells; modified from (Eggermont, 1990, Fig. 8.3, p. 153).

show that the simple stochastic correlative rule (3.1) can solve this stereo correspondence problem via a novel cooperative computation. We regard the two stereograms as the observations received in the retinae. The perception of the object hidden in the scenes is subject to a noisy ground interference. Imagine the disparity perception as a figure-ground segregation problem. The figure (the square in Figure 3.1A) and the ground (the random dots) are assumed to be two independent objects mixed in the scenes, which correspond to the the stereogram. Moreover, imagine the two-dimensional images as two one-dimensional signals located in a manifold coordinate. In particular, two two-dimensional images are vectorized into two one-dimensional vectors and then grouped into a 2×128^2 matrix $\mathbf{X}$. The left and right stereograms (after vectorization) are regarded as two mixing signal vectors $\mathbf{x}_1$ and $\mathbf{x}_2$. Visual perception is intended to find a demixing matrix $\mathbf{W}$, in an optimal way, to segregate the object. Namely, the resulting signals $\mathbf{Y} = \mathbf{WX}$ should be more easily perceived. Using (3.2) as an objective function, the adaption of $\mathbf{W}$ follows (3.1), where the initial parameter setup is: $\eta = 0.1, \gamma = 0.01$, $\mathbf{W} = \mathbf{I}$ (where $\mathbf{I}$ is an identity matrix). Without exception, the algorithm has constantly succeeded in detecting the "figure" within 10 iterations (Figures 3.1 and 3.2). Interestingly, as the learning process go on, the cross-correlation coefficient between the left and right images increases, starting from an initial value of 0.74 between $\mathbf{x}_1$ and $\mathbf{x}_2$ (Figure 3.1A) and

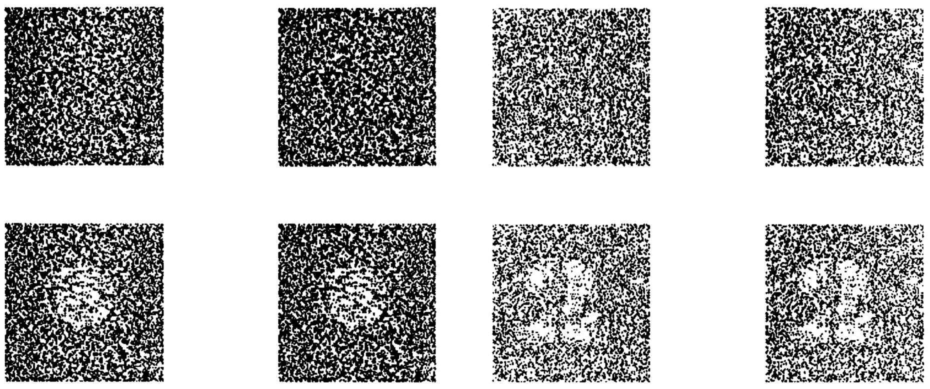

Figure 3.2: Another two experiments on binocular fusion of stereo images. In the first example, the "figure" appears as a circle; in the second example, the "figure" appears as a "2".

finishing at a value of 0.99 between $\mathbf{y}_1$ and $\mathbf{y}_2$ (Figure 3.1B) after 200 iterations,[10] whereas the kurtosis discrepancy increases from 0 to 0.01, suggesting higher-order statistics play a critical role in visual perception (Field, 1994; Olshausen and Field, 1996). In addition, changing the level of disparity does not affect our algorithmic performance, but it would influence the correlation statistic between the two stereo images.

We envision a functional wiring diagram (Figure 3.1C) for this stereo fusion task. The retinal cells receive the binocular stimuli projected from two eyes and pass to the lateral geniculate nucleus (LGN). The LGN cells relay the responses, through excitatory or inhibitory connections, to visual cortex. The correlative rule is carried iteratively via the thalamocortical (feedforward) and corticothalamic (feedback) loops (Harth et al., 1987; Mumford, 1995) to modify the connections. The success of (3.1) for the stereo fusion problem confirms the importance of synchrony or correlation between the visual cortex, LGN, and retinal cells, supported by numerous neurophysiological evidence (Eggermont, 1990; Miller, 1990; Singer, 1993; König and Engel, 1995; Alonso et al., 1996). The suggested correlative mechanism of synaptic plasticity between geniculocortical neurons and cortical cells provides a model for developing binocular disparity, as well as a new cooperative way for discovering depth.

The same principle can also be extended to the random-line stereograms (Julesz and Spivack, 1967; Julesz, 1971; Nishihara and Poggio, 1982). Figure 3.3 illustrates one example of a random-line stereogram and its associated stereo fusion results obtained from the learning rule (3.1). As expected, the learning rule also succeeds in finding the correct solution.

[10]Note that this does not mean the actual duration for detecting the transition from uncorrelated to correlated scenes; the neural mechanism behind the binocular fusion and rivalry processes is more sophisticated (Julesz and Tyler, 1976).

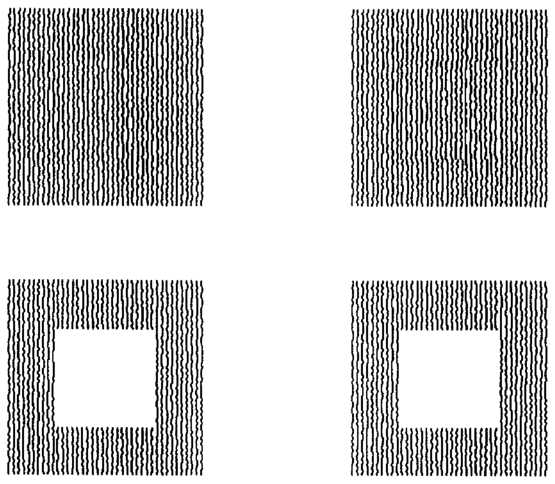

Figure 3.3: *Top row:* two 200 × 200 pixel random-line stereograms. *Bottom row:* the outputs produced by the algorithm after 10 iterations of correlative learning rule.

3.2.5 Perceptual Grouping

The second vision perception task is a motion perception problem. The principle of common fate is a perceptual grouping process demonstrated in a visual illusion experiment designed by Gestalt psychologists (e.g., Koffka, 1969; Rock and Palmer, 1990). We designed a sequence of frames that contain binary random-dot patterns. Given an individual static frame image, there is no cue for people to perceive an object in it. However, when the frames are played sequentially as in a movie,[11] we can readily perceive a moving disk-shaped object bouncing from left to right horizontally due to the so-called Gestalt common-fate grouping effect. Obviously, the frame-to-frame displacement of a group of dots is of sufficient disparity to enable us to group the dots into a moving object. Here we regard this perceptual grouping task as a figure-ground segregation problem. To illustrate this, we pick four pairs of consecutive frames (Figure 3.4a) from the movie that contains 66 frames in total. We view the consecutive two frames as two stereo observations, which are slightly different and highly correlated. Given a pair of consecutive frame images, we regard them as two mixing signal vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ after image vectorization. Using (3.2) as an objective function, we run the same learning procedure as in the binocular fusion task and observe the results. Interestingly, the correlative learning rule discovers the shape of the object as an obviously blank area (Figure 3.4b), which also indicates the corresponding position of the illusory object in the frame. This paradigm can be seen as a new cooperative computation of coherent visual (unstructured) motion perception

[11]The common fate phenomenon movie was kindly provided by Dave Landrigan. The video clip is made available at my website: http://soma.crl.mcmaster.ca/~zhechen/download/motion-percept.mov.

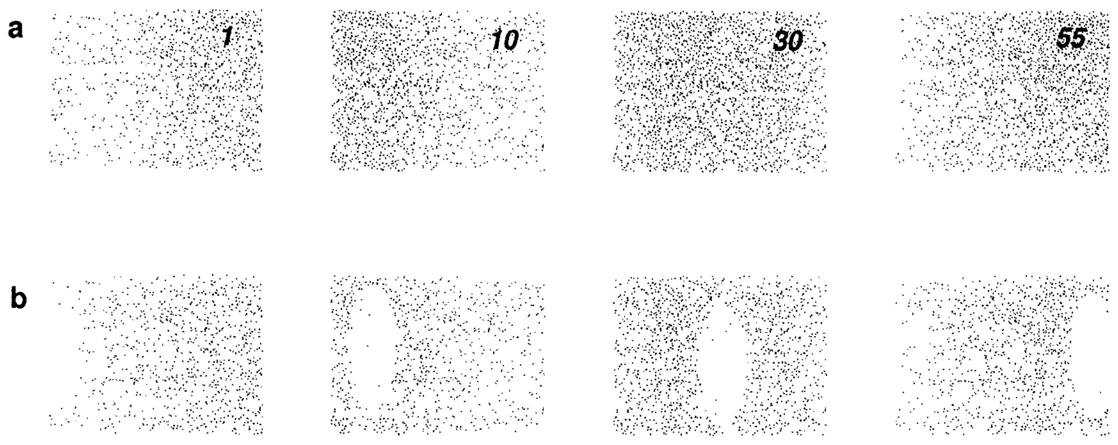

Figure 3.4: The common fate phenomenon and Gestalt grouping. (a) *Top row:* the selected 4 frames from a movie containing 66 frames, the legend numbers indicate their location. Each frame is a 240 × 320 pixel image. The frame-to-frame displacement of a group of dots is of sufficient disparity to enable us to group the dots into a moving object. (b) *Bottom row:* the selected outputs produced by the algorithm at different stages, each after 500 iterations of learning rule (3.1). Each resulting frame produces a clear perceptual grouping via a blank area that indicates the shape of the moving object.

(Anstis, 1980; Ullman, 1979; Yuille and Grzywacz, 1988; Sporns et al., 1991). Similar to the stereovision problem, the correspondence is established by detecting point-to-point correlations between two observations. In fact, it is exactly the difference between the two image observations that reveals the object. Despite the simplicity of the experimental design, this visual task does provide us some insight about the perception of coherent visual motion. The binocular disparity and binocular matching of stereo images produce a critical cue for motion perception.[12] It was early hypothesized (e.g., Singer, 1999; von der Malsburg, 1999), though still arguably (Shadlen and Movshon, 1999; Farid, 2002), that human visual system uses a temporal synchrony scheme for perceptual grouping. In a general setup of motion perception, the rigid (or non-rigid) body of the object is moving in a background-varying scene. Despite the varying background, the human is still efficiently capable of identifying the "figure", partially due to the effect of top-down expectation-driven attention of the brain.

We have noticed that the key to the success of the above-studied two tasks is to establish a correspondence between two observations via feature extraction (based on higher-order statistics). It is interesting to mention that, although there is no explicit mixing process involved (note that we have no direct access to matrix **S** or **A**), the way we treat the stereo images as two mixed sources containing non-Gaussian features allows

[12]Note that, although our designed experiment here treats the moving stimulus as a stereo visual input rather than a motion signal, we should not exclude the importance of the motion cues such as the optical flow and motion field, as used in many other motion detection approaches (Ullman, 1979; Verri et al., 1992).

us to perform an ICA-like computation. However, different from ICA, the stochastic correlative rule (3.1), being an algorithm for pattern extraction, avoids the complete separation and accomplishes the coincidence detection via a coherent computation.

We have also tried several popular ICA approaches, including the JADE (joint approximate diagonalization of eigen-matrices, Cardoso, 1999) and natural gradient (Amari et al., 1996) algorithms, to the above two visual tasks. In the binocular fusion experiment, two separation results obtained from the batch JADE algorithm and iterative natural gradient rule (2.11) (being an anti-Hebbian rule) are shown in Figure 3.5. As seen in the figure, the solutions arising from the ICA approaches are quite different from the one from the ALOPEX (see Figure 3.1). The ICA algorithms obviously attempt to separate out two distinct sources ("figure" and "ground") rather than establishing the correspondence through feature extraction and coincidence detection. Similar observations were also found in the Gestalt grouping experiment.

Remarks: It is noteworthy to comment that the natural gradient and the ALOPEX both belong to the iterative optimization approach; in solving the binocular fusion problem, they approach different solutions (Figures 3.5 and 3.1) in a number of different ways:

- It is found that the gradient-free ALOPEX is slightly more efficient (in terms of number of iterations to approach the solution) than the gradient-based algorithm such as the natural gradient: with pre-whitening step, natural gradient usually converges within 5 iterations (for Figure 3.5); without pre-whitening step, no convergence was observed within 500 iterations in our experiments, while the ALOPEX only takes $3 \sim 5$ iterations (without whitening) for the same experimental setup. This is probably due to the fact that the natural gradient algorithm attempts to minimize the mutual information between outputs $\mathbf{y}_1$ and $\mathbf{y}_2$, a task more difficult than the one used by the ALOPEX, which essentially performs feature extraction to produce two coherent outputs.

- The natural gradient algorithm often requires some prior knowledge in terms of choosing activation function for the non-Gaussian sources. In contrast, the ALOPEX is independent of the objective function to be used; it also requires neither activation function nor pre-whitening step in this visual task.

- It is also interesting to observe that the solution (Figure 3.1) obtained from the ALOPEX preserves the texture of the original visual images, whereas the solutions obtained from the ICA algorithms (Figure 3.5) seem to contradict our visual perception experience. This perhaps, in another way, explains why in approaching the solution, ALOPEX might be more biologically plausible.

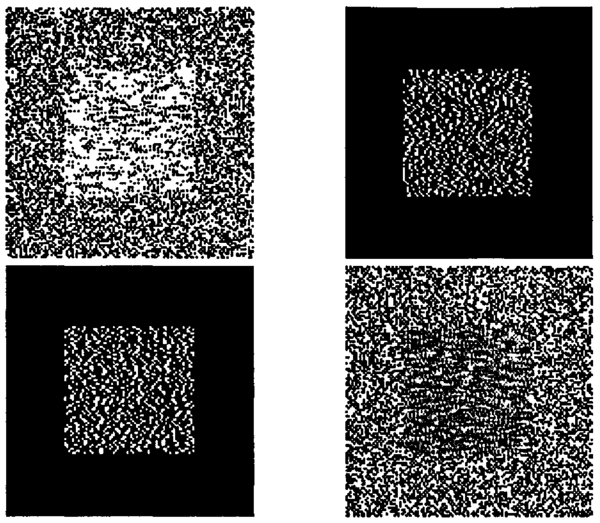

Figure 3.5: Source separation results for stereo images from the JADE (top row) and the natural gradient algorithm (with pre-whitening) after 5 iterations (bottom row).

3.3 Source Separation and Extraction: Cocktail Party Effect

3.3.1 Cocktail Party Problem

The *cocktail party problem*, first proposed by the British scientist Colin Cherry (then a visiting professor at MIT), is a psychoacoustic phenomenon that refers to the remarkable human ability to selectively attend to and recognize one source of auditory input in a noisy room environment, where the hearing interference is produced by competing speech sounds or a variety of noises that are often assumed to be independent of each other (Cherry, 1953). Following the early pioneering work of Cherry and his colleagues (see Chen, 2003b, for a detailed review), numerous efforts have been dedicated to the cocktail party problem, in such diverse fields as physiology, neurobiology, psychophysiology, cognitive psychology, biophysics, computer science, and engineering. Half a century after Cherry's seminal work, however, it seems fair to say that a complete understanding of the cocktail party phenomenon is still missing, and the story is far from complete; the enigma about the marvellous auditory perception capability of human beings remains elusive.

In addressing the cocktail party problem, three fundamental questions are of interest: *What is the cocktail party problem? How does the brain solve it? Is a machine capable of solving it?* The first two questions are human-oriented, which mainly involve the disciplines of neuroscience, cognitive psychology, and psychoacoustics; the last question is rooted in machine learning, which involves computer science and engineering disciplines. We refer the interested reader to (Chen, 2003b) for a comprehensive overview.

In relating our proposed algorithm to the above problem, we propose a correlative

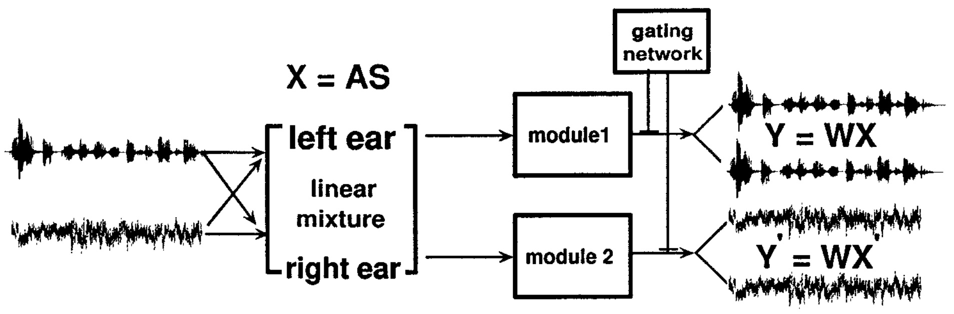

Figure 3.6: A functional diagram of sound mixing and auditory streaming. The two modules are responsible for segregating different objects in the auditory scene. To model the selective attention mechanism in the thalamus, a dynamic gating network is postulated to decide/switch which module is to be selected through a thalamocortical loop. Here $\mathbf{W}$ is a 2×2 matrix. Here $\mathbf{X}'$ denotes the rotated version of the original mixed signal matrix $\mathbf{X}$; $\mathbf{Y}$ and $\mathbf{Y}'$ recover two different auditory streams.

firing mechanism underlying the stochastic correlative learning rule, as described in (3.1), as a postulated solution to a simplified cocktail party problem with an instantaneous linear mixing setup. Motivated by (but different from) the ICA or blind source separation (BSS) approaches in the literature, our solution attempts to find the "interesting" figure out of the sound mixtures, instead of separating out all of the sources. With a slightly different goal-oriented procedure, our proposed solution overcomes the requirement of conventional ICA approaches regarding the number of sensors. We also suggest an associative neurophysiological (correlative firing) mechanism underlying the correlative learning rule.

3.3.2 Source Separation and Extraction

How to segregate the attended speech is an essential task in auditory perception (Bregman, 1990). In the auditory cortex, the brain attempts to reconstruct the original auditory scene, which requires sound localization ("where") and sound streaming ("what"). The goal of figure-ground segregation in this context is to recover the speech signal of interest ("figure") from a complex auditory scene. Consider a binaural hearing scenario where there are two audio sources x_1 and x_2. We speculate that in different parts of the auditory cortex, there exist distributed modules that are responsible for processing the mixing signals $\mathbf{X} = [\mathrm{x}_1, \mathrm{x}_2]^T$, received from the two ears after certain sophisticated preprocessing by the cochlea. Different modules perform different segregation tasks, and the selected attention mechanism is governed by a gated network that decides to focus on one stream or switch to another. Each module runs a stochastic correlative learning rule as (3.1) and produces two coherent outputs $\mathbf{Y} = [\mathrm{y}_1, \mathrm{y}_2]^T$. The computations in the modules are performed in parallel and the respective segregation outcomes are distinct (see Figure 3.6). It should be stressed that the learning rule (3.1) makes no attempt to conduct the same job

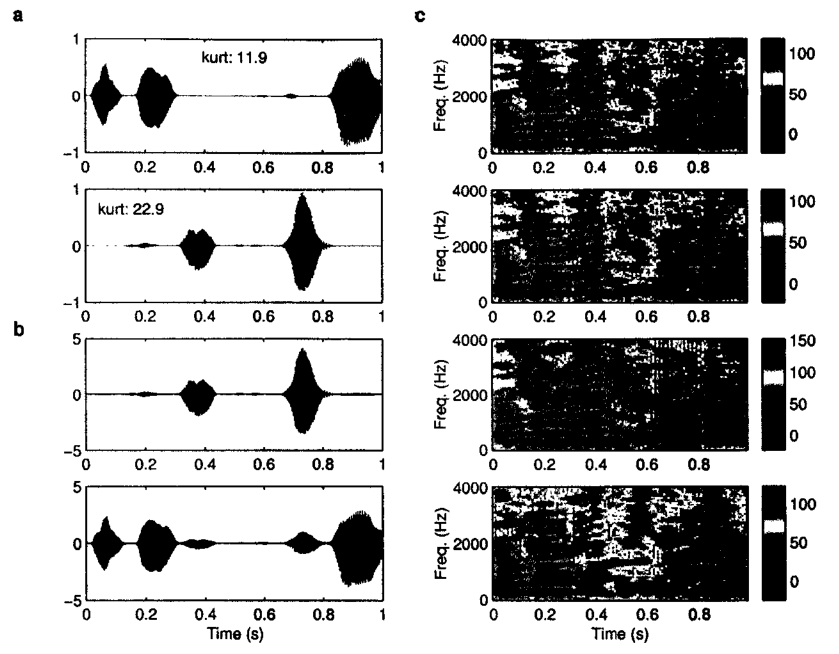

Figure 3.7: (a) The two speech signals used as source inputs: x_1 and x_2. The numbers indicate the kurtosis statistics of the signals (the same for the subsequent figures). (b) The recovered signals from two modules after 400 iterations of correlative learning rule. (c) The spectrograms of the clean and recovered signals.

as conventional ICA algorithms, whose goal is to separate all the sources (possibly more than two given merely two sensors). Here the outcomes from each module are identical, hence only one stream is attained in each module. This is consistent with human auditory perception experiences: at a particular moment, we only pay special attention to one or another source, but not together. In the segregation task, we are only interested in recovering the figure rather than separating or decomposing the scene (Bregman, 1990). The role of selective perception enables us to hear (or see) what we expect to hear (or see). Therefore, from the start of the sensory perception the concepts of "figure" and "ground" should be identified. In each module, sounds or voices should be perceived synchronously and coherently. This key observation was found in the experiments using our learning rule. We have suggested that Figure 3.6 is a simplified model of auditory perception. Nevertheless, no claim is made here that this is a complete story. In fact, it is more realistic to integrate multiple cues by introducing certain feedback pathways between the visual and auditory cortices (see Chen, 2003b, for discussions).

We have tried different numbers of sensors and sources in the *simulated* cocktail party problem. Motivated by a biological constraint, we focus on the cases with two sensors and different numbers of sources (up to 4) and different mixtures of source signals, as shown below. The speech signals used here have the sampling frequency 8kHz, unless specified

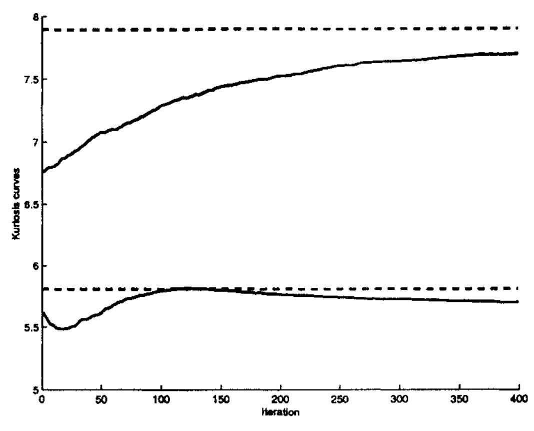

Figure 3.8: The learning curves of kurtosis of two segregated sources in Figure 3.7, where the mixing matrix is randomly initialized. The solid curves indicate the kurtosis curves change along the learning process. The dashed line indicate the true kurtosis values of two original sources. As seen, one curve actually converges after 100 iterations, then it seems to run into over-learning.

otherwise. The initial demixing matrix $\mathbf{W}$ is often set as $0.5 \times \mathbf{I}$.[13] The choices of η and γ determine the convergence speed and stability of the optimization process. We have consistently used $\eta = \gamma = 0.01$ for all the tasks reported here, and no divergence result has been found.

Two Sensors and Two Sources. In Figure 3.7, we used two speech signals (both super-Gaussian) as source signals, the mixing matrix $\mathbf{A}$ was randomly initialized. In the figure, the matrix

$$\mathbf{A} = \begin{bmatrix} 0.56 & 0.79 \\ -0.75 & 0.65 \end{bmatrix}$$

was used. Since each module can only recover one source, in order to recover another one in the scene, we assume another mixing matrix for the second module, denoted by $\mathbf{A}'$, subject to an orthogonal rotation operation of $\mathbf{A}$: $\mathbf{A}' = \mathbf{AR}$, where $\mathbf{R}$ is a rotation (unitary) matrix with a rotation angle θ:[14]

$$\mathbf{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}.$$

Hence, we can get another mixed signal for the second module as $\mathbf{X}' = \mathbf{A}'\mathbf{S} = (\mathbf{AR})\mathbf{S}$ (here $\mathbf{R}$ can be interpreted as a selective attention mechanism). Two modules were

[13]Note that due to the simulation effect here, the previous learned experience (i.e. pre-wired synapses) cannot be beneficial to the next trial; this is nevertheless not the case for the human brain.

[14]It is obvious that when $\theta = \pi/2$, it is equivalently performing a permutation operation to the mixing matrix $\mathbf{A}$.

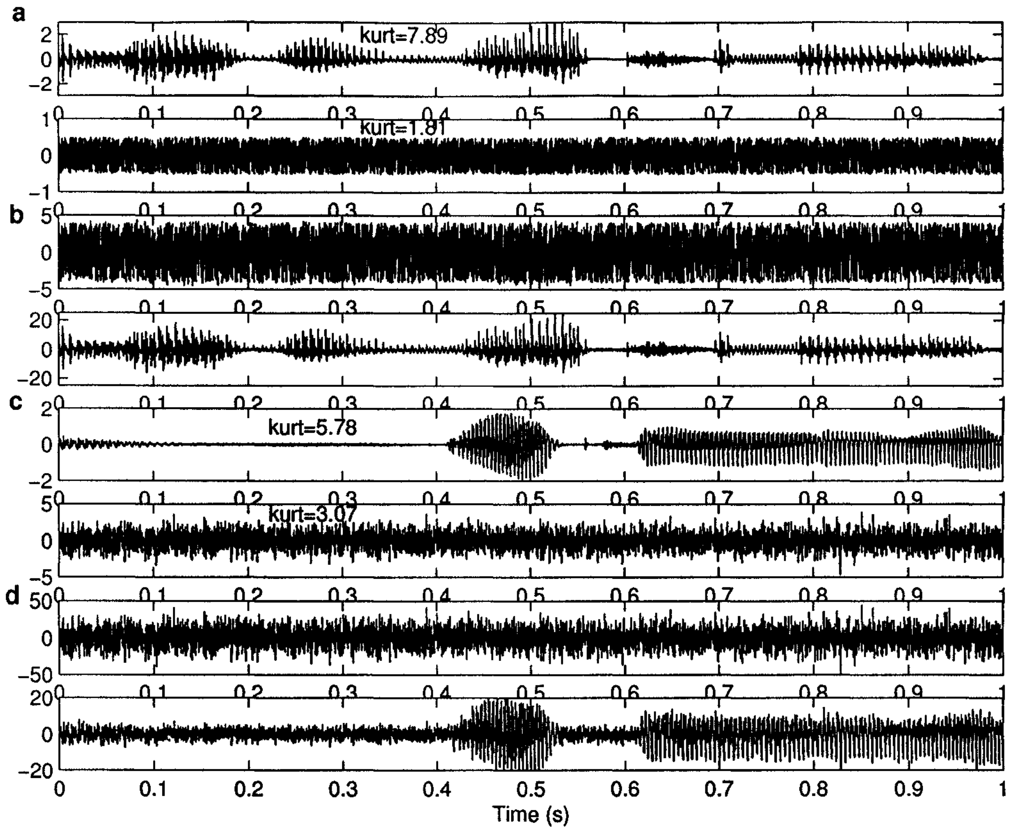

Figure 3.9: The two sensors and two sources case: (a) Source signals, one is a speech signal (super-Gaussian), another is uniform random noise (sub-Gaussian). (b) Separated signals. (c) Source signals, one is a speech signal (super-Gaussian), another is Gaussian noise. (d) Separated signals.

operated in parallel, and the outcomes produced the complete "figures" of the scene. Figure 3.8 shows the kurtosis curves of two modules' output along the learning process (within 400 iterations); clearly, the curves gradually converge to the true statistical values. Alternatively (but not equivalently),[15] without the knowledge of the mixing matrix $\mathbf{A}$, we can use $\mathbf{X}' = \mathbf{RX} = \mathbf{R}(\mathbf{AS})$ to introduce a variability between $\mathbf{X}$ and $\mathbf{X}'$ within the two different modules. Note that, different from conventional ICA algorithms, the product $(\mathbf{WA})$ is far from a diagonal matrix. For instance, in Figure 3.7, we have

$$(\mathbf{WA})_{module1} = \begin{bmatrix} 0.0952 & 4.5126 \\ -1.2080 & 4.2998 \end{bmatrix}, \quad (\mathbf{WA})_{module2} = \begin{bmatrix} 4.3671 & -0.1959 \\ 4.2247 & 1.3230 \end{bmatrix}.$$

Consequently, the output signals $\mathbf{Y} = \mathbf{WX} = \mathbf{W}(\mathbf{AS})$ do not resemble the source matrix $\mathbf{S}$ (even up to possible permutation and scaling). Another segregation result under a different situation (one super-Gaussian the other sub-Gaussian) is shown in Figures 3.9a and 3.9b.

[15]Note that this is not equivalent since the matrix multiplication is noncommutative.

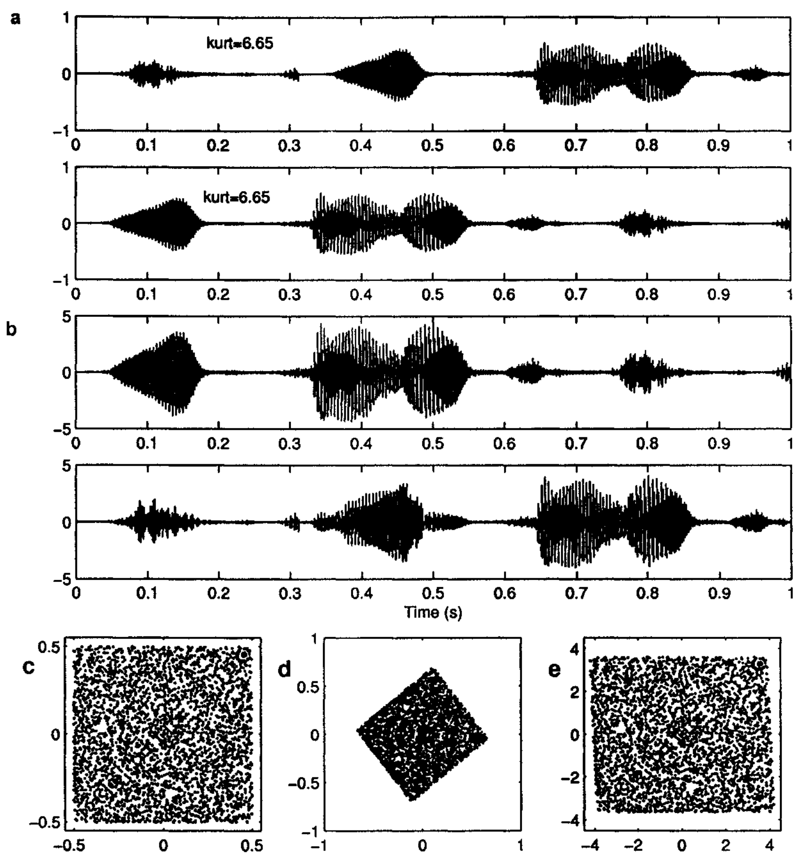

Figure 3.10: The two sensors and two sources case: (a) Two speech source signals. (b) Separated results. (c) Phase diagram of two source signals that are uniform distributed in $[-0.5, 0.5]$ and have the same kurtosis statistics. (d) Phase diagram of the mixed signals (of one module). (e) Phase diagram of the separation signals.

In order to test the robustness of the non-Gaussianity assumption about the sources, we have also attempted to separate a mixture of a speech signal (super-Gaussian) and a Gaussian noise source. It was found that the separation performance degraded somewhat, but the recovered speech signal was still clearly distinguishable (Figure 3.9d). Since the figure-ground segregation task is based on the characteristic of kurtosis, a natural question to ask is the *identifiability problem*: if two sources have the same kurtosis statistics, can the learning rule (3.1) recover them? Our experiments (see Figure 3.10) have confirmed a positive answer. One experiment is on the time-reversed speech signals. In another experiment, two uniform distributed signals were designed to be nearly uncorrelated (with correlation coefficient 0.01) but have the same kurtosis value of 1.80.

In addition, as a comparison with the conventional ICA approaches, we also conduct a square demixing experiment using the natural gradient (Amari et al., 1996) and JADE (Cardoso, 1999) algorithms. The source signals are two 2.5s duration speech signals with

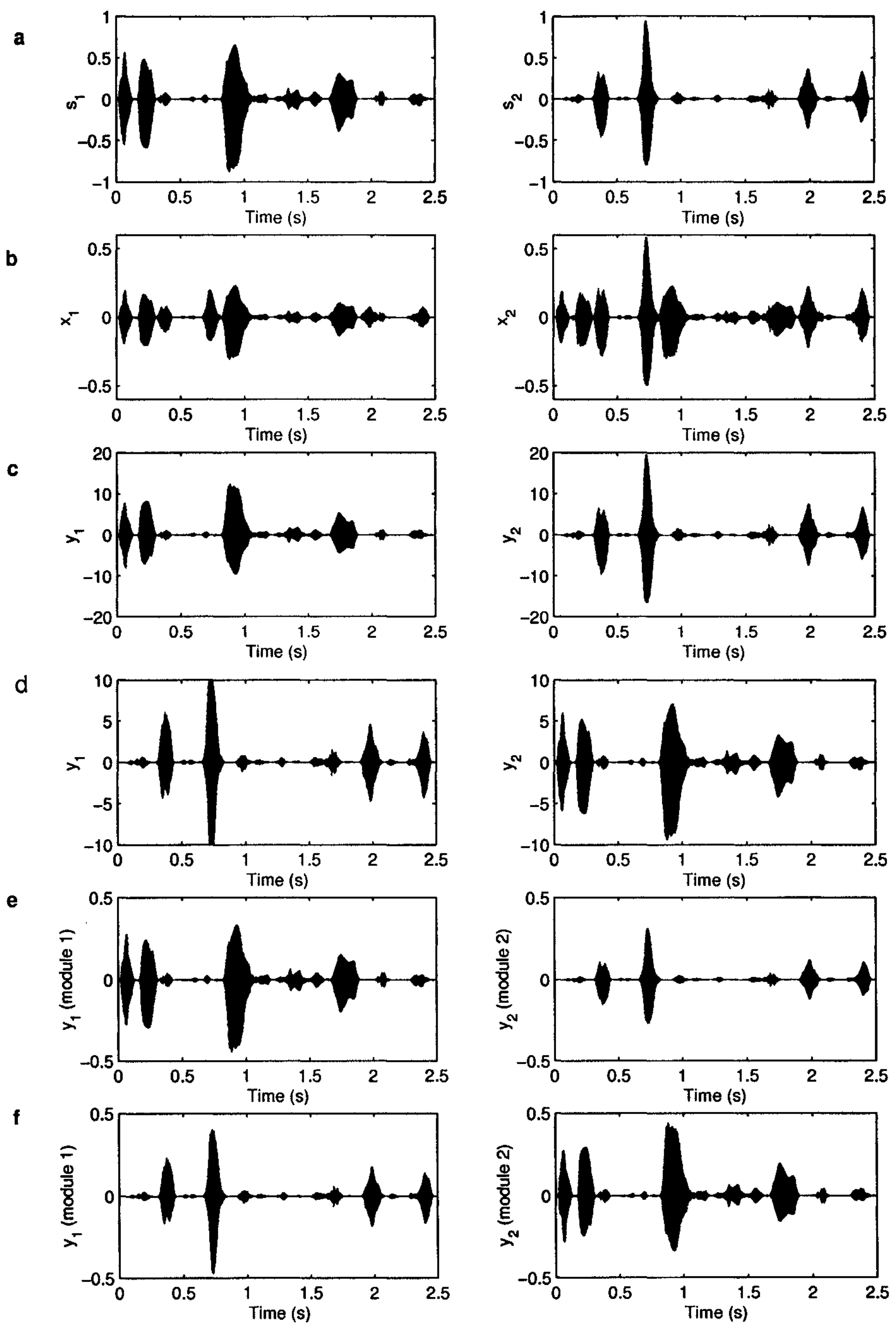

Figure 3.11: Experimental comparisons. (a) The two original source signals. (b) Two mixed signals. (c) Separated signals from the natural gradient rule after 300 iterations. (d) Separated signals from the JADE algorithm (batch approach). (e)(f) Separated signals (of two modules) from the ALOPEX with objective functions of maximum kurtosis (3.2) and negentropy (3.3) after 1000 iterations, respectively.

sampling frequency 16kHz. The instantaneous mixing matrix is set as

$$\mathbf{A} = \begin{bmatrix} 0.35 & 0.21 \\ 0.34 & 0.62 \end{bmatrix}.$$

For a fair comparison, the mixed signals (i.e., the matrix $\mathbf{X}$) are pre-whitened (with zero-mean and unit variance) before testing the three algorithms; all algorithms are performed in the batch mode, with the same initial demixing matrix $\mathbf{W} = 0.5 \times \mathbf{I}$; the learning rate for the natural gradient rule (2.11) is set as $\eta = 0.1$, whereas the parameters for ALOPEX are $\eta = 0.01$ and $\gamma = 0.01$. Figure 3.11 shows the separation results obtained from the natural gradient rule (2.11) with the objective function (3.6), JADE, and the ALOPEX learning rule (3.1) using the objective functions (3.2) as well as (3.3). It was found that the natural gradient and JADE algorithms converge faster than the ALOPEX (or have better performance within the same number iterations); this is not surprising since ALOPEX is a stochastic gradient-free optimization rule and the search direction is rather random, whereas the deterministic algorithms such as the natural gradient, JADE, and FastICA are more efficient due to the use of extra knowledge (e.g., choice of activation function, eigen-matrix decomposition, orthogonality constraint, etc.). Note that we make no claim here to find the global solution using the learning rule (3.1). Actually in perceptual learning, no global solution is required; rather, an efficient and robust procedure is more favorable. In addition, it should be noted that the conventional ICA algorithms (including natural gradient, FastICA, and JADE) are limited by the assumption of square mixing; namely, the number of sensors must be *no less than* the number of sources. However, this constraint does not apply to the ALOPEX while using the objective function (3.2), as we show below in the degenerate mixing cases.

Two Sensors and More Than Two Sources. When the number of sources is greater than the number of sensors, the system becomes under-determined. Since it is impossible to construct the complete orthogonal modules, the complete separation becomes intractable. In such cases, we turn to perform source extraction instead of source separation. Namely, only one stream of sources is extracted from the two mixtures. One of the experimental results for this situation is shown in Figure 3.12, where the mixing matrix

$$\mathbf{A} = \begin{bmatrix} 0.4120 & -0.3445 & 0.1677 \\ 0.3590 & 0.3191 & 0.2819 \end{bmatrix}$$

is used. The same principle can be extended to the case of two sensors and four sources. See Figure 3.13 for the experimental results, where the mixing matrix is

$$\mathbf{A} = \begin{bmatrix} 0.56 & 0.79 & 0.15 & -0.37 \\ -0.75 & 0.65 & -0.11 & 0.86 \end{bmatrix}.$$

It was observed in Figures 3.12 and 3.13 that although the systems were under-determined, the recovered signals were still distinguishable or intelligible (for speech signals). However, the success of recovering all the "figures" in a scene in the under-determined situation is not always guaranteed in every trial; rather, it is quite dependent on the mixing matrix $\mathbf{A}$. We will give more analysis on this issue in Section 3.5.

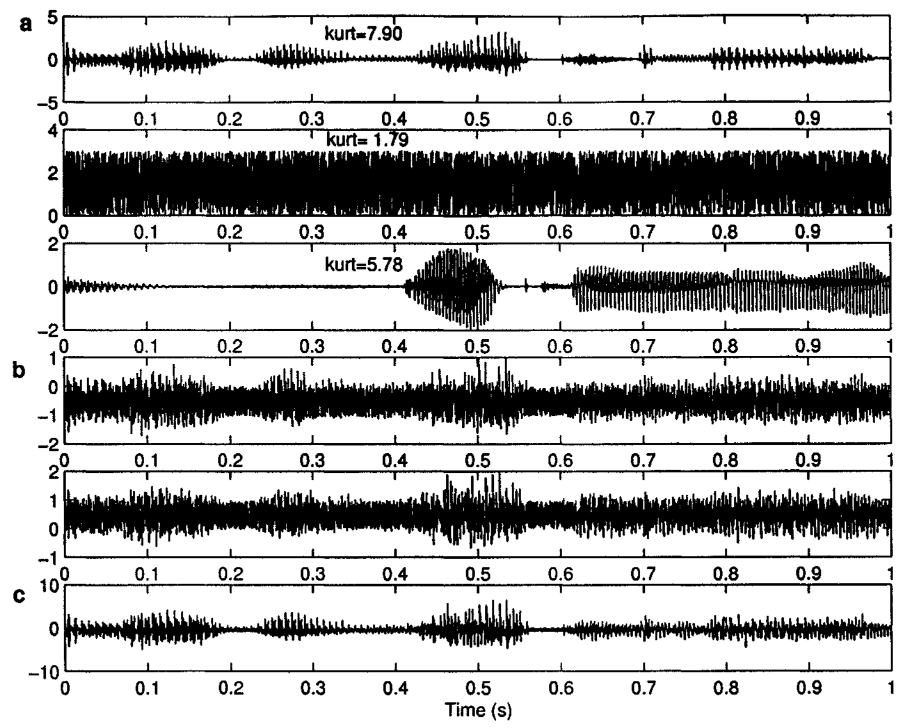

Figure 3.12: The two sensors and three sources case: (a) Source signals. (b) Mixed signals. (c) Recovered signal.

Remarks. Typically, the segregation results are usually good when the number of sources is identical to the number of the sensors; when the number of sources is greater than the number of sensors, the performance degrades somewhat but the "figure" is still fairly distinguishable or intelligible. In general, the signal-to-noise ratio (SNR) of the "figure" is improved. In an off-line learning setup, the stochastic correlative learning rule typically accomplished the tasks within 300 to 2000 iterations, independent of the selected non-Gaussian sources. However, the choice of mixing matrix often has an effect on the convergence speed as well as separation results. In some cases, for example, the algorithm cannot recover the whole scene if the mixing matrix **A** is nearly singular. This is because if the mixed signals are dominated by one stream, the recovered signal will be the dominated one, and the rotation trick will not help any more to recover the others. In the source separation/extraction experiments, it was found that the pre-whitening of the mixed signals always enhances and improves the final results. The whitening step essentially decorrelates between the mixed signals and makes the separation or extraction task easier (e.g., Hyvärinen et al., 2001), especially when the sources are correlated and the mixing matrix is far from orthogonality;[16] therefore, whitening is often used in practice

[16]If the mixing matrix **A** is almost orthogonal, then $\mathbf{A}\mathbf{A}^T$ is close to diagonal; in such a case and when the source signals are additionally uncorrelated, then the mixed signals are hardly correlated, namely, $R_{\mathbf{x}} \equiv \mathbb{E}[\mathbf{x}\mathbf{x}^T]$ is also close to diagonal. Orthogonality of the matrix **A** is beneficial for kurtosis-maximization-based ICA algorithms.

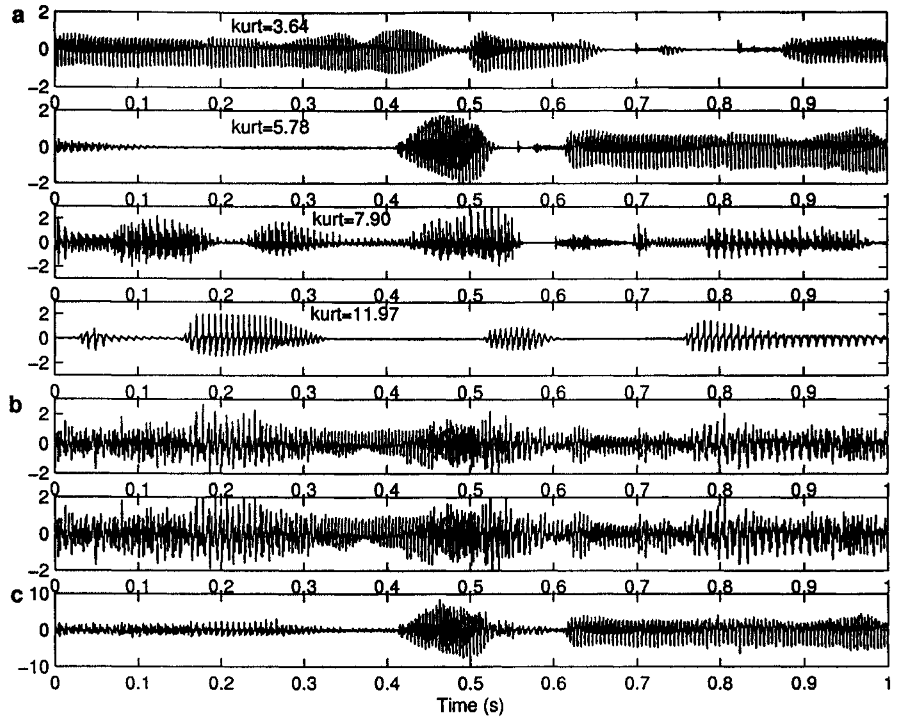

Figure 3.13: The two sensors and four sources case: (a) Source signals. (b) Mixed signals. (c) Recovered signal.

as a preprocessing step.

Using objective function (3.2) or (3.3), the learning rule (3.1) produced two duplicative outputs (in one module), indicating that the characteristic of (3.1) is to identify the "figure" rather than decompose the "ground". Since the essential function role of (3.1) is to perform source extraction, only one signal stream can be extracted from one module. As such, the dimensionality of the demixing matrix $\mathbf{W}$ can be set as $1 \times n$ ($n \geq 2$) instead of $2 \times n$, where n denotes the number of the sensors.

Thus far, the sound separation/extraction experiments reported here were performed within the off-line learning framework. Nevertheless, it is possible to extend the algorithm to perform a sequential separation task via the block-by-block processing, which further requires on-line estimation of the kurtosis or cumulant statistics appearing in (3.2), (3.4) and (3.6). For instance, we can sequentially estimate the *cumulant* and *normalized kurtosis* statistics as follows:

$$\begin{aligned}
\hat{\mu}_p(y(t)) &= (1-\lambda)\hat{\mu}_p(y(t-1)) + \lambda y^p(t), \\
\hat{\kappa}_2(y(t)) &= (1-\lambda)\hat{\kappa}_2(y(t-1)) + \lambda(\hat{\mu}_2 - \hat{\mu}_1^2), \\
\hat{\kappa}_4(y(t)) &= (1-\lambda)\hat{\kappa}_4(y(t-1)) + \lambda(-6\hat{\mu}_1^4 + 12\hat{\mu}_1^2\hat{\mu}_2 - 3\hat{\mu}_2^2 - 4\hat{\mu}_1\hat{\mu}_3 + \hat{\mu}_4), \\
kurt(y(t)) &= \frac{\hat{\kappa}_4(y(t))}{\hat{\kappa}_2^2(y(t))} - 3,
\end{aligned}$$

where $\hat{\mu}_p$ and $\hat{\kappa}_p$ denote the estimated p-order *raw moment* (moment about zero) and *cumulant* statistics, respectively, and λ is a discount factor. In addition, in real-life acoustic environments, the instantaneous linear mixing assumption is not valid, rather, the audio sources are often modeled as a convolution operation in temporal domain. By using the FFT or subband processing techniques, it is also possible to apply any source separation/extraction algorithm to the frequency domain (see e.g., Haykin, 2000; Hyvärinen et al., 2001).

3.4 Discussion

3.4.1 Correlative Learning and Limitations

To summarize, we have proposed to use a gradient-free, stochastic correlative learning rule for several figure-ground segregation tasks, whose common goal is to discover the "figure" from a complex "ground" scene and to establish the correspondence between two sensors in either stereovision or binaural hearing. Although the stereotype ALOPEX procedure is a biologically motivated algorithm (Harth and Tzanakou, 1974; Tzanakou et al., 1979), the biological plausibility of the learning rule (3.1) to sensory perception still requires further physiological experiments. For instance, the kurtosis objective function, while being statistically interesting, is unlikely to be employed in the brain, neither is it possible to accommodate the real-time, incremental, visual and auditory processing. A more biologically plausible objective function might be the population cells' synchronous firing rate (or the number of firing spikes) that maximizes the neurons' response (i.e., higher firing rate). Hence, a rate-coded correlation-based (Hebbian or anti-Hebbian) learning (e.g., Abbott and Song, 1999; Kempter et al., 1999; Gerstner and Kistler, 2002) is more realistic for biological computation. The biological objective function might not be unique, and it might also have many constraints.

The hypothesis of a Hebbian-like correlative mechanism of synaptic plasticity is widely accepted in neurobiology. One key characteristic of Hebbian learning is that the biological computation only requires local information available in the neurons. However, the conventional Hebbian learning does not involve any feedback mechanism, which is deemed ubiquitous and important in the brain. Including a designed global objective function might suggest a top-down expectation for synaptic plasticity, but whether it is biologically plausible is an open question. Research in neuroscience has also suggested other synapse developments distinct from the Hebbian synapse (e.g., von der Malsburg, 1981; Singer, 1993; König and Engel, 1995; Murphy, 2003).

It should be noted that our proposed figure-ground segregation framework is certainly over-simplified and is by no means a complete story. More realistic computational models should integrate bottom-up cues as well as top-down knowledge for figure-ground discrimination. We believe that perception should not impose itself as an NP-hard problem in terms of computational effort, which nevertheless does not imply the solution needs to be globally optimal. The investigated stochastic correlative learning rule is not intended to find a globally optimal solution in all the perception tasks, only a perceptually satisfactory solution is sufficient. As we have discussed earlier, the ALOPEX-like algorithms (Harth and Tzanakou, 1974; Tzanakou et al., 1979; Harth et al., 1988; Unnikrishnan and Venugopal, 1994; Tzanakou, 2000; Bia, 2001) have many attractive features from a biological computation perspective. To repeat some points here, first, it incorporates a feedback mechanism in the learning rule, and cycles and loops are important and ubiquitous in biology (Bell, 1999). Second, it is natural to relate the pre- and postsynaptic (cause and effect) activities by incorporating the higher-order correlation through the objective function. Third, the learning rule (3.1) is a sort of "trial-and-error" learning, which resembles some form of reinforcement-like learning. Fourth, the ALOPEX-like algorithms are temporally synchronous, parallel, and therefore scale well for large systems.

3.4.2 Timing Issue

The proposed stochastic correlative learning rule requires a number of iterative updates for the synapses; however, it is noted that the number of iterations in each task does not really reflect the computational time required for the neuronal cell. In fact, the timing issue is important in terms of biological plausibility as well as theoretical understanding. Being an extremely efficient computer, how much time does the brain really require to perform the perceptual tasks? We are not attempting to provide a complete answer here, but it is worthwhile to take a close look at this issue.

In the stereovision experiment, the proposed algorithm converges very fast (within 5 iterations) for the binocular fusion task. Supposing that one iteration between the thalamus and the cortex would take 5 ms, the brain therefore requires at least 25 ms to "compute" the stereovision, which seems fairly reasonable.

In the cocktail party effect experiment, the quality of solution often requires quite a bit of computation (300 to 1000 iterations, depending on the initial condition). Supposing that each synaptic computation iteration occurring in the auditory pathways takes 1 ms, this implies about 0.3 to 1s is required for detecting the signal of interest, which seems quite unreasonable in practice. One possible explanation is that some of the synapses might be hardwired due to the long-term evolution learned from the past experiences, thereby a pre-learned synapse (namely, the initial synaptic weights in $\mathbf{W}$) might reduce the real-time computational burden.

The same explanation might also apply to the unstructured motion perception task. Given the common 25 frames/s motion picture standard, assuming a delay of 5 ms per iteration, few hundreds of iterations might take 1 to 2 seconds to accomplish the task. This seems to be too much time required for the cortex. In addition to the hardwired synapse, we suspect that other motion cues may also serve as a catalyst in helping to detect a moving object.

3.4.3 Perception-Action Cycle

It is our belief that these above-mentioned characteristics of ALOPEX-like algorithms are crucial for various kinds of cognitive behavior including perception and action. The "perception-action" cycle is a key concern in artificial intelligence and neuroscience. Computational neuroscientists often model perceptual behavior as a feed-forward process, and somehow ignore the feedback mechanisms, or regard them as some side-effects (Bell, 1999). The conventional ICA is a representative example.[17] However, as we know, feedback mechanisms are ubiquitous in the human visual (see e.g., Mumford, 1995) as well as auditory (see e.g., Chen, 2003b) systems. As far as modeling and learning are concerned, we have to incorporate feedback mechanisms into the biological computations. How to do it still remains an open problem. Here, because of the above-discussed features of the ALOPEX-like algorithms, we suggest that the stochastic correlative learning might fit itself into the "perception-action" cycle for computational neuroscience. Tony Bell (2003) has suggested a vivid diagram of the "*look-think-do*" process (see Figure 3.14), which illustrates very well the loop and cycles in these three intelligent activities — the perception and action are inherently interwoven.[18] Stated in mathematical terms, we propose the following (certainly oversimplified) framework for an interacting "perception-action" process: Let $\mathbf{w}$ and $\mathbf{v}$ denote the parameters that govern the interacting perception/cognition and action processes, respectively; $E(\mathbf{w}, \mathbf{v})$ and $J(\mathbf{w}, \mathbf{v})$ represent two objective (cost or reward) functions that relate to the variables $\mathbf{w}$ and $\mathbf{v}$. In general, the "perception" and "action" proceed alternately to optimize two objective functions in the following forms:

$$\begin{aligned} \Delta \mathbf{w}_{t+1} &\leftarrow \eta \Delta \mathbf{w}_t \Delta E_t(\mathbf{w}_t, \mathbf{v}_t) + \gamma \mathbf{r}_t, \quad (3.7a) \\ \Delta \mathbf{v}_{t+1} &\leftarrow \eta \Delta \mathbf{v}_t \Delta J_t(\mathbf{w}_{t+1}, \mathbf{v}_t) + \gamma \mathbf{n}_t, \quad (3.7b) \end{aligned}$$

where $\mathbf{r}$ and $\mathbf{n}$ denote two random factors that influence the above two equations. Basically, in the perception process, while fixing the current $\mathbf{v}_t$, equation (3.7a) updates $\mathbf{w}_t$ to minimize/maximize the objective function E_t; in the action process, while taking account of the feedback $\mathbf{w}_{t+1}$, equation (3.7b) updates $\mathbf{v}_t$ to minimize/maximize another objective function J_t, given previous action and previous cost/reward. These two equations describes a sensor-motor coupling in the perception-action loop, which can be regarded as a sensorimotor coordination in an embedded system where the ALOPEX procedure is used as the multiple-objective function optimization engine.

3.4.4 Implications for Perceptual Learning and Beyond

Although thus far we have only focused on figure-ground segregation under the ICA-like formulation, the stochastic correlative learning rule, nevertheless, is very generic and can be applied to various (unsupervised) perceptual learning tasks (e.g., Fahle and Poggio,

[17]Recently, Shriki et al. (2001) have generalized the feed-forward ICA to a recurrent architecture with lateral inhibitions.

[18]The "sense-think-act" cycle was also discussed in (Pfeifer and Scheier, 1999, Chap. 12).

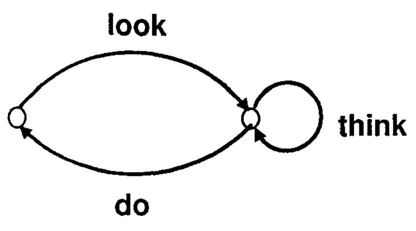

Figure 3.14: A schematic diagram of "*look-think-do*" process with loop and cycle (from Tony Bell).

2002). For instance, it can be used for learning the decorrelated features of visual scenes or learning the factorial representation of the data, or used for learning a feedback model of visual attention (Janakiraman and Unnikrishnan, 1992). It can also be applied for discovering surfaces in random-dot stereograms using a similar procedure described by Becker and Hinton (1992).

We believe that the ALOPEX-like stochastic correlative learning rule is not limited to the sensory systems. Rather, the application to the motor system is quite straightforward. Since (3.1) is model-free, the stochastic correlative learning rule can be used for supervised or reinforcement learning. Based on some promising results that will be reported later in Chapters 4 and 6, we believe that the stochastic correlative learning framework is potentially applicable to perceptual (auditory and visual) learning (Rao, 1999; Reymond et al., 2002), motion tracking, and motor control (Wolpert and Ghahramani, 2000). In addition, similar optimization procedures have been utilized for the first Brain to Computer Interface (Tzanakou, 2000), for curving fitting and combinatorial optimization (Harth et al., 1988), for learning decision-trees (Shah and Sastry, 1999; Sastry et al., 2002), for auditory stimulus optimization (Anderson and Tzanakou, 2002), and for target optimization in surgical treatment of Parkinson's disease (Hamilton et al., 2000).

3.5 Appendix on Linear Operator and Algebra

Under the instantaneous time-invariant linear mixing assumption, the source separation/extraction problem can be understood as solving a linear operator equation:

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}, \tag{3.8}$$

where $\boldsymbol{A} : X \rightarrow Y$ is a linear operator from a normed space X into a normed space Y. The goal is to recover the source(s) $\boldsymbol{x}$ from the observations $\boldsymbol{y}$, which is essentially to find an inverse operator: $\boldsymbol{A}^{-1} : Y \rightarrow X$. To understand the solution to such an inverse problem, we resort to the regularization theory (see Chen and Haykin, 2002, for a review). In a source separation context, operators $\boldsymbol{A}$ and $\boldsymbol{A}^{-1}$ correspond to the mixing matrix $\mathbf{A}$ and demixing matrix $\mathbf{W}$, respectively. Suppose $\mathbf{A}$ is an $n \times m$ matrix, where m denotes the number of sources, and n denotes the number of sensors. Note that $\mathbf{A}$ is not necessarily a square matrix. When $m = n$, ideally, the solution is given by the inverse operator: $\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{y}$, or, in the matrix form:

$$\mathrm{S} = \mathrm{A}^{-1}\mathrm{X} = \frac{1}{\det(\mathrm{A})}\mathrm{adj}(\mathrm{A})\mathrm{X}, \tag{3.9}$$

where adj(A) denotes the adjoint matrix of A. Equation (3.9) implies that matrix A is invertible, or $\det(\mathrm{A}) \neq 0$. When $m > n$, the linear system becomes under-determined, thus solving (3.8) becomes ill-posed. In such a case, recovering the m sources from n mixtures requires extra prior knowledge (e.g., the probability distribution) of the sources. In general, it is mathematically intractable. However, if we aim at extracting one source only instead of recovering all of them, the solution to (3.8) becomes somewhat well-posed.

Next, we consider the ensuing question: In a source extraction task, given a $2\times m$ ($m \geq 2$) linear mixing matrix, which source signal will be extracted among the m candidates pool? To answer that, we resort to linear algebraic analysis. Assume $rank(\mathrm{A}) = 2$ an denote $\mathrm{Q} = \mathrm{AA}^T$ as a 2×2 matrix, then $rank(\mathrm{Q}) = 2$. Given a symmetric matrix AA^T, the condition number is given as: $cond(\mathrm{AA}^T) = \lambda_n/\lambda_1$, where λ_n (here $n = 2$) and λ_1 are the maximum and minimum eigenvalues of the AA^T, respectively. When A is a square matrix, we have $cond(\mathrm{A}) = \sqrt{cond(\mathrm{AA}^T)}$. In the case when $cond(\mathrm{A})$ or $cond(\mathrm{Q})$ is large, the mixing is ill-conditioned, which makes the demixing and segregation difficult. In the transformed data space, given m independent sources, let us simply assume the correlation matrix $R_\mathrm{s} \equiv \mathbb{E}[\mathrm{ss}^T] \approx \frac{1}{N}\mathrm{SS}^T$ (N denotes the length of the source signal) is diagonal. Let $\mathrm{Q}' = \frac{1}{N}\mathrm{XX}^T = \frac{1}{N}(\mathrm{AS})(\mathrm{AS})^T = \frac{1}{N}\mathrm{A}(\mathrm{SS})^T\mathrm{A}^T$, taking an eigenvalue decomposition of Q' yields

$$\mathrm{Q}' = \mathrm{U}\Sigma\mathrm{U}^T \tag{3.10}$$

where U is an orthogonal matrix (i.e. $\mathrm{U}^T = \mathrm{U}^{-1}$) with each column being the eigenvector, and $\boldsymbol{\Sigma}$ is a diagonal matrix: $\boldsymbol{\Sigma} = \mathrm{diag}\{\sigma_1, \sigma_2\}$, where the diagonal elements correspond to the eigenvalues (assuming in an ascending order, $\sigma_2 \geq \sigma_1$). Similarly, we also have $\mathrm{Q} = \mathrm{AA}^T = \mathrm{V}\Lambda\mathrm{V}^T$ and $\mathrm{V}^T = \mathrm{V}^{-1}$. It can be shown that $\Sigma = \mathrm{D}\Lambda$, where $\mathrm{D} = \mathrm{diag}\{d_1, d_2\}$ ($d_2 \geq d_1$), where d_2 and d_1 are two dominant variance values among the m sources; namely, they are the two principal eigenvalues of the correlation matrix R_s. Hence, the principal eigenvalue associated with the principal axis in the transformed data space is determined by the product of the principal eigenvalue of AA^T and the dominant variance value of the source signals. Informally, we have the ensuing proposition:

Proposition. *Let* $\mathrm{X} = \mathrm{AS}$ *be an instantaneous linear mixing matrix, where the row vectors of matrix* S *represent* m *independent sources. If the condition number of* AA^T *is greater than a certain threshold, the separation/extraction task becomes intractable; otherwise, the recovered signal in the source extraction task corresponds to one of the row vectors of* S *whose projection aligns closest with the principal axis in the mixed signal space. By performing an orthogonal rotation on the mixed signal space, it is then possible to recover another signal in the complementary subspace.*

The single source extraction can also be analyzed from a constrained optimization perspective. Again, suppose we have two sensors and m sources; the n source signals, represented as a matrix $\mathbf{S} = [\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_m]^T$, are subject to the instantaneous linear mixing:

$$\begin{aligned} \mathbf{X} &= \mathbf{AS} \\ &= \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \end{bmatrix} \cdot [\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_m]^T \end{aligned} \tag{3.11}$$

which yields the mixed signals received at the two sensors: $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2]^T$. Upon passing the 2×2 demixing matrix $\mathbf{W}$, the outputs $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2]^T$ are represented as

$$\begin{aligned} \mathbf{Y} &= \mathbf{WX} \\ &= \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix}. \end{aligned} \tag{3.12}$$

In expanded form, we write

$$\begin{aligned} \mathbf{y}_1 &= w_{11}(a_{11}\mathbf{s}_1 + a_{12}\mathbf{s}_2 + \cdots + a_{1m}\mathbf{s}_m) + w_{12}(a_{21}\mathbf{s}_1 + a_{22}\mathbf{s}_2 + \cdots + a_{2m}\mathbf{s}_m), \\ \mathbf{y}_2 &= w_{21}(a_{11}\mathbf{s}_1 + a_{12}\mathbf{s}_2 + \cdots + a_{1m}\mathbf{s}_m) + w_{22}(a_{21}\mathbf{s}_1 + a_{22}\mathbf{s}_2 + \cdots + a_{2m}\mathbf{s}_m). \end{aligned}$$

Suppose we wish to extract a single source $\mathbf{s}_i$ at the two outputs; letting $\mathbf{y}_1 = \alpha_1 \mathbf{s}_i$ and $\mathbf{y}_2 = \alpha_2 \mathbf{s}_i$ (where α_1 and α_2 are two positive or negative nonzero scalars), then, in order to obtain perfect single-source (i.e. $\mathbf{s}_i$) extraction, we infer the necessary condition:

$$\begin{aligned} &(w_{11}a_{1i} + w_{12}a_{2i}) \neq 0, \\ &(w_{11}a_{1j} + w_{12}a_{2j})\mathbf{s}_j = 0 \quad (j = 1, \cdots, m; j \neq i), \\ &(w_{21}a_{1i} + w_{22}a_{2i}) \neq 0, \\ &(w_{21}a_{1j} + w_{22}a_{2j})\mathbf{s}_j = 0 \quad (j = 1, \cdots, m; j \neq i). \end{aligned}$$

If $\mathbf{s}_j \neq 0 \;\; (j = 1, \cdots, m; j \neq i)$, then it follows the equivalent condition:

$$\begin{aligned} &(w_{11}a_{1i} + w_{12}a_{2i}) \neq 0, \\ &(w_{11}a_{1j} + w_{12}a_{2j}) = 0 \quad (j = 1, \cdots, m; j \neq i), \\ &(w_{21}a_{1i} + w_{22}a_{2i}) \neq 0, \\ &(w_{21}a_{1j} + w_{22}a_{2j}) = 0 \quad (j = 1, \cdots, m; j \neq i). \end{aligned}$$

It is noted that when $m > 2$, this set of linear equations (involving 4 unknowns and $2m$ variables) are under-determined; namely, there is no solution for $\{w_{11}, w_{12}, w_{21}, w_{22}\}$ to satisfy an arbitrary choice of linear mixing matrix $\mathbf{A} = \{a_{ij}\}$.

Then the optimization problem can be written as:

$$\begin{aligned} &\arg\min_{w_{11}, w_{12}} \left\{ \sum_{j=1; j\neq i}^{m} |w_{11}a_{1j} + w_{12}a_{2j}| \right\} \quad \text{s.t.} \;\; w_{11}a_{1i} + w_{12}a_{2i} \neq 0 \\ &\arg\min_{w_{21}, w_{22}} \left\{ \sum_{j=1; j\neq i}^{m} |w_{21}a_{1j} + w_{22}a_{2j}| \right\} \quad \text{s.t.} \;\; w_{21}a_{1i} + w_{22}a_{2i} \neq 0, \end{aligned}$$

which can be understood as a form of maximizing signal-to-inference ratio. The gradient-free ALOPEX procedure seems to perform quite well for this constrained optimization problem given an appropriate setup of mixing matrix **A**. More analysis and discussion regarding the biological implications are presented in a forthcoming technical report (Chen and Haykin, 2004).

Chapter 4

Perceptual Learning for Neural Compensation

"A model is the more impressive the greater the simplicity of its premises."
— Albert Einstein

4.1 Background

Neural compensation (Becker and Bruce, 2002) was motivated by the design of adaptive hearing-aid devices for hearing-impaired persons. The goal of the Neurocompensator is to restore near-normal firing patterns in the auditory nerve in spite of the hair cell damage in the inner ear. A schematic diagram of normal/impaired hearing systems as well as the neural compensation is illustrated in Figure 4.1. Ideally, the Neurocompensator attempts to match the output of the compensated system, as closely as possible, to the output of the normal hearing system.

The early development of the Neurocompensator was described in (Bondy et al., 2004). In the previous work, the outputs of the normal and damaged models are compared directly at the level of the raw spike trains. However, auditory nerves have high spontaneous firing rates, and when driven by auditory input, convey predominantly steady-state information, whereas the transient information is most critical to speech perception. Beside, the algorithm in (Bondy et al., 2004) was tested on vowel sounds, which are relatively steady-state. Here, we apply a transient detection procedure to the auditory nerve spike trains to simulate higher levels of auditory processing, and we train and test the model on continuous speech containing both voiced and unvoiced components. Also, in the previous work, an ad hoc perturbation-like optimization procedure was used to learn the Neurocompensator parameters with a simple error metric. Moreover, it does not provide a probabilistic measure of how well the Neurocompensator compensates for the hearing loss, neither does it present an informative comparative metric between the compensated

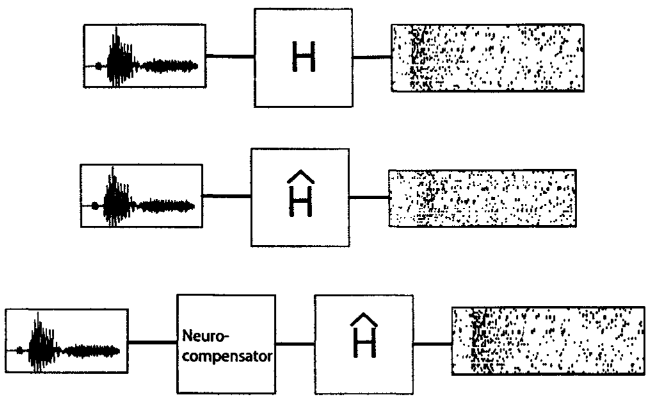

Figure 4.1: A schematic diagram of Neurocompensation. *Top:* normal hearing system. *Middle:* impaired hearing system. *Bottom:* Neurocompensator followed by the impaired hearing system. The H and Ĥ denote the input-output mappings in the normal and impaired ear models, respectively.

Table 4.1: Selected speech samples used in the experiments.

Speech Sample	Content
TIMIT-1	/The emperor had a mean temper./
TIMIT-2	/His scalp was blistered by today's hot sun./
TIMIT-3	/Would a tomboy often play outdoor?/
TIMIT-4	/Almost all of the colleges are now coeducational./
TIDIGITS-1	/one/
TIDIGITS-2	/one, two/
TIDIGITS-3	/nine, five, one/
TIDIGITS-4	/eight, one, o, nine, one/

and the normal hearing systems. It is our goal in this chapter to formulate a principled methodology and improve the optimization efficiency.

4.2 Methodology

Data. The audio data presented to the ear models can be either speech or natural sound. In our experiments, the speech data are selected from both the TIMIT and TIDIGITS databases. From the TIMIT database, a total of ten spoken sentences by different male and female speakers are used for the simulations reported here; the sample frequency of the speech data is 16kHz. In the TIDIGITS database, the data consist of English spoken digits (in the form of isolated digits or multiple-digit sequences) recorded in a quiet environment, with sample frequency 8kHz. All of the experimental data were subjected to resampling preprocessing (to 16kHz if applicable) prior to being presented to the auditory models. Some of the speech samples used in the experiments are listed in Table 4.1. Ideally, all of

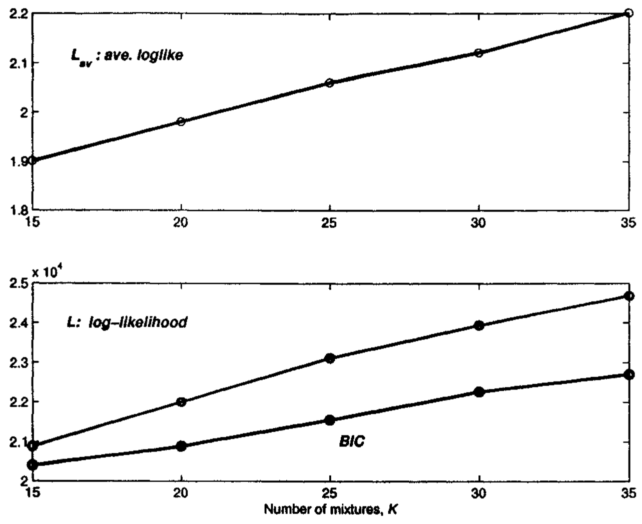

Figure 4.2: The averaged and joint log-likelihood and the BIC parameters against different numbers of mixtures, averaging on different trials for one set of spike-trains data.

the speech samples are truncated to with in the same length.

Auditory Models. The auditory peripheral model used here is based on the earlier work of Bruce and colleagues (Bruce et al., 2003). In particular, the model consists of a middle-ear filter, time-varying narrow- and wide-band filters, inner hair cell, outer hair cell, synapse model, and spike generator, describing the auditory periphery path from the middle ear to the auditory nerve. More recently, a new middle ear model and a new saturated exponential synapse-gain control have been incorporated into that model.[1] The hearing-impaired version of the model described in detail in (Bondy et al., 2004) simulates a typical steeply sloped high frequency hearing loss.

We further process the auditory representation generated by the auditory nerve models by applying an onset detection procedure (Bondy et al., 2003), consisting of a derivative mask with rectification and thresholding (see Section 4.5). This removes much of the noisy spontaneous spiking and high degree of steady-state information in the signal-driven spike trains. The resultant binary image is used here as the basis for comparing the neural codes generated by the normal and impaired models.

Probabilistic Modeling. In order to compare the neural codes of the normal and impaired models, we characterized the binary image by its probability density function (pdf) or probability mass function (pmf). To overcome the inherent noisiness of the spike-generating and onset detection processes, we chose a two-dimensional mixture of

[1]The detailed information of the auditory peripheral models is referred to Dr. Ian C. Bruce's website http://www.ece.mcmaster.ca/~ibruce/.

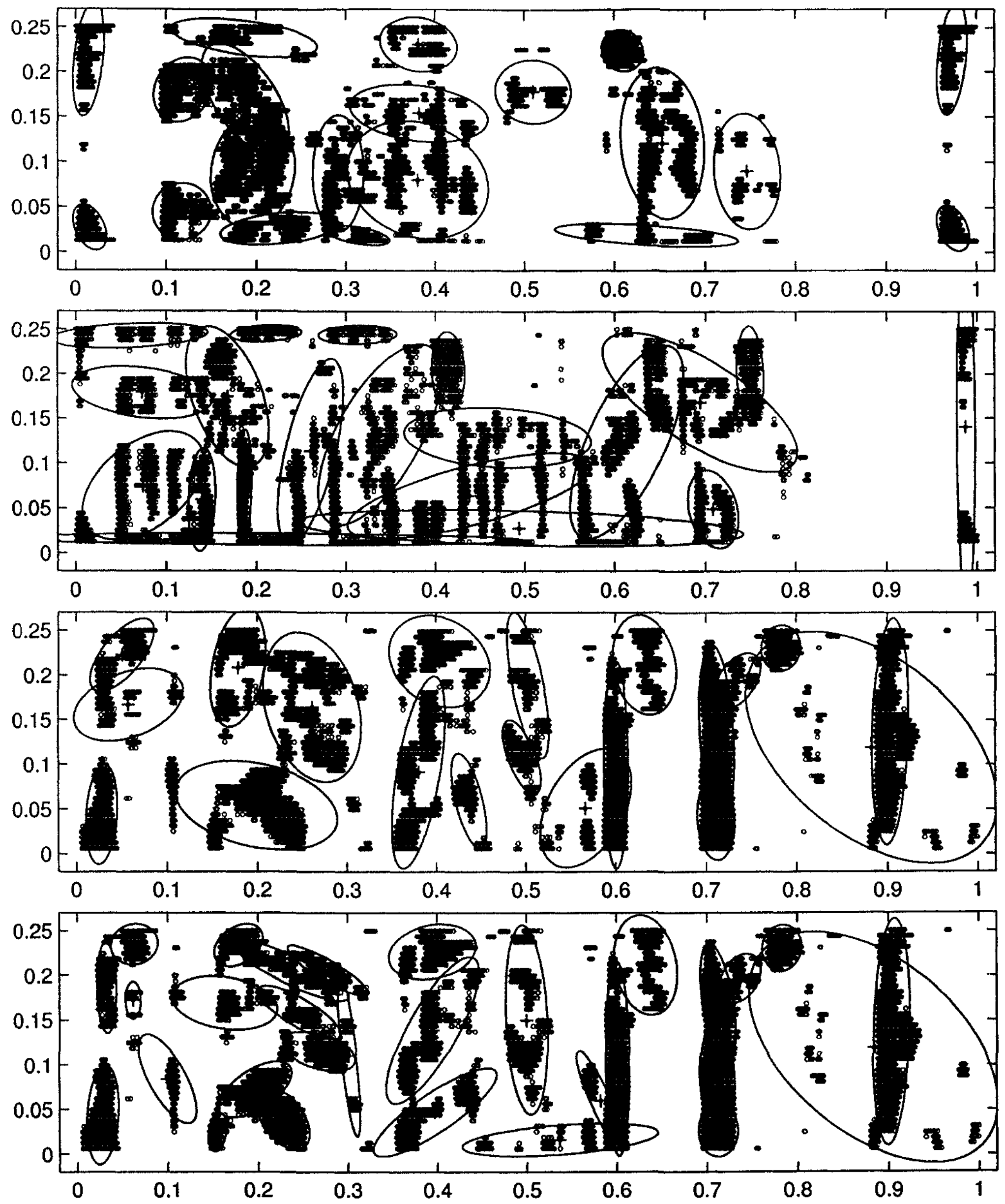

Figure 4.3: Three selected sets of spike-trains data (with scale ratio 0.25) calculated from the normal hearing model and their probabilistic fittings using 20 (the first three plots) or 30 (the fourth plot) Gaussian mixtures. For the third plot, $\mathcal{L} = 22009, \mathcal{L}_{av} = 1.97$, and $BIC(20) = 20891$; for the fourth plot, $\mathcal{L} = 23942, \mathcal{L}_{av} = 2.14$, and $BIC(30) = 22264$. It is evident that the fourth plot is a better fit than the third one.

Gaussians to characterize this distribution, given its spatial smoothing property across the spectral-temporal plane. Suppose that $D_1 \equiv \{\mathbf{x}_i\}_{i=1}^{\ell}$ and $D_2 \equiv \{\mathbf{z}_i\}_{i=1}^{\ell'}$ denote the neural codes (i.e. the onset binary images) that are calculated from the normal and

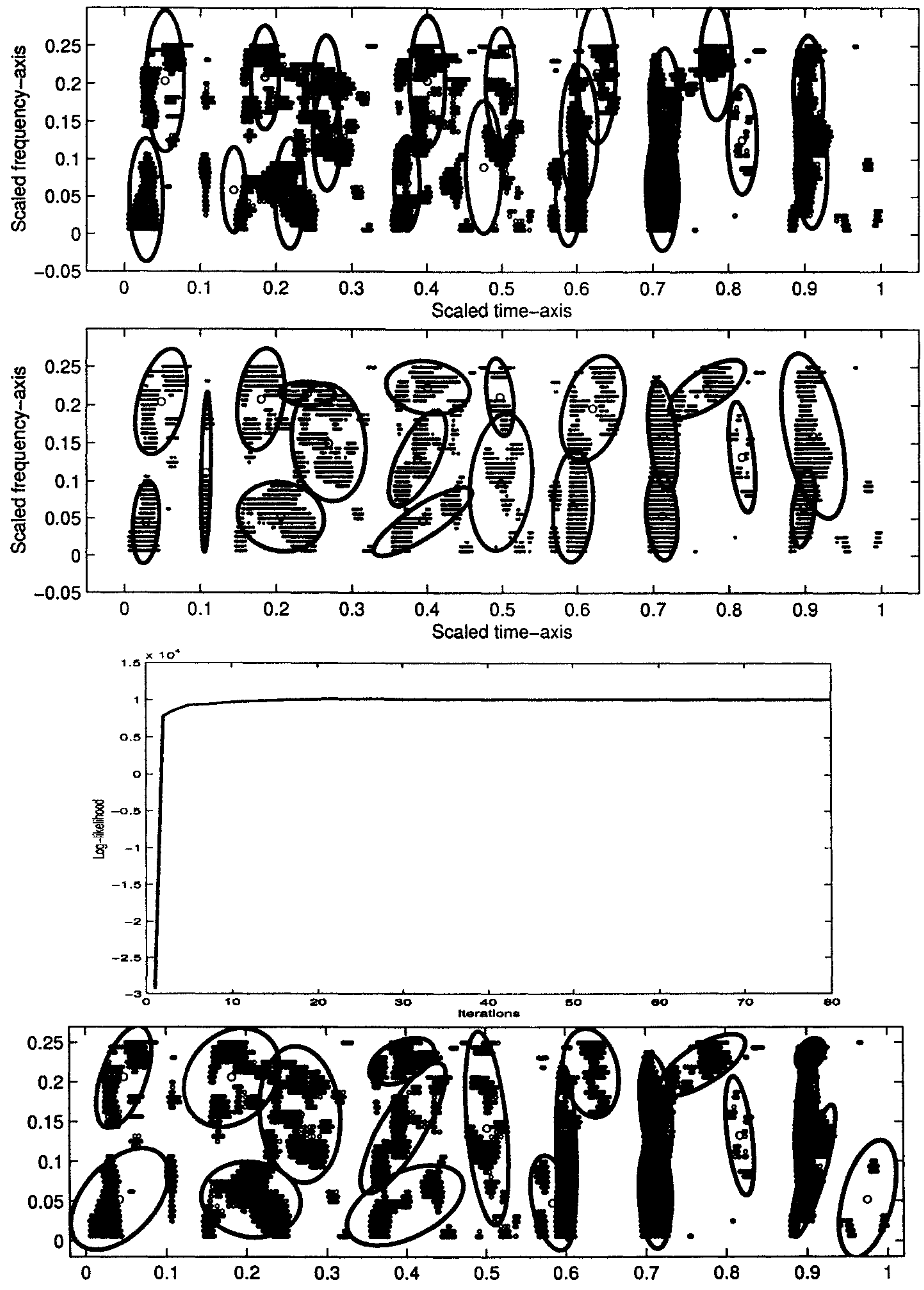

Figure 4.4: From top to bottom: (a) The initialized 20 Gaussian mixtures via K-means clustering. (b) The Gaussian mixture fitting after 80 iterations of the EM algorithm. (c) The log-likelihood convergence curve. (d). Another fitting result obtained from a different initial condition.

impaired hearing models (Bruce et al., 2003), respectively.[2] Assume that $p(D_1|M)$ is a probabilistic model that characterizes the data D_1, when M here is represented by a

[2]Note that in general, $\ell \neq \ell'$.

Gaussian mixture model, i.e. $M \equiv \{c_j, \boldsymbol{\mu}_j, \Sigma_j\}_{j=1}^K$.

Note that $D_1 \equiv \{\mathbf{x}_i\} \in \mathbb{R}^d$ are calculated from the normal ear model (with input-output mapping H) given the audio or speech data; suppose the data $\mathbf{x}$ are drawn from a two-dimensional ($d = 2$) mixture of Gaussian density:

$$\begin{aligned} p(\mathbf{x}) &= \sum_{j=1}^{K} p(j)p(\mathbf{x}|j) \\ &= \sum_{j=1}^{K} c_j \frac{1}{\sqrt{(2\pi)^d|\Sigma_j|}} \exp\Big(-\frac{1}{2}|\mathbf{x}-\boldsymbol{\mu}_j|^T \Sigma_j^{-1} |\mathbf{x}-\boldsymbol{\mu}_j|\Big), \end{aligned} \tag{4.1}$$

where c_j is the prior probability for the jth Gaussian component, with mean $\boldsymbol{\mu}_j$ and covariance matrix Σ_j^{-1}. With the assumption of i.i.d. (independent, identically distributed) samples,[3] we can calculate the joint likelihood of the data given the mixture model M:

$$p(D_1|M) = \prod_{i=1}^{\ell} p(\mathbf{x}_i). \tag{4.2}$$

Alternatively, we can calculate the log-likelihood

$$\mathcal{L} = \log p(D_1|M) = \sum_{i=1}^{\ell} \log p(\mathbf{x}_i), \tag{4.3}$$

and the associated average log-likelihood $\mathcal{L}_{av} = \mathcal{L}/\ell$.

In order to account for the model complexity, we can use penalized maximum likelihood incorporating a complexity metric such as the minimum description length (MDL) or Bayesian information criterion (BIC). For instance, for a K-mixture model, the BIC is defined as

$$BIC(K) = \sum_{i=1}^{\ell} \log p(\mathbf{x}_i|\boldsymbol{\theta}) - \frac{\ell_K}{2} \log \ell, \tag{4.4}$$

where ℓ_K represents the total number of free parameters in the model; in the mixture of Gaussians case,

$$\ell_K = K\Big(1 + d + \frac{d(d+1)}{2}\Big). \tag{4.5}$$

Figure 4.2 shows a comparison of different metrics for varying the number of mixture components.

The clustering is fitted via a mixture of elliptical Gaussians using the EM algorithm (see Appendix A). It is known that the EM algorithm is only guaranteed to converge

[3]Note that this assumption is not strictly valid for our spike-trains data.

monotonically to a local minimum or saddle point. In our early investigations (Gupta, 2004), several empirical findings were observed: (i) It is necessary to rescale the time and frequency ranges for better Gaussian mixture fitting; an optimal scale ratio of 0.25 applied to the *normalized* time-frequency coordinate is suggested; namely, the time-axis is constrained within the region [0,1], whereas the frequency-axis is within the region [0,0.25]. This is tantamount to scaling the variance of the coordinates and compressing the data in terms of their distance, which gives an advantageous ease for probabilistic fitting. (ii) For the spike-trains onset map, a total of 20 $\sim$ 30 mixtures of elliptical Gaussians is sufficient to characterize the data distribution (see Figure 4.3), although the optimal number of mixtures varies from one data set to another. For simplicity, a fixed number of mixtures determined empirically is assumed throughout our experiments, though it is realized that this is not a principled solution. In addition, Gaussian mixture fitting via the EM algorithm is well known to be sensitive to the initialized (mean and covariance) parameters (see Figure 4.4 for an illustration), both for the convergence speed and log-likelihood performance. With a better initialization scheme compared to (Gupta, 2004), we use the K-means clustering method (e.g., Duda et al., 2001) to initialize the mean parameters to accelerate the convergence. We found that 10 to 20 iterations of the batch EM algorithm produce reasonable fitting results for all data used thus far.

Spectral Enhancement. Spectral enhancement is achieved by neural compensation through the Neurocompensator. The principle of the Neurocompensator is to control the spectral contrast via the gain coefficients using the idea of divisive normalization (Schwartz and Simoncelli, 2001). In particular, the gain coefficient, G_i, at the ith frequency band, is calculated as

$$G_i = \frac{||f_i||^2}{\sum_j v_{ji}||f_j||^2 + \sigma}, \tag{4.6}$$

where i and j represent the indices of the frequency bands; G_i is a *nonlinear* function of the weighted input (frequency) power, $||f_i||^2$, divided by the weighted sum of all the frequencies' power; σ is a regularization constant that ensures that the gain G_i does not go to infinity. Applying gain coefficients to frequency bands is tantamount to implementing a bank of linear filters. The divisive normalization was originally aimed at suppressing the statistical dependency between the filters' responses (Schwartz and Simoncelli, 2001). Here, we employ a similar functional form, but rather than adapting the normalization coefficients to optimize information transmission, we adapt the parameters to optimize a measure of similarity between the codes generated by the two models.

For the present purpose, we propose a slightly different version of (4.6) as follows:

$$G_i = h\left(\frac{w_i||f_i||^2}{\sum_j v_{ji}||f_j||^2 + \sigma}\right), \quad \text{where} \quad w_i \propto G_i^{NAL-RP}, \tag{4.7}$$

where G_i^{NAL-RP} represents a positive coefficient based on NAL-RP (National Acoustics Laboratories—Revised Profound), a standard hearing-aid fitting protocol (Byrne et al.,

1990) that can be calculated from the ith frequency band (see Bondy et al., 2004); and $h(\cdot)$ is a continuous, smooth (e.g., sigmoid) function that constrains the range of the gains as well as assures the gains to vary smoothly in time. When $h(\cdot)$ is linear and $G_i^{NAL-RP} = 1$, equation (4.7) reduces to (4.6). On the other hand, when all $v_{ji} = 0$ and $h(\cdot)$ is linear, equation (4.7) reduces to the standard, fixed linear gain NAL-RP algorithm. For the hearing aid application, it is appropriate to constrain $G_i \geq 0$.[4] Now, the goal of the learning procedure is to find the optimal parameters $\{v_{ji}\}$ that compensate the hearing impairment or intelligibility according to a certain performance metric. Note that those parameters account for some psychophysical meaning, it is *not* a purely unconstrained optimization problem; rather it is subject to certain implicit constraints. For instance, we expect that for a fixed frequency bin j, $\{v_{ji}\}$ has an "on-center off-surround" effect; the gain coefficients G_i should be nonnegative, bounded, and varying smoothly over a short period of time. It is important to note that, unlike the traditional hearing aid algorithms, the parameters to be optimized are *not* independent, in the sense that the cross-frequency interference may cause modifying one parameter to indirectly affect the optimality of the others. All of these issues make the learning of the Neurocompensator a hard optimization problem, and the solution might not be unique. In our early investigations (Bondy et al., 2004), the optimization procedure and the error metric used therein were quite ad hoc, and certain instability during the optimization was also observed. Here, we attempt to recast this optimization problem in a more principled way.

Optimization. Let $\boldsymbol{\theta} \equiv \{v_{ji}\}$ denote the vector that contains all of the parameters to be estimated in the Neurocompensator. Let $D_2 = \{\mathbf{z}_i\}$ denote the data calculated from the deficient ear model (with input-output mapping $\hat{\mathrm{H}}$), after preprocessing the audio (speech) with the Neurocompensator parameterized by $\boldsymbol{\theta}$. Let $p(D_2|M, \boldsymbol{\theta})$ be the marginal likelihood of the impaired model's spike trains having been generated by a normal model, then the associated log-likelihood can be written as

$$\begin{aligned}\mathcal{L}'_{av} &= \frac{1}{\ell'} \log p(D_2|M, \boldsymbol{\theta}) = \frac{1}{\ell'} \log \Big(\prod_{i=1}^{\ell'} \sum_{k=1}^{K} c_k \mathcal{N}(\boldsymbol{\mu}_k, \Sigma_k; \mathbf{z}_i) \Big) \\ &= \frac{1}{\ell'} \sum_{i=1}^{\ell'} \log \Big(\sum_{k=1}^{K} c_k \mathcal{N}(\boldsymbol{\mu}_k, \Sigma_k; \mathbf{z}_i) \Big),\end{aligned}$$

where M is a Gaussian mixture model fitted to the normal hearing model's output, D_1, by maximizing $\log p(D_1|M)$, which can be optimized off-line as a preprocessing step. One way of optimizing the Neurocompensator would be to maximize $\mathcal{L}'_{av}$ with respect to $\boldsymbol{\theta}$; however, directly maximizing it may cause a "saturation", since the number of points in D_2, ℓ', might grow over ℓ.[5] A better objective function that does not suffer this pitfall, is the Kullback-Leibler (KL) divergence between the probability of observing the impaired

[4]The case $G_i < 0$ has an effect of phase reversal to the frequency domain.

[5]This has been confirmed in our experiments. The worst case of the "saturation" effect will be that $\{\mathbf{z}_i\}$ are uniformly distributed across the whole spike-trains map.

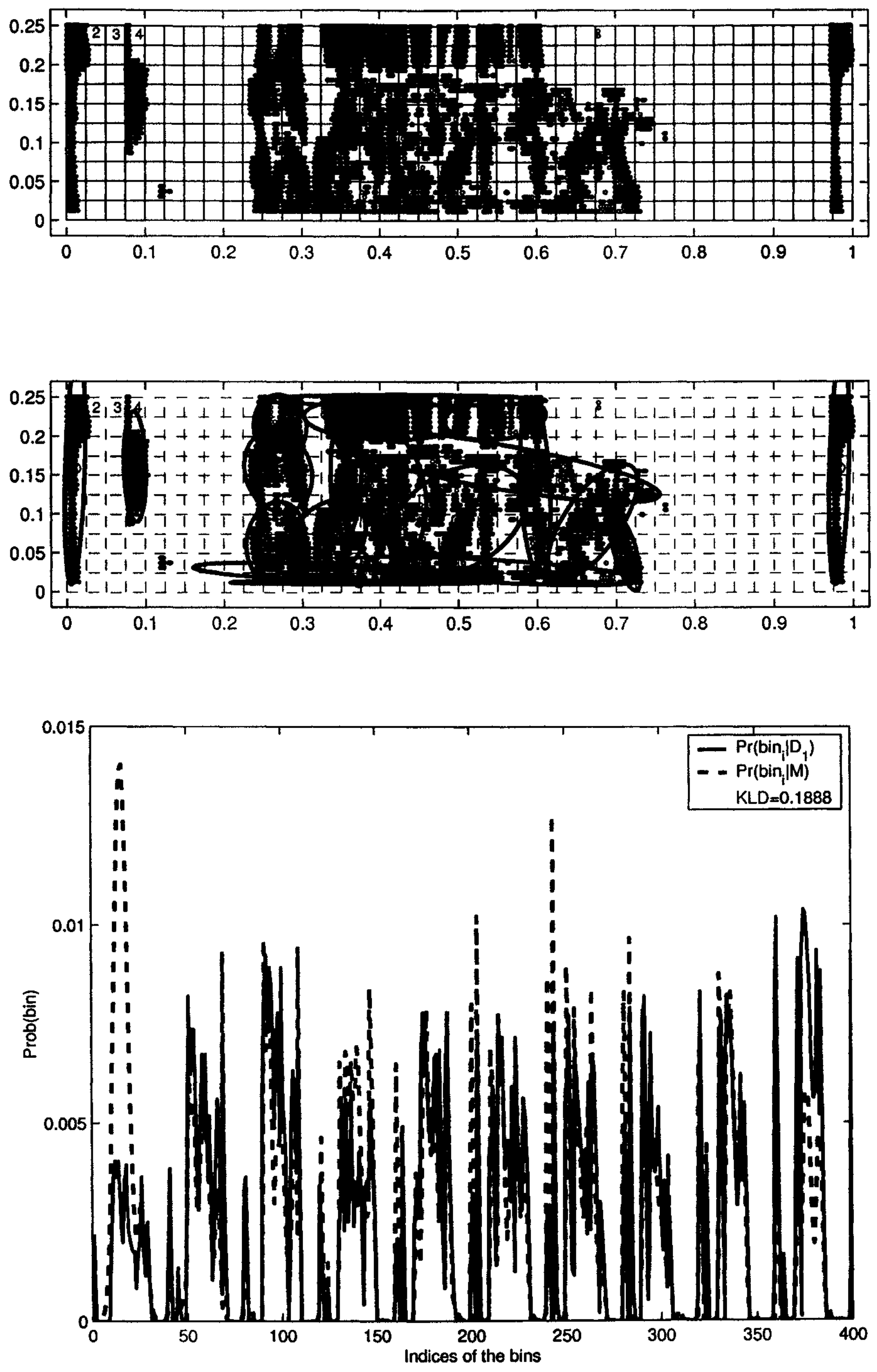

Figure 4.5: A grid quantization (*top panel*) compared with a Gaussian mixture fitting (*middle panel*) on the spike-trains map. Each map contains $40 \times 10 = 400$ bins; the Roman numbers inside the bins indicate their respective indices. *Bottom panel:* the approximation comparison between $p_1 = p(bin_i|D_1)$ and $p_2 = p(bin_i|M)$ $(i = 1, \cdots, 400)$, $\mathrm{KL}(p_1\|p_2) = 0.1888$.

model's output under the normal versus impaired density function. Unfortunately, calculating the latter is much more costly, because it must be done repeatedly, interleaved with optimization of the Neurocompensator parameters $\boldsymbol{\theta}$. We therefore consider a discrete sampling approach to estimate this density, which is computationally simpler than fitting a Gaussian mixture model.

Specifically, we quantize or discretize evenly the spike-trains onset map into a number of bins, where each bin contains zero or more of the spikes. To quantitatively measure the discrepancy between the normal spike-trains and reconstructed spike-trains maps, we calculate the probability of each bin that covers the spikes; this can be easily done by counting the number of the spikes in the bin and further normalizing by the total number of the spikes in the whole spike-trains map. In particular, the objective function to be *minimized* is a quantized form of the KL divergence:

$$E \equiv \mathrm{KL}(D_2 \| D_1) = \sum_{i}^{\#bins} p(bin_i|D_2) \log \frac{p(bin_i|D_2)}{p(bin_i|D_1)}, \tag{4.8}$$

where $p(bin_i|D_1)$ and $p(bin_i|D_2)$ represent the probabilities of the ith bin that contains the spikes in the normal and reconstructed spike-trains maps, respectively. Note that $p(bin_i|D_1)$ can be calculated (only once) in the preprocessing step. In our experiment, we quantize evenly the spike-trains map into a (40-time)×(10-frequency) mesh grid (see Figure 4.5 for illustration), with a total number of 400 bins.

However, equation (4.8) suffers from two drawbacks: (i) For some bins, the denominator $p(bin_i|D_1)$ can be zero, thereby causing a numerical problem. (ii) There is no smoothing between two discrete maps, hence it will suffer from the noise in the spiking and/or onset detection processes. Fortunately, since we have the Gaussian mixture probabilistic fitting for D_1 at hand, this can provide a spatial smoothing across the neighboring (time and frequency) bins, thereby counteracting the noise effect. To overcome the above two problems, we therefore substitute $p(bin_i|D_1)$ (quantized version) with $p(bin_i|M)$ (continuous version), where $p(bin_i|M)$ is calculated by fitting the center point in the ith bin with the Gaussian mixture model M, divided by a normalization factor: $\sum_j p(bin_j|M)$ (see Figure 4.5 bottom panel for illustration).[6] To do so, we modify (4.8) to obtain another objective function:

$$E \equiv \mathrm{KL}(D_2 \| M) = \sum_{i}^{\#bins} p(bin_i|D_2) \log \frac{p(bin_i|D_2)}{p(bin_i|M)}. \tag{4.9}$$

Note that $p(bin_i|M)$ is usually a nonzero value due to the overlapping Gaussian covering, although it can be very small.[7] As before, $p(bin_i|M)$ can be calculated in the preprocessing

[6]To see how close the approximation is, we calculate the KL divergence in the example of Figure 4.5: $\sum_{i=1}^{400} p(bin_i|D_1) \log \frac{p(bin_i|D_1)}{p(bin_i|M)} = 0.1888$.

[7]To avoid the numerical problem in practice, we add a very small value (10^{-16}) to the denominator to prevent overflowing.

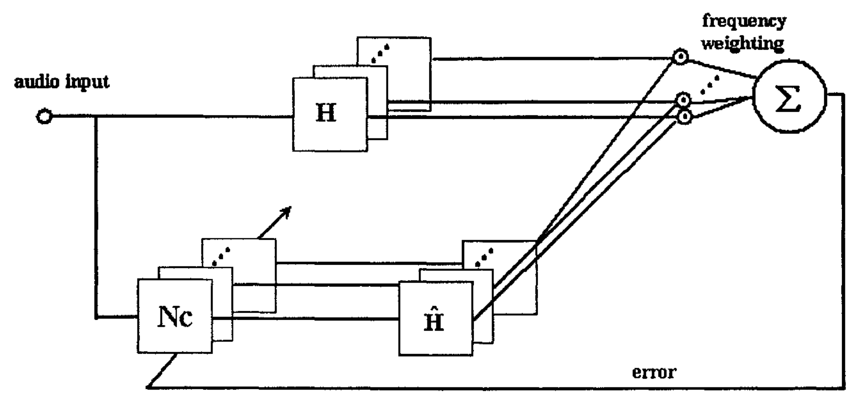

Figure 4.6: Block diagram of training the Neurocompensator (Nc). The normal (H) and impaired (Ĥ) auditory models' output is a set of the spike trains at different best frequencies, when are then subjected to an onset detection process, while the Neurocompensator is represented as a preprocessor which calculates gains for each of the different frequencies. The error (denoted by a sum of frequency-weighted signals) is actually the KL divergence between the probability distributions of the two models' outputs.

step. When $p(bin_i|D_2) = p(bin_i|M)$, it follows that $E = 0$, otherwise E is a nonnegative value given $0 \leq p(bin_i|D_2) < 1, 0 \leq p(bin_i|M) < 1$.[8] Since the probability $p(bin_i|D_2)$ can be zero, we have assumed that $0 \log 0 = 0$.

It is noted that direct calculation of the gradient $\frac{\partial E}{\partial \theta}$ in either (4.8) or (4.9) is inaccessible due to the characteristics of the ear model as well as the form of the objective function, hence we can only resort to gradient-free optimization, as discussed previously in Chapter 2. During the training phase, the gain coefficients are adapted to minimize the discrepancy between the reconstructed and the original spike trains (see Figure 4.6).

Thus far, the complete learning procedure is summarized as follows:

1. Initialize the parameters: $\{v_{ji}\} \in \mathcal{U}(-0.5, 0.5)$, $\sigma = 0.001$; randomly select one speech sample.

2. Load the selected speech data, the associated spike-trains fitting mixture parameters $M \equiv \{c_i, \mu_i, \Sigma_i\}$, and the probability $p(bin_i|M)$, the latter two of which are precalculated off-line.

3. Apply the short-term Fourier transform (STFT) to the speech data (128-point FFT with a 64-point overlapping Hamming window[9]); the results of time-frequency analysis then provide the temporal-spectral information across frequency bands.[10]

4. Apply the gain coefficients θ to the frequency bands according to (4.7), perform inverse Fourier transform to reconstruct the time-domain waveform.

[8]It seems difficult to derive the upper-bound of the objective function (4.9).

[9]For 16kHz sampling frequency, it corresponds to a duration of 8 ms.

[10]Depending on the frequency resolution requirement, the number of frequency bands can vary from 20 to 40; we use 20 frequency bands in the experiments.

5. Present the reconstructed waveform to the hearing-impaired ear model, produce a "Neuro-compensated" spike-trains map.

6. Using the quantized approximation to the hearing-impaired data probability density, and the precalculated Gaussian mixture model, calculate the objective function (4.9).

7. Apply the ALOPEX-like algorithm to optimize θ.

8. Repeat Steps 3 through 7 for a certain number of iterations.

9. Select another speech sample, repeat Step 2 through 8. Repeat the whole procedure until convergence.

As far as Step 7 in the optimization procedure is concerned, two kinds of optimization schemes can be considered:

- Synchronous optimization: namely, all of gain coefficients are treated with no difference; all the parameters are updated in parallel across different frequency bands. This scheme is simple, but due to the cross-frequency interdependence of the coefficients, it can be very slow, given a poor parameter initialization.

- Asynchronous optimization: namely, the gain coefficients in different frequency bands are treated differently and optimized sequentially with different priority. Starting with the highest frequency band, all the other parameters associated with the lower frequency bands are set as zeros, update only the parameters associated with the high-frequency band. Then freeze these parameters, switch to a lower (i.e., the second highest) frequency band, repeat the optimization, and so on. For each frequency band, the optimization stopping criterion is empirically set as repeating $10 \sim 15$ iterations. This sequential optimization can be justified by virtue of the fact that in a hearing-impaired system, it is the lower frequencies that tend to interfere with the detection of higher frequencies, and not the converse.

4.3 Experimental Results

To reduce the computational burden, we have consistently used a fixed number ($K = 20$) of Gaussian mixtures for fitting all of spike-trains data. We present results here based on the training speech samples listed in Table 4.1, totaling about 14.1 seconds of continuous speech.

We apply the improved version of the ALOPEX-B algorithm (Section 2.3.5) for optimization, where the objective function to be minimized is (4.9). Figure 4.7 shows the performance metric curve using the synchronous optimization scheme. We have not extensively investigated the asynchronous optimization scheme; but it was observed in an empirical test that the inappropriate initialization may cause unstable performance. For

Table 4.2: Training and testing results of the experimental data in Table 4.1. The last column indicates the approximation accuracy between the quantized pmf and continuous Gaussian mixture pdf on the neural codes obtained from the normal hearing system; it can be roughly viewed as a lower-bound for the values in the third and fourth columns. The second/third columns show the KL divergence of (4.9) before/after using the Neurocompensator; the numbers in bold font indicate the training results.

Speech Sample	$\mathrm{KL}_{init}(D_2 \| M)$	$\mathrm{KL}_{final}(D_2 \| M)$	$\mathrm{KL}_{final}(D_2 \| D_1)$	$\mathrm{KL}(D_1 \| M)$
TIMIT-1	1.2058	**0.4462**	1.2828	0.1885
TIMIT-2	0.6152	0.4697	1.9255	0.2493
TIMIT-3	0.6692	0.6105	1.7367	0.2741
TIMIT-4	0.6477	0.4666	1.8329	0.2743
TIDIGITS-1	1.0626	0.1798	0.5591	0.0547
TIDIGITS-2	1.0234	0.4345	1.5918	0.1634
TIDIGITS-3	0.4913	0.2013	0.5759	0.0871
TIDIGITS-4	0.6346	**0.2599**	0.3757	0.1888

this reason, we have restricted ourselves here to the synchronous optimization scheme. Figure 4.8 illustrates the final parameters of the Neurocompensator.

Note that finding the optimal $\boldsymbol{\theta}$ from normal spike-trains is an ill-posed inverse problem,[11] hence it is impossible to build a perfect inverse model. However, it is hoped that the reconstructed spike-trains image from the compensated hearing-impaired model is close to the one from the normal hearing model after the learning the Neurocompensator. Figure 4.9 shows the comparisons between the *normal, deficient*, and *Neurocompensated* waveforms and spike-trains maps of the training speech sample.

Upon completion of the training process, we freeze $\boldsymbol{\theta}$ and further test the Neurocompensator on some unseen speech samples. The training and testing KL divergence results of the experimental data are summarized in Table 4.2. Two testing results on two spoken speech signals are shown in Figure 4.10; it is seen that the Neurocompensated spike-trains maps are reasonably close to the normal ones, though not perfect. This is quite encouraging, given the fact that we have only used about 3.7 seconds of speech for training here; ideally, given sufficient computational power, we should use as many speech samples as possible for training. It is hoped that by averaging across more speech samples (with different contexts, speakers, spoken speeds, etc.), the learning process can yield a more accurate and robust solution.

4.4 Discussion

In this chapter, we have described a novel methodology for training a Neurocompensator, an ingredient of a learning-based, intelligent hearing-aid device. The whole learning pro-

[11]This is because the solution is neither unique nor stable (due to the noise involved).

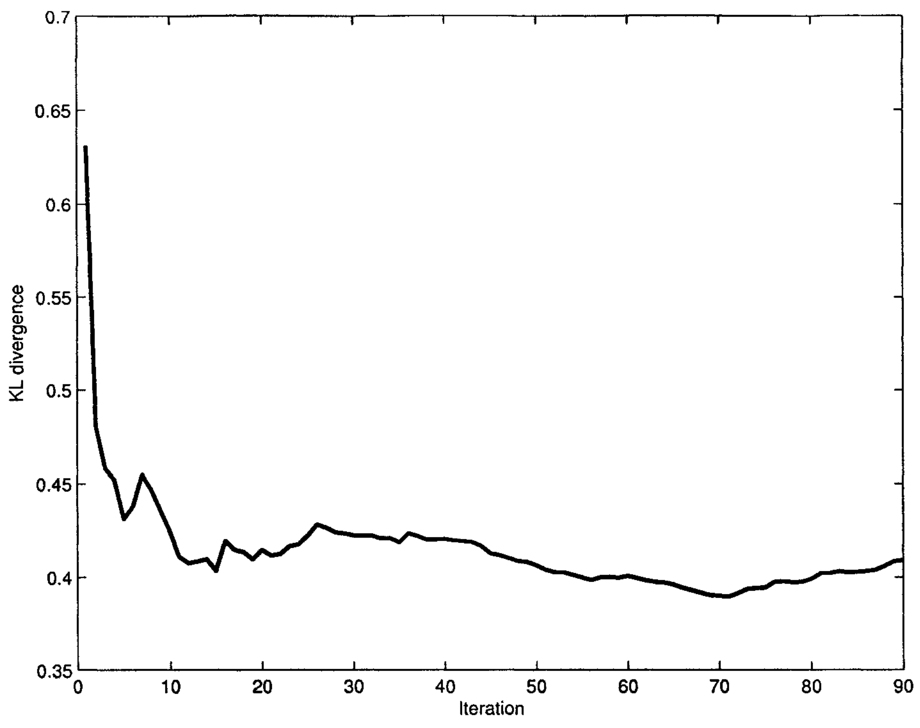

Figure 4.7: Learning curve of one speech sample using synchronous optimization. The KL divergence starts with 0.63 and stays around 0.4 after 90 iterations.

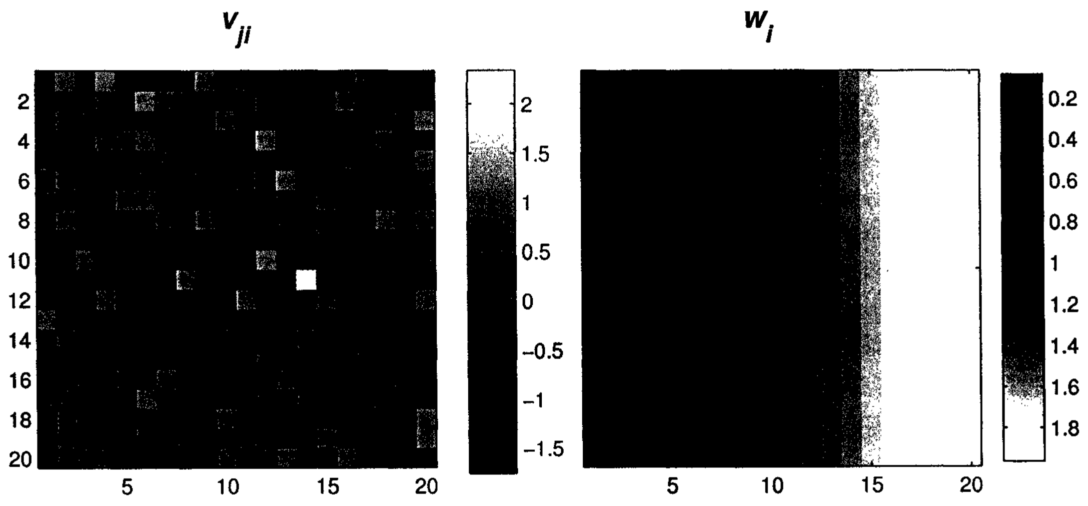

Figure 4.8: Visualization of the learned-weights $\{v_{ji}\}$ and the fixed-weights $\{w_i\}$ of the Neurocompensator. The learned parameters $\{v_{ji}\}$ are displayed in a 20-by-20 matrix, with each column representing the weights associated with the 20 frequency bands.

cess is achieved by (i) probabilistic modeling of auditory nerve model' spike trains and (ii) a gradient-free optimization procedure for parameter update. Based on our empirical experiments, it has been shown that the Neurocompensator provides a promising approach to adaptive compensation for reducing perceptual distortion due to hearing loss.

We have observed some problems with our current approach. In particular, we have found in the experiments: (i) The optimization solution is non-unique. As seen from Figure 4.9, there are still obvious differences between the normal and Neurocompensated spike-trains maps. We suspect that constraining the solution space and incorporating prior knowledge might somewhat alleviate this issue. (ii) The parameters are somewhat training data-dependent. In other words, one set of Neurocompensator parameters good for one speech sample does *not* necessarily produce a similarly good performance for another one (see Table 4.2). This problem should be somewhat alleviated by averaging across more training samples. Another solution to this problem may be to train a mixture of Neurocompensator modules adapted to different input statistics, such as different talkers under varying listening conditions. One could then use a trained classifier to select the best Neurocompensator for the current context.

One obvious weakness here is the use of a fixed number of mixtures for different spike-trains image data. In order to alleviate the computational burden of our procedure and focus on the optimization part, we have neglected to consider model selection in our probabilistic modeling. In the literature, however, there are some principled ways, such as Bayesian approaches (Roberts et al., 1998; Attias, 2000), merging-splitting approach (Ueda et al., 2000), or greedy approach (Verbeek et al., 2003), to tackle this issue.

Another important area for future investigation is the design of the gain function (4.7). We have found that the form of the gain function (e.g., the range and the shape of function $h(\cdot)$) has a vital effect on the optimization performance, particularly on the speed of convergence. The possibility of incorporating prior knowledge or adding constraints to the gain function might also accelerate the convergence speed of optimization. How to design an optimal form of the gain function remains a yet-unsolved problem.

After further development of our algorithm, the ultimate test of its efficacy will be to conduct human hearing tests. The hearing-impaired person(s) will listen to the reconstructed speech waveform yielded from the hearing-aid device (i.e. Neurocompensator) and compare the intelligibility quality with and without the hearing compensation. Note that once the training is accomplished, the hearing test requires no additional computational effort and is easily performed. Furthermore, once the Neurocompensator parameters are optimized, the algorithm represented by equation (4.7) could be straightforwardly and efficiently implemented in a digital hearing-aid circuit.

4.5 Appendix on Onset Map Generation

The spike-trains onset stimuli are used in our experiments for perceptual grouping. For completeness, the onset map generation procedure (Bondy et al., 2003) is briefly described here.

Onset characteristics of the auditory perception are calculated with a difference of exponential filters, $h_1[n]$, in each frequency band:

$$h_1[n] = \frac{n}{\alpha_1^2}\exp(-n/\alpha_1) - \frac{n}{\alpha_2^2}\exp(-n/\alpha_2), \tag{4.10}$$

where α_1 and α_2 are selected to pass frequencies from 4 to 32 Hz. These frequencies contribute most to intelligibility, with a signal's fine temporal structure only adding a small amount to intelligibility (Drullman et al., 1994).

The onset data are then integrated over a typical acoustic event time window, $h_2[n]$, which has a 6dB cutoff at 125Hz. This integrator is defined as follows:

$$h_2[n] = \frac{n}{\alpha_3^2}\exp(-n/\alpha_1). \tag{4.11}$$

For a sample rate of 11025Hz, the parameters are chosen to be $\alpha_1 = 0.06, \alpha_2 = 0.10, \alpha_3 = 0.001$. An adaptive threshold and refraction operation is then applied. The threshold value is selected to produce some percentage (say 0.1~0.5%) of active events in the discretized time-frequency grid when the refractory period is set as 1 ms. The greater the threshold value, the sparser are the spikes in the onset map; on the other hand, increasing the refractory period would thin out the continuous blocks in the onset map. Typical threshold value is within the region $[1, 1.7]$, and typical refractory period value is chosen between 100 and 150.

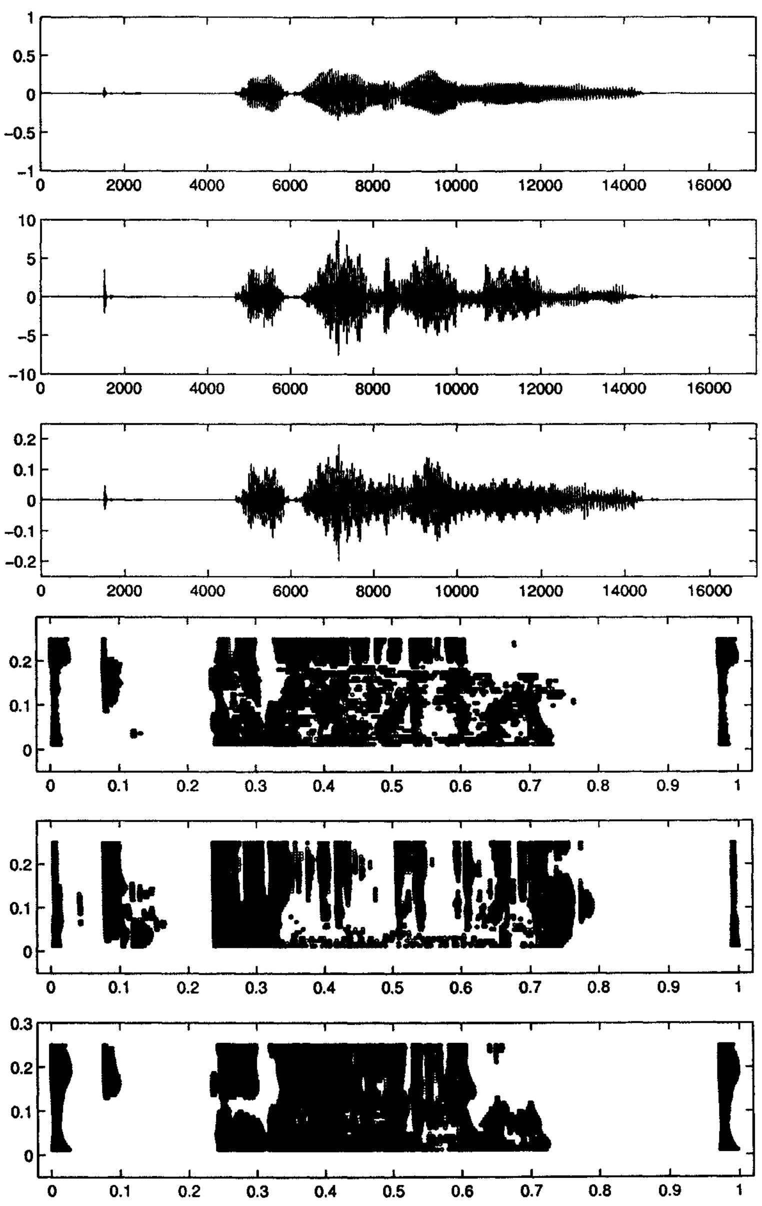

Figure 4.9: Comparisons of *normal*, *deficient*, and *Neurocompensated* speech waveforms (top three panels) and spike-trains onset maps (bottom three panels). The deficient speech waveform is produced by preprocessing the signal through the standard NAL-RP algorithm, with all gains set to $G_i \equiv G_i^{NAL-RP}$ for the 20 time-frequency bands and then reconstructing the signal by inverse FFT; the deficient spike-trains map is generated using the hearing-impaired model applied to the deficient waveform. The KL divergence between the deficient and normal spike trains is 0.664 before the learning, as opposed to 0.42 between the Neurocompensated and normal spike trains after the learning.

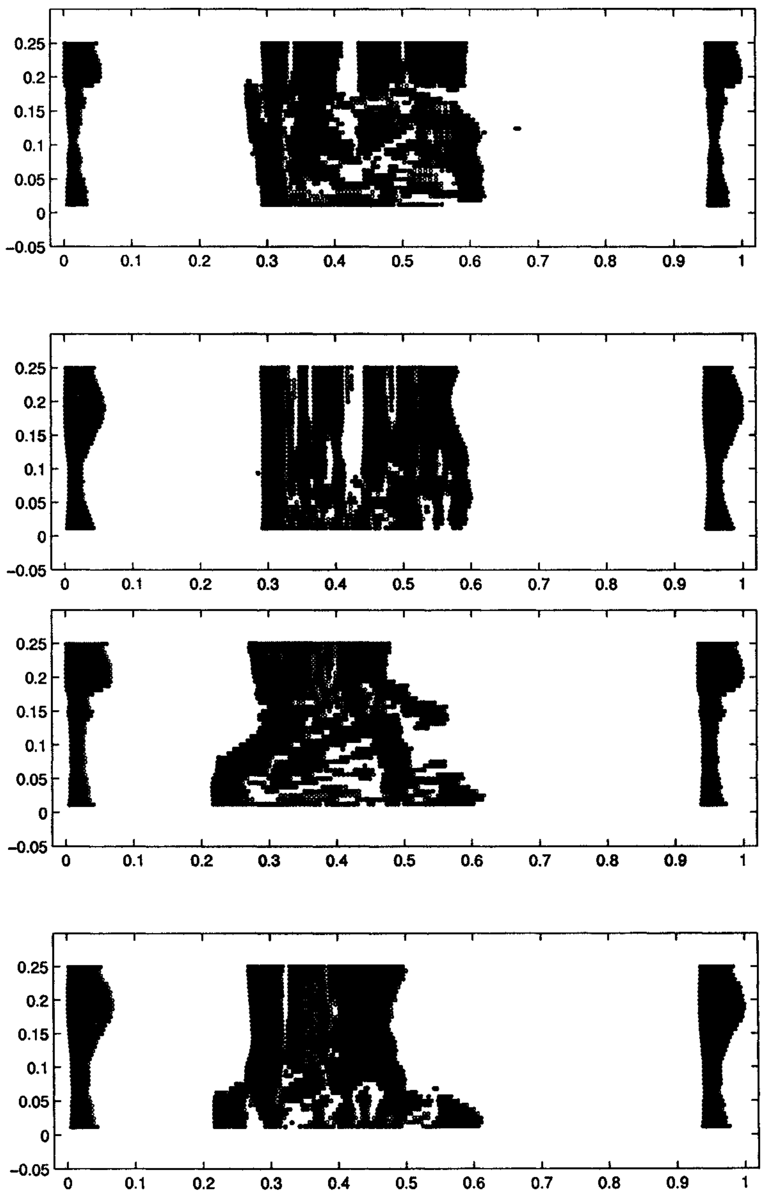

Figure 4.10: Testing results on two untrained continuous speech samples. Comparison is made between the *normal* and *Neurocompensated* spike-trains onset maps. The KL divergence of equation (4.8) is 0.2013 between the first two maps, and 0.5591 between the last two maps.

Chapter 5

Monte Carlo Methods for Bayesian Estimation

"The probability of any event is the ratio between the value at which an expectation depending on the happening of the event ought to be computed, and the value of the thing expected upon its happening."

— Thomas Bayes

5.1 Preliminaries

Definition 5.1 *Let $p(\mathbf{x}) = \frac{dP(\mathbf{x})}{d\mu}$ denote the Radon-Nikodým density of probability distribution $P(\mathbf{x})$ w.r.t. a measure μ. When $\mathbf{x} \in X$ is discrete and μ is a counting measure, $p(\mathbf{x})$ is a probability mass function (pmf); when $\mathbf{x}$ is continuous and μ is a Lebesgue measure, $p(\mathbf{x})$ is a probability density function (pdf).*

Intuitively, the true distribution $P(\mathbf{x})$ can be replaced by an *empirical distribution* given the simulated samples:

$$\hat{P}(\mathbf{x}) = \frac{1}{N_p} \sum_{i=1}^{N_p} \delta(\mathbf{x} - \mathbf{x}^{(i)}),$$

where $\delta(\cdot)$ is a Radon-Nikodým density w.r.t. μ of the point-mass distribution concentrated at the point $\mathbf{x}$. When $\mathbf{x} \in X$ is discrete, $\delta(\mathbf{x}-\mathbf{x}^{(i)})$ is 1 for $\mathbf{x} = \mathbf{x}^{(i)}$ and 0 elsewhere. When $\mathbf{x} \in X$ is continuous, $\delta(\mathbf{x} - \mathbf{x}^{(i)})$ is a Dirac-delta function, $\delta(\mathbf{x} - \mathbf{x}^{(i)}) = 0$ for all $\mathbf{x}^{(i)} \neq \mathbf{x}$, and $\int_X d\hat{P}(\mathbf{x}) = \int_X \hat{p}(\mathbf{x}) d\mathbf{x} = 1$.

Definition 5.2 *The Markov assumption holds when the current state only depends on a finite history of the previous state. A stochastic process satisfying the Markov assumption is called a Markov process (for continuous time) or a Markov chain (for discrete time).*

Definition 5.3 *Assuming a state vector* $\mathbf{x} \in \mathbb{R}^N$ *in a probability space* $(\Omega, \mathcal{F}, P)$, *and* $K(\cdot, \cdot)$ *is a transition kernel, then a first-order Markov chain is a sequence of random variable* $\{\mathbf{x}_t\}_{t \geq 0}$ *such that*

$$P(\mathbf{x}_t | \mathbf{x}_0, \cdots, \mathbf{x}_{t-1}) = P(\mathbf{x}_t | \mathbf{x}_{t-1}),$$

and $K(\mathbf{x}_{t-1}, \mathbf{x}_t) \propto p(\mathbf{x}_t | \mathbf{x}_{t-1})$. *(In discrete finite state case,* $K(\mathbf{x}_{t-1}, \mathbf{x}_t)$ *reduces to a matrix.) Hence, for Markov chains, the probability chain reads as*

$$P(\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_t) = P(\mathbf{x}_0) \prod_{i=1}^{t} P(\mathbf{x}_i | \mathbf{x}_{i-1}).$$

5.2 Bayesian Statistics and Bayesian Estimation

Bayesian theory (Bernardo and Smith, 1998; Robert, 2001) is a branch of mathematical probability theory that allows people to model the uncertainty about the world and the outcomes of interest by incorporating prior knowledge and observational evidence. Bayesian analysis, interpreting the probability as a *conditional measure of uncertainty*, is one of the popular methods to solve the inverse problems. The *Sufficiency Principle* and *Likelihood Principle* constitute two axiomatic principles in Bayesian inference (Robert, 2001).

There are three types of intractable problems inherently related to the Bayesian statistics:

1. **Normalization:** Given the prior $p(\mathbf{x})$ and likelihood $p(\mathbf{y}|\mathbf{x})$, the posterior $p(\mathbf{x}|\mathbf{y})$ is obtained by the product of prior and likelihood divided by a normalizing factor:

$$p(\mathbf{x}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{x})p(\mathbf{x})}{\int_X p(\mathbf{y}|\mathbf{x})p(\mathbf{x})d\mathbf{x}}. \tag{5.1}$$

2. **Marginalization:** Given the joint posterior $(\mathbf{x}, \mathbf{z})$, the marginal posterior is estimated by:

$$p(\mathbf{x}|\mathbf{y}) = \int_Z p(\mathbf{x}, \mathbf{z}|\mathbf{y}) d\mathbf{z}, \tag{5.2}$$

   as shown later, *marginalization* and *factorization* play an important role in Bayesian inference.

3. **Expectation:** Given the conditional pdf, some averaged statistics of interest can be calculated as:

$$\mathbb{E}_{p(\mathbf{x}|\mathbf{y})}[f(\mathbf{x})] = \int_X f(\mathbf{x}) p(\mathbf{x}|\mathbf{y}) d\mathbf{x}. \tag{5.3}$$

In Bayesian inference, all uncertainties (including states, model, and priors) are treated as random variables. The inference is performed within the Bayesian framework given all of the available information. And the objective of Bayesian inference is to use priors and causal knowledge, quantitatively and qualitatively, to infer the conditional probability, given the finite observations. *Optimization* and *integration* are the two fundamental numerical problems arising in statistical inference (Robert, 2001); as we see later, these two problems can be naturally tackled by Monte Carlo simulation. In a sequential estimation (filtering/smoothing) framework, we are particularly interested in applying the Bayes' rule (Bayes, 1763) for recursive Bayesian estimation.

5.3 Monte Carlo Sampling and Particle Filtering

In this section, we will briefly review some key concepts of Monte Carlo sampling and sequential Monte Carlo estimation (a.k.a. particle filtering) methods.

5.3.1 Generic State-Space Model

Let us consider a discrete-time generic state-space model:

$$\begin{aligned} \mathbf{x}_{t+1} &= \mathbf{f}(t, \mathbf{x}_t, \mathbf{u}_t, \mathbf{d}_t), && (5.4a) \\ \mathbf{y}_t &= \mathbf{g}(t, \mathbf{x}_t, \mathbf{u}_t, \mathbf{v}_t), && (5.4b) \end{aligned}$$

which respectively describe the time-varying state and measurement equations. $\mathbf{x}_t$ represents the state of interest; $\mathbf{y}_t$ is the measurement vector; $\mathbf{u}_t$ is the known input vector that appears in either state or measurement, or both equations; $\mathbf{f}$ and $\mathbf{g}$ are two generic vector-valued functions, which are potentially time-varying; $\mathbf{d}_t$ and $\mathbf{v}_t$ represent the dynamic and measurement noise processes, respectively, with appropriate dimensions. The state equation (5.4a) characterizes the state transition probability $p(\mathbf{x}_{t+1}|\mathbf{x}_t)$, whereas the measurement equation (5.4b) describes the likelihood $p(\mathbf{y}_t|\mathbf{x}_t)$. For simplicity, we assume that the noise covariances of dynamical and measurement noises, Σ_d and Σ_v, are known. The dynamic state-space model (5.4a) and (5.4b) can be illustrated via a graphical model in Figure 5.1.

5.3.2 Bayesian Filtering

Bayesian estimation theory provides the most elegant framework for solving (5.4a) and (5.4b) for the state vector $\mathbf{x}_t$, given the set of observations $\mathbf{y}_{0:t} \equiv \{\mathbf{y}_0, \mathbf{y}_1, \cdots, \mathbf{y}_t\}$. More specifically, given the observations and the state-space model, the requirement is to sequentially estimate the conditional filtering posterior $p(\mathbf{x}_t|\mathbf{y}_{0:t})$ evolving through time, which is intrinsically governed by the Kushner equation (Kushner, 1965) or Zakai equation in the stochastic differential equation (SDE) theoretic framework (see Chen, 2003a,

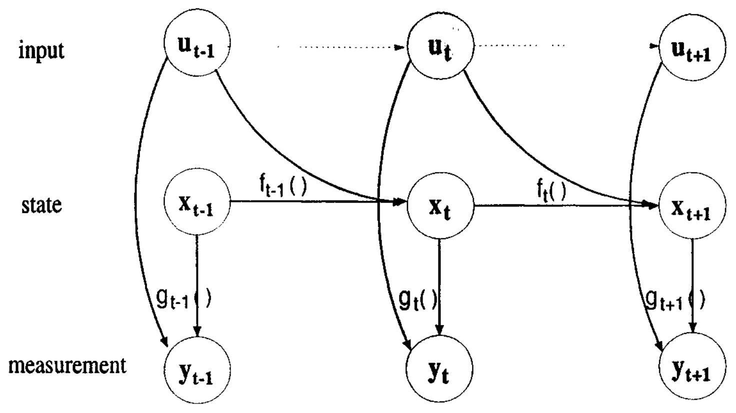

Figure 5.1: A graphical model illustration of the generic state-space model.

for a review). The celebrated Kalman filter (Kalman, 1960) is indeed an instance of Bayesian filtering under the linear, Gaussian assumption. Since the Gaussian density is only characterized by the first- and second-order statistics, Kalman filter turns out to be sufficient for the linear Gaussian model. However, in general cases of a non-Gaussian environment, estimating the posterior becomes intractable. Although various approximation techniques, such as the Gaussian sum filter (Anderson and Moore, 1979), grid-based discretization, piecewise approximation (Kramer and Sorenson, 1988), and point-mass approximation (Bucy and Senne, 1971), have been advocated in the literature, these approximate methods are either unreliable (inaccurate) or computationally prohibitive.

To circumvent these difficulties, the idea of Monte Carlo simulation was introduced to the filtering community in late 1960s and 1970s (Handschin and Mayne, 1969; Handschin, 1970; Zaritskii et al., 1975; West, 1992; Tanizaki, 1996). Partially due to the limited computational power available at that time, sequential Monte Carlo estimation did not attract much attention from researchers until very recently. Various sequential Bayesian filtering approaches have been developed in different areas under different aliases, such as the bootstrap filter (Gordon et al., 1993), sequential imputation (Liu and Chen, 1995), CONDESNSATION (Isard and Blake, 1998), or Monte Carlo filter (Kitagawa, 1996; Tanizaki, 2000). We will treat them in this paper under the generalized particle filtering framework. The reader is referred to (Doucet et al., 2001; Arulampalam et al., 2002; Chen, 2003a) for a comprehensive overview of the state-of-the-art research in this area.

Sequential-Importance-Sampling

In many situations, it is often hard, if not impossible, to directly draw samples from the target density $p(\mathbf{x})$, which is often non-Gaussian. In order to avoid the difficulty we can use importance sampling to draw the samples. To do that, we introduce a known function $q(\mathbf{x})$ known as a *proposal distribution* (or importance density), which is close in shape to the target $p(\mathbf{x})$. Suppose we want to evaluate the mean statistic of a generic function

$f(\mathrm{x})$; it turns out that

$$\begin{aligned}\mathbb{E}[f(\mathrm{x})] &= \int f(\mathrm{x})\frac{p(\mathrm{x})}{q(\mathrm{x})}q(\mathrm{x})d\mathrm{x} \\ &= \int f(\mathrm{x})W(\mathrm{x})q(\mathrm{x})d\mathrm{x},\end{aligned} \tag{5.5}$$

where the $W(\mathrm{x}) = p(\mathrm{x})/q(\mathrm{x})$ are called the *importance weights.* If the target density is a conditional posterior $p(\mathrm{x}_{0:t}|\mathrm{y}_{0:t})$ given the observations from 0 up to t,[1] (5.5) can be represented as

$$\begin{aligned}\mathbb{E}[f(\mathrm{x}_t)] &= \int f(\mathrm{x}_t)\frac{p(\mathrm{x}_{0:t}|\mathrm{y}_{0:t})}{q(\mathrm{x}_t|\mathrm{x}_{t-1},\mathrm{y}_t)}q(\mathrm{x}_t|\mathrm{x}_{t-1},\mathrm{y}_t)d\mathrm{x}_t \\ &= \frac{1}{p(\mathrm{y}_{0:t})}\int f(\mathrm{x}_t)W_t(\mathrm{x}_t)q(\mathrm{x}_t|\mathrm{x}_{t-1},\mathrm{y}_t)d\mathrm{x}_t\end{aligned} \tag{5.6}$$

where

$$W_t(\mathrm{x}_t) = \frac{p(\mathrm{y}_t|\mathrm{x}_t)p(\mathrm{x}_t)}{q(\mathrm{x}_t|\mathrm{x}_{t-1},\mathrm{y}_t)}. \tag{5.7}$$

Thus (5.6) can be rewritten as

$$\begin{aligned}\mathbb{E}[f(\mathrm{x}_t)] &= \frac{\int f(\mathrm{x}_t)W_t(\mathrm{x}_t)q(\mathrm{x}_t|\mathrm{x}_{t-1},\mathrm{y}_t)d\mathrm{x}_t}{\int p(\mathrm{y}_{0:t}|\mathrm{x}_t)p(\mathrm{x}_t)d\mathrm{x}_t} \\ &= \frac{\int f(\mathrm{x}_t)W_t(\mathrm{x}_t)q(\mathrm{x}_t|\mathrm{x}_{t-1},\mathrm{y}_t)d\mathrm{x}_t}{\int W_t(\mathrm{x}_t)q(\mathrm{x}_t|\mathrm{x}_{t-1},\mathrm{y}_t)d\mathrm{x}_t} \\ &= \frac{\mathbb{E}_{q(\mathrm{x}_t|\mathrm{x}_{t-1},\mathrm{y}_t)}[W_t(\mathrm{x}_t)f(\mathrm{x}_t)]}{\mathbb{E}_{q(\mathrm{x}_t|\mathrm{x}_{t-1},\mathrm{y}_t)}[W_t(\mathrm{x}_t)]},\end{aligned} \tag{5.8}$$

where the expectation in both the numerator and denominator are performed with respect to the proposal distribution $q(\mathrm{x})$. By drawing N_p i.i.d. samples $\{\mathrm{x}_t^{(i)}\}$ from $q(\mathrm{x}_t|\mathrm{x}_{t-1},\mathrm{y}_t)$, we can approximate (5.8) as

$$\begin{aligned}\mathbb{E}[f(\mathrm{x}_t)] &\approx \frac{\frac{1}{N_p}\sum_{i=1}^{N_p} W_t^{(i)} f(\mathrm{x}_t^{(i)})}{\frac{1}{N_p}\sum_{i=1}^{N_p} W_t^{(i)}} \\ &= \sum_{i=1}^{N_p} \tilde{W}_t^{(i)} f(\mathrm{x}_t^{(i)}) \equiv \hat{f}(\mathrm{x}),\end{aligned} \tag{5.9}$$

where the *normalized importance weights* are given as

$$\tilde{W}_t^{(i)} = \frac{W_t^{(i)}}{\sum_{j=1}^{N_p} W_t^{(j)}}. \tag{5.10}$$

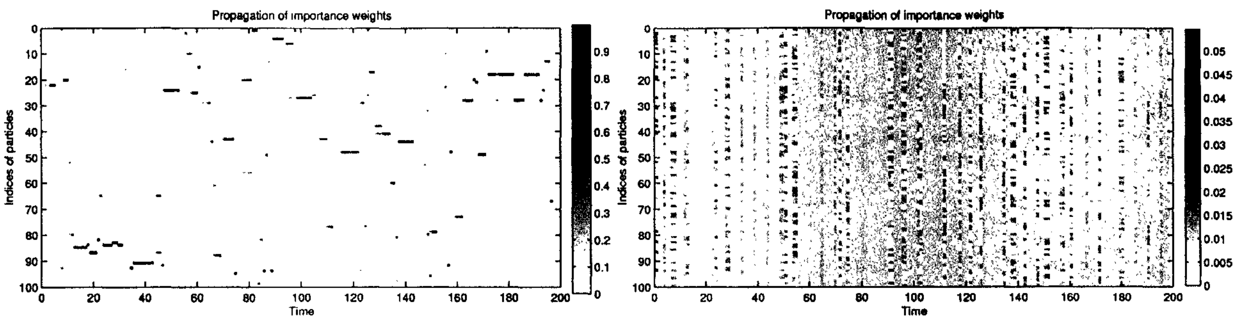

Figure 5.2: An illustration of weight degeneracy problem without (*left panel*) and with (*right panel*) resampling in a one-dimensional nonlinear state estimation problem. The resampling threshold in this example is set as $N_T = N_p/2 = 50$.

In a sequential filtering/estimation framework, by choosing a factorized proposal distribution, it can be shown that the importance weights are updated recursively as follows (Doucet et al., 2000):

$$
\begin{aligned}
W_t^{(i)} &= \frac{p(\mathbf{x}_{0:t}^{(i)}|\mathbf{y}_{0:t})}{q(\mathbf{x}_{0:t}^{(i)}|\mathbf{y}_{0:t})} \\
&\propto \frac{p(\mathbf{y}_t|\mathbf{x}_t^{(i)})p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)})p(\mathbf{x}_{0:t-1}^{(i)}|\mathbf{y}_{0:t-1})}{q(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)},\mathbf{y}_{0:t})q(\mathbf{x}_{0:t-1}^{(i)}|\mathbf{y}_{0:t-1})} \\
&= W_{t-1}^{(i)}\frac{p(\mathbf{y}_t|\mathbf{x}_t^{(i)})p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)})}{q(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)},\mathbf{y}_{0:t})} \\
&= W_{t-1}^{(i)}\frac{p(\mathbf{y}_t|\mathbf{x}_t^{(i)})p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)})}{q(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)},\mathbf{y}_t)}. \qquad (5.11)
\end{aligned}
$$

Equation (5.11) underlies the basic principle of *sequential importance sampling* (SIS) filter. In practice, however, it has been shown (Kong et al., 1994; Liu and Chen, 1995) that the distribution of the importance weights becomes more and more skewed as time increases; hence, after some iterations, only very few particles have non-zero importance weights (see Figure 5.2 for an illustration). This phenomenon is often referred to as *weight degeneracy* or *sample impoverishment*. An intuitive solution is to multiply the particles with high normalized importance weights, and discard the particles with low normalized importance weights, which can be done in the resampling (selection) step. To monitor the degeneracy, a suggested measure called *effective sample size*, N_{eff}, was introduced in (Kong et al.,

[1]We use notations $\mathbf{x}_{0:t}$ and $\mathbf{y}_{0:t}$ to denote the collections of variables $\mathbf{x}$ and $\mathbf{y}$ from time 0 to t, respectively.

1994; Doucet et al., 2000)

$$\begin{aligned} N_{eff} &= \frac{N_p}{1 + \mathrm{Var}_{q(\cdot|\mathbf{y}_{0:t})}[\tilde{W}(\mathbf{x}_{0:t})]} \\ &= \frac{N_p}{\mathbb{E}_{q(\cdot|\mathbf{y}_{0:t})}[(\tilde{W}(\mathbf{x}_{0:t}))^2]} \leq N_p. \end{aligned} \tag{5.12}$$

The second line of (5.12) follows from the fact that $\mathrm{Var}[\xi] = \mathbb{E}[\xi^2] - (\mathbb{E}[\xi])^2$ and $\mathbb{E}_q[\tilde{W}] = 1$. In practice, the true N_{eff} is not available, thus its estimate, $\hat{N}_{eff}$, is suggested for use (Kong et al., 1994):

$$\hat{N}_{eff} = \frac{1}{\sum_{i=1}^{N_p} (\tilde{W}_t^{(i)})^2}. \tag{5.13}$$

When $\hat{N}_{eff}$ is below a predefined threshold N_T, the resampling procedure is performed. In a sequential filtering framework, the resampling step is almost inevitable, however, it also introduces some random variations. It should be pointed out that resampling does *not* really prevent the weight degeneracy problem; rather, it just improves the sample efficiency by discarding the particles associated with insignificant weights. Since it replaces the particles with high importance weights with many replicates, it also introduces certain correlations within them, especially when there are only a few dominant weights. This problem is sometimes called the *loss of diversity*. To improve this, Markov chain Monte Carlo (MCMC) techniques (see Appendix B for details) usually come into the rescue (Gilks et al., 1996; Gilks and Berzuini, 2001). In the literature, there exist a variety of resampling schemes (Gordon et al., 1993; Liu and Chen, 1998; Carpenter et al., 1999; Kitagawa, 1996)) and resampling schedules; we refer the reader to (Doucet et al., 2001; Chen, 2003a) for more information. Table 5.1 summarizes a generic form of SIS particle filtering with multinomial resampling. A schematic diagram of generic particle filtering is illustrated in Figure 5.3.

Another important issue in particle filtering is the choice of a proposal distribution, which plays an important role in determining the estimation performance. Usually, choosing a proposal distribution is problem-dependent and demands a good understanding of the problem at hand. Some potential criteria for a good proposal distribution include:

- The support of the proposal distribution should cover that of the posterior distribution; in other words, the proposal should have a broad distribution.
- The proposal distribution has a long-tailed behavior to account for outliers, which ensures the unnormalized importance weights are upper bounded.
- The sampling procedure is easy to implement.
- Account is taken of transition prior and likelihood, as well as the recent observation data.

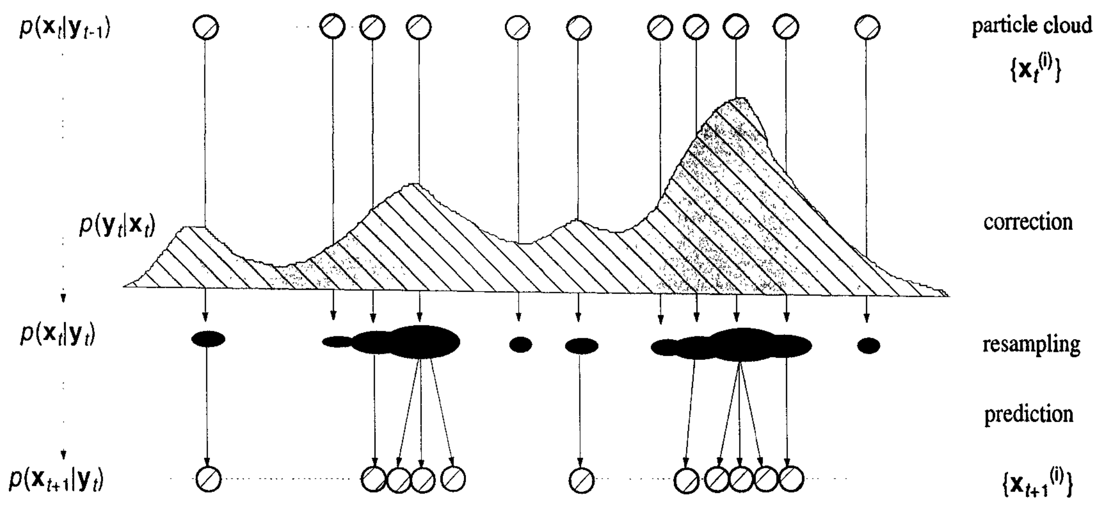

Figure 5.3: An illustration of generic particle filter with importance sampling and resampling.

- A minimum variance is achieved.
- The proposal distribution is close (in shape) to the true posterior.

However, satisfying all of these goals is not easy and we do not know what the posterior is supposed to look like. It has been shown (Doucet et al., 2000) that the choice of proposal $q(\mathbf{x}_t|\mathbf{x}_{0:t-1}^{(i)}, \mathbf{y}_{0:t}) = p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}, \mathbf{y}_t)$ minimizes the variance of $W_t^{(i)}$.

Lemma 1 (Doucet et al., 2000) $q(\mathbf{x}_t|\mathbf{x}_{0:t-1}^{(i)}, \mathbf{y}_{0:t}) = p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}, \mathbf{y}_t)$ *is the optimal importance function that minimizes the variance of the importance weights conditional on* $\mathbf{x}_{t-1}^{(i)}$ *and* $\mathbf{y}_{0:t}$*, for which a zero variance* $Var_q[\tilde{W}_t^{(i)}] = 0$ *is achieved (therefore no resampling is required).*

By choosing such an optimal proposal distribution, the importance weights can be recursively calculated as:

$$W_t^{(i)} = W_{t-1}^{(i)} p(\mathbf{y}_t|\mathbf{x}_{t-1}^{(i)}), \tag{5.14}$$

which follows from equation (5.11) and the Bayes rule: $p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}, \mathbf{y}_t) = \frac{p(\mathbf{y}_t|\mathbf{x}_t^{(i)})p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)})}{p(\mathbf{y}_t|\mathbf{x}_{t-1}^{(i)})}$. However, this optimal choice suffers from two drawbacks (Doucet et al., 2000): it requires (i) sampling from probability distribution $p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}, \mathbf{y}_n)$ and (ii) evaluating the integral $p(\mathbf{y}_t|\mathbf{x}_{t-1}^{(i)}) = \int p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})d\mathbf{x}_t$. In general, it is difficult to achieve these goals, except for special cases with analytic (approximate) solutions (e.g., Doucet et al., 2000). In practice, choosing different proposal distributions essentially leads to different kinds of particle filters.

Table 5.1: SIS filter with resampling.

```
For time steps $t = 0, 1, 2, \cdots$
1: For $i = 1, \cdots, N_p$, draw the samples $\mathrm{x}_t^{(i)} \sim q(\mathrm{x}_t|\mathrm{x}_{0:t-1}^{(i)}, \mathrm{y}_{0:t})$ and set $\mathrm{x}_{0:t}^{(i)} = \{\mathrm{x}_{0:t-1}^{(i)}, \mathrm{x}_t^{(i)}\}$.
2: Calculate the importance weights $W_t^{(i)}$ according to (5.11).
3: Normalize the importance weights $\tilde{W}_t^{(i)}$ according to (5.10).
4: Calculate $\hat{N}_{eff}$ according to (5.13), return if $\hat{N}_{eff} > N_T$, otherwise generate a new particle set $\{\mathrm{x}_t^{(j)}\}$ by resampling with replacement $N_p$ times from the previous set $\{\mathrm{x}_{0:t}^{(i)}\}$ with probabilities $\Pr(\mathrm{x}_{0:t}^{(j)} = \mathrm{x}_{0:t}^{(i)}) = \tilde{W}_{0:t}^{(i)}$, reset the weights $\tilde{W}_t^{(i)} = 1/N_p$.
```

Table 5.2: SIR filter using transition prior as proposal distribution.

```
For time steps $t = 0, 1, 2, \cdots$
1: Importance Sampling: for $i = 1, \cdots, N_p$, draw samples $\hat{\mathrm{x}}_t^{(i)} \sim p(\mathrm{x}_t|\mathrm{x}_{t-1}^{(i)})$, set $\hat{\mathrm{x}}_{0:t}^{(i)} = \{\mathrm{x}_{0:t-1}^{(i)}, \hat{\mathrm{x}}_t^{(i)}\}$.
2: Importance Weight Update: $W_t^{(i)} = p(\mathrm{y}_t|\hat{\mathrm{x}}_t^{(i)})$.
3: Normalize the importance weights according to (5.10).
4: Resampling: Generate $N_p$ new particles $\mathrm{x}_t^{(i)}$ from the set $\{\hat{\mathrm{x}}_t^{(i)}\}$ according to the importance weights $\tilde{W}_t^{(i)}$.
```

Sampling-Importance-Resampling (SIR) Filter

The *sampling-importance-resampling* (SIR; a.k.a. Bayesian bootstrap) filter (Gordon et al., 1993) is very similar to the SIS filter. A difference between them is that in SIR the resampling step is always performed at every step, where in SIS it is not. In practice, the proposal distribution of the SIR filter is usually chosen as transition prior density, which is easy to implement, in which case it turns out that the current importance weights only depend on the likelihood. Table 5.2 summarizes such an algorithm. Many improvement schemes, such as stratified sampling (Carpenter et al., 1999) and auxiliary variable (Pitt and Shephard, 1999), have been developed for the SIR filter; see (Doucet et al., 2001; Chen, 2003a) for more details.

5.3.3 Metropolis-Hastings Algorithm

The Metropolis-Hastings algorithm (Metropolis et al., 1953; Hastings, 1970), being the first Monte Carlo algorithm, is a general and powerful MCMC technique (see Appendix B for background and details). In what follows, we briefly describe its idea and implementation.

Let $\pi(\mathrm{x})$ denote the target (equilibrium) probability distribution, and suppose that

$q(\mathbf{x}, \mathbf{x}')$ is the proposal distribution that does not satisfy the *detailed balance* condition:

$$\pi(\mathbf{x})q(\mathbf{x}, \mathbf{x}') = \pi(\mathbf{x}')q(\mathbf{x}', \mathbf{x}) \tag{5.15}$$

Without loss of generality, suppose $\pi(\mathbf{x})q(\mathbf{x}, \mathbf{x}') > \pi(\mathbf{x}')q(\mathbf{x}', \mathbf{x})$, which means the probability moving from $\mathbf{x}$ to $\mathbf{x}'$ is greater (more frequent) than the probability moving from $\mathbf{x}'$ to $\mathbf{x}$. Intuitively, we want to change this situation to reduce the number of moves from $\mathbf{x}$ to $\mathbf{x}'$. In doing so, we introduce a *probability of move*, $0 < \alpha(\mathbf{x}, \mathbf{x}') < 1$; if the move is not performed, the process returns $\mathbf{x}$ as a value from the target distribution.

In doing so, the transition probability from $\mathbf{x}$ to $\mathbf{x}'$ becomes:

$$\begin{aligned} p_{\mathrm{MH}}(\mathbf{x}, \mathbf{x}') &= q(\mathbf{x}, \mathbf{x}')\alpha(\mathbf{x}, \mathbf{x}'), & (5.16)\\ \alpha(\mathbf{x}, \mathbf{x}') &= \begin{cases} \min\left[\frac{\pi(\mathbf{x}')q(\mathbf{x}', \mathbf{x})}{\pi(\mathbf{x})q(\mathbf{x}, \mathbf{x}')}, 1\right], & \text{if } \pi(\mathbf{x})q(\mathbf{x}, \mathbf{x}') > 0,\\ 1 & \text{otherwise} \end{cases} & (5.17)\end{aligned}$$

Thus, the probability that the Markov process stays at $\mathbf{x}$ can be written as:

$$1 - \int_X q(\mathbf{x}, \mathbf{x}')\alpha(\mathbf{x}, \mathbf{x}')d\mathbf{x}', \tag{5.18}$$

and the associated transition kernel is given by

$$K_{\mathrm{MH}}(\mathbf{x}, d\mathbf{x}') = q(\mathbf{x}, \mathbf{x}')\alpha(\mathbf{x}, \mathbf{x}')d\mathbf{x}' + \left[1 - \int_X q(\mathbf{x}, \mathbf{x}')\alpha(\mathbf{x}, \mathbf{x}')d\mathbf{x}'\right]\delta_{\mathbf{x}}(d\mathbf{x}'). \tag{5.19}$$

To summarize, the Metropolis-Hastings sampling algorithm reads as follows:

1. For $i = 1, \cdots, N_p$, at iteration $t = 0$, draw a starting point $\mathbf{x}_0$;
2. generate a uniform random variable $u \sim \mathcal{U}(0, 1)$, and $\mathbf{x}' \sim q(\mathbf{x}_t, \cdot)$;
3. If $u < \alpha(\mathbf{x}_t, \mathbf{x}')$, set $\mathbf{x}_{t+1} = \mathbf{x}'$, else $\mathbf{x}_{t+1} = \mathbf{x}_t$;
4. $t \leftarrow t+1$, repeat steps 2 and 3, until certain (say k) steps (i.e. burn-in time), store $\mathbf{x}^{(i)} = \mathbf{x}_k$.
5. $i \leftarrow i+1$, repeat the procedure until N_p samples are drawn, return the sample set $\{\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(N_p)}\}$.

5.4 Improved Schemes for Particle Filtering

5.4.1 A Brief Overview

In the past few years, many efforts have been devoted to improving the particle filters' performance (see e.g., Doucet et al., 2001; Chen, 2003a, for more information). We briefly describe several improved schemes and illustrate the underlying weakness behind the conventional SIR filter (a.k.a. Bayesian bootstrap filter).

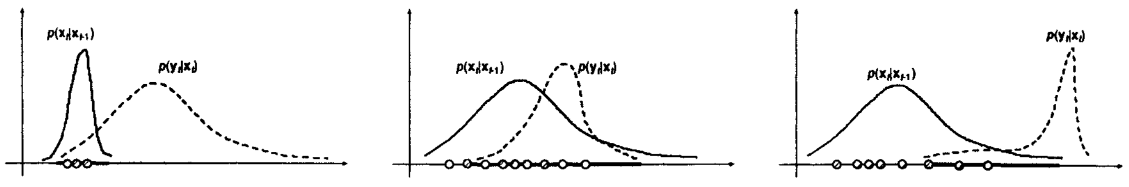

Figure 5.4: *Left:* $\Sigma_{\mathrm{d}} < \Sigma_{\mathrm{v}}$, transition prior $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ is peaked compared to the flat likelihood $p(\mathbf{y}_t|\mathbf{x}_t)$, and their overlapping region is indicated by the thick line; *Middle:* $\Sigma_{\mathrm{d}} \approx \Sigma_{\mathrm{v}}$, the supports of the prior and likelihood largely overlap, where the prior proposal works well; *Right:* an illustration of a poor approximation of the transition prior as proposal distribution when the likelihood is peaked when $\Sigma_{\mathrm{d}} > \Sigma_{\mathrm{v}}$. Sampling from the prior does not generate sufficient particles in the overlapping region.

One of the reasons of using a transition prior $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ as proposal $q(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{y}_t)$ (as in the Bayesian bootstrap filter) is its simplicity. By doing so, in light of (5.11), the importance weights are updated as:

$$W_t^{(i)} = W_{t-1}^{(i)} p(\mathbf{y}_t|\mathbf{x}_t^{(i)}), \tag{5.20}$$

This simplification sidesteps sampling from the proposal and makes the weights proportional to the likelihood $p(\mathbf{y}_t|\mathbf{x}_t)$. However, the transition prior essentially neglects the information of the observation $\mathbf{y}_t$; in addition, it does not take into account of the noise statistics (i.e., Σ_{d} and Σ_{v}) appearing in the state-space model. Without too much difficulty, one can imagine that if the samples drawn from the transition prior do not cover the likelihood region, the performance of the particle filter will be very poor since the contributions of most particles are insignificant (see Figure 5.4 for a one-dimensional example illustration). For this reason, many improved versions of particle filtering have been developed.

Auxiliary Particle Filtering. The idea of the auxiliary particle filter (APF, Pitt and Shephard, 1999) is to introduce an auxiliary variable, ξ, with a role as the index indicating which mixture the particles belong to. The APF differs from the SIR filter in that it reverses the order of sampling and resampling, which is possible when the importance weights only depend on current states. APF consists of a two-step procedure: First, simulate the particles with a *predictive* likelihood; second, reweigh the particles and draw the augmented states. The APF can also be interpreted as a one-step ahead filtering: the particles $\mathbf{x}_{t-1}^{(i)}$ is propagated to $\xi_t^{(i)}$ in the next time step in order to assist the sampling. The performance of the APF is usually better than that of the SIR since it uses more information including the recent observation $\mathbf{y}_t$; but when the likelihood is not insensitive to the state, the difference between the APF and SIR will be insignificant.

Gaussian Proposal Particle Filtering. As the name indicates, the proposal distribution is approximated as a Gaussian distribution. The sufficient statistics (mean and covariance) of the Gaussian proposal is estimated recursively via the extended Kalman filter (Doucet et al., 2000; Wan and van der Merwe, 2001). In general, these kinds of particle filters produce better performance, but they are often more computationally costly. When the proposal distribution is non-Gaussian and approximated via the unscented transformation (Julier et al., 2000; Julier and Uhlmann, 2004) or unscented Kalman filter (UKF, Wan and van der Merwe, 2001), it yields the so-called unscented particle filter (UPF, van der Merwe et al., 2000). In some special cases where the linear Gaussian or Gaussian mixture are assumed, mixture Kalman filter (MKF, Doucet et al., 2000; Chen and Liu, 2000), Gaussian and Gaussian sum particle filters (Kotecha and Djurić, 2003a,b), can be developed along this line.

MCMC Particle Filtering. When the state space is very large (say $N_{\mathrm{x}} > 10$), the performance of particle filters depends heavily on the choice of the proposal distribution. In order to tackle more general and more complex probability distributions, MCMC methods are often required. In the particle filtering framework, MCMC is used for drawing the samples from an invariant distribution, either in sampling or resampling step. Many researchers have tried to integrate the MCMC techniques to particle filtering, (e.g., Liu and Chen, 1998; MacEachern et al., 1999; Pitt and Shephard, 1999; Fearnhead, 2002). One special kind of MCMC particle filter is the resample-move algorithm, which combines the SIR and MCMC sampling. The basic idea is as follows (Gilks and Berzuini, 2001): The particles are grouped into a set $S_t = \{\mathrm{x}_t^{(i)}\}_{i=1}^{N_p}$ at time step t, and they are propagated through the state-space equations by using SIR and MCMC sampling, at time $t+1$, the resampled particles are moved according to a Markov chain transition kernel to form a new set S_{t+1}; in the *rejuvenation* stage, two steps are performed: (i) In the resample-step, draw the samples $\{\mathrm{x}_t^{(i)}\}$ from S_t such that they are selected with a probability proportional to $\{W(\mathrm{x}_t^{(i)})\}$; (ii) In the move-step, the selected particles are moved to a new location by sampling from a Markov chain transition kernel.

Bearing the goal to circumvent the weakness of the conventional SIR filter while keeping the its simplicity, in the next two subsections, we propose two improved schemes using *gradient proposal* and *Turbo processing principle.*

5.4.2 Gradient Proposal Particle Filtering

As discussed earlier, the conventional SIR particle filter does not take into account the recent observation into the proposal. To overcome this, we propose to use the gradient information of the observation to select the "informative" particles. The main idea behind the proposed new filter is to introduce a gradient MOVE-step followed by the sampling for the proposal distribution, which is plugged in after the sampling step in the conventional SIR filter; it is also more efficient than the ad hoc *prior boosting* method (Gordon et al., 1993). This new algorithm essentially calculates the gradient information from the likelihood model and guides the particles towards the low-error region, along the gradient-descent direction; by assuming an additive measurement noise model in (5.4b) and denoting its first-order gradient w.r.t. the state vector as $\xi(\mathbf{y})$, the MOVE-step is described by

$$\begin{aligned}\mathbf{x}_t &= \tilde{\mathbf{x}}_t - \eta\xi(\mathbf{y}) \\ &= \tilde{\mathbf{x}}_t - \eta\frac{\partial|\mathbf{y}-\mathbf{g}(\mathbf{x})|^2}{\partial\mathbf{x}}\bigg|_{\mathbf{y}=\mathbf{y}_t,\mathbf{x}=\tilde{\mathbf{x}}_t}\end{aligned} \tag{5.21}$$

where $0 < \eta < 1$ is a small-valued step-size parameter, $\tilde{\mathbf{x}}_t$ denotes the predicted estimate from the state equation (i.e., the *a priori* sample drawn from $p(\cdot|\mathbf{x}_{t-1})$), and $\mathbf{x}_t$ denotes the sample after the MOVE-step (which is regarded as the *a posteriori* sample). As expected from (5.21), the inclusion of current observation $\mathbf{y}_t$ and the calculation of gradient information will tend to push the samples to a high-likelihood region, thereby providing more reliable predictive samples for the next step. In this case, the proposal distribution can be derived as

$$\begin{aligned}q(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{y}_t) &= \int p(\mathbf{x}_t|\tilde{\mathbf{x}}_t,\mathbf{y}_t)p(\tilde{\mathbf{x}}_t|\mathbf{x}_{t-1})d\tilde{\mathbf{x}}_t \\ &= \int \delta(\mathbf{x}_t - \tilde{\mathbf{x}}_t + \eta\xi(\mathbf{y}_t))p(\tilde{\mathbf{x}}_t|\mathbf{x}_{t-1})d\tilde{\mathbf{x}}_t \\ &= p(\mathbf{x}_t + \eta\xi(\mathbf{y}_t)|\mathbf{x}_{t-1}) \\ &= p(\tilde{\mathbf{x}}_t|\mathbf{x}_{t-1}).\end{aligned}$$

In summary, the new algorithm on particle filtering with gradient proposal reads as follows:

1. For $i = 1, \cdots, N_p$, sample $\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0)$, $W_0^{(i)} = 1/N_p$.
2. Importance sampling: $\tilde{\mathbf{x}}_t^{(i)} \sim p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})$.
3. For each sample $\{\tilde{\mathbf{x}}_t^{(i)}\}$, run the MOVE-step via (5.21) using the observation $\mathbf{y}_t$ to obtain the sample $\{\mathbf{x}_t^{(i)}\}$.
4. Importance weights update:

$$W_t^{(i)} = W_{t-1}^{(i)}p(\mathbf{y}_t|\mathbf{x}_t^{(i)})\frac{p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})}{p(\tilde{\mathbf{x}}_t|\mathbf{x}_{t-1}^{(i)})},$$

   and normalize the importance weights via (5.10).
5. Calculate $\hat{N}_{eff}$ from (5.13), if $\hat{N}_{eff} > N_T$, go to Step 7; otherwise go to Step 6.
6. Resampling as in the SIR filter.
7. Repeat steps 2 through 5.

The idea of gradient proposal is similar to the HySIR algorithm (deFreitas et al., 2000) that was developed for training neural networks. The difference between ours and theirs is the complexity. In the HySIR algorithm, every particle is propagated and updated through the extended Kalman filter (EKF) equations, thus the complexity is $\mathcal{O}(N_p N_{\mathbf{x}}^2)$. Our algorithm's additional computational overhead is only $\mathcal{O}(N_p N_{\mathbf{x}})$.

It is noteworthy that the above-proposed algorithm is tantamount to choosing a two-step proposal distribution, namely:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{y}_t) = p(\mathbf{x}_t|\mathbf{z}_t), \quad \mathbf{z}_t \sim p(\mathbf{z}_t|\mathbf{x}_{t-1}, \mathbf{y}_t),$$

where $\mathbf{z}_t$ is an intermediate latent variable that we have used in (5.21) as $\mathbf{z}_t \equiv \tilde{\mathbf{x}}_t$. Such a proposal distribution is also similar (but not identical) to the one obtained by approximate linearization (Doucet et al., 2000). However, unlike the scheme therein to sample $\mathbf{x}_t$ from the localized proposal, we choose the proposal as a gradient-included transition prior $p(\mathbf{x}_t|\mathbf{z}_t)$, where the latent variables $\mathbf{z}_t$ are obtained via pushing the samples to the high-likelihood region. Compared to the linearized proposal therein, our scheme is intuitively simpler and problem-independent; namely, it can be used for any state-space model where the gradient of the measurement equation can be calculated. A more sophisticated scheme is to replace the scalar step-size η with a step-size matrix $\eta\Sigma_{\mathbf{v}}^{-1}$. The choice of η is usually problem-specific; a typical value in our experiments is $\eta \in [0.001, 0.005]$. In addition, this technique is applicable to the filtering as well as the smoothing problem. It should be cautioned that the particle filtering procedure using gradient information is not asymptotically consistent, in a sense that the gradient step introduces certain bias to the estimate, and the empirical distribution of particles is not theoretically guaranteed to converge to the right posterior. However, it can be regarded as a "mode" tracker that seeks the peak of the posterior.

In certain cases, the order of Steps 2 and 3 can be reversed; namely, the samples are first driven by the gradient-descent MOVE step and then drawn from the transition prior. However, in such a case we should use the previous measurement $\mathbf{y}_{t-1}$ instead of current observation $\mathbf{y}_t$ (see Section 5.7 for detailed discussion). In so doing, the MOVE-step and the importance weight update will become

$$\begin{aligned}
\tilde{\mathbf{x}}_{t-1} &= \mathbf{x}_{t-1} - \eta \frac{\partial |\mathbf{y} - \mathbf{g}(\mathbf{x})|^2}{\partial \mathbf{x}} \bigg|_{\mathbf{y}=\mathbf{y}_{t-1}, \mathbf{x}=\mathbf{x}_{t-1}} \\
W_t^{(i)} &= W_{t-1}^{(i)} p(\mathbf{y}_t|\mathbf{x}_t^{(i)}) \frac{p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})}{p(\mathbf{x}_t|\tilde{\mathbf{x}}_{t-1}^{(i)})},
\end{aligned}$$

where $\mathbf{x}_t^{(i)} \sim p(\mathbf{x}_t|\tilde{\mathbf{x}}_{t-1})$ represents the simulated sample from the intermediate estimate $\tilde{\mathbf{x}}_{t-1}^{(i)}$ of the MOVE-step. Nevertheless, by doing so we have ignored the information from the most recent observation $\mathbf{y}_t$.

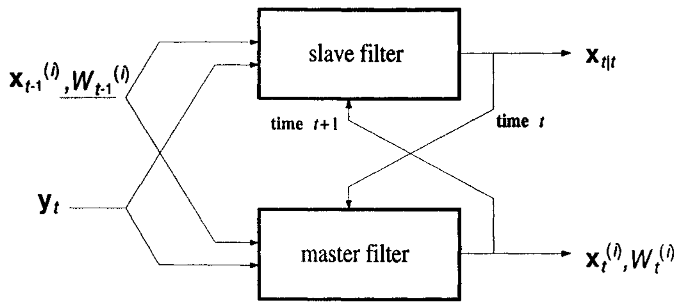

Figure 5.5: A schematic diagram of Turbo-particle filtering.

5.4.3 Turbo-Particle Filtering

Recalling Lemma 5.3.2, constructing an *optimal* (only in the sense of Lemma 5.3.2) proposal distribution $q(\mathrm{x}_t|\mathrm{x}_{0:t-1}^{(i)}, \mathrm{y}_t) = p(\mathrm{x}_t|\mathrm{x}_{t-1}^{(i)}, \mathrm{y}_t)$ suffers from two difficulties, which generally are analytically intractable. In the sequel, we propose a new solution to sidestep these difficulties. The idea was motivated from Turbo decoding in communications (Berrou and Glavieux, 1996), where two decoders are run in parallel to achieve *maximum a posterior* (in Bayesian sense) decoding. Research in the coding community has shown that Turbo decoding performs very well under low signal-to-noise ratio (SNR) and almost approaches the Shannon limit. The secret of the success of Turbo decoding is that it breaks down the solution into two parts, solves them separately, and then combines them; two decoders take advantage of each other's (previous-step) outputs and therefore self-boost the performance iteratively (see e.g., Haykin and Moher, 2004, for details). Bearing the same philosophy in mind, we introduce the Turbo principle (feedback, divide-and-conquer, and iterative processing) for particle filtering. Basically, we use one filter (so-called *slave filter*) to produce a first-stage (rough) estimate of $\hat{\mathrm{x}}_t$; and we run another filter (so-called *master filter*) to yield the second-stage (ultimate) estimate, which uses $\hat{\mathrm{x}}_t$ as well as its own previous estimate $\mathrm{x}_{t-1}^{(i)}$ for constructing a suboptimal proposal in a recursive particle filtering fashion.

A schematic diagram of Turbo-particle filtering is illustrated in Figure 5.5. In Figure 5.5, there are two filters to be run iteratively. The slave filter, being an extended Kalman filter (EKF) for example, is used to produce a rough estimate $\hat{\mathrm{x}}_{t|t}$, given the current observation y_t and previous state estimates $\mathrm{x}_{t-1}^{(i)}$. Note that in the *prediction step*, every particle $\mathrm{x}_{t-1}^{(i)}$ is passed through the state equation, where the predicted covariance can be estimated by the sample covariance, $\hat{\mathbf{P}}_{t|t-1}$, or calculated through linearization; in the *filtering step*, instead of using all of the samples $\{\hat{\mathrm{x}}_{t|t-1}^{(i)}\}$, we only use its mean value,[2]

[2]Alternatively, one can use the approximate the MAP value: $\hat{\mathrm{x}}_{t|t-1} = \hat{\mathrm{x}}_{t|t-1}^{MAP}$, which requires additional evaluation of each particle's likelihood.

$\hat{\mathbf{x}}_{t|t-1} = \langle \hat{\mathbf{x}}^{(i)}_{t|t-1} \rangle$, to perform the EKF update:

$$\begin{aligned} \hat{\mathbf{x}}_{t|t} &= \hat{\mathbf{x}}_{t|t-1} + \hat{\mathbf{P}}_{t|t-1}\mathbf{G}_t^T(\mathbf{G}_t\hat{\mathbf{P}}_{t|t-1}\mathbf{G}_t^T + \Sigma_{\mathrm{v}})^{-1}(\mathbf{y}_t - \mathbf{g}(\hat{\mathbf{x}}_{t|t-1})), && (5.22)\\ \mathbf{P}_{t|t} &= \hat{\mathbf{P}}_{t|t-1} + \hat{\mathbf{P}}_{t|t-1}\mathbf{G}_t^T(\mathbf{G}_t\hat{\mathbf{P}}_{t|t-1}\mathbf{G}_t^T + \Sigma_{\mathrm{v}})^{-1}\mathbf{G}_t\hat{\mathbf{P}}_{t|t-1}, && (5.23) \end{aligned}$$

where $\mathbf{G}_t$ is the Jacobian matrix of the measurement equation, $\mathbf{P}_{t|t}$ is the filtered state covariance. Note that the filtered estimate $\hat{\mathbf{x}}_{t|t}$ is more accurate than the predicted estimate $\hat{\mathbf{x}}_{t|t-1}$, since it utilizes the most recent observation $\mathbf{y}_t$. In the meantime, the master filter, given $\mathbf{y}_t$ and the previous simulated samples $\{\mathbf{x}^{(j)}_{t-1}\}$, as well as the first-stage estimate $\hat{\mathbf{x}}_{t|t}$, runs a particle filtering procedure with a constructed suboptimal proposal distribution, and further produces a second-stage posterior estimate $\mathbf{x}^{(i)}_t$. After a complete step, the master filter propagates its samples to the slave filter for the next iteration. Note that in the slave filter, no resampling is required, whereas the master filter runs resampling only if necessary. Essentially, the two filters are trying to solve the same filtering problem but looking at it from different perspectives; each one takes advantage of the result of the other at the previous step and thereby produces the solution in a cooperative way. Due to its similarity to Turbo decoding, we call the proposed filter structure as *Turbo-particle filter* (TPF). In what follows, we will derive the update equation mathematically in detail.

Let us write the filtering posterior in a slightly different way:

$$\begin{aligned} p(\mathbf{x}_t|\mathbf{y}_{0:t}) &= p(\mathbf{x}_t|\mathbf{y}_t, \mathbf{y}_{0:t-1})\\ &= \frac{p(\mathbf{y}_t|\mathbf{x}_t, \mathbf{y}_{0:t-1})p(\mathbf{x}_t|\mathbf{y}_{0:t-1})}{p(\mathbf{y}_t|\mathbf{y}_{0:t-1})}\\ &\propto p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{0:t-1}). && (5.24) \end{aligned}$$

Next, suppose we can draw samples $\{\mathbf{x}^{(i)}_t\}$ from a proposal distribution, we need to find the importance ratios to appropriately weigh the samples. Using the importance sampling trick, we have

$$W^{(i)}_t = \frac{p(\mathbf{y}_t|\mathbf{x}^{(i)}_t)p(\mathbf{x}^{(i)}_t|\mathbf{y}_{0:t-1})}{q(\mathbf{x}^{(i)}_t|\mathbf{y}_t)}, \qquad (5.25)$$

where $q(\mathbf{x}^{(i)}_t|\mathbf{y}_t)$ is the proposal distribution. Assuming that at time t, we have N_p simulated particles for approximating the posterior density of the previous time step:

$$p(\mathbf{x}_{t-1}|\mathbf{y}_{0:t-1}) \approx \sum_{j=1}^{N_p} \tilde{W}^{(j)}_{t-1}\delta(\mathbf{x}_{t-1} - \mathbf{x}^{(j)}_{t-1}),$$

where $\tilde{W}^{(j)}_{t-1}$ are the normalized importance weights with the sum equal to unity. Hence, we have

$$\begin{aligned} p(\mathbf{x}^{(i)}_t|\mathbf{y}_{0:t-1}) &= \int p(\mathbf{x}^{(i)}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{y}_{0:t-1})d\mathbf{x}_{t-1}\\ &\approx \sum_{j=1}^{N_p} \tilde{W}^{(j)}_{t-1}p(\mathbf{x}^{(i)}_t|\mathbf{x}^{(j)}_{t-1}). && (5.26) \end{aligned}$$

Substituting (5.26) into (5.25) yields the importance weights update:

$$W_t^{(i)} = \frac{p(\mathbf{y}_t|\mathbf{x}_t^{(i)}) \sum_{j=1}^{N_p} \tilde{W}_{t-1}^{(j)} p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(j)})}{q(\mathbf{x}_t^{(i)}|\mathbf{y}_t)}. \tag{5.27}$$

Now equation (5.27) naturally combines the prior and likelihood information, using previous weights and samples as well as the most recent observation $\mathbf{y}_t$. The proposal distribution is suboptimal in the Lemma 5.3.2's sense; it is only suboptimal in that it assumes a Gaussian approximation around the mean statistic. Using the Gaussian approximation proposal (from the EKF), we can draw samples $\{\mathbf{x}_t^{(i)}\}$ from $q = \mathcal{N}(\hat{\mathbf{x}}_{t|t}, \mathbf{P}_{t|t})$, where $\mathbf{P}_{t|t}$ is obtained from (5.23). Finally, the master filter produces a (second-stage) mean estimate $\hat{\mathbf{x}}_t = \sum_{i=1}^{N_p} \tilde{W}_t^{(i)} \mathbf{x}_t^{(i)}$.

In summary, a complete-step of Turbo-particle filtering runs as follows:

1. At time t, for $j = 1, \dots, N_p$, given $\mathbf{x}_{t-1}^{(j)}$ (obtained from the master filter in the previous step) and $\mathbf{y}_t$, run the EKF updates (for the slave filter) to calculate the approximated Gaussian sufficient statistics $(\hat{\mathbf{x}}_{t|t}, \mathbf{P}_{t|t})$.
2. Draw N_p samples $\{\mathbf{x}_t^{(i)}\}$ from $\mathcal{N}(\hat{\mathbf{x}}_{t|t}, \mathbf{P}_{t|t})$.
3. For the master filter, for $i = 1, \dots, N_p$, calculate the importance weights via (5.27), and normalize them to get $\{\tilde{W}_t^{(i)}\}$.
4. Calculate the second-stage posterior estimate: $\hat{\mathbf{x}}_t = \sum_{i=1}^{N_p} \tilde{W}_t^{(i)} \mathbf{x}_t^{(i)}$.
5. Calculate $\hat{N}_{eff}$, if $\hat{N}_{eff} < N_p/2$, perform the resampling (optional).
6. Copy the N_p particles to the slave filter for the next step.

Remarks:

- Although in the above discussion we use an EKF for the slave filter, there is generally no restriction regarding the choices of implementation. For instance, the slave filter can take a form of the unscented Kalman filter (UKF) or a Gaussian sum filter; likewise, the master filter can also use a more sophisticated proposal, such as the Gaussian mixture density.[3]
- In contrast to the "*mode-tracking*" gradient proposal particle filter developed earlier, TPF employs a "*mode-shifting*" scheme to construct the "support particles" around the first-stage EKF posterior estimate, which are more informative by using the

[3]This is trivial since at the stage of slave filtering one can just use previous N_p samples to calculate the N_p pairs of "mean-covariance" statistics with a bank of EKFs, and then draw samples individually from those N_p pairs of Gaussians; in this case, the importance weights' update is evaluated from (5.11).

recent observation. The idea of the TPF is close but not identical to the existent improved particle filtering schemes (e.g., Doucet et al., 2000; deFreitas et al., 2000; Chen and Liu, 2000) in the literature;[4] although they, in one way or the other, make use of the EKF updates, the essence and implementation (e.g., the importance weights update) of their ideas are utterly different.

- Given a finite number of samples, the approximation of the $p(\mathbf{x}_t^{(i)}|\mathbf{y}_{0:t-1})$ in (5.26) may be too crude (especially when Σ_d is small); hence it might be better to use a kernel estimator (Silverman, 1986) to smooth the prior.

- The TPF is straightforward to implement. Compared to the conventional SIR filter, the complexity of the TPF is increased from $\mathcal{O}(N_p)$ (excluding the $\mathcal{O}(N_p)$ resampling complexity) to $\mathcal{O}(N_p^2)$.[5] However for the TPF, we argue that the required number of particles can be chosen to be smaller than that of the SIR filter for achieving a similar performance; on the other hand, the necessity of resampling (Step 5) becomes less frequent or even unnecessary since the weight degeneracy is no longer a problem; above all, the complexity increase may be justified by its improved performance and robustness. These will be all demonstrated in our later simulation experiments.

5.5 Sequential State Estimation in Tracking Applications

In this section, we investigate the above-studied Bayesian sequential state estimation techniques, particularly particle filtering in several tracking problems. The first two synthetic target tracking problems deal with the nonlinearity in the state-space models. The final tracking problem, being a real-life challenging task in wireless communication, deals with the non-Gaussian noise in the state-space model. As a comparison with the traditional Bayesian trackers (Kalman filter and EKF), we will see that our proposed particle filters yield a significant performance improvement in these tracking experiments.

5.5.1 Bearings-Only Target Tracking

The bearings-only tracking is a classic nonlinear filtering problem and often serves as a benchmark for particle filters (Gordon et al., 1993). Let $(\nu, \dot{\nu}, \eta, \dot{\eta})$ denote the positions and velocities of a target that is moving in the x and y coordinate. The dynamic state-space model is described as

$$\begin{aligned} \mathbf{x}_{t+1} &= \mathbf{F}\mathbf{x}_t + \mathbf{C}\mathbf{d}_t, &\quad (5.28a)\\ y_t &= \tan^{-1}(\eta_t/\nu_t) + v_t, &\quad (5.28b) \end{aligned}$$

[4]In particular, the proposal distribution obtained from local linearization of state-space model (Doucet et al., 2000) is mostly close to ours.

[5]This is due to the need of calculating $p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(j)})$.

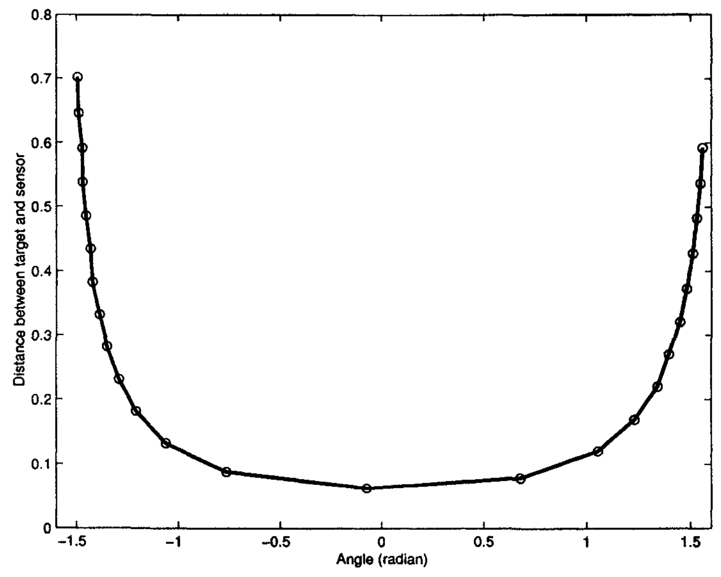

Figure 5.6: The distance between the target and sensor against the angle (bearing).

where $\mathbf{x}_t = [\nu_t, \dot{\nu}_t, \eta_t, \dot{\nu}_t]^T$, $\mathbf{d}_t = [d_{1,t}, d_{2,t}]^T$, and

$$\mathbf{F} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}.$$

The observation is a noisy bearing (angle), and $\mathbf{d}_t \sim \mathcal{N}(\mathbf{0}, 0.001^2\mathbf{I})$, $v_t \sim \mathcal{N}(0, 0.005^2)$. The goal of filtering is to reconstruct the trajectory $\mathbf{x}_{0:t}$ given the observed bearings $y_{0:t}$ and initial condition $\mathbf{x}_0 = [-0.05, 0.001, 0.7, -0.055]^T$. The priors of particle filters are set up as $p(\mathbf{x}_0) = \mathcal{N}(\mathbf{0}, \mathrm{diag}\{0.05^2, 0.005^2, 0.03^2, 0.01^2\})$ and $\mathbb{E}[\mathbf{x}_0] = [-0.06, 0.0015, 0.65, -0.05]^T$. Note that here the prior and likelihood are both peaked and that the variances of the states are of several orders of difference. In addition to the MSE, we also calculate another performance metric, the normalized MSE (NMSE):

$$\text{NMSE} = \frac{1}{T}\sum_{t=1}^{T} \frac{||\mathbf{x}_t - \hat{\mathbf{x}}_t||^2}{||\mathbf{x}_t||^2},$$

which is basically the MSE metric normalized by the power of the state variables.

It was found that the tracking performance is sensitive to the initial conditions, especially $p(\mathbf{x}_0)$. Even with a large number of particles (say $N_p = 2000$), if the variance of $p(\mathbf{x}_0)$ is large, the divergence rate of particle filters is still very high. This is because if the prior is too broad, most of samples possibly fall outside the high likelihood region, which causes all of importance weights to be very small or negligible; in the extreme case, the denominator of importance weights $\sum_{i=1}^{N_p} W_t^{(i)}$ will be close to zero. Specifically in

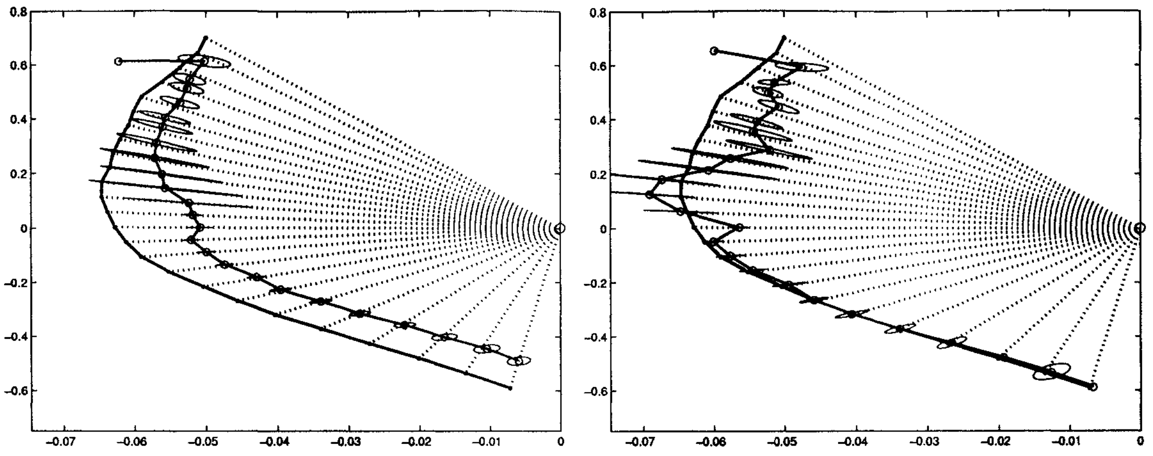

Figure 5.7: Bearings-only tracking. Dot-solid line: true target trajectory in 25 steps; circle-dashed line: one estimate obtained by the SIR filter (left) and the gradient-proposal filter (right). The ellipse describes 95% confidence interval of the estimate. The origin indicates the sensor's position.

this example, at the 14th step where the distance of target is closet to the sensor (see Figure 5.6), the estimation problem becomes most nonlinear and particle filters are mostly likely to degenerate (Gordon et al., 1993). To prevent the divergence, a Metropolis-move step, as described below, can be performed for the particle filter right after the resampling step.

- Starting with any time step t, for $i = 1, \cdots, N_p$, proceed the following:
- Generate a uniform random variable $u \sim \mathcal{U}(0,1)$, and draw a sample $\mathrm{x}' \sim q(\mathrm{x}^{(i)}, \mathrm{x}')$ or from a random walk $\mathrm{x}' \sim \mathcal{N}(\mathrm{x}^{(i)}, \Sigma_\mathrm{d})$;
- Calculate the acceptance rate $\alpha(\mathrm{x}^{(i)}, \mathrm{x}') = \min\left[\frac{p(\mathrm{y}|\mathrm{x}')}{p(\mathrm{y}|\mathrm{x}^{(i)})}, 1\right]$;
- If $u < \alpha(\mathrm{x}^{(i)}, \mathrm{x}')$, set $\mathrm{x}_{new}^{(j)} = \mathrm{x}'$, else $\mathrm{x}_{new}^{(j)} = \mathrm{x}^{(i)}$;
- Repeat above steps and store $\mathrm{x}_{new}^{(j)}$ after $5 \sim 10$ iterations of "burn-in" time;
- Return the samples $\{\mathrm{x}_{new}^{(j)}\}_{j=1}^{N_p}$.

First, we compare the proposed gradient particle filter with the conventional SIR filter, with the same initial conditions, resampling scheme, and the same number of particles ($N_p = 100$). It turns out that for the gradient proposal particle filter, the odds of divergence is less (1/50) than that of the SIR filter (3/50), the estimate of trajectory at the end is more accurate. This is because the gradient information provides some hints to prevent the filter from divergence. The selected Monte Carlo estimation results are illustrated in

Table 5.3: Experimental results of bearing-only target tracking based on 50 Monte Carlo runs (excluding the divergence trials).

	EKF	SIR ($N_p = 100$)	SIR+gradient ($N_p = 100$)	TPF ($N_p = 30$)
MSE	0.0026	0.0021	0.00014	0.00006
NMSE	0.0168	0.0150	0.0078	0.0009
divergence rate	0/50	3/50	1/50	0/50

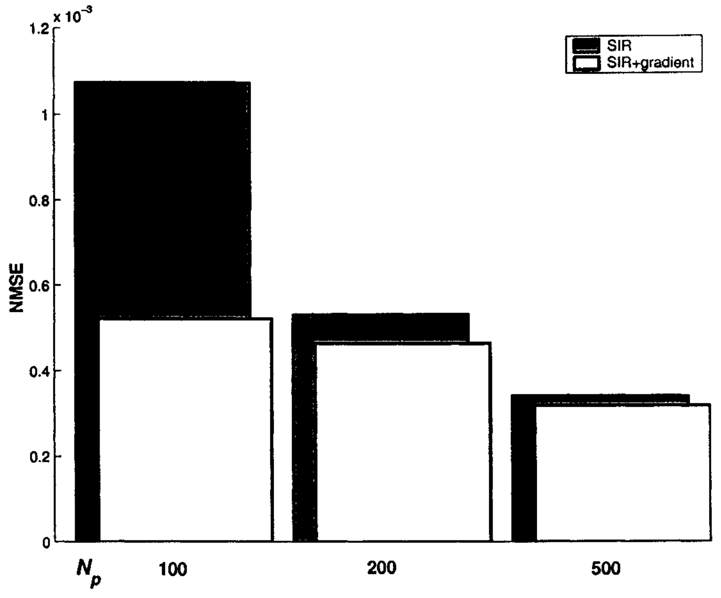

Figure 5.8: Performance (NMSE) comparison between the SIR and gradient proposal particle filters using varying numbers of particles, each based on 20 independent Monte Carlo runs.

Figure 5.7. The Monte Carlo average results of three filters in terms of one-step ahead prediction error is summarized in Table 5.3. For the EKF, no divergence is observed in 50 trials, but its predicted as well as filtered errors (for both positions and velocities) are relatively very high (even with a perfect initial condition of $\hat{\mathbf{x}}_0$, it is still worse than our particle filters). The MCMC step was found to be helpful to prevent the occurrence of divergence.

The NMSE performance comparison between the SIR and gradient proposal particle filters with different numbers of particles, is shown in Figure 5.8. It is clear that the gradient proposal particle filter outperforms the normal SIR filter under different situations, the superiority is more evident especially with a small N_p. In other words, with the same achieved performance, the gradient proposal particle filter requires fewer particles. As also expected, the difference between two algorithms gradually reduces as N_p increases.

Next, we also compare the Turbo-particle filter (TPF) with the conventional SIR filter. In the experiments, we use $N_p = 30$ particles for the master filter, for which resampling is

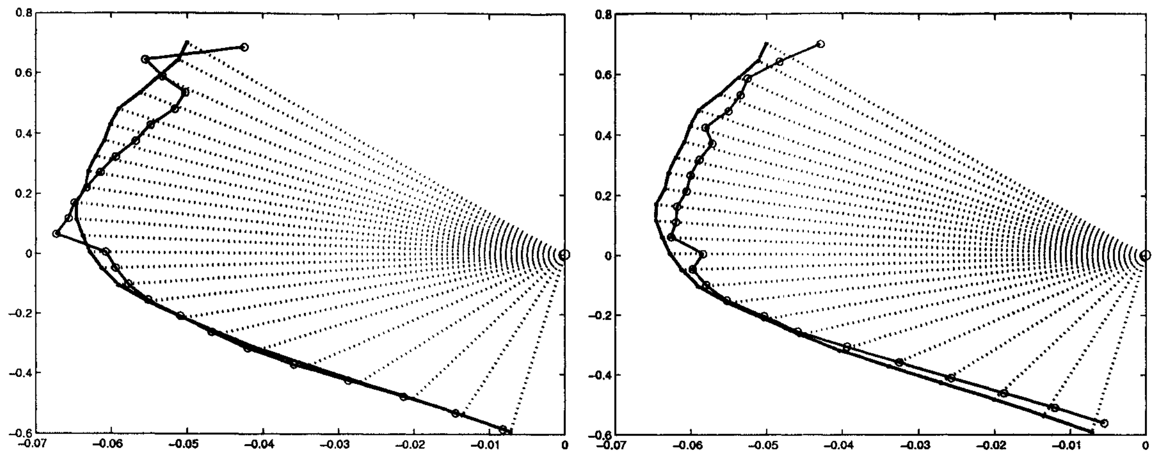

Figure 5.9: Two Monte Carlo prediction results using Turbo-particle filtering for bearings-only target tracking.

conducted only if $\hat{N}_{eff} < N_p/2$. Since the state dynamics of this tracking problem is linear, we can calculate the state covariance matrix (without approximation) in the prediction step for the slave filter. In practice, we use a weighting coefficient, $0.1 \leq \lambda \leq 0.2$, to balance the accuracy of the predicted state covariance: $\lambda \mathbf{P}_{t|t-1} + (1-\lambda)\hat{\mathbf{P}}_{t|t-1}$, where $\mathbf{P}_{t|t-1}$ denotes the calculated covariance value, whereas $\hat{\mathbf{P}}_{t|t-1}$ denotes the covariance statistic from the samples. The Monte Carlo experimental results are summarized in Table 5.3 and illustrated in Figure 5.9. As seen in the Table 5.3, the TPF's performance is significantly improved in terms of both MSE and NMSE compared to other filters, even with a small amount of the particles (less than one third of the number in other filters). Also, the TPF's solution is obviously more robust, since no divergence rate was observed. This is not surprising because the EKF has provided a good first-stage estimate for the second-stage particle filtering;[6] in fact, EKF is quite robust in terms of the divergence issue (see Table 5.3), although its prediction performance is much worse. By combing the features of the EKF and particle filter, as anticipated, TPF has yielded a very good performance for this bearing-only tracking problem. In addition, the variance of the importance weights is reduced, as evidenced by the curve of effective particle numbers illustrated in Figure 5.10. The reason why the TPF performs so well here is partially because (i) the state dynamics is linear, the EKF prediction is exact; and (ii) the state data region is small, therefore the linear approximation of the Jacobian is relatively accurate. We may expect that when these two conditions are violated, the TPF's performance will degrade. In such a case, we might replace the EKF with a more robust UKF (Julier and Uhlmann, 2004) procedure, though in the above tracking example it did not make much difference for the same reason.

[6]This is particularly true for this bearing-only tracking problem by noting that the state dynamics is a linear Gaussian model.

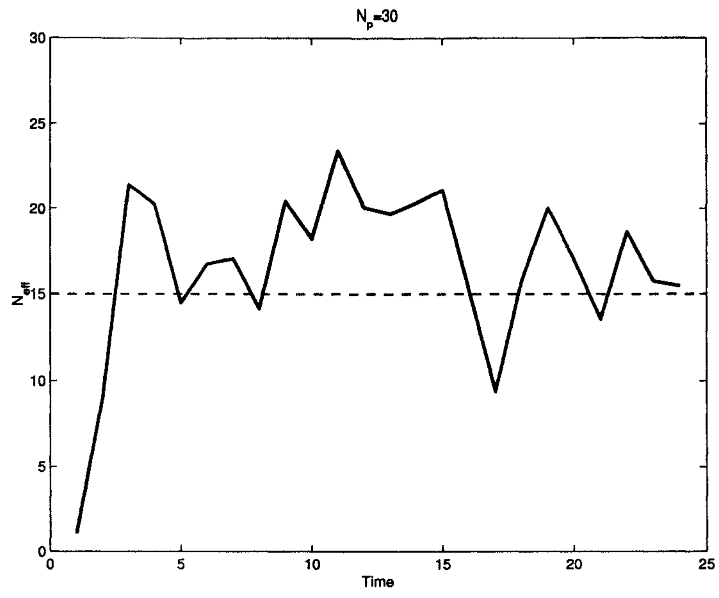

Figure 5.10: The $\hat{N}_{eff}$ curve of the master filter in Turbo-particle filtering, with $N_p = 30$; the dashed line indicates the threshold value. Typically, the weight degeneracy problem is less severe compared to the conventional SIR filter.

5.5.2 Target Tracking with Coordinate Turn

Next, we consider a typical target tracking through a coordinated turn (CT), where the state-space model is described by a stochastic differential equation (SDE) and approximated by a second-order weak Taylor approximation. See (Morelande and Gordon, 2005) for detailed background. Let $\mathbf{x}_t = [\xi_t, \varsigma_t, s_t, \theta_t, \omega_t]^T$ denote the state vector containing *target position* in x and y coordinate, *target speed, heading,* and *turn rate.* Under the SDE theory, the constant speed and turn rate (in the ideal CT model) will be instead modified as a Wiener process.

Specifically, the second-order Taylor approximation of the continuous-time state equation is described as (Morelande and Gordon, 2005):

$$\mathbf{x}_t = \mathbf{f}(\mathbf{x}_\tau) + \mathbf{G}(\mathbf{x}_\tau)\mathbf{w}_t \tag{5.29}$$

where $\delta = t - \tau$ (we use $\delta = 1$ in discretization), and

$$\begin{aligned}
\mathbf{f}(\mathbf{x}_\tau) &= \begin{pmatrix} \xi_\tau + \delta s_\tau \cos(\theta_\tau) - \delta^2 s_\tau \omega_\tau \sin(\theta_\tau)/2 \\ \varsigma_\tau + \delta s_\tau \sin(\theta_\tau) + \delta^2 s_\tau \omega_\tau \sin(\theta_\tau)/2 \\ s_\tau \\ \theta_\tau + \delta \omega_\tau \\ \omega_\tau \end{pmatrix}, \\
\mathbf{G}(\mathbf{x}_\tau) &= \mathbf{E}(\mathbf{x}_\tau)\mathbf{V}_\delta,
\end{aligned}$$

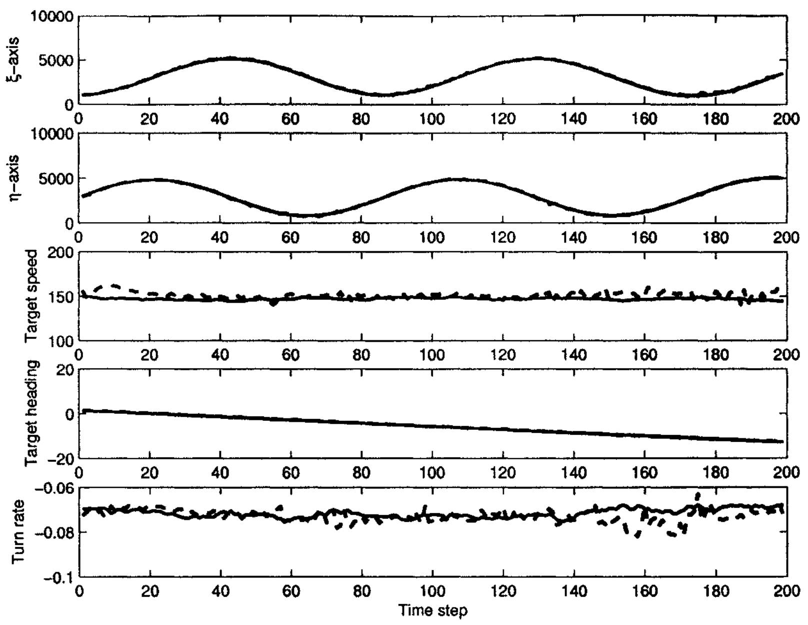

Figure 5.11: One typical result (solid line: true trajectory; dashed line: predicted trajectory) in tracking with CT experiment using gradient proposal particle filter ($N_p = 500$, RMSE=87.5).

with

$$\mathbf{E}(\mathbf{x}_\tau) = \begin{pmatrix} \sigma_s \cos(\theta_\tau) & 0 & 0 & 0 \\ \sigma_s \sin(\theta_\tau) & 0 & 0 & 0 \\ 0 & 0 & \sigma_s & 0 \\ 0 & \sigma_\omega & 0 & 0 \\ 0 & 0 & 0 & \sigma_\omega \end{pmatrix},$$

$$\mathbf{V}_\delta = \begin{pmatrix} \sqrt{\delta^3/3} & 0 \\ \sqrt{3\delta}/2 & \sqrt{\delta}/2 \end{pmatrix} \otimes \mathbf{I}_{2\times 2}$$

where $\otimes$ denotes the Kronecker product; $\mathbf{w}_t$ is a Wiener process approximated by standardized white Gaussian noise $\mathcal{N}(\mathbf{0}, \mathbf{I}_{4\times 4})$.

The measurement equation consists of a "range-bearing" pair:

$$\mathbf{y}_t = \left[\sqrt{\xi_t^2 + \varsigma_t^2},\ \tan^{-1}(\varsigma_t/\xi_t)\right]^T + \mathbf{v}_t \tag{5.30}$$

where the Gaussian measurement noise $\mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, \Sigma_\mathrm{v})$ is independent on the initial state and the Wiener process $\mathbf{w}_t$. The data trajectory was generated using the Euler approximation with sampling period of 1 second and 1000 intervals per sampling instant. Measurements are collected for 200 seconds with a constant sampling period. The noise and initial parameter setup in our experiment is as follows: $\sigma_s^2 = 1/5$, $\sigma_\omega^2 = 5 \times 10^{-7}$,

Table 5.4: The RMSE performance comparison (based on 20 Monte Carlo runs) with varying number of particles.

Filter	No. particles, N_p	RMSE
SIR	500	144.3
SIR	1000	91.6
SIR	2000	81.9
SIR+gradient	200	142.3
SIR+gradient	500	90.2
SIR+gradient	1000	74.3
TPF	100	158.7
TPF	200	149.2

$\Sigma_{\mathrm{v}} = \mathrm{diag}\{100, (\pi/180)^2\}$, $\mathbf{x}_0 \sim \mathcal{N}(\boldsymbol{\mu}_0, \Sigma_0)$ with

$$\begin{aligned}\boldsymbol{\mu}_0 &= [1000, 2650, 150, \pi/2, -\pi/45]^T,\\ \Sigma_0 &= \mathrm{diag}\{400, 400, 25, (5\pi/180)^2, (0.2\pi/180)^2\}.\end{aligned}$$

Note that here the dynamic noise is peaked whereas the measurement noise is rather flat. Table 5.4 shows the averaged RMSE (RMSE $\equiv \sqrt{|\xi - \hat{\xi}|^2 + |\varsigma - \hat{\varsigma}|^2}$) comparison between the bootstrap filter (with prior proposal) and the SIR filter (with gradient proposal) and TPF, with varying number of particles. Figure 5.11 shows one typical tracking result. As anticipated and evidenced again, the gradient proposal particle filter outperforms the bootstrap filter with the same number of particles; and with less number of particles ($N_p = 100$), TPF achieves approximately the same performance as the bootstrap filter ($N_p = 500$).

5.5.3 MIMO Wireless Channel Estimation.

In what follows, we apply the sequential Bayesian estimation technique for tracking fluctuations in the channel response of a narrowband multiple-input, multiple-output (MIMO) wireless communication system, using real-life recorded data. The channel response of the wireless channel is known to be time-varying (hence so called "fading channel") due to the multipath effect during the propagation process, thereby presenting a challenge for reliable wireless communications (see e.g., Haykin and Moher, 2004). The motivation for channel tracking/estimation is to permit the use of a blind or semiblind strategy for acquiring knowledge of the channel matrix at the receiver, thereby improving the receiver noise performance of the MIMO wireless communication system. The work reported here is a challenging real-life problem and provide a testbed for different sequential channel estimation techniques.[7]

[7]This is a joint work with Kris Huber, and the following content is partially taken from a coauthored paper (Haykin, Huber, and Chen, 2004).

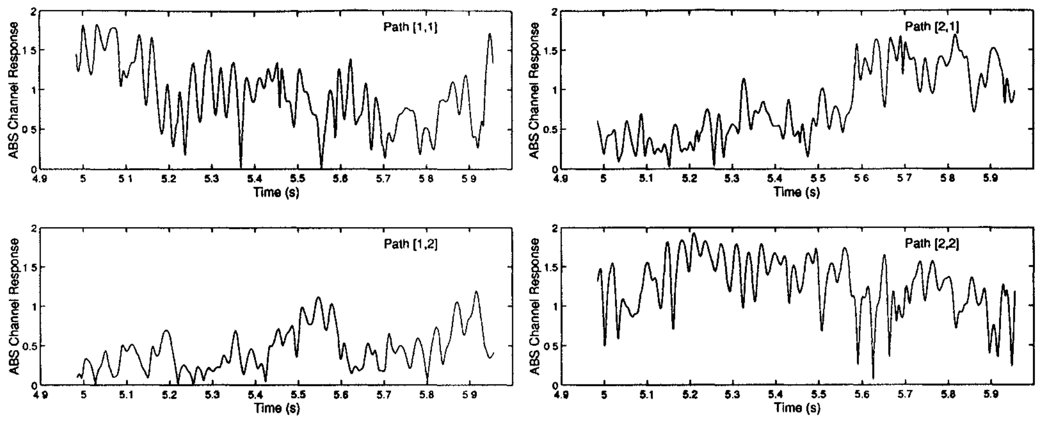

Figure 5.12: A typical time snapshot of channel response for each of the four independent paths. Each curve illustrates the instantaneous magnitude (absolute value) of the channel fading gain at a particular time.

Data. The real-life wireless channel data used in the succeeding experiments are the results of a survey of narrowband MIMO channel measurements collected in midtown Manhattan, New York city, January 2001.[8] A narrowband channel sounder was built to measure the complex channel coefficients between a 16-element transmit antenna array and a 16-element receive antenna array. See (Haykin et al., 2004) for more details about the experimental data. A typical time snapshot for each path realization of a sample two-transmitter, two-receiver ($N_t = 2, N_r = 2$) MIMO system (a total of four independent paths) is given in Figure 5.12. The normalized time-Doppler rate was 0.01. Clearly the dynamic oscillations of these paths make tracking the wireless channel a challenging endeavor. Since we are simulating a 2-by-2 MIMO setup, each state vector contains a total of four individual paths, which correspond to a total of eight channel coefficients when considering both the real and imaginary components.

Wireless Channel Dynamics. First, let us consider the state equation. Assuming a memory span of p blocks, the channel is approximated by previous memory that consists of an AR(p) model:[9]

$$x_{jk,n} = \sum_{\ell=0}^{p} \beta_{l,n} x_{jk,n-\ell} + d_{jk,n}, \quad \text{for } j = 1, 2, \cdots, N_r, \quad k = 1, 2, \cdots, N_t \tag{5.31}$$

where the $\beta_{\ell,n}$ are time-varying autoregressive (AR) coefficients, and $d_{jk,n}$ is the *dynamic noise* that drives the state equation. In effect, (5.31) describes the evolution of each scalar coefficient of a (constructive) channel matrix $\mathbf{X}_n$ as an AR model of order p. Here, we

[8]This was a joint project between McMaster University and Bell Laboratories, Lucent Technologies.

[9]In order to be consistent with the common notations in communications, hereafter we use subscript n to denote the time index for both state and symbol variables.

use a first-order AR model:[10]

$$x_{jk,n} = \beta_n x_{jk,n-1} + d_{jk,n}. \tag{5.32}$$

where the AR coefficient β_n is inferred from the *Jakes' fading model*, a well-established model for wireless fading channel (Jakes, 1974). The dynamic noise is modeled as a mixture of two nonzero-mean Gaussian components:

$$d_{jk,n} \sim \delta\mathbb{C}(\mu, \sigma^2) + (1-\delta)\mathbb{C}(-\mu, \sigma^2). \tag{5.33}$$

where $\mathbb{C}(\mu, \sigma^2)$ denotes a complex Gaussian distribution with mean μ and variance σ^2. The choices of δ, as well as the noise statistics μ and σ^2 in (5.33), were estimated off-line based on the real-life data at hand and fixed in the experiments. For detailed discussion and justification, the reader is referred to (Haykin et al., 2004).

Next, the measurement equation is described as

$$y_{j,n} = \sum_{k=1}^{N_t} s_{k,n} x_{jk,n} + v_{j,n} \quad \text{for } j = 1, 2, \cdots, N_r \tag{5.34}$$

where $s_{k,n}$ is the block of encoded symbols radiated by the kth transmit antenna at time n, $v_{j,n}$ is the measurement noise at the input of the jth receive antenna at time n, $x_{jk,n}$ is the channel coefficient from the kth transmit antenna to the jth receive antenna at time n, and $y_{j,n}$ is the signal observed at the input of the jth receive antenna. The measurement noise we use is the *Middleton Class A noise model*, which has been used extensively to model the impulsive noise commonly generated in an indoor/urban wireless environment (Blackard et al., 1993; Wang and Poor, 1999). The probability density of this model is given by

$$p(v) = (1-\epsilon)\mathbb{C}(0, \varsigma^2) + \epsilon\mathbb{C}(0, \kappa\varsigma^2), \tag{5.35}$$

where $0 \leq \epsilon \leq 1$. The first component $\mathbb{C}(0, \varsigma^2)$ represents the ambient background noise, while $\mathbb{C}(0, \kappa\varsigma^2)$ represents an impulsive component with probability ϵ. By varying the parameters ϵ and κ we can fix the total variance as $\sigma^2 = (1-\epsilon)\varsigma^2 + \epsilon\kappa\varsigma^2$.

Channel and Symbol Joint Estimation. The proposed MIMO transceiver structure is shown in Figure 5.13. Binary source information is first mapped using 8-PSK (phase-shift keying) scheme and sent to the channel encoder. For the experimental purpose, we adopted a 2-by-2 antenna configuration using a popular coding scheme known as *space-time block encoding* (Alamouti, 1998). The encoded symbols $\mathbf{s}_n$ are then transmitted over the wireless channel where they are attenuated by the channel matrix $\mathbf{X}_n$. On the receiver side, the Bayesian estimator is built around the channel codes being used. After a training

[10]Recent results (Huber and Haykin, 2004) have used a dynamic higher-order AR model for tracking improvement.

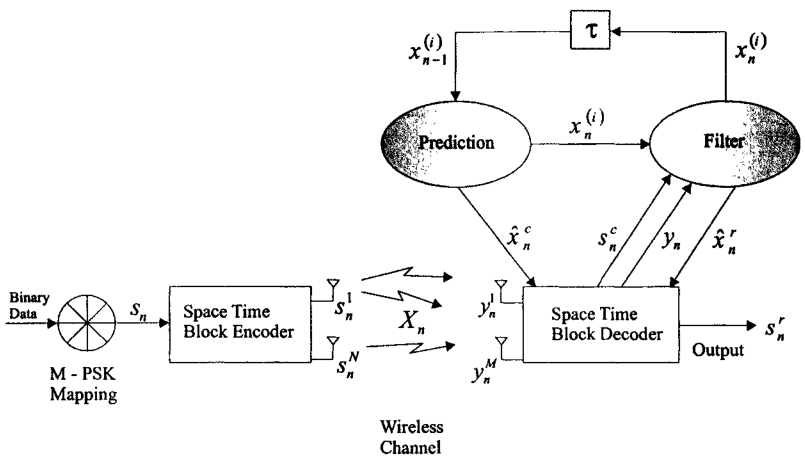

Figure 5.13: A schematic block diagram of joint estimation (tracking-decoding) in a MIMO wireless system.

period that is used to initialize the filter, the prediction portion of the estimator updates the particles from the previous state estimate $\mathbf{x}_{n-1}^{(i)}$ to the current predicted state estimate $\mathbf{x}_n^{(i)}$. The mean state estimate $\hat{\mathbf{x}}_n^c$ of the predicted particles is sent to the *space-time block decoder* (STBD), while the particles themselves are propagated to the filter portion of the receiver. The STBD then uses the mean channel estimate, as well as the received signal $\mathbf{y}_n$ (from the receive antennas) in order to produce a coarse estimate $\mathbf{s}_n^c$ of the transmitted symbols $\mathbf{s}_n$. The filter portion of the estimator then uses the predicted particles, along with the received signal and coarse estimate of the transmit symbols to produce new filtered versions of the particles $\mathbf{x}_n^{(i)}$ as well as obtain a refined mean state estimate $\hat{\mathbf{x}}_n^r$ of the channel. Finally, the refined channel estimate $\hat{\mathbf{x}}_n^r$ and the received signal $\mathbf{y}_n$ are processed by the STBD to give the ultimate symbol estimates of the transmitted symbols $\mathbf{s}_n^r$. The filtered particles, upon time delay τ, are propagated again in the prediction of the channel state for time $n+1$.

Setup. The parameter values used for (5.23) and (5.24) were set as follows: $\beta = 0.999$, $\mu = 0.044$, $\sigma^2 = 0.0039$. We assumed that the fading rate remains unchanged (i.e., the user is moving at a constant speed), hence β_n is maintained at the same value for all time n. In order to demonstrate the improved performance offered by sequential channel tracking, we implement the Bayesian receivers under both (*blind channel tracking* and *semiblind channel tracking*) scenarios discussed in (Haykin et al., 2004). Finally, we show the results for the optimal case when the receiver has perfect channel state information; while this is an ideal case, it provides a meaningful lower bound for our receiver performance.

Performance Analysis. First, we consider the performance of the Bayesian sequential Monte Carlo receivers under the blind channel tracking scenario. We consider several popular Bayesian receivers, namely, the Kalman filter, the mixture Kalman filter (MKF),

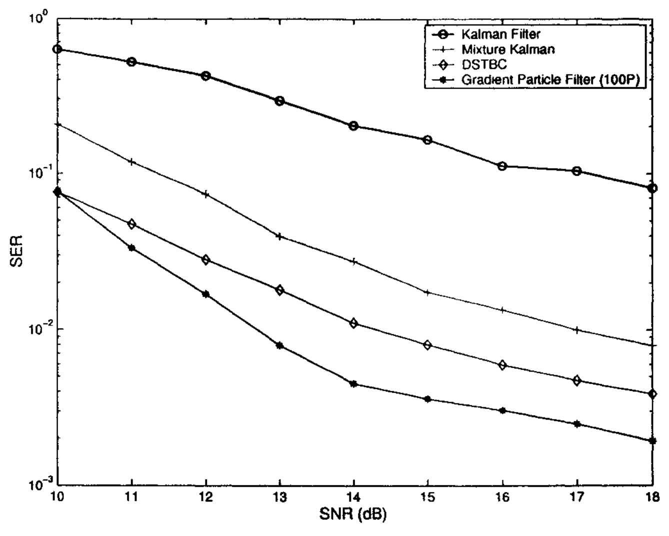

Figure 5.14: Symbol error rate (SER) performance for blind channel tracking.

and finally the particle filters. For the MKF the number of Monte Carlo samples was empirically set as $N_p = 50$. Simulation results showed little improvement for increasing N_p, while it degraded notably for $N_p \leq 25$. Compared to the common SIR particle filter using the prior as proposal, the particle filter including the gradient information into the proposal is referred to as gradient proposal particle filter. Finally, with blind channel tracking in mind, for completeness we plot the performance of a popular differential scheme, using the *differential space-time block codes* (DSTBC, Tarokh and Jafarkhani, 2000).

In Figure 5.14, the SER of the receivers in the blind channel tracking case is given, which is calculated using 20,000 symbols. It is seen from Figure 5.14 that at low SNR only the particle filter is able to offer a comparable performance to that of the DSTBC. The degraded performance at lower SNR is owing to the fact that the Bayesian receivers must deduce the channel state based solely on their estimate of the transmitted symbols. At low transmit power, too many symbol estimation errors are incurred in order for the receiver to obtain a meaningful estimate of the channel, namely, they are suffering from a form of error propagation. However, as the noise power decreases, the error rate falls and then for the most part only correct decisions are reinforced.

It is clearly demonstrated that the Kalman filter is unable to maintain reliable tracking of the channel (i.e., it diverges due to excessive error propagation). Only as the noise power of the channel becomes reduced is the Kalman filter able to operate in a blind environment. Similarly, the MKF initially scores poorly; however, unlike the Kalman filter, the MKF does not diverge and its performance rapidly improves as SNR is increased. On the other hand, the gradient proposal particle filter does not suffer as much as the others. This is due to the feedback effect of the gradient technique. Since the gradient information is very efficient at placing particles in high likelihood regions of the posterior, it provides the

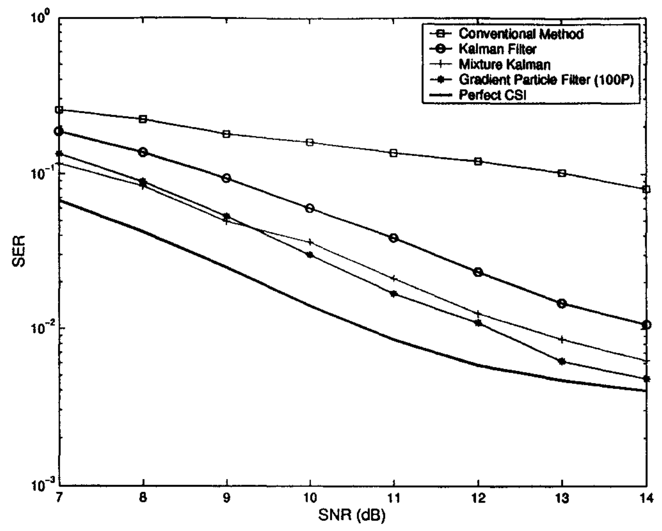

Figure 5.15: Symbol error rate (SER) performance for semiblind channel tracking.

particle filter with a more accurate estimate of the channel. As can be seen in Figure 5.14, at medium SNR larger than 10dB, the gradient proposal particle filter is superior to that of the DSTBC, and by 13dB, the gradient proposal particle filter has beaten the DSTBC by a full 3dB.

Next we consider the performance of the receivers under the semiblind channel tracking circumstance, in which, in addition to the data symbols, the receiver receives a periodic known symbol sequence. Here the transmitter sends one training block for every 10 transmitted blocks (i.e., with a training ratio of 10%). As seen from Figure 5.15, the conventional receiver that assumes a static channel between successive training blocks clearly performs the worst. In order to improve the performance, it is necessary to increase the training rate (i.e. sending training sequence more often) or reduce the symbol constellation size. However, this is wasteful of both channel bandwidth and transmitted power and thereby not a desirable solution. On the other hand, the Bayesian receivers offer much better performance. Unlike the blind channel tracking, the inclusion of training symbols prevents all of the Bayesian filters from divergence. The Kalman filter is now able to track the channel, despite its poorer performance compared to the MKF and gradient proposal particle filter. In the semiblind scenario, it is observed that the gradient proposal particle filter performs the best among the Bayesian trackers. In fact, as the SNR approaches 14dB, the gradient proposal particle filter is performing nearly as well as the optimal receiver, which assumes the perfect knowledge of the fading gains.

Gradient Analysis. Monte Carlo experimental results illustrating the performance gain in using the gradient information are shown in Table 5.5 and illustrated in Figure 5.16. In considering the curves in Figure 5.16, it is clearly evident that the benefit of the gradient technique is to dramatically reduce the number of required particles, while maintaining

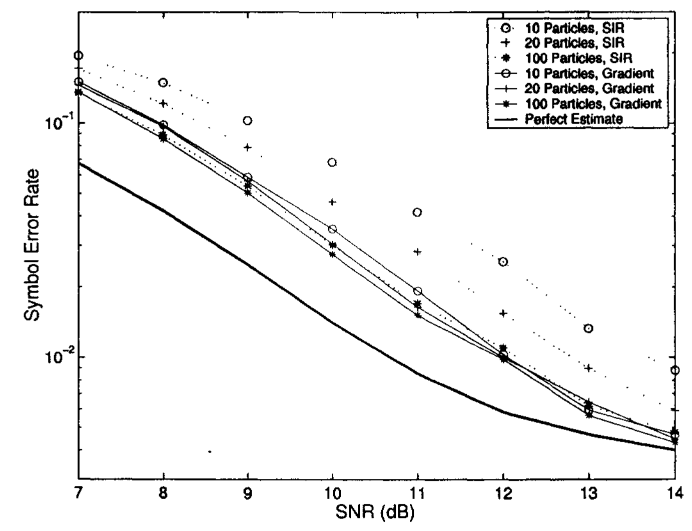

Figure 5.16: Symbol error rate (SER) performance curves for blind channel tracking.

a comparable performance to the SIR algorithm. For example at 12dB SNR and using 10 particles, we can obtain a 2dB performance improvement over the SIR filter. In terms of SER comparison, it is seen in Table 5.5 that when the number of particles is less than 100, the gradient proposal particle filter obtains the same SER as the SIR filter, with only a quarter of the particles needed for the SIR filter. The reason for this dramatic improvement is due to the improved proposal afforded by the gradient information. The gradient proposal is significantly efficient in placing the particles in high likelihood regions, thus fewer particles are required to adequately reconstruct the posterior. It is also important to note that if the number of particles is large (see the rightmost column of Table 5.5), then the gradient information will offer little to further improvement. Once the state space has become over populated with particles (due to a large N_p), the SIR filter, while being less efficient, may still place enough particles in high likelihood regions to adequately estimate the posterior. Thus the gradient scheme cannot improve on the best possible performance attainable by the SIR filter with a large number of particles; the merit of the gradient proposal particle filter is enhanced robustness, and significantly reduced level of complexity.

Complexity Analysis. Due to the nature of sequential Monte Carlo sampling, it is difficult to obtain accurate expressions for run-time analysis of each tracking algorithm. While it is true that the complexity orders of the Kalman and particle filters can roughly be described as $\mathcal{O}(N_{\rm x}^2)$ and $\mathcal{O}(N_p N_{\rm x})$ respectively ($N_{\rm x}$ denotes the dimensionality of the state, and N_p denotes the number of particles), these values are only approximate and offer little information as to their CPU run-time in a real system. For example, the form of the proposal, predictive model, resampling algorithm, as well as the number of Monte Carlo samples may have significant influences on the actual run-time of each algorithm. In an attempt to provide a fair comparison of all Bayesian trackers, each algorithm is

Table 5.5: The MSE of wireless channel estimate and SER for various numbers of particles (at 10dB SNR) based on 100 Monte Carlo runs.

Number of	SIR		SIR+gradient	
Particles	MSE	SER	MSE	SER
10	0.0615	0.0681	0.0233	0.0353
20	0.0431	0.0460	0.0206	0.0305
40	0.0338	0.0387	0.0196	0.0291
100	0.0257	0.0301	0.0188	0.0275
200	0.0227	0.0285	0.0183	0.0272

normalized with respect to a single Kalman filter in terms of complexity and performance. The CPU time is calculated based on 1000 blocks of data; the Kalman filter is defined to have a complexity factor of unity. Then, an algorithm with a complexity of three, say, would take three times as long to process the same amount of data. Similarly for the performance measure, the Kalman filter is operated in the semiblind case at 10dB SNR as benchmark. Again, an algorithm showing better performance than the Kalman filter will have performance measure greater than one.

Figure 5.17 illustrates the complexity/performance trade-off between the three types of Bayesian receivers (the number in brackets after the receiver name represents the number of Monte Carlo samples or the number of Kalman filter mixtures). As can be seen by the solid black bar in the figure, all the filters using Monte Carlo estimation produce similar performance. However, their complexities vary significantly. For example, the MKF has a run-time complexity of almost 30 times that of the Kalman filter, well above that of the other types of filters.

Returning to the influence of gradient information on overall performance, we see that the gradient proposal particle filter operating with 20 particles has a complexity factor of 1.6 and a performance factor of 2.8. Comparing this to the common SIR filter using 20 particles, we see the complexity factor is slightly less than 1.5, whereas the performance factor is only 1.8. Considering the complexity/performance trade-off, it is seen that using gradient information increases run-time complexity by only 6%, while providing a 36% increase in performance over the common SIR filter! Thus for a minimal increase in complexity we again see the benefit of the gradient proposal particle filter.

A Further Comparison with Turbo-Particle Filter. Finally, we also investigate and compare the performance of the early-developed Turbo-particle filter (TPF) in the semiblind channel tracking scenario. Similar to the earlier investigations, we used the 2-by-2 (two transmitters and two receivers) MIMO system; the channel fading (time-Doppler) rate is 0.01 (fast fading) in our experimental setup. To calculate the SER, we have used $30 \sim 50$ particles for the TPF, while using 100 particles for both the conventional SIR and the gradient proposal particle filter. Since the state-space model here is linear and non-Gaussian, we essentially run the approximate Kalman filtering in the slave filter. Using

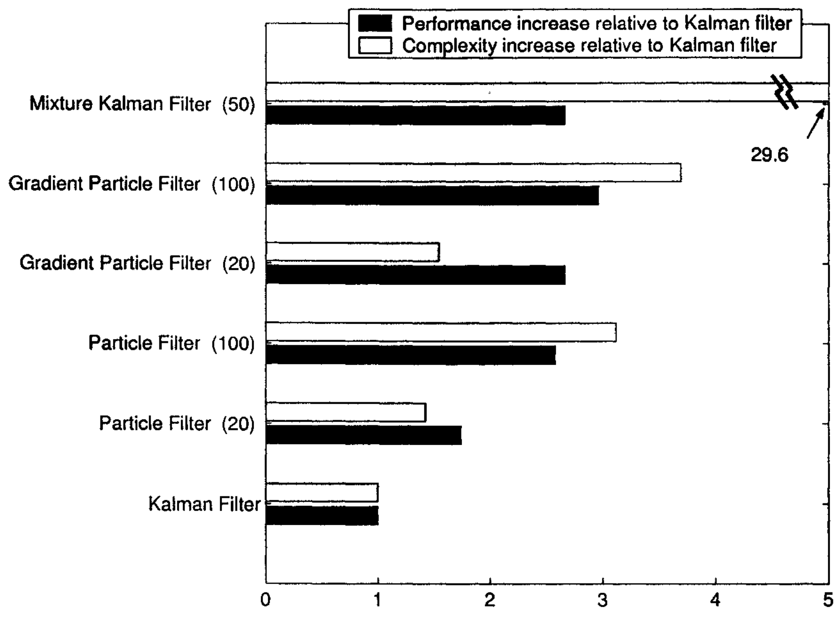

Figure 5.17: The performance/complexity comparison for various Bayesian receivers. The number in brackets after the trackers represents the number of Monte Carlo samples drawn at each time instant.

10,000 QPSK symbols, the SER performance curves are shown in Figure 5.18 with the STBD. Compared with the SIR and gradient proposal particle filter, it is seen that the TPF produces equal or better performance with less number of particles.

It is noteworthy to mention that we have also tried a different decoding scheme in the experiments, such as the Vertical-BLAST (Bell Laboratories Layered Space-Time) [URL: http://www.bell-labs.com/project/blast/], or V-BLAST, another popular decoding algorithm designed for MIMO wireless communication (see e.g., Wolniansky et al., 1998; Haykin and Moher, 2004).[11] Similar observations are found in the related experiments, see Figure 5.19 for the illustration. In the experiments, we used two transmit antennae and three receive antennae; the channel fading rate is also 0.01; a total of 20,000 QPSK symbols are used to calculate the SER curves (from 10 to 18dB). As seen again, with equal or less number of particles, the TPF has achieved better performance than the Kalman filter and the gradient proposal particle filter. With 100 particles, at low SNR the TPF almost reaches the lower bound of the ideal state (i.e., with the perfect knowledge of the channel). On the other hand, at high SNR the performance curves get flat and

[11]As a rough comparison, V-BLAST often requires more receivers than the number of transmitters, and its performance is less dependent on the number of transmitters; it often works better in the high SNR scenario. In contrast, STBD often uses more transmitters than receivers, and its performance is less dependent on the number of the receivers; with sufficient number of transmitters, STBD can work reasonably well under a rather low SNR. With the same condition, STBD often achieves a lower SER than V-BLAST since it uses redundant data, whereas V-BLAST is more bandwidth-efficient; these two schemes illustrate the trade-off between the *data redundancy* and *spectral efficiency*.

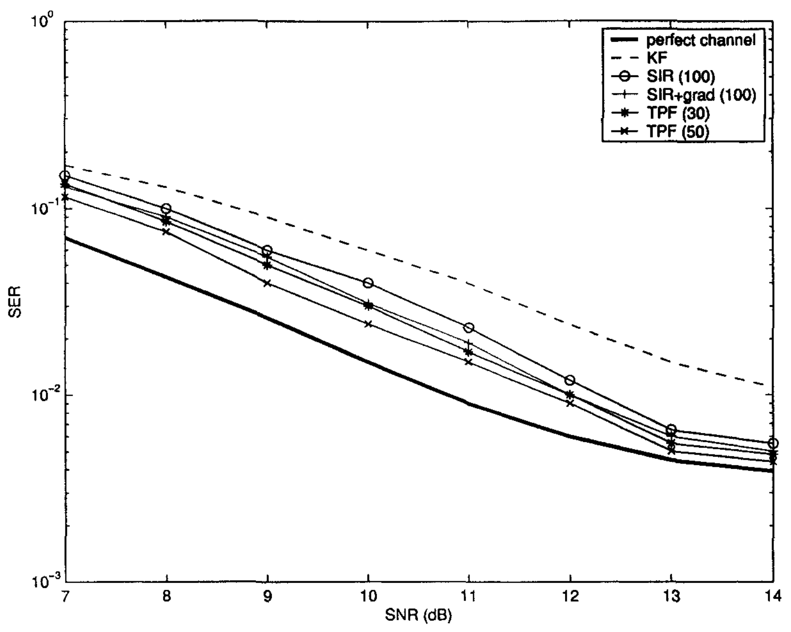

Figure 5.18: Symbol error rate (SER) comparison in semiblind channel tracking. The number in the bracket indicates the number of particles in use.

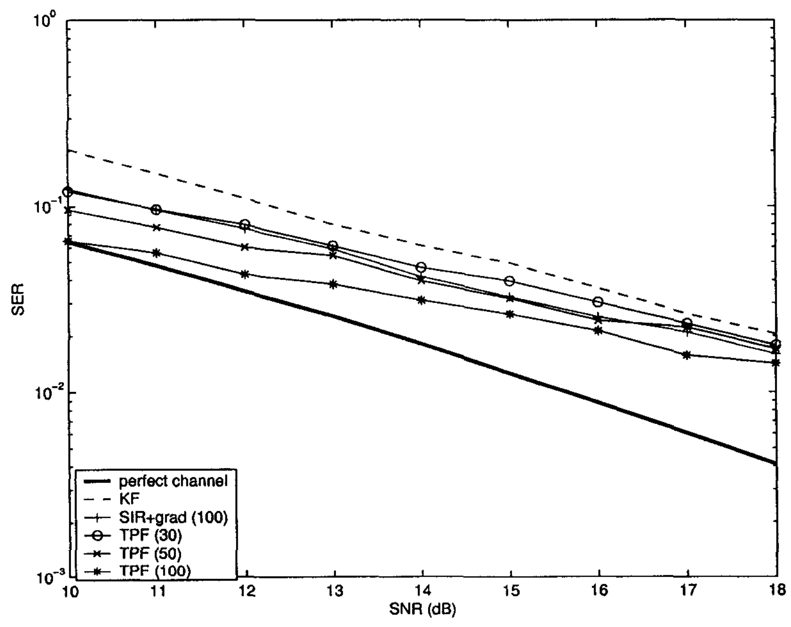

Figure 5.19: Symbol error rate (SER) performance curves using TPF and V-BLAST in semiblind channel tracking. The number in the bracket indicates the number of particles in use.

gradually deviate from the idea state. This "bottleneck" phenomenon (especially for high channel fading rate) is because at high SNR the measurement noise effect almost vanishes, and the state equation model (i.e., the AR model and dynamic noise) will dominate the performance; it can be imagined that if there is some model mismatch in the state equation (currently, we use fixed AR coefficient and fixed dynamic noise model), the performance will be still poor no matter how high the SNR is. This problem can somewhat be rescued using a dynamical higher-order AR model to track the time-varying AR coefficients (Kris Huber, personal communication); however, this is beyond our focus here.

In terms of computational complexity, the TPF is more costly than the conventional SIR filter as well as the gradient proposal particle filter. As a comparison with Figure 5.17, the relative performance/complexity factor of the TPF (using 30 particles) w.r.t. the Kalman filter is about 2.3/3. In contrast, the performance/complexity factor of the gradient proposal particle filter with 20 and 100 particles is 1.5/2.8 and 3.6/3, respectively; therefore, we pay the price of increased complexity for better performance. In light of $\frac{1.5}{2.8} < \frac{2.3}{3} < \frac{3.6}{3}$, it is seen that we have an option for the complexity and performance tradeoff: if the complexity and memory storage is the main concern, we may use TPF with a small number of particles; on the other hand, if the high performance gain is preferred, we choose a gradient proposal particle filter with a large number of particles.

5.6 Discussion

In this chapter, we have presented a systematical overview of Monte Carlo methods for Bayesian estimation. For particle filters, choosing a good proposal distribution is pivotal to the ultimate performance. However, selection of the proposal distribution is often problem dependent, choosing different proposals essentially leads to different particle filters. Although there is no general guideline regarding the selection of the proposal, we did find some heuristic and empirical observations from the experiments. In addition, in order to improve the basic bootstrap filter (using transition prior as proposal and ignoring the recent observation), we proposed two improved particle filtering schemes and demonstrated their potential merits in several tracking applications.

Our first improved scheme is based on the heuristics and can be viewed as a form of *data augmentation*. Using the gradient information from the measurement equation naturally utilizes the observation information and thus push the particles to the high-likelihood region. The implementation of this idea is simple, with only a slight computational complexity increase compared with the bootstrap filter. Surprisingly, the brute-force gradient proposal is quite efficient under various noise situations, especially in situations with a small Σ_v (compared with Σ_d). We have found that in the above-studied tracking experiments, the gradient proposal particle filter always outperforms the bootstrap filter using the same number of the particles; the difference is especially highlighted when the number of particles is small. In the wireless MIMO channel tracking case, we have seen that, in terms of *relative* complexity and performance gain (compared with the Kalman filter), the relative complexity factors of the SIR (with 100 particles) and the gradient SIR filters (with 20 particles) are 3.2 and 1.5, respectively; while their performance gains are 2.6 and 2.7, respectively! This huge gain is rather significant when computational complexity issue is concerned in industrial practice.

Our second improved scheme is motivated from the Turbo decoding principle. The Turbo-particle filter (TPF) uses two important concepts for Turbo processing: *feedback* and *divide-and-conquer.* Through the feedback, the slave filter uses the most recent observation and further provides a one-step-ahead posterior estimate for constructing a Gaussian proposal for the master filter. By using the two-filter structure, the difficulty in constructing an optimal proposal distribution is broken into two parts, and each part is conquered by one filter. The two filters self-bootstrap iteratively and produce a suboptimal Bayesian solution. It should be cautioned that we have *not* claimed that the TPF always outperforms the conventional bootstrap filter. The performance difference between them might be very small depending on the estimation problem at hand. Given the simplicity of the latter, the bootstrap filter might be still favorable in some cases; on the other hand, in other cases (such as the linear dynamics with non-Gaussian noise, or partial Gaussian dynamics in either state or measurement equation), TPF might have its own advantage of variance reduction (since Kalman filter gradually reduces the state-error variance). The empirical observations we have evidenced might serve as a general guideline: when the variances Σ_{d} and Σ_{v} are both small, or either the state or measurement equation is linear, TPF usually performs quite well; when Σ_{d} is small compared to a larger Σ_{v}, the conventional bootstrap filter would produce a good performance given a sufficient number of particles; when Σ_{d} is flat compared to a peaked likelihood with a small Σ_{v}, the bootstrap filter usually is not efficient, and therefore other improved schemes (such as the auxiliary variable, or the gradient proposal) are often required to rescue.

Choosing different improved particle filtering schemes depends on the problem at hand as well as the prior knowledge of the dynamical and measurement noise. For instance, since the gradient proposal needs the calculation of the gradient (Jacobian) of the measurement equation, it is therefore more efficient for the nonlinear state-space model with linear measurement equation. On the other hand, since the TPF involves the EKF update in the slave filter, it can be imagined that if the state equation is linear and Gaussian, the state covariance matrix update can be replaced with the sample covariance estimate, and the computation is still exact; hence it is quite efficient for the nonlinear state-space model with linear state equation (e.g., the target tracking example).

Eventually, the issue of performance and complexity tradeoff of particle filtering is central: choosing a better proposal distribution (with more computation per step) might reduce the total memory storage and complexity (in terms of required particle number and the resampling frequency). In practice, we might design certain decision criteria for using different particle filters under different scenarios; in other words, we have to study the problem first and then choose a problem-specific filter.

5.7 Appendix: MOVE-Step Analysis in Gradient Proposal Particle Filter

This appendix is devoted to examining the MOVE-step in gradient proposal particle filter, regarding the use of the observation and step-size parameter.[12]

For simplicity of analysis, let us first consider a linear Gaussian state-space model:

$$\begin{aligned} \mathbf{x}_t &= \mathbf{F}\mathbf{x}_{t-1} + \mathbf{d}_t, & (5.36)\\ \mathbf{y}_t &= \mathbf{G}\mathbf{x}_t + \mathbf{v}_t, & (5.37) \end{aligned}$$

where the dynamical and measurement noises are uncorrelated, and $\mathbf{d}_t \sim \mathcal{N}(0, \Sigma_\mathbf{d})$ and $\mathbf{v}_t \sim \mathcal{N}(0, \Sigma_\mathbf{v})$. Before propagating new particles, the gradient proposal particle filter moves each particle (say, $\mathbf{x}_{t-1}^{(i)}$) to $\mathbf{x}_{t-1}^{(i)} - \eta\xi(\mathbf{y})$ according to the gradient MOVE-step, where η is a small step-size scalar, and $\xi(\mathbf{y})$ denotes the gradient estimation:

$$\xi(\mathbf{y}) = \left\{\nabla_\mathbf{x}|\mathbf{y} - \mathbf{G}\mathbf{x}|^2\right\}\Big|_{\mathbf{x}=\mathbf{x}_{t-1}^{(i)}}, \tag{5.38}$$

with $\nabla_\mathbf{x}$ denoting the gradient operator. Now let us compare the performance of $\xi(\mathbf{y}_t)$ and $\xi(\mathbf{y}_{t-1})$ (namely, using current and previous observations).

The success measure of the MOVE-step we use here is the L_2 norm error between the target state and the particle state after the MOVE-step. Let us define

$$\begin{aligned} Error &= |\mathbf{x}_{t-1} - \mathbf{x}_{t-1}^{(i)} + \eta\xi(\mathbf{y})|^2 \\ &= |\mathbf{x}_{t-1} - \mathbf{x}_{t-1}^{(i)}|^2 + 2\eta\xi(\mathbf{y})^T(\mathbf{x}_{t-1} - \mathbf{x}_{t-1}^{(i)}) + \eta^2\xi(\mathbf{y})^T\xi(\mathbf{y}). & (5.39) \end{aligned}$$

Thus, the MOVE-step can be considered beneficial if the error in (5.39) is smaller than $|\mathbf{x}_{t-1} - \mathbf{x}_{t-1}^{(i)}|^2$; this occurs if

$$\eta^2\xi(\mathbf{y})^T\xi(\mathbf{y}) < -2\eta\xi(\mathbf{y})^T(\mathbf{x}_{t-1} - \mathbf{x}_{t-1}^{(i)}). \tag{5.40}$$

We rewrite the function $\xi(\mathbf{y})$ of (5.38) in an explicit form:

$$\begin{aligned} \xi(\mathbf{y}) &= \nabla_\mathbf{x}\Big(\mathbf{y}^T\mathbf{y} - \mathbf{y}^T\mathbf{G}\mathbf{x}_{t-1}^{(i)} - (\mathbf{G}\mathbf{x}_{t-1}^{(i)})^T\mathbf{y} + (\mathbf{G}\mathbf{x}_{t-1}^{(i)})^T(\mathbf{G}\mathbf{x}_{t-1}^{(i)})\Big) \\ &= 2\mathbf{G}^T(\mathbf{G}\mathbf{x}_{t-1}^{(i)} - \mathbf{y}). & (5.41) \end{aligned}$$

Case 1: When the measurement $\mathbf{y} = \mathbf{y}_t$ is used. In this case, we have

$$\begin{aligned} \xi(\mathbf{y}_t) &= 2\mathbf{G}^T(\mathbf{G}\mathbf{x}_{t-1}^{(i)} - \mathbf{G}\mathbf{x}_t - \mathbf{v}_t) \\ &= 2\mathbf{G}^T\Big(\mathbf{G}\mathbf{x}_{t-1}^{(i)} - \mathbf{G}(\mathbf{F}\mathbf{x}_{t-1} + \mathbf{d}_t) - \mathbf{v}_t\Big) \\ &= -2\mathbf{G}^T\Big(\mathbf{G}(\mathbf{F}\mathbf{x}_{t-1} + \mathbf{d}_t - \mathbf{x}_{t-1}^{(i)}) + \mathbf{v}_t\Big). & (5.42) \end{aligned}$$

[12]The following content partially originated from the discussion and personal communication with Dr. Mark Morelande.

Substituting (5.42) into (5.40), the criterion for improving the error performance becomes

$$4\eta\Big\{\Big(\mathbf{G}(\mathbf{F}\mathbf{x}_{t-1}+\mathbf{d}_t-\mathbf{x}_{t-1}^{(i)})+\mathbf{v}_t\Big)^T\mathbf{G}\mathbf{G}^T\Big(\mathbf{G}(\mathbf{F}\mathbf{x}_{t-1}+\mathbf{d}_t-\mathbf{x}_{t-1}^{(i)})+\mathbf{v}_t\Big)\Big\}$$
$$< 4\Big(\mathbf{G}(\mathbf{F}\mathbf{x}_{t-1}+\mathbf{d}_t-\mathbf{x}_{t-1}^{(i)})+\mathbf{v}_t\Big)^T\mathbf{G}(\mathbf{x}_{t-1}-\mathbf{x}_{t-1}^{(i)}). \qquad (5.43)$$

Taking expectations over the $\mathbf{d}_t$ and $\mathbf{v}_t$ in both sides of (5.43) yields:

$$\eta\Big\{(\mathbf{F}\mathbf{x}_{t-1}-\mathbf{x}_{t-1}^{(i)})^T\mathbf{H}^2(\mathbf{F}\mathbf{x}_{t-1}-\mathbf{x}_{t-1}^{(i)})+\mathrm{tr}\Big((\Sigma_\mathbf{d}\mathbf{H}+\Sigma_\mathbf{v})\mathbf{H}\Big)$$
$$< (\mathbf{F}\mathbf{x}_{t-1}+\mathbf{d}_t-\mathbf{x}_{t-1}^{(i)})^T\mathbf{H}(\mathbf{x}_{t-1}-\mathbf{x}_{t-1}^{(i)}) \qquad (5.44)$$

where tr(·) denotes the trace of the matrix, and $\mathbf{H}=\mathbf{G}^T\mathbf{G}$. The criterion of (5.44) can be further rewritten as

$$\begin{aligned}&(\mathbf{x}_{t-1}-\mathbf{x}_{t-1}^{(i)})^T(\eta\mathbf{H}-\mathbf{I})\mathbf{H}(\mathbf{x}_{t-1}-\mathbf{x}_{t-1}^{(i)})\\ &+\eta\mathrm{tr}\Big((\Sigma_\mathbf{d}\mathbf{H}+\Sigma_\mathbf{v})\mathbf{H}\Big)\\ &+\eta(\mathbf{F}\mathbf{x}_{t-1}-\mathbf{x}_{t-1})^T\mathbf{H}^2(\mathbf{F}\mathbf{x}_{t-1}-\mathbf{x}_{t-1})\\ &+(\mathbf{F}\mathbf{x}_{t-1}-\mathbf{x}_{t-1})^T(\eta\mathbf{H}-\mathbf{I})\mathbf{H}(\mathbf{x}_{t-1}-\mathbf{x}_{t-1}^{(i)})\\ &+\eta(\mathbf{x}_{t-1}-\mathbf{x}_{t-1}^{(i)})^T\mathbf{H}^2(\mathbf{F}\mathbf{x}_{t-1}-\mathbf{x}_{t-1})<0.\end{aligned} \qquad (5.45)$$

Case 2: When the measurement $\mathbf{y}=\mathbf{y}_{t-1}$ is used. Similarly, we can derive the criterion for which the MOVE-step is considered to be beneficial:

$$(\mathbf{x}_{t-1}-\mathbf{x}_{t-1}^{(i)})^T(\eta\mathbf{H}-\mathbf{I})\mathbf{H}(\mathbf{x}_{t-1}-\mathbf{x}_{t-1}^{(i)})+\eta\mathrm{tr}(\Sigma_\mathbf{v}\mathbf{H})<0. \qquad (5.46)$$

Comparing (5.45) and (5.46), it can be seen that the error performance of the MOVE-step differs considerably from using current and previous measurement. While using the current observation, the error criterion will depend on the step-size, the bias between the true state and the particle state, and the covariance matrices of both dynamical and measurement noise. In the contrast, while using the previous observation, the error criterion will only depend on the step-size and the variance of the measurement noise; namely, it is independent of the bias between the true state and the particle state. As we expect, in some cases the error performance using *current* measurement is advantageous, but in certain cases it might also degrade the performance.

The following simple example (Mark Morelande, personal communication) tries to illustrate the difference between the above two cases. We consider an example where a target is moving in a one-dimensional axis with its noisy position as measurement. Let $\mathbf{x}_t$ denote a two-dimensional (position and velocity) state vector and y_t denote the one-dimensional measurement, and

$$\begin{aligned}\mathbf{F} &= \begin{bmatrix}1 & T\\ 0 & 1\end{bmatrix},\ \Sigma_\mathbf{d}=0.01\times\begin{bmatrix}T^3/3 & T^2/2\\ T^2/2 & T\end{bmatrix},\\ \mathbf{G} &= \begin{bmatrix}1 & 0\end{bmatrix},\ \Sigma_\mathbf{v}=0.01,\end{aligned}$$

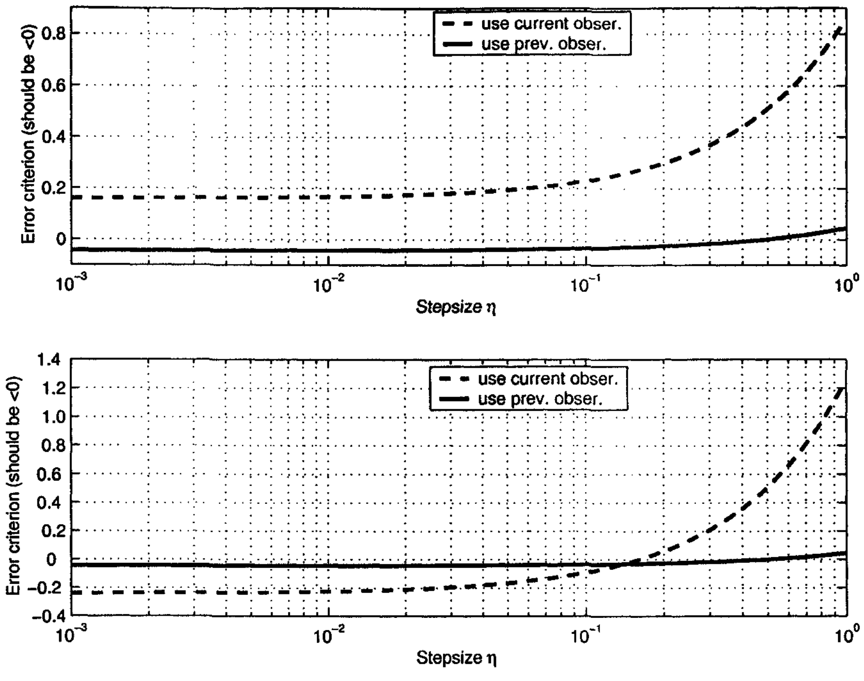

Figure 5.20: *Top panel*: the error criteria of using previous (solid) and current (dashed) observations for a *positive bias* in the particle state. *Bottom panel*: the error criteria of using previous (solid) and current (dashed) observations for a *negative bias* in the particle state.

where $T = 1$ is the sampling-time interval, and the true state is $\mathbf{x}_{t-1} = [5.0, 1.0]^T$, and

$$\mathbf{H} = \mathbf{G}^T\mathbf{G} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}.$$

Now let us plot the error criterion against step-size parameter η for the above two cases. Note that the gradient proposal is successful only when the error criterion is *negative*.

To distinguish the different behavior, we consider two situations,[13] one for *positive bias*, and the other for *negative bias* existing in the simulated particle state $\mathbf{x}_{t-1}^{(i)}$. As illustrated in the top panel of Figure 5.20, when $\mathbf{x}_{t-1}^{(i)} = [5.2, 1.06]^T$ and the bias $(\mathbf{x}_{t-1}^{(i)} - \mathbf{x}_{t-1})$ is positive, the performance of the MOVE-step with previous observation is improved (i.e., the error criterion is negative) and almost always successful for various choices of η; however, the performance while using current observation in this case is unsuccessful no matter what η value is chosen. On the other hand, when $\mathbf{x}_{t-1}^{(i)} = [4.8, 0.94]^T$ and the bias $(\mathbf{x}_{t-1}^{(i)} - \mathbf{x}_{t-1})$ is negative, the performance of the MOVE-step using the current observation is better (for the small step-size η) and will degrade as steps-size parameter increases;[14]

[13]Here we have excluded the discussion on the situations where some components have positive biases and the others have negative biases; but the general conclusion is similar.

[14]This also illustrates the reason why in practice we always let the step-size parameter be a small-valued scalar.

In contrast, the performance while using the previous observation, as expected, is independent on the bias, and therefore almost always successful for small step-size. This is illustrated in the bottom panel of Figure 5.20.

Therefore, we can conclude that when the MOVE-step is conducted *before* the sampling-step, we should use *previous* observation, and when the MOVE-step is conducted *after* the sampling-step, we should use *current* observation. Namely, in calculation of the gradient, we shall use $\mathbf{y}_{t-1}$ for simulated particles $\mathbf{x}_{t-1}^{(i)}$, and use $\mathbf{y}_t$ for $\mathbf{x}_t^{(i)}$.

Generalization to Nonlinear Estimation Problem. For the nonlinear and/or non-Gaussian state-space model, the above results can not be applied exactly, but the discussion still approximately holds under the assumption that the tracking error is relatively small by replacing $\mathbf{F}$ and $\mathbf{G}$ by the corresponding Jacobian matrices computed for the target state at time $t-1$.

For illustration purpose, let us consider a one-dimensional nonlinear state estimation problem (Gordon et al., 1993):

$$x_t = 0.5x_{t-1} + \frac{25x_{t-1}}{1+x_{t-1}^2} + 8\cos(1.2(t-1)) + d_t, \tag{5.47}$$

$$y_t = \frac{x_t^2}{20} + v_t, \tag{5.48}$$

where $x_0 = 0.1$, d_t and v_t are zero-mean Gaussian white noise processes with variances 10 and 1, respectively. The initial condition is set up as $\{x_0^{(i)}\} \sim \mathcal{N}(0.1, 1)$.

Given (5.48), we can calculate the Jacobian to approximate the measurement matrix $\mathbf{G}$ of (5.37), as well as $\mathbf{H} = \mathbf{G}^T\mathbf{G}$ and the gradient $\xi(y_t)$:

$$\begin{aligned}\hat{\mathbf{G}}_t &= x_t/10,\\ \hat{\mathbf{H}}_t &= \hat{\mathbf{G}}_t^T\hat{\mathbf{G}}_t = x_t^2/100,\\ \xi(y_t) &= -\frac{1}{10}x_t(y_t - x_t^2/20).\end{aligned}$$

Substituting $\hat{\mathbf{G}}_t$ and $\hat{\mathbf{H}}_t$ into (5.46) yields the corresponding *approximate* criterion for the nonlinear state-space model:

$$(\mathbf{x}_t - \mathbf{x}_t^{(i)})^T(\eta\hat{\mathbf{H}}_t - \mathbf{I})\hat{\mathbf{H}}_t(\mathbf{x}_t - \mathbf{x}_t^{(i)}) + \eta\,\mathrm{tr}(\Sigma_{\mathbf{v}}\hat{\mathbf{H}}_t) < 0. \tag{5.49}$$

For the above one-dimensional problem, we can simplify (5.49) as

$$(x_t - x_t^{(i)})^2(\eta 10^{-2}x_t^2 - 1)x_t^2 + \eta x_t^2 < 0. \tag{5.50}$$

If $\eta x_t^2/100 \ll 1$, then the condition (5.50) becomes that given $x_t \neq 0$,

$$(x_t - x_t^{(i)})^2 > \eta, \tag{5.51}$$

If $\eta x_t^2/100 \ll 1$, then the condition (5.50) becomes that given $x_t \neq 0$,

$$(x_t - x_t^{(i)})^2 > \eta,$$

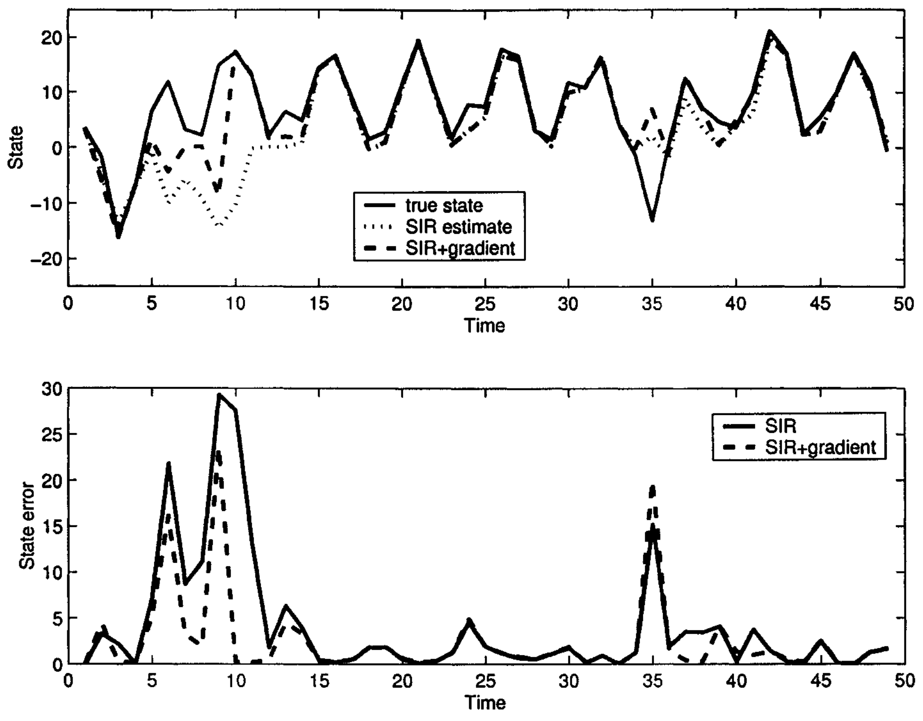

Figure 5.21: *Top panel*: the true state (solid) and the particle filter estimates (using 100 particles) in the one-dimensional nonlinear state estimation problem. *Bottom panel*: the comparison of state estimate error (absolute value) between without and with the gradient MOVE-step (under the same conditions).

or

$$|x_t - x_t^{(i)}| > \sqrt{\eta}. \tag{5.52}$$

Equation (5.52) provides a *stepsize-dependent* bound, which, in turn, suggests that using a small step-size provides a higher-chance of success in the gradient MOVE-step. Our Monte Carlo simulation result (see Figure 5.21 for an illustration) also confirmed the empirical analysis.

Chapter 6

Monte Carlo Sampling-Based ALOPEX Algorithms

"Learning without thought is useless, thought without learning is dangerous."
— Confucius

6.1 State-Space Model for Parameter Estimation

In Chapter 5, we have seen that the state-space model serves a principled tool for sequential state estimation. In this chapter, we will recast the parameter estimation problem into the same framework in order to accommodate sequential Monte Carlo methods. For the purpose of exposition simplicity, we focus on the parameter estimation problem in a supervised learning framework; but the discussion also generalizes to unsupervised learning or generic optimization problem.

Let us formulate a generic parameter estimation problem in the form of a state-space model:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \boldsymbol{\nu}_t, \qquad (6.1a)$$

$$\mathbf{y}_t = f(\boldsymbol{\theta}_t, \mathbf{x}_t) + \mathbf{v}_t, \qquad (6.1b)$$

where the nonlinear measurement equation, parameterized by $\boldsymbol{\theta}$, determines the mapping $f : X \mapsto Y$, given a number of inputs $\mathbf{x}_t$ and outputs $\mathbf{y}_t$. The additive terms $\boldsymbol{\nu}_t$ and $\mathbf{v}_t$ are dynamical noise and measurement noise, respectively. In our case, f can be a neural network, or a certain parameterized model.

In the sequential Monte Carlo estimation framework, $\boldsymbol{\theta}_t$ is estimated via particle filtering that follows a recursive Bayesian estimation procedure (see Chapter 5). Simply put, a particle filter uses a number of random samples called "particles", sampled directly from the state space, to represent the posterior, and updates the posterior by involving the new observations; the "particle system" is properly *located, weighted*, and *propagated* recursively according to Bayes' rule.

Among many variations, one of the most popular particle filters is the SIR filter (Gordon et al., 1993); we will focus on the SIR filter in the rest of the chapter. As discussed in Chapter 5, a notorious phenomenon of the SIR filter is the *weight degeneracy* problem. To monitor the number of the effective samples (denoted as N_{eff}), one empirical measure of sample efficiency is the variance of the importance weights. Besides that, we also suggest (Chen, 2003a) another efficiency measure based on the Kullback-Leibler (KL) divergence between the proposal and target densities, denoted by $D(q\|p)$. Given N_p samples drawn from the proposal q, the KL divergence $D(q\|p)$ is approximated by

$$\begin{aligned} D(q\|p) &= \mathbb{E}_q\left[\log\frac{q(\boldsymbol{\theta})}{p(\boldsymbol{\theta})}\right] \approx \frac{1}{N_p}\sum_{i=1}^{N_p}\log\frac{q(\boldsymbol{\theta}^{(i)})}{p(\boldsymbol{\theta}^{(i)})} \\ &= -\frac{1}{N_p}\sum_{i=1}^{N_p}\log(W(\boldsymbol{\theta}^{(i)})), \end{aligned} \tag{6.2}$$

where $\boldsymbol{\theta}^{(i)} \sim q(\boldsymbol{\theta})$. When $q = p$ and $W(\boldsymbol{\theta}^{(i)}) = 1$ for all i, $D(q\|p) = 0$. Since $D(q\|p) \geq 0$, $-\log(W(\boldsymbol{\theta}^{(i)}))$ should be non-negative. In practice, we instead calculate the logarithm of the normalized importance weights $\hat{N}_{\mathrm{KL}} = -\frac{1}{N_p}\sum_{i=1}^{N_p}\log(\tilde{W}(\boldsymbol{\theta}^{(i)}))$, which achieves the minimum value $\hat{N}_{\mathrm{KL}}^{\min} = \log(N_p)$ when all $\tilde{W}(\boldsymbol{\theta}^{(i)}) = 1/N_p$. Our previous studies have confirmed that $\hat{N}_{\mathrm{KL}}$ is a good measure that is also consistent with $\hat{N}_{eff}$: when $\hat{N}_{\mathrm{KL}}$ is small, $\hat{N}_{eff}$ is usually large; and vice versa.

6.2 Sampling-Based ALOPEX Algorithms

In what follows, we propose two novel sampling-based ALOPEX algorithms by combining ALOPEX-B and a sequential Monte Carlo estimation technique. The algorithms are recursive and fall under the Bayesian estimation framework. Like the ALOPEX-like algorithms, they are gradient-free, and they are suitable for either on-line (sequential) or off-line (batch) learning.

Note that equation (6.1a) uses a random-walk model. In order to avoid the "blind" random walk behavior, we employ a "relaxation" model (Mackay, 1998) in place of (6.1a):

$$\boldsymbol{\theta}_{t+1}^{(i)} = \boldsymbol{\mu}_t + \alpha(\boldsymbol{\theta}_t^{(i)} - \boldsymbol{\mu}_t) + \sqrt{1-\alpha^2}\sigma\boldsymbol{\nu}_t \tag{6.3}$$

where $\boldsymbol{\mu}_t = \sum_{i=1}^{N_p}\tilde{W}_t^{(i)}\boldsymbol{\theta}_t^{(i)}$; the noise vector $\boldsymbol{\nu}_t$ is standard Gaussian-distributed: $\boldsymbol{\nu}_t \sim \mathcal{N}(0, \mathbf{I})$, and σ is the standard deviation controlling the degree of variation in $\boldsymbol{\theta}$, which often requires some prior knowledge of the problem. The relaxing parameter $\alpha \in [-1, 1]$ controls the degree of over-relaxation (or under-relaxation):

- when $\alpha = -1$, (6.3) reduces to an extreme over-relaxation $\boldsymbol{\theta}_{t+1}^{(i)} = 2\boldsymbol{\mu}_t - \boldsymbol{\theta}_t^{(i)}$;
- when $\alpha = 0$, (6.3) reduces to a random walk $\boldsymbol{\theta}_{t+1}^{(i)} = \boldsymbol{\mu}_t + \sigma\boldsymbol{\nu}_t$;
- when $\alpha = 1$, (6.3) reduces to a stationary point $\boldsymbol{\theta}_{t+1}^{(i)} = \boldsymbol{\theta}_t^{(i)}$.

In summary, our first sampling-based ALOPEX algorithm (termed Algorithm-1 hereafter) proceeds as follows:

---

1. For $i = 1, \cdots, N_p$, initialize $\boldsymbol{\theta}_0^{(i)} \sim p(\boldsymbol{\theta}_0)$, $W_0^{(i)} = 1/N_p$.
2. Predict $\boldsymbol{\theta}_t^{(i)}$ from (6.3).
3. Update the samples $\boldsymbol{\theta}_t^{(i)}$ via the modified ALOPEX-B algorithm (2.23) through (2.25).
4. Evaluate importance weights $W_t^{(i)} = W_{t-1}^{(i)} p(\mathrm{y}_t|\boldsymbol{\theta}_t^{(i)}, \mathrm{x}_t)$ and $\tilde{W}_t^{(i)} = \frac{W_t^{(i)}}{\sum_{j=1}^{N_p} W_t^{(j)}}$.
5. Calculate $\hat{N}_{eff}$ and $\hat{N}_{\mathrm{KL}}$, if $\hat{N}_{eff} > 0.8N_p$ or $\hat{N}_{\mathrm{KL}} > 3\log(N_p)$, go to step 7; otherwise go to step 6.
6. (Systematic) Resampling: generate a new particle set $\{\boldsymbol{\theta}_t^{(j)}\}$ and reset the weights $\tilde{W}_t^{(j)} = 1/N_p$.
7. Repeat steps 2 through 5.

Note that when $N_p = 1$, Algorithm-1 reduces to a generalized form of ALOPEX-B, which involves an additional randomness through (6.3). In addition, there is no reason why we cannot use specific $\alpha^{(i)}$ for different $\boldsymbol{\theta}^{(i)}$; α can also be time-varying, but we have not investigated these issues here. We fixed α for each specific problem in the experiments reported later, but the optimal α often varies from one problem to another.

It is of interest to compare our algorithm with other sampling-based optimization algorithms, e.g., Fisher-scoring (Briegel and Tresp, 1999) and HySIR (deFreitas et al., 2000; deFreitas, 1999) for training neural networks. The complexity of our algorithm ($\mathcal{O}(N_pN)$) is much smaller than these two algorithms ($\mathcal{O}(N_pN^2)$) simply because of avoiding the calculation of the Jacobian matrix. Our algorithm is also much simpler than another sampling-based gradient-free estimation technique: the unscented particle filter (UPF), which is typically of $\mathcal{O}(N_pN^3)$ complexity (van der Merwe et al., 2000; Wan and van der Merwe, 2001).

We next propose another sampling-based ALOPEX algorithm (hereafter termed as Algorithm-2) that is motivated by the hybrid Monte Carlo (HMC) method (Duane et al., 1987; Mackay, 1998). The idea of HMC is to augment the state space $\boldsymbol{\theta}$ with a momentum variable $\boldsymbol{\rho}$. The energy-conserving Hamiltonian dynamics is defined as:

$$\mathcal{H}(\boldsymbol{\theta}, \boldsymbol{\rho}) = E(\boldsymbol{\theta}) + \mathcal{K}(\boldsymbol{\rho}), \tag{6.4}$$

where $E(\boldsymbol{\theta})$ is the *potential energy* function, whereas $\mathcal{K}(\boldsymbol{\rho}) = \boldsymbol{\rho}^T\boldsymbol{\rho}/2$ is the *kinetic energy*. The samples are drawn from the joint distribution

$$\begin{aligned} p_{\mathcal{H}}(\boldsymbol{\theta}, \boldsymbol{\rho}) &= \frac{1}{Z}\exp(-\mathcal{H}(\boldsymbol{\theta}, \boldsymbol{\rho})) \\ &= \frac{1}{Z}\exp(-E(\boldsymbol{\theta}))\exp(-\mathcal{K}(\boldsymbol{\rho})), \end{aligned} \tag{6.5}$$

where Z is a normalizing constant. Note that the term $\exp(-E(\boldsymbol{\theta}))$ is essentially the likelihood up to a normalizing factor. The momentum dynamics can be approximated by the ensuing difference equations

$$\boldsymbol{\rho}_t = \nabla\boldsymbol{\theta}_t \approx \Delta\boldsymbol{\theta}_t \tag{6.6a}$$

$$\nabla\boldsymbol{\rho}_t = -\frac{\partial E(\boldsymbol{\theta}_t)}{\partial\boldsymbol{\theta}_t} \approx -\frac{\Delta E(\boldsymbol{\theta}_t)}{\Delta\boldsymbol{\theta}_t}, \tag{6.6b}$$

where, obviously, all of the terms are intermediate results obtained from the ALOPEX-like algorithm without additional computing overhead. By doing so, the posterior of $\boldsymbol{\theta}_{t+1}$ is proportional to $p(\boldsymbol{\theta}_{t+1}|\boldsymbol{\theta}_t)p_{\mathcal{H}}(\boldsymbol{\theta}_t, \boldsymbol{\rho}_t) = p(\boldsymbol{\theta}_{t+1}|\boldsymbol{\theta}_t)\exp(-\frac{1}{2}\boldsymbol{\rho}_t^T\boldsymbol{\rho}_t)p(\mathbf{y}_t|\boldsymbol{\theta}_t)$. Equivalently, while keeping the importance weights proportional to the likelihood, (6.3) is changed to

$$\begin{aligned} \tilde{\boldsymbol{\theta}}_{t+1}^{(i)} &= \boldsymbol{\theta}_{t+1}^{(i)} + \beta\Delta\boldsymbol{\theta}_t^{(i)} \\ &= \boldsymbol{\mu}_t + \alpha(\boldsymbol{\theta}_t^{(i)} - \boldsymbol{\mu}_t) + \beta\Delta\boldsymbol{\theta}_t^{(i)} + \sqrt{1-\alpha^2}\sigma\boldsymbol{\nu}_t, \end{aligned} \tag{6.7}$$

where β is a momentum coefficient. Equation (6.7) essentially describes a second-order auto-regressive (AR) model compared to the first-order models (6.1a) and (6.3); it also implies that $p(\tilde{\boldsymbol{\theta}}_{t+1}|\boldsymbol{\theta}_t, \Delta\boldsymbol{\theta}_t) \propto p(\boldsymbol{\theta}_{t+1}|\boldsymbol{\theta}_t)\exp(-\Delta\boldsymbol{\theta}_t^T\Delta\boldsymbol{\theta}_t)$. Algorithm-2 differs from Algorithm-1 only in the second step where (6.3) is replaced by (6.7).

Remark: It is noteworthy to comment that although the above two sampling-based ALOPEX algorithms are discussed in the supervised learning framework, they can readily be used for unsupervised learning, in which the likelihood function is related to the specified objective function.

6.3 Training Neural Networks

6.3.1 Network Architectures

Feedforward network: Typical examples of this type include the multilayer perceptron (MLP) and radial basis function (RBF) network.

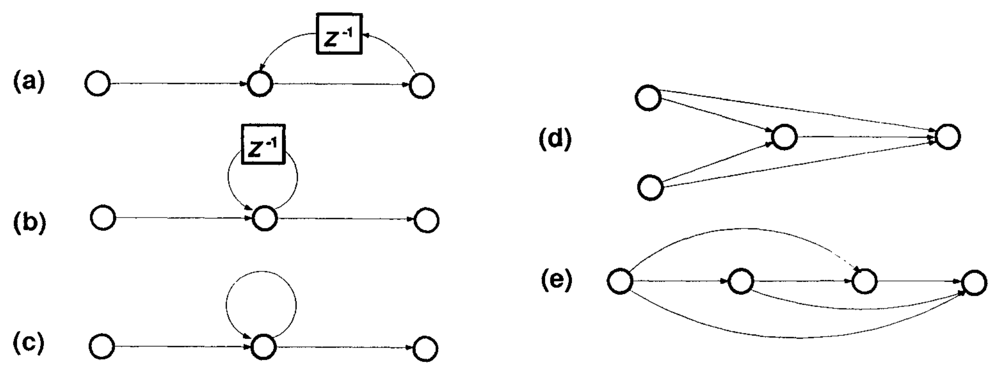

Figure 6.1: A schematic diagram of different network architectures. (a) a time-delay feedback from output to hidden nodes; (b) a time-delay feedback from hidden to hidden nodes; (c) a self-feedback in hidden node with no time-delay; (d) a densely-connected architecture; (e) a feedforward multilayer architecture with short-cut connections.

Recurrent network: There exist different kinds of recurrent networks depending on the structure of feedback connections, either locally-recurrent or fully-recurrent:

- *Elman network* (Elman, 1990): with time-delay feedback from hidden nodes to hidden nodes. The time-delay hidden outputs are stored in the so-called *context units*, which can be viewed as an augmented set of input nodes.
- *Jordan network* (Jordan, 1989): with time-delay feedback from output nodes to hidden nodes. The time-delay network outputs are likewise stored in the context units. Elman and Jordan networks are the most popular locally-recurrent MLP (RMLP) models.
- *Self-feedback network*: with feedback from hidden nodes to hidden nodes, but no time-delay. An example of this type is the echo state network (Jaeger, 2001; Jaeger and Haas, 2004).
- *Hopfield network* (Hopfield, 1982): with fully-connected recurrent feedbacks except the self-feedback.

Hierarchical network: Examples of this type often include the fully connected network, or the network with cross-layer or short-cut connections, or the network with specific modular or template structure, such as the convolutional neural network (LeCun et al., 1998).

A schematic diagram of some exemplar network structures is shown in Figure 6.1. Note that the gradient-based optimization procedure for hierarchical or recurrent networks, such as back-propagation through time (BPTT) or real-time recurrent learning (RTRL), is relatively sophisticated and often requires special treatment for different network architectures.

6.3.2 Benchmark Problems

For comparison, the following five benchmark problems: `XOR`, `parity`, `encoder-decoder`, `encoder`, and `decoder`, taken from (Unnikrishnan and Venugopal, 1994; Bia, 2001), are used for MLP networks training; and the other three benchmark problems: `sequential XOR-`D, `delay-`D, `sequential parity`, taken from (Bia, 2001), are used for (Elman-type) RMLP networks training. The main purpose of these benchmark experiments is to compare the convergence speed between the ALOPEX-B algorithm (Bia, 2001) and our developed sampling-based ALOPEX algorithms. We only compare our algorithms with the ALOPEX-B algorithm simply because we also found that it converges faster than its counterpart (Unnikrishnan and Venugopal, 1994). In these tasks, no testing (generalization) error is measured; the network architecture are chosen by heuristics.

All of the benchmark tasks are performed in an off-line learning fashion. For the ALOPEX-B algorithm, no exhaustive effort was made to find the optimal parameters η and λ; we used the recommended values as (Bia, 2001) in our experiments. $\{\theta_0^{(i)}\}$ are uniformly distributed within the region $[-1.5, 1.5]$.[1] Once $\theta_j(0)$ is generated, an initial Gaussian prior $\mathcal{N}(\theta_j(0), 0.5)$ is used for generating the samples $\{\theta_j^{(i)}\}$. The error measure is the mean-squared error (MSE):

$$E = \frac{1}{2\ell}\sum_{t=1}^{\ell} \|\mathbf{y}_t - \hat{\mathbf{y}}_t\|^2. \tag{6.8}$$

For the sampling-based ALOPEX algorithms, we only monitor the minimum MSE among all $\{\theta^{(i)}\}$; the one achieving minimum MSE is regarded as the *maximum a posteriori* (MAP) estimate. The typical parameter setup is as follows, although the optimal choices are problem-dependent: (i) relaxing parameter $\alpha \in [-0.5, 0.2]$; (ii) step-size parameter $\gamma = -0.01$; (iii) learning rate $\eta \in [0.05, 0.1]$; (iv) momentum coefficient $\beta = \alpha/10$; (v) forgetting factor $\lambda \in [0.1, 0.5]$; (vi) standard deviation of noise $\sigma \in [0.01, 0.02]$. The number of particles for these tasks are set as $N_p = 10$, unless stated otherwise.

The optimization results are summarized in Table 6.1. As seen in the table, the proposed sampling-based ALOPEX algorithms outperform the (modified) ALOPEX-B algorithm for all of the benchmark problems in terms of convergence speed. As expected, the Algorithm-2 performs slightly better than the Algorithm-1. These results indicate very clearly a performance-complexity tradeoff for the algorithmic design.

6.3.3 Pattern Recognition

Digit Recognition. First, we consider a toy pattern recognition problem for 10 binary (black and white) digit classification. The synthetic data are shown in Figure 6.2, where each digit is represented as a 35-bit 7×5 array. An MLP network with architecture

[1]By using this region, we implicitly assume that the input data are properly normalized within the region $[0, 1]$; or equivalently, if the input data is within the region $[-1, 1]$, then $\theta_0 \in \mathcal{U}(0, 1.5)$.

Table 6.1: Experimental results of the benchmark problems. The numbers are the averaged (based on 20 Monte Carlo simulations) and minimum epochs for achieving a training error MSE = 0.001; no testing error is calculated in these problems.

Problem	network size	ALOPEX-B ave.	ALOPEX-B best	Algorithm-1 ave.	Algorithm-1 best	Algorithm-2 ave.	Algorithm-2 best
`XOR`	MLP2-2-1	7568	1333	1008	845	879	633
`encoder-decoder`	MLP4-2-4	4080	2657	2138	1749	1906	1734
`parity`	MLP4-4-1	65372	25519	5589	5240	5467	4546
`encoder`	MLP8-3-3	2594	2139	1402	1059	1374	926
`decoder`	MLP3-3-8	24220	10126	4951	4428	4848	4430
`Delay-1`	RMLP1-2-1	1210	1108	451	412	382	334
`Delay-2`	RMLP1-3-1	5031	4291	1922	1581	1309	1101
`Delay-3`	RMLP1-4-1	596	356	239	201	215	149
`sequential XOR-0`	RMLP1-2-1	6985	6235	3531	2946	3013	2511
`sequential XOR-1`	RMLP1-4-1	7032	6323	3155	2843	2983	2325
`sequential XOR-2`	RMLP1-6-1	11323	10832	7932	7059	7251	6501
`sequential parity`	RMLP1-3-1	13138	11098	7832	7031	7238	6235

net35-5-4 is used for training; no model selection procedure (for hidden units) is undertaken here. With the common MSE metric, a 100% classification rate is achieved for the 10 training samples after the MSE value approaches 0.01. The optimization curves of the ALOPEX-B, modified ALOPEX-B, and sampling-based ALOPEX (Algorithm-2) are shown in Figure 6.2. In order to test the generalization and robustness, we add different amount of Gaussian noise into the the training patterns and reexamine the correct classification rate; it is found that the network's performance is noise-tolerant within 5% white noise, beyond that misclassification occurs.

Next, we consider a more challenging gray-level digit recognition problem, where the real-life data were obtained from the USPS database and further preprocessed appropriately. Each sample represented as an 8×8 pixel image (Figure 6.3); for each class, 400 of them are used for training set, 300 of them for validation set, and the rest 400 of them for testing set. A total number of $10 \times 1100 = 11000$ samples are used in the experiments. We use the sampling-based ALOPEX algorithm to train an MLP net64-8-4. As anticipated, the convergence of learning is slow (Figure 6.4) due to a large amount of data. Upon completion of training (for approaching MSE=0.02), the network achieves 4.83% and 15.1% misclassification rates for the training and testing data, respectively.

Two-Spiral Classification. Finally, we consider a two-spiral classification problem, which is known as a hard optimization problem due to its complex classification boundary, as illustrated in Figure 6.5. The data consist of two intertwined spirals with a total 194 training samples. Our early investigations of using a simple MLP network trained by conventional back-propagation or ALOPEX algorithm confirmed previous observations of

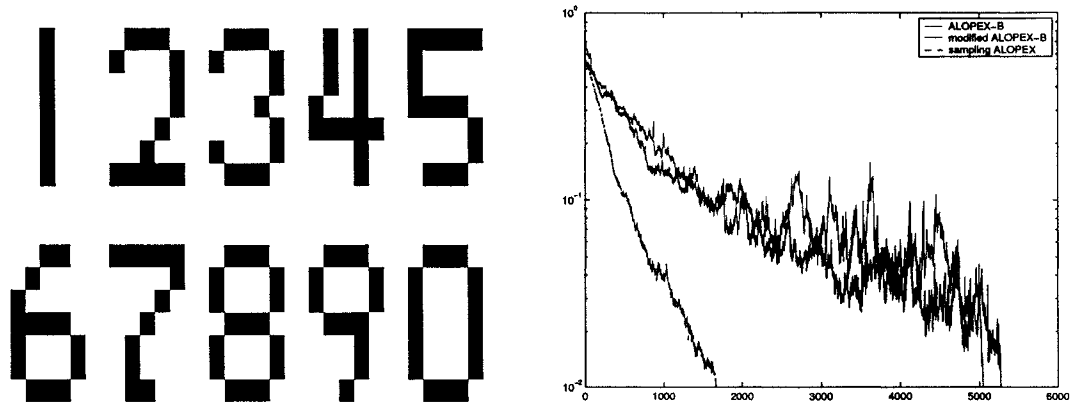

Figure 6.2: *Left:* binary digits. *Right:* optimization curves of three algorithms starting with the same initial conditions.

other researchers (Lang and Witbrock, 1989; Fahlman and Lebiere, 1989).

Here we use a fully-connected network with an architecture net2-5-5-5-1 (see Figure 6.5), which has cross-layer connections. This kind of network was early proposed by (Lang and Witbrock, 1989) trained with back-propagation; but it can be imagined that the calculation of the back-propagating error and derivatives is rather sophisticated. For the optimization purpose, we use the sampling-based ALOPEX algorithm (with $N_p = 30$ particles) in place of the conventional back-propagation algorithm. Different kinds of objective functions are investigated. The first is a regularized MSE metric with a "weight-decay" term:

$$E(\boldsymbol{\theta}) = \frac{1}{2\ell}\sum_{i=1}^{\ell} \|y_i - \hat{y}_i\|^2 + \lambda\|\boldsymbol{\theta}\|^2, \tag{6.9}$$

with a small-valued regularization parameter λ; and the second takes a form of the cross-entropy (Hinton, 1989; Bishop, 1995):

$$E(\boldsymbol{\theta}) = \sum_{i=1}^{\ell} -y_i \log \hat{y}_i - (1-y_i)\log(1-\hat{y}_i), \tag{6.10}$$

or its modified form:

$$E(\boldsymbol{\theta}) = \sum_{i=1}^{\ell} y_i \log \frac{y_i}{\hat{y}_i} + (1-y_i)\log\frac{1-y_i}{1-\hat{y}_i}. \tag{6.11}$$

The difference between (6.10) and (6.11) is that the minimum of (6.11) is zero, whereas (6.10) has a nonzero minimum. See (Bishop, 1995) for discussions on the properties of these objective functions in pattern classification.

The misclassification rates obtained from using (6.11) are 2.4% and 1.8% for training and testing data, respectively. In contrast, an MLP net2-25-1 trained by the same algorithm using (6.9) achieves 9.3% misclassification rate.

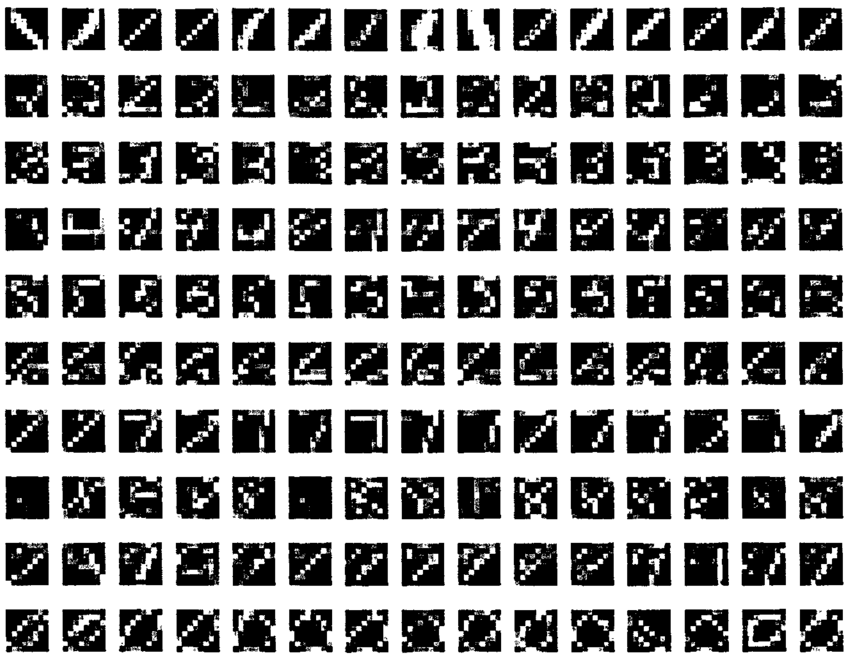

Figure 6.3: Selected training digit samples used in the experiments.

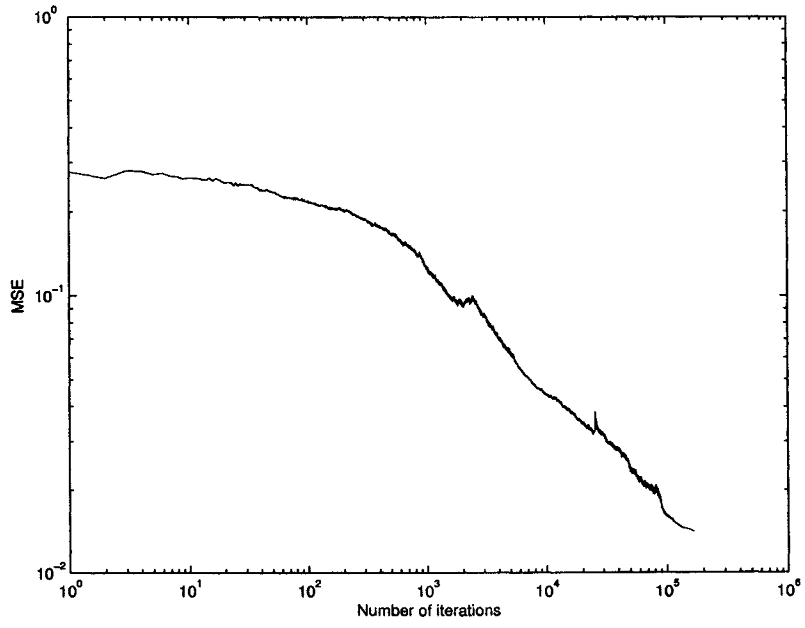

Figure 6.4: Learning curve for gray-level digit recognition.

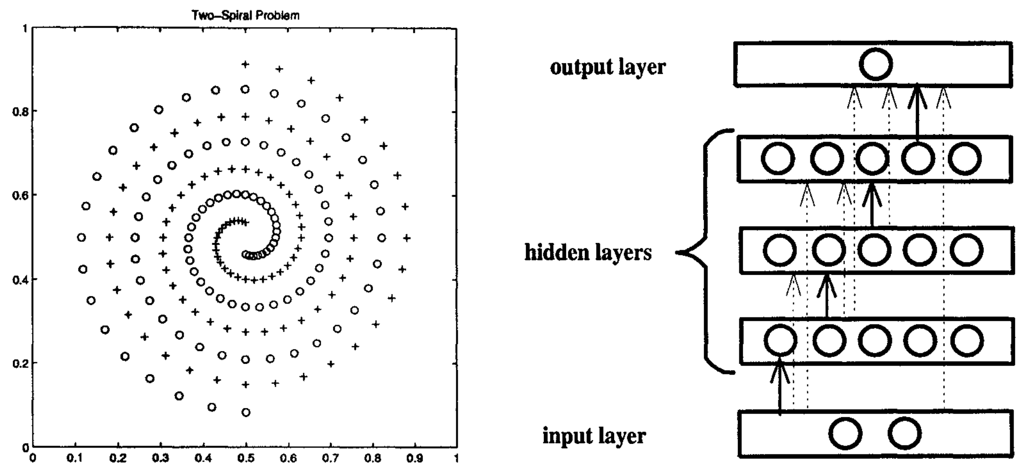

Figure 6.5: *Left:* two-spiral patterns. *Right:* network architecture.

6.3.4 Option Prices Prediction

In the past decade, connectionist models such as MLP and RBF networks, have been used successfully in financial time series forecasting and analysis (see e.g., Hutchinson, 1994; Hutchinson et al., 1994). The financial data (e.g., stock exchange, interest rate, foreign exchange, etc.) are known as nonlinear and non-stationary, thus providing a good testbed for neural network modeling and prediction.

The real-life financial data used here consist of five pairs of call and put option contracts on the FTSE100 index (daily close prices from February 1994 to December 1994).[2] The accessible data include ***strike prices, call option prices*** and ***put option prices.*** [3]

In the literature, the classic *Black-Scholes* formula was proposed for the call option price (Black and Scholes, 1973):

$$C = f(S, X, T) \tag{6.12}$$

where C is the call option price, T represents the maturity time, and S and X represent the stock (asset) price and the strike (exercise) price of the option, respectively. The form of parametric function f often depends on specific underlying asset and the market. In reality, the call option price data are inherently generated from complex and stochastic dynamics, which rely on many factors that introduce various kinds of noise to the data.

[2]The experimental data and the HySIR code are downloaded from the webpage of Dr. Nando de Freitas: http://www.ubc.ca/~nando/.

[3]Knowledge background (deFreitas, 1999): A ***derivative*** is a financial instrument whose value replies on some basic cash product. An ***option*** is a particular type of derivative that gives the holder the right to do something. For example, a *call option* allows the holder to buy a cash product at a specified date in the future. The price at which the option is exercised is known as the ***strike price***, while the date at which the option lapses is referred to as the *maturity time*. A *put option* allows the holder to sell the underlying cash product.

Table 6.2: Comparative experimental results of option pricing prediction. The values are averaged one-step prediction MSE based on 20 Monte Carlo runs with different initial random seeds.

Strike price	2925	3025	3125	3225	3325
ALOPEX-B	0.2891	0.2231	0.1921	0.1837	0.1071
Algorithm-1	0.0403	0.0404	0.0383	0.0352	0.0242
Algorithm-2	0.0399	0.0395	0.0366	0.0310	0.0231
EKF	0.0408	0.0396	0.0401	0.0307	0.0215
HySIR	0.0389	0.0379	0.0369	0.0293	0.0194

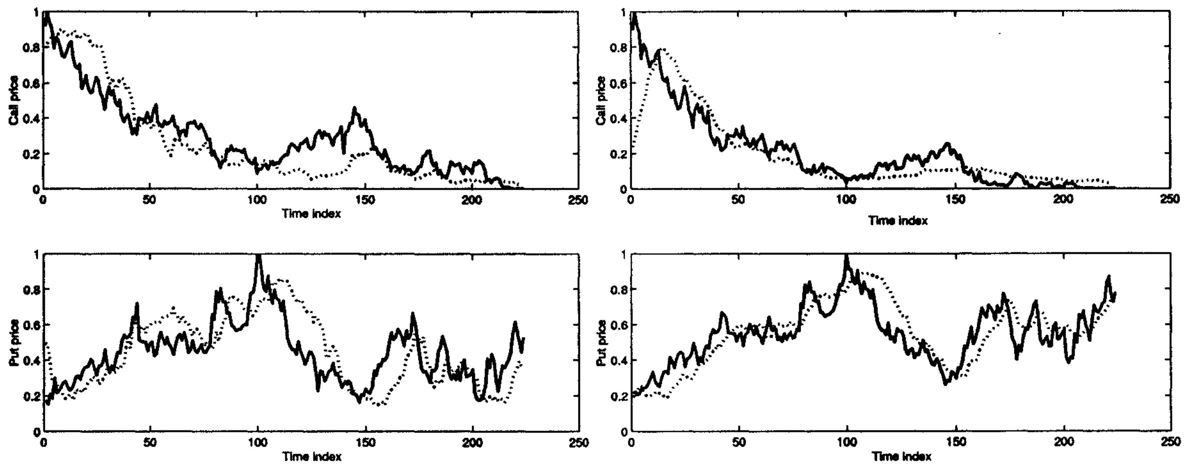

Figure 6.6: Call and put option prices prediction curves (left: strike price data 3125; right: strick price data 3325) produced by Algorithm-2 in one Monte Carlo trial. Solid line: true value; dotted line: predicted value.

Due to this reason, the *Black-Scholes* parametric model often suffers from the violation of the underlying assumptions, such as *lognormality* or *sample-path continuity*; it is also not robust to the colored (correlated) noise. The nonstatonarity of the financial data often necessitates sequential tracking, which requires that the model be updated correspondingly on-line. This is in contrast to the common approach that uses a fixed-weight neural network for the out-of-the-sample data, assuming a suboptimal network being trained off-line given sufficient training data set. Our approach here does not impose such a restriction, although a pretrained network (including model selection) with off-line data set will be intuitively helpful.

In the sequel, two different approaches to the problem of option prices prediction are investigated.

Generic approach. In a generic approach, we use a *time-varying* nonparametric model (MLP network) to track the stochastic dynamics. We use strike price X and maturity

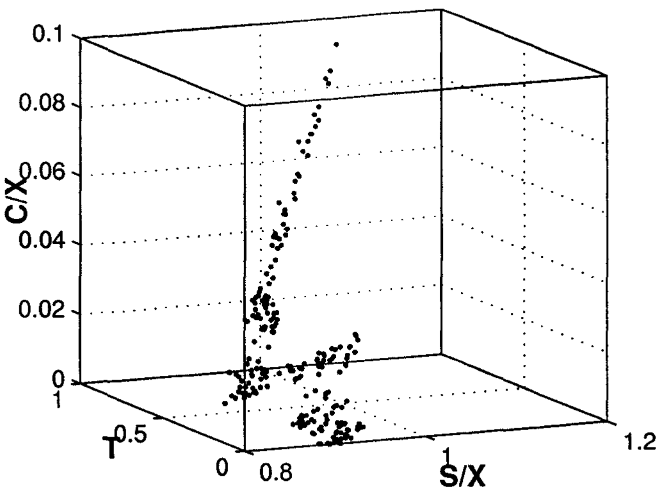

Figure 6.7: Scatter plot of C/X, S/X, and normalized maturity time T for strike price data 3325.

time T as two inputs (with appropriate normalization preprocessing) feeding an MLP with architecture net2-6-2 (two inputs, two outputs, and six hidden units), where the two outputs correspond to the call option and put option prices. We have tried different options data and compared our algorithms with the conventional extended Kalman filter (EKF) and HySIR algorithms (deFreitas et al., 2000). The specific parameters for this task are: $\sigma = 0.8$, $\alpha = -0.7$. Using $N_p = 50$ particles, the Monte Carlo average results are summarized in Table 6.2. Generally, when the number of particles is increased, the prediction performance is also improved. The prediction curves (of one trial) of call and put option prices for the strike price data 3125 and 3325 with Algorithm-2 are shown in Figure 6.6, respectively. As seen from the figures, the sampling-based ALOPEX algorithm produces a reasonable tracking trajectory of the highly non-stationary price data, though the exact prediction results are not very accurate.

From Table 6.2, it is observed that the modified ALOPEX-B algorithm fails to track the sequential data; the performance of sampling-based ALOPEX algorithms is significantly better than ALOPEX-B, close to or slightly better than EKF, and slightly worse than the HySIR algorithm. Under the same conditions, the HySIR algorithm's algorithmic complexity ($\mathcal{O}(N_p N^2 N_{out})$, where N_{out} denotes the number of the MLP output neurons (deFreitas et al., 2000)), however, is much greater than that of our algorithms ($\mathcal{O}(N_p N)$). In terms of CPU time, our algorithms need slightly more time per step than the EKF for this task. Nevertheless, it is expected that when the size and structural complexity of the neural network are increased, our algorithms may exhibit more computational advantage. It may thus be said that our proposed sampling-based ALOPEX algorithms provide a good trade-off between performance and computational complexity for tracking the option prices tendency. In addition, they are also amenable to parallel implementation.

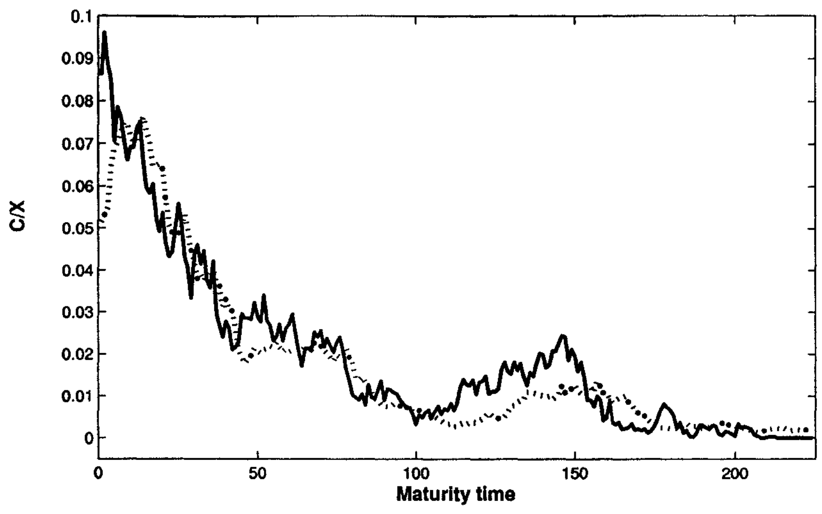

Figure 6.8: The C/X prediction curve (for strike price data 3225) produced by Algorithm-2. Solid line: true value; dotted line: predicted value.

Data-driven approach. In terms of financial data prediction, it is often beneficial to explore the structural properties of the data, even if the data are of limited size. For the financial data at hand, we also investigate data-driven predictive model. Under certain assumptions, (6.12) can be simplified by normalizing the call option price C and stock price S by the strike price X; in particular, we have[4]

$$C/X = f(S/X, T). \tag{6.13}$$

The correlation analysis between C/X and S/X and normalized T is shown as a scatter plot in Figure 6.7. In the data-driven approach, we use an MLP net2-4-1 to model the dynamics (6.13) and test the tracking performance of Algorithm-2. Using 50 particles, one prediction curve for the call option prices is shown in Figure 6.8. Compared to the generic approach (see Figure 6.6), the data-driven approach appears to produce more accurate prediction results.

6.3.5 System Identification

In the following, we will test the sampling-based ALOPEX algorithms for the system identification problem (Ljung, 1999; Sjöberg et al., 1995; Johansson, 1993). The purpose of the experiments is to illustrate the suitability of the proposed algorithms for an on-line "black-box" (neural network) modeling approach. See Figure 6.9 for an illustration.

Robot-Arm Dynamics. Let us consider a two-link robot-arm system, which is illustrated in Figure 6.10; the solid and dashed lines in the left panel of Figure 6.10 show the

[4]Theoretically, this normalization is valid at least when the stock returns are independently distributed (Hutchinson, 1994).

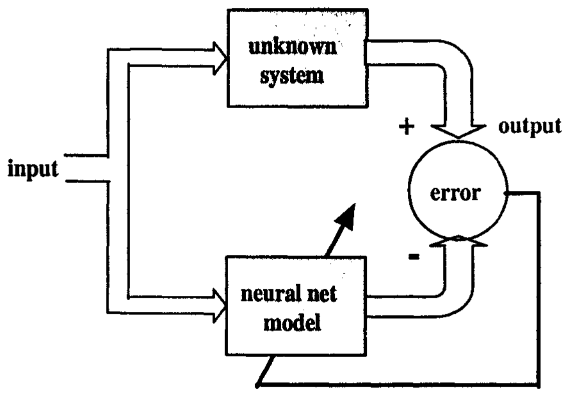

Figure 6.9: Block diagram of system identification using a black-box modeling approach.

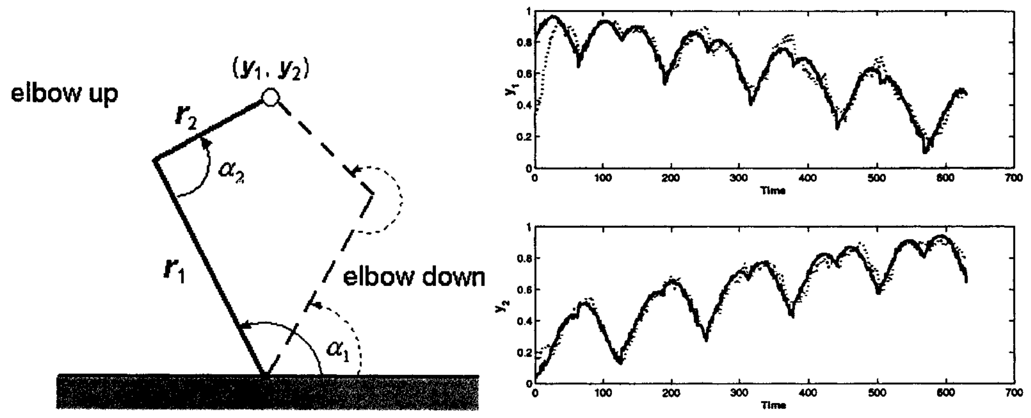

Figure 6.10: *Left panel:* Two-line robot arm. *Right panel:* predicted trajectories (dotted line).

"elbow up" and "elbow down" situation, respectively. For a given pair of angles (α_1, α_2), the end-effector position of the robot arm is determined, whose system is described by the Cartesian coordinates:

$$\begin{aligned} y_1 &= r_1 \cos(\alpha_1) - r_2 \cos(\alpha_1 + \alpha_2), \\ y_2 &= r_1 \sin(\alpha_1) - r_2 \sin(\alpha_1 + \alpha_2), \end{aligned}$$

where $r_1 = 0.8$, $r_2 = 0.2$; $\alpha_1 \in [0.3, 1.2]$ and $\alpha_2 \in [\pi/2, 3\pi/2]$. Finding the mapping from (α_1, α_2) to (y_1, y_2) is referred to as *forward kinematics.* Reformulating the system dynamics in a state-space form so as to obtain sequential data for the problem at hand, we may write

$$\mathbf{x}_{t+1} = h(\mathbf{x}_t) + \mathbf{w}_t, \tag{6.14a}$$

$$\mathbf{y}_t = \begin{bmatrix} \cos(\alpha_{1,t}) & -\cos(\alpha_{1,t} + \alpha_{2,t}) \\ \sin(\alpha_{1,t}) & -\sin(\alpha_{1,t} + \alpha_{2,t}) \end{bmatrix} \begin{bmatrix} r_1 \\ r_2 \end{bmatrix} + \mathbf{v}_t, \tag{6.14b}$$

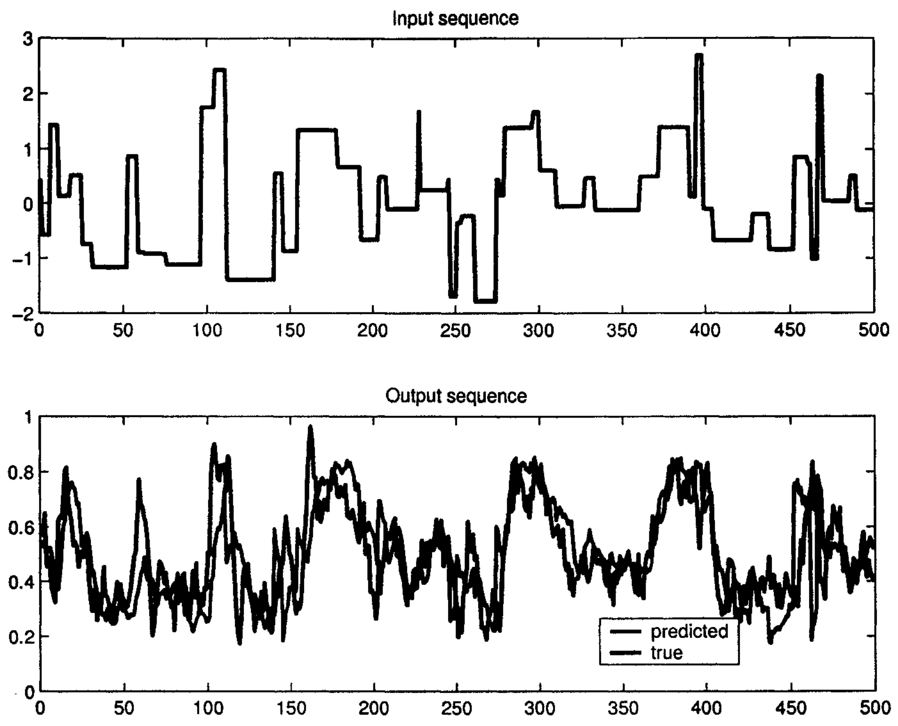

Figure 6.11: *Top panel:* input sequence. *Bottom panel:* original and predicted output sequences.

where $h(\cdot)$ is a piecewise linear function, $\mathbf{x} = [\alpha_1, \alpha_2]^T$, $\mathbf{y} = [y_1, y_2]^T$, and the noise vectors are chosen as $\mathbf{w}_t \sim \mathcal{N}(0, \text{diag}\{0.008^2, 0.08^2\})$, $\mathbf{v}_t \sim \mathcal{N}(0, 0.005 \times \mathbf{I})$. The task of system identification is to train a neural network, given the input-output pairs, to learn the underlying robot arm dynamics and to provide a predictive model for the dynamics. A total set of 630 pairs of input-output data is constructed, where the input sequence follows a piecewise linear dynamics subject to a Gaussian noise perturbation as described in (Chen, 2003a). In order to track the system dynamics, we apply Algorithm-2 to train a two-layer MLP net2-6-2, using 20 particles. The system identification results are shown in the right panel of Figure 6.10. As shown in the figure, the network quickly tracks the system dynamics, roughly within about 50 iterations.

On-line and Off-line Identification of Nonlinear Dynamics. We further consider another system identification task. The data were obtained by simulation of an open-loop stable, nonlinear continuous system, which is unknown to the modeler and therefore is regarded as a black-box.[5] The data consist of a sequence of input-output pairs with 500 samples; they were further scaled with zero mean and normalized appropriately before applying for training. Similar to the previous example, we use Algorithm-2 to train a

[5]The data file, `spmdata.mat`, was obtained from the Matlab© (the MathWorks, Inc.) system identification toolbox (Nø rgaard, 2000).

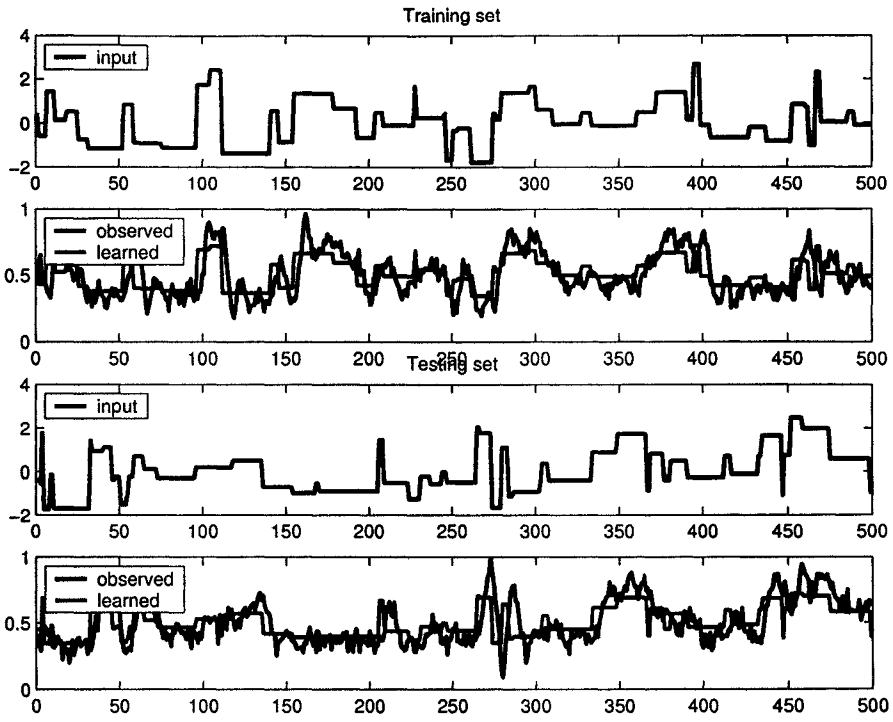

Figure 6.12: Off-line learning results of training and testing sets using SISO structure.

two-layer MLP net1-4-1, using 50 particles. Again, note that the learning is performed in an on-line fashion. One trial of prediction result is shown in Figure 6.11. As seen from the figure, the on-line system identification result is reasonably good. In addition, we can also apply the network for off-line learning to this system identification task; once the learning is accomplished, new data are used to test the generalization performance. It should be noted that since in batch learning all the data are accessible, the data can be analyzed for model order validation, and the network architecture is therefore *not* necessarily to be net1-4-1.

First, let us consider a single-input, single-output (SISO) identification structure; namely, we use the same network architecture net1-4-1 as in on-line learning. Figure 6.12 shows the off-line learning results (using 10 particles) of the training and testing data sets. As observed in the figure, the SISO identification result obtained from off-line learning is utterly different from Figure 6.11; the network's behavior seems to recover the input rather than to approximate the output.

Second, we consider a multiple-input-single-output (MISO) identification structure, where the model order of the data is more than one. Specifically, we use a network architecture net2-6-1 to fit the data, where the current output depends on the current input as well as the previous output; namely, we assume the data has a nonlinear autoregressive-moving-average (ARMA) structure. For comparison, we also fit the data with a linear second-order output error (OE) model (using the Matlab© "oe" function in the system

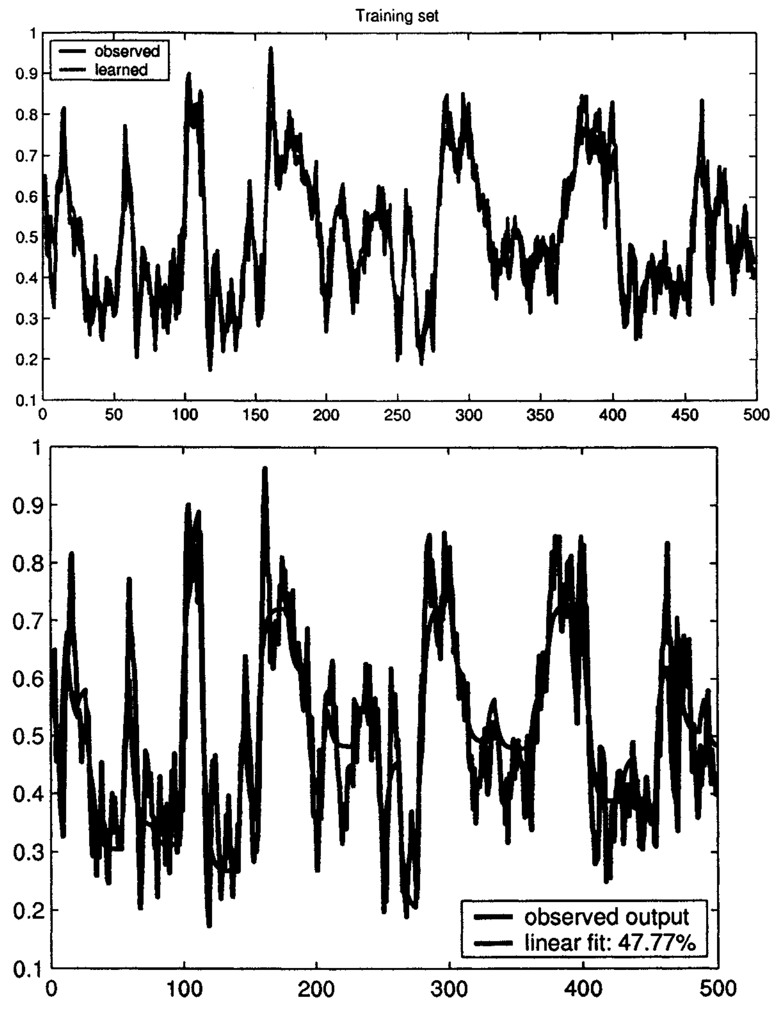

Figure 6.13: *Top panel:* off-line training result from the nonlinear neural network. *Bottom panel:* identification result from the linear OE model.

identification toolbox). As expected, the linear identification result is less oscillating, with a smoothing effect on the output observations. In contrast, the nonlinear off-line MISO neural network model produces a better prediction result than the linear model (Figure 6.13), as well as the on-line learning (Figure 6.11) and off-line SISO model (Figure 6.12). In addition, it can be imagined that if we further increase the order of the (linear or nonlinear) ARMA model, the prediction performance will be further improved for the batch learning.

6.4 Supervised Learning via Error Entropy Minimization

Information-theoretic learning (Principe et al., 2000; Erdogmus, 2002) has recently attracted intense attention for adaptive systems in the literature. The key theme of information-theoretic (supervised or unsupervised) learning is to, directly or indirectly, estimate the information-theoretic measures, such as the entropy, KL divergence, or generalized Ciszer's divergence. A challenge of the information-theoretic (supervised or unsupervised) learning is the need of estimating the probability density using an efficient nonparameteric estimator, especially in a high-dimensional space. For difference purposes, different nonparametric entropy estimators, such as the higher-order moment expansion, Parzen's kernel estimator, and KD-tree, have been developed. Due to the use of an informative metric, information-theoretic learning has demonstrated a lot of potential in machine learning.

For a computational simplicity reason, Erdogmus and Principe (2002a,b) proposed to use quadratic Renyi entropy to replace the conventional Shannon entropy,[6] and used its Parzen's kernel estimator as a potential criterion for supervised learning. As reported in (Erdogmus and Principe, 2002a), the neural networks trained with entropy metric, compared with the MSE metric, is preferable in the non-Gaussian distributed error scenario. Nevertheless, it was also observed that the gradient-based learning often suffers from the local minimum problem, even the objective function of entropy is used. In addition, when the quadratic Renyi entropy is involved, the gradient estimate by the back-propagating error can be quite computationally demanding for a recurrent neural network. Motivated by these two issues, we investigate the sampling-based ALOPEX algorithm for training a neural network using error entropy minimization scheme (Erdogmus and Principe, 2002a,b). It is our hope that the comparatively simple gradient-free and cost function independent optimization methods provide an alternative approach for the entropy minimization scenario. Note that using our learning algorithm, the optimization procedure need not be modified, except the objective function is treated differently.

6.4.1 Preliminaries

Let us first briefly review some important concepts regarding entropy. Without loss of generality, let ξ be a univariate continuous random variable, and $p(\xi)$ be its probability density function. The Shannon entropy is defined as (Shannon, 1948):

$$H(\xi) = -\int_{-\infty}^{\infty} p(\xi)\log p(\xi)d\xi. \tag{6.15}$$

Note that the global minimum of Shannon entropy is attained only when $p(\xi)$ is a Dirac-delta function.

[6]The Shannon's entropy is a special case of α-Renyi's entropy when $\alpha = 1$ (from L'Hôpital's rule).

Another generalized information metric, the Renyi entropy (Renyi, 1976) is defined as:

$$H_\alpha(\xi) = \frac{1}{1-\alpha} \log \int_{-\infty}^{\infty} p^\alpha(\xi) d\xi. \tag{6.16}$$

It is known that when $\alpha \to 1$, the Renyi entropy reduces to the Shannon entropy (Cover and Thomas, 1991). Note that when $\alpha = 2$, (6.16) yields the quadratic Renyi entropy:

$$H_2(\xi) = -\log \int_{-\infty}^{\infty} p^2(\xi) d\xi. \tag{6.17}$$

Using the Parzen's kernel windowing technique (Parzen, 1967), the pdf of random variable ξ (which corresponds to the error signal in the supervised learning case) can be written as:

$$\hat{p}(\xi) = \frac{1}{\ell} \sum_{i=1}^{\ell} K(\xi - \xi_i; \sigma^2), \tag{6.18}$$

where ℓ denotes the number of total observations; and $K(\cdot; \sigma^2)$ is a symmetric, continuous, differentiable, and unimodal kernel function with σ being the width parameter. A common kernel is the Gaussian kernel and σ corresponds to the standard deviation. Plugging the Parzen's estimator (6.18) to (6.17) yields the quadratic Renyi entropy estimate (Erdogmus and Principe, 2002a):

$$\hat{H}_2(\xi) = -\log \hat{V}(\xi), \tag{6.19}$$

$$\hat{V}(\xi) = \Big(\frac{1}{\ell}\sum_{i=1}^{\ell} K(\xi - \xi_i; \sigma^2)\Big)^2 = \frac{1}{\ell^2}\sum_{i=1}^{\ell}\sum_{j=1}^{\ell} K(\xi_i - \xi_j; 2\sigma^2), \tag{6.20}$$

where $V(\xi)$, the argument of the logarithm in quadratic Renyi entropy is called the *information potential* (Principe et al., 2000). Note that when $\xi = 0$, $H_2(\xi)$ achieves the minimum $-\log K(0; 2\sigma^2)$ by observing that

$$\sum_{i=1}^{\ell}\sum_{j=1}^{\ell} K(\xi_i - \xi_j; 2\sigma^2) \leq \ell^2 K(0; 2\sigma^2). \tag{6.21}$$

In order to minimize the error entropy, we often need to calculate the gradient, the first-order derivative w.r.t. the unknown parameters (Erdogmus and Principe, 2002a,b). For the (feedforward or recurrent) multilayer networks, calculating the gradient $\frac{\partial \hat{H}_2(\xi)}{\partial \theta} = \frac{-1}{\hat{V}(\xi)} \frac{\partial \hat{V}(\xi)}{\partial \theta}$ can be computationally demanding; in addition, it can still suffer from the local minimum problem. These two issues motivate us to use the ALOPEX-type algorithms developed earlier for training the neural networks.

6.4.2 Experimental Result

As noted in (Erdogmus and Principe, 2002a), since entropy is independent of the expected mean of the random variable, training with entropy metric will lead the network to converge to a set of optimal weights, which may not yield zero-mean error; this problem can be rescued by properly modifying the bias of the output unit of the neural network to yield a zero-mean error over the training data set just after the training process is terminated.

We consider training a RMLP network for a chaotic time series prediction problem (Haykin and Principe, 1998; Patel and Haykin, 2001), which can be viewed as an off-line regression problem. Specifically, the continuous-time Lorenz dynamics is described by three coupled nonlinear differential equations:

$$\dot{x}_t = -ax_t + ay_t, \tag{6.22a}$$

$$\dot{y}_t = -x_t z_t + rx_t - y_t, \tag{6.22b}$$

$$\dot{z}_t = x_t y_t - bz_t, \tag{6.22c}$$

where $a = 10, b = 8/3$, and $r = 28$. Using a step-size of 0.01, a total of noise-free 10000 samples are generated. After discarding the first 2000 samples to avoid the transient period, the subsequent 5000 samples are used as training set and the rest 3000 samples are used as testing set; all of samples are normalized within the region $(-1, 1)$. The input-output data are constructed, in light of the *Takens' Theorem* (Takens, 1981), as follows

$$\begin{aligned} \mathbf{x}_t &= \left[x_t, x_{t-\tau}, \cdots, x_{t-(d_E-2)\tau}, x_{t-(d_E-1)\tau}\right]^T, \\ y_t &= x_{t+\tau}, \end{aligned}$$

where $d_E = 3$ and $\tau = 4$ are the *embedding dimension* and the *embedding delay* parameters of the Lorenz x-axis data, respectively. The recurrent network architecture is chosen to be net4-8-1, with one feedback loop from the output to input layer. For optimization convenience, we choose the objective function as a form of modified quadratic Renyi entropy:

$$\begin{aligned} E &= \hat{H}_2(e) + \log K(0; 2\sigma^2) \\ &= -\log\left(\sum_{i=1}^{\ell}\sum_{j=1}^{\ell} K(e_i - e_j; 2\sigma^2)\right) + 2\log\ell + \log K(0; 2\sigma^2) \\ &= -\log\left(\sum_{i=1}^{\ell}\sum_{j=1}^{\ell} K(e_i - e_j; 2\sigma^2)\right) + Const, \end{aligned} \tag{6.23}$$

which, different from (6.19), has a zero minimum such that $\exp(-E_{\min}) = 1$; the error between the desired and network outputs is defined as: $e_i = y_i - \hat{y}_i \quad i = 1, \cdots, \ell$; the constant offset can be calculated in advance.

The training phase is terminated after 10,000 iterations or no further improvement is observed; the step-size parameters are gradually decreased after the first 1000 iterations.

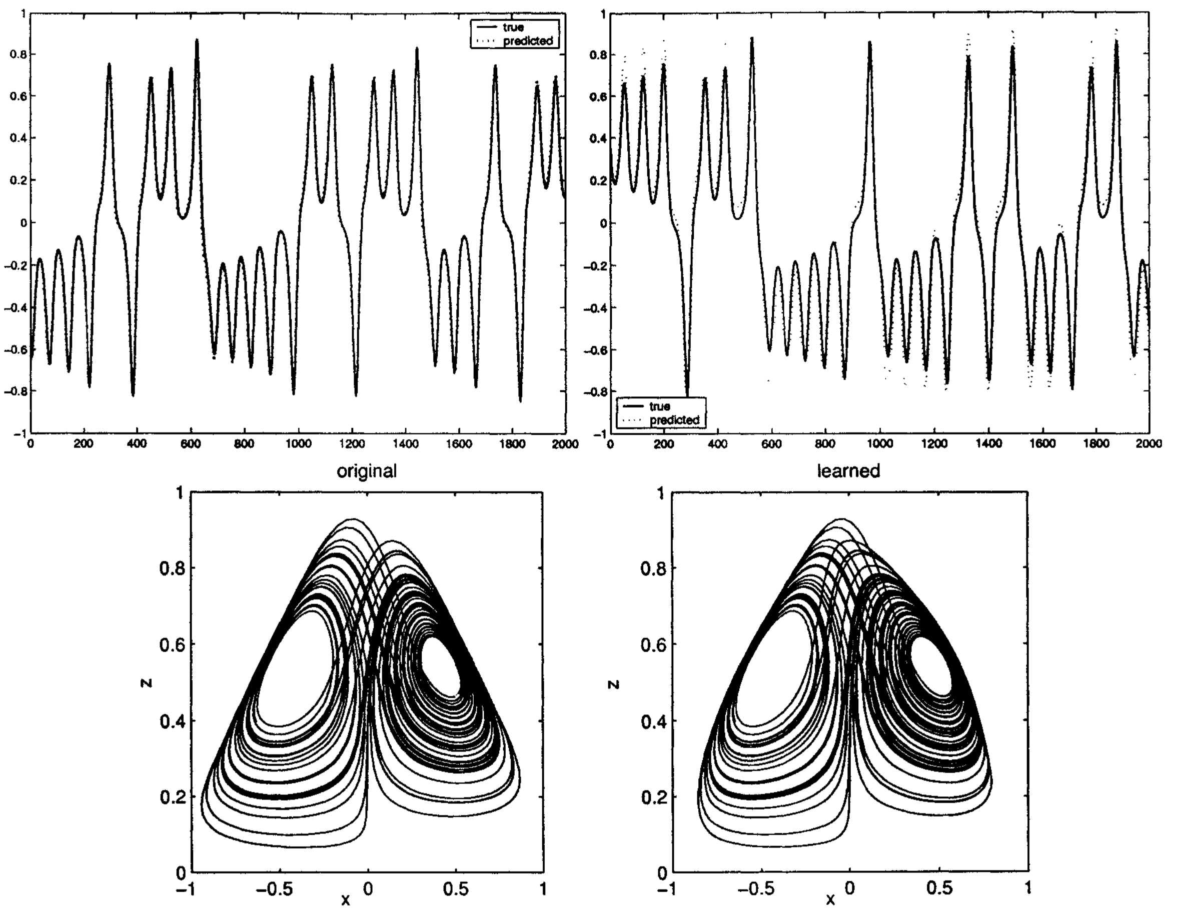

Figure 6.14: *Top left panel:* prediction result on training data. *Top right panel:* prediction result of testing data. *Bottom two panels:* a comparison between the original and the learned Lorenz dynamics.

Due to the nature of the difficulty in learning chaotic dynamics, the error curve usually exhibits certain oscillation at the initial learning stage. Upon completion of training, a close-loop or open-loop evaluation can be performed for the testing set. The four-step-ahead prediction result of training data and the iterated prediction result of testing data are shown in Figure 6.14. As seen from the figure, the network has approximated reasonably well the chaotic dynamics. Figure 6.15 shows the histogram of the prediction errors for the training set, which indicates a non-Gaussian characteristic.

Although theoretically appealing, a disadvantage of error-entropy minimization is the requirement of an accurate kernel estimator. However, the width of the optimal kernel size is often unknown and requires a trial-and-error procedure; the selection procedure becomes even complicated when the error is high-dimensional. Another main disadvantage of this learning paradigm is that it is computationally prohibitive (even for off-line learning) to calculate the error entropy if the number of the training samples, ℓ, is very large, which requires $\mathcal{O}(\ell^2)$ operations; it is even worse for the sampling-based ALOPEX algorithm,

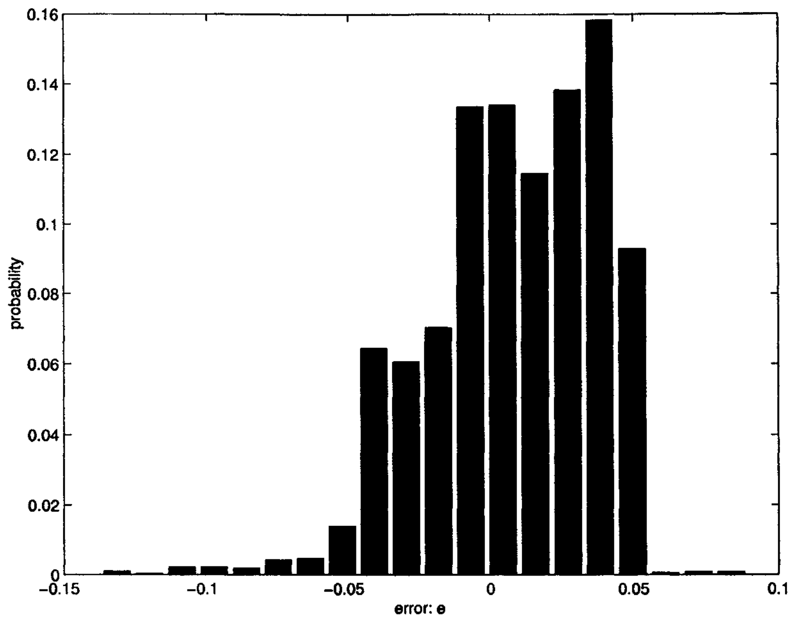

Figure 6.15: Prediction error histogram of the training data for Lorenz time series prediction.

whose complexity is of $\mathcal{O}(\ell^2 N_p)$. Hence, the tradeoff between the performance gain and the computational cost is an important issue to be considered in practice. In our experiments, no significant performance difference was observed between using the MSE criterion and entropy minimization; however, the computational time using entropy minimization is significantly higher.

6.4.3 A Note on the Objective Functions

Although the ALOPEX-type optimization procedure is independent on the objective function to be used, the form of the objective function has a direct influence on the optimization or learning performance; it is worth making some comments here.

- The most popular objective function is the minimum mean-squared-error (MMSE), which assumes a form of L_2 norm (let e be the error signal, L_2 norm is defined as $E(\mathbf{e}) = \|\mathbf{e}\|^2$). From a statistical estimation viewpoint, L_2 norm corresponds to the maximum likelihood estimation by assuming the error signal $\mathbf{e}$ is Gaussian distributed: $p(\mathbf{e}) \propto \exp(-\|\mathbf{e}\|^2)$. If e is zero-mean, then MMSE can be viewed as a minimum variance estimation. With a fixed variance, Gaussian density is known to have a maximum entropy (Cover and Thomas, 1991):

$$H(\mathbf{x}) \leq \frac{1}{2} \log \det(\mathrm{Cov}[\mathbf{x}]) + \frac{d}{2} \log(2\pi \exp(1)),$$

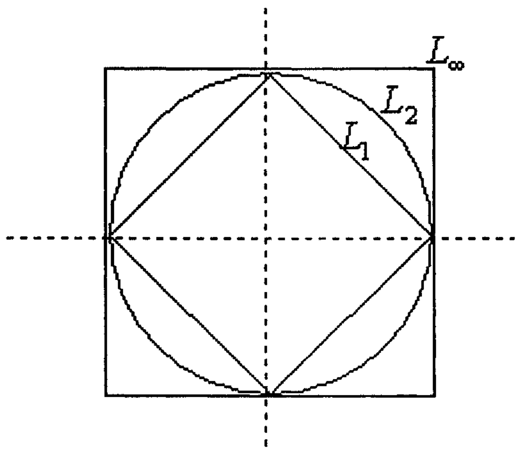

Figure 6.16: A geometrical illustration of different norm criteria.

where the second term at the right-hand-side is a constant. Therefore, under the constant entropy constraint, MMSE can also be viewed as a minimax estimator.

- L_1 norm takes the absolute value of the error, $E(e) = |e|$; from a statistical estimation viewpoint, it assumes the error e is Laplacian distributed: $p(e) \propto \exp(-|e|)$. L_∞ norm is defined as $|\mathrm{e}|_\infty = \max_i |e_i|$; minimizing the L_∞ norm is a minimax estimation problem. A geometrical illustration of the difference between different norm criteria is illustrated in Figure 6.16.

- If the objective function is a form of the cross-entropy (in two-class classification), the data are assumed to be i.i.d. and follow the Bernoulli distribution:

$$\prod_i (\hat{y}_i)^{y_i}(1-\hat{y}_i)^{1-y_i},$$

where y_i denotes the desired posterior probability of the ith pattern, and $\hat{y}_i$ denotes the network output associated to the ith input pattern. Minimizing the negative log-likelihood of the above distribution yields the form of cross-entropy (6.10). See Bishop (1995) for more discussions on the objective (error) function.

- If the objective function is a form of the entropy, $E(e) = H(e)$; let $V = \exp(-H(e))$, then V denotes the *effective state space volume* of the random variable e. Ideally, to minimize the error entropy, we hope that $p(e)$ is a Dirac-delta function; in contrast, Gaussian density will produce a maximum entropy given the fixed variance.

6.5 Discussion

6.5.1 Tricks of the Trade

It is noted that there are many tuning parameters involved in our proposed sampling-based ALOPEX algorithms. In practice, finding these optimal parameters might be time-consuming and discouraging. In light of our empirical experiments, we summarize some rules of thumb for selecting those free parameters:

- Step-size/learning rate parameters: For the ALOPEX-B algorithm, η is often chosen in the range $[0.05, 0.1]$, γ is fixed to be 0.01 in most of our experiments.

- Forgetting parameter: In the ALOPEX-B algorithm, λ is often taken from the region $[0.35, 0.7]$; the smaller the λ, the less influence is induced by previous error estimates. For on-line learning (on sequential data), λ is often set as a small value.

- Relaxing parameter: α is taken from the region $[-1, 1]$. when $\alpha > 0$, it is over-relaxation, when $\alpha < 0$, it is under-relaxation. In the initial training, α can be set as positive to accelerate the initial convergence; as the error surface becomes more hilly, we can switch to under-relaxation. In our experiments, α is always set as a *negative* value for on-line learning.

- Momentum coefficient: In light of physical interpretation (Qian, 1999), gradient-type optimization can be imagined as moving a massless particle (i.e., $\boldsymbol{\theta}$) towards the bottom of a potential well. Imagining the massless particle as a particle with a quantitative mass, we know from Newton mechanics that the greater the mass, the greater is the momentum. Since the normalized importance weights are directly related to the likelihood values, ideally it is hoped that the "important" particles (with higher likelihood) are more active, therefore we assign greater momentum values to them, and smaller momentum values to the "idle" particles. Heuristically, for the ith particle, we may set $\beta^{(i)} = \tilde{W}^{(i)}\beta_0$, where $\beta_0 = 1 - \eta$ is a constant. Besides the preceding sophistication, an alternative, simple setup can be: $\beta = \eta/10$.

- Diffusion coefficient: σ initially is set as a small constant (depending on the region of the parameter $\boldsymbol{\theta}$); as batch learning progresses, it may follow an annealing schedule after 1000 iterations $\sigma = \sigma_0/\log(t)$. In on-line learning, σ does not employ any annealing schedule.

- If parameter $\boldsymbol{\theta}$ is subject to a positive constraint (e.g., the width parameter of the RBF), one can introduce a surrogate parameter, $\boldsymbol{\vartheta} \equiv \ln \boldsymbol{\theta}$ or $\boldsymbol{\theta} \equiv \exp(\boldsymbol{\vartheta})$, and then use the ALOPEX procedure to update the surrogate parameter $\boldsymbol{\vartheta}$ (with a different prior, of course).

6.5.2 Statistical Physics Interpretation

The ALOPEX algorithm itself originated from a statistical physics idea, similar to the Metropolis algorithm (Metropolis et al., 1953) and simulated annealing (Kirkpatrick et al., 1983). It is therefore befitting that we explore a statistical physics interpretation of the sampling-based ALOPEX algorithms in terms of an interacting particle system (IPS). The IPS (Liggett, 1985) can be regarded as a dynamic interactive system with a collection of many particles interacting according to simple and local rules. IPS has been successfully utilized to model such diverse phenomena as magnetism, population growth, and propagation of information and opinions.

Imagine the sampling-based ALOPEX algorithm as an interactive dynamical composition system. On the one hand, the elements in the system are *spatially independent* (i.i.d. samples), and *temporally correlated* (correlative learning rule); on the other hand, the elements are *globally correlated* (from the correlation learning rule, the change of each element is influenced by others), but also *locally independent*. Finally, the system is not only *cooperative* (in parameter space, because every element contributes to the same energy function) but also *competitive* (in sample space, because different samples try to find the minimum energy, so the one that finds a locally minimal energy has the highest likelihood). In light of these observations, the sampling-based ALOPEX algorithm provides a simulation analog for systems with combined cooperative and competitive behavior, which we deem to be a feature of the human brain.

6.5.3 Hindsight

Upon finishing the work reported herein over one year ago, some day I happened to reread Marvin Minsky's illuminating review paper "Steps towards artificial intelligence". Surprisingly enough, I found that in the article, Minsky has discussed a similar idea over 40 years ago; in particular, it reads (Minsky, 1961):

> Multiple simultaneous optimizers search for a (local) maximum value of some function $E(x_1, \dots, x_n)$ of several parameters. Each unit u_i independently "jitters" its parameter x, perhaps randomly, by adding a variation $d_i(t)$ to a current mean value $m_i(t)$. The changes in the quantities x_i and E [namely, Δx_i and ΔE] are correlated, and the result is used to slowly change m_i. The filters are to remove DC components. This technique, a form of coherent detection, usually has an advantage over methods dealing separately and sequentially with each parameter. Cf. the discussion of "informative feedback" in Wiener [1948, p. 133]. A great variety of hill-climbing systems have been studied under the names of "adaptive" or "self-optimizing" servomechanisms.

The thoughtful reader can readily see the above statement is indeed a description of the idea underlying the stochastic correlative learning algorithms.

Chapter 7

Conclusions

7.1 Summary

In summary, we have systematically investigated stochastic/Monte Carlo approaches for different machine learning problems. The concept of correlation plays a vital role throughout the thesis, and we argue that correlation serves as a mathematical framework for many statistical learning algorithms, as reviewed in Chapter 2. In particular, the ALOPEX, being a stochastic correlative learning rule, establishes the foundation of this thesis. The motivation for using ALOPEX or its variants as optimization tools for learning arises from its several appealing features: (i) gradient-free; (ii) network architecture independence; (iii) objective function independence; (iv) synchronous update; (v) simplicity and ease of hardware implementation. In so pursing the research, we have kept several goals in our mind:

- exploring flexible optimization procedure for difficult or global optimization problems;
- combining stochastic and deterministic algorithms (in analogy with the *free will* and *volition*) to boost conventional algorithms, for that Monte Carlo approaches play a key role;
- applying sequential Monte Carlo and MCMC methods for seeking a Bayesian solution in various estimation and inference scenarios.

As evidenced in our investigations, Monte Carlo optimization provides a promising framework for achieving those goals; in addition, it also offers a tradeoff between the algorithmic performance and computational complexity. In general, increasing computational resources improves optimization performance.

In this thesis, we have proposed several new theoretical and algorithmic developments; we have also succeeded in applying the developed algorithms to many (including some novel) applications. The main contributions of the dissertation are highlighted here:

Stochastic correlative learning algorithms for perceptual learning:

- Proposing a modified version of the ALOPEX algorithm to improve the convergence speed.
- Proposing a biologically plausible stochastic correlative firing mechanism for several figure-ground segregation tasks in sensory (visual and auditory) perception, and applying (for the first-time) the ALOPEX to accomplish several perceptual tasks. In the simplified cocktail party problem, the algorithm is capable of extracting the auditory stream using two sensors given more than two (up to 4) non-Gaussian sources.
- Applying the developed ALOPEX algorithm to optimize the design parameters of a Neurocompensator with a novel objective function, as an ingredient of a model-based hearing-aid device.

Monte Carlo methods for Bayesian inference and learning:

- Developing a gradient proposal particle filter and a Turbo-particle filter, applying the improved algorithms to two synthetic target tracking problems and a real-life MIMO wireless channel estimation problem.
- By combining particle filtering and the ALOPEX procedure, developing two novel Monte Carlo sampling-based ALOPEX algorithms for sequential parameter optimization. In particular, applying them to various neural network learning problems, including off-line pattern recognition and regression, online tracking and system identification.
- Applying the ALOPEX-type algorithms to supervised learning using the quadratic Renyi entropy minimization scheme.

The contributions have been justified by a number of peer-reviewed publications listed in Chapter 1.

7.2 Closing Remarks

Correlation constitutes a fundamental mechanism of the brain, with crucial role in various human functions including human perception, association, learning, and memory recall; it is therefore not surprising to see (Chapter 2) that many existing learning rules can be traced to the root of correlative learning. Correlation-based learning and gradient-free optimization open an avenue for many machine learning and neural computation topics. In particular, we believe that some of the work reported in this thesis is appealing for learning the hierarchical networks, which have modular and feedback structures; when the algorithms are viewed in the context of spiking neurons, they do possess a neurobiologically plausible format. Specifically, the benefit can be twofold:

- First, in perceptual learning, once we design a global objective function, we are capable of applying the stochastic correlative learning rule for updating the unknown parameters. Multiple objective functions are also allowed to be alternatingly optimized, for example, the "look-think-do" process discussed in Chapter 3.
- Second, we are also capable of performing some sort of spiking neuron optimization (Chapter 4 can be regarded as an effort of this type). Now, we can monitor the firing rate of the neuron (or population of neurons) and use that information to adapt the action of a neural network; we can also measure the correlative or synchronous firing rate between different groups of neurons and use it to optimize the neural codes.

As seen in this thesis, Monte Carlo sampling methods provide a powerful tool for probabilistic inference and optimization. We also believe that some of the work reported in the thesis might have a beneficial impact on the industrial practice. In tackling real-time sequential data, we are obliged to design simple yet reliable toolkits for either prediction, control, or communications. With the ever-growing computing power, now we have the option and freedom to choose a tradeoff between computational complexity and performance gain. As shown in many instances discussed in Chapters 5 and 6, a careful design of Monte Carlo simulations might significantly boost the performance of conventional (often being deterministic) approach.

Appendix A

Expectation-Maximization (EM) Algorithm

A.1 EM Algorithm as a Lower-Bound Maximization

Given some observation data $\boldsymbol{x}$ and a model family parameterized by $\boldsymbol{\theta}$, the goal of the EM algorithm is to find $\boldsymbol{\theta}$ such that the log-likelihood $\log p(\boldsymbol{x}|\boldsymbol{\theta})$ is maximized (Dempster et al., 1977).

Theorem A.1 (Cover and Thomas, 1991) *If f is a convex function and x is a random variable, then*

$$\mathbb{E}[f(x)] \geq f(\mathbb{E}[x]) \tag{A.1}$$

Moreover, if f is strictly convex, then equality in (A.1) implies that $x = \mathbb{E}[x]$ with probability 1, i.e. x is a constant.

Assume that $a_j \geq 0$, $\sum_j a_j = 1$, $g(j) \geq 0$, in light of the Jensen's inequality (A.1), we have

$$\sum_j g(j) a_j \geq \prod_j g(j)^{a_j}, \tag{A.2}$$

which is also known as the *geometric mean inequality*.

Let $f(\boldsymbol{\theta}) = p(\boldsymbol{x}, \boldsymbol{\theta})$, by introducing a hidden (latent) variable $\boldsymbol{z}$ (without loss of generality, we assume $\boldsymbol{z}$ to be continuous-valued), we rewrite $f(\boldsymbol{\theta})$ as

$$f(\boldsymbol{\theta}) = p(\boldsymbol{x}, \boldsymbol{\theta}) = \int_{\boldsymbol{z}} p(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\theta}) d\boldsymbol{z}. \tag{A.3}$$

Given a proposal distribution $q(\boldsymbol{z})$ subject to $\int_{\boldsymbol{z}} q(\boldsymbol{z}) = 1$, in light of the Jensen's inequality we obtain:

$$f(\boldsymbol{\theta}) = \int_{\boldsymbol{z}} d\boldsymbol{z} \frac{p(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\theta})}{q(\boldsymbol{z})} q(\boldsymbol{z}) \geq g(\boldsymbol{\theta}, q(\boldsymbol{z})) = \prod_{\boldsymbol{z}} \left(\frac{p(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\theta})}{q(\boldsymbol{z})} \right)^{q(\boldsymbol{z})} \tag{A.4}$$

Define the logarithm of the lower-bound $g(\boldsymbol{\theta}, q(\boldsymbol{z}))$ as $G(\boldsymbol{\theta}, q)$:

$$G(\boldsymbol{\theta}, q) = \log g(\boldsymbol{\theta}, q(\boldsymbol{z})) = \int_{\boldsymbol{z}} d\boldsymbol{z}\Big(q(\boldsymbol{z}) \log p(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\theta}) - q(\boldsymbol{z}) \log q(\boldsymbol{z})\Big) \tag{A.5}$$

Now we want to optimize $\boldsymbol{\theta}$ to maximize the lower-bound $G(\boldsymbol{\theta}, q)$, which is equivalently to maximize $f(\boldsymbol{\theta})$ since $\int_{\boldsymbol{z}} q(\boldsymbol{z}) d\boldsymbol{z} = 1$; in light of the Lagrange multiplier, we have an alternative form of $G(\boldsymbol{\theta}, q)$:

$$G(\boldsymbol{\theta}, q) = \lambda\Big(1 - \int_{\boldsymbol{z}} q(\boldsymbol{z}) d\boldsymbol{z}\Big) + \int_{\boldsymbol{z}} d\boldsymbol{z}\Big(q(\boldsymbol{z}) \log p(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\theta}) - q(\boldsymbol{z}) \log q(\boldsymbol{z})\Big) \tag{A.6}$$

Taking the derivative of $G(\boldsymbol{\theta}, q)$ with respect to $q(\boldsymbol{z})$ and setting to be zero yields

$$\frac{\partial G(\boldsymbol{\theta}, q)}{\partial q(\boldsymbol{z})} = -\lambda + \log p(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\theta}) - \log q(\boldsymbol{z}) - 1 = 0, \tag{A.7}$$

which further follows that $\log q(\boldsymbol{z}) = \log p(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\theta}) - (\lambda + 1)$ and

$$q(\boldsymbol{z}) = \frac{p(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\theta})}{\int_{\boldsymbol{z}} p(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\theta}) d\boldsymbol{z}} = p(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{\theta}) \tag{A.8}$$

Replacing the above equation to (A.4), it follows that

$$\begin{aligned} g(\boldsymbol{\theta}, q(\boldsymbol{z})) &= \prod_{\boldsymbol{z}} \left(\frac{p(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\theta})}{p(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{\theta})}\right)^{q(\boldsymbol{z})} \\ &= \prod_{\boldsymbol{z}} p(\boldsymbol{x}, \boldsymbol{\theta})^{q(\boldsymbol{z})} \\ &= p(\boldsymbol{x}, \boldsymbol{\theta})^{\int_{\boldsymbol{z}} q(\boldsymbol{z}) d\boldsymbol{z}} \\ &= p(\boldsymbol{x}, \boldsymbol{\theta}). \end{aligned} \tag{A.9}$$

Hence maximization of $g(\boldsymbol{\theta}, q(\boldsymbol{z}))$ is indeed the maximization of $f(\boldsymbol{\theta})$ given the current estimate $\boldsymbol{\theta}$. For $G(\boldsymbol{\theta}, q(\boldsymbol{z}))$, we may also rewrite (A.5) as

$$\begin{aligned} G(\boldsymbol{\theta}, q(\boldsymbol{z})) &= \mathbb{E}_{q(\boldsymbol{z})}\left[\log \frac{p(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\theta})}{q(\boldsymbol{z})}\right] \\ &= -\mathbb{E}_{q(\boldsymbol{z})}\left[\log \frac{q(\boldsymbol{z})}{p(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{\theta})}\right] + \log p(\boldsymbol{x}, \boldsymbol{\theta}) \\ &= -D\big(q(\boldsymbol{z}) \| p(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{\theta})\big) + \log p(\boldsymbol{x}, \boldsymbol{\theta}). \end{aligned} \tag{A.10}$$

The first term of the right-hand-side of equation (A.10) is the KL divergence between distributions $q(\boldsymbol{z})$ and $p(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{\theta})$. When $q(\boldsymbol{z}) = p(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{\theta})$, the distance metric between two distributions is zero and $\{G(\boldsymbol{\theta}, q(\boldsymbol{z}))\}_{\max} = \log p(\boldsymbol{x}, \boldsymbol{\theta})$.

In summary, EM algorithm consists of two alternating steps:

- E-step: Find $q(\boldsymbol{z})$ to get the lower bound of $f(\boldsymbol{\theta})$;
- M-step: Given the current estimate $\boldsymbol{\theta}$, maximize (or increase) the lower-bound $G(\boldsymbol{\theta}, q(\boldsymbol{z}))$ over $\boldsymbol{\theta}$.

For more discussions on generalized EM algorithms (in either E or M step), see (McLachlan and Krishnan, 1997; Neal and Hinton, 1998).

A.2 EM Algorithm as an Alternating Free-Energy Maximization

From the statistical physics perspective, the EM algorithm can be understood as an alternate maximization of the free energy (Neal and Hinton, 1998).

Given the observed data $\boldsymbol{x}$, we can rewrite the log-likelihood of parameter $\boldsymbol{\theta}$ in the following form:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}) &= \log p(\boldsymbol{x}, \boldsymbol{\theta}) = \log \Big(\sum_{\boldsymbol{z}} p(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\theta})\Big) \\ &= \max_{q \in \mathcal{P}} \mathcal{F}(q, \boldsymbol{\theta}), \qquad \text{(A.11)}\end{aligned}$$

where $\mathcal{P}$ denotes the set of all probability distributions defined on the missing variable $\boldsymbol{z}$, and $\mathcal{F}(q, \boldsymbol{\theta})$ is the so-called free energy:

$$\begin{aligned}\mathcal{F}(q, \boldsymbol{\theta}) &= \mathcal{L}(\boldsymbol{\theta}) - D\big(q(\boldsymbol{z}) \| p(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{\theta})\big) \\ &= \mathcal{L}(\boldsymbol{\theta}) - \int_{\boldsymbol{z}} q(\boldsymbol{z}) \log\Big(\frac{q(\boldsymbol{z})}{p(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{\theta})}\Big) d\boldsymbol{z} \qquad \text{(A.12)} \\ &= \mathbb{E}_{q(\boldsymbol{z})}\Big[\log p(\boldsymbol{x}, \boldsymbol{z}|\boldsymbol{\theta})\Big] + H(q) \\ &= \int q(\boldsymbol{z}) \log p(\boldsymbol{x}, \boldsymbol{z}|\boldsymbol{\theta}) d\boldsymbol{z} - \int q(\boldsymbol{z}) \log q(\boldsymbol{z}) d\boldsymbol{z}, \qquad \text{(A.13)}\end{aligned}$$

where the first term of (A.13) denotes the *energy*, whereas the second term denotes the *entropy* (which is independent on $\boldsymbol{\theta}$). The EM algorithm consists of alternating maximization steps with respect to q and $\boldsymbol{\theta}$, respectively:

- E-step: Fix $\boldsymbol{\theta}$, find and solve $q = \arg\max_{q' \in \mathcal{P}} \mathcal{F}(q', \boldsymbol{\theta})$;
- M-step: Fix q, find and solve $\boldsymbol{\theta} = \arg\max_{\boldsymbol{\theta}'} \mathcal{F}(q, \boldsymbol{\theta}')$.

A.3 EM Algorithm for Fitting a Gaussian Mixture Model

Consider a d-dimensional multivariate Gaussian mixture model

$$\begin{aligned} p(\mathbf{x}) &= \sum_{j=1}^{K} p(j)p(\mathbf{x}|j) \\ &= \sum_{j=1}^{K} c_j \frac{1}{\sqrt{(2\pi)^d|\Sigma_j|}} \exp\Big(-\frac{1}{2}|\mathbf{x}-\boldsymbol{\mu}_j|^T\Sigma_j^{-1}|\mathbf{x}-\boldsymbol{\mu}_j|\Big), \end{aligned} \tag{A.14}$$

where K denotes the number of mixtures, $(\boldsymbol{\mu}_j, \Sigma_j)$ denotes the mean and (full) covariance matrix of the jth mixture; $p(j) \equiv c_j$ denotes the prior probability of the jth mixture, and $p(\mathbf{x}|j)$ denotes the probability of $\mathbf{x}$ generated from the jth mixture.

Given ℓ i.i.d. observations $\{\mathbf{x}_i\}_{i=1}^{\ell}$, the EM algorithm fitting a K-mixture Gaussian can be derived as follows (Bishop, 1995; Xu and Jordan, 1996; Duda et al., 2001):

- E-step:

$$p_{ij} \equiv p(j|\mathbf{x}_i) = \frac{p(\mathbf{x}_i|j)c_j}{\sum_{k=1}^{K} p(\mathbf{x}_i|k)c_k} = \frac{p(\mathbf{x}_i|j)c_j}{p(\mathbf{x}_i)}. \tag{A.15}$$

- M-step:

$$c_j^{new} = \frac{1}{\ell}\sum_{i=1}^{\ell} p(j|\mathbf{x}_i) = \frac{p_j}{\ell}, \tag{A.16}$$

$$\boldsymbol{\mu}_j^{new} = \frac{\sum_{i=1}^{\ell} p(j|\mathbf{x}_i)\mathbf{x}_i}{\sum_{i=1}^{\ell} p(j|\mathbf{x}_i)} = \frac{\sum_i p_{ij}\mathbf{x}_i}{\sum_i p_{ij}} = \frac{\sum_i p_{ij}\mathbf{x}_i}{\ell c_j^{new}}, \tag{A.17}$$

$$\Sigma_j^{new} = \frac{\sum_{i=1}^{\ell} p_{ij}(\mathbf{x}_i - \boldsymbol{\mu}_j^{new})(\mathbf{x}_i - \boldsymbol{\mu}_j^{new})^T}{\ell c_j^{new}}. \tag{A.18}$$

The computational complexity of above procedure is $\mathcal{O}(d\ell + K\ell^2)$.

Given ℓ i.i.d. samples $\{\mathbf{x}_i\}_{i=1}^{\ell}$ that satisfy (A.14), the log-likelihood of the data is calculated as

$$\mathcal{L} = \log\prod_{i=1}^{\ell} p(\mathbf{x}_i) = \sum_{i=1}^{\ell} \log p(\mathbf{x}_i). \tag{A.19}$$

Running the E and M steps alternatingly will produce a monotonically increasing likelihood or log-likelihood sequence until a local maximum or saddle point is approached. The convergence analysis of the EM algorithm for Gaussian mixture model is referred to (Xu and Jordan, 1996).

Appendix B

Markov Chain Monte Carlo (MCMC) Methods

B.1 Background

The Markov process is a typical stochastic process for modeling the physical world; the Markov property (Definition 5.2) underlies (and certainly simplifies) the causality inside many physical processes. Within the Markov assumption, by introducing the notion of hidden "*state*", we can obtain the *hidden Markov process*, which is defined as a *probabilistic function of Markov chains*. See Ephraim and Merhav (2002) for an excellent review regarding the related topics.

We start with introducing some basic concepts underlying the Markov chain theory.

transiency: The burn-in period that Marov chain needs to reach the equilibrium.

recurrent: Let P_i^t be the probability returning to $\mathbf{x}_i$ first time at time t, where $P_i^0 = 0, P_i^t = \Pr(\mathbf{x}_t = i, \mathbf{x}_k \neq i, k = 1, \dots, t-1 | \mathbf{x}_0 = i)$; let

$$P_i = \sum_{t=1}^{\infty} P_i^t,$$

if $P_i = 1$, i is recurrent; if $P_i < 1$, i is transient. Define $m_i = \sum_t tP_i^t$, if $m_i < \infty$, i is positive recurrent; if $m_i = \infty$, i is null recurrent.

irreducibility: Any state can be reached from any other state in a finite number of iterations.

reversible: The reversibility is defined by the following *detailed balance* condition:

$$\pi(\mathbf{x})K(\mathbf{x}, \mathbf{x}') = \pi(\mathbf{x}')K(\mathbf{x}', \mathbf{x}),$$

where $\pi(\mathbf{x})$ is the invariant probability distribution, and $K(\mathbf{x}, \mathbf{x}')$ denotes the transition kernel from state $\mathbf{x}$ to state $\mathbf{x}'$. Put in words, the unconditional probability of moving $\mathbf{x}$ to $\mathbf{x}'$ is equal to the unconditional probability of moving $\mathbf{x}'$ to $\mathbf{x}$.

ergodicity: If a Markov chain is irreducible, aperiodic and positive recurrent, then it is ergodic; in such a situation, there exists a *unique* stationary (steady-state) distribution π independent of the initial state.

Markov chain theory is mainly concerned with finding the conditions under which there exists an invariant distribution Q and conditions under which iterations of transition kernel K converge to an invariant distribution. The invariant distribution Q satisfies:[1]

$$\begin{aligned} Q(d\mathbf{x}') &= \int_X K(\mathbf{x}, d\mathbf{x}')\pi(\mathbf{x})d\mathbf{x}, \\ \pi(\mathbf{x}') &= \int_X K(\mathbf{x}, \mathbf{x}')\pi(\mathbf{x})d\mathbf{x}. \end{aligned}$$

Roughly speaking, MCMC algorithms turn the Markov chain theory around. The invariant distribution Q is assumed to be known which corresponds to the target density $\pi(\mathbf{x})$, but the transition kernel is unknown. MCMC methods produce Markov chains that are *aperiodic, irreducible* and fulfill the *reversible condition*; the samples are generated by a *homogeneous, reversible, ergodic* Markov chain with an invariant distribution. In order to generate samples from $\pi(\cdot)$, the MCMC methods attempt to find a transition kernel $K(\mathbf{x}; d\mathbf{x}')$ that asymptotically leads to $\pi(\cdot)$, given an arbitrary starting point.

Metropolis-Hastings Algorithm. The Metropolis-Hastings algorithm (Metropolis et al., 1953; Hastings, 1970) is the first Monte Carlo algorithm; it is "among the top 10 scientific algorithms of the 20th century" according to the January/February 2000 issue of *Computing in Science & Engineering*. The algorithm is named after one of the inventors, Nick Metropolis, dated back in Manhattan project at the Los Alamos Laboratory during the World World II; it was generalized by Hastings in 1970.

The underlying idea of the Metropolis algorithm is simple: Assume that $q(\mathbf{x}, \mathbf{x}')$ is the proposal distribution that does not satisfy the reversibility condition. Without loss of generality, suppose $\pi(\mathbf{x})q(\mathbf{x}, \mathbf{x}') > \pi(\mathbf{x}')q(\mathbf{x}', \mathbf{x})$, which means the probability moving from $\mathbf{x}$ to $\mathbf{x}'$ is greater (more frequent) than the probability moving from $\mathbf{x}'$ to $\mathbf{x}$. Intuitively, we want to change this situation to reduce the number of moves from $\mathbf{x}$ to $\mathbf{x}'$. In doing so, we introduce a *probability of move*, $0 < \alpha(\mathbf{x}, \mathbf{x}') < 1$; if the move is not performed, the process returns $\mathbf{x}$ as a value from the target distribution.

[1]We assume that the state $\mathbf{x}$ is continuous-valued; for discrete-valued state, the integration should be substituted by the sum operation.

Now, the transition probability from $\mathbf{x}$ to $\mathbf{x}'$ becomes:

$$\begin{aligned} p_{\mathrm{MH}}(\mathbf{x},\mathbf{x}') &= q(\mathbf{x},\mathbf{x}')\alpha(\mathbf{x},\mathbf{x}'), && \text{(B.1)} \\ \alpha(\mathbf{x},\mathbf{x}') &= \begin{cases} \min\left[\frac{\pi(\mathbf{x}')q(\mathbf{x}',\mathbf{x})}{\pi(\mathbf{x})q(\mathbf{x},\mathbf{x}')}, 1\right], & \text{if } \pi(\mathbf{x})q(\mathbf{x},\mathbf{x}') > 0, \\ 1 & \text{otherwise} \end{cases} && \text{(B.2)} \end{aligned}$$

Thus, the probability that the Markov process stays at $\mathbf{x}$ can be written as:

$$1 - \int_X q(\mathbf{x},\mathbf{x}')\alpha(\mathbf{x},\mathbf{x}')d\mathbf{x}', \tag{B.3}$$

and the transition kernel is given by

$$K_{\mathrm{MH}}(\mathbf{x}, d\mathbf{x}') = q(\mathbf{x},\mathbf{x}')\alpha(\mathbf{x},\mathbf{x}')d\mathbf{x}' + \left[1 - \int_X q(\mathbf{x},\mathbf{x}')\alpha(\mathbf{x},\mathbf{x}')d\mathbf{x}'\right]\delta_{\mathbf{x}}(d\mathbf{x}'). \tag{B.4}$$

Remarks:

- If $q(\mathbf{x},\mathbf{x}') = q(\mathbf{x}',\mathbf{x})$ (symmetric), the probability of move only depends on the ratio $\pi(\mathbf{x}')/\pi(\mathbf{x})$ (Metropolis algorithm), without the need of the normalizing constant.

- Design of $\alpha(\mathbf{x},\mathbf{x}')$ is not unique and is crucial to the MCMC algorithmic efficiency. For instance, the probability of move in (B.2) can be alternatively defined by $\alpha(\mathbf{x},\mathbf{x}') = \frac{\pi(\mathbf{x}')q(\mathbf{x}',\mathbf{x})}{\pi(\mathbf{x}')q(\mathbf{x}',\mathbf{x})+\pi(\mathbf{x})q(\mathbf{x},\mathbf{x}')}$, which also satisfies the detailed balance condition.

- If $q(\cdot,\cdot)$ is independent of current sample value, it reduces to a *Metropolized independence sampler* (Hastings, 1970).

- The draws are regarded as the samples from the target density only after the chain has passed the transient phase; in practice, the convergence of Markov chains often needs diagnosis, although theoretically it is guaranteed to converge under mild regularity conditions.

- The efficiency of Metropolis algorithm is determined by the ratio of the accepted samples to the total number of samples. Too large or too small variance of the proposal distribution may cause inefficient sampling.

B.2 MCMC Algorithms for Machine Learning

In recent decades, thanks to the ever-increasing computational power, MCMC algorithms have been increasingly popular and powerful for machine learning. We briefly mention a few of them in the below. See (Andrieu et al., 2003) for an excellent tutorial.

- Simulated annealing (Kirkpatrick et al., 1983) is one of the first efforts to use MCMC scheme for optimization. Without loss of generality, suppose the goal is to find the optimal $\mathbf{x}^*$ that achieves the lowest potential energy $E(\mathbf{x})$; also assume that the posterior is of an exponential form:

$$p(\mathbf{x}) = \frac{1}{Z} \exp[-E(\mathbf{x})/kT]. \tag{B.5}$$

In the equilibrium state, $p(\mathbf{x})$ satisfies the Boltzmann-Gibbs distribution (where k denotes the Boltzmann constant, and T denotes the temperature). Note that the probability ratio

$$\frac{p(\mathbf{x}')}{p(\mathbf{x})} = \exp\Big[- \frac{E(\mathbf{x}') - E(\mathbf{x})}{T} \Big] = \exp[-\Delta E(\mathbf{x})/kT]$$

is independent on the partition function Z. The procedure is summarized as follows:

1. Starting with initial point $\mathbf{x}_0$ and initial temperature T_0;
2. For $t = 1, 2, \cdots$, move $\mathbf{x}_t$ to $\mathbf{x}_{t+1}$;
3. Draw $u \sim \mathcal{U}(0, 1)$;
4. If $u < \exp[-(E(\mathbf{x}_{t+1}) - E(\mathbf{x}_t))/T_t]$, accept the move and set $\mathbf{x}_{t+1} = \mathbf{x}_{t+1}$, otherwise keep the present value: $\mathbf{x}_{t+1} = \mathbf{x}_t$ (or add some random noise);
5. Reduce the temperature T, repeat the Step 2 through 5.

- Gibbs sampling (Geman and Geman, 1984) can be viewed as a Metropolis method in which the proposal distribution is defined in terms of the conditional distribution of the joint distribution and every proposal is always accepted. Gibbs sampler uses the concept of alternating (marginal) conditional sampling. Suppose $\mathbf{x} = [x_1, x_2, \cdots, x_N]^T$, the sampling procedure runs as follows:

1. At iteration $t = 0$, draw $\mathbf{x}_0$ from the prior distribution $p(\mathbf{x}_0)$;
2. At iterations $t = 1, 2, \cdots$, draw $x_{1,t}$ from $p(x_1|x_{2,t-1}, x_{3,t-1}, \cdots, x_{N,t-1})$;
3. For $k = 2, \cdots, N-1$, draw $x_{k,n}$ from $p(x_k|x_{1,t}, \cdots, x_{k-1,t-1}, \cdots, x_{k+1,t-1}, \cdots, x_{N,t-1})$;
4. Draw $x_{N,t}$ from $p(x_N|x_{1,t}, x_{2,t}, \cdots, x_{N-1,t})$.

The above sampling order is systematic; however, it can also be replaced by a random scan order. From the missing data perspective, there is a close connection between Gibbs sampling and EM algorithm in exponential family models (Hastie et al., 2001).

- Data augmentation was first proposed by Dempster et al. (1977) in a deterministic framework for the EM algorithm, and later generalized by Tanner and Wong (1987) for posterior distribution estimation in a stochastic framework, which can be viewed as a *Rao-Blackwellization* (Robert and Casella, 1999) of the marginal density. It can be also viewed as a two-component Gibbs sampling, which performs a Monte Carlo E-step for the EM algorithm. See (van Dyk and Meng, 2001) and the references therein for more details.

- Dynamic weighting (Wong and Liang, 1997) can be viewed as a data augmentation approach for simulating Markov chain, in that it introduces importance weights into dynamic Monte Carlo process to provide a means for the system to make *large* transitions not allowable by the standard Metropolis transition rule. It has been successfully used for Monte Carlo sampling and global optimization of several NP-hard problems.

- Hybrid Monte Carlo (Duane et al., 1987) method augments the state space $\mathbf{x}$ with a *momentum variable* $\boldsymbol{\rho}$ and uses two proposals for drawing samples. The first proposal randomizes the momentum variable with the state $\mathbf{x}$ unchanged; the second proposal changes both $\mathbf{x}$ and $\boldsymbol{\rho}$ using the simulated Hamilton dynamics. In practice, one has to use *leapfrog* discretization for simulating the dynamics (see below).

B.3 Hybrid Monte Carlo Simulation

In the hybrid Monte Carlo (HMC) simulation, the original state space $\mathbf{x} \in \mathbb{R}^N$ is augmented with a momentum variable $\boldsymbol{\rho} \in \mathbb{R}^N$, with an associated Hamiltonian dynamics:

$$\mathcal{H}(\mathbf{x}, \boldsymbol{\rho}) = E(\mathbf{x}) + \mathcal{K}(\boldsymbol{\rho}), \tag{B.6}$$

where $E(\mathbf{x})$ is the *potential energy* function, whereas $\mathcal{K}(\boldsymbol{\rho}) = \frac{1}{2}\boldsymbol{\rho}^T\boldsymbol{\rho}$ is called the *kinetic energy*.[2] The samples $\{\mathbf{x}\}$ are then drawn from the joint distribution:

$$P_{\mathcal{H}}(\mathbf{x}, \boldsymbol{\rho}) = \frac{1}{Z}\exp(-\mathcal{H}(\mathbf{x}, \boldsymbol{\rho})) = \frac{1}{Z}\exp(-E(\mathbf{x}))\exp(-\mathcal{K}(\boldsymbol{\rho})), \tag{B.7}$$

where Z is a normalizing constant. The ideal (continuous-time) Hamiltonian dynamics

$$\frac{\partial \mathcal{H}}{\partial \mathbf{x}} = -\dot{\boldsymbol{\rho}}, \quad \frac{\partial \mathcal{H}}{\partial \boldsymbol{\rho}} = \dot{\mathbf{x}}, \quad \dot{\mathbf{x}} = \boldsymbol{\rho} \tag{B.8}$$

is *time-reversible, volume-preserving,* and *energy-preserving* (i.e., $\frac{d\mathcal{H}}{dt} = 0$), and the resulting moves leave the posterior invariant. Namely, given a starting point $(\mathbf{x}^{(0)}, \boldsymbol{\rho}^{(0)}) \sim P$, then after t steps of Hamiltonian dynamics evolution, the new configuration $(\mathbf{x}^{(t)}, \boldsymbol{\rho}^{(t)})$ also follows the probability distribution P. In computer simulations, it was common (Duane et al., 1987; Neal, 1996; Mackay, 1998; MacKay, 2003) to use a discretized leapfrog step to simulate the Hamiltonian dynamics:

$$\begin{aligned} \mathbf{x}(t+\tau) &= \mathbf{x}(t) + \tau\boldsymbol{\rho}(t+\tau/2) \qquad &\text{(B.9)} \\ \boldsymbol{\rho}(t+\tau/2) &= \boldsymbol{\rho}(t-\tau/2) - \tau\frac{\partial E(\mathbf{x})}{\partial \mathbf{x}}\Big|_{\mathbf{x}(t)} \qquad &\text{(B.10)} \end{aligned}$$

[2]An alternative definition is $\mathcal{K}(\boldsymbol{\rho}) = \sum_{j=1}^{N} \frac{\rho_j^2}{2m_j}$, where $\rho_j (j = 1, \cdots, N)$ is the momentum component and m_j is the "mass" associated with the jth element of $\mathbf{x}$.

where τ is a small step-size parameter. The step-size controls the sampling efficiency: if it is too small, it takes a long time to converge to the equilibrium state; if it is too large, the dynamics becomes unstable and causes a high rejection rate. The discreteized leapfrog lasts L steps and produces a new configuration $(\mathbf{x}^{(L)}, \boldsymbol{\rho}^{(L)})$. When $L = 1$, the HMC method reduces to the Langevin algorithm, being an approximation to the Langevin diffusion process. Each leapfrog move within the HMC per iteration remains time-reversible and volume-preserving, but the energy $\mathcal{H}$ is no longer a constant; therefore, Duane et al. (1987) suggested to run a Metropolis step to correct the discrepancy.

In order to improve the efficiency of HMC sampling, it is advised that individual momentum components ρ_j use different step-size parameters for exploration; this is tantamount to replacing a uniform step-size scalar with a diagonal step-size matrix for the momentum vector $\boldsymbol{\rho}$. In the general case, (B.6) is rewritten as $\mathcal{H} = E(\mathbf{x}) + \frac{1}{2}\boldsymbol{\rho}^T\mathbf{H}\boldsymbol{\rho}$, where $\mathbf{H}$ is a scaled version of the Hessian matrix. Though theoretically appealing, this scheme does not scale well for the large systems.

Horowitz (1991) also suggested a *momentum persistence* approach to improve the rate of exploration in the HMC; in which he replaced $\boldsymbol{\rho}$ wherever with $\boldsymbol{\rho}\cos\omega + \boldsymbol{\nu}\sin\omega$, where $\boldsymbol{\nu} \sim \mathcal{N}(0, \mathbf{I})$ is a standard Gaussian distributed vector variable, and ω is set as a small angle (or equivalently, $\boldsymbol{\rho}$ is replaced by $\alpha\boldsymbol{\rho} + \sqrt{1-\alpha^2}\boldsymbol{\nu}$ where $0 < \alpha \leq 1$).

Bibliography

L. F. Abbott and S. Song. Temporally asymmetric Hebbian learning, spike timing and neural response variability. In M. Kearns, S. Solla, and D. Cohn, editors, *Advances in Neural Information Processing Systems, 11*, pages 69–75, Cambridge, MA, 1999. MIT Press.

D. H. Ackley, G. E. Hinton, and T. J. Sejnowski. A learning algorithm for boltzmann machines. *Cognitive Science*, 9:147–169, 1985.

S. Alamouti. A simple transmit diversity technique for wireless communications. *IEEE Journal on Selected Areas in Communications*, 16:1451–1458, 1998.

J-M. Alonso, W. M. Usrey, and R. C. Reid. Precisely correlated firing in cells of the lateral geniculate nucleus. *Nature*, 383:815–819, 1996.

S. Amari, A. Cichocki, and H. H. Yang. A new learning algorithm for blind signal separation. In G. Tesauro, D. S. Touretzky, and T. K. Leen, editors, *Advances in Neural Information Processing Systems, 8*, pages 757–763, Cambridge, MA, 1996. MIT Press.

B. D. O. Anderson and J. B. Moore. *Optimal Filtering.* Prentice Hall, Englewood Cliffs, NJ, 1979.

J. Anderson and E. Rosenfeld, editors. *Neurocompting: Foundations of Research.* MIT Press, Cambridge, MA, 1988.

J. A. Anderson. A memory storage model utilizing spatial correlation functions. *Kybernetik*, 5(3):113–119, 1969.

J. A. Anderson. A simple neural network generating an interactive memory. *Mathematical Biosciences*, 14:197–220, 1972.

M. J. Anderson and E. Tzanakou. Auditory stimulus optimization with feedback from fuzzy clustering of neuronal responses. *IEEE Trans. Info. Tech. Biomed.*, 6(2):159–169, 2002.

C. Andrieu, N. de Freitas, A. Doucet, and Jordan M. I. An introduction to MCMC for machine learning. *Machine Learning*, 50:5–43, 2003.

S. M. Anstis. The perception of apparent movement. *Philosophical Transactions of the Royal Society, B*, 290:153–168, 1980.

M. Arbib, editor. *The Handbook of Brain Theory and Neural Networks.* MIT Press, Cambridge, MA, 2nd edition, 2003.

M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

J. J. Atick. Could information theory provide an ecological theory of sensory processing? *Network*, 3:213–251, 1992.

H. Attias. A variational Bayesian framework for graphical models. In S. A. Solla, T. K. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems, 12*, pages 201–215, Cambridge, MA, 2000. MIT Press.

D. H. Ballard, G. E. Hinton, and T. J. Sejnowski. Parallel visual computation. *Nature*, 306:21–26, November 1983.

H. B. Barlow. Unsupervised learning. *Neural Computation*, 1:295–311, 1989.

T. R. Bayes. Essay towards solving a problem in the doctrine of chances. *Phil. Trans. R. Soc. Lond.*, 53:370–418, 1763.

S. Becker. Unsupervised learning with global objective functions. In M. Arbib, editor, *Handbook of Brain Theory and Neural Networks*, pages 997–1001, Cambridge, MA, 1995. MIT Press.

S. Becker and I. C. Bruce. Neural coding in the auditory periphery: insights from physiology and modeling lead to a novel hearing compensation algorithm. In *Workshop in Neural Information Coding*, Les Houches, France, 2002.

S. Becker and G. E. Hinton. A self-organizing neural network that discovers surfaces in random-dot stereograms. *Nature*, 355:161–163, January 1992.

A. Bell. Levels and loops: The future of artificial intelligence and neuroscience. *Philosophical Transactions of The Royal Society, B*, 354:2013–2020, 1999.

A. Bell and T. J. Sejnowski. An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7:1129–1159, 1995.

J. M. Bernardo and A. F. M. Smith. *Bayesian Theory.* Wiley, New York, 2nd edition, 1998.

C. Berrou and A. Glavieux. Near optimum error correcting coding and decoding: Turbo codes. *IEEE Transactions on Communication*, 44:1261–1271, 1996.

D. P. Bertsekas. *Nonlinear Programming.* Athena Scientific, Belmont, MA, 2nd edition, 1999.

D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming.* Athena Scientific, Belmont, MA, 1996.

G. Bi and M. Poo. Synaptic modification of correlated activity: Hebb's postulate revisited. *Ann. Rev. Neuroscience*, 24:139–166, 2001.

A. Bia. Alopex-B: A new, simple, but yet faster version of the Alopex training algorithm. *International Journal of Neural Systems*, 11(6):497–507, 2001.

E. L. Bienenstock, L. N. Cooper, and P. W. Munro. Theory for the development of neuron selectivity: orientation specificity and binocular interaction in visual cortex. *Journal of Neuroscience*, 2:32–48, 1982.

C. M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, 1995.

F. Black and M. Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81:637–659, 1973.

K. Blackard, T. Rappaport, and C. Bostian. Measurements and models of radio frequency impulsive noise for indoor wireless communications. *IEEE Journal on Selected Areas in Communications*, 11(7):991–1001, September 1993.

B. S. Blais, N. Intractor, H. Shouval, and L. N. Cooper. Receptive field formation in natural scene environments: comparison of single cell learning rules. *Neural Computation*, 10:1797–1813, 1998.

T. V. P. Bliss and T. Lomo. Long-lasting potentation of synaptic transmission in the dendate area of anaesthetized rabbit following stimulation of the prefrant path. *Journal of Physiology*, 232:551–556, 1973.

J. Bondy, S. Becker, I. Bruce, L. Trainor, and S. Haykin. A novel signal-processing strategy for hearing-aid design: neurocompensation. *Signal Processing*, 84:1239–1253, 2004.

J. Bondy, Bruce I., R. Dong, S. Becker, and S. Haykin. Modeling intelligibility of hearing-aid compresion circuits. In *Proceedings of 37th Asilomar Conference on Signals, Systems, and Computers*, pages 720–724, 2003.

A. S. Bregman. *Auditory Scene Analysis.* MIT Press, Cambridge, MA, 1990.

T. Briegel and V. Tresp. Fisher scoring and a mixture of modes approach for approximate inference and learning in nonlinear state space models. In *Advances in Neural Information Processing Systems, 11*, page M. Kearns and S. Solla and D. Cohn, Cambridge, MA, 1999. MIT Press.

A. E. Brockwell, A. L. Rojas, and R. E. Kass. Recursive Bayesian decoding of motor cortical signals by particle filtering. *Journal of Neurophysiology*, 91(4):1899–1907, 2004.

I. C. Bruce, M. B. Sachs, and E. Young. An auditory-periphery model of the effects of acoustic trauma on auditory nerve responses. *Journal of Acoustical Soceity of America*, 113(1):369–388, 2003.

R. S. Bucy and K. D. Senne. Digital synthesis of non-linear filters. *Automatica*, 7:287–298, June 1971.

W. Byrne, A. Parkinson, and P. Newall. Hearing aid gain and frequency response requirements for the severely/profoundly hearing impaired. *Ear and Hearing*, 11:40–49, 1990.

J.-F. Cardoso. Blind signal separation: statistical principles. *Proceedings of the IEEE*, 86 (10):2029–2025, October 1998.

J.-F. Cardoso. High-order contrasts for independent component analysis. *Neural Computation*, 11(1):157–192, 1999.

J. Carpenter, P. Clifford, and P. Fearnhead. Improved particle filter for nonlinear problems. *IEE Proceedings - F*, 146(1):2–7, 1999.

R. Chen and J. S. Liu. Mixture Kalman filters. *Journal of Royal Statistical Society (B)*, 62:493–508, 2000.

Z. Chen. Bayesian filtering: From Kalman filters to particle filters, and beyond. Tech. Rep., Adaptive Systems Lab, McMaster University, Feburary 2003a.

Z. Chen. An odyssey of the cocktail party problem. Tech. Rep., Adaptive Systems Lab, McMaster University, July 2003b.

Z. Chen and S. Haykin. On different facets of regularization theory. *Neural Computation*, 14(12):2791–2846, 2002.

E. C. Cherry. Some experiments on the recognition of speech, with one and two ears. *Journal of the Acoustical of Society of America*, 25:975–979, 1953.

P. Comon. Independent component analysis, a new concept? *Signal Processing*, 36: 287–314, 1994.

J. E. Cook. Correlated activity in the CNS: a role on every timescale? *Trends in Neuroscience*, 14:397–401, 1991.

L. N. Cooper. A possible organization of animal memory and learning. In B. Lundqvist and S. Lundqvist, editors, *Collective Properties of Physical Systems*, pages 252–264, New York, 1973. Academic Press.

T. M. Cover and J. A. Thomas. *Elements of Information Theory.* Wiley, New York, 1991.

A. I. Dale. *A History of Inverse Probability: From Thomas Bayes to Karl Pearson.* Springer-Verlag, New York, 1991.

P. Dayan and L. F. Abbott. *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems.* MIT Press, Cambridge, MA, 2001.

P. Dayan, G. E. Hinton, R. Neal, and R. S. Zemel. The Helmholtz machine. *Neural Computation*, 7:1022–1037, 1995.

J. F. G. deFreitas. *Bayesian methods for neural networks.* PhD thesis, Engineering Department, Cambridge University, 1999.

J. F. G. deFreitas, M. Niranjan, A. H. Gee, and A. Doucet. Sequential Monte Carlo methods to train neural network models. *Neural Computation*, 12(4):955–993, 2000.

A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm (with discussions). *Journal of the Royal Statistical Society, Series B*, 39: 1–38, 1977.

A. Doucet, N. de Freitas, and N. Gordon, editors. *Sequential Monte Carlo Methods in Practice.* Springer, New York, 2001.

A. Doucet, S. Godsill, and C. Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10:197–208, 2000.

R. Drullman, J. M. Festen, and R. Plomp. Effect of reducing slow temporal modulations on speech reception. *Journal of Acoustical Society of America*, 95(5):2670–2680, May 1994.

S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. Hybrid Monte Carlo. *Physics Letter B*, 55:2774–2777, 1987.

R. O Duda, P. E. Hart, and D. G. Stork. *Pattern Classification.* Wiley, New York, 2nd edition, 2001.

J. J. Eggermont. *The Correlative Brain: Theory and Experiment in Neural Interaction.* Springer-Verlag, New York, 1990.

J. J. Eggermont. Functional aspects of synchrony and correlation in the auditory nervous system. *Concepts in Neuroscience*, 4(2):105–129, 1993.

J. L. Elman. Finding structure in time. *Cognitive Science*, 14:179–211, 1990.

Y. Ephraim and N. Merhav. Hidden Markov processes. *IEEE Transactions on Information Theory*, 48(6), 2002.

D. Erdogmus. *Information theoretic learning: Renyi's entropy and its applications to adaptive systems training.* PhD thesis, Department of Electrical Engineering, University of Florida, 2002.

D. Erdogmus and J. C. Principe. An entropy minimization algorithm for supervised training of nonlinear systems. *IEEE Transactions on Signal Processing*, 50(7):1780–1786, July 2002a.

D. Erdogmus and J. C. Principe. Generalized information potential criterion for adaptive systems training. *IEEE Transactions on Neural Networks*, 13(5):1035–1044, September 2002b.

M. Fahle and T. Poggio, editors. *Perceptual Learning.* MIT Press, Cambridge, MA, 2002.

S. E. Fahlman and C. Lebiere. The cascade-correlation learning architecture. In D. Touretzky, editor, *Advances in Neural Information Processing Systems, 2*, pages 524–532, San Mateo, CA, 1989. Morgan Kaufmann.

H. Farid. Temporal synchrony in perceptual grouping: a critique. *Trends in Cognitive Sciences*, 6(7):284–288, 2002.

P. Fearnhead. MCMC, sufficient statistics, and particle filters. *Journal of Computational and Graphical Statistics*, 11:846–862, 2002.

D. J. Field. What is the goal of sensory coding? *Neural Computation*, 6:559–601, 1994.

R. Fletcher. *Practical Methods of Optimization.* Wiley, New York, 2nd edition, 2000.

J. B. Fraleigh and R. A. Beauregard. *Linear Algebra.* Addison-Wesley, New York, 2nd edition, 1990.

J. H. Freidman. Exploratory projection pursuit. *Journal of the American Statistical Association*, 82:249–266, 1987.

O. Fujita. Trial-and-error correlation learning. *IEEE Transactions on Neural Networks*, 4(4):720–722, 1993.

S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

W. Gerstner. Coding properties of spiking neurons: reverse and cross-correlations. *Neural Networks*, 14:559–610, 2001.

W. Gerstner and W. M. Kistler. Mathematical formulations of Hebbian learning. *Biological Cybernetics*, 87:404–415, 2002.

W. R. Gilks and C. Berzuini. Following a moving target – Monte Carlo inference for dynamic Bayesian models. *Journal of Royal Statistical Society, Series B*, 63:127–146, 2001.

W. R. Gilks, S. Richardson, and D. J. Spiegelhalter, editors. *Markov Chain Monte Carlo Methods in Practice.* Chapman & Hall, London, 1996.

M. Girolami and C. Fyfe. An extended exploratory projection pursuit network with linear and nonlinear anti-Hebbian lateral connections applied to the cocktail party problem. *Neural Networks*, 10(9):1607–1618, 1997.

P. W. Glimcher. *Decisions, Uncertainty, and the Brain: The Science of Neuroeconomics.* MIT Press, Cambridge, MA, 2003.

D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, Reading, MA, 1989.

N. Gordon, D. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-gaussian Bayesian state estimation. *IEE Proceedings-F*, 140:107–113, 1993.

S. Grossberg. Adaptive pattern classification and universal recoding: I. parallel development and coding of neural feature detectors. *Biological Cybernetics*, 23:121–134, 1976.

S. Grossberg and L. Wyse. A neural network architecture for figure-ground separation of connected scenic figures. *Neural Networks*, 4:723–742, 1991.

S. Grossberg and L. Wyse. Figure-ground separation of connected scenic figures: boundaries, filling in and opponent processing. In G. A. Carpenter and S. Grossberg, editors, *Neural Networks for Vision and Image Processing*, Cambridge, MA, 1992. MIT Press.

S. Gupta. Efficient testing of the Neurocompensator through the development of an unsupervised learning clustering algorithm and an adaptive psychometric function. Bachelor Thesis, Department of Psychology, McMaster University, 2004.

J. L. Hamilton, E. Tzanakou, and R. M. Lehman. Neural networks trained with simulation data for outcome prediction in pallidotomy for parkinson's disease. In *Proc. 22nd IEEE Engr. Med. Biol.*, pages 1–4, New York, 2000. IEEE Press.

J. E. Handschin. Monte Carlo techniques for prediction and filtering of non-linear stochastic processes. *Automatica*, 6:555–563, 1970.

J. E. Handschin and D. Q. Mayne. Monte Carlo techniques to estimate conditional expectation in multi-state non-linear filtering. *International Journal of Control*, 9(5): 547–559, 1969.

E. Harth. Visual perception: a dynamic theory. *Biological Cybernetics*, 22:169–180, 1976.

E. Harth, T. Kalogeropoulos, and A. S. Pandya. A universal optimization network. In *Proceedings of Symp. Maturing Technology and Emerging Horizons in Biomedical Engineering*, pages 97–107, 1988.

E. Harth and E. Tzanakou. Alopex: A stochastic method for determining visual receptive fields. *Vision Research*, 14:1475–1482, 1974.

E. Harth, K. P. Unnikrishnan, and A. S. Pandya. Perception as an optimization process. In *Proceedings of IEEE Conf. Computer Vision and Pattern Recognition*, pages 662–665, 1986.

E. Harth, K. P. Unnikrishnan, and A. S. Pandya. The inversion of sensory processing by feedback pathways: A model of visual cognitive functions. *Science*, 237:184–187, 1987.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Stastical Learning: Data Mining, Inference, and Prediction.* Springer, New York, 2001.

W. Hastings. Monte carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:99–109, 1970.

S. Haykin. *Neural Networks: A Comprehensive Foundation.* Prentice Hall, Upper Saddle River, NJ, 2nd edition, 1999.

S. Haykin, editor. *Unsupervised Adaptive Filtering*, volume I,II. Wiley-Interscience, New York, 2000.

S. Haykin. *Adaptive Filter Theory.* Prentice Hall, Upper Saddle River, NJ, 4th edition, 2002.

S. Haykin, K. Huber, and Z. Chen. Bayesian sequential state estimation for mimo wireless communications. *Proceedings of the IEEE*, 92(3):439–453, March 2004.

S. Haykin and M. Moher. *Modern Wireless Communications.* Prentice Hall, Upper Saddle River, NJ, 2004.

S. Haykin and J. Principe. Making sense of a complex world. *IEEE Signal Processing Magazine*, 15(3):66–81, May 1998.

D. Hebb. *Organization of Behavior: A Neuropsychological Theory.* Wiley, New York, 1949.

L. Hérault and R. Horaud. Figure-ground discrimination: A combinatorial optimization approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9): 899–914, 1993.

G. E. Hinton. Connectionist learning procedure. *Artificial Intelligence*, 40:185–234, 1989.

G. E. Hinton. Training products of experts by minimizing contrastive divergence. Technical Report, GCNU TR 2000-004, Gatsby Computational Neuroscience Unit, University College London, 2000.

G. E. Hinton, P. Dayan, R. Frey, and R. Neal. The "wake-sleep" algorithm for unsupervised neural networks. *Science*, 268:1158–1161, May 1995.

G. E. Hinton and T. Sejnowski, editors. *Unsupervised Learning: Foundations of Neural Computation.* MIT Press, Cambridge, MA, 1999.

G. E. Hinton and T. J. Sejnowski. Optimal perceptual learning. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 448–453, Washington, DC, 1983.

G. E. Hinton and T. J. Sejnowski. Learning and relearning in boltzmann machines. In D. Rumelhart and J. McClelland, editors, *Parallel Distributed Processing: Explorations in the Microstructure Cognition*, volume 1, pages 282–317, Cambridge, MA, 1986. MIT Press.

J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of National Academy of Science, USA*, 79:2554–2558, July 1982.

A. M. Horowitz. A generalized guided Monte Carlo algorithm. *Physics Letters B*, 268: 247–252, 1991.

J. M. Hutchinson. *A radial basis function approach to financial time series analsyis.* PhD thesis, Department of Electrical Engineering and Computer Science, MIT, 1994.

J. M. Hutchinson, A. W. Lo, and T. Poggio. A nonparametric approach to pricing and hedging derivative securities via learning networks. *Journal of Finance*, 49(3):851–889, 1994.

A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis.* Wiley, New York, 2001.

A. Hyvärinen and E. Oja. A fast fixed-point algorithm for independent component analysis. *Neural Computation*, 9:1483–1492, 1997.

N. Intrator. Feature extraction using an unsupervised neural networks. *Neural Computation*, 4:98–107, 1992.

M. Isard and A. Blake. CONDENSATION: conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.

H. Jaeger. The "echo state" approach to analysing and training recurrent neural networks. Tech. Rep., GMD report 148, German National Research Center for Information Technology, 2001.

H. Jaeger and H. Haas. Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. *Science*, 304:78–80, April 2004.

W. C. Jakes, editor. *Microwave Mobile Communications.* Wiley, New York, 1974.

W. James. *Psychology (Briefer Course).* Holt, New York, 1890.

J. Janakiraman and K. P. Unnikrishnan. A feedback model of visual attention. In *Proc. IJCNN'92*, pages 541–546, 1992.

R. Johansson. *System Modeling and Identification.* Prentice Hall, Englewood Cliffs, NJ, 1993.

M. I. Jordan. Serial order: a parallel distributed processing approach. In J. L. Elman and D. E. Rumelhart, editors, *Advances in Connectionist Theory*, Hillsdale, NJ, 1989. Erlbaum.

B. Julesz. Binocular depth perception of computer-generated patterns. *Bell Systems Technical Journal*, 39:1125–1162, 1960.

B. Julesz. *Foundations of Cyclopean Perception.* University of Chicago Press, Chicago, IL, 1971.

B. Julesz and G. J. Spivack. Stereopsis based on vernier acuity cues alone. *Science*, 157: 563–565, 1967.

B. Julesz and C. W. Tyler. Neurontropy, an entropy-like measure of neural correlation, in binocular fusion and rivalry. *Biological Cybernetics*, 23:25–32, 1976.

S. Julier and J. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, 2004.

S. Julier, J. Uhlmann, and H. F. Durrant-Whyte. A new method for nonlinear transformation of means and covariances in filters and estimators. *IEEE Transactions on Automatic Control*, 45(3):477–482, 2000.

R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME, Journal of Basic Engineering*, 82:35–45, March 1960.

R. Kempter, W. Gerstner, and J. L. van Hemmen. Hebbian learning and spiking neurons. *Physical Review E*, 59(4):4498–4514, 1999.

A. I. Khinchin. Korrelationstheorie der statistischen stochcstischen prozesse. *Mathem. Annalen*, 109:604–615, 1933.

K. Kienker, P, T. J. Sejnowski, G. E. Hinton, and L. E. Schumacher. Separating figure from ground with a parallel network. *Perception*, 15:197–216, 1986.

S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, May 1983.

G. Kitagawa. Monte Carlo filter and smoother for non-gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5(1):1–25, 1996.

D. Knills and W. Richard, editors. *Perception as Bayesian Inference.* Cambridge University Press, UK, 1995.

K. Koffka. *The Task of Gestalt Psychology.* Princeton University Press, Princeton, NJ, 1969.

T. Kohonen. Correlation matrix memories. *IEEE Transactions on Computers*, 21:353–359, 1972.

T. Kohonen. *Self-organization and Associatie Memory.* Springer, Berlin, 1984.

T. Kohonen. *Self-organizing Maps.* Springer, Berlin, 3rd edition, 2001.

A. Kong, J. S. Liu, and W. H. Wong. Sequential imputations and Bayesian missing data problems. *Journal of American Statistical Association*, 89:278–288, 1994.

P. König and A. K. Engel. Correlated firing in sensory-motor systems. *Current Opinions in Neurobiology*, 5:511–519, 1995.

P. König, A. K. Engel, and W. Singer. Integrator or coincidence detector? the role of the cortical neuron revisited. *Trends in Neuroscience*, 19:130–137, 1996.

J. H. Kotecha and P. M. Djurić. Gaussian particle filtering. *IEEE Transactions on Signal Processing*, 51(10):2592–2601, March 2003a.

J. H. Kotecha and P. M. Djurić. Gaussian sum particle filtering. *IEEE Transactions on Signal Processing*, 51(10):2602–2612, March 2003b.

S. C. Kramer and H. W. Sorenson. Recursive Bayesian estimation using piece-wise constant approximations. *Automatica*, 24:789–901, 1988.

H. J. Kushner. On the differential equations satisfied by conditional probability densities of Markov processes with applications. *Journal of SIAM Control*, 2:106–119, 1965.

K. J. Lang and M. J. Witbrock. Learning to tell two spirals apart. In D. Touretzky, G. Hinton, and T. Sejnowski, editors, *Proceedings of the 1988 Connectionist Models Summer School*, San Mateo, CA, 1989. Morgan Kaufmann.

F. H. Lange. *Correlation Techniques: Foundations and Applications of Correlation Analysis in Modern Communications, Measurement and Control.* Van Nostrand, Princeton, NJ, 1967.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2323, November 1998.

T. S. Lee and D. Mumford. Hierarchical Bayesian inference in the visual cortex. *Journal of Optical Society of America, A*, 20(7):1434–1448, 2003.

T. M. Liggett. *Interacting Particle Systems.* Springer-Verlag, New York, 1985.

R. Linsker. Self-organization in a perceptual system: How network models and information theory may shed light on neural organization. In S. J. Hanson and C. R. Olson, editors, *Connectionist Modeling and Brain Function: The Developing Interface*, pages 351–392, Cambridge, MA, 1990. MIT Press.

R. Linsker. A local learning rule that enables information maximization for arbitrary input distributions. *Neural Computation*, 9:1661–1665, 1997.

J. S. Liu and R. Chen. Blind deconvolution via sequential imputation. *Journal of American Statistical Association*, 90:567–576, 1995.

J. S. Liu and R. Chen. Sequential monte carlo methods for dynamical systems. *Journal of American Statistical Association*, 93:1032–1044, 1998.

L. Ljung. *System Indentification: Theory for the User.* Prentice Hall, Englewood Cliffs, NJ, 2nd edition, 1999.

S. N. MacEachern, M. Clyde, and J. S. Liu. Sequential importance sampling for nonparametric Bayes models: The next generation. *Canadian Journal of Statistics*, 27:251–267, 1999.

D. J. C. Mackay. Introduction to Monte Carlo methods. In M. I. Jordan, editor, *Learning in Graphical Models.* Kluwer Academic Publishers, 1998.

D. J. C. MacKay. *Information Theory, Inference, and Learning Algorithms.* Cambridge University Press, UK, 2003.

D. Marr, G. Palm, and T. Poggio. Analysis of a cooperative stereo algorithm. *Biological Cybernetics*, 28:223–239, 1978.

D. Marr and T. Poggio. Cooperative computation of stereo disparity. *Science*, 194:283–287, 1976.

G. L. McLachlan and T. Krishnan. *The EM Algorithm and Extensions.* John Wiley & Sons, New York, 1997.

N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21: 1087–1091, March 1953.

K. D. Miller. Correlation-based models of neural development. In M. Gluck and D. Rumelhart, editors, *Neuroscience and Connectionist Theory*, pages 267–353, Hilsdale, NJ, 1990. Lawrence Erlbaum.

K. D. Miller. Equivalence of a sprouting-and-retraction model and correlation-based plasticity models of neural development. *Neural Computation*, 10:529–547, 1998.

M. Minsky. Steps towards artificial intelligence. *Proceedings of the IRE*, 49:8–30, 1961.

M. R. Morelande and N. J. Gordon. Target tracking through a coordinated turn. In *Proceedings of ICASSP2005*, Philadelphia, PA, 2005.

D. Mumford. Thalamus. In M. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 981–984, Cambridge, MA, 1995. MIT Press.

T. H. Murphy. Activity-dependent synapse development: changing the rules. *Nature Neuroscienc*, 6(1):9–11, 2003.

K. Nakano. Associatron—a model of associative memory. *IEEE Transactions on Systems, Man, and Cybernetics*, 2(3):380–388, 1972.

R. Neal. *Bayesian Learning for Neural Networks.* Springer, Berlin, 1996.

R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. Kluwer Academic Publishers, 1998.

N. K. Nishihara and T. Poggio. Hidden cues in random-line stereograms. *Nature*, 300: 347–349, 1982.

M. Nø rgaard. *Neural Network Based System Identification Toolbox: for Use with MATLAB.* The MathWorks Inc., Natick, MA, 2000.

E. Oja. A simplified neuron model as a principal component analyzer. *Journal of Mathematical Biology*, 15:267–273, 1982.

E. Oja. Neural networks, principal components, and subspaces. *International Journal of Neural Systems*, 1:61–68, 1989.

B. A. Olshausen and D. J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–609, 1996.

E. Parzen. *Time Series Analysis Papers.* Holden-Day, Inc., San Francisco, 1967.

G. Patel and S. Haykin. Chaotic dynamics. In S. Haykin, editor, *Kalman Filtering and Neural Networks*, pages 83–122, New York, 2001. Wiley.

R. Pfeifer and C. Scheier. *Understanding Intelligence.* MIT Press, Cambridge, MA, 1999.

M. Pitt and N. Shephard. Filtering via simulation: Auxillary particle filter. *Journal of American Statistical Association*, 94:590–599, 1999.

W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C.* Cambridge University Press, UK, 2nd edition, 1992.

J. Principe, D. Xu, and J. W. Fisher. Information-theoretic learning. In S. Haykin, editor, *Unsupervised Adaptive Filtering*, volume 1, pages 265–319, New York, 2000. Wiley.

N. Qian. Computing stereo disparity and motion with known binocular cell properties. *Neural Computation*, 6:390–404, 1994.

N. Qian. On the momentum term in gradient descent learning algorithms. *Neural Networks*, 12:145–151, 1999.

N. Qian and T. J. Sejnowski. Learning to solve random-dot stereograms of dense and transparent surfaces with recurrent backpropagation. In *Proceedings of 1988 Connectionist Models Summer School*, pages 435–443, Harcourt, 1989. London.

R. P. Rao. An optimal estimation approach to visual perception and learning. *Vision Research*, 39:1963–1989, 1999.

R. P. N. Rao, B. A. Olshausen, and M. S. Lewicki, editors. *Probabilistic Models of the Brain: Perception and Neural Function.* MIT Press, Cambridge, MA, 2002.

A. Renyi. *Selected Papers of Alfred Renyi*, volume 2. Akademia Kiado, Budapest, 1976.

G. Reymond, J. Droulez, and A. Kemeny. Visuovestibular perception of self-motion modeled as a dynamic optimization process. *Biological Cybernetics*, 87:301–314, 2002.

C. P. Robert. *The Bayesian Choice: A Decision-Theoretic Motivation.* Springer, New York, 2nd edition, 2001.

C. P. Robert and G. Casella. *Monte Carlo Statistical Methods.* Springer, Berlin, 1999.

S. J. Roberts, D. Husmeier, I. Rezek, and W. Penny. Bayesian approaches to Gaussian mixture modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1133–1144, November 1998.

I. Rock and S. Palmer. The legacy of Gestalt psychology. *Scientific American*, pages 84–90, December 1990.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by propagating error. *Nature*, 323:533–536, October 1986.

T. E. Sanger. Optimal unsupervised learning in a single-layer linear feedforward neural network. *Neural Networks*, 2(6):459–473, 1989.

P. S. Sastry, M. Magesh, and K. P. Unnikrishnan. Two timescale analysis of Alopex algorithm for optimization. *Neural Computation*, 14:2729–2750, 2002.

O. Schwartz and E. Simoncelli. Natural sound statistics and divisive normalization in the auditory system. In T. Leen, T. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems, 13*, pages 166–172, Cambridge, MA, 2001. MIT Press.

T. J. Sejnowski. Statistical constraints on synaptic plasticity. *Journal of Theoretical Biology*, 69:385–389, 1977a.

T. J. Sejnowski. Storing covariance with nonlinearly interacting neurons. *Journal of Mathematical Biology*, 4:303–321, 1977b.

T. J. Sejnowski and G. E. Hinton. Separating figure from ground with a Boltzmann machine. In M. A. Arbib and A. R. Hanson, editors, *Vision, Brain and Cooperative Computation*, Cambridge, MA, 1987. MIT Press.

T. J. Sejnowski and G. Tesauro. The Hebb rule for synaptic plasticity: algorithms and implementations. In J. H. Byrne and W. O. Berry, editors, *Neural Models of Plasticity*, pages 94–103, San Diego, CA, 1989. Academic Press.

M. N. Shadlen and J. Movshon. Synchrony unbound: a critical evaluation of the temporal binding hypothesis. *Neuron*, 24:67–77, 1999.

M. N. Shadlen and W. T. Newsome. Noise, neural codes and cortical organization. *Current Opinion in Neurobiology*, 4:569–579, 1994.

S. Shah and P. S. Sastry. New algorithms for learning and pruning oblique decision trees. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 29:494–505, November 1999.

C. E. Shannon. A mathematical theory of communication. *Bell Systems Technical Journal*, 27:379–423, 623–656, 1948.

O. Shriki, H. Sompolinsky, and D. Lee. An information maximization approach to overcomplete and recurrent representations. In T. Leen, T. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems, 13*, pages 612–618, Cambridge, MA, 2001. MIT Press.

B. W. Silverman. *Density Estimation for Statistics and Data Analysis.* Chapman and Hall, London, 1986.

W. Singer. Synchronization of cortical activity and its putative role in information processing and learning. *Annual Review of Physiology*, 55:349–374, 1993.

W. Singer. Neuronal synchrony: a versatile code for the definition of relations. *Neuron*, 24:49–65, 1999.

J. Sjöberg, Q. Zhang, L. Ljung, A. Benveniste, B. Delyon, P. Glorennec, H. Hjalmarsson, and A. Juditsky. Nonlinear black-box modeling in system identification: A unified overview. *Automatica*, 31(12):1691–1724, 1995.

O. Sporns, G. Tononi, and G. M. Edelman. Modeling perceptual grouping and figure-ground segregation by means of active reentrant connections. *Proceedings of National Academy of Sciences, USA*, 88:129–133, 1991.

G. S. Stent. A physiological mechanism of Hebb's postulate of learning. *Proceedings of National Academy of Sciences, USA*, 70:997–1001, 1973.

R. S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44, 1988.

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction.* MIT Press, Cambridge, MA, 1998.

F. Takens. On the numerical determination of the dimension of an attractor. In D. Rand and L. S. Young, editors, *Dynamical Systems and Turbulence*, pages 366–381, Berlin, 1981. Springer-Verlag.

H. Tanizaki. *Nonlinear Filters: Estimation and Applications.* Springer-Verlag, New York, 2nd edition, 1996.

H. Tanizaki. Nonlinear and non-Gaussian state-space modeling with Monte Carlo techniques: A survey and comparative study. In C. R. Rao and D. N. Shanbhag, editors, *Handbook of Statistics.* North-Holland, 2000.

M. A. Tanner and W. H. Wong. The calculation of posterior distribution by data augmentation (with discussion). *Journal of American Statistical Association*, 82:528–550, 1987.

V. Tarokh and H. Jafarkhani. A differential detection scheme for transmit diversity. *IEEE Journal on Selected Areas in Communications*, 18(7):1168–1174, 2000.

E. Tzanakou. *Supervised and Unsupervised Pattern Recognition: Feature Extraction and Computational Intelligence.* CRC Press, Roca Raton, FL, 2000.

E. Tzanakou, R. Michalak, and E. Harth. The Alopex process: visual receptive fields by response feedback. *Biological Cybernetics*, 35:161–174, 1979.

N. Ueda, R. Nakano, Z. Ghahramani, and G. Hinton. SMEM algorithm for mixture models. *Neural Computation*, 12:2109–2128, 2000.

S. Ullman. *The Interpretation of Visual Motion.* MIT Press, Cambridge, MA, 1979.

K. P. Unnikrishnan and K. P. Venugopal. Alopex: A correlation-based learning algorithm for feedforward and recurrent neural networks. *Neural Computation*, 6(3):469–490, 1994.

R. van der Merwe, J. F. G. de Freitas, A. Doucet, and E. Wan. The unscented particle filter. TR-30, Cambridege University Engineering Department, August 2000.

D. A. van Dyk and X.-L. Meng. The art of data augmentation (with discussion). *Journal of Computational and Graphical Statistics*, 10:1–111, 2001.

J. J. Verbeek, N. Vlassis, and B. Kröse. Efficient greedy learning of Gaussian mixture models. *Neural Computation*, 15:469–485, 2003.

A. Verri, M. Straforini, and V. Torre. Computational aspects of motion perception in natural and artificial vision systems. *Philosophical Transactions of the Royal Society of London, B*, 337:429–443, 1992.

C. von der Malsburg. The correlation theory of brain function. Internal Rep. 81-2, Dept. Neurobiology, Max-Plank-Institute for Biophysical Chemistry, 1981.

C. von der Malsburg. The what and why of binding: the modeler's perspective. *Neuron*, 24:95–104, 1999.

C. von der Malsburg and W. Schneider. A neural cocktail-party processor. *Biological Cybernetics*, 54:29–40, 1986.

E. Wan and R. van der Merwe. The unscented Kalman filter. In S. Haykin, editor, *Kalman Filtering and Neural Networks*, pages 221–280, New York, 2001. Wiley.

D. L. Wang and G. J. Brown. Separation of speech from interfering sounds based on oscillatory correlation. *IEEE Transactions on Neural Networks*, 10(3):684–697, 1999.

X. Wang and V. Poor. Robust multiuser detection in non-gaussian channels. *IEEE Transactions on Signal Processing*, 47(2):289–305, February 1999.

M. West. Mixture models, Monte Carlo, Bayesian updating and dynamic models. *Computer Science and Statistics*, 24:325–333, 1992.

B. Widrow and M. E. Hoff. Adaptive switch circuits. In *IRE WESCON Convention Record*, pages 96–104, 1960.

B. Widrow and S. D. Stearns. *Adaptive Signal Processing.* Prentice Hall, Englewood Cliffs, NJ, 1985.

N. Wiener. Generalized Harmonic analysis. *Acta Mathematica*, 55:117–258, 1930.

N. Wiener. *Cybernetics: Or Control and Communications in the Animal and the Machine.* Wiley, New York, 1948.

N. Wiener. *Extrapolation, Interpolation and Smoothing of Time Series.* MIT Press, Cambridge, MA, 1949.

D. J. Willshaw, O. P. Buneman, and H. C. Longuet-Higgins. Nonholographic associative memory. *Nature*, 222:960–962, 1969.

D. J. Willshaw and P. Dayan. Optimal plasticity from matrix memories: What goes up must come down. *Neural Computation*, 2:85–93, 1990.

D. J. Willshaw and C. von der Malsburg. How patterned neural connections can be set up by self-organization. *Proceedings of the Royal Society, Series B*, 194:431–445, 1976.

P. W. Wolniansky, G. J. Foschini, G. D. Golden, and R. A. Valenzuela. V-BLAST: An architecture for realizing very high data rates over the rich-scattering wireless channel. In *Proc. ISSSE-98*, Pisa, Italy, 1998.

D. M. Wolpert and Z. Ghahramani. Computational principles of movement neuroscience. *Nature Neuroscience*, 3:1212–1217, 2000.

W. H. Wong and F. Liang. Dynamic importance weighting in Monte Carlo and optimization. *Proceeding of National Academy of Sciences, USA*, 94:14220–14224, 1997.

X. Xie and H. S. Seung. Equivalence of backpropagation and contrastive Hebbian learning in a layered network. *Neural Computation*, 15(2):441–454, 2003.

L. Xu and M. I. Jordan. On convergence properties of the EM algorithm for gaussian mixtures. *Neural Computation*, 8:129–151, 1996.

H. H. Yang and S. Amari. On-line learning algorithms for blind separation – maximum entropy and minimum mutual information. *Neural Computation*, 9:1457–1482, 1997.

A. L. Yuille and N. M. Grzywacz. A computational theory for the perception of coherent visual motion. *Nature*, 335:71–74, 1988.

V. S. Zaritskii, V. B. Svetnik, and L. I. Shimelevich. Monte Carlo technique in problems of optimal data processing. *Automation and Remote Control*, 12:95–103, 1975.