EURASIP Book Series on Signal Processing and Communications

# Advances in Nonlinear Signal and Image Processing

Edited by: Stephen Marshall
and Giovanni L. Sicuranza

# Advances in Nonlinear Signal and Image Processing

EURASIP Book Series on Signal Processing and Communications, Volume 6

# Advances in Nonlinear Signal and Image Processing

*Edited by: Stephen Marshall and Giovanni L. Sicuranza*

Hindawi Publishing Corporation
http://www.hindawi.com

EURASIP Book Series on Signal Processing and Communications
Editor-in-Chief: Alex Gershman


Hindawi Publishing Corporation
410 Park Avenue, 15th Floor, #287 pmb, New York, NY 10022, USA
Nasr City Free Zone, Cairo 11816, Egypt
Fax: +1-866-HINDAWI (USA Toll-Free)



ISBN 977-5945-37-2

# Contents

# Preface

In recent years the area of nonlinear signal and image processing has emerged as a distinct research field in its own right. It has formed from a fusion of techniques which, whilst coming from very different sources, share many characteristics. These techniques have appeared for two main reasons:

(i) the availability of high-powered computing at modest cost which can allow many operations to be carried out in real time or near real time,

(ii) the emergence of new applications requiring complex solutions for which classical linear techniques simply do not work.

Whilst linear techniques have many advantages, they mainly arise in a mathematical framework based on orthogonal spaces. The problem is tackled by mapping the data into lower-order spaces. For example, in Fourier analysis the modeling of both signals and systems by a summation of complex sinusoids leads to an elegant analysis with all cross-terms conveniently vanishing from the calculation. Filters can in effect be designed one frequency at a time. Such powerful superposition properties result in neat closed-form solutions of optimal filters. In many branches of engineering, from circuit analysis to audio processing, these techniques have worked well. In transient and discrete systems, Laplace and z transforms have proved to be similarly invaluable. However, solutions are limited to those based on superposition of sinusoids. Relaxation of this constraint is the basis of nonlinear signal and image processing. It can lead to better solutions to problems at the price of increased mathematical complexity, particularly in operator design.

The vitality of the field is proven by

(i) the success of the series of workshops on nonlinear signal and image processing organized every two years by the NSIP Board (http://poseidon.csd.auth.gr/NSIP/);

(ii) the special issues published on this topic, and in particular the recent special issues of the EURASIP Journal on Applied Signal Processing;

(iii) the special sessions organized at international conferences, and in particular the special session organized at EUSIPCO-04 in Vienna;

(iv) the number of contributions appearing in journals and presented at international conferences.

As nonlinear signal processing matures, it is being applied to many new applications and in new contexts. Therefore, the aim of this book is to present a review of emerging new areas of interest involving nonlinear signal and image processing theories, techniques, and tools. In consideration of the different topics dealt with, it has been a natural choice to publish an edited book where each chapter is written by leading experts in their respective fields. Moreover, particular attention has

been paid to having a common structure for every chapter. In fact, each chapter includes an introductory part describing briefly the state of the art of the area in which the proposed topic is discussed. The advantages of using a nonlinear approach are then clearly delineated and new methods and results are demonstrated. A summary and a large number of references complete each contribution.

The first chapter describes an explicit method to deal with stochastic differential equations whose solution exhibits nonstationary properties, such as a spectrum that is changing in time. The basis of the method is to transform the differential equation into the phase space of time and frequency. Using this method the authors have addressed a number of problems previously intractable. In this chapter, a new approach that can handle nonstationary stochastic systems is presented with particular reference to the Langevin Brownian motion equation with time-dependent coefficients.

Chapter 2 considers aperture filters which represent a new filtering approach. They are not limited by constraints of linearity and so can address many complex filtering problems. The filters are subject to windowing in the amplitude domain as well as the spatial domain. This allows practical design to take place through filtering using a realistic-sized training set.

Because finite sets are not closed under the ordinary arithmetic operations of addition and multiplication, filters that map one sequence taking values in a finite set into another such sequence are necessarily nonlinear. Chapter 3 considers the general problems of designing these nonlinear filters and developing useful characterizations like power spectra for sequences taking values in finite sets. Both the general case, where there is no additional algebraic structure available to exploit, and the specialized case, where some useful structure is present, are discussed. Applications to DNA sequence analysis and database cleaning are considered.

A vibrant research area has recently been created with the introduction of biological cDNA microarrays. These microarrays make it possible to simultaneously measure the expressions level for thousands of genes. The large volume of data produced requires that new techniques be developed to model the subtle and complex interactions of these genes. Chapter 4 describes methods for gene classification based on a ternary model which have emerged from nonlinear signal processing.

The adoption of nonlinear methods in speech analysis and synthesis has become a common practice over the last two decades. Chapter 5 explores and details why nonlinear methods should be considered in this research domain. In particular, this chapter illustrates the main nonlinear methods adopted, such as neural networks, nonlinear dynamics, and fractal methods. In addition, the insights offered for speech analysis by nonlinear methods are discussed. Key results on synthesis by a variety of nonlinear methods are shown too.

Chapter 6 deals with an important problem in the communication area related to the characteristics of commonly used components such as the power amplifiers (PAs). It is well known that a linear PA is inefficient, while an efficient PA comes with undesirable nonlinear effects. In this chapter, two types of nonlinear distortions, spectral regrowth and in-band distortion, are first discussed. Then,

the authors focus on digital baseband predistortion linearization of nonlinear PAs, with the aim of achieving both high efficiency as well as high linearity. Some recent developments on predistorter design with memory are presented together with testbed measurement results.

Active noise control (ANC) is another topic that has been considered over many years with reference to linear models. On the other hand, there is the evidence that nonlinear distortions can affect actual applications, in turn requiring models with an intrinsic complexity higher than that of linear models. In Chapter 7, nonlinear single-channel and multichannel ANC schemes based on polynomial and other useful models are described. Particular emphasis is posed on both efficient structures and fast updating rules.

In Chapter 8, an overview of watermarking schemes based on chaotic generators is presented. In fact, sequences generated by chaotic generators can be used as watermark signals instead of the widely used pseudorandom watermarks. The statistical properties of watermark sequences generated by piecewise-linear Markov maps embedded in an additive or a multiplicative way are presented. An example of applying the chaotic watermarking framework on audio signals, demonstrating its superiority with respect to both robustness and inaudibility, is described.

The modeling of texture is an important area in many fields of research. One such area is the monitoring of corrosion of aircraft parts. Nonlinear multiscale analysis tools known as granulometries have been used to quantify and classify an evolving texture over time. In Chapter 9, a deterministic link is established between the moments of the granulometric profiles and the parameters which characterize the textures. This is used to estimate the severity of the corrosion of a number of samples.

The final three chapters of the book have a common theme that concerns strategies for handling multivalued data. Many nonlinear processing techniques, including weighted order statistics (WOS) and variations on the median filter possess a stage in which an output is generated based on the rank order of the data samples. For scalar values this is a fully defined operation, but in the case of multivalued or vector data it is not clear how to extend these concepts. Chapter 10 presents a reasoned evaluation of techniques for extending weighted median filters to multivalued data.

A specific case of multivalued data is color information. Whichever representation is used, color inevitably results in a three-dimensional entity. Chapter 11 gives an overview of nonlinear techniques specifically tuned to the processing of color image data. Issues such as noise reduction and color bleeding are addressed. The way in which information from the different color bands is combined is discussed with detailed examples.

Finally, Chapter 12 considers the problem of edge detection in color images. The concept of color morphological gradient is described and this is evaluated against other color edge detection techniques in the presence of Gaussian noise. As well as determining their robustness in edge detection, the metrics used also include the ability of the color edge detectors to resolve edge direction.

**Acknowledgments**

As editors of this book, we would like to express our warm gratitude to all the authors for the timely contribution of high-quality texts. We also thank the staff of Hindawi Publishing Corporation for the invaluable assistance and cooperation during all phases of the production of this book. Finally, we would like to dedicate this book to our wives Joan and Maria Lucia for their continuous encouragement during the long time spent in its organization and preparation.

*Stephen Marshall*
*Giovanni L. Sicuranza*

# 1 Nonstationary stochastic differential equations

Lorenzo Galleani and Leon Cohen

## 1.1. Introduction

Historically, stochastic methods have been developed for the time-invariant stationary case. However, typically in nature, stochastic processes are nonstationary, but very little work has been done on this case because of the difficulties involved. Nonstationary processes come about in two general ways. First the physical parameters of the process can vary in time and secondly even for a process that is generally thought to be stationary there is a nonstationary part, the transient, which is usually neglected, but which is very important and interesting. We believe that it is usually neglected because proper methods have not been developed to study it. In this chapter we develop a new approach that can handle nonstationary stochastic systems. We now motivate the need for this development by examples.

Particularly simple and important is the classical stochastic equation, the Langevin equation for Brownian motion,

$$m\frac{d\mathbf{v}(t)}{dt} + \beta\mathbf{v}(t) = \mathbf{f}(t), \tag{1.1}$$

where $\mathbf{v}(t)$ is the velocity of the particle,[1] $\beta$ is the friction coefficient, $m$ is the mass of the particle, and $\mathbf{f}(t)$ is the random force, taken to be a Gaussian random process described statistically by

$$\begin{aligned} E[\mathbf{f}(t)] &= 0, \\ E[\mathbf{f}(t_1)\mathbf{f}(t_2)] &= N_0\delta(t_1 - t_2), \end{aligned} \tag{1.2}$$

where $E$ is the ensemble average operator. This equation has been studied intensely since its inception in 1908. The fundamental aim is to obtain the statistical properties of the output, $\mathbf{v}(t)$. Perhaps the most fundamental result is the power spectrum

[1] We use bold letters for stochastic quantities.

of $\mathbf{v}(t)$, first obtained by Wang and Uhlenbeck in 1945 in their classic paper on the subject [19]. In general the power spectrum of a stochastic process $\mathbf{x}(t)$ is defined by[2] [16]

$$S_{\mathbf{x}}(\omega) = \frac{1}{\sqrt{2\pi}} \int R_{\mathbf{x}}(\tau) e^{-i\tau\omega} d\tau, \tag{1.3}$$

where $R_{\mathbf{x}}(\tau)$ is the autocorrelation function:

$$R_{\mathbf{x}}(\tau) = E\big[\mathbf{x}(t)\mathbf{x}^*(t+\tau)\big]. \tag{1.4}$$

This definition is of course valid for wide sense stationary processes. The power spectrum of $\mathbf{v}(t)$ for the Langevin equation is

$$S_{\mathbf{v}}(\omega) = \frac{1}{\sqrt{2\pi}} \frac{N_0}{\beta^2 + \omega^2}. \tag{1.5}$$

We now ask how it is possible that time has disappeared in this result, since clearly the solution of (1.1) is time dependent. The reason is that the system has been allowed to evolve for an infinite time! Although not always explicitly stated, the way infinite time enters into the result is that the system is turned on at $t = -\infty$, and hence for any finite time an infinite amount of time has passed. This has the effect of totally neglecting the approach to equilibrium. Our aim is to obtain the full evolution of the system. In particular, if we start the system at a finite time, then we know that it must evolve to (1.5), and this is illustrated in Figure 1.1. Our goal is to develop a method that will allow us to obtain the region of time marked by the question mark.

Now consider the Langevin equation with time-dependent coefficients. In particular we take the friction term to be time-dependent,

$$m\frac{d\mathbf{v}(t)}{dt} + \beta(t)\mathbf{v}(t) = \mathbf{f}(t) \tag{1.6}$$

then clearly we have a situation that in general will never achieve equilibrium and hence will always be a nonstationary stochastic process. How does one study such a situation? Our method allows one to do so. In the next section we discuss how to define a time dependent spectrum and subsequently we obtain an equation of evolution that governs it.

## 1.2. Time-dependent power spectrum

Since we are dealing with time-dependent phenomena we need a generalization of the standard power spectrum. That is, we need a time-dependent power spectrum and this is commonly called the instantaneous spectrum. Here, as a definition for

[2] All integrals without limits imply integration from $-\infty$ to $+\infty$.

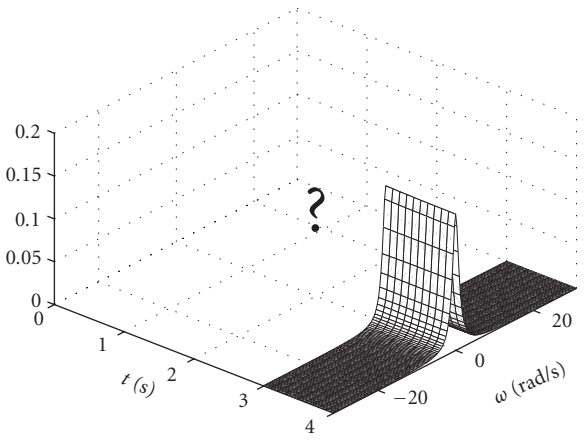



FIGURE 1.1. Instantaneous spectrum of Brownian motion from (1.1), the Langevin equation. The question mark replaces the transient behavior of the random system. The instantaneous spectrum of the transient can be obtained exactly with our method, as explained in Section 1.5.

the instantaneous spectrum of a random process $\mathbf{x}(t)$ we use the *Wigner spectrum* [14, 15]

$$\overline{W}_{\mathbf{x}}(t,\omega) = \frac{1}{2\pi}\int E\left[\mathbf{x}^*\left(t-\frac{\tau}{2}\right)\mathbf{x}\left(t+\frac{\tau}{2}\right)\right]e^{-i\tau\omega}d\tau. \tag{1.7}$$

It is the ensemble average of the Wigner distribution [2, 4, 21]. The Wigner spectrum is a function that describes the power in both time and frequency. A fundamental property of the Wigner spectrum is that if we integrate out time then we obtain the standard power spectrum

$$\int \overline{W}_{\mathbf{x}}(t,\omega)dt = \left|\mathbf{X}(\omega)\right|^2 = S_{\mathbf{x}}(\omega). \tag{1.8}$$

In (1.7) we have used $\mathbf{x}(t)$ to denote an arbitrary random process. Here we deal mostly with velocity, and hence $\mathbf{x}(t)$ will be $\mathbf{v}(t)$.

### 1.3. The equation of motion for a nonstationary stochastic system

Our aim is to obtain the equation of motion for the Wigner spectrum for a stochastic process that is the solution of an ordinary stochastic differential equation

$$a_n\frac{d^n\mathbf{x}(t)}{dt^n} + a_{n-1}\frac{d^{n-1}\mathbf{x}(t)}{dt^{n-1}} + \cdots + a_1\frac{d\mathbf{x}(t)}{dt} + a_0\mathbf{x}(t) = \mathbf{f}(t), \tag{1.9}$$

where $\mathbf{f}(t)$ is a given stochastic process. One could solve this equation for $\mathbf{x}(t)$ and then find the ensemble average as needed to calculate the Wigner spectrum, and then do the integration as required. This is generally a difficult procedure. We have

developed a method that is more direct: we obtain the equation of evolution for the Wigner spectrum and solve it [7–12]. To accomplish that, we write the differential equation in polynomial form

$$P(D)\mathbf{x}(t) = \mathbf{f}(t), \tag{1.10}$$

where $D$ and $P(D)$ are, respectively,

$$\begin{aligned} D &= \frac{d}{dt}, \\ P(D) &= a_n D^n + a_{n-1} D^{n-1} + \cdots + a_1 D + a_0. \end{aligned} \tag{1.11}$$

The differential equation for the Wigner spectrum $\overline{W}_{\mathbf{x}}(t,\omega)$, of $\mathbf{x}(t)$, is then given by

$$P^*(A)P(B)\overline{W}_{\mathbf{x}}(t,\omega) = \overline{W}_{\mathbf{f}}(t,\omega), \tag{1.12}$$

where

$$A = \frac{1}{2}\frac{\partial}{\partial t} - i\omega, \qquad B = \frac{1}{2}\frac{\partial}{\partial t} + i\omega. \tag{1.13}$$

For the case of time-dependent coefficients,

$$a_n(t)\frac{d^n\mathbf{x}(t)}{dt^n} + a_{n-1}(t)\frac{d^{n-1}\mathbf{x}(t)}{dt^{n-1}} + \cdots + a_1(t)\frac{d\mathbf{x}(t)}{dt} + a_0(t)\mathbf{x}(t) = \mathbf{f}(t), \tag{1.14}$$

where $a_0(t),\ldots,a_n(t)$ are the time-varying deterministic coefficients. We rewrite (1.14) in polynomial notation

$$P(D,t)\mathbf{x}(t) = \mathbf{f}(t), \tag{1.15}$$

where now

$$P(D,t) = a_n(t)D^n + a_{n-1}(t)D^{n-1} + \cdots + a_1(t)D + a_0(t). \tag{1.16}$$

The equation for the Wigner spectrum is

$$P^*(A,\mathcal{E})P(B,\mathcal{F})\overline{W}_{\mathbf{x}}(t,\omega) = \overline{W}_{\mathbf{f}}(t,\omega), \tag{1.17}$$

where the additional operators are

$$\mathcal{E} = t + \frac{1}{2i}\frac{\partial}{\partial\omega}, \qquad \mathcal{F} = t - \frac{1}{2i}\frac{\partial}{\partial\omega}. \tag{1.18}$$

The star sign indicates complex conjugation of the coefficients used in the expansion of the functions $a_k(t)$. Equations (1.12) and (1.17) are partial differential equations in time and frequency.

### 1.3.1. Remarks

We point out that it is a common procedure to transform random differential equations to other domains. The most common transformation is to the frequency domain. In particular the transformation of (1.9) to the frequency domain is [16]

$$S_{\mathbf{x}}(\omega) = \left|H(\omega)\right|^2 S_{\mathbf{f}}(\omega), \tag{1.19}$$

where $H(\omega)$ is the transfer function given by

$$H(\omega) = \frac{1}{a_n(i\omega)^n + a_{n-1}(i\omega)^{n-1} + \cdots + a_1 i\omega + a_0}. \tag{1.20}$$

The $a$ coefficients are constant with time. The assumption in this transformation is for $\mathbf{f}(t)$ to be a wide sense stationary process, which implies that $\mathbf{x}(t)$ is also wide sense stationary. In our method the assumptions on the stationarity of the coefficients and the input random process $\mathbf{f}(t)$ are not necessary.

## 1.4. The nonstationary Wiener process

We now apply our method to the Wiener process [3, 5]

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{f}(t) \tag{1.21}$$

with initial conditions $X(0) = \mathbf{x}_0$, where $\mathbf{x}_0$ is a stochastic variable. We will turn the system on at a finite time and hence we will obtain the complete solution of the problem. That is, we will get the total solution, including the transient behavior. First we point out that it is well known that for (1.21) the power spectrum is given by [16]

$$P_{\mathbf{x}}(\omega) = \frac{N_0}{\sqrt{2\pi}} \frac{1}{\omega^2}. \tag{1.22}$$

This is obtained by using the standard result of linear time-invariant (LTI) systems. It is important to appreciate that the $1/\omega^2$ spectrum for the Wiener process is reached in either of two different ways:

(1) the initial condition is set to zero at $t = 0$ and by taking $t = \infty$ in the general solution of (1.21);
(2) alternatively by setting the initial condition to zero at $t = -\infty$ and this has the effect of the passage of an infinite amount of time.

For both types of initial conditions, the stationary spectrum is reached because an infinite amount of time has passed. Using our method we will obtain the

full solution, that is, the full evolution of the instantaneous spectrum. Rewriting (1.21) in the polynomial form of (1.10)

$$D\mathbf{x}(t) = \mathbf{f}(t) \tag{1.23}$$

and using (1.12), we obtain

$$AB\overline{W}_{\mathbf{x}}(t,\omega) = \overline{W}_{\mathbf{f}}(t,\omega). \tag{1.24}$$

Using the definitions of the operators, we have that

$$AB = \frac{1}{4}\frac{\partial^2}{\partial t^2} + \omega^2, \tag{1.25}$$

and hence

$$\frac{1}{4}\frac{\partial^2 \overline{W}_{\mathbf{x}}(t,\omega)}{\partial t^2} + \omega^2 \overline{W}_{\mathbf{x}}(t,\omega) = \overline{W}_{\mathbf{f}}(t,\omega), \tag{1.26}$$

where $\overline{W}_{\mathbf{f}}(t,\omega)$ is the Wigner spectrum of $\mathbf{f}(t)$. If $\mathbf{f}(t)$ is white Gaussian noise with autocorrelation

$$R_{\mathbf{f}}(\tau) = N_0\delta(\tau), \tag{1.27}$$

then one can readily show that

$$\overline{W}_{\mathbf{f}}(t,\omega) = \frac{N_0}{2\pi}, \tag{1.28}$$

and hence

$$\frac{1}{4}\frac{\partial^2 \overline{W}_{\mathbf{x}}(t,\omega)}{\partial t^2} + \omega^2 \overline{W}_{\mathbf{x}}(t,\omega) = \frac{N_0}{2\pi}. \tag{1.29}$$

The exact solution to (1.29) is

$$\overline{W}_{\mathbf{x}}(t,\omega) = \frac{N_0}{2\pi\omega^2}[1-\cos 2\omega t] + \frac{E[\mathbf{x}_0^2]}{\pi\omega}\sin 2\omega t, \quad t \geq 0, \tag{1.30}$$

and $\overline{W}_{\mathbf{x}}(t,\omega) \equiv 0$ when $t < 0$. For $E[\mathbf{x}_0^2] = 0$, one has

$$\overline{W}_{\mathbf{x}}(t,\omega) = \frac{N_0}{2\pi\omega^2}[1-\cos 2\omega t] = \frac{N_0}{\pi}\frac{\sin^2 \omega t}{\omega^2}. \tag{1.31}$$

In Figure 1.2 we show the transient of the Wiener process.

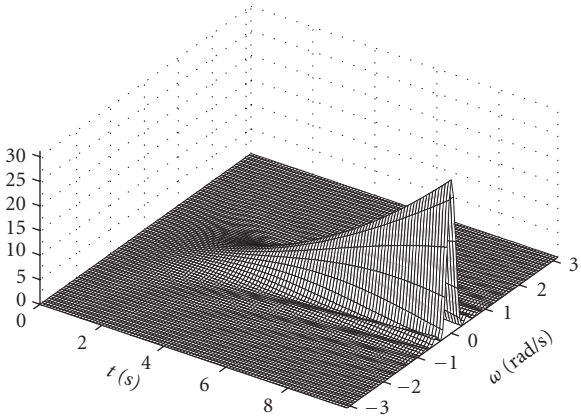



FIGURE 1.2. The instantaneous spectrum of the Wiener process defined in (1.21). The spectrum grows to infinity at $\omega = 0$ and reaches the singularity in the classical frequency spectrum that has a $1/\omega^2$ distribution. Notice that the Wigner spectrum, given in (1.31), is nonnegative.

It is interesting to note that for $\omega \to 0$,

$$\lim_{\omega\to 0} \overline{W}_{\mathbf{x}}(t,\omega) = \lim_{\omega\to 0} \frac{N_0}{2\pi\omega^2}[1-\cos 2\omega t] = \frac{N_0}{\pi}t^2, \tag{1.32}$$

and hence at $\omega = 0$ the instantaneous spectrum grows to infinity and approaches the singularity in the classical power spectrum of (1.22) with a second order in time.

## 1.5. The Langevin equation: the full exact solution

Our aim is to show how to find the transient spectrum of the Langevin equation. We apply our method by first rewriting the Langevin equation in polynomial form

$$[D+\beta]\mathbf{v}(t) = \mathbf{f}(t), \tag{1.33}$$

where for convenience we take $m = 1$. Using (1.12), the equation for the Wigner spectrum is now

$$[A+\beta][B+\beta]\overline{W}_{\mathbf{x}}(t,\omega) = \overline{W}_{\mathbf{f}}(t,\omega). \tag{1.34}$$

We substitute for the operators $A$ and $B$ from (1.13) to obtain

$$\frac{1}{4}\frac{\partial^2}{\partial t^2}\overline{W}_{\mathbf{x}}(t,\omega) + \beta\frac{\partial}{\partial t}\overline{W}_{\mathbf{x}}(t,\omega) + (\beta^2+\omega^2)\overline{W}_{\mathbf{x}}(t,\omega) = \overline{W}_{\mathbf{f}}(t,\omega), \tag{1.35}$$

where $\overline{W}_{\mathbf{f}}(t,\omega)$ is the Wigner spectrum of the input random process $\mathbf{f}(t)$.

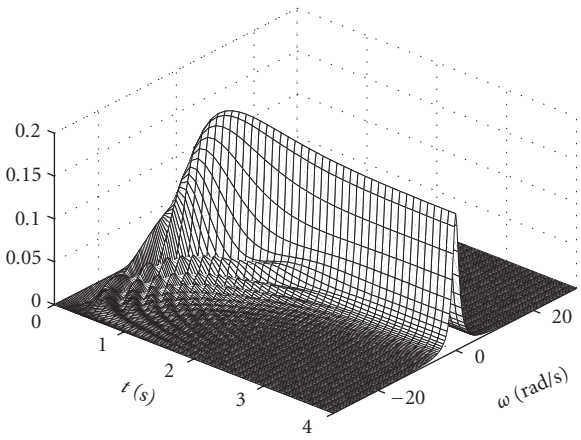



FIGURE 1.3. The complete instantaneous spectrum of Brownian motion, given in (1.36). The picture shows also the transient part that was hidden in Figure 1.1. Notice the overshoot at the beginning of the transient and the increased spread of the Wigner spectrum. As $t \to \infty$, the Wigner spectrum reaches the stationary solution, that is, the Lorentzian distribution of (1.38).

The solution to (1.35) can be found with the standard methods for differential equations. It is

$$\overline{W}_{\mathbf{v}}(t,\omega) = \frac{1}{\pi}\left(E[\mathbf{v}_0^2] - \frac{N_0}{2\beta}\right)e^{-2\beta t}\frac{\sin 2\omega t}{\omega} + \frac{N_0}{2\pi}\frac{1}{\beta^2+\omega^2} - \frac{N_0}{2\pi}\frac{e^{-2\beta t}}{\beta^2+\omega^2}\left(\cos 2\omega t - \frac{\omega}{\beta \sin 2\omega t}\right), \quad t \geq 0, \tag{1.36}$$

and $\overline{W}_{\mathbf{v}}(t,\omega) \equiv 0$ for negative times. In the solution we are considering a random initial condition $\mathbf{v}_0$, that is,

$$\mathbf{v}(0) = \mathbf{v}_0. \tag{1.37}$$

This solution allows us to understand the transient behavior of the Langevin equation. In Figure 1.3 we show a typical case.

We see that the instantaneous spectrum has an overshoot effect at the beginning of the transient. Also the spread of the solution is larger for small times. Then the solution rapidly approaches the stationary spectrum given by the Lorentzian distribution of (1.5), the standard result. The behavior of the frequencies in the transient suggests the possibility of designing systems in order to have smooth transitions of the instantaneous spectrum from zero to the stationary solution. This can be a fundamental issue in minimizing the electromagnetic interference of an electronic device that is suddenly turned off, or in reducing the risk of breaking down an engine or a vibrating structure by a sudden rough start. These two

physical situations are examples of the importance of transients in designing and understanding a physical system. We note that as time goes to infinity we have that

$$\lim_{t\to\infty} \overline{W}_{\mathbf{v}}(t,\omega) = \frac{N_0}{\beta_0^2+\omega^2} \tag{1.38}$$

which is the classical Wang-Uhlenbeck result.

### 1.6. Quantum Langevin equation

Yang and Cohen [22] used the method described above to consider the quantum Langevin equation [1, 6, 13, 17, 18, 20]

$$m\frac{d^2\mathbf{X}(t)}{dt^2} + m\beta\frac{d\mathbf{X}(t)}{dt} + V'(\mathbf{X}) = \boldsymbol{\xi}(t), \tag{1.39}$$

where $\mathbf{X}$ is the position operator, $V(\mathbf{X})$ is the external potential, and $\boldsymbol{\xi}(t)$ is the noise operator that satisfies

$$\left\langle \left[\boldsymbol{\xi}(t), \boldsymbol{\xi}(t')\right]_+ \right\rangle = \frac{\gamma\hbar}{\pi}\int \omega e^{i\omega(t-t')}\coth\frac{\hbar\omega}{2kT}d\omega, \tag{1.40}$$

and where $\hbar$, $k$, and $T$ are the reduced Planck's constant, the Boltzman constant, and the temperature, respectively. Also, $\gamma = m\beta$. If one considers (1.39) as a classical-type equation, then it is called the *quasiclassical* Langevin equation. In such a case, one replaces the operators by ordinary variables

$$m\frac{d^2x(t)}{dt^2} + m\beta\frac{dx(t)}{dt} + V'(x) = \xi(t) \tag{1.41}$$

and the autocorrelation function is then

$$R(\tau) = \frac{\gamma\hbar}{2\pi}\int \omega e^{i\omega\tau}\coth\frac{\hbar\omega}{2kT}d\omega. \tag{1.42}$$

We consider here the case of no external potential and rewrite (1.39) as

$$\dot{p}(t) + \beta p(t) = \xi(t) \tag{1.43}$$

with

$$\begin{aligned} R_\xi(\tau) &= 2DZ\int \omega e^{i\omega\tau}\coth Z\omega\, d\omega, \\ Z &= \frac{\hbar}{2kT}. \end{aligned} \tag{1.44}$$

Using the standard Wiener-Khinchin theorem, the power spectrum is given by

$$S_\xi(\omega) = 2DZ\omega \coth Z\omega. \tag{1.45}$$

We note that as $Z \to 0$,

$$\lim_{Z\to 0} R_\xi(\tau) = 2D\delta(\tau). \tag{1.46}$$

Using our general approach we have that the differential equation for the Wigner spectrum is

$$\left[\frac{1}{4}\frac{\partial^2}{\partial t^2} + \beta\frac{\partial}{\partial t} + \beta^2 + \omega^2\right]\overline{W}_p(t,\omega) = \frac{DZ}{\pi}\omega\coth Z\omega. \tag{1.47}$$

The exact solution is

$$\overline{W}_p(t,\omega) = \frac{W_\xi}{\beta^2+\omega^2}\left[1 - e^{-(2\gamma/m)t}\cos 2\omega t\right] \tag{1.48}$$

$$= \frac{1}{\beta^2+\omega^2}\frac{DZ}{\pi}\omega\coth Z\omega\left[1 - e^{-(2\gamma/m)t}\cos 2\omega t\right]. \tag{1.49}$$

We see that (1.49) can now be thought of as the generalization of the Wang-Uhlenbeck process for quantum noise.

The quantum Wiener process is obtained by taking $\gamma = 0$ and hence the differential equation is

$$\dot{p}(t) = \xi(t). \tag{1.50}$$

The governing equation is

$$\frac{1}{4}\frac{\partial^2\overline{W}_p(t,\omega)}{\partial t^2} + \omega^2\overline{W}_p(t,\omega) = \frac{DZ}{\pi}\omega\coth Z\omega \tag{1.51}$$

giving

$$W_p(t,\omega) = \frac{DZ}{\pi}\omega\coth Z\omega(1-\cos 2\omega t) = \frac{2DZ}{\pi}\omega\coth Z\omega\sin^2\omega t. \tag{1.52}$$

## 1.7. Time-variant random systems

In many physical situations a random process can be seen as the output of a random system whose parameters change with time. The variation can be due to a sudden breakdown, aging, or cyclic variations of temperature, humidity, and other physical quantities that can influence the system behavior. In these cases the process is inherently nonstationary.

We now consider the case addressed by (1.6), namely, a time-dependent Brownian motion process. We restate here the problem for convenience, rewriting it in polynomial notation

$$[D + \beta(t)]\mathbf{v}(t) = \mathbf{f}(t), \tag{1.53}$$

where we take $\beta(t)$ to be time dependent with a linear law

$$\beta(t) = \beta_0 + \epsilon t \tag{1.54}$$

and also we take $\epsilon \ll 1$. This condition implies a slow variation with time. We now transform the equation in time to an equation in time-frequency, using (1.17), to obtain

$$\left[\frac{1}{4}\frac{\partial^2}{\partial t^2} + \frac{\epsilon^2}{4}\frac{\partial^2}{\partial \omega^2} + \beta(t)\frac{\partial}{\partial t} + \epsilon\omega\frac{\partial}{\partial \omega} + \beta^2(t) + \omega^2\right]\overline{W}_{\mathbf{v}}(t,\omega) = \frac{N_0}{2\pi}. \tag{1.55}$$

Taking into account that $\epsilon$ is small, we neglect the terms containing $\epsilon$,

$$\left[\frac{1}{4}\frac{\partial^2}{\partial t^2} + \beta(t)\frac{\partial}{\partial t} + \beta^2(t) + \omega^2\right]\overline{W}_{\mathbf{v}}(t,\omega) = \frac{N_0}{2\pi}. \tag{1.56}$$

Since there are no derivatives with respect to $\omega$, we can solve this equation as an ordinary differential equation with respect to time. We have to choose the initial conditions, which we fix using the following reasoning. We assume that the system was started at $t = -\infty$ with $\beta = \beta_0$ and hence it has reached equilibrium as given by (1.19). At $t = 0$ we turn on the time dependence as given by (1.54). Therefore the initial conditions to (1.56) are

$$\overline{W}_{\mathbf{v}}(0,\omega) = \frac{N_0}{\beta_0^2 + \omega^2}, \tag{1.57}$$

$$\frac{\partial}{\partial t}\overline{W}_{\mathbf{v}}(t,\omega) = 0, \quad t = 0. \tag{1.58}$$

The first initial condition, (1.57), is the Wigner spectrum of the stationary solution. This stationary distribution has been reached because an infinite amount of time has passed since the system was turned on at $t = -\infty$. As a consequence of being in a stationary phase before $t = 0$ there must be no change in time of the instantaneous spectrum. This condition is stated by (1.58).

Using these initial conditions, (1.56) can be easily solved with numerical algorithms. In Figure 1.4 we plot the solution $\overline{W}_{\mathbf{v}}(t,\omega)$. In Figure 1.5 a numerical simulation of the Wigner spectrum is shown. The agreement confirms our approach.

The behavior of the instantaneous spectrum is in accordance with intuition. The shape of the Wigner spectrum is in fact very similar to the Lorentzian distribution, for any given time. A similarity with the lowpass filter represented by

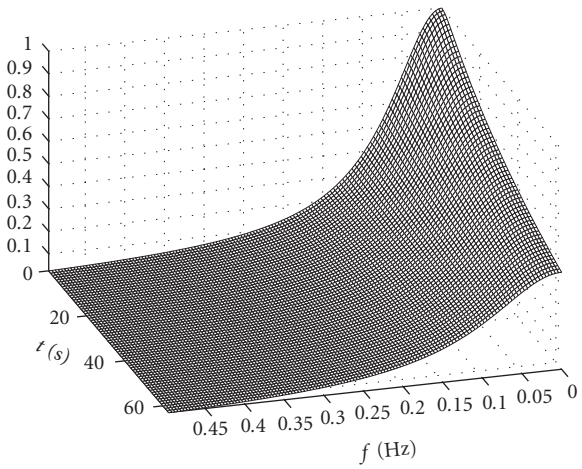



FIGURE 1.4. Approximation of the Wigner spectrum for the time-varying Brownian motion defined in (1.53). Notice the lowpass behavior of the instantaneous spectrum. As time increases, the bandwidth increases due to the variation of the parameter $\beta(t)$.

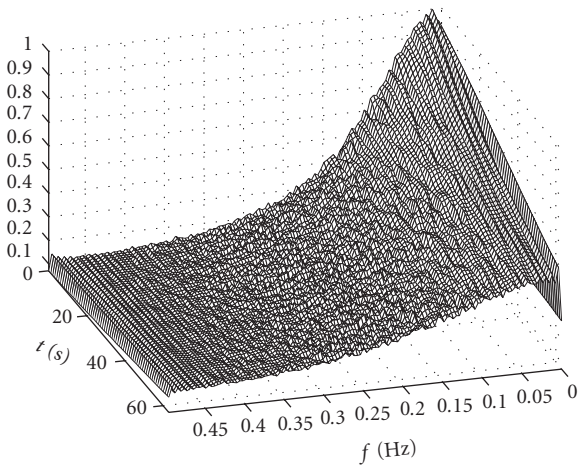



FIGURE 1.5. The instantaneous spectrum of the time-varying Brownian motion defined in (1.53) obtained through numerical simulation. The result proves the quality of the approximation shown in Figure 1.4.

the RC circuit can be used to explain the change in spread that the instantaneous spectrum exhibits. An RC circuit driven by white noise has the same equation of Brownian motion. In this case we can think of a time-varying RC circuit, where the time constant $\tau = \text{RC}$ is a function of time, $\tau = \tau(t)$. It is well known that when $\tau$ is large, the RC filter has a strong lowpass behavior, which corresponds to a small spread of the Lorentzian spectrum. On the contrary, when $\tau$ is small, the

RC circuit has a weak lowpass behavior. Since the time constant is related to the $\beta$ coefficient by an inverse proportionality, the increase of the $\beta$ coefficient, which we assumed for our case, turns out to be a decrease of the $\tau$ constant. Therefore we expect the time-varying system under study to become a weaker lowpass filter as time goes by, which is precisely what happens in Figure 1.4.

## 1.8. Summary

The study of nonstationary random processes is important because it presents fascinating theoretical issues and also because of the many applications to real-world problems. We have developed a method that allows one to study these processes by transforming the governing differential equation into an equation of motion in the phase space of time frequency. The approach is based on a direct transformation of the random differential equation defining the process. As the examples show it seems that this results in a remarkable simplification. We believe that this is the case because when one has a nonstationary process the natural representation is neither time, nor frequency, but time-frequency phase space.

We have extended our method to multi-input multi-output systems governed by differential equations, and we are currently using the approach to study the transient behavior of linear systems of arbitrary degree. Also, we have developed methods to approximate the instantaneous spectrum of time-varying systems and these results will appear in another publication.

## Acknowledgment

This work is supported by the Air Force Information Institute Research Program (Rome, NY) and the NSA HBCU/MI Program.

## Bibliography

[1] R. Benguria and M. Kac, "Quantum Langevin equation," *Physical Review Letters*, vol. 46, no. 1, pp. 1–4, 1981.

[2] L. Cohen, "Time-frequency distributions—a review," *Proceedings of the IEEE*, vol. 77, no. 7, pp. 941–981, 1989.

[3] L. Cohen, "The history of noise," in *Noise in Communication*, vol. 5473 of *Proceedings of SPIE*, pp. 85–109, Maspalomas Gran Canaria Island, Spain, May 2004.

[4] L. Cohen, *Time-Frequency Analysis*, Prentice-Hall, New York, NY, USA, 1995.

[5] J. L. Doob, *Stochastic Processes*, John Wiley & Sons, New York, NY, USA, 1953.

[6] G. W. Ford, J. T. Lewis, and R. F. O'Connell , "Quantum Langevin equation," *Physical Review A*, vol. 37, no. 11, pp. 4419–4428, 1988.

[7] L. Galleani and L. Cohen, "Direct time-frequency characterization of linear systems governed by differential equations," *IEEE Signal Processing Letters*, vol. 11, no. 9, pp. 721–724, 2004.

[8] L. Galleani and L. Cohen, "Time-varying noise," in *Advanced Signal Processing Algorithms, Architectures, and Implementations XIV*, vol. 5559 of *Proceedings of SPIE*, pp. 241–244, Denver, Colo, USA, August 2004.

[9] L. Galleani and L. Cohen, "Time-frequency characterization of random systems," in *Advanced Signal Processing Algorithms, Architectures, and Implementations XIII*, vol. 5205, pp. 21–37, San Diego, Calif, USA, August 2003.

[10] L. Galleani and L. Cohen, "Time-frequency Wigner distribution approach to differential equations," in *Nonlinear Signal and Image Processing: Theory, Methods, and Applications*, K. E. Barner and G. R. Arce, Eds., CRC Press, Boca Raton, Fla, USA, 2003.
[11] L. Galleani and L. Cohen, "The Wigner distribution for classical systems," *Physics Letters A*, vol. 302, no. 4, pp. 149–155, 2002.
[12] L. Galleani and L. Cohen, "Approximation of the Wigner distribution for dynamical systems governed by differential equations," *EURASIP Journal of Applied Signal Processing*, vol. 2002, no. 1, pp. 67–72, 2002.
[13] C. W. Gardiner and P. Zoller, *Quantum Noise*, Springer, Berlin, Germany, 1999.
[14] W. D. Mark, "Spectral analysis of the convolution and filtering of non-stationary stochastic processes," *Journal of Sound and Vibration*, vol. 11, no. 1, pp. 19–63, 1970.
[15] W. D. Mark, "Power spectrum representation for nonstationary random vibration," in *Random Vibration—Status and Recent Developments*, I. Elishakoff and R. H. Lyon, Eds., Elsevier Science, Amsterdam, Holland, 1986.
[16] A. Papoulis and S. U. Pillai, *Probability, Random Variables and Stochastic Processes*, McGraw-Hill, New York, NY, USA, 2002.
[17] W. P. Schleich, *Quantum Optics in Phase Space*, John Wiley & Sons, Berlin, Germany, 2001.
[18] M. O. Scully and M. S. Zubairy, *Quantum Optics*, Cambridge University Press, London, UK, 1997.
[19] M. C. Wang and G. E. Uhlenbeck, "On the theory of the Brownian motion II," *Reviews of Modern Physics*, vol. 17, no. 2-3, pp. 323–342, 1945.
[20] U. Weiss, *Quantum Dissipative Systems*, World Scientific, Singapore, 1999.
[21] E. P. Wigner, "On the quantum correction for thermodynamic equilibrium," *Physical Review*, vol. 40, no. 5, pp. 749–759, 1932.
[22] Y. Yang and L. Cohen, "Time-frequency Wigner evolution of the quantum Langevin equation," *Journal of Modern Optics*, vol. 52, no. 16, pp. 2223–2232, 2005.

Lorenzo Galleani: Politecnico di Torino, Corso Duca degli Abruzzi 24, 10129 Torino, Italy
*Email*: galleani@polito.it

Leon Cohen: Hunter College, City University of New York, 695 Park Avenue, New York, NY 10021, USA
*Email*: leon.cohen@hunter.cuny.edu

# 2 Aperture filters: theory, application, and multiresolution analysis

Roberto Hirata Jr., Marcel Brun, Junior Barrera, and Edward R. Dougherty

## 2.1. Introduction

The heuristic design of digital image processing (DIP) operators is a difficult and laborious task. It is difficult because it requires a lot of experience and technical knowledge of the user. It is laborious because it requires several adjustments to achieve robustness. Difficult and laborious tasks are not for anyone who wants to solve a practical problem using images. For instance, a biologist who wants to count the number of marked cells in a microscopic image may benefit in quality and precision by using image processing techniques. Assuming that this professional does not have the knowledge to combine image operators correctly, but has the ability to manually edit and produce pairs of images (input and desired output), it would be desirable if a system programmed by examples[1] [13] could project an image operator to "imitate" her (his) desire. A complete automatic design system is composed of at least three subsystems: (1) a preprocessing DIP system to create the input/output examples, (2) a DIP system that involves statistical and computational learning [32] to project the operator, (3) an application DIP system to integrate the solution to be deployed. A general system like this does not exist but an effort to build it for some classes of operators (called $W$-operators) has been implemented [12]. In this chapter, we review a class of $W$-operators called *aperture*. We begin with a historical review, then we present the idea of designing window operators, we review the idea of aperture operators, and finally we give some examples of applications.

### 2.1.1. Short historical review

The idea of automatic design of signal operators, from the statistical point of view, has its roots in Wiener's and Pugachev's works [41, 50, 51] for the statistical design

[1]The method has been previously known by "automatic programming." We will continue to use the word "automatic" to designate the design because it is shorter than the new nomenclature.

of linear filters. From the mathematical morphology (MM) point of view, the idea has its roots on the representation of the operator by its base [1, 2, 37], an extension of Matheron's decomposition theorem [38]. MM has been successful in providing practical and intuitive solutions to computer vision problems [3, 46, 47] and because the representation theorems, it showed to be an adequate framework for automatic design of image operators. Because it models images or image operators as points of complete lattices, algebraic restrictions on image operators are easily modeled as restrictions on lattices.

Dougherty conceived the idea and showed how to join both approaches to design $W$-operators, first for binary [25] and later for gray-scale operators [26]. The logical character of the binary representations has been presented [19], the optimal design has been treated for gray-scale operators [21] and the subject has been studied and applied in the context of statistical and computational learning [7, 17]. Several problems have been considered for operators between binary images: (1) estimation precision when the operator is estimated by image realizations [22]; (2) optimal filters for resolution conversion [36]; (3) use of prior information [6, 16, 23]; (4) partially restricted filter design [44]; (5) multistage filter design [45].

Increasing signal operators have played a central role in the study of optimal nonlinear filters for gray-scale signals. Optimization of these operators has been characterized via a minimal search space for the morphological basis [26], and a recursive error representation in the context of computational mathematical morphology has been used to estimate the optimal basis [35]. Stack filters, which are defined by a single increasing binary function acting on a gray-scale signal by operating individually on the threshold sets of the signal, form a constrained class of increasing filters that, because they are defined by increasing binary Boolean functions, can be optimized in a way similar to increasing binary filters [14, 33, 53]. Since the same operator is applied at every level, a relatively small number of signals gives a large amount of data for automatic design of the single operator. Further constrained stack filters, such as weighted medians, can be useful when applied to certain kinds of noise degradation [52]. The concept of a stack filter can be extended to have an increasing Boolean function working on binary variables representing the signal in a band of gray levels about a particular gray level [34]. The filter class is larger, but it is still a subclass of the class of all increasing filters. Design of nonincreasing filters has been considered in the context of computational-morphological representation [28], but without constraint, filter design is intractable except for very small windows and a small number of gray levels. Even for binary filters, unconstrained automatic design suffers dramatically as the window size increases owing to practical limits on the amount of training data [20, 22]. Design theory fits nicely into the theories of computational learning and pattern recognition, and therefore learning theory and the error-convergence theorems from pattern recognition can be applied [7, 17]. For large windows or gray-scale filtering, statistical design requires prior information, which typically involves constraining the designed filter to a smaller class of operators, but can also involve other techniques such as design by alteration of a prior-defined filter

[6, 23] or the Bayesian assumption that the characterizing conditional probabilities are random variables possessing prior distributions [16]. An alternative approach has been proposed to design gray-level filters [43]. It proved to be useful for some kinds of restorations but the convergence was not characterized although the algorithm can converge for a nonoptimal filter [15] and the above mentioned problems remain.

A large step towards making the design of gray-level operators feasible via statistical estimation has been done by Barrera and Dougherty [4]. In this work, they introduced the $WK$-operators, a class of $W$-operators that are locally defined and also translational invariant in the gray-level range. This chapter presents the developments achieved since the publication of that paper.

## 2.2. Window operators

Digital images or signals can be represented by functions from a nonempty set $E$ (usually a subset of $Z$ or $Z \times Z$, where $Z$ is the set of integers) to an interval $L$ (usually a nonnegative integer interval), $L = [0, l-1]$, with $l \in Z^+$. Digital images are usually denoted by small caps letters, for instance, $f, g$, or $h$. An arbitrary point of $E$ is usually denoted by $x$ or $t$. The set of all possible images from $E$ to $L$ will be denoted by $L^E$. A mapping $\Psi$ from $L^E$ to $L'^E$ (where $L'$ is a nonnegative interval of $Z$, not necessarily equal to $L$) will be called a *filter* or *operator*. For instance, the usual threshold operator on gray-level images, $\Psi_c(f)(x) = 1$ if $f(x) > c$, or 0 otherwise, $f \in [0,255]^E$, $c \in [0,255]$, and $x \in E$, is an operator that takes gray-level to binary images.

A finite subset $W$ of $E$, customarily containing the origin (of $E$), is called a *window* and the number of points in $W$ is denoted by $|W|$. For instance, the set $W_{1\times 3} = \{-1, 0, 1\}$ is a one line by three-columns window centered at the origin of $E$. A *window configuration*, or simply, a configuration, is a function $u$ from $W$ to $L$, and the space of all possible configurations from $W$ to $L$ will be denoted by $L^W$ or, sometimes, just $D$.

Figure 2.1 shows a $W_{1\times 5}$ window and the functions $u$ resulting from observing $h$ through $W$ at two different points, $t$ and $t+1$, $t \in E$. Configurations usually result from translating a function $h$ by $-t$, and observing the values of $h$ within $W$. Formally, if $W = \{w_1, w_2, \ldots, w_n\}$, $n = |W|$, and we associate the points of $W$ to a vector $(w_1, w_2, \ldots, w_n)$, then a configuration $u$, seen by $W$ when translating $h$ by $t$, can be written as

$$u_{h(t)} = (h(w_1 - t), h(w_2 - t), \ldots, h(w_n - t)). \tag{2.1}$$

Digital signals can be modeled by digital random functions and, in this sense, $u_{h(t)}$ is a realization of a random vector $\mathbf{X} = (X_1, X_2, \ldots, X_n)$, that is, $u_{h(t)} = \mathbf{x} = (x_1, x_2, \ldots, x_n)$, where $\mathbf{x}$ denotes a realization of $\mathbf{X}$.

An important subclass of operators from $L^E$ to $L'^E$ is the class of *$W$-operators*. They are named after window operators because their output at any point $t \in E$

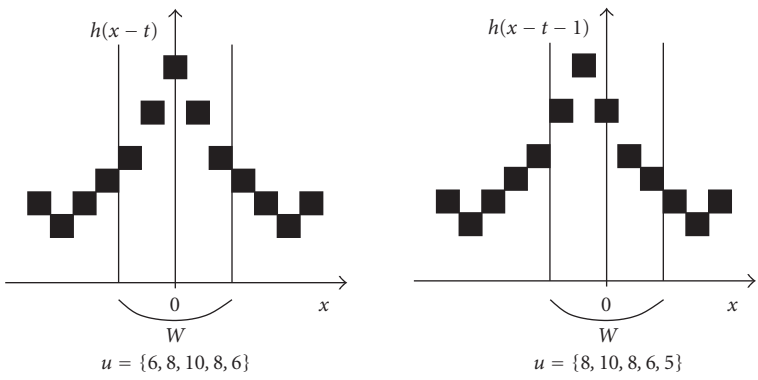



FIGURE 2.1. Two window configurations.

can be decided evaluating the pixel values inside a window centered at $t$. Mathematically, it means that they can be characterized by local functions $\psi$ from $L^W$ to $L'$, called *characteristic functions* [1, 2, 11]. $W$-operators are translation invariant [1] (t.i.) and locally defined [10] (l.d.) within $W$. Being t.i. means that $\psi$ is the same function for any $t \in E$. Being l.d. means that it is enough to know the values inside $W$ to know the output of $\Psi$. Mathematically,

$$\Psi(h)(t) = \psi(h(w_1 - t), h(w_2 - t), \ldots, h(w_n - t)) = \psi(u_{h(t)}). \tag{2.2}$$

From now on, to simplify the notations, we will use $L' = L = Z$, unless when specified differently.

$W$-operators are important to automatic design because by looking to observation pairs $(\mathbf{x}, \psi(\mathbf{x}))$ one can estimate $\Psi$. The main issue is that the number of possible window configurations is exponential to the number points in $W$, that is, $l^{|W|}$, where $l$ is the number of possible gray-levels (usually, $l = 256$), and also that number of possible $W$-operators depends exponentially on the output range $L$, that is, $l^{(l^{|W|})}$.

## 2.3. Aperture operators

The motivation for introducing a new class of operators has come from the inherent difficulty of estimating gray-level $W$-operators, which comes from the size of that class of operators. A possible solution to diminish the number of possible operators is to add new constraints to the class of $W$-operators: being translation invariant and locally defined in the gray-level range by a window $K$, $K = [-k, k] \subseteq Z$. If $K$ is not large, a good statistical estimation is possible. These operators have been initially called *WK-operators* [4] and later *aperture* operators.

Let $u$ be a window configuration ($u \in Z^W$) and let $z$, $z \in Z$. The *vertical translation* of $u$ by $z$ is given by, for any $s \in W$, $u_z(s) = (u + z)(s) = u(s) + z$. An

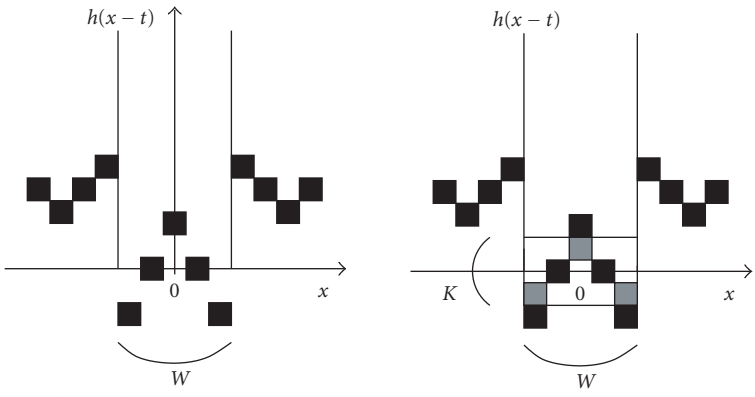



FIGURE 2.2. Window configuration $u_{h(x-t)}$ translated vertically by $-z$ (a) and the clipping to construct $\nu$ (b).

*aperture configuration* is a function $\nu$ from $W$ to $K$ built by translating vertically a window configuration $u$ by an integer $z$ and projecting the points outside $W \times K$ into $W \times K$, that is, the values larger or lesser than $|k|$ are clipped to $k$, or $-k$. We denote the clipping operation by $(\bullet)^*$. Figure 2.2(a) shows the vertical translation of $u$ (Figure 2.1) by $-z = -8$, $u_{-z} = \{-2, 0, 2, 0, -2\}$, and Figure 2.2(b) shows the aperture configuration $\nu = (u_{h(t)-z})^* = \{-1, 0, 1, 0, -1\}$, that is, the clipping to the range $K = [-1, 1]$. Formally, an aperture configuration is given by, for any $s \in W$,

$$\nu(s) = \left(u_{-z}(s)\right)^* = \min\left\{\max\left\{-k, u(s) - z\right\}, k\right\}. \tag{2.3}$$

Let $\Psi$ be a gray-level image operator. We say that $\Psi$ is *invariant by vertical translation* if and only if, for any $z \in Z$ and any $f \in Z^E$, $\Psi(f + z) = \Psi(f) + z$. Figure 2.3 illustrates this concept. Operators that are invariant by spatial and vertical translations are also denominated $T$-operators [31].

Let $\Psi$ be a $W$-operator, $\Psi : L^E \to L^E$, and let $\psi$ be its characteristic function, $\psi : L^W \to L$. $\psi$ is said to be *locally defined* in $K$ if and only if, for any $u \in L^W$ and any $z \in L$,

$$\psi(u) = z + \phi_z(\nu), \tag{2.4}$$

where $\phi_z$ is a function from $K^W$ to $M$ that depends on $z$, $\nu \in K^W$ and $M$, $M = [-m, m]$, $m \in Z^+$. A function $\psi : L^W \to L$ that is locally defined in $K$ and invariant by vertical translation is said to be a *$K$-characteristic function*. This means that $\psi(u) = z + \phi(\nu)$, that is, $\phi$ does not depend on $z$.

Based on the definitions above, we can define a class of operators called $\zeta_O$-aperture. They are $W$-operators that are locally defined in $K$ and are invariant

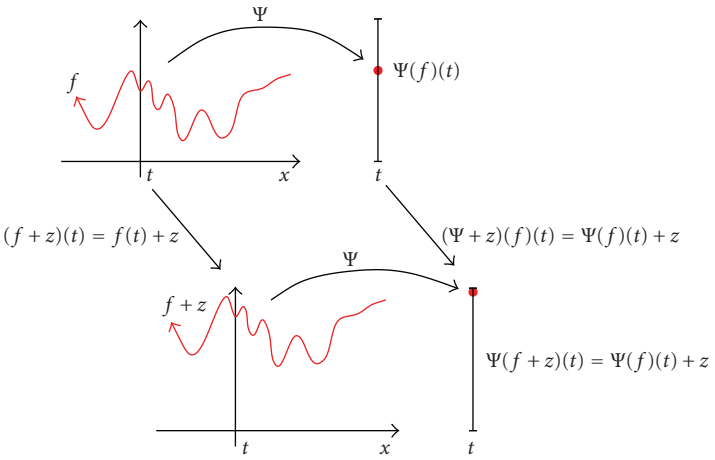



FIGURE 2.3. Invariance by vertical translation.

by translation in $Z$, therefore, they can be written in terms of $K$-characteristic functions.

Let $\zeta_O$ be a function from $L^W$ to $L$, $u = u_{h(t)}$ (to simplify the notation) and let $\phi$ be a function from $K^W$ to $M$. Then, a $\zeta_O$-aperture, $\Psi$, is a mapping from $L^E$ to $L^E$, given by

$$\Psi(h)(t) = \zeta_O(u) + \phi\left(\left(u_{-\zeta_O(u)}\right)^*\right). \tag{2.5}$$

The number of $K$-characteristic functions in the class of $\zeta_O$-aperture is equal to $|M|^{(|K|^{|W|})}$. If $|K|$ and $|M|$ are small, then the quantity of operators from $K^W$ to $M$ is significantly smaller than the amount of operators from $L^W$ to $L$.

Some useful $W$-operators for image segmentation cannot be described by the subclass of $\zeta_O$-apertures. For instance, the threshold operator is not a $\zeta_O$-aperture because it is not invariant by vertical translations, that is, $\Psi_c(f + z) \neq \Psi_c(f) + z$. In order to make the description highly expressive and represent all $W$-operators, let $\zeta_O$ and $\zeta_I$ be two functions from $L^W$ to $L$; $\zeta_O$ associated to vertical translations in the observed image, that is, it is the function that places the aperture window and $\zeta_I$ associated to vertical translations in the ideal image, that is, the function that places the result of characteristic function. An operator $(\zeta_O,\zeta_I)$-aperture, $\Psi$, is a mapping from $L^E$ to $L^E$, given by

$$\Psi(h)(t) = \zeta_I(u) + \phi\left(\left(u_{-\zeta_O(u)}\right)^*\right). \tag{2.6}$$

The definition of a $(\zeta_O,\zeta_I)$-aperture operator generalizes the definition of a $W$-operator when $K = M = L$ and $\zeta_O = \zeta_I = 0$; it also generalizes the definition of operators that classify (labels) points of an image when $\zeta_I = 0$.

### 2.3.1. Design of aperture operators

Designing an image operator from input/output image examples is basically a problem of finding an operator $\Psi$ from pairs of random signals, $(h_i, g_i) \in L^E \times L^E$, $h_i$ to be observed and $g_i$ to be estimated that minimizes an error measure between $\Psi(h_i)(t)$ and $g_i(t)$, the value of the random signals about $t$, for all $t \in E$.

Constraining the class of operators to the class of $W$-operators, $\Psi$ is locally defined by some characteristic function $\psi$, and the problem resumes to find $\psi$, instead of $\Psi$. The statistical design of $W$-operators requires the estimation of the conditional probabilities of finding a certain gray-level $Y$ given that a certain window configuration $\mathbf{X}$ has been observed, for each one of all the possible configurations in a window $W$. The constraining being extended to the class of aperture operators requires then the estimation of $\phi$ instead of $\psi$ and its representation in some computational form. $\phi$ is estimated from training pairs of the form $(\mathbf{x}^*, y^*)$ where $\mathbf{x}^* = (u_{h_i(t)} - \zeta_O(u_{h_i(t)}))^*$ and $y = (g_i(t) - \zeta_I(u_{h_i(t)}))^*$. From the computational and statistical learning area, the first element of the pair is called a *pattern* and the second, its *label*; the pair is called a *training example* if it is used for training the operator and *testing example* if it is used for testing the operator. If a particular pattern is observed $m$ times, then there are $m$ labels (not necessarily equal) associated to it. The final label to be associated to a pattern depends on the error measure one wants to minimize.

The MAE (*mean absolute error*) of $\Psi$, is given by $\mathrm{MAE}(\Psi) = E[Y - \Psi(\mathbf{X})]$, and the MSE (*mean square error*) of $\Psi$ is given by $\mathrm{MSE}(\Psi) = E[(Y - \Psi(\mathbf{X}))^2]$. If one wants to minimize the MSE error of an aperture operator, it is easy to show that, for a given $x^*$, taking the mean of the labels $y^*$ associated to $x^*$ provides an estimate $\phi_N$ of $\phi$. To be more precise, the estimation has to be the discretization of the mean, that is, $\phi_N(x^*) = \lfloor 0.5 + (1/N(x^*)) \sum_{i=1}^{N(x^*)} (y_i^* \mid x^*) \rfloor$. It is important to notice that, if $M$ is not sufficiently large, $\mathrm{MSE}(\Psi) = \mathrm{MSE}(\psi) \neq \mathrm{MSE}(\phi)$. Therefore, if possible, $M$ should be large enough to not have restrictions in the output.

### 2.3.2. Positioning the aperture

For an aperture signal operator $\Psi$ parameterized by $W$, $K$, and $M$, the positioning functions $\zeta_O$ and $\zeta_I$ have a strong influence on the estimation error. Since $\zeta_O$ places the aperture $W \times K$ on some gray-level inside $L$, this position must be chosen so that most of the signal is observed through it and the quantity of signal clipped by the aperture is maximal. Since the error due to the restriction to $K$ depends on probability mass $(\mathbf{X}, Y)$ outside $W \times K$, to reduce it, it may be beneficial to choose the positioning as a function of the observations $X_1, X_2, \ldots, X_n$.

Aperture positioning, where $K = M = [-3, 3]$, is illustrated in Figures 2.4 and 2.5, that show white noise and blurring, respectively. In each figure, parts (a) and (c) give the observed signal and parts (b) and (d) the ideal signals. Signals are shown as solid points and $\zeta_O = \zeta_I = \zeta$. The signal $\times$ shows the center of the aperture and the shadowed points show the projection into the aperture. Figures 2.4(a) and 2.4(b) show the aperture positioned vertically on the observed value;

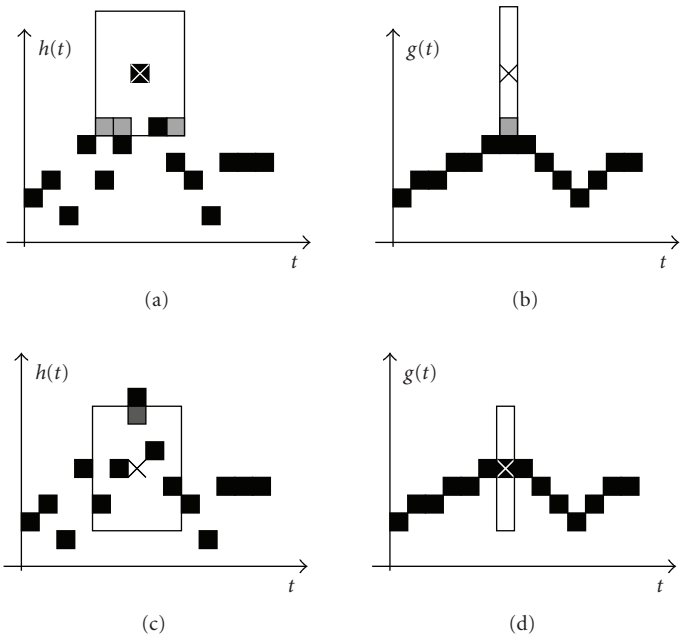



FIGURE 2.4. Aperture placement for additive white noise: (a) aperture on observed signal with placement at observed value; (b) aperture on ideal signal with placement at observed value; (c) aperture on observed signal with placement at median; (d) aperture on ideal signal with placement at median.

$\mathbf{x}^*$ differs from $\mathbf{x}$ by four points and $(y - \zeta(\mathbf{x}))^* \neq y - \zeta(\mathbf{x})$. In Figures 2.4(c) and 2.4(d), the aperture is positioned vertically on the median of $\mathbf{x}$, $\mathbf{x}^*$ differs from $\mathbf{x}$ by one point and $(y - \zeta(\mathbf{x}))^* = y - \zeta(\mathbf{x})$. Figures 2.5(a) and 2.5(b) show the aperture positioned vertically on the observed value; $\mathbf{x} = \mathbf{x}^*$, and $(y - \zeta(\mathbf{x}))^* = y - \zeta(\mathbf{x})$. In Figures 2.5(c) and 2.5(d), the aperture is positioned vertically on the median of $\mathbf{x}$, $\mathbf{x} = \mathbf{x}^*$, and $(y - \zeta(\mathbf{x}))^* \neq y - \zeta(\mathbf{x})$.

### 2.3.3. Representation of the operator

Having associated labels to the observed vectors during training, we need to provide a more compact representation for the operator. Here we need to recognize that the set of observed vectors is often (and in our case will almost always be) a proper subset of the set $\mathcal{X}$ of all possible vectors and that $\mathcal{X}$ is the true domain of $\phi$. It is critical that the representation extends the definition of $\phi$ from the training vectors to $\mathcal{X}$ in a consistent manner, meaning that if $l$ is the final label for a vector, then, upon representation, the operator must assign $l$ to that vector. Given this consistency, the definition of $\phi$ is unambiguous. In machine learning, this extension is called *generalization* because the extension "generalizes" a learned concept. Generalization is important because the performance of a designed operator depends on the way in which it maps vectors not observed during training.

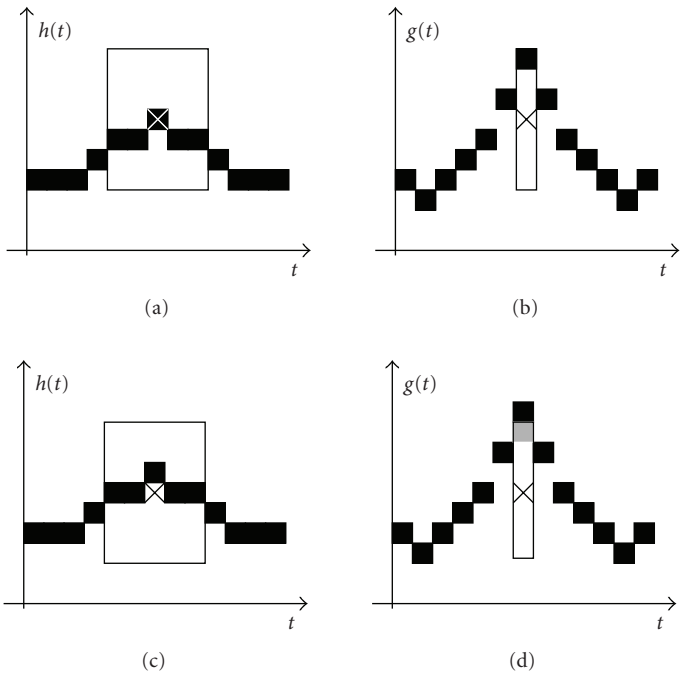



FIGURE 2.5. Aperture placement for blurring: (a) aperture on observed signal with placement at observed value; (b) aperture on ideal signal with placement at observed value; (c) aperture on observed signal with placement at median; (d) aperture on ideal signal with placement at median.

The morphological representation of an operator by its basis is one possible way to achieve generalization and Machine Learning area provides several other representations (decision trees, $k$-neighbors, neural network, SVM, etc.) [32, 39]. The morphological representation is based on maximal intervals of the kernel of the operator. In the following paragraphs we discuss this representation.

Let $a, b \in L^W$. A function $\lambda_{a,b}$ from $L^W$ to $\{0, 1\}$ given by [2]

$$\lambda_{a,b}(u) = \begin{cases} 1 & \text{if } u \in [a, b], \\ 0 & \text{otherwise,} \end{cases} \tag{2.7}$$

for any $u \in L^W$, is called *supgenerating function* of interval $[a, b]$.

Let $\psi$ be a characteristic function from $L^W$ to $L$, and $y \in L$ an arbitrary gray-level. The *kernel* of $\psi$ is given by [2]

$$\mathcal{K}(\psi)(y) = \{u \in L^W : y \leq \psi(u)\}. \tag{2.8}$$

Let $\phi$ be a characteristic function from $K^W$ to $M$, and let $\mathcal{K}(\phi)$ be the kernel of $\phi$. Let $\lambda_{a,b}$ be the supgenerating function of the interval $[a, b]$. Then $\phi$ can be

represented by [4]

$$\phi(v) = \bigvee \left\{ y \in M : \bigvee \{\lambda_{a,b}(v) : [a,b] \subseteq \mathcal{K}(\phi)(y)\} = 1 \right\}, \tag{2.9}$$

for any aperture configuration $v \in K^W$. The aperture operator can be written as

$$\Psi(h)(t) = \zeta_I(u) + \bigvee \left\{ y \in M : \bigvee \{\lambda_{a,b}(v) : [a,b] \in \mathbf{B}(\phi)(y)\} = 1 \right\}, \tag{2.10}$$

where **B** denotes the basis of the operator (the set of maximal intervals of $\mathcal{K}$).

The representation by the basis of an operator is an elegant way to express the operator and, also, it is useful to apply algebraic restrictions to the operator. However, by the time we started working on aperture operators, there was not enough computer power to do the experiments using this representation. The consequence is that, we used decision trees (DT) to represent the operator. They are fast to build, extend the learned operator consistently, and tend to provide good generalization [8, 39, 40]. The idea behind a decision tree is to find a test on one (*parallel DT*) or more of the pattern variables (*oblique DT*), say $T_0$, that partitions the training set into two or more "purer" subsets, that is, subsets of equally labeled examples or at least consisting of examples that are less mixed than before partitioning. A subset is *pure* if all examples in it have the same label. For each nonpure subset we can apply this idea again, finding new tests, say $T_{1.1}, T_{1.2}, \ldots, T_{1.m}$, and new purer subsets. Proceeding recursively for each nonpure subset yields a partition where each part has resulted from a sequence of tests and is pure. Practically, to label a given pattern we have to, starting from the root of the tree, apply the test associated with the current node and follow the edge to the correct child according to the test. The process stops when we reach a leaf, which holds the label to be assigned to the pattern.

For oblique decision trees, we have adapted and used the *OC1 algorithm* [40]. This algorithm improves previous methods [8, 42] by introducing an efficient randomized part in the algorithm to find the coefficients of an oblique decision hyperplane at each tree node if the best axis-parallel hyper-plane is not better. This last condition guarantees that the method will be at least as powerful as standard (axis-parallel) decision tree methods.

### 2.3.4. Deblurring of 2D images

To demonstrate the application of aperture operators to images, we design aperture filters to deblur a random Boolean function model [48] whose primary function is pyramidal. Blurring is accomplished by a $3 \times 3$ nonflat convolution kernel (Figure 2.6(a)). We have used 10 training and 10 test images, each of size $256 \times 256$ points. Figure 2.7 shows parts of an original and a blurred image. Figures 2.8 and 2.9 give plots of the MAE and MSE, respectively, after filtering compared to the number of training examples. Six aperture filters have been tested ($3 \times 3 \times 3, 3 \times 3 \times 5, 17p \times 3, 17p \times 5, 5 \times 5 \times 3, 5 \times 5 \times 5$), where $n \times m \times k$ refers

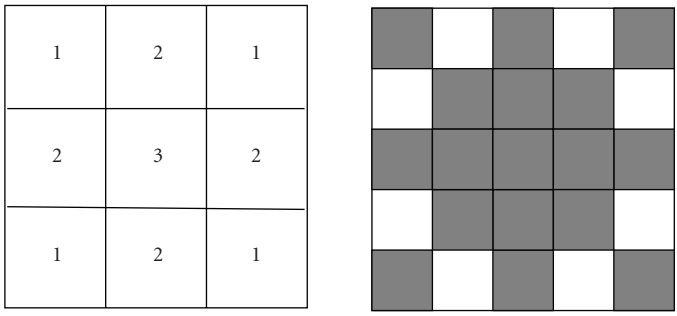



FIGURE 2.6. Convolution kernel and 17-point window.

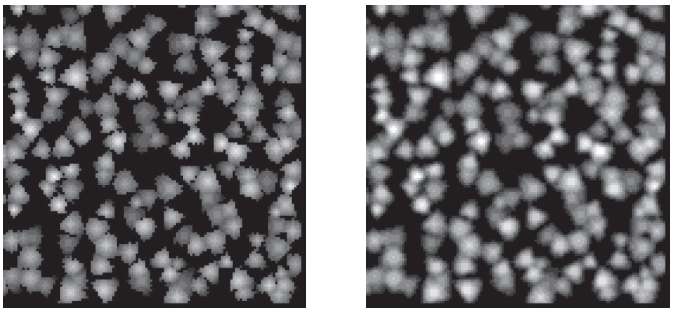
to an aperture and $17p$ refers to the domain window shown in Figure 2.6(b). The aperture giving the lowest MAE is $17p \times 5$. This result is very similar to the $5 \times 5 \times 5$ aperture. All tested filters outperform optimal window restricted linear filter [27] over a $7 \times 7$ point window. The filter giving the lowest MSE is for the $3 \times 3 \times 5$ aperture; however the other aperture filters are not stabilized at 600,000 examples and will continue to improve with more training. Indeed, referring to Figure 2.8 we see that the MAEs of the other aperture filters have not stabilized, and yet at 600,000 examples the larger apertures are already performing better than the smaller apertures relative to MAE.

For visual aspects, the aperture $17p \times 5$ performs better than the other tested filters because it provides superior edge restoration, which is often the case for an optimal nonlinear filter. Figures 2.10, 2.11, 2.12, and 2.13 show part of one test image, the result of the optimal $7 \times 7$ linear filter, the result of the $3 \times 3 \times 5$ aperture filter, and the result of the $17p \times 5$ aperture filter, respectively. These images demonstrate the superior edge restoration of the $17p \times 5$ aperture filter and its superior visual quality. This visual superiority is more striking because we have selected an image portion for which the $7 \times 7$ linear filter slightly outperforms the $17p \times 5$ aperture filter relative to MSE.

The filter outputs differ from the original image fragment at 132, 97, and 78 points (disregarding the border) for the optimal linear, $3 \times 3 \times 5$ aperture, and $17p \times 5$ aperture filters, respectively. The filter outputs differ from the original image fragment by more than one gray level at 28, 17, and 11 points for the optimal $17p \times 5$ aperture, $3 \times 3 \times 5$ aperture, and linear filters, respectively. This explains why the MSE for the $17p \times 5$ aperture filter is not as good for these image fragments.

## 2.4. Envelope aperture

Envelope design of image operators is a hybrid technique that does not rely only on machine design (that involves estimating the optimal filter from sample data) but relies also on human design. The envelope constraint involves using two humanly heuristically designed filters that imposes a lower and an upper bound on



(a) (b)

FIGURE 2.7. Random Boolean function: (a) original image; (b) blurred image.

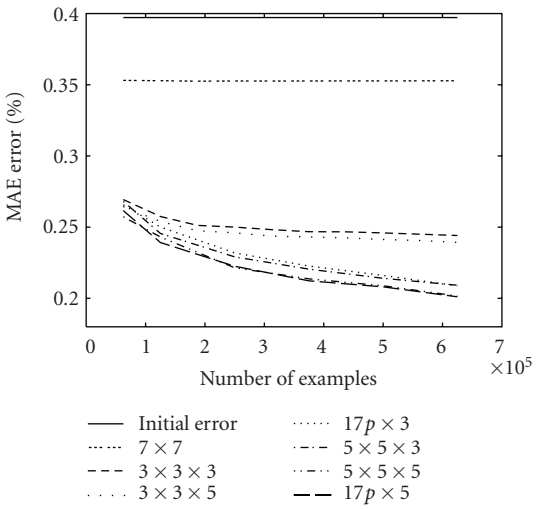



FIGURE 2.8. MAE plots for deblurring random Boolean function.

the designed operator. If the filters are chosen so that the optimal filter lies between them, then there must be an advantage to using the envelope. Unfortunately, if the lower and upper bounds are too far apart, this advantage is negligible (in the extreme, null because one can always choose zero and infinity for the lower and upper bounds). The method has been employed for binary operators [5] and for aperture filters [9].

### 2.4.1. Envelope constraint

Envelope constraint is defined via a pair of gray-level characteristic functions $\alpha, \beta : D \to L$, with $\alpha \leq \beta$. These define the *envelope* $(\alpha, \beta)$ and the *envelope constraint*

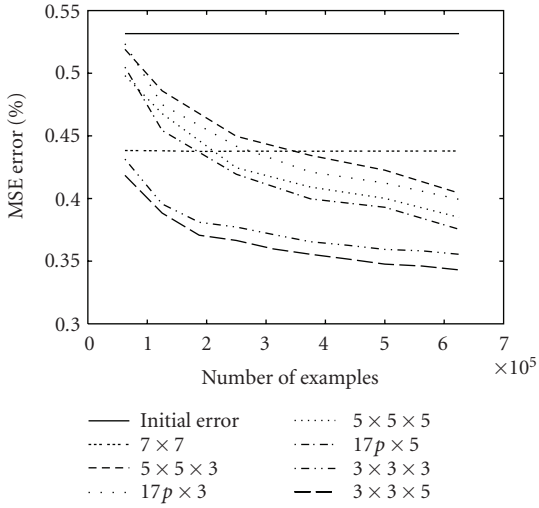



FIGURE 2.9. MSE plots for deblurring random Boolean function.

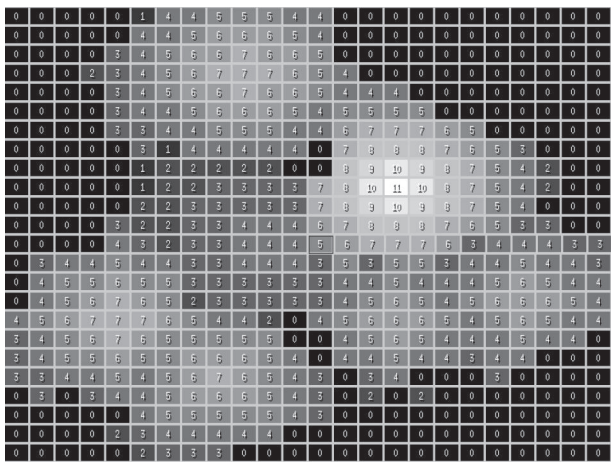


FIGURE 2.10. Fragment of random Boolean realization.

*class*

$$\mathbf{Q}_{\alpha,\beta} = \{\psi \in L^D : \alpha \leq \psi \leq \beta\}. \tag{2.11}$$

Envelope constraint is appropriate when we possess the prior knowledge that $\alpha \leq \psi_{\text{opt}} \leq \beta$, or at least that the inequality approximately holds. In this case, if $\psi$ is a machine-designed characteristic function and $\psi(\mathbf{x}) < \alpha(\mathbf{x})$, then we will use $\alpha(\mathbf{x})$ in place of $\psi(\mathbf{x})$. An analogous decision is made if $\psi(\mathbf{x}) > \beta(\mathbf{x})$. Formally, for each

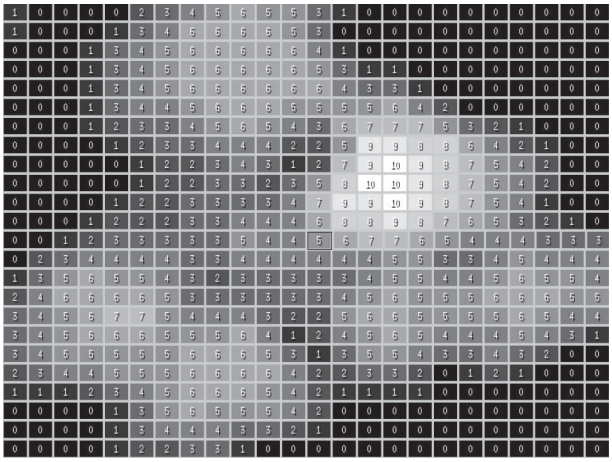


FIGURE 2.11. Output of optimal linear filter applied to fragment.

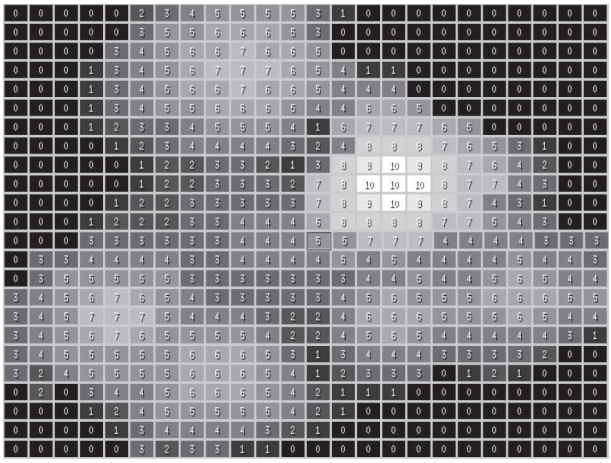


FIGURE 2.12. Output of $3 \times 3 \times 5$ aperture filter applied to fragment.

operator $\psi : D \to L$, we define its *envelope-constrained* operator $\psi_{\text{con}}$ by

$$\psi_{\text{con}}(\mathbf{x}) = (\psi \vee \alpha) \wedge \beta = \begin{cases} \alpha(\mathbf{x}) : \psi(\mathbf{x}) < \alpha(\mathbf{x}), \\ \psi(\mathbf{x}) : \alpha(\mathbf{x}) \le \psi(\mathbf{x}) \le \beta(\mathbf{x}), \\ \beta(\mathbf{x}) : \psi(\mathbf{x}) > \beta(\mathbf{x}). \end{cases} \tag{2.12}$$

According to the definition, $\alpha \le \psi_{\text{con}} \le \beta$, so that $\psi_{\text{con}} \in \mathbf{Q}_{\alpha,\beta}$.

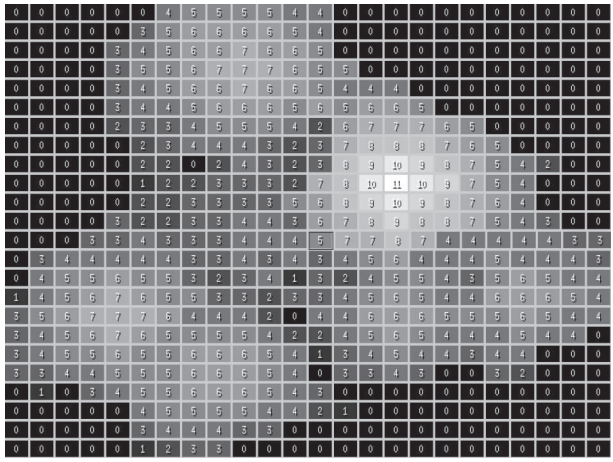


FIGURE 2.13. Output of $17p \times 5$ aperture filter applied to fragment.

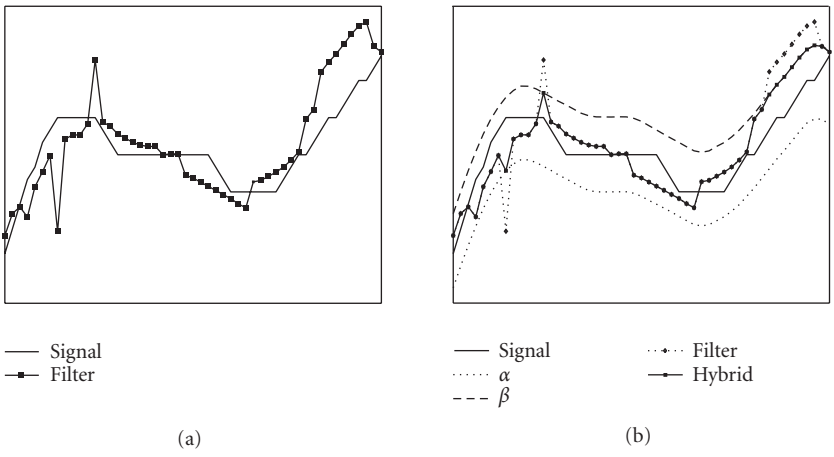



FIGURE 2.14. Envelope example.

Figure 2.14(a) shows a signal and an estimate of this signal (e.g., a filter applied over a noisy sample of this signal). Figure 2.14(b) shows the result of applying an envelope, defined by $\alpha$ and $\beta$, to the designed filter. If the envelope is well designed, as in this case, the hybrid resulting filter will be closer to the ideal signal.

Usually, the envelope is heuristically designed, for example alternating sequential filters or rank-order filters for noise filtering [24, 31]. The envelope can also be designed relative to a simpler statistically optimal filter, such as a linear filter. If $\psi_{\text{lin}}$ is the optimal linear filter, we can define $\alpha = \psi_{\text{lin}} - \epsilon$ and $\beta = \psi_{\text{lin}} + \epsilon$. The smaller the parameter $\epsilon$, the smaller the range $\beta(\mathbf{x}) - \alpha(\mathbf{x}) = 2\epsilon$, and therefore the stronger the constraint on $\psi_{\text{con}}$.

### 2.4.2. Optimality of the constrained optimal filter

A key issue in constrained filter design is the existence of a practical way to design optimal constrained operators. For instance, the solution to this problem has been shown for resolution constraint [18] and increasing binary filters [30, 49]. In the case of gray-level envelopes, the filter $\psi^c_{\text{opt}} = (\psi_{\text{opt}} \vee \alpha) \wedge \beta$, obtained by envelope constraining the optimal filter $\psi_{\text{opt}}$, is the optimal filter in the class $\mathbf{Q}_{\alpha,\beta}$. Its practical import is that it allows us to estimate the optimal constrained filter by constraining the estimation of the unconstrained optimal filter. Besides that, one can prove [9] that for any $\psi \in \mathbf{Q}_{\alpha,\beta}$, the error increase $\Delta(\psi, \psi^c_{\text{opt}})$ of using $\psi$ instead of $\psi^c_{\text{opt}}$ is greater or equal to zero.

### 2.4.3. Precision of estimation under envelope constraints

When designing the optimal operator from pairs of samples, the designed operator is only an approximation of the optimal operator. For different samples, different estimators of the optimal operator are created. Hence, for a sample size of $N$, the goodness of an envelope is measured by the expected advantage, $E[A_{\psi_N}(\alpha,\beta)]$, of the designed operator $\psi_N$, over all possible samples of size $N$. As is typical of constraints, envelope constraint is not beneficial when $N$ is very large. In such cases, it is better to design the optimal operator over the whole class $L^D$ than the optimal constrained operator in $\mathbf{Q}_{\alpha,\beta}$; however, for small samples and a good envelope $(\alpha,\beta)$, the constraint can be beneficial. To quantify asymptotic behavior as $N \to \infty$, we suppose that $\psi_N$ is a strongly consistent estimator of $\psi_{\text{opt}}$, meaning that $\psi_N \to \psi_{\text{opt}}$ almost surely as $N \to \infty$. Strongly consistent estimators are commonly used for digital filter design and one can prove [9] that, if $\psi_N$ is a strongly consistent estimator of $\psi_{\text{opt}}$, then $\limsup_{N\to\infty} E[A_{\psi_N}(\alpha,\beta)] \leq 0$.

Envelope design provides a systematic means by which humanly designed operators can be used to assist in machine design. If the lower and upper bounds are well chosen, then envelope design can be advantageous in the common situation in which there is insufficient data to adequately design filters requiring more than a small number of variables without some form of constraint or prior knowledge. Fundamental theorems have been presented that characterize the advantage of envelope-constrained filters and their relationship to unconstrained optimal filters. Envelope constraint has been applied to noise filtering and deblurring. Envelopes based on rank, morphological, and statistically designed linear operators have been applied.

### 2.4.4. Deblurring 2D images by enveloping

This example shows the use of envelope apertures to deblurring of modified Boolean model images like the ones presented in a previous section. Twenty different smooth functions have been generated and summed to each of the 10 ideal images and to each of the 10 test images of the first experiment. Each smooth function has been generated by choosing four random points at the corners of

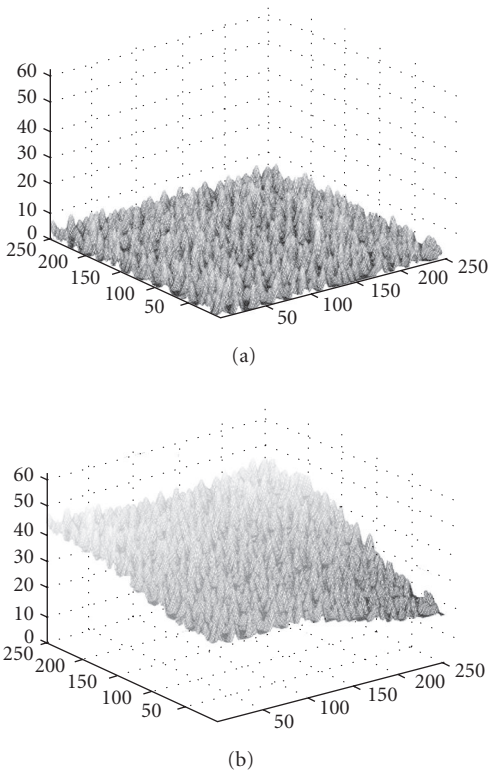



FIGURE 2.15. Surface of the random Boolean function: one image (a) combined with surface (b).

the image (uniform distribution between 0 and 50) and generating a smooth surface connecting those points. The final image is formed by placing small pyramids at different gray-level positions on a smooth image. The idea here is to model a pyramidal texture on a variable background. Figure 2.15 shows a 3D view of one original image (a) and the same image with the correspondent surface added (b) to it. The blurring has been generated using the same convolution kernel as previously and these new pairs (blurring and ideal) have been used to design deblurring operators. Figure 2.16(a) shows part of one image resulting from adding one of the previous ideal images to one surface, and Figure 2.16(b) shows the profile of a linear cut. Figure 2.16(c) shows the same image after blurring, with the profile cut being shown in part (d).

In this experiment, the performances of aperture, linear, and two envelope filters have been analyzed. The first envelope filter, $\text{Env}_l$, is derived from the optimal $9 \times 9$ window linear filter $\psi_{\text{lin}}$, where $\alpha$ and $\beta$ are given by

$$\begin{aligned} \alpha_l(f) &= \psi_{\text{lin}}(f) - 1, \\ \beta_l(f) &= \psi_{\text{lin}}(f) + 1. \end{aligned} \tag{2.13}$$

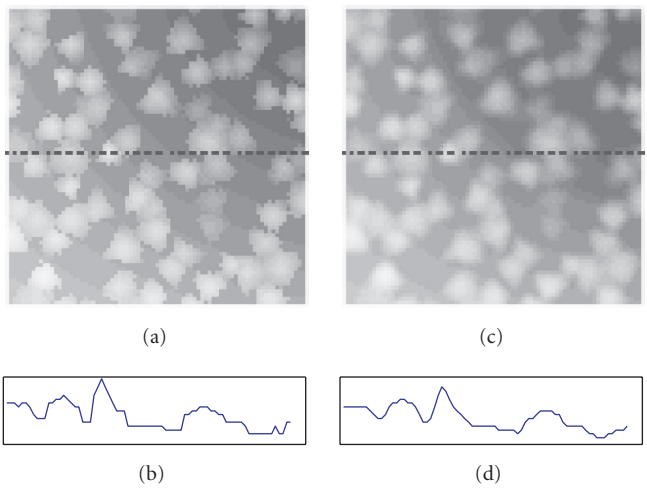



FIGURE 2.16. Random Boolean function: combined images (a) and a profile (b). Blurred image (c) and a profile (d).

TABLE 2.1. MSE errors for deblurring noisy signals.

| Train | Aperture | Linear | $\text{Env}_l$ | $\text{Env}_r$ |
|---|---|---|---|---|
| 1 | 0.47807 | 0.46528 | 0.41496 | 0.45838 |
| 2 | 0.4472 | 0.46545 | 0.39822 | 0.43305 |
| 4 | 0.41817 | 0.46521 | 0.38359 | 0.40795 |
| 6 | 0.40277 | 0.46579 | 0.37533 | 0.39452 |
| 8 | 0.39389 | 0.4657 | 0.37156 | 0.38714 |
| 10 | 0.38657 | 0.46579 | 0.36806 | 0.38108 |

The second envelope filter, $\text{Env}_r$, is defined by the minimum and maximum rank operators. $\alpha$ and $\beta$ are given by

$$\begin{aligned} \alpha_r(f) &= \varepsilon_B(f), \\ \beta_r(f) &= \delta_B(f), \end{aligned} \tag{2.14}$$

where $B$ is a $3 \times 3$ structuring element, $\varepsilon_B$ and $\delta_B$ are, respectively, an erosion and a dilation by $B$. The aperture filter is over a $3 \times 3$ window with range $K = [-5, 5]$.

Table 2.1 and Figure 2.17 show the MSE values of the designed aperture, linear, and envelope filters. The results confirm the advantage of the envelope. The improved performance of the envelope filter $\text{Env}_r$ over that of the aperture filter is slight; however, the improved performance of the envelope filter $\text{Env}_l$ is substantial, especially for a small number of training images. Even for ten training images, both envelope filters provide improved performance.

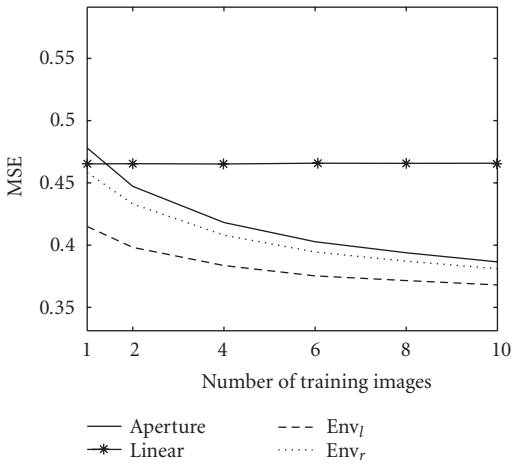



FIGURE 2.17. MSE for Boolean model images.

## 2.5. Multiresolution aperture

The design of a resolution constrained operator is based on adequately combining information from two or more different resolutions to facilitate design. This section reviews multiresolution design of aperture operators for gray-scale images. Spatial resolution constraint, range resolution constraint and the combination of both constraints are characterized, and the error increase by using the constrained filter in place of the optimal unconstrained one is analyzed. Pyramidal multiresolution design involves applying the resolution constraint approach hierarchically, from the higher to the lower resolution space. This approach is also characterized and its error increase analyzed. Two applications for deblurring are shown and compared to optimal linear filters. The results confirm the usefulness of the approach.

### 2.5.1. Down-sampling resolution constraint

Let $W_0$, $W_1$ be two windows such that $W_1 \subset W_0 \subset E$, let $L = [0,\ldots,l-1]$ be the gray-level range, and let $D_1 = L^{W_1}$ and $D_0 = L^{W_0}$ be the configuration spaces over $W_1$ and $W_0$, respectively. Let $\rho : D_0 \to D_1$ be a *subsampling* mapping that assigns to each configuration $\mathbf{x} \in D_0$ a configuration $\mathbf{z} = \rho(\mathbf{x})$, which is a vector whose values are those of $\mathbf{x}$ over the subwindow $W_1$. Let $L^{D_0}$ be the space of all functions from $D_0$ to $L$. The mapping $\rho$ defines a constraint $\mathbf{Q} \subset L^{D_0}$ by: $\psi \in \mathbf{Q}$ if and only if $\rho(\mathbf{x}) = \rho(\mathbf{x}') \Rightarrow \psi(\mathbf{x}) = \psi(\mathbf{x}')$, for any $\mathbf{x}, \mathbf{x}' \in D_0$. The size of the configuration space is reduced for the constrained filter because $|D_1| = |L|^{|W_1|}$ and $|D_0| = |L|^{|W_0|} = |L|^{(|W_0|-|W_1|+|W_1|)} = |L|^{(|W_0|-|W_1|)}|D_1|$. Using the mapping $\rho$, an optimal constrained operator in $L^{D_0}$ is estimated via the optimal operator in $L^{D_1}$.

For range resolution constraint, consider window $W \subset E$ and integer intervals $L_0 = L = [0,\ldots,l_0 - 1]$ and $L_1 = [0,\ldots,l_1 - 1]$, where $l_0$ is an even number and $l_1 = l_0/2$, be the gray-level ranges for the observation spaces $D_0 = L_0^W$ and $D_1 = L_1^W$. Let $\rho : D_0 \to D_1$ be a *quantization*, that is, a mapping that assigns to each configuration $\mathbf{x} \in D_0$ a configuration $\mathbf{z} = \rho(\mathbf{x}) = \lfloor \mathbf{x}/2 \rfloor \in D_1$, where $\lfloor \bullet \rfloor$ denotes the floor of $\bullet$. In this case, a configuration $\mathbf{x} \in D_0$ with up to $l_0$ gray-levels is quantized to half the gray-levels to obtain the configuration $\mathbf{z} \in D_1$ with up to $l_0/2$ gray-levels. To estimate the optimal constrained filter in $\mathbf{Q}$, $\mathbf{Q} \subset L^{D_0}$ defined by $\rho$, the number of parameters to be estimated is significantly smaller than for the optimal filter in $L^{D_0}$, because $|D_1| = |L_1|^{|W|}$ and $|D_0| = |L_0|^{|W|} = (2|L_1|)^{|W|} = 2^{|W|}|L_1|^{|W|} = 2^{|W|}|D_1|$. Analogous comments and analysis apply to other quantizations than $l_1 = l_0/2$.

Spatial and range constraints can be combined to obtain a new space that is a constraint of $D = L^W$. Consider windows $W_1 \subset W_0$ and let $L_0 = L = [0,\ldots,l_0 - 1]$ and $L_1 = [0,\ldots,l_1 - 1]$, with $l_0$ an even number and $l_1 = l_0/2$. Let $D_0 = L_0^{W_0}$, $D_{01} = L_0^{W_1}$, $D_1 = L_1^{W_1}$, and $\rho_1 : D_0 \to D_{01}$ be defined by spatial resolution constraint from $W_0$ to $W_1$. Let $\rho_2 : D_{01} \to D_1$ define a resolution constraint over $W_1$ by $\rho_2(\mathbf{x}) = \lfloor \mathbf{x}/2 \rfloor$, for any $\mathbf{x} \in D_{01}$. The composition $\rho : D_0 \xrightarrow{\rho_1} D_{01} \xrightarrow{\rho_2} D_1$ is a mapping from $D_0$ to $D_1$ which takes a configuration $\mathbf{x} \in L_0^{W_0}$, subsamples and quantizes it to obtain a configuration $\mathbf{z} = \rho(\mathbf{x}) \in L_1^{W_1}$. The mapping $\rho$ defines a resolution constraint on the space $L^{D_0}$.

Down-sampling can be defined more generally [18]. Consider a subsampling where each pixel $\mu_i \in W_1$, $1 \le i \le |W_1|$, corresponds to a subwindow $W_{\mu_i} \subset W_0$, where the subwindows form a partition of $W_0$. A down-sampling is defined via mappings $\zeta_i : L_0^{W_{\mu_i}} \to L_1$ that assign values to each pixel $\mu_i \in W_1$ as functions of the configuration $\mathbf{x} \in D_0$ restricted to $W_{\mu_i}$. Formally, $\rho : D_0 \to D_1$ is defined by

$$\mathbf{z} = \rho(\mathbf{x}) = \left(\zeta_1\left(\mathbf{x}|_{W_{\mu_1}}\right),\ldots,\zeta_{|W_1|}\left(\mathbf{x}|_{W_{\mu_{|W_1|}}}\right)\right). \tag{2.15}$$

This definition extends all the previous ones. For example, if $L_1 = L_0 = L$ and $W_{\mu_i}$ consists of the single pixel in $W_0$ that is in the same place as $\mu_i$, then the mappings are equivalent to the previously defined spatial resolution constraint.

### 2.5.2. Resolution constraint

To define resolution constraint in its general form, consider two configuration spaces $D_0$ and $D_1$ related by a surjective mapping $\rho : D_0 \to D_1$. $\rho$ determines an equivalence relation on $D_0$ by $\mathbf{x} \sim \mathbf{x}'$ if and only if $\rho(\mathbf{x}) = \rho(\mathbf{x}')$. Therefore, for each $\mathbf{z} \in D_1$, there exists an equivalence class $C[\mathbf{z}]$, given by $C[\mathbf{z}] = \rho^{-1}(\mathbf{z})$. For gray-level operators over a fixed range $L$, optimal design is relative to the product space $L \times D_0$ with probability mass $P(y,\mathbf{x})$, $y \in L$, $\mathbf{x} \in D_0$. A probability mass is induced on $D_1$ by $P(\mathbf{z}) = P(\rho^{-1}(\mathbf{z}))$ and on $L \times D_1$ by

$$P(y,\mathbf{z}) = P\big(\{y\} \times \rho^{-1}(\mathbf{z})\big). \tag{2.16}$$

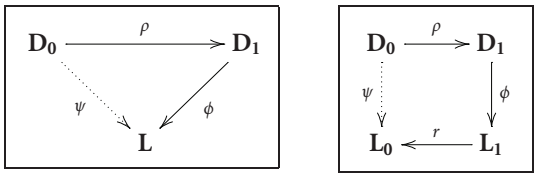



FIGURE 2.18. Commutative diagrams. (Reproduced from Journal of Mathematical Imaging and Vision, Vol.3, 2002, pages 199–222, "Multiresolution design of aperture operators," Roberto Hirata Junior, Marcel Brun, Junior Barrera, and Edward R. Dougherty, Figures 4 and 5 © 2002 with kind permission of Springer Science and Business Media.)

As in the binary case [18], an operator $\phi$ on $D_1$ induces an operator $\psi_\phi$ on $D_0$ by $\psi_\phi(\mathbf{x}) = \phi(\rho(\mathbf{x}))$ (Figure 2.18(a)). $\psi_\phi$ is *spatially resolution constrained*, in accordance with the function $\rho$: if $\rho(\mathbf{x}) = \rho(\mathbf{x}')$ then $\psi_\phi(\mathbf{x}) = \psi_\phi(\mathbf{x}')$.

Conversely, if $\psi$ is any operator on $D_0$ satisfying the resolution constraint, then it induces an operator on $D_1$ by $\phi_\psi(\mathbf{z}) = \psi(\mathbf{x})$, where $\mathbf{x}$ is any vector in $C[\mathbf{z}]$. Let $L^{D_0}$ and $L^{D_1}$ be the classes of gray-level operators on $D_0$ and $D_1$, respectively. The mapping $\phi \to \psi_\phi$ defines an injection $L^{D_1} \to L^{D_0}$. If $\mathbf{Q}$ is the subset of $L^{D_0}$ composed of operators satisfying the resolution constraint, then the mapping $\psi \to \phi_\psi$ defines a bijection $\mathbf{Q} \to L^{D_1}$ whose inverse is given by the mapping $\phi \to \psi_\phi$. This bijective relation allow us to identify operators on $D_1$ with resolution constrained operators on $D_0$. A key point here is whether an optimal operator on $D_1$ can be associated to an optimal resolution constrained operator on $D_0$. If this is the case, then consistent estimators for the optimal operators in $L^{D_1}$ will induce consistent estimators for the optimal operator in $\mathbf{Q}$. The following theorem shows this is always true for any loss function.

Theorem 2.1 (error preservation). *Let $\ell : L \times L \to \mathcal{R}^+$, be a loss function used to define the risk functions $R : L^{D_0} \to \mathcal{R}^+$, $R(\psi) = E[\ell(Y, \psi(\mathbf{X}))]$, and $R : L^{D_1} \to \mathcal{R}^+$, $R(\phi) = E[\ell(Y, \phi(\mathbf{Z}))]$. Then $R(\psi) = R(\phi_\psi)$, for all $\psi \in \mathbf{Q}$.*

The proof can be found in [29].

An immediate consequence of the above theorem is that, under spatial resolution constraint, MSE and MAE are preserved by the mapping $\psi \to \phi_\psi$ and, therefore, the optimal filter on $D_1$ induces the optimal resolution constrained filter on $D_0$.

### 2.5.3. Nonpreservation of error in range resolution constraint

The error-preservation theorem has been demonstrated under the assumption of resolution constraint in both the spatial domain and range, in particular with $D_0 = L_0^{W_0}$ and $D_1 = L_1^{W_1}$, and the condition that both $\psi$ and the induced mapping $\phi_\psi$ possess the same range $L$, that is, $\psi \in L^{D_0}$ and $\phi_\psi \in L^{D_1}$. A natural situation is when $\psi$ is being used to estimate the value $Y$ of an ideal image based

on the random vector $\mathbf{X}$ in an observation window, with $L = L_0$. Owing to error preservation, the optimal filter in $\mathbf{Q}$ is found by determining the optimal filter in $L^{D_1}$. Now consider an alternate situation in which resolution constraint is not only applied to the observation $\mathbf{X}$ but to the ideal value $Y$. Here, $\psi : D_0 \to L_0$, there exists a surjection $s : L_0 \to L_1$, and $\phi_\psi : D_1 \to L_1$. $\psi(\mathbf{X})$ serves as an estimator of $Y$, whereas $\phi_\psi(\mathbf{Z})$ serves as an estimator of $s(Y)$. The question now is, if $\phi$ is the optimal estimator for $s(Y)$ in $L^{D_1}$, does this guarantee that $\psi_\phi$ is the optimal estimator for $Y$ in $\mathbf{Q}$? As we will see, it does not.

To see the problem, consider the simplified situation in which there is no spatial constraint, so that there is a single window $W$, $D_0 = L_0^W$, and $D_1 = L_1^W$. Consider a surjection $s : L_0 \to L_1$ and a corresponding injection $r : L_1 \to L_0$, in which case, the composition $s \circ r$ is the identity on $L_1$. The pair $(r, s)$ defines a constraint class $\mathbf{Q} \in L_0^{D_0}$ in accordance with the constraint mapping $\rho(\mathbf{x}) = (s(x_1), s(x_2), \ldots, s(x_{|W|}))$. $\psi \in \mathbf{Q}$ if and only if $\rho(\mathbf{x}) = \rho(\mathbf{x}')$ implies $\psi(\mathbf{x}) = \psi(\mathbf{x}')$, and $\psi(\mathbf{x}) \in r(L_1)$ for any $\mathbf{x} \in D_0$. An operator $\phi \in L_1^{D_1}$ induces a resolution-constrained operator $\psi_\phi \in L_0^{D_0}$ by $\psi_\phi(\mathbf{x}) = r(\phi(\rho(\mathbf{x})))$ (Figure 2.18(b)).

On the other hand, if $\psi$ is a resolution constrained operator in $L_0^{D_0}$, then it induces an operator $\phi_\psi \in L_1^{D_1}$ by $\phi_\psi(\mathbf{z}) = s(\psi(\mathbf{x}))$ for any $\mathbf{x} \in D_0$ such that $\rho(\mathbf{x}) = \mathbf{z}$. The mapping $\phi \to \psi_\phi$ is a bijection from $L_1^{D_1}$ onto $\mathbf{Q}$ having inverse $\psi \to \phi_\psi$.

Now consider the special case in which $s(j) = \lfloor j/2 \rfloor$ and $r(i) = 2i$. The MSE for $\psi$ as an estimator of $Y$ and $\phi_\psi$ as an estimator of $s(Y) = \lfloor Y/2 \rfloor$ are related by

$$\mathrm{MSE}\left(\phi_\psi\right) = \frac{1}{4}\,\mathrm{MSE}(\psi) + \frac{1}{4}\sum_{\mathbf{x}\in D_0}\sum_{i=0}^{l_1-1} 2\left[\left(\psi(\mathbf{x}) - 2i\right) - \frac{1}{2}\right]P(2i+1, \mathbf{x}). \tag{2.17}$$

If $P(2i + 1, \mathbf{x}) = 0$ for any $i \in L_1$, meaning the probability mass is concentrated in $r(L_1)$, then $\mathrm{MSE}(\phi_\psi) = (1/4)\,\mathrm{MSE}(\phi)$ and the optimal operator in $L_1^{D_1}$ induces the optimal operator in $\mathbf{Q}$; otherwise, it need not.

### 2.5.4. Multiresolution design

In this section, we show how spatial resolution constraint can be iterated and some consequences of this approach. Consider windows $W_2 \subset W_1 \subset W_0$, where the configurations for $W_1$ and $W_2$ are obtained by down-sampling from the configurations for $W_0$ and $W_1$, respectively. Let $D_0 = L^{W_0}$, $D_1 = L^{W_1}$, and $D_2 = L^{W_2}$ be the configuration spaces for the windows $W_0$, $W_1$, and $W_2$. Let $\rho_1 : D_0 \to D_1$ and $\rho_2 : D_1 \to D_2$ be the down-sampling mappings defined by $\mathbf{z} = \rho_1(\mathbf{x})$ and $\mathbf{v} = \rho_2(\mathbf{z})$. The mappings $\rho_1$ and $\rho_2$ define a mapping $\rho_{12} : D_0 \to D_2$ by $\rho_{12} = \rho_2 \circ \rho_1$.

More generally, let $D_0$, $D_1$, and $D_2$ be configuration spaces related by the surjective mappings $\rho_1 : D_0 \to D_1$ and $\rho_2 : D_1 \to D_2$. $\rho_1$ induces an equivalence relation on $D_0$ by $\mathbf{x} \sim_1 \mathbf{x}'$ if and only if $\rho_1(\mathbf{x}) = \rho_1(\mathbf{x}')$. This equivalence relation defines a partition $P_1$ of the space $D_0$, where $P_1 = \{C[\mathbf{z}] : \mathbf{z} \in D_1\}$ and $C[\mathbf{z}] = \rho_1^{-1}(\mathbf{z})$. In the same way, $\rho_2$ induces a partition $\{C[\mathbf{v}] : \mathbf{v} \in D_2\}$ on the space $D_1$. The mapping $\rho_{12} : D_0 \xrightarrow{\rho_1} D_1 \xrightarrow{\rho_2} D_2$ induces another partition $P_2$ on

$D_0$ by $\mathbf{x} \sim_2 \mathbf{x}'$ if and only if $\rho_{12}(\mathbf{x}) = \rho_{12}(\mathbf{x}')$. This partition is coarser than $P_1$, in the sense that $\rho_{12}^{-1}(\mathbf{v}) = \cup\{C[\mathbf{z}] : \mathbf{z} \in C[\mathbf{v}]\}$. Treating the analysis in a general setting, multiresolution analysis for filter design results from recursive partitioning the configuration spaces $D_0, D_1, D_2, \ldots$, relative to recursively applied resolution mappings $\rho_1, \rho_2, \ldots$, where $\rho_k : D_{k-1} \to D_k$. For configuration space $D_k$, there corresponds a resolution constrained optimal filter $\psi_k$. The best filtering resolution for a sample size $N$ is the one for which $\mathrm{MSE}(\psi_{k,N})$ is minimal.

### 2.5.5. Pyramidal design of optimal resolution constrained operators

The resolution constrained design approach presented so far produces a function $\psi_1$ on $D_1$ and applies it on $D_0$ (where $D_1$ is a constrained space of $D_0$) by $\psi_0(\mathbf{x}) = \psi_1(\mathbf{z})$ if $\mathbf{z} = \rho(\mathbf{x})$, where $\rho : D_0 \to D_1$. That means that, even having data to estimate the filter relative to $D_0$, we only use data relative to $D_1$ via $\rho$.

An alternative approach is to use data from both spaces. If we have a good estimate of $p_{\mathbf{x}}$ (the probability distribution of $\mathbf{x}$) and $\psi_{0,N}(\mathbf{x}) \neq \psi_{1,N}(\mathbf{z})$, then it would be prudent to use $\psi_{0,N}(\mathbf{x})$ in place of $\psi_{1,N}(\mathbf{z})$. On the other hand, if we have a poor estimate, or no estimate, of $p_{\mathbf{x}}$, but a better estimate of $p_{\rho(\mathbf{x})}$, then it can be beneficial to use $\psi_{1,N}(\mathbf{z})$. An operator designed in this way is called a *multiresolution operator* because rather than applying only $\psi_{0,N}$ for all $\mathbf{x}$, or applying only $\psi_{1,N}$ for all $\mathbf{z}$, the precision of the probability estimates are considered and the function is chosen accordingly. In the simplest case, requiring only that $\mathbf{x}$ be observed at least once, the multiresolution estimator is given by

$$\psi_{(0,1),N}(\mathbf{x}) = \begin{cases} \psi_{0,N}(\mathbf{x}) & \text{if } N(\mathbf{x}) > 0, \\ \psi_{1,N}(\mathbf{x}) & \text{if } N(\mathbf{x}) = 0. \end{cases} \tag{2.18}$$

This idea can be repeated for any number of constraints, applied iteratively by a sequence of resolution constraint operators $\rho_1, \rho_2, \ldots, \rho_m$, ultimately, until the size of $W_i$ ($W_i \subset W_{i-1}$) is 1 and the size of $L_i$ ($L_i \subset L_{i-1}$) is 2. In this case, the multiresolution estimator is given by

$$\psi_{(0,\ldots,m),N}(\mathbf{x}) = \begin{cases} \psi_{0,N}(\mathbf{x}) & \text{if } N(\mathbf{x}) > 0, \\ \psi_{1,N}(\mathbf{x}) & \text{if } N(\mathbf{x}) = 0,\ N(\rho_1(\mathbf{x})) > 0, \\ \quad\vdots & \\ \psi_{m-1,N}(\mathbf{x}) & \text{if } N(\mathbf{x}) = \cdots N(\rho_{m-2}(\mathbf{x})) = 0,\ N(\rho_{m-1}(\mathbf{x})) > 0, \\ \psi_{m,N}(\mathbf{x}) & \text{if } N(\mathbf{x}) = 0, \ldots, N(\rho_{m-1}(\mathbf{x})) = 0. \end{cases} \tag{2.19}$$

Figure 2.19 shows an example where $\rho_0$ maps each four pixels from window $W_0$ to one pixel in $W_1$ and $\rho_1$ maps each two pixels from window $W_1$ to one pixel in $W_2$. This is a typical case for binary or gray-scale spatial multiresolution design of operators. A slight modification of this approach requires that $N(\mathbf{x})$ be sufficiently large (i.e., $\psi_{(0,1),N}(\mathbf{x}) = \psi_{0,N}(\mathbf{x})$, if $N(\mathbf{x}) > \alpha$, where $\alpha \in Z^+$).

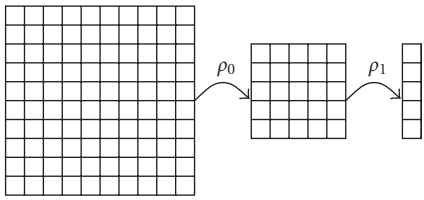



FIGURE 2.19. Mapping windows for multiresolution operators.

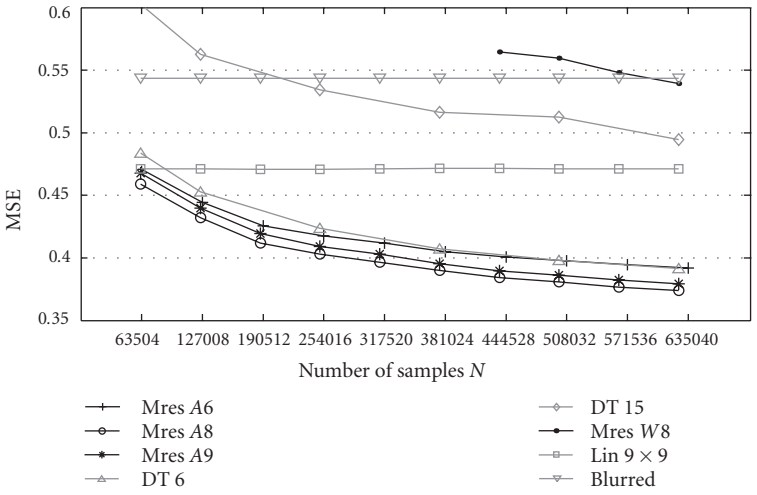



FIGURE 2.20. MSE error comparison: large range. (Reproduced from Journal of Mathematical Imaging and Vision, Vol.3, 2002, pages 199–222, "Multiresolution design of aperture operators," Roberto Hirata Junior, Marcel Brun, Junior Barrera, and Edward R. Dougherty, Figure 18 © 2002 with kind permission of Springer Science and Business Media.)

Although the optimal filter at the higher resolution is better, the designed filter using multiresolution can outperform the standard designed filter owing to better probability estimates. In this sense, the pyramidal multiresolution design is a type of machine learning algorithm, where there is induction (generalization) [39] given by the successive constraints.

### 2.5.6. Deblurring 2D images by multiresolution

This application shows the use of multiresolution apertures to deblurring the modified Boolean model images previously shown. Figure 2.20 shows the MSE errors for some of the best designed operators in each class. It is constructed analogously to previous experiments. The curve labeled *blurred* is the error between the blurred and the ideal images. The curve labeled Lin $9 \times 9$ is the result of the optimal linear for a $9 \times 9$ points window. The curves labeled Mres $A6$, Mres $A8$, Mres $A9$, and Mres $W8$ show the MSE errors for multiresolution aperture and

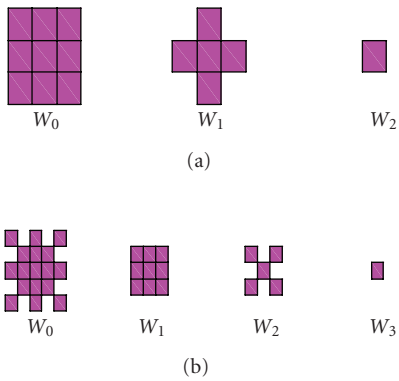



FIGURE 2.21. Pyramids for experiments "6" and "8." (Reproduced from Journal of Mathematical Imaging and Vision, Vol.3, 2002, pages 199–222, "Multiresolution design of aperture operators," Roberto Hirata Junior, Marcel Brun, Junior Barrera, and Edward R. Dougherty, Figure 19 © 2002 with kind permission of Springer Science and Business Media.)

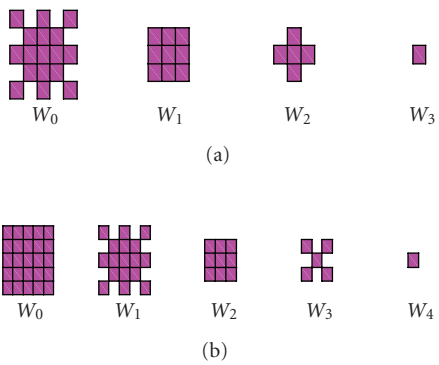



FIGURE 2.22. Pyramids for experiments "9" and "11." (Reproduced from Journal of Mathematical Imaging and Vision, Vol.3, 2002, pages 199–222, "Multiresolution design of aperture operators," Roberto Hirata Junior, Marcel Brun, Junior Barrera, and Edward R. Dougherty, Figure 20 © 2002 with kind permission of Springer Science and Business Media.)

$W$-operators, respectively, designed using the pyramids shown in Figure 2.21(a) (Mres $A6$), Figure 2.21(b) (Mres $A8$ and Mres $W8$[2]) and Figure 2.22(a) (Mres $A9$). Each aperture in the pyramid has the same gray-scale range $K$, with $k = 5$. The curves labeled DT6 and DT15 show the MSE for the aperture operators designed using the square $3 \times 3$ window and the window of 17 points inside a $5 \times 5$ point window.

The best performing multiresolution $W$-operator is for Mres $W8$, and its performance is very poor, with its MSE being worse than the MSE of the blurred

[2]This one for comparison purposes.

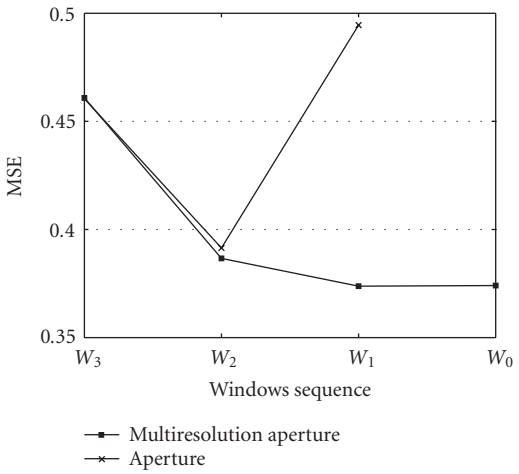



FIGURE 2.23. MSE error comparison: multiresolution × nonmultiresolution. (Reproduced from Journal of Mathematical Imaging and Vision, Vol.3, 2002, pages 199–222, "Multiresolution design of aperture operators," Roberto Hirata Junior, Marcel Brun, Junior Barrera, and Edward R. Dougherty, Figure 21 © 2002 with kind permission of Springer Science and Business Media.)

image until $N = 635040$. The large gray range makes the cost of design exorbitant. The nonmultiresolution aperture filter DT15 outperforms Mres $W8$, but still requires too much sample data. Much better performance is achieved by the nonmultiresolution aperture filter DT6. The multiresolution aperture filter Mres $A6$ outperforms DT6 for smaller $N$, but DT6 catches up at $N = 381024$. The key point is that multiresolution aperture filters Mres $A8$ and Mres $A9$ outperform all multiresolution $W$-filters and nonmultiresolution aperture filters. The only difference between Mres $A8$ and Mres $A9$ is the window $W_2$ in their pyramids. In both cases window $W_2$ has 5 points, but experimentally the diagonal cross used for Mres $A8$ works better than the horizontal-vertical cross used for Mres $A9$ for the image model used here. A final note of interest is that the three multiresolution aperture filters considered in the figure all outperform the optimal linear filter for relatively small sample sizes. This demonstrates the ability of learned nonlinear filters to significantly outperform linear filters for deblurring large-range gray-scale images.

The manner in which pyramidal multiresolution design provides improved performance for large windows is illustrated in Figure 2.23, which shows MSE curves for multiresolution aperture operators and nonmultiresolution aperture operators designed with the same number of samples. The multiresolution pyramid is composed of the windows $W_0, W_1, \ldots, W_4$ shown in Figure 2.24. For instance, the sequence labeled $W_2$ is composed of the windows $W_2$, $W_3$, and $W_4$. The error curve shows that the best aperture operator for the given design samples is $W_2$. The error increases for larger or smaller windows. The error decreases for multiresolution aperture operators as more levels are added to the pyramid.

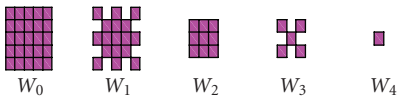



FIGURE 2.24. Pyramids: $\{W_3, W_4\}$, $\{W_2, W_3, W_4\}, \ldots, \{W_0, W_1, W_2, W_3, W_4\}$. (Reproduced from Journal of Mathematical Imaging and Vision, Vol.3, 2002, pages 199–222, "Multiresolution design of aperture operators," Roberto Hirata Junior, Marcel Brun, Junior Barrera, and Edward R. Dougherty, Figure 22 © 2002 with kind permission of Springer Science and Business Media.)

Figures 2.25 to 2.30 show a small region of a test image, its blurring, and the result of the best filter for each class (linear, multiresolution $W$-operator, nonmultiresolution aperture, and multiresolution aperture). Figures 2.25 and 2.26 show a region of 500 points of the original image and the blurred image, respectively. The latter figure shows 161 points (marked by black edges) with different values. Most of them differ by 1 (129 points) or 2 (30 points). The estimated MSE for the region is 0.534. Figure 2.27 shows the result of the $9 \times 9$ linear filter. The MSE drops to 0.434, but the number of erroneous points does not decrease. The error decrease is due to the decrease of points that differ more than 1 (with a corresponding increase of the number of points that differ by 1). Figure 2.28 shows the result of the best multiresolution $W$-operator filter. In this case, the MSE rises to 0.552, but the number of erroneous points drop to 130. The number of points with difference 1 decreases to 102, while the number of points with difference greater than 1 increases. Figure 2.29 shows the results of the best nonmultiresolution aperture filter (17 points window). The MSE and number of erroneous points drop to 0.342 and 89, respectively. Especially visible is the improved restoration at grain edges. The best result is shown in Figure 2.30. It results from a multiresolution aperture filter starting with a 17-point window. The MSE and number of erroneous points drop to 0.222 and 73, respectively. Edge restoration is further improved over the nonmultiresolution aperture filter.

## 2.6. Summary

We have discussed how it is possible to treat automatic $W$-operator design for gray-level images in the context of finite discrete lattices. This imposes new restrictions to the space of operators and the trade-off is the potential suboptimality of the designed operator. Aperture operators have been characterized, their representation presented, their design analyzed and two representations, via lattice intervals or decision trees, have been given. Unfortunately, both representations are computationally expensive for most useful operators and the more refined is the representation, the more memory is necessary.

Hybrid design has been presented and its usability characterized. Its error due to the restrictions has been analyzed and it has been shown that a considerable improvement can be achieved if good human constraints are designed. Multiresolution design has been presented and analyzed. The representation of this class of

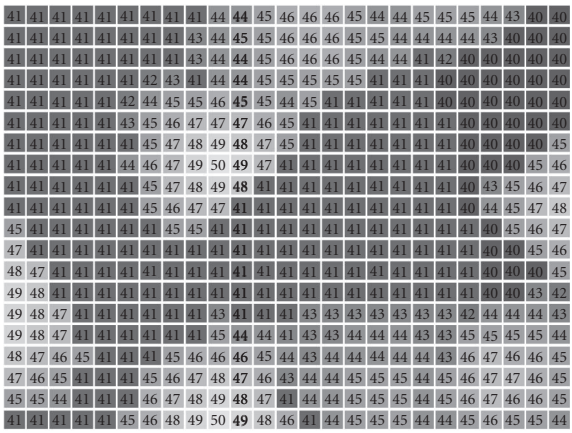


FIGURE 2.25. Section of the original image. (Reproduced from Journal of Mathematical Imaging and Vision, Vol.3, 2002, pages 199–222, "Multiresolution design of aperture operators," Roberto Hirata Junior, Marcel Brun, Junior Barrera, and Edward R. Dougherty, Figure 23 © 2002 with kind permission of Springer Science and Business Media.)

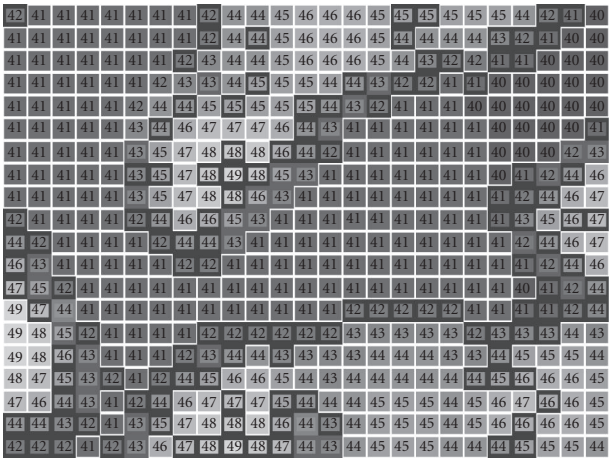


FIGURE 2.26. Blurred image. (Reproduced from Journal of Mathematical Imaging and Vision, Vol.3, 2002, pages 199–222, "Multiresolution design of aperture operators," Roberto Hirata Junior, Marcel Brun, Junior Barrera, and Edward R. Dougherty, Figure 24 © 2002 with kind permission of Springer Science and Business Media.)

operators has been characterized and generalization is inherent in the representation.

Hundreds of experiments have been done (some shown here or in other publications) and they show the superiority of aperture operators over optimal restricted window linear operators for signals and images, using both the MAE and MSE criteria. For some problems, the number of training images has not been enough to make the aperture operator better than the linear filter for MSE, only

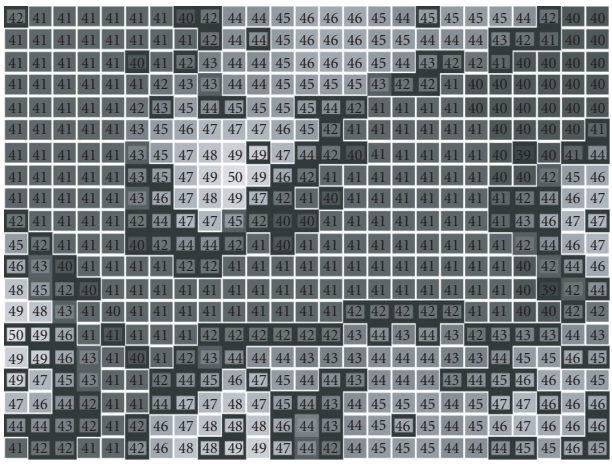


FIGURE 2.27. Result from best linear operator. (Reproduced from Journal of Mathematical Imaging and Vision, Vol.3, 2002, pages 199–222, "Multiresolution design of aperture operators," Roberto Hirata Junior, Marcel Brun, Junior Barrera, and Edward R. Dougherty, Figure 25 © 2002 with kind permission of Springer Science and Business Media.)

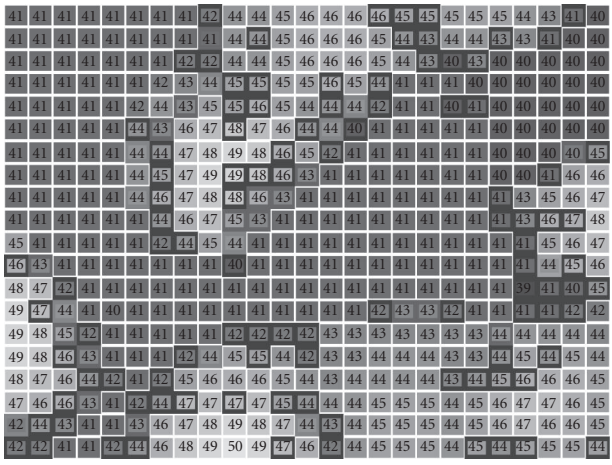


FIGURE 2.28. Result from best multiresolution $W$-operator. (Reproduced from Journal of Mathematical Imaging and Vision, Vol.3, 2002, pages 199–222, "Multiresolution design of aperture operators," Roberto Hirata Junior, Marcel Brun, Junior Barrera, and Edward R. Dougherty, Figure 26 © 2002 with kind permission of Springer Science and Business Media.)

for MAE. Nevertheless, the visual result is better for aperture operators because these are less error prone on image edges. Moreover, the graphics show that the aperture operators would eventually be better than the linear operators for MSE if more examples were given for training the aperture.

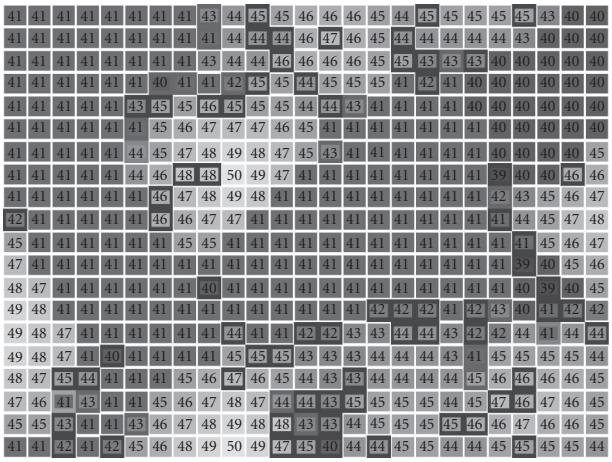


FIGURE 2.29. Result from best nonmultiresolution aperture. (Reproduced from Journal of Mathematical Imaging and Vision, Vol.3, 2002, pages 199–222, "Multiresolution design of aperture operators," Roberto Hirata Junior, Marcel Brun, Junior Barrera, and Edward R. Dougherty, Figure 27 © 2002 with kind permission of Springer Science and Business Media.)

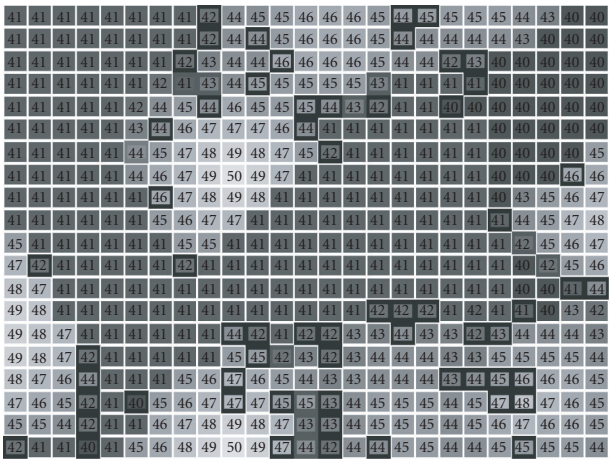


FIGURE 2.30. Result from best multiresolution aperture. (Reproduced from Journal of Mathematical Imaging and Vision, Vol.3, 2002, pages 199–222, "Multiresolution design of aperture operators," Roberto Hirata Junior, Marcel Brun, Junior Barrera, and Edward R. Dougherty, Figure 28 © 2002 with kind permission of Springer Science and Business Media.)

Decision trees have been initially chosen because they are fast to implement and they can represent the full class of aperture operators. However, there are two main disadvantages: (1) since the representation of the operator is done by a partition of the space of possible configurations and the morphological representation is done by maximal intervals from the operator's kernel, it is difficult to integrate the decision-tree representation with the software; (2) the representation by

decision trees is not adequate to impose algebraic restrictions to the design of the operator.

The superiority of nonlinear operators in relation to linear ones is not difficult to understand when one thinks in terms of lattices. Besides the inherent suboptimality of linearity relative to the most general class of operators, when a linear operator is applied to an image, the result usually cannot be represented by a point in the output lattice. What is usually done is that one makes the result discrete in order to have a representation in the lattice. Therefore, the discretization in practice transforms linear operators into nonlinear ones. This is true from electronic acquisition devices to the theory of optimal linear filters in discrete spaces. In the latter case, the discretization makes the principal hypothesis of the optimal linear theory (that one is finding the inverse transform of a linear operator) void.

There are several problems still to be addressed in the area of automatic design of operators: strategies to choose the aperture (i.e., $W$, $K$, and $M$), strategies/algorithms to speed up the search of the operator, and compact ways to represent the operator, to name a few.

## Acknowledgments

The authors want to thank Elsevier and Springer Science and Business for giving us permission to reproduce parts of some material already published. Figures 2.4(a) and 2.4(c), 2.5(a) and 2.5(c), 2.6(a) and 2.6(b), 2.7(a) and 2.7(b), 2.8, 2.9, 2.10, 2.11, 2.12, and 2.13 and some text are reproduced here from [55] with permission from Elsevier. Figures 2.14, 2.15, 2.16, 2.17, 2.18(a) and 2.18(b), 2.20, 2.21, 2.22, 2.23, 2.24, 2.25, 2.26, 2.27, 2.28, 2.29, and 2.30 and some text are reproduced here from [56] and [57] with kind permission of Springer Science and Business. Roberto Hirata Jr. is partially supported by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq). Marcel Brun has been supported by Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) during the development of this work. Junior Barrera is partially supported by CNPq, FAPESP, and the National Institutes of Health (NIH), Grant 1 D43 TW07015-01. Edward R. Dougherty has, in the recent years, been supported by the National Human Genome Research Institute, the National Science Foundation, the National Cancer Institute, the Translational Genomics Research Institute, and the University of Texas M. D. Anderson Cancer Center.

## Bibliography

[1] G. J. F. Banon and J. Barrera, "Minimal representations for translation-invariant set mappings by mathematical morphology," *SIAM Journal on Applied Mathematics*, vol. 51, no. 6, pp. 1782–1798, 1991.

[2] G. J. F. Banon and J. Barrera, "Decomposition of mappings between complete lattices by mathematical morphology, part I: general lattices," *Signal Processing*, vol. 30, no. 3, pp. 299–327, 1993.

[3] J. Barrera, G. J. F. Banon, R. A. Lotufo, and R. Hirata Jr., "MMach: a mathematical morphology toolbox for the Khoros system," *Journal of Electronic Imaging*, vol. 7, no. 1, pp. 174–210, 1998.

[4] J. Barrera and E. R. Dougherty, "Representation of gray-scale windowed operators," in *Mathematical Morphology and Its Applications to Image and Signal Processing*, H. J. A. M. Heijmans and J. B.

T. M. Roerdink, Eds., vol. 12 of *Computational Imaging and Vision*, pp. 19–26, Kluwer Academic, Dordrecht, The Netherlands, 1998.

[5] J. Barrera, E. R. Dougherty, and M. Brun, "Hybrid human-machine binary morphological operator design. An independent constraint approach," *Signal Processing*, vol. 80, no. 8, pp. 1469–1487, 2000.

[6] J. Barrera, E. R. Dougherty, and N. S. T. Hirata, "Design of optimal morphological operators from prior filters," *Acta Stereologica*, vol. 16, no. 3, pp. 193–200, 1997.

[7] J. Barrera, E. R. Dougherty, and N. S. Tomita, "Automatic programming of binary morphological machines by design of statistically optimal operators in the context of computational learning theory," *Journal of Electronic Imaging*, vol. 6, no. 1, pp. 54–67, 1997.

[8] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*, The Wadsworth Statistics/Probability Series, Wadsworth, Belmont, Calif, USA, 1984.

[9] M. Brun, R. Hirata Jr., J. Barrera, and E. R. Dougherty, "Nonlinear filter design using envelopes," *Journal of Mathematical Imaging and Vision*, vol. 21, no. 1-2, pp. 81–97, 2004.

[10] J. Barrera and G. P. Salas, "Set operations on closed intervals and their applications to the automatic programming of MMach's," Tech. Rep. RT-MAC-9604, Instituto de Matemática e Estatística, USP, São Paulo, Brasil, April 1996.

[11] J. Barrera and G. P. Salas, "Set operations on closed intervals and their applications to the automatic programming of morphological machines," *Journal of Electronic Imaging*, vol. 5, no. 3, pp. 335–352, 1996.

[12] J. Barrera, R. Terada, R. Hirata Jr., and N. S. T. Hirata, "Automatic programming of morphological machines by PAC learning," *Fundamenta Informaticae*, vol. 41, no. 1-2, pp. 229–258, 2000.

[13] A. Cypher, D. C. Halbert, D. Kurlander, et al., Eds., *Watch What I Do: Programming by Demonstration*, MIT Press, Cambridge, Mass, USA, 1993.

[14] E. J. Coyle and J.-H. Lin, "Stack filters and the mean absolute error criterion," *IEEE Transactions Acoustics, Speech, Signal Processing*, vol. 36, no. 8, pp. 1244–1254, 1988.

[15] C. Cuciurean-Zapan, E. R. Dougherty, and Y. Chen, "Behavior of adaptive digital erosions," in *Statistical and Stochastic Methods for Image Processing*, vol. 2823 of *Proceedings of SPIE*, pp. 100–108, Denver, Colo, USA, August 1996.

[16] E. R. Dougherty and J. Barrera, "Bayesian design of optimal morphological operators based on prior distributions for conditional probabilities," *Acta Stereologica*, vol. 16, no. 3, pp. 167–174, 1997.

[17] E. R. Dougherty and J. Barrera, "Pattern recognition theory in nonlinear signal processing," *Journal of Mathematical Imaging and Vision*, vol. 16, no. 3, pp. 181–197, 2002.

[18] E. R. Dougherty, J. Barrera, G. Mozelle, S. Kim, and M. Brun, "Multiresolution analysis for optimal binary filters," *Journal of Mathematical Imaging and Vision*, vol. 14, no. 1, pp. 53–72, 2001.

[19] E. R. Dougherty and R. M. Haralick, "Unification of nonlinear filtering in the context of binary logical calculus, part I: binary filters," *Journal of Mathematical Imaging and Vision*, vol. 2, no. 2-3, pp. 173–183, 1992.

[20] E. R. Dougherty, S. Kim, and Y. Chen, "Coefficient of determination in nonlinear signal processing," *Signal Processing*, vol. 80, no. 10, pp. 2219–2235, 2000.

[21] E. R. Dougherty and R. P. Loce, "Optimal mean-absolute-error hit-or-miss filters: morphological representation and estimation of the binary conditional expectation," *Optical Engineering*, vol. 32, no. 4, pp. 815–827, 1993.

[22] E. R. Dougherty and R. P. Loce, "Precision of morphological-representation estimators for translation-invariant binary filters: increasing and nonincreasing," *Signal Processing*, vol. 40, no. 2-3, pp. 129–154, 1994.

[23] E. R. Dougherty and R. P. Loce, "Optimal binary differencing filters: design, logic complexity, precision analysis, and application to digital document processing," *Journal of Electronic Imaging*, vol. 5, no. 1, pp. 66–86, 1996.

[24] E. R. Dougherty and R. A. Lotufo, *Hands-on Morphological Image Processing*, SPIE Press, Bellingham, Wash, USA, 2003.

[25] E. R. Dougherty, "Optimal mean-square N-observation digital morphological filters I. Optimal binary filters," *CVGIP: Image Understanding*, vol. 55, no. 1, pp. 36–54, 1992.

[26] E. R. Dougherty, "Optimal mean-square N-observation digital morphological filters II. Optimal gray-scale filters," *CVGIP: Image Understanding*, vol. 55, no. 1, pp. 55–72, 1992.
[27] E. R. Dougherty, *Random Processes for Image and Signal Processing*, SPIE/IEEE Press, Bellingham, Wash, USA, 1999.
[28] E. R. Dougherty and D. Sinha, "Computational mathematical morphology," *Signal Processing*, vol. 38, no. 1, pp. 21–29, 1994.
[29] R. Hirata Jr., M. Brun, J. Barrera, and E. R. Dougherty, "Image restoration by multiresolution nonlinear filters," in *Nonlinear Image Processing and Pattern Analysis XII*, vol. 4304 of *Proceedings of SPIE*, pp. 256–264, San Jose, Calif, USA, January 2001.
[30] N. S. T. Hirata, E. R. Dougherty, and J. Barrera, "A switching algorithm for design of optimal increasing binary filters over large windows," *Pattern Recognition*, vol. 33, no. 6, pp. 1059–1081, 2000.
[31] H. J. A. M. Heijmans, *Morphological Image Operators*, Academic Press, Boston, Mass, USA, 1994.
[32] T. Hastie, R. Tibshirani, and J. H. Friedman, *The Elements of Statistical Learning*, Springer, New York, NY, USA, 2001.
[33] P. Kuosmanen and J. T. Astola, "Optimal stack filters under rank selection and structural constraints," *Signal Processing*, vol. 41, no. 3, pp. 309–338, 1995.
[34] J.-H. Lin and E. J. Coyle, "Minimum mean absolute error estimation over the class of generalized stack filters," *IEEE Transactions Acoustics, Speech, Signal Processing*, vol. 38, no. 4, pp. 663–678, 1990.
[35] R. P. Loce and E. R. Dougherty, "Mean-absolute-error representation and optimization of computational-morphological filters," *Graphical Models and Image Processing*, vol. 57, no. 1, pp. 27–37, 1995.
[36] R. P. Loce and E. R. Dougherty, *Enhancement and Restoration of Digital Documents: Statistical Design of Nonlinear Algorithms*, SPIE Optical Engineering Press, Bellingham, Wash, USA, 1997.
[37] P. Maragos, "A representation theory for morphological image and signal processing," *IEEE Transactions on Pattern Analysis And Machine Intelligence*, vol. 11, no. 6, pp. 586–599, 1989.
[38] G. Matheron, *Random Sets and Integral Geometry*, John Wiley & Sons, New York, NY, USA, 1975.
[39] T. M. Mitchell, *Machine Learning*, McGraw-Hill Series in Computer Science, McGraw-Hill, New York, NY, USA, 1997.
[40] S. K. Murthy, S. Kasif, and S. Salzberg, "A system for induction of oblique decision trees," *Journal of Artificial Intelligence Research*, vol. 2, pp. 1–32, 1994.
[41] V. S. Pugachev, *Theory of Random Functions and Its Application to Control Problems*, Pergamon Press, Oxford, UK, 1965, Revised translation by O. M. Blunn.
[42] J. R. Quinlan, "Simplifying decision trees," *International Journal of Man-Machine Studies*, vol. 27, no. 3, pp. 221–234, 1987.
[43] P. Salembier, "Structuring element adaptation for morphological filters," *Journal of Visual Communication and Image Representation*, vol. 3, no. 2, pp. 115–136, 1992.
[44] O. V. Sarca, E. R. Dougherty, and J. T. Astola, "Secondarily constrained Boolean filters," *Signal Processing*, vol. 71, no. 3, pp. 247–263, 1998.
[45] O. V. Sarca, E. R. Dougherty, and J. T. Astola, "Two-stage binary filters," *Journal of Electronic Imaging*, vol. 8, no. 3, pp. 219–232, 1999.
[46] J. Serra, *Image Analysis and Mathematical Morphology*, Academic Press, London, UK, 1982.
[47] P. Soille, *Morphological Image Analysis*, Springer, Berlin, Germany, 1999.
[48] F. Preteux and M. Schmitt, "Boolean texture analysis and synthesis," in *Image Analysis and Mathematical Morphology*, J. Serra, Ed., vol. 2, pp. 377–400, Academic Press, London, UK, 1988.
[49] I. Tăbuş, D. Petrescu, and M. Gabbouj, "A training framework for stack and Boolean filtering-fast optimal design procedures and robustness case study," *IEEE Transactions Image Processing*, vol. 5, no. 6, pp. 809–826, 1996.
[50] N. Wiener, *Extrapolation, Interpolation, and Smoothing of Stationary Time Series. With Engineering Applications*, John Wiley & Sons, New York, NY, USA, 1949.
[51] N. Wiener, *Nonlinear Problems in Random Theory*, Technology Press Research Monographs, MIT Press, Cambridge, Mass, USA, 1963.

[52] L. Yin, J. T. Astola, and Y. A. Neuvo, "Optimal weighted order statistic filters under the mean absolute error criterion," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '91)*, vol. 4, pp. 2529–2532, Toronto, Ontario, Canada, April 1991.
[53] L. Yin, J. T. Astola, and Y. A. Neuvo, "Adaptive stack filtering with applications to image processing," *IEEE Transactions Signal Processing*, vol. 41, no. 1, pp. 162–184, 1993.
[55] R. Hirata Jr., E. R. Dougherty, and J. Barrera, "Aperture filters," *Signal Processing*, vol. 80, no. 4, pp. 697–721, 2000.
[55] R. Hirata Jr., E. R. Dougherty, and J. Barrera, "Aperture filters," *Signal Processing*, vol. 80, no. 4, pp. 697–721, 2000.
[56] M. Brun, E. R. Hirata Jr., J. Barrera, and E. R. Dougherty, "Nonlinear Filter design using envelopes," *Journal of Mathematical Imaging and Vision*, vol. 21, no. 1, pp. 81–97, 2004.
[57] R. Hirata Jr., M. Brun, J. Barrera, and E. R. Dougherty, "Multiresolution design of aperture operators," *Journal of Mathematical Imaging and Vision*, vol. 16, no. 3, pp. 199–222, 2002.

Roberto Hirata Jr.: Instituto de Matemática e Estatística, Universidade de São Paulo, Rua do Matão 1010, CEP 05508-090, São Paulo, SP, Brazil

*Email*: hirata@ime.usp.br

Marcel Brun: Translational Genomics Research Institute, 445 North Fifth Street, Phoenix, AZ 85004, USA

*Email*: mbrun@tgen.org

Junior Barrera: Instituto de Matemática e Estatística, Universidade de São Paulo, Rua do Matão 1010, CEP 05508-090, São Paulo, SP, Brazil

*Email*: jb@ime.usp.br

Edward R. Dougherty: Department of Electrical Engineering, Texas A&M University, Room 9, Zachry Engineering Center, College Station, TX 77843-3128, USA

*Email*: e-dougherty@tamu.edu

# 3 Finite-set signal processing

Ronald K. Pearson and Moncef Gabbouj

Because finite sets are not closed under the ordinary arithmetic operations of addition and multiplication, filters that map one sequence taking values in a finite set into another are necessarily nonlinear. This chapter considers the general problems of designing and characterizing these nonlinear filters and developing useful characterizations like power spectra for sequences taking values in finite sets, including both the general case where there is no additional algebraic structure available to exploit and specialized cases where some useful structure is present (e.g., partial or total orders). Applications to DNA sequence analysis and database cleaning are considered.

## 3.1. Introduction

This chapter is concerned with the unique issues that arise when processing discrete-time signal sequence $\{x_k\}$ that takes values in some finite set $\Sigma$. In statistics [3] and cluster analysis [22, 30], variables defined on a finite set with no additional mathematical structure are called *nominal* or *categorical* variables. Two very large and rapidly growing application areas where these finite-set issues arise are the development of data mining methods for the *streaming data model* [7] and the analysis of biological sequence data (e.g., DNA or protein sequences). The streaming data model is used to describe either datasets that are too large to fit into main memory or data sequences that arrive continuously (e.g., telecommunications, financial services, e-commerce, or sensor network data) and it leads naturally to techniques based on the moving data windows common in signal processing applications [7, 8, 27, 33, 51]. Similarly, the characterization of regular substructures in DNA sequences is important in a number of biological problems, including promoter analysis, the detection of recurring anomalies in tumor cells, and the study of certain genetic diseases like fragile-X mental retardation [12, 13, 23, 29]. A specific example discussed further in Section 3.3.2 is shown in Figure 3.1: the sequence index corresponds to the position of each nucleotide base along the DNA sequence, and the values indicate which of the four possible bases (A = adenine, C = cytosine, G = guanine, and T = thymine) appears at each position. The two

key points of this chapter are first that it is important to explicitly consider the lack of mathematical structure that is inherent in the nominal data model, and second that in favorable cases it is a simple matter to extend real-valued signal processing techniques (e.g., the classical Blackman-Tukey spectrum estimator) to the case of nominal variables.

The analysis of real-valued signal sequences dates back at least to Schuster's introduction of the periodogram at the end of the nineteenth century for characterizing sunspots [50]. Important developments since then include correlation and spectral analysis methods, signal and anomaly (e.g., spike) detection procedures, and digital smoothing and sharpening filters. Linearity has played a key role in these developments, with linear filters figuring prominently (e.g., linear lowpass and bandpass filters for smoothing and sharpening [24], linear matched filters for signal detection [48, Sections 4.9–4.12], and linear prewhitening filters for spectral analysis [46]). Conversely, despite their practical utility in many settings and their considerable analytical advantages, linear signal processing techniques are neither universally effective nor universally applicable. Recognition of this fact has led to the development of nonlinear extensions of many important linear characterization methods, including higher-order extensions of spectral and correlation analysis methods for non-Gaussian signals [37], lower-order extensions of these methods to deal with infinite-variance $\alpha$-stable distributions [38], nonlinear time-frequency characterization methods to deal with nonstationary signals whose statistical character changes over time [15], and nonlinear filters with improved impulsive noise rejection, edge preservation, and other desirable characteristics relative to linear filters [6].

For the finite-set signal processing problem of interest here, linearity is not applicable because the usual operations of addition and multiplication are not applicable to sequences taking values in a finite set $\Sigma$. That is, even if we encode these data values as real numbers, only the finite set containing the single element $\{0\}$ is closed under addition, and only the set $\{0, 1\}$ is closed under multiplication. In addition, note that coding categorical variables with numerical representations introduces an ordering, converting them to *ordinal variables* [3, 22, 30] for which it is possible to say that any element is "smaller" or "larger" than any other. For example, the DNA sequence representation shown in Figure 3.1 orders the bases alphabetically, a useful device for plotting, but one without biological relevance. The introduction of such an artificial ordering on a nominal variable frequently makes the results obtained sensitive to the details of the encoding that induces this ordering. It is known, for example, that spectrum estimation results obtained for numerically encoded DNA sequences exhibit an undesirable dependence on the details of this encoding [12, 13, 29]. This problem is discussed further in Section 3.3.3 in connection with the characterization of nominal variable sequences more generally.

In cases where the ordinal data model is reasonable, it is possible to exploit it directly and much has been done in this direction. Important examples include rank-based filters [6], rank-based spectrum estimation [4], and permutation entropy characterizations [9]. While it lacks the overwhelming power of linearity,

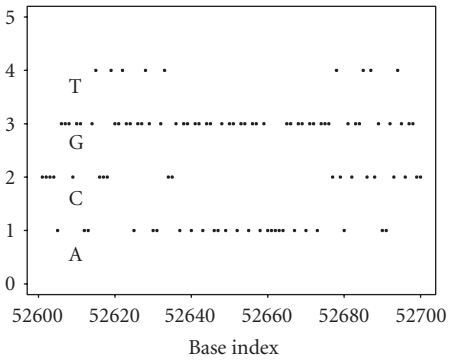



FIGURE 3.1. A very short, highly structured DNA base subsequence from human chromosome 22.

the complete order characterization on which these methods are based does provide an extremely useful framework for the analysis and development of new signal processing methods. In the problems of interest here, this order characterization is not available, making these developments and analyses somewhat more challenging. Instead, we have two basic characterization options: counting the number of times a specific value occurs in a sequence or subsequence, and comparing elements or subsequences. Conversely, it is sometimes possible to introduce relevant *partial orders*, allowing characterizations that are in some sense intermediate between the complete lack of mathematical structure inherent in the nominal data model and the total order that characterizes the ordinal data model. These ideas are introduced in Section 3.2.

More generally, this chapter is organized as follows. First, Section 3.2 presents a brief summary of some of the fundamental mathematical ideas that arise in dealing with finite sets, with particular emphasis on the exploitation of any limited structure that we may be able to impose on the problem of interest (e.g., partial orders). Next, Section 3.3 considers the problem of characterizing sequences defined on finite sets, including both spectrum estimation, applicable to DNA sequence characterization, and moving-window *interestingness measures*, potentially useful in streaming data analysis applications. Section 3.4 then considers the problem of designing (necessarily nonlinear) digital filters for sequences taking values on finite sets. Finally, Section 3.5 briefly considers some of the key issues that arise when extending the basic ideas presented here from sequences taking values in simple finite sets with little or no structure to more complex settings like sequences of XML documents, and Section 3.6 concludes the chapter with a brief summary.

## 3.2. Fundamental notions

The following paragraphs briefly consider the general question of how we can characterize a sequence $\{x_k\}$ of $N$ elements taking values in a given set $\Sigma$ containing

a finite number $M$ of elements. It is important to consider this question because its answers determine our options in developing the signal processing methods discussed in subsequent sections of this chapter.

As noted in the preceeding discussion, in the analysis of nominal data sequences, we have two basic characterization options: counting and comparison. To describe these options more fully, let $\Lambda$ be an arbitrary subset of $\Sigma$ (including the improper inclusion $\Lambda = \Sigma$), and let $\{s_k\}$ be an arbitrary subsequence of $\{x_k\}$, which may contain a single element of $\{x_k\}$, all elements of $\{x_k\}$, or anything in between. The two basic characterization operations for $\{x_k\}$ are the following.

(1) Counting: we can count the number of times each value in the subset $\Lambda$ occurs in the subsequence $\{s_k\}$.
(2) Comparison: we can determine whether all elements of the subsequence $\{s_k\}$ are contained in the subset $\Lambda$ or not.

As subsequent examples will demonstrate, these options are in fact more versatile than they may appear at first glance. In particular, the first of these options forms the basis for the interestingness measures discussed in Section 3.3.3 and the Dedekind majority filter discussed in Section 3.4, while the second of these options forms the basis for the finite-set spectrum estimator discussed in Sections 3.3.1 and 3.3.2. The key to the utility of these basic characterizations lies in our choices of subsequence $\{s_k\}$ and subset $\Lambda$.

Our characterization options increase greatly if the finite set $\Sigma$ possesses additional mathematical structure. For example, a set $\Sigma$ equipped with a binary operation $\circ$ defines a *group* if it satisfies the following four axioms [35, page 43]:

(1) *closure*: $x \circ y$ is a well-defined element of $\Sigma$ for all $x, y \in \Sigma$;
(2) *associativity*: $x \circ (y \circ z) = (x \circ y) \circ z$ for all $x, y, z \in \Sigma$;
(3) *unit element*: there exists a necessarily unique element $e \in \Sigma$ such that $e \circ x = x \circ e = x$ for all $x \in \Sigma$;
(4) *invertibility*: for every $x \in \Sigma$, there exists a unique $y \in \Sigma$ such that $x \circ y = y \circ x = e$.

Less structured alternatives result if $(\Sigma, \circ)$ satisfies only a subset of these axioms. For example, axioms 1, 2, and 3 together define a *monoid*, while axioms 1 and 2 together define a *semigroup*. Finally, if $(\Sigma, \circ)$ satisfies only axiom 1, the result is a *groupoid*, one of the least restrictive mathematical structures possible and the main one that will be considered here.

To see both the generality and potential utility of this last notion, consider the following simple example. If the set $\Sigma$ contains two elements, say 0 and 1, the range of possible binary operations $\circ$ defining a groupoid is described by the truth table given in Table 3.1. That is, for any choice of elements $x$ and $y$ from the set $\Sigma = \{0, 1\}$, the groupoid operation $\circ$ is defined by $x \circ y = T_{xy}$. To satisfy closure, it is only necessary that $T_{xy} \in \Sigma$ for all $x, y \in \Sigma$, which means in this example that for each of the four possible values for the pair $(x, y)$, the result $T_{xy}$ can assume either the value 0 or the value 1. Consequently, there are $2^4 = 16$ possible two-element groupoid operations $\circ$, all of which are listed in Table 3.2. Readers familiar with digital logic will recognize these as the Boolean operations on which sequential logic circuits are based.

Table 3.1. Generic truth table for an arbitrary two-element groupoid.

| $x \backslash y$ | 0 | 1 |
|---|---|---|
| 0 | $T_{00}$ | $T_{01}$ |
| 1 | $T_{10}$ | $T_{11}$ |

Table 3.2. The 16 possible binary groupoid operations.

| No. | Name | $T_{00}$ | $T_{01}$ | $T_{10}$ | $T_{11}$ | Boolean | Eff. arity |
|---|---|---|---|---|---|---|---|
| 1 | ZERO | 0 | 0 | 0 | 0 | $x \circ y = 0$ | 0 |
| 2 | AND | 0 | 0 | 0 | 1 | $x \circ y = x \cdot y$ | 2 |
| 3 | — | 0 | 0 | 1 | 0 | $x \circ y = x \cdot \bar{y}$ | 2 |
| 4 | $x$ | 0 | 0 | 1 | 1 | $x \circ y = x$ | 1 |
| 5 | — | 0 | 1 | 0 | 0 | $x \circ y = \bar{x} \cdot y$ | 2 |
| 6 | $y$ | 0 | 1 | 0 | 1 | $x \circ y = y$ | 1 |
| 7 | EXOR | 0 | 1 | 1 | 0 | $x \circ y = x \cdot \bar{y} + \bar{x} \cdot y$ | 2 |
| 8 | OR | 0 | 1 | 1 | 1 | $x \circ y = x + y$ | 2 |
| 9 | NOR | 1 | 0 | 0 | 0 | $x \circ y = \overline{x + y}$ | 2 |
| 10 | EXNOR | 1 | 0 | 0 | 1 | $x \circ y = \overline{x \cdot \bar{y} + \bar{x} \cdot y}$ | 2 |
| 11 | NOT $y$ | 1 | 0 | 1 | 0 | $x \circ y = \bar{y}$ | 1 |
| 12 | — | 1 | 0 | 1 | 1 | $x \circ y = x + \bar{y}$ | 2 |
| 13 | NOT $x$ | 1 | 1 | 0 | 0 | $x \circ y = \bar{x}$ | 1 |
| 14 | — | 1 | 1 | 0 | 1 | $x \circ y = \bar{x} + y$ | 2 |
| 15 | NAND | 1 | 1 | 1 | 0 | $x \circ y = \overline{x \cdot y}$ | 2 |
| 16 | ONE | 1 | 1 | 1 | 1 | $x \circ y = 1$ | 0 |

This example illustrates two important points. First, for any finite set $\Sigma$, many different groupoids can be defined by constructing the appropriate truth table, analogous to Table 3.2 for the binary case just considered. In the general case, this truth table has $M^2$ positions, where $M$ is the number of elements in $\Sigma$. To satisfy closure, the result $x \circ y$ defined by each position of this table must belong to $\Sigma$. Since this restriction can be met in $M$ ways independently for each of the $M^2$ positions, it follows that there are $M^{M^2}$ possible truth tables, implying the existence of this number of distinct groupoids. It is important to note that this number grows extremely rapidly: for $M = 3$, there are 19, 683 possible groupoids, approximately $4.3 \times 10^9$ for $M = 4$, and approximately $3 \times 10^{17}$ for $M = 5$.

The second key point of this example is that the potential utility of these groupoids for a given set $\Sigma$ varies strongly with our choice of the defining operation. One measure of utility is the *effective arity* of these groupoids, defined as the number of the variables, $x$ and/or $y$, that the result $x \circ y$ actually depends on. For example, in the 16 binary groupoids listed in Table 3.2, the operations ZERO

($x \circ y \equiv 0$) and ONE ($x \circ y \equiv 1$) have effective arity zero since they do not depend on either $x$ or $y$. Similarly, the operations numbered 4, 6, 11, and 13 in Table 3.2 have effective arity one since they depend only on the value of one of the two variables $x$ or $y$. The remaining 12 entries in Table 3.2 are all of effective arity two, since their value depends on the values of both $x$ and $y$ in all cases. In general, we can expect the utility of groupoids to increase with their effective arity, but even among groupoids of maximum effective arity, practical utility can vary greatly. As a specific example, combinatorial logic circuits are based on Boolean algebra, which is defined by the operations AND, OR, and NOT [19]. It is easy to show, however, that all three of these operations can be implemented using only the NAND operation. For example, NOT $x$ is equal to the NAND of $x$ with the constant input 1, for any $x$. Similarly, using this construction following the NAND operation yields the AND operation (i.e., $x$ AND $y$ is NOT ($x$ NAND $y$)), and the OR operation can be constructed from the NAND operation with two NOT operations (i.e., $x$ OR $y$ is (NOT $x$) NAND (NOT $y$)). In contrast, the NOT operation cannot be constructed from either the AND or the OR operation, implying that these groupoids are less inherently flexible than the NAND groupoid. More generally, it is not difficult to show that six of the arity 2 operations defined in Table 3.2 (numbers 3, 5, 9, 12, 14, and 15) are flexible enough to generate all of the other operations listed there but the other six are not.

As noted in the introduction, another simple but extremely useful mathematical structure on finite sets is *order structure*. As a specific and particularly important example, a *partially ordered set* or *poset* is defined as a set $\Sigma$, together with a partial order $\prec$ that satisfies the following three conditions for all $x, y, z \in \Sigma$ [18, page 2]:

(1) *reflexivity:* $x \prec x$,
(2) *antisymmetry:* $x \prec y$ and $y \prec x$ imply that $x = y$,
(3) *transitivity:* $x \prec y$ and $y \prec z$ imply that $x \prec z$.

The most familiar example is the usual arithmetic order $\leq$ on the real numbers, but other examples include alphabetic order on words and the incomplete data partial order on multisets defined below. If two elements of $x, y \in \Sigma$ satisfy either $x \prec y$ or $y \prec x$, they are said to be comparable, otherwise they are said to be noncomparable, denoted as $x \| y$. If all elements of $\Sigma$ are comparable, the poset $(\Sigma, \prec)$ is called a *chain*, while if none of the elements of $\Sigma$ is comparable, $(\Sigma, \prec)$ is called an *antichain*. Note that in the terminology of cluster analysis or categorical data analysis, a chain corresponds to the notion of an *ordinal variable*, while an antichain corresponds to the notion of a *nominal variable*. In the most general case of a poset, some elements may be compared while others cannot, thus representing a variable class intermediate between nominal and ordinal.

Given any poset $(\Sigma, \prec)$, the *Dedekind groupoid* $(\Sigma, \circ)$ is obtained by defining the following binary operation [45]:

$$x \circ y = \begin{cases} x & \text{if } x \prec y, \\ y & \text{if } x \not\prec y. \end{cases} \tag{3.1}$$

Motivation for considering the Dedekind groupoid here is that it forms the basis for the class of Dedekind filters introduced in Section 3.4. The following results of Petre [45] provide useful background for that discussion.

First, note that the binary operation $\circ$ is *commutative* if $x \circ y = y \circ x$ for all $x, y \in \Sigma$. Petre shows that the Dedekind groupoid $(\Sigma, \circ)$ is commutative if and only if the poset $(\Sigma, \prec)$ is a chain [45, Proposition 2]. Conversely, $(\Sigma, \prec)$ is an antichain if and only if $x \circ y = y$ for all $x, y \in \Sigma$ [45, Proposition 3]. Intermediate between these two results is the fact that for any poset, $x \prec y$ is equivalent to $x \circ y = y \circ x = x$ for all $x, y \in \Sigma$.

Petre shows that the Dedekind groupoid is a semigroup if and only if the poset $(\Sigma, \prec)$ does not exhibit any subset of three elements $x$, $y$, and $z$ such that $x$ and $y$ are comparable but $z$ is not comparable with either $x$ or $y$ [45, Proposition 4]. Note that this condition (i.e., the nonexistence of this forbidden configuration) is met by both chains and antichains, but it may not be satisfied by posets of intermediate character. Finally, Petre also shows that the Dedekind groupoid exhibits a unit element $e$ if and only if it is the greatest element of the poset $(\Sigma, \prec)$, that is, if $x \prec e$ for all $x \in \Sigma$ [45, Corollary 2]. As a consequence, it follows that the Dedekind groupoid $(\Sigma, \circ)$ is a monoid if and only if the poset $(\Sigma, \prec)$ has the greatest element and does not have three elements satisfying the forbidden condition described above. These monoid requirements are not satisfied by antichains (i.e., there is no greatest element), but they are satisfied by chains.

The following example provides a specific illustration of these ideas and forms the basis for the Dedekind filter introduced in Section 3.4. First, note that the elements of a set $\Sigma$ are necessarily distinct; a collection $\mathcal{M}$ of objects that can occur one or more times is called a *multiset* or *bag* [5]. The moving-window data filters discussed in Section 3.4 are based on a multiset $\mathcal{M}$ obtained by extracting a subsequence $\{s_k\}$ from a longer sequence $\{x_k\}$ of elements from some finite set $\Sigma$. The class of Dedekind filters introduced in Section 3.4 is based on the *incomplete data partial order* $\prec$ on a data window $\mathcal{W}$. This data window is an indexed multiset whose elements $w_k$ either belong to the finite set $\Sigma$, or are *missing data elements*, denoted as NULL and interpreted to mean that the true value of $w_k$ is unknown and could be any of the elements of $\Sigma$. The use of NULL values for missing data introduces some important practical complications and it has been strongly criticized in the database community [17]. Nevertheless, the problem of treating missing data is an extremely important one in many data analysis applications [44] and the incomplete data partial order defined here provides a useful framework for addressing these problems in the context of filter design. Specifically, define the partial order $\prec$ on $\mathcal{W}$ as follows. First, define $\kappa(x)$ as the number of times $x$ appears in $\mathcal{W}$. Then, define $x \prec y$ as

$$x \prec y \quad \text{if } x = y \qquad \text{or} \qquad \kappa(x) > \kappa(y). \tag{3.2}$$

Note that the partial order $(\mathcal{W}, \prec)$ is not a chain since elements $x \neq y$ which both occur the same number of times in $\mathcal{W}$ are not comparable (i.e., $x \not\prec y$ and $y \not\prec x$). Hence, by Petre [45, Proposition 2], the Dedekind groupoid $(\mathcal{W}, \circ)$ generated

by this partial order is not commutative. The binary operator $\circ$ on which this groupoid is based is given explicitly by

$$x \circ y = \begin{cases} x & \text{if } x = y \text{ or } \kappa(x) > \kappa(y), \\ y & \text{if } x \neq y,\ \kappa(x) \leq \kappa(y). \end{cases} \tag{3.3}$$

Next, suppose that $x$, $y$, and $z$ are distinct elements of $\Sigma$ and suppose that either $x \prec y$ or $y \prec x$, implying that $\kappa(x) \neq \kappa(y)$. Next, assume that $z$ is not comparable with either $x$ or $y$; this implies that $\kappa(z) = \kappa(x) = \kappa(y)$, which is a contradiction. Hence, there can be no triple of distinct elements $x$, $y$, and $z$ for which $x$ and $y$ are comparable but $z$ is not comparable to either $x$ or $y$, so it follows from Petre [45, Proposition 4] that the Dedekind groupoid $(\mathcal{W}, \circ)$ is associative. Finally, note that by definition, the value NULL occurs zero times in $\mathcal{W}$, so $x \prec \text{NULL}$ for any $x \in \Sigma$, from which it follows that NULL represents a unit element in the Dedekind groupoid, further implying that this groupoid is a monoid with greatest element NULL. As noted, these ideas are discussed further in Section 3.4.

## 3.3. Characterization on finite sets

To illustrate the range of issues and possibilities that arise in characterizing sequences defined on finite sets, the following sections consider two specific examples: spectrum estimation and moving-window interestingness measures. Motivation for the first of these topics comes from the extreme practical importance of nonparametric spectrum estimation for the exploratory characterization of real-valued data sequences [31, 46]. The extension of these ideas from real-valued sequences to sequences of categorical variables is described in Sections 3.3.1 and 3.3.2, closely following the development presented at EUSIPCO '04 [43]. Motivation for the second of these topics comes from the fact that moving-window data characterizations are natural for the streaming data model mentioned in the introduction, in which data sequences may be profitably viewed as being of infinite length [7, 27, 33]. Section 3.3.3 considers the development of moving-window *interestingness measures* for sequences of nominal variables, based on some of the ideas discussed by Hilderman and Hamilton [28].

### 3.3.1. Correlations and spectra

In some cases, it is possible to extend a well-developed signal processing approach for real-valued sequences to the case of sequences defined on finite sets, retaining much of the machinery developed for the real-valued case. One specific example is the Blackman-Tukey spectral estimator [31], based on the following notions. For a stationary sequence $\{x_k\}$ of real-valued random variables, the power spectral density $S_{xx}(f)$ is defined as the discrete Fourier transform of the autocorrelation

function:

$$S_{xx}(f) = \sum_{k=-\infty}^{\infty} R_{xx}(k)e^{-i2\pi kfT}. \tag{3.4}$$

In this definition, it is assumed that $\{x_k\}$ is a uniformly sampled time series with intersample spacing $T$, and $R_{xx}(k)$ is the autocorrelation function:

$$\begin{aligned} R_{xx}(k) &= E\{(x_j - E\{x_j\})(x_{j+|k|} - E\{x_j\})\} \\ &= \rho(x_j, x_{j+|k|})\sigma^2, \end{aligned} \tag{3.5}$$

where $\sigma^2$ is the variance of the sequence $\{x_k\}$ and $\rho(x_j, x_{j+k})$ is the correlation coefficient between $x_j$ and $x_{j+k}$. One way of converting this definition into a computational procedure is to consider the Blackman-Tukey estimator [31, page 77]:

$$\widehat{S}_{xx}(f) = \sum_{k=-M}^{M} w_k \widehat{R}_{xx}(k)e^{-i2\pi kfT}, \tag{3.6}$$

where the real numbers $\{w_k\}$ define a *lag window*, included to manage the bias-variance tradeoff inherent in spectrum estimation [31, 46].

To adapt this formulation to $\Sigma$-valued sequences, it is only necessary to specify a useful autocorrelation estimator $\widehat{R}_{xx}(k)$. In cluster analysis, the correlation coefficient $\rho$ between two real-valued data vectors provides the basis for a useful *dissimilarity measure* between vectors [30, page 19]:

$$d_{xy} = \frac{1 - \rho_{xy}}{2}. \tag{3.7}$$

To have a valid dissimilarity measure, it is necessary that $d_{xx} = 0$ for all data vectors $\mathbf{x}$, and that $d_{xy} \geq 0$ and $d_{yx} = d_{xy}$ for all data vectors $\mathbf{x}$ and $\mathbf{y}$. Since, as discussed below, it is a simple matter to define dissimilarity measures for sequences taking values in a finite set $\Sigma$, we reverse the relation defined in (3.7) to obtain the desired autocorrelation measure:

$$\widehat{R}_{xx}(k) = \rho(x_j, x_{j+k}) = 1 - 2d(x_j, x_{j+k}). \tag{3.8}$$

Here, we consider the following dissimilarity measure between subsequences of fixed length $K$:

$$d(x_j, x_{j+k}) = \frac{1}{K}\sum_{i=0}^{K-1} \delta(x_{i+j}, x_{i+j+k}), \tag{3.9}$$

where $\delta(x_i, x_j) = 0$ if $x_i = x_j$ and 1 otherwise. Note that this element-based dissimilarity measure represents an application of the comparison operation defined

in Section 3.2, obtained by taking the subsequence $\{s_k\}$ to be the single element $x_i$ and the subset $\Lambda$ to be the single element $x_j$.

A disadvantage of the autocorrelation estimates $\hat{R}_{xx}(k)$ obtained from (3.8) and (3.9) is $\hat{R}_{xx}(k) \not\to 0$ as $|k| \to \infty$. As a consequence, a large zero-frequency peak and its associated side-lobes appear in the estimated power spectrum. These features obscure the spectral characteristics of interest, so we remove them by replacing $\hat{R}_{xx}(k)$ with $\hat{R}_{xx}(k) - \bar{R}$ in (3.6), where $\bar{R}$ is the average of the computed autocorrelation values over $k$. Note that this difficulty is somewhat analogous to the spectral characterization problem for long-memory processes where the slow decay of autocorrelations to zero introduces a pole in the power spectrum at zero frequency [11, page 6]. Here, however, the problem is simply a computational nuisance rather than a phenomenon of significant inherent interest.

To see the reason for this lack of decay of the estimated autocorrelations at remote lags $k$, note that for a real-valued sequence of independent, identically distributed (i.i.d.) random variables (i.e., a white noise sequence), the autocorrelation function is

$$R_{xx}(k) = \begin{cases} 1 & k = 0, \\ 0 & k \neq 0. \end{cases} \tag{3.10}$$

Now, consider an i.i.d. sequence $\{x_k\}$ that takes each value in the finite set $\Sigma$ with equal probability, where $\Sigma$ contains $M$ elements. It follows from (3.9) that the expected dissimilarity between elements $x_j$ and $x_{j+k}$ in this sequence is

$$\begin{aligned} E\{d(x_j, x_{j+k})\} &= \begin{cases} 0, & k = 0, \\ 1 - \dfrac{1}{M}, & k \neq 0, \end{cases} \\ \Longrightarrow \hat{R}_{xx}(k) &= \begin{cases} 1, & k = 0, \\ -1 + \dfrac{2}{M}, & k \neq 0. \end{cases} \end{aligned} \tag{3.11}$$

### 3.3.2. Example: DNA sequence spectra

Figure 3.2 summarizes two results obtained using the spectrum estimation procedure just described, applied to a perfectly periodic sequence of length $L = 100$, corresponding to 20 repetitions of the subsequence GGCTG. The higher-amplitude (solid) curves in these plots represent the spectrum estimates obtained using two different lag windows, and the lower-amplitude (dotted) curves represent the permutation-based validation results discussed below. Figure 3.2(a) was obtained using the rectangular window $w_k = 1$ for $-25 \leq k \leq 25$ and Figure 3.2(b) shows the results obtained using the triangular Bartlett window defined on the same support set [46, page 439]. As expected, the Bartlett window reduces variability at the expense of increased bias, which appears here in the form of reduced-intensity spectral peaks. In both cases, a clear peak is evident at the fundamental frequency

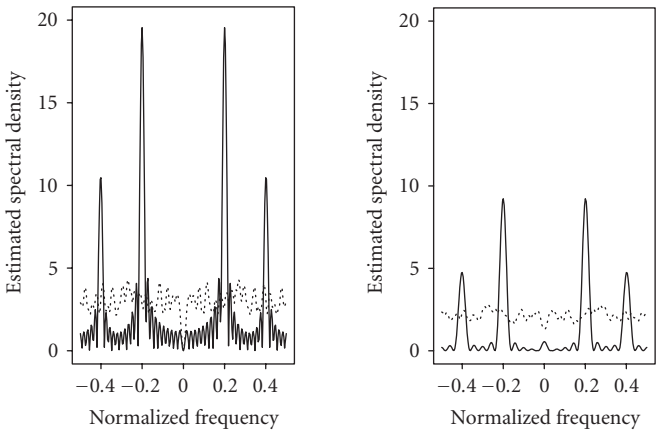



FIGURE 3.2. Estimated spectrum and permutation limits using (a) rectangular plot and (b) Bartlett plot lag windows (triangular window).

$f = 0.2$, corresponding to the period $P = 5$ of the repetitive sequence, and a weaker peak appears at the second harmonic, $f = 0.4$. The key points here are first that much of the machinery of classical spectral analysis (e.g., classical lag windows) can be applied to the finite-set formulation proposed here and second, that this procedure gives the correct results in the case of simple periodic sequences.

The lower-amplitude (dotted) curves in Figure 3.2 were obtained by applying a variant of the computational negative controls (CNC) strategy proposed recently for validating cluster analysis results [42]. Specifically, after the spectrum estimate was obtained from the original data sequence, the same spectrum estimation procedure was applied to each of 50 random permutations of this data sequence. These results provide a useful frame of reference since the permutations destroy any regular sequential structure present in the original data sequence, giving essentially 50 white noise sequences with the same distribution of values as this original sequence. Since these sequences should exhibit constant power spectra, only those features in the original spectrum that significantly exceed these randomized spectra should be regarded as significant. The lower curve in Figure 3.2 represents the maximum value obtained, at each frequency $f$, from these 50 randomized results. In Figure 3.2, both of the plots show clear evidence of the periodic structure present in the sequence since both the fundamental and the second harmonic peaks clearly exceed this lower reference curve.

Figure 3.3 shows the results obtained for two contaminated versions of the periodic sequence considered above, obtained by substituting some of the original sequence values with randomly selected elements of the base set $\Sigma$. In estimating spectra from real-valued data sequences, the presence of *outliers*, nonrepresentative data values that often appear as local "spikes," can introduce significant biases [39]. In particular, it is known that outliers "raise the noise floor" in

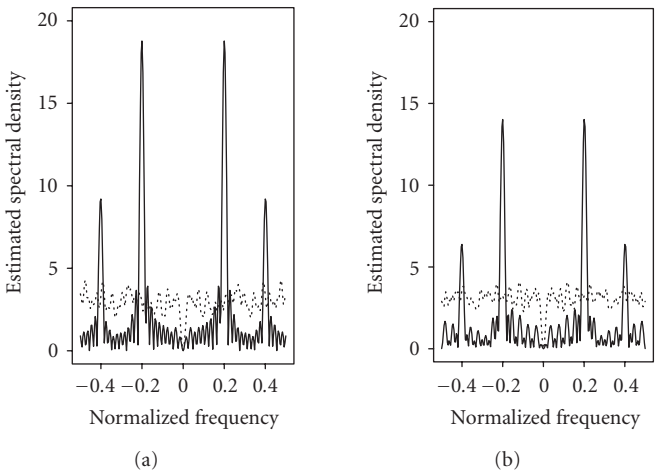



FIGURE 3.3. Effects of (a) 5% and (b) 20% random substitutions on BT spectra.

real-valued spectrum estimation, obscuring high-frequency details [36]. In dealing with sequences taking values from a finite set, the problem is somewhat different, but sequence distortions have generally analogous effects, as may be seen in Figure 3.3. Specifically, Figure 3.3(a) shows the results obtained for this periodic sequence contaminated with 5% random substitutions, while Figure 3.3(b) shows the results obtained with 20% random substitutions. The rectangular lag window was used in both cases since it gives intense spectral peaks that are easily distinguished from the randomized reference results. In contrast with the real-valued outlier problem, the fact that all data values must belong to the small set $\Sigma$ bounds the magnitude of possible contaminants and this restriction appears to substantially reduce their severity. In particular, although comparison of Figures 3.2 and 3.3 shows clearly that increased contamination levels cause degradation of the spectral peaks, both the fundamental and the second harmonic peaks remain significant relative to the CNC baseline even with 20% contamination.

Two other important phenomena in DNA sequence characterization are random insertions and random deletions, collectively known as *indels*. Figure 3.4 shows the results obtained for the periodic sequence considered above, but with 5% and 20% random deletions in Figures 3.4(a) and 3.4(b), respectively. Figure 3.5 shows the corresponding results obtained for the same original sequence, but with 5% (Figure 3.5(a)) or 20% (Figure 3.5(b)) random insertions. Careful comparison of Figures 3.4 and 3.5 suggests that random insertions may be slightly more damaging than random deletions, but both effects appear to be quite similar. In particular, it is clear that indels generally represent a more serious problem for estimated sequence spectra than random substitutions.

Finally, to illustrate the application of this spectrum estimation procedure to a real data example, Figure 3.6 shows the results obtained for the sequence of 100 bases shown in Figure 3.1. This sequence was extracted from human chromosome

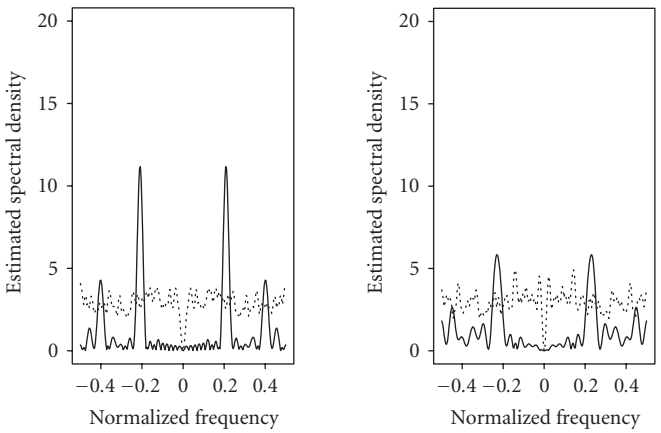



FIGURE 3.4. Effects of (a) 5% and (b) 20% random deletions on BT spectra.

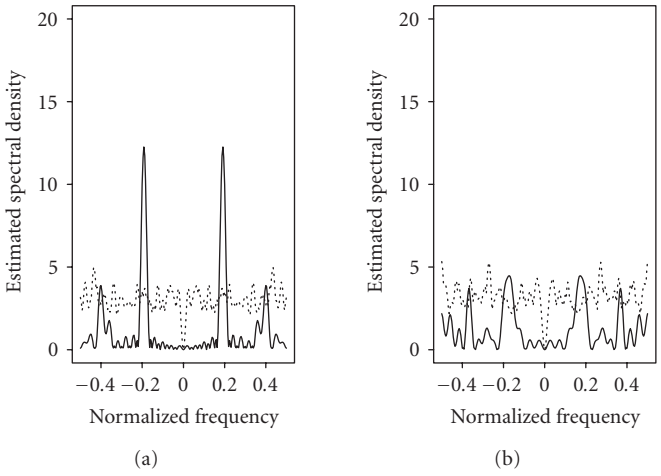



FIGURE 3.5. Effects of (a) 5% and (b) 20% random insertions on BT spectra.

22 (July 2003 Assembly, from http://genome.ucsc.edu). This chromosome is approximately 50,000,000 bases long and the sequence considered here corresponds to bases 40,052,600 through 40,052,699, selected because expert annotation indicates the presence of an approximately periodic repetition of the sequence GGA in the middle of this subsequence. The presence of a peak at $f \simeq 0.33$ that is significant relative to the CNC background spectrum in Figure 3.6 is consistent with this characterization.

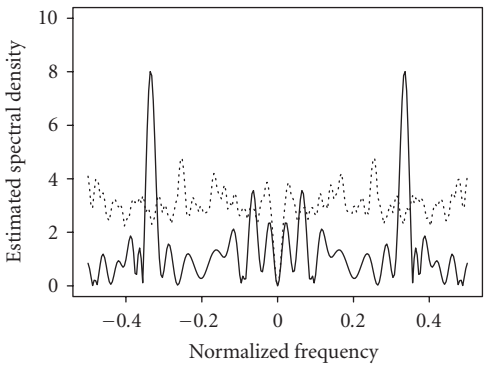



FIGURE 3.6. Spectrum estimate for 100 bases from human chromosome 22.

### 3.3.3. Moving-window interestingness measures

As noted in the introduction, another important emerging application area for digital signal processing techniques is the development of data mining methods for the streaming data model [7]. While data mining has thus far been more closely associated with statistics than with signal processing, the notion of sliding-window filtering has been discussed [33], including the critical question of window width specification. Broadly speaking, data mining is concerned with the detection of "interesting" patterns in a dataset, and this has led to the development of a number of numerical *interestingness measures* [28]. The following paragraphs briefly consider the implementation of some of these interestingness measures in connection with the streaming data model for the detection of *locally interesting patterns* in a data sequence. This idea is essentially an extension of characterizations like the short-time Fourier transform (STFT) [15] or moving-window scale and skewness measures [44, Section 7.5.1] that are useful in characterizing the local behavior of real-valued data sequences to the case of sequences taking values on a finite set. Although the techniques are quite different, the motivation for this type of analysis is analogous to the problem of modeling flat stretches, bursts, and outliers in real-valued time series considered by Le et al. [32].

While it is not a time-series dataset, the key ideas behind moving-window interestingness measures for data sequences taking values on a finite set are nicely illustrated by the *mushroom* dataset from the UCI Machine Learning Archive, available at http://www.ics.uci.edu/~mlearn/MLRepository.html. The archive contains over 100 standard examples that have been widely used in the machine learning community as benchmarks for evaluating classification algorithms. The mushroom dataset contains 22 measured characteristics, all categorical, of 8124 different mushrooms, along with their classification as either edible or poisonous. These characterizations were obtained from *The Audubon Society Field Guide to North American Mushrooms* [34], which notes that there is no simple rule for determining the edibility of mushrooms, providing the original motivation for this classification benchmark [49]. We consider this dataset because it may be viewed

as a collection of 22 moderately long categorical data sequences, nicely illustrating the key ideas of interest here.

For example, one of the categorical variables contained in this dataset is gill color, which can assume any of 12 values, defined and coded as follows:

(1) black (k),
(2) brown (n),
(3) buff (b),
(4) chocolate (h),
(5) gray (g),
(6) green (r),
(7) orange (o),
(8) pink (p),
(9) purple (u),
(10) red (e),
(11) white (w),
(12) yellow (y).

It was noted in Section 3.1 that in the characterization of DNA sequence data, the use of numerical coding introduces an unnatural ordering that can have undesirable consequences. The gill-color sequence provides a nice illustration of this point. Specifically, Figure 3.7 shows four plots of the gill-color sequence for mushrooms 4700 through 6700, encoded as the integers 1 through 12 in each of the following four ways:

(A) alphabetically by color,
(B) alphabetically by coding symbol,
(C) in order of increasing frequency $p_i$,
(D) ordered to give an approximately symmetric, unimodal distribution of $p_i$ values.

In addition, the standard deviation computed from the numerical representation for each coded sequence is given in Figure 3.7. The significant differences between these numerical values, which vary by approximately a factor of two, give one indication of the undesirability of this representation strategy. In particular, note that there are $12! \simeq 4.79 \times 10^8$ possible labellings of these 12 nominal values by the integers 1 through 12, and the differences seen in both the appearance of the plots in Figure 3.7 and the numerical standard deviation values demonstrate that results based on this representation are strongly dependent on the specific coding chosen. Unfortunately, there is generally no "natural" encoding that is to be preferred to any other, making this strong dependence highly undesirable.

This last observation motivates one of the five axiomatic characterizations for a "good" interestingness measure offered by Hilderman and Hamilton [28]: property P4, permutation invariance. More specifically, Hilderman and Hamilton consider 13 numerical interestingness measures, originally proposed in a variety of different fields (e.g., biology, economics, information theory, and linguistics). While many of these measures are quite closely related (indeed, several reduce to exactly the same measure under the normalization considered here), they do represent a variety of different characterizations exhibiting different properties.

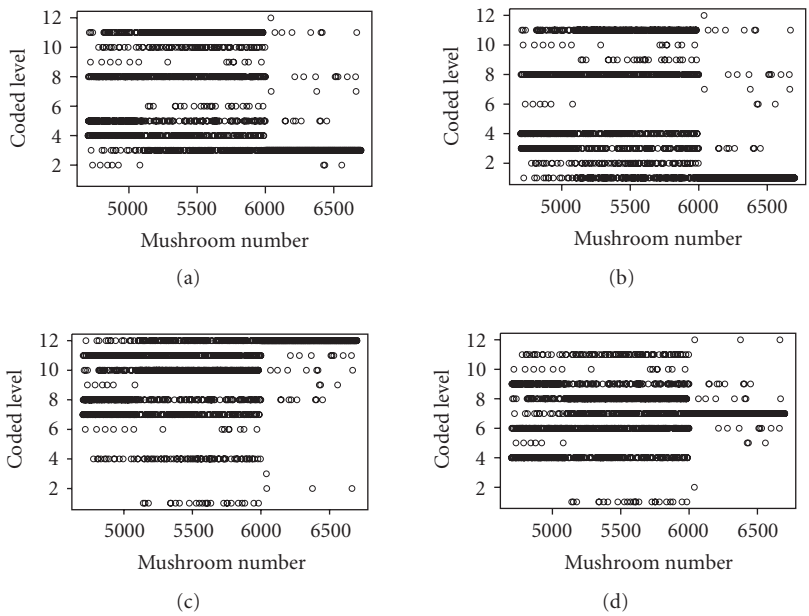



FIGURE 3.7. Graphical representations of the same sequence of 2000 mushrooms under the four different codings listed in the text, along with the standard deviations of each numerical sequence: (a) coding A, standard deviation = 3.07; (b) coding B, standard deviation = 3.76; (c) coding C, standard deviation = 2.47; and (d) coding D, standard deviation = 1.76.

The primary motivation for considering them here is that they are all computable from the empirical frequencies $p_i = n_i/N$ associated with the $m$ distinct values $c_i$ of an arbitrary categorical variable $c$, where $n_i$ denotes the number of times the value $c_i$ occurs in a sequence of $N$ observations. In particular, all of these measures are based entirely on the counting characterization discussed in Section 3.2, where the subset $\Lambda$ corresponds to the set $\Sigma$ of all possible values for the categorical variable of interest.

Of the five axiomatic characterizations of a good interestingness measure $I$ offered by Hilderman and Hamilton, the following three are of interest to us.

(P1) minimum value principle: the measure should exhibit its minimum value for the uniform case, $p_i = 1/m$ for all $i$.

(P2) maximum value principle: the measure should exhibit its maximum value for the discrete case, $p_i = 1$ for some $i$ and $p_j = 0$ for all $j \neq i$.

(P4) permutation invariance principle: the value of the measure should be invariant under any relabelling of the categories $c_i$.

The statements of P1 and P2 given here differ slightly from those of Hilderman and Hamilton, who do not admit empty categories (i.e., they require $n_i \geq 1$ for all $i$, implying that $p_i > 0$ for all $i$), but the issue of empty categories is an important one in practice, so we relax this condition to allow $p_i = 0$. In particular, an important issue in the moving-window measures considered here is the distinction between

an *internally categorized variable* $c$, whose possible values are specified completely by the observed data, implying that $n_i \geq 1$ for all $i$, and an *externally categorized variable*, whose possible values are specified independently of the data, admitting the possibility that $n_i = 0$ for some $i$. For example, the attribute *gill-spacing* included in the mushroom dataset can, according to the accompanying documentation for the dataset, assume any one of 3 possible values (close (c), crowded (w), or distant (d)), but in the dataset, only 2 of these values are actually observed. Hence, if we regard gill-spacing as an internally categorized variable, $c_i$ only assumes the 2 values close (c) and crowded (w) and $p_i > 0$ for both of these values, while if we regard this variable as externally categorized, it can assume all three values, of which one (distant (d)) is associated with a zero empirical frequency $p_i$. The distinction is particularly important for moving-window characterizations where an internal categorization can exhibit a strongly varying number of levels as the window moves through the data sequence. In contrast, external categorization permits the number of levels for the variable to remain fixed. As a consequence, external categorization leads to much more useful moving-window characterizations.

Of the 13 interestingness measures considered by Hilderman and Hamilton, eleven satisfy axioms P1, P2, and P4 described above. As they are described, not all of these measures satisfy axioms P1 and P2 directly, but they can all be normalized to do so, and this normalization actually reduces three of these measures to others from the list. Of the remaining eight, we have chosen the following four normalized interestingness measures.

(1) The normalized Simpson (variance) measure:

$$I_1 = \frac{m \sum_{i=1}^{m} p_i^2 - 1}{m - 1}. \tag{3.12}$$

(2) The normalized Shannon (entropy) measure:

$$I_2 = 1 + \frac{1}{\ln m} \sum_{i=1}^{m} p_i \ln p_i. \tag{3.13}$$

(3) The normalized Gini (mean difference) measure:

$$I_3 = \frac{1}{2(m-1)} \sum_{i=1}^{m} \sum_{j=1}^{m} |p_i - p_j|. \tag{3.14}$$

(4) The normalized Bray measure:

$$I_4 = \left(\frac{m}{m-1}\right)\left[1 - \sum_{i=1}^{m} \min\left\{p_i, \frac{1}{m}\right\}\right]. \tag{3.15}$$

Note that all of these interestingness measures are normalized so that $I = 0$ corresponds to the uniform case $p_i = 1/m$ and $I = 1$ corresponds to the discrete case $p_i = 1$ for some $i$ and $p_j = 0$ for all $j \neq i$. All other empirical frequency

distributions exhibit intermediate values of $I$, strictly between these limits. It is also important to note that we assume $m > 1$ here: this is not a significant restriction for externally categorized variables since an externally categorized variable with a single level is a constant, but the restriction is important for internally categorized variables, particularly if they are based on a moving data window, since locally constant segments within a longer sequence frequently occur in practice. This observation is one of the main reasons we restrict consideration to externally categorized variables here.

Moving-window implementations of these normalized interestingness measures are implemented as follows. As in the case of the moving window filters discussed in Section 3.4, computations are based on a moving data window of the form

$$\mathbf{w}_k = \{x_{k-K}, \ldots, x_{k-1}, x_k, x_{k+1}, \ldots, x_{k+K}\}, \tag{3.16}$$

where $\{x_k\}$ denotes the data sequence to be characterized and $K$ is an integer window half-width parameter. In particular, note that $\mathbf{w}_k$ represents a moving data window of total width $2K + 1$, centered at data observation $x_k$ and including the $K$ immediate past data samples and the $K$ immediate future data samples. The results presented here characterize a data sequence of total length $N$ for all $k$ from $k = K + 1$ to $k = N - K$ to avoid *end effects* (i.e., artifacts arising from lack of data for $k < K + 1$ or $k > N - K$). The alternative approach of data sequence extension popular in the nonlinear filtering literature is discussed in Section 3.4.

Four moving-window characterizations of the mushroom gill-color data sequence are shown in Figure 3.8, all based on a moving data window of width 21 ($K = 10$). That is, each of the four plots applies one of the interestingness measures described above to the moving data windows defined for $k = K + 1$ to $k = N - K$ and displays the resulting interestingness value $I(k)$. Figure 3.8(a) shows the results obtained for the Simpson measure defined in (3.12), Figure 3.8(b) shows the corresponding results for the Shannon measure defined in (3.13), Figure 3.8(c) shows the Gini measure defined in (3.14), and Figure 3.8(d) shows the Bray measure defined in (3.15). It is clear from these plots that while the gross behavior of these four measures is the same, the details are quite different. In particular, all four of these measures indicate an abrupt change in the character of the sequence at $k \sim 6000$, but the range of variation seen in these four measures is quite different. Specifically, the Simpson measure suggests the strongest contrast between the sequence for $k \sim 6000$ to $k \sim 7000$ and the rest of the data sequence, while the Gini measure shows the smallest contrast. The Shannon and Bray measures are intermediate in behavior, with the Shannon measure closer to the Simpson measure and the Bray measure closer to the Gini measure.

In addition to the choice of interestingness measure, another extremely important characterization choice is that of the moving-window width. Figure 3.9 shows how one of the interestingness measures just considered (the Simpson measure) varies as a function of the half-width parameter $K$ for the mushroom gill-color data sequence. Figure 3.9(a) shows the results obtained with an 11-point

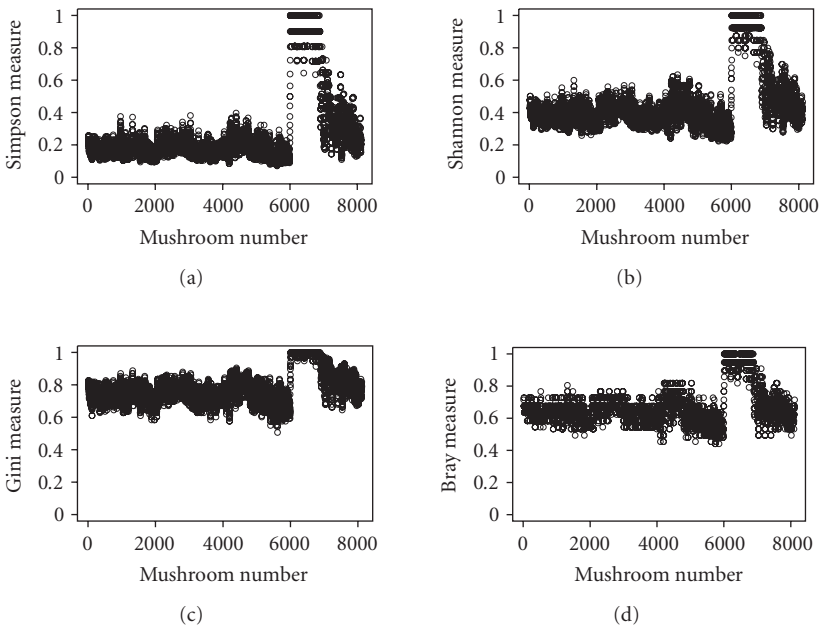



FIGURE 3.8. Application of four moving-window interestingness measures to the gill-color data sequence from the mushroom dataset. In all cases, the window width is 21 points ($K = 10$) and the four measures considered are (a) Simpson, (b) Shannon, (c) Gini, and (d) Bray.

moving window ($K = 5$), Figure 3.9(b) shows the result obtained before with a window of width 21 ($K = 10$), Figure 3.9(c) shows the results for a window of width 51 ($K = 25$), and Figure 3.9(d) shows the results for a window of width 101 ($K = 50$). Since increasing the window width decreases the effective bandwidth of the characterization, the resulting interestingness measure is less able to distinguish localized features. Conversely, the variability of the results decreases substantially as the window width increases, a consequence of the fact that each $I(k)$ value is based on a larger data subsample. This tradeoff is exactly analogous to that seen in moving-window characterizations for real data sequences like the short-time Fourier transform [15].

Since all of the window widths considered here are small compared to the total length of the data sequence, the "convolutional broadening" effects of the moving data window are not severe enough to offset the smoothness of the results even for the widest window considered. Hence, the results obtained for $K = 50$ here may be the most useful in directing us to the most interesting portions of this data sequence. In particular, note that in addition to the dramatic plateau between $k \sim 6000$ and $k \sim 7000$ noted earlier (and clearly visible in all of the numerically coded representations shown in Figure 3.7), there is evidence in the results for $K = 50$ both of a secondary plateau of much lower amplitude for $k$ between $k \sim 4200$ and $k \sim 4800$, and of what appears to be a significant difference

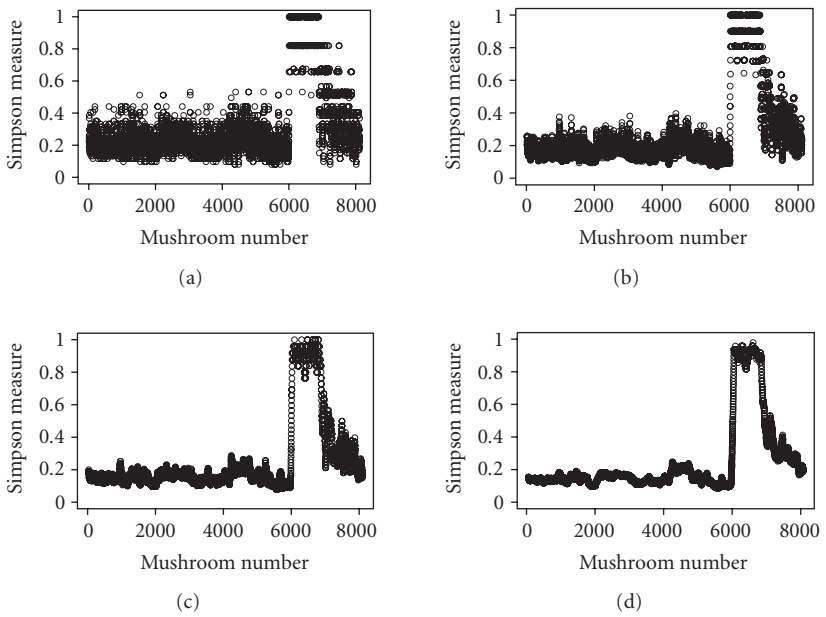



FIGURE 3.9. Effects of moving-window width on the Simpson interestingness measure computed from the mushroom gill-color data sequence. Results shown are for window widths of (a) 11 points ($K = 5$), (b) 21 points ($K = 10$), (c) 51 points ($K = 25$), and (d) 101 points ($K = 50$).

in the character of the data sequence for $k \gtrsim 7000$ and $k \lesssim 6000$. Note that neither of these features is at all evident in the results shown for $K = 5$.

### 3.4. Filters on finite sets

Digital filters represent a very large class of sequence transformations designed to map a given sequence $\{x_k\}$ into another, related sequence $\{y_k\}$ that is "better" or "more useful" in some application-specific sense than the original sequence $\{x_k\}$. As noted in the introduction to this chapter, there is an extremely well-developed theory of linear filters for real-valued sequences, which can be used for smoothing to eliminate low-level noise, sharpening to enhance important contrasts, or a variety of other applications [24, 48]. In addition, since linear filters are inadequate for some tasks even in the case of real-valued sequences, there is also a large body of results available for the design and analysis of nonlinear digital filters; see, for example, the book by Astola and Kuosmanen [6]. Some of these nonlinear filters are more generally applicable to sequences of ordinal variables defined on finite sets; important examples include both the median filter [20] and more complex combinations of order statistics like the LUM filter [25]. In general, however, the problems of designing useful filters and analyzing their behavior become more challenging as we reduce the level of mathematical structure imposed on the sequence-value set. As a consequence, the problems of designing and analyzing

useful filters for nominal data sequences is especially challenging. In particular, note that none of the filters just described is applicable to this case.

Since there appear to be more open questions than available results in the design of digital filters for nominal data sequences, the primary focus of the following discussion is on key ideas and insights that should be useful in developing these filters. The specific problem considered here is that of designing moving-window filters $\mathcal{F}$ that map one sequence $\{x_k\}$ taking values in the finite set $\Sigma$ into another such sequence $\{y_k\}$. Adopting the moving-window assumption provides a simple, explicit basis for formulating the filtering problem, and while it does entail some loss of generality (i.e., it excludes recursive filter structures), this restriction is almost universally imposed on nonlinear filters for real-valued data sequences. For example, the overwhelming majority of the nonlinear filters discussed by Astola and Kuosmanen are moving-window filters [6].

An important issue that arises with moving-window filters is the problem of end effects, discussed briefly in Section 3.3.3 in connection with moving-window-based sequence characterizations. While that discussion adopted a sequence truncation strategy to deal with end effects (i.e., the window-based characterization $I(k)$ was not evaluated for the first $K$ or the last $K$ elements of the original data sequence), this strategy is generally undesirable in filtering problems, where a filtered sequence $\{y_k\}$ of the same length as the original sequence $\{x_k\}$ is generally wanted. A standard approach to this problem is via sequence extension, replicating the first value of the sequence $K$ times and prepending it to the original, with the last value of the sequence handled similarly. That is, the original data sequence $\{x_k\}$ of length $N$ is replaced with a new sequence $\{\tilde{x}_k\}$ of length $N + 2K$, given by

$$\tilde{x}_k = \begin{cases} x_1, & k = -K+1, \ldots, -1, 0, \\ x_k, & k = 1, 2, \ldots, N, \\ x_N, & k = N+1, \ldots, N+K. \end{cases} \tag{3.17}$$

Applying a moving-window filter of width $2K + 1$ based on the data window $\mathbf{w}_k$ defined in (3.16) then generates an output sequence $\{y_k\}$ that is well defined for $k = 1, 2, \ldots, N$.

More specifically, the moving-window filter design problem considered here is that of specifying the function $\Phi : \Sigma^{2K+1} \to \Sigma$ in the equation

$$y_k = \Phi(x_{k-K}, \ldots, x_k, \ldots, x_{k+K}). \tag{3.18}$$

As in the case of groupoids discussed in Section 3.2, it is important to note that the number of possible filters (i.e., the number of functions that can be specified) is much too large to attempt to characterize completely. Specifically, note that the function $\Phi(\cdots)$ is specified by a set of response values, one for each possible combination of input data values in its $2K + 1$ arguments. If $\Sigma$ contains $M$ elements, it

follows that the number of responses required to specify a single function $\Phi(\cdots)$ is $M^{2K+1}$. Since each of these values can be specified independently and can assume any of the $M$ values from $\Sigma$, the total number of distinct filter functions $\Phi(\cdots)$ is

$$N_\Phi(M,K) = M^{M^{2K+1}}. \tag{3.19}$$

To see how large this number is even for moderate values of $M$ and $K$, note that $N_\Phi(2,4) \simeq 1.34 \times 10^{154}$, $N_\Phi(3,2) \simeq 8.72 \times 10^{115}$, and $N_\Phi(6,1) \simeq 1.20 \times 10^{168}$. As a practical consequence, then, there are so many possible filter functions that the design of *useful* filters on finite sets requires careful thought. The following two sections offer some potentially useful ideas.

### 3.4.1. The Dedekind majority filter

One possible filter for sequences of nominal data values is the *Dedekind majority filter*, based on the Dedekind groupoid and the incomplete data partial order introduced in Section 3.2. The motivating problem is that of data cleaning, where a frequently effective solution is the *Hampel filter* [39, 44], a member of the class of *decision-based filters* [6]. There, the basic idea is that isolated, large-amplitude noise spikes represent outliers in the moving data window $\mathbf{w}_k$. The Hampel filter strategy is to test the central observation $x_k$ for its concordance with the rest of the values in the data window. If $x_k$ is deemed inconsistent, it is replaced with a more representative value computed from the data window; otherwise, the data value $x_k$ is unmodified.

Because it measures the distance from $x_k$ to a representative data value computed from $\mathbf{w}_k$, the Hampel filter is not applicable to sequences defined on finite sets. One possible extension of this idea to such sequences is the following. First, define $b_k(\mathbf{w}_k)$ as the "best" value in the data window $\mathbf{w}_k$:

$$b_k(\mathbf{w}_k) = \begin{cases} x^* & \text{if } \kappa(x^*) \geq K+1, \\ \text{NULL} & \text{otherwise,} \end{cases} \tag{3.20}$$

where $\kappa(x^*)$ is the number of times the value $x^*$ appears in $\mathbf{w}_k$. Note that the value $x^*$ appearing in this definition is unique if it is defined, since at most, one value can appear $K+1$ or more times in a sequence of length $2K+1$. The Dedekind majority filter is defined by

$$y_k = b_k(\mathbf{w}_k) \circ x_k = \begin{cases} b_k(\mathbf{w}_k) & \text{if } b_k(\mathbf{w}_k) \prec x_k, \\ x_k & \text{if } b_k(\mathbf{w}_k) \not\prec x_k, \end{cases} \tag{3.21}$$

where $\circ$ is the binary operation defining the Dedekind groupoid and $\prec$ is the incomplete data partial order defined in Section 3.2.

To see how this filter works, first recall that NULL is the unit element for the Dedekind groupoid associated with the incomplete data partial order. Hence, if no

element $x^*$ appears $K+1$ or more times in $\mathbf{w}_k$, it follows that $b(\mathbf{w}_k) = \text{NULL}$ and the filter output is $y_k = \text{NULL} \circ x_k = x_k$, regardless of the value of $x_k$. Conversely, if a majority element $x^*$ does exist, then either $x_k = x^*$, implying that

$$y_k = b(\mathbf{w}_k) \circ x_k = x_k \circ x_k = x_k, \tag{3.22}$$

or else $\kappa(x^*) > \kappa(x_k)$, implying that

$$y_k = b(\mathbf{w}_k) \circ x_k = x^* \circ x_k = x^* \neq x_k. \tag{3.23}$$

In words, the Dedekind majority filter replaces the central value $x_k$ with the window majority value $x^*$ whenever this majority value is well defined (i.e., non-NULL) and distinct from $x_k$; otherwise, the filter leaves the central data value unmodified, exactly as in the case of the Hampel filter for real-valued data sequences.

The advantage of describing this filter in terms of the incomplete data partial order and its associated Dedekind groupoid is that the filter defined in (3.21) provides a basis for immediately generalizing this simple majority filter to something else by replacing the "best" element in the data window $b_k(\mathbf{w}_k)$ defined in (3.20) with a different choice of "best element." One potentially promising application of this idea is in the development of multivariable data cleaning filters discussed briefly in Section 3.5.

### 3.4.2. Bottom-up filter design

The filter design problem of interest here can be approached from at least two opposite perspectives: a top-down approach, in which desirable qualitative behavior is specified from which we attempt to characterize functions $\Phi(\cdots)$ that exhibit this behavior [40], or a bottom-up approach in which we seek behavior-preserving ways of combining known examples of useful filters into more complex structures [41]. The first of these approaches leads us in the direction of functional equations, which attempt to represent all functions exhibiting certain desirable qualitative characteristics [1]. In favorable cases, these equations can be solved to yield very useful results, but this task becomes more difficult in cases like the one considered here where we have very little mathematical structure to exploit. In contrast, a number of simple bottom-up constructions that have led to useful nonlinear filters for real-valued sequences extend easily to the finite-set case.

The simplest bottom-up filter design approach is *cascade connection*, in which the output of one filter $\mathcal{F}_1$ serves as the input of a second filter $\mathcal{F}_2$. For convenience in the following discussions, denote this interconnection as $\mathcal{F}_2 \circ \mathcal{F}_1$, where it is understood that filter $\mathcal{F}_1$ acts on the original input sequence and filter $\mathcal{F}_2$ acts on the output of this first filter. Similarly, longer filter cascades will be denoted by $\mathcal{F}_3 \circ \mathcal{F}_2 \circ \mathcal{F}_1$, and so forth. The cascade construction is reasonably popular even

for linear filters in the real-valued case, even when both filters $\mathcal{F}_1$ and $\mathcal{F}_2$ are the same [24, Section 6.6]. This idea is also quite popular for nonlinear real-valued sequence filters, as in the case of the data sieve described by Bangham, consisting of the cascade interconnection of several median filters [10]. Note that so long as both filters map sequences taking values in the finite set $\Sigma$, so do their cascade interconnections. Also, note that if filters $\mathcal{F}_1$ and $\mathcal{F}_2$ are distinct, the cascade interconnections $\mathcal{F}_1 \circ \mathcal{F}_2$ and $\mathcal{F}_2 \circ \mathcal{F}_1$ are generally not equivalent. (This statement does not hold for linear filters, where the order of interconnection is irrelevent, but it does generally hold for nonlinear filters like the ones considered here.)

The cascade interconnection strategy is an extremely useful one here since it is applicable to filters acting on finite sets, but it is somewhat limited in its flexibility. Another construction that is also completely applicable to the finite-set case is the more flexible *FIR clone construction* [41]. The term *clone* as it is used here comes from universal algebra, where it is defined as follows [52, page 11]. Suppose $q$ and $r$ are arbitrary integers, $\phi_i : \Sigma^q \to \Sigma$ are arbitrary mappings for $i = 1, 2, \ldots, r$, and $G : \Sigma^r \to \Sigma$ is another arbitrary mapping. A *clone of* $\Sigma$ is a collection of such mappings that is closed under *clone superpositions*, of the form

$$\Phi(\mathbf{x}) = G\big(\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots, \phi_r(\mathbf{x})\big), \tag{3.24}$$

for all $\mathbf{x} \in \Sigma^q$. Further, a clone must also contain all *projections* of the form

$$P_i(\mathbf{x}) = x_i, \tag{3.25}$$

where $x_i$ is the $i$th component of the vector $\mathbf{x}$. It is easy to show that the class of moving-window filters defined by (3.18) constitutes a clone, regardless of the set of admissible values for the sequence $\{x_k\}$ [41]. In addition, it is also straightforward to show that the cascade interconnection of two filters discussed above may be realized as a clone superposition involving the two filters and a suitably chosen set of projections [41]. The advantage of the clone construction is that it is more flexible, leading to multifilter composite structures like the FIR-median hybrid (FMH) filter class [26] or the LUM smoothing filter [25]. In addition, the clone construction leads naturally to all *weighted* versions of the simpler filters, in which elements of the data window are *replicated* [41]. The key point here is that all of these results carry over to the finite-set case, providing a potentially very flexible approach to the bottom-up design of these filters.

Finally, it is worth briefly discussing two other bottom-up design approaches, one is a significant generalization of the other but both require the existence of a groupoid structure. In particular, suppose that $(\Sigma, \diamond)$ constitutes a groupoid for some binary operation $\diamond$ and consider the *parallel interconnection* of two filters $\mathcal{F}_1$ and $\mathcal{F}_2$, here assumed to be moving-window filters based on the functions $\Phi_1(\cdots)$ and $\Phi_2(\cdots)$, respectively. In the parallel interconnection, both filters are driven by a common input sequence $\{x_k\}$, and their outputs are combined via

the binary operation $\diamond$ to give

$$y_k = \Phi_1(x_{k-K}, \ldots, x_k, \ldots, x_{k+K}) \diamond \Phi_2(x_{k-K}, \ldots, x_k, \ldots, x_{k+K}). \tag{3.26}$$

Clearly, this construction can be extended by forming parallel combinations of more than two filters, but a much more flexible extension is obtained by noting that the cascade interconnection is always possible, so the component filters in these parallel interconnections can themselves be composites obtained from the cascade interconnections of simpler filters. In this way, we can build up much more complex structures, an approach that has been pursued fairly extensively in the block-oriented modeling of biological systems using linear dynamic models and memoryless nonlinearities [14].

### 3.5. Variations and extensions

The discussion up to this point has focused on the development of signal processing tools for a univariate sequence $\{x_k\}$ of nominal data values. The following paragraphs briefly consider some extensions of these ideas to two more complex settings: the case of multivariable sequences, and the case of sequences of more general data objects like graphs or XML documents. (The abbreviation XML stands for Extensible Markup Language, a very flexible hierarchical data structure that is rapidly growing in popularity both for data representation and data exchange, in part because it accommodates both data elements themselves and the metadata describing those elements.)

With multivariable data sequences, it is important to distinguish between the case where all of the components of the data vector $\mathbf{x}_k$ are nominal variables and the mixed-variable case where these components may be of different types (e.g., ordinal or real-valued). If all of the components are nominal variables, the simplest way (conceptually) of extending the results presented here would be to convert the vector $\mathbf{x}_k$ of $p$ variables defined on the sets $\{\Sigma_i\}$ to a single categorical variable. That is, each $p$-tuple $(x_1, x_2, \ldots, x_p)$ defined on $\Sigma_1 \times \Sigma_2 \times \cdots \times \Sigma_p$ would be treated as a single categorical variable that can assume any value in a set $\Sigma^*$ of size $M = M_1 \cdot M_2 \cdot \cdots \cdot M_p$, where $M_i$ is the number of values in the original set $\Sigma_i$. Despite the apparent simplicity of this approach, it suffers from two disadvantages. First, the size of the composite set $\Sigma^*$ can quickly become very large, potentially causing computational difficulties. For example, applying this approach to a four-dimensional vector of nominal variables taking 12, 8, 6, and 3 values converts it to a single variable with 1,728 distinct values. The second disadvantage of this composite-univariate strategy is more fundamental: it provides no mechanism for dealing with any associations that may exist between the different variables. By retaining the multivariate nature of the original problem formulation, we can attempt to incorporate categorical variable association measures and other results from categorical data analysis [3] into our signal processing strategies.

In the case of mixed variables, one possible strategy would be to convert other data types to categorical approximations. In fact, a related idea is described by Dain et al. for the problem of rule learning involving continuous-valued features

$f$ which are split at some threshold value $c$, defining a binary variable that is 0 if $f < c$ and 1 if $f \geq c$ [16]. While such an approach can be used to convert a problem with mixed data types into one with only categorical data types, this conversion is necessarily an approximation (whose quality may be difficult to assess) and, perhaps more seriously, it entails a loss of potentially exploitable mathematical structure. For example, in applications like data cleaning, it may be possible to use more highly structured (i.e., ordinal or real-valued) variables as a basis for ranking the "quality" or "reliability" of each vector in a vector-valued data sequence $\{\mathbf{x}_k\}$. This additional information could then be used to construct a "best" value $b_k^i(\mathbf{w}_k)$ for a nominal value in the $i$th component of the composite data vector $\mathbf{x}_k$. This result could then be used in a modified version of the Dedekind majority filter described in Section 3.4.1.

For sequences of more general data objects, two key observations are first that these objects may have very useful additional mathematical structure that can be exploited, and second that the adverse consequences of imposing unwarranted structure (e.g., unnatural order structure) can be as severe as in the simpler case of nominal data sequences considered here. An example illustrating the first point is the procedure described by Agarwal and Pregibon [2] for efficiently updating very large communication network graphs. There, the key idea was to define vector addition and scalar multiplication operations (using edge weights) that formed the basis for defining a linear vector space of graphs. Similarly, Garofalakis and Kumar [21] describe a distance measure between XML documents. The key point is that in both of these cases, significant but relevant mathematical structure can be imposed on complex data objects in ways that are not obvious from the outset. The development and exploitation of these mathematical structures in characterizing and filtering (e.g., cleaning) sequences of these complex data objects appears to be a very promising research area.

## 3.6. Summary

The primary focus of this chapter has been on the development of useful signal processing tools for sequences taking values in a finite set $\Sigma$. In marked contrast to the case of real-valued sequences, the extreme sparsity of available mathematical structure for sequences taking values on finite sets greatly limits our options in developing these tools. As a specific and important example, the fact that nontrivial finite sets are not closed under the usual operations of addition and multiplication, even if the data values are encoded numerically, means that linearity has no role to play in these signal processing problems. In addition, the results presented here have also emphasized the undesirability of using data-coding strategies that impose unnatural orders on the finite set $\Sigma$, as the results obtained can be strongly influenced by artifacts of this induced ordering.

In cases where there is no additional mathematical structure on $\Sigma$, corresponding to the nominal data model in statistics and cluster analysis, the basic characterization operations available are counting and comparison, as discussed in Section 3.2. Despite the apparent paucity of these operations, one of the primary

points of this chapter has been to illustrate that in favorable cases, they can form the basis for useful signal processing tools for sequences defined on finite sets. In particular, Section 3.3.1 showed how the classical Blackman-Tukey spectrum estimator can be simply extended to the finite-set case, along with much of its associated practical machinery (e.g., lag windows). Similarly, the availability of characterizations like the interestingness measures described by Hilderman and Hamilton [28] that are inherently based on discrete probabilities also provide a basis for finite-set signal processing tools, obtained by simply combining these measures with suitably chosen moving data windows. Like the short-time Fourier transform and other moving-window data characterizations for real-valued sequences, the resulting finite-set characterizations can highlight interesting structure in nominal variable sequences, as illustrated in Section 3.3.3.

The lack of additional mathematical structure in nominal data sequences seems to have the greatest impact on the digital filter design problem, and the question of how best to design these filters seems to be largely open at present. One avenue that appears promising is that of identifying and exploiting useful induced partial-order information in multivariable problems where additional variables (which may be ordinal or real-valued) can be used as a reasonable basis for comparing categorical values within a moving data window, an idea discussed briefly in Section 3.5. More generally, the design of moving-window data filters appears to be an area where the existence of even-limited mathematical structure (e.g., partial orders or groupoids) can be strongly exploited. Two specific illustrations of this point are the parallel combination and closely related block-oriented composite filter structures discussed in Section 3.4.2. Conversely, even in cases where there is no additional mathematical structure to exploit, once a few useful simple filters have been developed for the application at hand, bottom-up design strategies based on the cascade and clone constructions described in Section 3.4.2 can be exploited to develop a wide variety of potentially useful extensions.

Finally, Section 3.5 briefly considered a number of possible extensions of the basic ideas presented here to other settings, beyond the case of univariate sequences of nominal variables. One of the main points of that discussion was the fact that these developments should first fully exploit whatever additional mathematical structure these applications exhibit, and second carefully avoid formulations that impose unwarranted structure that can cause significant processing artifacts. As the examples of linear structure on graphs [2] and distances between XML documents [21] illustrate, however, it is frequently possible to impose reasonable and highly exploitable mathematical structure on these problems, even in cases where this is not immediately obvious.

## Bibliography

[1] J. Aczél and J. Dhombres, *Functional Equations in Several Variables*, Cambridge University Press, Cambridge, UK, 1989.

[2] D. Agarwal and D. Pregibon, "Enhancing communities of interest using Bayesian stochastic blockmodels," in *Proceedings of 4th SIAM International Conference on Data Mining (SDM '04)*, pp. 291–299, Lake Buena Vista, Fla, USA, April 2004.

[3] A. Agresti, *Categorical Data Analysis*, John Wiley & Sons, New York, NY, USA, 2nd edition, 2002.

[4] M. Ahdesmäki, H. Lähdesmäki, R. K. Pearson, H. Huttunen, and O. Yli-Harja, "Robust detection of periodic time series measured from biological systems," *BMC Bioinformatics*, vol. 6, no. 1, p. 117, 2005.

[5] J. Albert, "Algebraic properties of bag data types," in *Proceedings of 17th International Conference on Very Large Data Bases (VLDB '91)*, pp. 211–219, Barcelona, Catalonia, Spain, September 1991.

[6] J. Astola and P. Kuosmanen, *Fundamentals of Nonlinear Digital Filtering*, CRC Press, Boca Raton, Fla, USA, 1997.

[7] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom, "Models and issues in data stream systems," in *Proceedings of 21st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS '02)*, pp. 1–16, Madison, Wis, USA, June 2002.

[8] B. Babcock, M. Datar, R. Motwani, and L. O'Callaghan, "Maintaining variance and $k$-medians over data stream windows," in *Proceedings of 22nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS '03)*, pp. 234–243, San Diego, Calif, USA, June 2003.

[9] C. Bandt and B. Pompe, "Permutation entropy—a natural complexity measure for time series," *Physical Review Letters*, vol. 88, no. 17, p. 174102, 2002.

[10] J. A. Bangham, "Properties of a series of nested median filters, namely the data sieve," *IEEE Transactions on Signal Processing*, vol. 41, no. 1, pp. 31–42, 1993.

[11] J. Beran, *Statistics for Long-Memory Processes*, Chapman & Hall/CRC, New York, NY, USA, 1994.

[12] K. M. Bloch and G. R. Arce, "Analyzing protein sequences using signal analysis techniques," in *Computational and Statistical Approaches to Genomics*, W. Zhang and I. Shmulevich, Eds., pp. 113–124, Kluwer Academic, Boston, Mass, USA, 2002.

[13] M. Buchner and S. Janjarasjitt, "Detection and visualization of tandem repeats in DNA sequences," *IEEE Transactions on Signal Processing*, vol. 51, no. 9, pp. 2280–2287, 2003.

[14] H.-W. Chen, "Modeling and identification of parallel nonlinear systems: structural classification and parameter estimation methods," *Proceedings of the IEEE*, vol. 83, no. 1, pp. 39–66, 1995.

[15] L. Cohen, *Time-Frequency Analysis*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1995.

[16] O. Dain, R. K. Cunningham, and S. Boyer, "IREP++, a faster rule learning algorithm," in *Proceedings of 4th SIAM International Conference on Data Mining (SDM '04)*, pp. 138–146, Lake Buena Vista, Fla, USA, April 2004.

[17] C. J. Date, *An Introduction to Database Systems*, Addison-Wesley, Reading, Mass, USA, 7th edition, 2000.

[18] B. A. Davey and H. A. Priestley, *Introduction to Lattices and Order*, Cambridge University Press, Cambridge, UK, 1990.

[19] L. L. Dornhoff and F. E. Hohn, *Applied Modern Algebra*, Macmillan, New York, NY, USA, 1978.

[20] N. C. Gallagher Jr. and G. L. Wise, "A theoretical analysis of the properties of median filters," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 29, no. 6, pp. 1136–1141, 1981.

[21] M. Garofalakis and A. Kumar, "Correlating XML data streams using tree-edit distance embeddings," in *Proceedings of 22nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS '03)*, pp. 143–154, San Diego, Calif, USA, June 2003.

[22] A. D. Gordon, *Classification*, Chapman & Hall/CRC, New York, NY, USA, 2nd edition, 1999.

[23] D. Gusfield, *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*, Cambridge University Press, Cambridge, UK, 1997.

[24] R. W. Hamming, *Digital Filters*, Prentice-Hall, Englewood Cliffs, NJ, USA, 3rd edition, 1989.

[25] R. C. Hardie and C. Boncelet, "LUM filters: a class of rank-order-based filters for smoothing and sharpening," *IEEE Transactions on Signal Processing*, vol. 41, no. 3, pp. 1061–1076, 1993.

[26] P. Heinonen and Y. Neuvo, "FIR-median hybrid filters," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 35, no. 6, pp. 832–838, 1987.

[27] M. A. Hernández and S. J. Stolfo, "Real-world data is dirty: data cleansing and the merge/purge problem," *Data Mining and Knowledge Discovery*, vol. 2, no. 1, pp. 9–37, 1998.

[28] R. J. Hilderman and H. J. Hamilton, "Evaluation of interestingness measures for ranking discovered knowledge," in *Proceedings of 5th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD '01)*, pp. 247–259, Hong Kong, April 2001.

[29] D. H. Johnson and W. Wang, "Symbolic signal processing," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '99)*, vol. 3, pp. 1361–1364, Phoenix, Ariz, USA, March 1999.

[30] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*, John Wiley & Sons, New York, NY, USA, 1990.

[31] S. M. Kay, *Modern Spectral Estimation: Theory and Application*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1988.

[32] N. D. Le, R. D. Martin, and A. E. Raftery, "Modeling flat stretches, bursts, and outliers in time series using mixture transition distribution models," *Journal of the American Statistical Association*, vol. 91, no. 436, pp. 1504–1515, 1996.

[33] C.-H. Lee, C.-R. Lin, and M.-S. Chen, "Sliding-window filtering: an efficient algorithm for incremental mining," in *Proceedings of 10th ACM International Conference on Information and Knowledge Management (CIKM '01)*, pp. 263–270, Atlanta, Ga, USA, November 2001.

[34] G. H. Lincoff, *The Audubon Society Field Guide to North American Mushrooms*, Alfred A. Knopf, New York, NY, USA, 1981.

[35] S. MacLane and G. Birkhoff, *Algebra*, Macmillan, New York, NY, USA, 2nd edition, 1979.

[36] R. D. Martin and D. J. Thomson, "Robust-resistant spectrum estimation," *Proceedings of the IEEE*, vol. 70, no. 9, pp. 1097–1115, 1982.

[37] C. L. Nikias and A. P. Petropulu, *Higher Order Spectral Analysis: A Nonlinear Signal Processing Framework*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1993.

[38] C. L. Nikias and M. Shao, *Signal Processing with Alpha-Stable Distributions and Applications*, John Wiley & Sons, New York, NY, USA, 1995.

[39] R. K. Pearson, "Outliers in process modeling and identification," *IEEE Transactions on Control Systems Technology*, vol. 10, no. 1, pp. 55–63, 2002.

[40] R. K. Pearson, "Scale-invariant nonlinear digital filters," *IEEE Transactions on Signal Processing*, vol. 50, no. 8, pp. 1986–1993, 2002.

[41] R. K. Pearson, M. Gabbouj, and J. Astola, "Nonlinear FIR filter clones," in *Proceedings of IEEE/Eurasip Workshop on Nonlinear Signal and Image Processing (NSIP '03)*, p. 5s, Trieste, Italy, June 2003.

[42] R. K. Pearson, T. Zylkin, J. S. Schwaber, and G. E. Gonye, "Quantitative evaluation of clustering results using computational negative controls," in *Proceedings of 4th SIAM International Conference on Data Mining (SDM '04)*, pp. 188–199, Lake Buena Vista, Fla, USA, April 2004.

[43] R. K. Pearson, G. E. Gonye, and M. Gabbouj, "Finite set DSP, with applications to DNA sequences," in *Proceedings of 12th European Signal Processing Conference (EUSIPCO '04)*, pp. 1899–1902, Vienna, Austria, September 2004.

[44] R. K. Pearson, *Mining Imperfect Data: Dealing with Contamination and Incomplete Records*, SIAM, Philadelphia, Pa, USA, 2005.

[45] G. Petre, "Dedekind groupoids for posets," *Annals of the University of Craiova, Mathematics and Computer Science Series*, vol. 31, pp. 74–78, 2004.

[46] M. B. Priestley, *Spectral Analysis and Time Series*, Academic Press, London, UK, 1981.

[47] D. Pyle, *Data Preparation for Data Mining*, Morgan Kaufmann, San Francisco, Calif, USA, 1999.

[48] L. L. Scharf, *Statistical Signal Processing: Detection, Estimation, and Time Series Analysis*, Addison-Wesley, Reading, Mass, USA, 1991.

[49] J. C. Schlimmer, "Concept acquisition through representational adjustment," Tech. Rep. 87-19, Department of Information and Computer Science, University of California, Irvine, Calif, USA, 1987.

[50] A. Schuster, "On the investigation of hidden periodicities with application to a supposed 26-day period of meterological phenomena," *Terrestrial Magnetism and Atmospheric Electricity*, vol. 3, pp. 13–41, 1898.

[51] S. Y. Sung, Z. Li, and P. Sun, "A fast filtering scheme for large database cleansing," in *Proceedings of 11th ACM International Conference on Information and Knowledge Management (CIKM '02)*, pp. 76–83, McLean, Va, USA, November 2002.

[52] Á. Szendrei, *Clones in Universal Algebra*, Les Presses de l'Université de Montréal, Montreal, Quebec, Canada, 1986.

Ronald K. Pearson: ProSanos Corporation, Harrisburg, PA 17101, USA
*Email*: ronald.pearson@prosanos.com

Moncef Gabbouj: Department of Information Technology, Institute of Signal Processing, Tampere University of Technology, 33101 Tampere, Finland
*Email*: moncef.gabbouj@tut.fi

# 4 Nonlinear signal modeling and structure selection with applications to genomics

Ioan Tabus, Jorma Rissanen, and Jaakko Astola

## 4.1. Introduction

Modeling is a prerequisite for the most fundamental signal processing tasks of signal analysis, detection, classification, denoising, and compression. While linear models are widely used, and they allow elegant theoretical and algorithmic developments, their nonlinear alternatives offer advantages in terms of modeling power and improved performance, which often eclipse the extra cost due to increased complexity. In this chapter we discuss nonlinear signal modeling with two main goals. The first is to present several examples of nonlinear models arising naturally in processing genomic data. The second is to discuss methods for evaluating the complexity of these nonlinear models with information theoretic methods.

Several signal processing methods are useful for processing genomic data. Microarray data come originally in the form of microarray images, which need to be preprocessed (denoised and segmented) before the quantities of interest, the gene expression ratios, are estimated from the image. All these stages benefit greatly from nonlinear techniques. The stages involving statistical inference on the gene expression values can also be formulated as optimal design problems for the structure and the parameters of certain nonlinear predictors. In this chapter we investigate mainly the gene prediction problem, and we propose nonlinear predictors in which the structure is selected by the minimum description length (MDL) principle.

We present first the Boolean, ternary, and perceptron predictors. We resort to the earliest form of the MDL principle, the two-part code, as a tool for selecting the proper size of the prediction window, because the selection is done on an immediate and intuitive description of the data and the model parameters. When comparing predictors of different complexities we show that the best description is achieved by the Boolean and the ternary predictors, for they give a better fit to the data with a lower model complexity. To illustrate the results, both synthetic and experimental data are used.

We then introduce a more advanced way to evaluate the overall cost of describing the data by Boolean regression models. These models are useful tools for various applications in nonlinear filtering, nonlinear prediction, classification and clustering, and data compression. We discuss the normalized maximum likelihood (NML) universal model for these classes of models. Examples of the problem of the discrimination of cancer types with the universal NML model for Boolean regression demonstrate the ability of the NML model to select sets of the feature genes that are capable of discrimination at error rates significantly smaller than what is achievable with other discrimination methods.

## 4.2. Preliminaries: modeling and predicting gene expressions

### 4.2.1. An introduction to gene expression data

One of the most important experimental advances in biology in the last decade is the development of high-throughput measurement methods for simultaneously probing thousands of gene expressions in the biological probes of interest. This permits probing deep into the functioning of the cell since at any moment the fundamental processes of life deal with the production of proteins, based directly on the information contained in the genes. The abundance (also called expression) of a certain gene in a biological sample is an indication of how intense the production of the corresponding protein or proteins is (in some cases the same gene can be used to generate several different proteins). However, the mere abundance of a certain gene in the cell at a certain time is not a guarantee that the corresponding proteins are produced, because the translation from gene to protein is mediated by a number of factors, like the availability of enabling enzymes in the cell, or the absence of certain inhibiting enzymes. The enzymes involved are proteins, produced on the information existing in other genes. This (overly) simplified image of the complex metabolic processes shows that the functioning of the cell at a certain time is related to a large extent to the amount or abundance of a number of genes that are connected functionally to enable biological processes.

The concept of the gene expression was introduced four decades ago with the discovery of messenger RNA when the theory of genetic regulation of protein synthesis was described [5]. The availability of cDNA microarrays makes it possible to measure simultaneously the expressions level for thousands of genes. Gene expression data obtained in microarray experiments may often be discretized as binary or ternary data, the values 1, 0, $-1$ carrying the meaning of over expressed, normal, and repressed, respectively, which are the descriptors needed when defining regulatory pathways.

For quite some time the interactions of the genes, either directly or through the Means of their corresponding proteins, were studied for specific biological processes, and the rules of each interaction, called a biological pathway, are usually specified in terms of enabling/inhibiting factors, sometimes represented graphically in the form of logical circuits. Studying the pathways is of fundamental importance in biology and medicine since understanding the mechanisms of a

biological process allows one to influence the process by modifying some of the enabling factors.

The new microarray technologies are able to provide measurements of thousands of gene expressions, sometimes by a differential method, where the expression of a gene in a sample of interest is measured against the expression in a reference sample. This is because the resulting gene expression ratio is less subject to the variation of the experimental and processing conditions. There are several alternative technologies for obtaining the gene expressions. Most of them are automated involving a robot arm which deposits the substances on the allocated spots on a microarray slide. The microarray slides are rectangular of dimensions in the order of centimeters having a large number of spots or wells arranged in a regular grid, each spot being the place where the abundance of a certain gene in a certain biological sample is measured. During the technology phases the genetic material from the probe at each spot is hybridized to the substance deposited on the spot, and by means of some color markings a high abundance of a gene in the probe is transformed into a high-intensity color component. In the case of a differential measuring system the probe and the reference are marked with red and green colors, respectively, resulting in a composite color of the spot image. What is called a gene expression results finally from processing the image of the spot with the goal of integrating the contribution of the hybridization process in the area of the spot, while taking into account the particular features of the technological process.

After the creation and processing of a microarray slide, the net result is in the form of one value of the expression for each of the thousands of genes represented in the array (or in case of differential measurements, two values, one in the probe of interest and the other in the reference probe). In a complete experiment many microarray slides are produced and measured, for example, one slide for each patient.

### 4.2.2. Prediction of the gene expression values

In the present chapter we concentrate on processing the outcomes of such complete experiments, where we have a number of $N$ patients, and for each we have the measurement of $p$ gene expressions. In the following we denote by $x_{i,j}$ the expression of gene $i$ for patient $j$.

One of the main research avenues opened by the existence of various microarray technologies is a generalization of the biological pathways. These are used to describe the activation of certain biological functions in terms of the interaction between the genes and the proteins from the explanations based only on a handful of genes or on virtually all the known genes. In recent years a main research goal turned out to be uncovering the network of interactions between the genes. Analogously to the measurement of gene expressions, new technologies begin to be introduced for the measurement of protein expressions; see, for example, [10]. It will soon become possible to study experimentally the interaction of the gene network in connection with the protein network. Since it is still believed that the interactions in the network are sparse, that is, each gene interacts only with a small

number of other genes, the methodologies for uncovering the network still rely mostly on local modeling, where individual functions are established for the regulation of every gene. However, a number of approaches exist for modeling the full network or large parts of it, when enough information exists about the temporal behavior of the genes [3, 22, 23].

The methods described in this chapter aim at identifying the nonlinear relationships that are able to predict the expression of one gene, given the expressions of other genes and possibly the values of environmental factors measured. We define a nonlinear gene predictor as

$$\hat{x}_{ij} = f\left(x_{i_1 j}, x_{i_2 j}, \ldots, x_{i_k j}\right), \tag{4.1}$$

where $f$ is a nonlinear function, $k$ the order of the predictor, and $\{i_1, i_2, \ldots, i_k\}$ the structural indices of the predictor. The task of determining a predictor refers to an ensemble of problems: first, select the model structure (find the order $k$ and the structural indices), and then specify the nonlinear function $f(\cdot)$.

A good prediction model is desired to make the predictions themselves, but it also should be able to uncover the biological relationships between genes. For the former goal the parameters in the model will necessarily have to be tuned to their optimal values, and the structure of the model will also have to be selected optimally to prevent overfitting.

### 4.2.3. Classification based on gene expression values

A second large class of applications, where the methods presented in this chapter are relevant, is the classification of diseases based on the transcriptome information [11, 21]. Since the gene expression information may play an essential role in the characterization of the metabolic processes in cells and tissues, it is tempting to attempt profiling the type of a disease based on the gene expressions of the biological sample studied. Most work has been done in this respect for identifying the types of the various forms of cancer disease. Again, apart from the interest in the performance of such a classifier, which soon promises to complement the traditional methods of diagnosis, there is also a separate interest of its own in the structure of the optimal classifier for the genes involved in separating the types of disease. This is because knowing the genes responsible for each type may be of help in designing specific treatments for each type of disease. Associating a class label $y$ with each type of a disease, and having available a training sample formed of the expressions $x_{ij}$ for each patient $j$ and gene $i$, the design problem for the classifier consists of determining the structure and the parameters of the nonlinear function

$$\hat{y}_j = f\left(x_{i_1 j}, x_{i_2 j}, \ldots, x_{i_k j}\right) \tag{4.2}$$

to be found to minimize a criterion based on the classification error.

### 4.3. Several classes of nonlinear functions and associated design methods

When selecting classes of nonlinear methods for modeling the gene interactions two important features of gene expression data need to be taken into account. On the one hand, the number of measured gene expressions in each slide is extremely high, even tens of thousands of genes. Therefore selecting the sparse structure of the predictor; that is, selecting a small number of genes out of all the existing genes raises the issue of having to compare a combinatorially very large number of candidates. On the other hand, the amount of microarray slides in almost all biological experiments is not particularly large, rarely exceeding 100 slides, which makes fitting the parameters in very complex models difficult due to the lack of enough observed cases. As a consequence, the models employed in gene expression predictors and classifiers are usually simple like the perceptron models, the Boolean models, and some simple neural networks. As hinted in the previous section, the selection of the structure of the predictor is of main interest in many applications; that is, one wants to find the genes that are involved in a certain biological process or in profiling a certain disease, while the detailed functional form of the predictor/classifier is of lesser importance.

#### 4.3.1. The Boolean model with binary inputs

The simplest model that biologists found appealing [6] from the earliest times is the Boolean model. Such a model operates on binary inputs, which creates the need to quantize the gene expressions before they can be used in the model. An advantage of the model is its one-to-one compatibility with the existing pathway representations based on Boolean schemes. As an example, Figure 4.1 illustrates the FAS signaling pathway (obtained from http://www.biocarta.com/) which is involved in immune surveillance to remove transformed cells and virus-infected cells. One of the interactions shows that *Caspase8* is produced when *Pro-caspase8* is on and inhibited when both *FAP* and *I-FLICE* are on. These models have the additional advantage of being simple to understand and manipulate by biologists with an easy integration of various local functions into an overall function.

Denoting the quantized binary expressions of *Pro-caspase8*, *FAP*, *I-FLICE*, and *Caspase8* as $x_1$, $x_2$, $x_3$ and $x_4$, respectively, the Boolean model can be expressed as

$$x_4 = x_1 \bar{x}_2 \bar{x}_3, \tag{4.3}$$

where by the overbar we denote the negation of a binary variable and by the product the logical "and" function.

In the following we use $x_{ij}$ for the expression value and sometimes the quantized expression value, and we specify in each context the number of the quantization levels and the centers of the Voronoi cells (hence $x_{ij}$ may be binary, ternary, or even continuous valued, depending on the context).

An optimal design of a Boolean predictor is straightforward, and we describe it here just for the sake of completeness and ease of comparison with the design

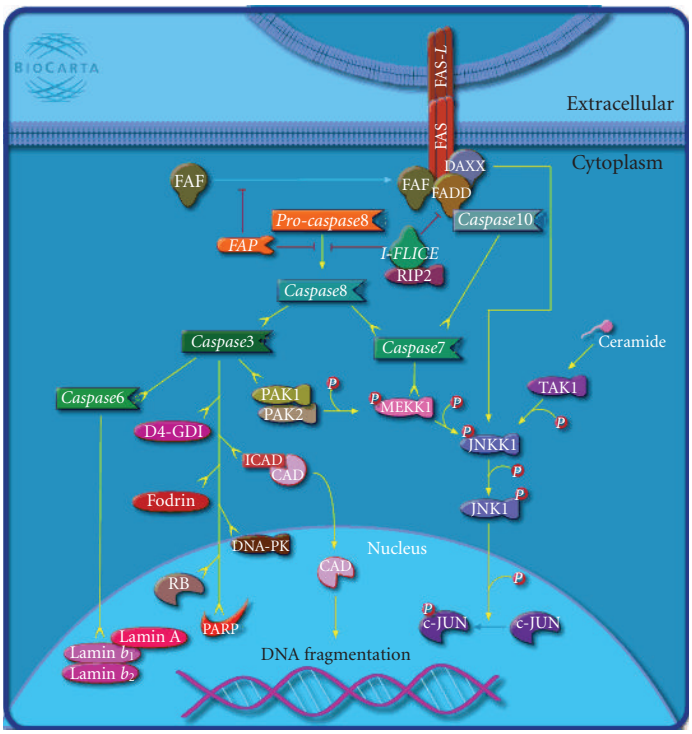



FIGURE 4.1. A biological pathway where gene and protein interactions are represented by logical operators. (Reproduced with permission from Biocarta, http://www.biocarta.com/.)

of more complex models. We illustrate the design for a given structure, say $\hat{x}_4 = f(x_1, x_2, x_3)$, from a training set of quadruples $\{(x_{1j}, x_{2j}, x_{3j}; x_{4j}),\ j = 1, \ldots, N\}$. In general the prediction given by the model $\hat{x}_{4j}$ and the measured value $x_{4j}$ are different, and we refer to their difference as modeling error

$$e_j = \hat{x}_{4j} \oplus x_{4j} = f(x_{1j}, x_{2j}, x_{3j}) \oplus x_{4j}, \tag{4.4}$$

where $\oplus$ denotes the "exclusive or" operator. We get that $\hat{x}_{4j} = e_j \oplus x_{4j}$ and $x_{4j} = e_j \oplus \hat{x}_{4j}$. The Boolean function minimizing the error criterion

$$\sum_{j=1}^{N} \left( f(x_{1j}, x_{2j}, x_{3j}) \oplus x_{4j} \right) \tag{4.5}$$

can easily be found by counting. For each triplet of binary values $(b_1, b_2, b_3) \in \mathcal{B}^3$ we have that the count $n_0(b_1, b_2, b_3)$ of the number of times $x_{4j}$ is 0 when the input variables have values $x_{1j} = b_1$, $x_{2j} = b_2$, $x_{3j} = b_3$, and similarly the count $n_1(b_1, b_2, b_3)$ of the number of times $x_{4j}$ is 1. Then the optimal Boolean function

minimizing the criterion (4.5) is given by

$$f(b_1,b_2,b_3)=\begin{cases}0 & \text{if } n_0(b_1,b_2,b_3)\geq n_1(b_1,b_2,b_3),\\ 1 & \text{otherwise.}\end{cases} \tag{4.6}$$

If not enough cases are observed, it is clear that for some combinations $(b_1,b_2,b_3)\in \mathcal{B}^3$ both counts $n_0(b_1,b_2,b_3)$ and $n_1(b_1,b_2,b_3)$ may be zero (or equal), and no clear-cut decision can be made. For the criterion (4.5) the undefined values do not matter, but when the model is applied to data outside the training set the nondefiniteness of the model for some input combinations should be solved in a principled way. These issues will be discussed later in Section 4.4.3.1.

The solution to the prediction problem presented is immediately applicable to the classification problem, where the training set is of the form $\{(x_{1j},x_{2j},\ldots,x_{kj};y_j),\ j=1,\ldots,N\}$ and the class predictor is $\hat{y}_j=f(x_{1j},x_{2j},\ldots,x_{kj})$.

### 4.3.2. Three predictors for ternary data

We consider next three predictor classes for ternary-valued gene expressions: a hard Boolean predictor, a hard ternary predictor, and finally, a perceptron predictor. We discuss the parameter estimation for all of them, and we present later the selection of the predictor order based on the MDL principle. The ternary-valued data are written as $\{0,1,2\}$. For $x\in\{-1,0,1\}$ use first the mapping $x\leftarrow x+1$ to transform $x$ to the set $\{0,1,2\}$.

#### 4.3.2.1. The specification and design of the hard Boolean predictor

Denote by $\underline{x}=[x_1,\ldots,x_k]$ prediction window of dimension $k$, where $x_i\in\{0,1,2\}$. Define the thresholded vectors $\underline{x}^{b_1}=[x_1^{b_1},\ldots,x_k^{b_1}]$ and $\underline{x}^{b_2}=[x_1^{b_2},\ldots,x_k^{b_2}]$, where

$$x_i^{b_1}=\begin{cases}0 & \text{if } x_i=0,\\ 1 & \text{if } x_i\geq 1,\end{cases}\qquad x_i^{b_2}=\begin{cases}0 & \text{if } x_i\leq 1,\\ 1 & \text{if } x_i=2.\end{cases} \tag{4.7}$$

The predictor is defined as

$$\hat{y}=f(\underline{x}^{b_1})+f(\underline{x}^{b_2}), \tag{4.8}$$

where $f(\cdot)$ is a Boolean function with $k$ variables.

The design of the Boolean predictor proceed as described in Section 4.3.1, where the training set is given by $\{[x_{1j}^{b_1},\ldots,x_{kj}^{b_1},y_j],[x_{1j}^{b_2},\ldots,x_{kj}^{b_2},y_j]:j=1,\ldots,N\}$, and the corresponding optimal Boolean function results from the counts as in (4.6).

#### 4.3.2.2. The specification and design of the ternary predictor

The optimal ternary predictor is found by quantization of the conditional expectation to three intervals:

$$\hat{y} = h^*(\underline{x}) = \begin{cases} 0 & \text{if } E(y \mid \underline{x}) \leq 0.5, \\ 1 & \text{if } 0.5 < E(y \mid \underline{x}) \leq 1.5, \\ 2 & \text{if } 1.5 < E(y \mid \underline{x}). \end{cases} \tag{4.9}$$

#### 4.3.2.3. The specification and design of the perceptron predictor

The perceptron is found by quantization of the best linear combination of samples in the predictor window to three intervals:

$$\hat{y} = \begin{cases} 0 & \text{if } \underline{w}^T + w_0 \underline{x} \leq 0.5, \\ 1 & \text{if } 0.5 < \underline{w}^T \underline{x} + w_0 \leq 1.5, \\ 2 & \text{if } 1.5 < \underline{w}^T \underline{x} + w_0. \end{cases} \tag{4.10}$$

The parameters $\underline{w}$ and $w_0$ can be obtained by the perceptron algorithm [7, 8].

### 4.3.3. Model selection based on a two-part code length

Our approach to prediction aims at finding flexible classes of models with good predictive properties, but we also consider the complexity of the models, the balance being set by the minimum description length (MDL) principle, as described in the next section.

#### 4.3.3.1. Modeling and data compression

Does a model capture the data generation mechanism? If yes, we can describe the data more concisely by describing the model and listing the errors than by copying the data in "raw" point by point, that is, encoding without a model or, perhaps, with the trivial identity model.

A well established fact is that both the complexity of models and the data can be measured by the universal yardstick of code length. The MDL principle [1, 12–14] considers the following axiom as a basis for a theory of modeling: given the data and a model class, select the model in the class with which both the data and the model can be encoded with the shortest code length. This means that the code length needed results from a two-part encoding process: encode first the model parameters and then the residuals, given the model defined by the parameters. When the parameters range over the reals they must be optimally quantized. In reality we only need the code length rather than the actually encoded strings for the data and the quantized parameters; by a deeper theory of coding the shortest

code length can be computed from a special parameter-free model that is *universal* for the model class. The same approach can be applied to different model classes, which may be compared by calculating the shortest code length for each and selecting the class with the shortest overall code length.

To apply the technique to our data we postulate a parametric model for the dependency $\hat{x}_\ell = f(x_1, \ldots, x_k, \theta)$, fit its parameters $\theta$ to the available data, and find the code length for the parameters and the residuals. If we get an important reduction in the code length with a model class, we claim that the dependency modeled is significant. If different model classes are consistently capturing dependencies, they are very likely to be rules of the nature.

*Encoding without a model.* We take as an example the case of $N = 30$ patients, which we use in our experiments. Encoding the expressions for the gene $q_\ell$ without a model requires $L = \log_2(3^{30}) = 47.55$ bits. Other codes for the target gene alone may be used, but we find it convenient to make no assumption about the distribution of $-1, 0, 1$ at this point.

*Encoding the residuals.* When we select a predictive model we have to decide on a code for the residuals. At a first glance the residuals may be more difficult to encode than the original sequence since the dynamic range is now expanded to $\{-2, -1, 0, 1, 2\}$. However, there is a reason why the residuals may be easier to encode: a good predictor will cause the sequence of the residuals to contain the symbol 0 many times. With arithmetic coding, or simply with run-length coding, this can be taken advantage of, and we get a smaller code length than that of the original sequence. A less optimal result can be obtained with an even simpler code as described next.

The residuals will be encoded in a way to penalize the nonzero errors. For each error we send the actual (correct) value encoded as "0" = 01, "1" = 10, " $-$ 1" = 11, the codeword 00 signaling the end of the residual sequence. After sending the actual value, we also transmit the location of the error with $\lceil \log_2 N \rceil = \lceil \log_2 30 \rceil = 5$ bits. An error will therefore be transmitted with 7 bits.

#### 4.3.3.2. The hard Boolean predictor: the description length

We consider the two-part code: first we encode the model "parameters" needed to specify the function $f^*(\cdot)$, and then the residuals (prediction errors).

The model cost for encoding the function $f^*(\cdot)$ is

$$C_M(k) = 2^k \text{ bits} \tag{4.11}$$

if we assume a uniform a priori distribution for all the Boolean functions. Other selections are also possible, that is, to favor the more active Boolean functions (e.g., according to the absolute difference between the number of ones and zeros in $f$). Therefore, if there are $N_e$ nonzero prediction errors, the overall code length is the

model length plus the prediction error length:

$$L(k) = 2^k + 7N_e + 2 \text{ bits.} \tag{4.12}$$

Starting from a data set of several possible gene factors, we can select the best combination by designing sequentially the predictors with the window size $k = 2, 3, 4, 5$ and computing $L(k)$ for each predictor. The MDL principle selects $k$ as the optimal window size for which $L(k)$ is minimum.

#### 4.3.3.3. The ternary predictor: the description length

To encode the model we have two possibilities.

(1) The values of the prediction window $\underline{x}$ can be ordered lexically as [0 0 0], [0 0 1], [0 0 2],.... We may specify the optimal ternary predictor by the string of values $h^*([0\ 0\ 0])$, $h^*([0\ 0\ 1])$, $h^*([0\ 0\ 2])$,..., which needs $2 \cdot 3^k$ bits. Therefore the cost of encoding the ternary model is $C_M = 2 \cdot 3^k$ bits, which is the same for all models, irrespective of the data. However, note that we do not know the optimal values $h^*(\underline{x})$ for the prediction windows $\underline{x}$ which have not been seen.

(2) Therefore, in the second alternative the cost of the model depends on the actual data and we have to specify the optimal predictions only for the seen prediction windows. Denoting by $n_x(k)$ the number of different prediction windows found in the data set, the cost of encoding the model becomes $C_M = 2n_x(k)$ bits, which is upper bounded by $2N$, where $N$ is the number of measurements. This variant is used in our experimental determinations. Note that a similar variant can be used also for the Boolean predictor.

The prediction errors will be encoded the same way as in the case of the Boolean prediction. Therefore, if $N_e$ nonzero prediction errors are obtained with the ternary predictor, the code length for the prediction residuals is $7N_e + 2$.

The overall code length is then the model length plus the prediction error length, or

$$L(k) = 2n_x(k) + 7N_e + 2 \text{ bits.} \tag{4.13}$$

Apparently the ternary predictor is defined only for the prediction windows seen in the experiment. There is an obvious way to use the observed data to define $h^*(\underline{x})$ for the unseen prediction windows $\underline{x}$. A simple way is to take $h^*(\underline{x}) = h^*(\underline{x}')$, where $\underline{x}'$ is a seen prediction window of size smaller than $\underline{x}$. However, in the current gene expression problem the goal is to identify outstanding gene dependencies, given the observed data, and it is unnecessary to specify the predictor for the unseen windows.

#### 4.3.3.4. The perceptron predictor: the description length

For ternary-valued data the numbers of all distinct perceptrons with two or three inputs are known, [24]. A perceptron model with two input genes requires $C_M = \log_2 471 = 8.88$ bits, while a perceptron model with three input genes requires $C_M = \log_2 85629 = 16.38$ bits.

### 4.3.4. An artificial example

We take first an artificial example, where, knowing the true state of nature, we will be able to check the success of our procedure.

We assume $N = 30$ measurements (patients), at a patient $i$ 40 variables (genes) $x_{1,i}, \ldots, x_{40,i}$ and one target, denoted by $y(i)$. Suppose the measurements are quantized to three levels: $0, 1, 2$. The number of the sequences of 30 symbols with ternary values is $3^{30} = 2 \cdot 10^{14}$, which exceeds hugely the 40 sequences we have in the experiment. Suppose that there is a function $f^* : \{0,1,2\}^4 \to \{0,1,2\}$, $y(i) = f^*(x_{t_1,i}, x_{t_2,i}, x_{t_3,i}, x_{t_4,i})$, which describes the target exactly.

The data were generated by choosing randomly 40 sequences of 30 measurements from all the possible $2 \cdot 10^{14}$ ternary sequences, with a uniform prior on them. The target function $f^* : \{0,1,2\}^4 \to \{0,1,2\}$ is selected in the following way: for any input window $(j_1, j_2, j_3, j_4)$ the value $f^*(j_1, j_2, j_3, j_4)$ is obtained by sampling a random variable over the values $0, 1, 2$ with the probability distribution $p = [1/3, 1/3, 1/3]$. The target gene is then constructed as the exact function $y(i) = f^*(x_1(i), x_2(i), x_3(i), x_4(i))$. Therefore, the true window for the target is $[x_1(i), x_2(i), x_3(i), x_4(i)]$, referred to as $[1, 2, 3, 4]$ for short. We show in the following that the Boolean and the ternary predictors can be used in conjunction with the MDL principle to select the prediction window candidates.

*The target alone.* The complexity of the target without conditioning on other genes is evaluated by three methods: (a) a prior uniform distribution for all sequences, which gives $L(y^N) = -\lceil \log_2 3^{30} \rceil = 48$ bits; (b) an adaptive arithmetic coding, which gives the value $L(y^N) = 47$ bits; (c) move-to-front coding (which favors correlated sequence), giving the length $L(y^N) = 52$ bits.

Since the target was generated as an uncorrelated sequence of ternary values, the move-to-front procedure gives rather different results from the first two methods, but we can safely assume that the complexity of the target alone is in the range of 47–52 bits.

*The window size* = 2. No predictor with only two genes can achieve a description length lower than the complexity of the target. In the left of Table 4.1 we list the best models of order two, which give a description length less than 72, the total of 17 windows. The "correct" window, which is of order four containing $\text{gene}_1$, $\text{gene}_2$, $\text{gene}_3$, and $\text{gene}_4$, has a "trace" in the list of the best models, which occupies the second position with the window ($\text{gene}_1$, $\text{gene}_3$).

*The window size* = 3. The predictors with three genes are more successful in describing the target. In the middle of Table 4.1 we list the best models of order three, which give a description length less than 55, making the total of 28 windows.

*The window size* = 4. We list the best models of order four (the ones having length less than 49, making the total of 23 windows). We show in bold the "true" model, which was ranked the 19th out of 91390 possible prediction windows. Observe that the variations of the true window have been selected frequently as winners in the final list.

TABLE 4.1. Artificial data. The prediction windows of the best predictors of lengths two, three, and four (the framed one of length four was used to generate the data).

| Descr. length | $g_1$ | $g_2$ |
|---|---|---|
| 60.00 | 15 | 40 |
| 65.00 | **1** | **3** |
| 67.00 | 4 | 13 |
| 67.00 | 9 | 40 |
| 67.00 | 10 | 11 |
| 67.00 | 10 | 34 |
| 67.00 | 14 | 20 |
| 67.00 | 20 | 31 |
| 67.00 | 20 | 32 |
| 67.00 | 20 | 40 |
| 67.00 | 26 | 34 |
| 67.00 | 27 | 40 |
| 67.00 | 38 | 40 |
| 72.00 | 3 | 34 |
| 72.00 | 11 | 34 |
| 72.00 | 27 | 32 |
| 72.00 | 32 | 36 |

| Descr. length | $g_1$ | $g_2$ | $g_3$ |
|---|---|---|---|
| 44.00 | 1 | 3 | 12 |
| 47.00 | 10 | 32 | 36 |
| 48.00 | 16 | 20 | 31 |
| 48.00 | 30 | 32 | 38 |
| 49.00 | 12 | 15 | 40 |
| 49.00 | 14 | 19 | 23 |
| 50.00 | 2 | 10 | 32 |
| 50.00 | 10 | 31 | 34 |
| 50.00 | 15 | 20 | 32 |
| 50.00 | 22 | 33 | 34 |
| 52.00 | 6 | 20 | 25 |
| 52.00 | 8 | 10 | 12 |
| 52.00 | 8 | 20 | 32 |
| 52.00 | 13 | 18 | 19 |
| 53.00 | **1** | **2** | **3** |
| 53.00 | 10 | 11 | 34 |
| 53.00 | 10 | 15 | 40 |

| Descr. length | $g_1$ | $g_2$ | $g_3$ | $g_4$ |
|---|---|---|---|---|
| 46.00 | 1 | 2 | 3 | 12 |
| 46.00 | 1 | 3 | 12 | 38 |
| 46.00 | 12 | 30 | 32 | 38 |
| 46.00 | 30 | 32 | 34 | 38 |
| 46.00 | 8 | 10 | 31 | 34 |
| 46.00 | 1 | 12 | 32 | 38 |
| 46.00 | 12 | 30 | 32 | 38 |
| 48.00 | 1 | 3 | 12 | 18 |
| 48.00 | 10 | 27 | 32 | 36 |
| 48.00 | 4 | 16 | 20 | 31 |
| 48.00 | 2 | 10 | 25 | 32 |
| 48.00 | 2 | 10 | 29 | 32 |
| 48.00 | 10 | 25 | 31 | 34 |
| 48.00 | 15 | 20 | 25 | 32 |
| 48.00 | 15 | 20 | 32 | 40 |
| 48.00 | 6 | 20 | 25 | 40 |
| 48.00 | 8 | 20 | 32 | 34 |
| 48.00 | **1** | **2** | **3** | **4** |
| 48.00 | 10 | 11 | 13 | 34 |
| 48.00 | 15 | 20 | 32 | 40 |
| 49.00 | 1 | 3 | 12 | 34 |
| 49.00 | 12 | 16 | 20 | 31 |

### 4.3.5. A comparison of the three classes of models in a small scale experiment

We consider here a comparison of the three ternary predictors by the MDL criterion on the experimental data from [8] having $N = 30$ and $p = 13$, presented in Table 4.7. (Note that the table is transposed according to the convention used in text for $x_{ij}$.) The target gene is *AHA*.

In Table 4.2 we present the values of the description length and also the number of the errors, obtained with the best predictors in each of the three classes for various sizes of the prediction window, when the genes are restricted only to the set (*RCH1, p53, PC-1*). With the Boolean predictor the best description length is obtained with the model *HB*(*RCH1, p53*). Appending *PC-1* to the prediction window does not improve the descriptive power of the model. The same conclusion can be drawn for the hard ternary predictors, and the perceptrons. The MDL criterion makes therefore a clear and consistent selection of the best prediction

Table 4.2. "Three gene" experiment. The description length for three predictors: hard Boolean predictor (HBP), hard ternary predictor (HT), and the perceptron predictor (Per).

| Hard Boolean predictor | | | |
|---|---|---|---|
| | AHA = HBP *(RCH1, p53, PC-1)* | AHA = HBP *(RCH1, p53)* | AHA = HBP *(p53)* |
| $N_e$ | 2 | 2 | 12 |
| *Length* | 24 | 20 | 88 |
| **Hard ternary predictor** | | | |
| | AHA = HT *(RCH1, p53, PC-1)* | AHA = HT *(RCH1, p53)* | AHA = HT*(p53)* |
| $N_e$ | 1 | 2 | 7 |
| *Length* | 25 | 24 | 55 |
| **Perceptron predictor** | | | |
| | AHA = Per *(RCH1, p53, PC-1)* | AHA = Per *(RCH1, p53)* | AHA = Per*(p53)* |
| $N_e$ | 2 | 2 | 7 |
| *Length* | 30.38 | 22.87 | 53.24 |
| $c_d$ (MSE) [8] | 0.946 | 0.785 | 0.624 |

Table 4.3. "Twelve gene" experiment. The description length of the best predictors with window size 2.

| Length | 20 | 38 | 45 |
|---|---|---|---|
| $\text{Gene}_1$ | *p53* | *p53* | *p53* |
| $\text{Gene}_2$ | *RCH1* | *BCL3* | *ATF3* |
| Method | HB | HT | HT |

window, while the coefficient of the determination $c_d$ [8] simply shows a better fit of the model to the data with an increasing window size. The problem with $c_d$ is that it expresses only the fit to the training data. To avoid overtraining one needs another (validation) data set. The MDL principle is a method which implicitly penalizes too large models and which can be shown to provide consistent order estimations unlike the several cross-validation methods.

We extend now the experiment to predicting the values of the target *AHA* based on the combinations of two, three, or four of the genes in the first 12 columns of Table 4.7. In Tables 4.3, 4.4, 4.5 we show the best predictors for each window size. The best overall predictor is seen to be AHA = *HB*(*p53, ATF3, RCH1*), which has the Boolean function $f(x_1, x_2, x_3) = x_1x_2 + x_2x_3$, and the description length $L = 10$. This is much lower than what can be obtained by any other window size or a combination of genes. By comparing the description length of the predictors of different sizes, we can now conclude that most of the "good" predictors of order 3 or 4 are worse than the corresponding predictors of lower order; that is, all predictors of size three with the description length 24 are actually worse than the predictor of order two AHA = *HB*(*p53, RCH1*), which has the description length 20.

Table 4.4. "Twelve gene" experiment. The description length of the best predictors with window size 3.

| Length | 10 | 24 | 24 | 24 | 24 | 24 | 24 | 24 | 24 | 24 | 30 | 39 | 42 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $Gene_1$ | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* |
| $Gene_2$ | *ATF3* | *BCL3* | *FRA1* | *RELB* | *IAP1* | *PC-1* | *MBP1* | *SSAT* | *MDM2* | *p21* | *ATF3* | *ATF3* | *RELB* |
| $Gene_3$ | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *BCL3* | *RELB* | *BCL3* |
| Method | HB | HB | HB | HB | HB | HB | HB | HB | HB | HB | HT | HT | HT |

Table 4.5. "Twelve gene" experiment. The description length of the best predictors with window size 4.

| Length | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 18 | 27 | 27 | 29 | 29 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $Gene_1$ | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* |
| $Gene_2$ | *ATF3* | *ATF3* | *ATF3* | *IAP1* | *PC-1* | *MBP1* | *SSAT* | *MDM2* | *p21* | *PC-1* | *MBP1* | *PC-1* | *MBP1* |
| $Gene_3$ | *BCL3* | *FRA1* | *RELB* | *ATF3* | *ATF3* | *ATF3* | *ATF3* | *ATF3* | *ATF3* | *BCL3* | *PC-1* | *IAP1* | *FRA1* |
| $Gene_4$ | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* |
| Method | HB | HB | HB | HB | HB | HB | HB | HB | HB | HT | HT | HT | HT |

| Length | 29 | 29 | 29 | 30 | 30 | 30 | 30 | 31 | 31 | 31 | 32 | 32 | 32 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $Gene_1$ | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* | *p53* |
| $Gene_2$ | *SSAT* | *SSAT* | *MDM2* | *MBP1* | *SSAT* | *MDM2* | *p21* | *FRA1* | *PC-1* | *p21* | *RELB* | *RELB* | *IAP1* |
| $Gene_3$ | *FRA1* | *PC-1* | *PC-1* | *IAP1* | *MBP1* | *MBP1* | *MBP1* | *BCL3* | *RELB* | *PC-1* | *BCL3* | *FRA1* | *BCL3* |
| $Gene_4$ | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* | *RCH1* |
| Method | HT | HT | HT | HT | HT | HT | HT | HT | HT | HT | HB | HB | HB |

We are therefore capable of organizing the huge number of "good predictors" in Tables 4.3, 4.4, 4.5 and discarding all the overly complex ones to conclude that the relevant predictors in light of the data are only $HB(p53, RCH1)$, $HB(p53, ATF3, RCH1)$, and $HT(p53, ATF3, BCL3)$.

## 4.4. Normalized maximum likelihood models for a class of Boolean regressor models

The selection of the model order based on the two-part code discussed in the previous sections is widespread for a number of reasons. First, it is easily interpretable and intuitive in that it separates nicely the costs of the model and the remaining nonmodeled part of the string (erroneous according to the model). Secondly, one of the asymptotic forms of the two-part code has been shown in many applications to be quite successful, and it was further shown to be equivalent with the Bayesian information criterion. However, in the recent years further advances in the MDL principle have been made, and a specific code, induced by the so-called normalized maximum likelihood (NML) model, was shown to possess important optimality properties. We illustrate in the rest of the chapter the derivation of the NML model and its application to a class of nonlinear models, first introduced by us in [19]. We follow it closely in this presentation.

Table 4.6. The best 18 triplets of genes for predicting the class label according to the NML model for the class $\mathcal{M}(\theta, 3, f)$.

| Code length | Classification error (%) | Triplet of genes | | | Gene accession numbers | | |
|---|---|---|---|---|---|---|---|
| 6.9 | 0.912 | 1834 | 2288 | 5714 | M23197 | M84526 | HG1496-HT1496 |
| **7.9** | **0.010** | **1834** | **3631** | **6277** | **M23197** | **U70063** | **M30703** |
| 7.9 | 0.891 | 758 | 4250 | 4342 | D88270 | X53586 | X59871 |
| 8.0 | 0.652 | 2288 | 4847 | 6376 | M84526 | X95735 | M83652 |
| **8.7** | **0.008** | **1834** | **3631** | **5373** | **M23197** | **U70063** | **S76638** |
| **8.7** | **0.007** | **1834** | **3631** | **6279** | **M23197** | **U70063** | **X97748** |
| 8.7 | 0.910 | 1144 | 1217 | 1882 | J05243 | L06132 | M27891 |
| 8.8 | 0.649 | 302 | 2288 | 6376 | D25328 | M84526 | M83652 |
| 8.8 | 0.055 | 1144 | 1834 | 1882 | J05243 | M23197 | M27891 |
| 8.8 | 0.063 | 1834 | 1882 | 6049 | M23197 | M27891 | U89922 |
| **8.8** | **0.004** | **1144** | **1882** | **5808** | **J05243** | **M27891** | **HG2981-HT3127** |
| 8.8 | 0.584 | 2288 | 3932 | 6376 | M84526 | U90549 | M83652 |
| 8.9 | 0.558 | 2288 | 5518 | 6376 | M84526 | X95808 | M83652 |
| 8.9 | 0.560 | 1399 | 2288 | 6376 | L21936 | M84526 | M83652 |
| 8.9 | 0.620 | 1241 | 2288 | 6376 | L07758 | M84526 | M83652 |
| 8.9 | 0.605 | 2288 | 3660 | 6376 | M84526 | U72342 | M83652 |
| 8.9 | 0.582 | 2288 | 4399 | 6376 | M84526 | X63753 | M83652 |
| 8.9 | 0.556 | 2288 | 4424 | 6376 | M84526 | X65867 | M83652 |

The NML model for the linear regression problem was introduced and analyzed recently [15]. We restate classification as a modeling problem in terms of a class of parametric models, for which the maximum likelihood parameter estimates can be easily computed. We review first the NML model for Bernoulli strings as the solution to a minmax optimization problem. We then introduce a model class for the case where the binary strings to be modeled are observed jointly with several other binary strings (regression variables). We derive the NML model for this model class. We further provide a fast evaluation procedure and apply it to a classification problem.

### 4.4.1. The NML model for Bernoulli strings

In this section we assume that a Bernoulli variable $Y$ with $P(Y = 0) = \theta$ is observed repeatedly $n$ times, generating the string $y^n = y_1, \ldots, y_n$. We look for a distribution $q(y^n)$ over all strings of length $n$ such that the ideal code length $\log(1/q(y^n))$ assigned to a particular string $y^n$ by this distribution is as close as possible to the ideal code length $\log(1/P(y^n \mid \widehat{\theta}(y^n)))$, obtainable with the Bernoulli models. The words "ideal code length" are used because they need not be integer valued as real code lengths must be. For long strings they differ from the real code lengths by at most unity. In the coding scenario the decoder is allowed to use a predefined distribution, $q(\cdot)$, but it cannot use $P(y^n \mid \widehat{\theta}(y^n))$ because it is not a

Table 4.7. A small-scale experiment where gene expressions are quantized to ternary values. The columns represent different genes, while the rows represent the various conditions (patients).

| *RCH1* | *BCL3* | *FRA1* | *REL-B* | *ATF3* | *IAP-1* | *PC-1* | *MBP-1* | *SSAT* | *MDM2* | *p21* | *p53* | *AHA* |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| −1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| −1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| −1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| −1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| −1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | −1 | −1 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | −1 | 0 |
| −1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | −1 | −1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | −1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | −1 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | −1 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | −1 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | −1 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | −1 | 0 |
| −1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | −1 | −1 |
| −1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | −1 | −1 |
| −1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | −1 | −1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | −1 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | −1 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | −1 | 0 |

distribution. The latter, however, will be a target which no model in the class can beat since it maximizes $P(y^n \mid \theta)$ and therefore minimizes the ideal code length $\log(1/P(y^n \mid \theta))$. The distribution $q(y^n)$ is selected such that the "regret" of using $q(y^n)$ instead of $P(y^n \mid \hat{\theta}(y^n))$, namely,

$$\log \frac{1}{q(y^n)} - \log \frac{1}{P(y^n \mid \hat{\theta}(y^n))} = \log \frac{P(y^n \mid \hat{\theta}(y^n))}{q(y^n)}, \tag{4.14}$$

is minimized for the worst case $y^n$, that is,

$$\min_q \max_{y^n} \log \frac{P(y^n \mid \hat{\theta}(y^n))}{q(y^n)}. \tag{4.15}$$

Theorem 4.1 (Shtar'kov [17]). *The minimizing distribution is given by*

$$q(y^n) = \frac{P(y^n \mid \hat{\theta}(y^n))}{C_n}, \tag{4.16}$$

*where*

$$C_n = \sum_{m=0}^{n} \binom{n}{m} \left(\frac{m}{n}\right)^m \left(1 - \frac{m}{n}\right)^{n-m}. \tag{4.17}$$

A strong optimality property of the NML models was recently proven in [16], where the following minmax problem was formulated: find the (universal) distribution which minimizes the average regret

$$\min_q \max_g E_g \log \frac{P(Y^n \mid \hat{\theta}(Y^n))}{q(Y^n)}, \tag{4.18}$$

where $g(\cdot)$, the generating distribution of the data, and $q(\cdot)$ run through any sets that include the NML model.

Theorem 4.2 (see [16]). *The minimizing distribution $q(\cdot)$ in the minmax problem* (4.18) *is given by* (4.16) *and* (4.17).

If we replace "minmax" by "maxmin," the worst case distribution is unique and also given by (4.16) and (4.17).

### 4.4.2. The NML model for a Boolean class

We consider a binary random variable $Y$, which is observed jointly with a binary regressor vector $\underline{X} \in \mathcal{B}^k$. In a useful model class a carefully selected Boolean function $f : \mathcal{B}^k \to \{0, 1\}$ should provide a reasonable prediction $f(\underline{X})$ of $Y$ in the sense that the absolute error $\mathcal{E} = |Y - f(\underline{X})|$ has a large probability of being 0. Since $\mathcal{E}, Y, f(\underline{X})$ are binary valued, we have $\mathcal{E} = |Y - f(\underline{X})| = Y \oplus f(\underline{X})$, which also implies $Y = f(\underline{X}) \oplus \mathcal{E}$, where $\oplus$ is modulo 2 sum.

We therefore consider a corruption model defined as follows:

$$Y = f(\underline{X}) \oplus \mathcal{E} = \begin{cases} f(\underline{X}) & \text{if } \mathcal{E} = 0, \\ \overline{f(\underline{X})} & \text{if } \mathcal{E} = 1, \end{cases} \tag{4.19}$$

where $f(\cdot)$ is a Boolean function and the error $\mathcal{E}$ is independently drawn from a Bernoulli source with parameter $\theta$; that is, $P(\mathcal{E}=1)=1-\theta$ and $P(\mathcal{E}=0)=\theta$, or for short

$$P(\mathcal{E}=b)=\theta^{1-b}(1-\theta)^{b}, \quad \text{for } b\in\{0,1\}. \tag{4.20}$$

Denote by $\underline{b}_i\in\{0,1\}^k$ the vector having as entries the bits in the binary representation of integer $i$, that is, $\underline{b}_0=[0,\ldots,0,0]$, $\underline{b}_1=[0,\ldots,0,1]$, and so forth. Further, define by (4.19) and (4.20) the conditional probability for code $\underline{b}_i\in\{0,1\}^k$,

$$P(Y=y\mid \underline{X}=\underline{b}_i)=\theta^{1-y\oplus f(\underline{b}_i)}(1-\theta)^{y\oplus f(\underline{b}_i)}. \tag{4.21}$$

The Boolean regression problem will be stated as finding the optimal universal model (in a minmax sense to be specified shortly) for the following class of models:

$$\mathcal{M}(\theta,k,f)=\left\{P(y\mid f,\underline{b}_i,\theta)=\theta^{(1-y\oplus f(\underline{b}_i))}(1-\theta)^{(y\oplus f(\underline{b}_i))}\right\}, \tag{4.22}$$

where $y\in\{0,1\},\theta\in[0,1],\underline{b}_i\in\{0,1\}^k$.

When the sequence $y^n=y_1,\ldots,y_n$ and the sequence of binary regressor vectors $\underline{b}^n=\underline{b}_{i_1},\ldots,\underline{b}_{i_n}$ are observed, a member of the class $\mathcal{M}(\theta,k,f)$ assigns to the sequence $y^n$ the following probability:

$$\begin{aligned} P(y^n\mid\theta,k,f,\underline{b}^n)&=\prod_{j=1}^{n}\theta^{(1-y_j\oplus f(\underline{b}_{i_j}))}(1-\theta)^{(y_j\oplus f(\underline{b}_{i_j}))}\\ &=\theta^{n_0}(1-\theta)^{n-n_0}, \end{aligned} \tag{4.23}$$

where $n_0$ is the number of zeros in the sequence $\{\varepsilon_j=y_j\oplus f(\underline{b}_{i_j})\}_{j=1}^{n}$. The ML estimate of the model parameters,

$$(\hat{\theta}(y^n),\hat{f}_{y^n})=\arg\max_{\theta,f}P(y^n\mid\theta,k,f,\underline{b}^n), \tag{4.24}$$

can be obtained in two stages, first by maximizing with respect to $f$,

$$\max_{f}P(y^n\mid\theta,k,f,\underline{b}^n), \tag{4.25}$$

and then by observing that the optimal $f(\cdot)$ does not depend on $\theta$. For a fixed $\theta>0.5$, the function $P(y^n\mid\theta,k,f,\underline{b}^n)=\theta^{n_0}(1-\theta)^{n-n_0}$ decreases monotonically with $n_0$, and (4.25) is maximized by maximizing $n_0$, or, equivalently, by minimizing $n-n_0$:

$$\begin{aligned} \min_{f}(n-n_0)&=\min_{f}\sum_{j=1}^{n}\mid y_j-f(\underline{b}_{i_j})\mid\\ &=\min_{f}\sum_{\ell=0}^{2^n}m_{\ell_0}f(\underline{b}_\ell)+m_{\ell_1}(1-f(\underline{b}_\ell)), \end{aligned} \tag{4.26}$$

where $m_{\ell_0}$ and $m_{\ell_1}$ denote the number of times $y_j = 0$ and $y_j = 1$, respectively, have been seen at the regressor vector $\underline{b}_{i_j} = \underline{b}_\ell$.

Equation (4.26) shows that $f$ is optimal for the mean absolute error (MAE) criterion. It can also be seen that the assignment of $f(\underline{b}_\ell)$ depends only on $m_{\ell_0}, m_{\ell_1}$, and the solution is

$$\hat{f}_{y^n}(\underline{b}_\ell) = \begin{cases} 0 & \text{if } m_{\ell_0} \geq m_{\ell_1}, \\ 1 & \text{if } m_{\ell_0} < m_{\ell_1}, \end{cases} \tag{4.27}$$

which can be readily computed from the data set. Denote by $n_0^*(y^n)$ the number of zeros in the sequence $\{\varepsilon_j = y_j \oplus \hat{f}_{y^n}(\underline{b}_{i_j})\}_{j=1}^n$. To completely solve the ML estimation problem we have to find

$$\max_\theta P(y^n \mid \theta, k, \hat{f}_{y^n}, \underline{b}^n), \tag{4.28}$$

for which the maximizing parameter is $\hat{\theta}(y^n) = n_0^*(y^n)/n$. Therefore

$$P(y^n \mid \hat{\theta}(y^n), k, \hat{f}_{y^n}, \underline{b}^n) = \left(\frac{n_0^*(y^n)}{n}\right)^{n_0^*(y^n)} \left(1 - \frac{n_0^*(y^n)}{n}\right)^{n - n_0^*(y^n)}. \tag{4.29}$$

We need to define a distribution $q(y^n)$ over all possible sequences $y^n$ which is the best in the minmax sense

$$\min_q \max_{y^n} \frac{P\left(y^n \mid \hat{\theta}(y^n), k, \hat{f}_{y^n}, \underline{b}^n\right)}{q(y^n)}. \tag{4.30}$$

This is clearly given by the NML model,

$$q(y^n) = \frac{P\left(y^n \mid \hat{\theta}(y^n), k, \hat{f}_{y^n}, \underline{b}^n\right)}{C_n(k, \underline{b}^n)}, \tag{4.31}$$

where

$$C_n(k, \underline{b}^n) = \sum_{y^n} \left(\frac{n_0^*(y^n)}{n}\right)^{n_0^*(y^n)} \left(1 - \frac{n_0^*(y^n)}{n}\right)^{n - n_0^*(y^n)}. \tag{4.32}$$

Note that $n_0^*$ depends on $y^n$ through $\hat{f}_{y^n}$ in a complicated manner. When $k = 0$, the normalization factor is $C_n(0, \underline{b}^n) = C_n$, given in (4.17).

Alternative expressions for the coefficient $C_n(k, \underline{b}^n)$ provide faster evaluation. Let $\{\underline{b}_{j_1}, \ldots, \underline{b}_{j_K}\}$ be the set of the distinct elements in the set $\{\underline{b}_\ell \mid \underline{b}_\ell \in \underline{b}^n\}$, and let $K = K(\underline{b}^n)$ be the number of the distinct regressor vectors. Denote by $z^q$ the subsequence of $y^n$ observed when the regressor vector is $\underline{b}_{j_q}$. Let $n_q$ be the length of the subsequence $z^q$ having $m_q$ zeros.

We observe that (4.32) can be alternatively expressed as

$$C_n(k, \underline{b}^n) = \sum_{n_1^*=0}^{n} \left(\frac{n_1^*}{n}\right)^{n_1^*} \left(1 - \frac{n_1^*}{n}\right)^{n-n_1^*} S_{K,n_1,\ldots,n_K}(n_1^*), \tag{4.33}$$

where $S_{K,n_1,\ldots,n_K}(n_1^*)$ is the number of sequences $y^n$ having $n_1^* = \sum_{q=1}^{K} \min(m_q, n_q - m_q)$ ones in the residual sequence. The numbers $S_{K,n_1,\ldots,n_K}(n_0^*)$ can be easily computed recursively in $K$. Denote first

$$h_\ell(m) = \begin{cases} 0 & \text{if } m > \dfrac{n_\ell}{2}, \\ \dbinom{n_\ell}{m} & \text{if } m = \dfrac{n_\ell}{2}, \\ 2\dbinom{n_\ell}{m} & \text{else,} \end{cases} \tag{4.34}$$

which is the number of sequences of length $n_\ell$ having either $m$ bits set to 1 or $n_\ell - m$ bits set to 1, for $0 \le m \le n_\ell/2$. By combining each of the $S_{K-1,n_1,\ldots,n_{K-1}}(n_1^* - m_K)$ sequences having $n_1^* - m_K$ ones in the residual sequence with each of the $h_K(m_K)$ sequences having either $m_K$ bits set to 1 or $n_K - m_K$ bits set to 1, we get sequences having $(n_1^* - m_K) + \min(m_K, n_K - m_K) = n_1^*$ occurrences of 1 in their residual sequence. Therefore the following recurrence relation holds:

$$S_{K,n_1,\ldots,n_K}(n_1^*) = \sum_{m_K=0}^{n_K} h_K(m_K) S_{K-1,n_1,\ldots,n_{K-1}}(n_1^* - m_K), \tag{4.35}$$

where by convention $S_{K-1,n_1,\ldots,n_{K-1}}(n_1^* - m_K) = 0$ for negative arguments, $n_1^* - m_K < 0$.

We note that the recurrence is simply a convolution sum, $S_{K,n_1,\ldots,n_K} = h_K \otimes S_{K-1,n_1,\ldots,n_{K-1}}$, and we conclude that

$$S_{K,n_1,\ldots,n_K} = h_1 \otimes h_2 \otimes \cdots \otimes h_K. \tag{4.36}$$

We can easily see that $S_{K_1,n_1,\ldots,n_{K_1}}(i) = 0$ for $i > (\sum_q^{K_1} n_q)/2$ due to the fact that the optimal residual sequence cannot have more than $(\sum_q^{K_1} n_q)/2$ ones. Also, from (4.34) we note that only $(1/2^K) \prod_{q=1}^{K} n_q$ terms have to be added when all the convolution sums in (4.36) are evaluated.

### 4.4.3. An experiment from cancer genomics

We illustrate the classification procedure based on the NML model for classes of Boolean regression models for the microarray DNA data Leukemia (ALL/AML) of [4], publicly available at http://www.broad.mit.edu/cgi-bin/cancer/datasets.cgi.

The microarray contains 6817 human genes, sampled from 72 cases of cancer, of which 47 are of ALL type and 25 of AML type. The data is preprocessed as recommended in [2, 4]. The resulting data matrix $\widetilde{X}$ has 3571 rows and 72 columns.

We design a two-level quantizer by the LBG algorithm [9], and the decision threshold results at 2.6455. All the entries in the matrix $\widetilde{X}$ are used as a training set but we note that no information about the true classes is used during the quantization stage. The entries in the matrix $\widetilde{X}$ are quantized to binary values, which results in the binary matrix $X$.

#### 4.4.3.1. Extending the classification for unseen cases of the Boolean regressors

The Boolean regressors observed in the training set may not span all the $2^k$ possible binary vectors. If the binary vector $\underline{b}_q$ is not observed in the training set, the classification decision $f^*(\underline{b}_q)$ remains undecided during the training stage. We select the value of $f^*(\underline{b}_q)$ by use of the nearest-neighbor voting, which amounts to the decision by the majority vote of the neighbors $\underline{b}_\ell$ situated at Hamming distance 1, for which $f^*(\underline{b}_\ell)$ was decided during the training stage. If after voting there still is a tie, we take the majority vote of the neighbors at Hamming distance 2, and continue if necessary until a clear decision is reached.

#### 4.4.3.2. Estimation of classification errors achieved with Boolean regression models with $k = 3$

The Leukemia data set was considered recently in a study for comparing several classification methods [2]. The evaluation of the performance there is based on the classification error as estimated in a cross-validation 2 : 1 experiment. In order to compare our classification results with the results in [2], we estimate the classification error in the same way, namely, by dividing at random the 72 patient set into a training set of $n_T = 48$ patients and a test set of $n_s = 24$ patients, finding the optimal predictor $f^*(\cdot)$ over the training set, classifying the test set by use of the predictor $f^*(\cdot)$ (the extension for the unseen cases in the training set is done as in Section 4.4.3.1), and counting the number of classification errors produced in the test set. The random split is repeated $n_r = 10000$ times, and the estimated classification error is computed as the percentage of the total number of errors observed in the $(n_r \cdot n_s)$ test classifications. For comparison, we mention that the best classification methods tested in [2] have classification errors higher than 1%. As we can observe in Table 4.6, there are several predictors with three genes, achieving classification rates as low as 0.004%. We note a remarkable consensus in ranking of the gene triplets, according to the NML code length and the estimated classification error rates.

As to the genes involved in the optimal predictors in Table 4.6, we note that five genes belong to the set of 50 "informative" genes selected in [4], namely, $M23197$, $M84526$, $M27891$, $M83652$, $X95735$.

### 4.5. Summary

The Boolean regression classes of models are powerful modeling tools with the associated NML models, which can be easily computed and used in MDL inference, in particular for factor selection.

The MDL principle for classification with the class of Boolean models provides an effective classification method, as seen in the important application of cancer classification based on gene expression data. The NML model for the class $\mathcal{M}(\theta, k, f)$ was used for the selection of informative feature genes. With the sets of feature genes selected by the NML model, we achieved classification error rates significantly lower than those reported recently for the same data set.

## Bibliography

[1] A. Barron, J. Rissanen, and B. Yu, "The minimum description length principle in coding and modeling," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2743–2760, 1998.

[2] S. Dudoit, J. Fridlyand, and T. P. Speed, "Comparison of discrimination methods for the classification of tumors using gene expression data," Tech. Rep. 576, Deptartment of Statistics, University of California, Berkeley, Calif, USA, 2000.

[3] C. D. Giurcăneanu, I. Tăbuş, and J. Astola, "Clustering time series gene expression data based on sum-of-exponentials fitting," *EURASIP Journal on Applied Signal Processing*, vol. 2005, no. 8, pp. 1159–1173, 2005.

[4] T. R. Golub, D. K. Slonim, P. Tamayo, et al., "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, no. 5439, pp. 531–537, 1999.

[5] F. Jacob and J. Monod, "Genetic regulatory mechanisms in the synthesis of proteins," *Journal of Molecular Biology*, vol. 3, pp. 318–356, June 1961.

[6] S. A. Kauffman, *The Origins of Order: Self-Organization and Selection in Evolution*, Oxford University Press, New York, NY, USA, 1993.

[7] E. R. Dougherty, S. Kim, and Y. Chen, "Coefficient of determination in nonlinear signal processing," *Signal Processing*, vol. 80, no. 10, pp. 2219–2235, 2000.

[8] S. Kim, E. R. Dougherty, Y. Chen, et al., "Multivariate measurement of gene expression relationships," *Genomics*, vol. 67, no. 2, pp. 201–209, 2000.

[9] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communications*, vol. 28, no. 1, pp. 84–95, 1980.

[10] C. Mircean, I. Shmulevich, D. Cogdell, et al., "Robust estimation of protein expression ratios with lysate microarray technology," *Bioinformatics*, vol. 21, no. 9, pp. 1935–1942, 2005, Advance Access originally published online on January 12, 2005.

[11] C. Mircean, I. Tăbuş, T. Kobayashi, et al., "Pathway analysis of informative genes from microarray data reveals that metabolism and signal transduction genes distinguish different subtypes of lymphomas," *International Journal of Oncology*, vol. 24, no. 3, pp. 497–504, 2004.

[12] J. Rissanen, "Modelling by shortest data description," *Automatica*, vol. 14, no. 5, pp. 465–471, 1978.

[13] J. Rissanen, "Universal coding, information, prediction, and estimation," *IEEE Transactions on Information Theory*, vol. 30, no. 4, pp. 629–636, 1984.

[14] J. Rissanen, "Stochastic complexity and modeling," *The Annals of Statistics*, vol. 14, no. 3, pp. 1080–1100, 1986.

[15] J. Rissanen, "MDL denoising," *IEEE Transactions on Information Theory*, vol. 46, no. 7, pp. 2537–2543, 2000.

[16] J. Rissanen, "Strong optimality of the normalized ML models as universal codes and information in data," *IEEE Transactions on Information Theory*, vol. 47, no. 5, pp. 1712–1717, 2001.

[17] Y. M. Shtar'kov, "Universal sequential coding of individual messages," *Problemy Peredachi Informatsii*, vol. 23, no. 3, pp. 3–17, 1987, translation in *Problems of Information Transmission*.
[18] I. Tăbuş and J. Astola, "On the use of MDL principle in gene expression prediction," *EURASIP Journal on Applied Signal Processing*, vol. 2001, no. 4, pp. 297–303, 2001.
[19] I. Tăbuş, J. Rissanen, and J. Astola, "A classifier based on normalized maximum likelihood model for classes of Boolean regression models," in *Proceedings of 11th European Signal Processing Conference (EUSIPCO '02)*, vol. 1, pp. 119–122, Toulouse, France, September 2002.
[20] I. Tăbuş, J. Rissanen, and J. Astola, "Classification and feature gene selection using the normalized maximum likelihood model for discrete regression," *Signal Processing*, vol. 83, no. 4, pp. 713–727, 2003, Special issue on Genomic Signal Processing.
[21] I. Tăbuş, C. Mircean, W. Zhang, I. Shmulevich, and J. Astola, "Transcriptome-based glioma classification using informative gene set," in *Genomic and Molecular Neuro-Oncology*, W. Zhang and G. Fuller, Eds., pp. 205–220, Jones and Bartlett, Sudbury, Mass, USA, 2003.
[22] I. Tăbuş and J. Astola, "Clustering the non-uniformly sampled time series of gene expression data," in *Proceedings of 7th International Symposium on Signal Processing and its Applications (ISSPA '03)*, vol. 2, pp. 61–64, Paris, France, July 2003.
[23] I. Tăbuş, C. D. Giurcăneanu, and J. Astola, "Genetic networks inferred from time series of gene expression data," in *Proceedings of 1st International Symposium on Control, Communications and Signal Processing (ISCCSP '04)*, pp. 755–758, Hammamet, Tunisia, March 2004.
[24] Y. Yamamoto and M. Mukaidono, "Meaningful special classes of ternary logic functions—Regular ternary logic functions and ternary majority functions," *IEEE Transactions on Computers*, vol. 37, no. 7, pp. 799–806, 1988.

Ioan Tabus: Institute of Signal Processing, Tampere University of Technology,
P.O. Box 553, 33101 Tampere, Finland.
*Email*: ioan.tabus@tut.fi

Jorma Rissanen: Institute of Signal Processing, Tampere University of Technology,
P.O. Box 553, 33101 Tampere, Finland.
*Email*: jrrissanen@yahoo.com

Jaakko Astola: Institute of Signal Processing, Tampere University of Technology,
P.O. Box 553, 33101 Tampere, Finland.
*Email*: jaakko.astola@tut.fi

# 5 Nonlinear methods for speech analysis and synthesis

Steve McLaughlin and Petros Maragos

## 5.1. Introduction

Perhaps the first question to ask on reading this chapter is why should we consider nonlinear methods as offering any insight into speech signals given the success of current, mostly linear-based, speech analysis methods. There are known to be a number of nonlinear effects in the speech production process. Firstly, it has been accepted for some time that the vocal tract and the vocal folds do not function independently of each other, but that there is in fact some form of coupling between them when the glottis is open [27] resulting in significant changes in formant characteristics between open and closed glottis cycles [7]. More controversially, Teager and Teager [69] have claimed (based on physical measurements) that voiced sounds are characterized by highly complex air flows in the vocal tract involving jets and vortices, rather than well-behaved laminar flow. In addition, the vocal folds will themselves be responsible for further nonlinear behavior, since the muscle and cartilage which comprise the larynx have nonlinear stretching qualities. Such nonlinearities are routinely included in attempts to model the physical process of vocal fold vibration, which have focused on two or more mass models [24, 28, 65], in which the movement of the vocal folds is modeled by masses connected by springs, with nonlinear coupling. Observations of the glottal waveform have shown that this waveform can change shape at different amplitudes [59] which would not be possible in a strictly linear system where the waveform shape is unaffected by amplitude changes. Models of the glottal pulse also include nonlinearities, for example, the use of nonlinear shaping functions [58–60] or the inclusion of nonlinear flow [22].

In order to arrive at the simplified linear model, a number of major assumptions are made:

(i) the vocal tract and speech source are uncoupled (thus allowing source-filter separation);
(ii) airflow through the vocal tract is laminar;

(iii) the vocal folds vibrate in an exactly periodic manner during voiced speech production;
(iv) the configuration of the vocal tract will only change slowly.

These imply a loss of information which means that the full speech signal dynamics can never be properly captured. These inadequacies can be seen in practice in speech synthesis where, at the waveform generation level, current systems tend to produce an output signal that lacks naturalness. This is true even for concatenation techniques which copy and modify actual speech segments.

Given the statements above then should make clear why nonlinear methods will offer useful insights and suggest useful methods that we can adopt to enhance speech synthesis. The chapter is structured as follows. First some discussion on speech aerodynamics and modulations in speech are discussed. Then the discussion moves on to consider why conventional linear methods work as well as they do. The discussion then moves on to consider what nonlinear methods we might use and for what. These range from modulation models, fractal methods, using Poincaré maps for epoch detection, the use of unstable manifolds, and functional approximation methods. Then consideration is briefly given to the use of nonlinear methods in automatic speech recognition. Finally some conclusions are drawn and suggestions for potential areas of research are considered.

### 5.1.1. Speech aerodynamics

For several decades the traditional approach to speech modeling has been the linear (source-filter) model where the true nonlinear physics of speech production is approximated via the standard assumptions of linear acoustics and 1D plane wave propagation of the sound in the vocal tract. This approximation leads to the well-known linear prediction model for the vocal tract where the speech formant resonances are identified with the poles of the vocal tract transfer function. The linear model has been applied to speech coding, synthesis, and recognition with limited success [55, 56]. To build successful applications, deviations from the linear model are often modeled as second-order effects or error terms. However, there is strong theoretical and experimental evidence [2, 25, 36, 40, 62, 69, 70] for the existence of important nonlinear aerodynamic phenomena during the speech production that cannot be accounted for by the linear model. Thus, the linear model can be viewed only as a first-order approximation to the true speech acoustics which also contain second-order and nonlinear structure. The investigation of speech nonlinearities can proceed in at least two directions: (i) numerical simulations of the nonlinear differential (Navier-Stokes) equations [72] governing the 3D dynamics of the speech airflow in the vocal tract, as, for example, in [57, 70], and (ii) development of nonlinear signal processing systems suitable to detect such phenomena and extract related information. Most of the research in this aspect of nonlinear speech processing, for example, as reviewed in [35, 54], has focused on the second approach, which is computationally much simpler, that is, to develop models and extract related acoustic signal features describing two types of nonlinear phenomena in speech, *modulations* and *turbulence*. Turbulence can be explored both

from the geometric aspect, which brings us to *fractals* [32], and from the nonlinear dynamics aspect, which leads us to *chaos* [1, 47].

In this chapter we summarize the main concepts, models, and algorithms that have been used or developed in the three above nonlinear methodologies for speech analysis and synthesis.

### 5.1.2. Speech turbulence

Conservation of momentum in the air flow during speech production yields the Navier-Stokes equation [72]:

$$\rho\left(\frac{\partial \vec{u}}{\partial t} + \vec{u}\cdot\nabla\vec{u}\right) = -\nabla p + \mu\nabla^2\vec{u}, \tag{5.1}$$

where $\rho$ is the air density, $p$ is the air pressure, $\vec{u}$ is the (vector) air particle velocity, and $\mu$ is the air viscosity coefficient. It is assumed that flow compressibility is negligible [valid since in speech flow (Mach numbers)$^2 \ll 1$] and hence $\nabla\cdot\vec{u} = 0$. An important parameter characterizing the type of flow is the Reynolds number $\mathrm{Re} = \rho UL/\mu$, where $U$ is a velocity scale for $\vec{u}$ and $L$ is a typical length scale, for example, the tract diameter. For the air we have very low $\mu$ and hence high Re. This causes the inertia forces to have a much larger order of magnitude than the viscous forces $\mu\nabla^2\vec{u}$. A *vortex* is a region of similar (or constant) vorticity $\vec{\omega}$, where $\vec{\omega} = \nabla\times\vec{u}$. Vortices in the air flow have been experimentally found above the glottis by Teager and Teager [69] and Thomas [70] and theoretically predicted by Kaiser [25], Teager and Teager [69], and McGowan [40] using simple geometries. There are several mechanisms for the creation of vortices: (1) velocity gradients in boundary layers, (2) separation of flow, which can easily happen at cavity inlets due to adverse pressure gradients (see [69] for experimental evidence), and (3) curved geometry of tract boundaries, where due to the dominant inertia forces the flow follows the curvature and develops rotational components. After a vortex has been created, it can propagate downstream as governed by the vorticity equation [72]

$$\frac{\partial\vec{\omega}}{\partial t} + \vec{u}\cdot\nabla\vec{\omega} = \vec{\omega}\cdot\nabla\vec{u} + \nu\nabla^2\vec{\omega}, \quad \nu = \frac{\mu}{\rho}. \tag{5.2}$$

The term $\vec{\omega}\cdot\nabla\vec{u}$ causes vortex twisting and stretching, whereas $\nu\nabla^2\vec{\omega}$ produces diffusion of vorticity. As Re increases (e.g., in fricative sounds or during loud speech), all these phenomena may lead to instabilities and eventually result in *turbulent flow*, which is a "state of continuous instability" [72] characterized by broad-spectrum rapidly varying (in space and time) velocity and vorticity. Many speech sounds, especially fricatives and stops, contain various amounts of turbulence. In the linear speech model this has been dealt with by having a white noise source exciting the vocal tract filter.

Modern theories that attempt to explain turbulence [72] predict the existence of eddies (vortices with a characteristic size $\lambda$) at multiple scales. According to the

energy cascade theory, energy produced by eddies with large size $\lambda$ is transferred hierarchically to the small-size eddies which actually dissipate this energy due to viscosity. A related result is the Kolmogorov law

$$E(k, r) \propto r^{2/3} k^{-5/3}, \tag{5.3}$$

where $k = 2\pi/\lambda$ is the wavenumber in a finite nonzero range, $r$ is the energy dissipation rate, and $E(k, r)$ is the velocity wavenumber spectrum, that is, Fourier transform of spatial correlations. This multiscale structure of turbulence can in some cases be quantified by *fractals*. Mandelbrot [32] and others have conjectured that several geometrical aspects of turbulence (e.g., shapes of turbulent spots, boundaries of some vortex types found in turbulent flows, shape of particle paths) are fractal in nature. We may also attempt to understand aspects of turbulence as cases of chaos. Specifically, chaotic dynamical systems converge to attractors whose sets in phase space or related time series signals can be modeled by fractals; references can be found in [47]. Now there are several mechanisms in high-Re speech flows that can be viewed as routes to chaos; for example, vortices twist, stretch, and fold (due to the bounded tract geometry) [32, 72]. This process of twisting, stretching, and folding has been found in low-order nonlinear dynamical systems to give rise to chaos and fractal attractors.

### 5.1.3. Speech modulations

By "speech resonances" we will loosely refer to the oscillator systems formed by local vocal tract cavities emphasizing certain frequencies and de-emphasizing others. Although the linear model assumes that each speech resonance signal is a damped cosine with constant frequency within 10–30 ms and exponentially decaying amplitude, there is much experimental and theoretical evidence for the existence of *amplitude modulation* (AM) *and frequency modulation* (FM) in speech resonance signals, which make the amplitude and frequency of the resonance vary instantaneously within a pitch period. First, due to the *airflow separation* [69], the air jet flowing through the vocal tract during speech production is highly unstable and oscillates between its walls, attaching or detaching itself, and thereby changing the effective cross-sectional areas and air masses. This can cause modulations of the air pressure and velocity fields, because slow time variations of the elements of simple oscillators can result in amplitude or frequency modulation of the oscillator's sinusoidal response. Also, during speech production vortices can easily be generated and propagated along the vocal tract [70, 72], while acting as modulators of the energy of the jet. Motivated by this evidence, Maragos et al. [36] proposed to model each speech resonance with an AM-FM signal

$$x(t) = a(t) \cos\left[\phi(t)\right] = a(t) \cos\left[2\pi \int_0^t f(\tau) d\tau\right] \tag{5.4}$$

and the total speech signal as a superposition of such AM-FM signals, $\sum_k a_k(t) \cos[\phi_k(t)]$, one for each formant. Here $a(t)$ is the instantaneous amplitude signal and $f(t)$ is the instantaneous cyclic frequency representing the time-varying formant signal. The short-time formant frequency average $f_c = (1/T)\int_0^T f(t)dt$, where $T$ is in the order of a pitch period, is viewed as the carrier frequency of the AM-FM signal. The classical linear model of speech views a formant frequency as constant, that is, equal to $f_c$, over a short-time (10–30 ms) frame. However, the AM-FM model can both yield the average $f_c$ and provide additional information about the formant's instantaneous frequency deviation $f(t) - f_c$ and its amplitude intensity $|a(t)|$.

### 5.1.4. So if speech is nonlinear, why do linear methods work? Conventional speech synthesis approaches

Conventionally the main approaches to speech synthesis depend on the type of modeling used. This may be a model of the speech organs themselves (articulatory synthesis), a model derived from the speech signal (waveform synthesis), or alternatively the use of prerecorded segments extracted from a database and joined together (concatenative synthesis).

Modeling the actual speech organs is an attractive approach, since it can be regarded as being a model of the fundamental level of speech production. An accurate articulatory model would allow all types of speech to be synthesized in a natural manner, without having to make many of the assumptions required by other techniques (such as attempting to separate the source and vocal tract parts out from one signal) [19, 24, 28]. Realistic articulatory synthesis is an extremely complex process, and the data required is not at all easy to collect. As such, it has not to date found any commercial application and is still more of a research tool.

Waveform synthesizers derive a model from the speech signal as opposed to the speech organs. This approach is derived from the linear source-filter theory of speech production [17]. The simplest form of waveform synthesis is based on linear prediction (LP) [38]. The resulting quality is extremely poor for voiced speech, sounding very robotic.

Formant synthesis uses a bank of filters, each of which represents the contribution of one of the formants. The best known formant synthesizer is the Klatt synthesizer [26], which has been exploited commercially as DECTalk. The synthesized speech quality is considerably better than that of the LP method, but still lacks naturalness, even when an advanced voice-source model is used [16].

Concatenation methods involve joining together prerecorded units of speech which are extracted from a database. It must also be possible to change the prosody of the units, so as to impose the prosody required for the phrase that is being generated. The concatenation technique provides the best quality synthesized speech available at present. It is used in a large number of commercial systems, including British Telecom's Laureate [45] and the AT&T Next-Gen system [3]. Although there is a good degree of naturalness in the synthesized output, it is still clearly

distinguishable from real human speech, and it may be that more sophisticated parametric models will eventually overtake it.

Techniques for time and pitch scaling of sounds held in a database are also extremely important. Two main techniques for time-scale and pitch modification in concatenative synthesis can be identified, each of which operates on the speech signal in a different manner. The pitch synchronous overlap add (PSOLA) [41] approach is nonparametric as opposed to the harmonic method, which actually decomposes the signal into explicit source and vocal tract models. PSOLA is reported to give good quality, natural-sounding synthetic speech for moderate pitch and time modifications. Slowing down the speech by a large factor (greater than two) does introduce artifacts due to the repetition of PSOLA bells. Some tonal artifacts (e.g., whistling) also appear with large pitch scaling, especially for higher pitch voices, such as female speakers and children.

McAulay and Quatieri developed a speech generation model that is based on a glottal excitation signal made up of a sum of sine waves [39]. They then used this model to perform time-scale and pitch modification. Starting with the assumption made in the linear model of speech that the speech waveform $x(t)$ is the output generated by passing an excitation waveform $e(t)$ through a linear filter $h(t)$, the excitation is defined as a sum of sine waves of arbitrary amplitudes, frequencies, and phases. A limitation of all these techniques is that they use the linear model of speech as a basis.

## 5.2. What nonlinear methods might we use?

### 5.2.1. Modulation model and energy demodulation algorithms

In the modulation speech model, each speech resonance is modeled as an AM-FM signal and the total speech signal as a superposition of several such AM-FM signals, one for each formant. To isolate a single resonance from the original speech signal, bandpass filtering is first applied around estimates of formant center frequencies.

For demodulating a single resonance signal, Maragos et al. [36] used the nonlinear Teager-Kaiser energy-tracking operator $\Psi[x(t)] = [\dot{x}(t)]^2 - x(t)\ddot{x}(t)$, where $\dot{x} = dx/dt$, to develop the following nonlinear algorithm:

$$\sqrt{\frac{\Psi[\dot{x}(t)]}{\Psi[x(t)]}} \approx 2\pi f(t), \qquad \frac{\Psi[x(t)]}{\sqrt{\Psi[\dot{x}(t)]}} \approx |a(t)|. \tag{5.5}$$

This is the *energy separation algorithm* (ESA) and it provides AM-FM demodulation by tracking the physical energy implicit in the source producing the observed acoustic resonance signal and separating it into its amplitude and frequency components. It yields very good estimates of the instantaneous frequency signal $f(t) \geq 0$ and of the amplitude envelope $|a(t)|$ of an AM-FM signal, assuming that $a(t)$, $f(t)$ do not vary too fast (small bandwidths) or too greatly compared with the carrier frequency $f_c$.

There is also a *discrete* version of the ESA, called DESA [36], which is obtained by using a discrete energy operator on discrete-time nonstationary sinusoids. The DESA is an efficient approach to demodulating speech resonances for the following several reasons. (i) It yields very *small errors* for AM-FM demodulation. (ii) It has an extremely *low computational complexity*. (iii) It has an excellent time resolution, almost *instantaneous*; that is, operates on a 5-sample moving window. (iv) It has a useful and intuitive interpretation of tracking and separating the true physical energy of the acoustic source. (v) It can detect transient events. Extensive experiments on speech demodulation using the DESA in [36, 51, 52] indicate that these amplitude/frequency modulations *exist* in real speech resonances and are necessary for its *naturalness*, as found from synthesizing speech via an AM-FM vocoder [52] that uses the AM-FM model.

The main disadvantage of the DESA is a moderate sensitivity to noise. This can be reduced by using *regularized* versions of the continuous ESA adapted for discrete data. Two such continuous approaches were developed by Dimitriadis and Maragos [13]. The first, called *Spline*-ESA, interpolates the discrete-time signal with *smoothing splines* to create a continuous-time signal, applies the continuous-time ESA (5.5), and finally samples the information-bearing signals to obtain estimates of the instantaneous amplitude and frequency of the original discrete signal. In the second approach, called *Gabor*-ESA, the signal derivatives in the original ESA are replaced by signal convolutions with corresponding derivatives of the Gabor filters' impulse response.

The ESAs are efficient demodulation algorithms only when they are used on narrowband AM-FM signals [6]. This constraint makes the use of *filterbanks* (i.e., parallel arrays of bandpass filters) inevitable for wideband signals like speech. Thus, each short-time segment (analysis frame) of a speech signal is simultaneously filtered by all the bandpass filters of the filterbank, and then each filter output is demodulated using the ESA. Ongoing research in speech modulations has been using filterbanks with Gabor bandpass filters whose center frequencies are spaced either linearly or on a mel-frequency scale [13, 52]. Figure 5.1 shows an example of demodulating three bands of a speech phoneme into their instantaneous amplitude and frequency signals.

While the instant frequency signals produced by demodulating resonances of speech vowels have a quasiperiodic structure, those of fricatives look random. Since fricative and stop sounds contain turbulence, Maragos and Dimakis [11, 35] proposed a *random modulation* model for resonances of fricatives and stops where the instant phase modulation signal is a random process from the $1/f$ noise family. Specifically, they modeled each such speech resonance $R(t)$ as

$$R(t) = a(t)\cos\left(2\pi f_c t + p(t)\right), \qquad E\Big[\,|P(\omega)|^2\Big] \propto \frac{\sigma^2}{|\omega|^\gamma}, \tag{5.6}$$

where $p(t)$ is a random nonlinear phase signal, $P(\omega)$ is its Fourier transform, and $E[\cdot]$ denotes expectation. The power spectral density (PSD), measured either by a sample periodogram $|P(\omega)|^2$ or empirically via filterbanks, is assumed to obey a $1/\omega^\gamma$ power law; such processes are called the "$1/f$ noises." In [35] the

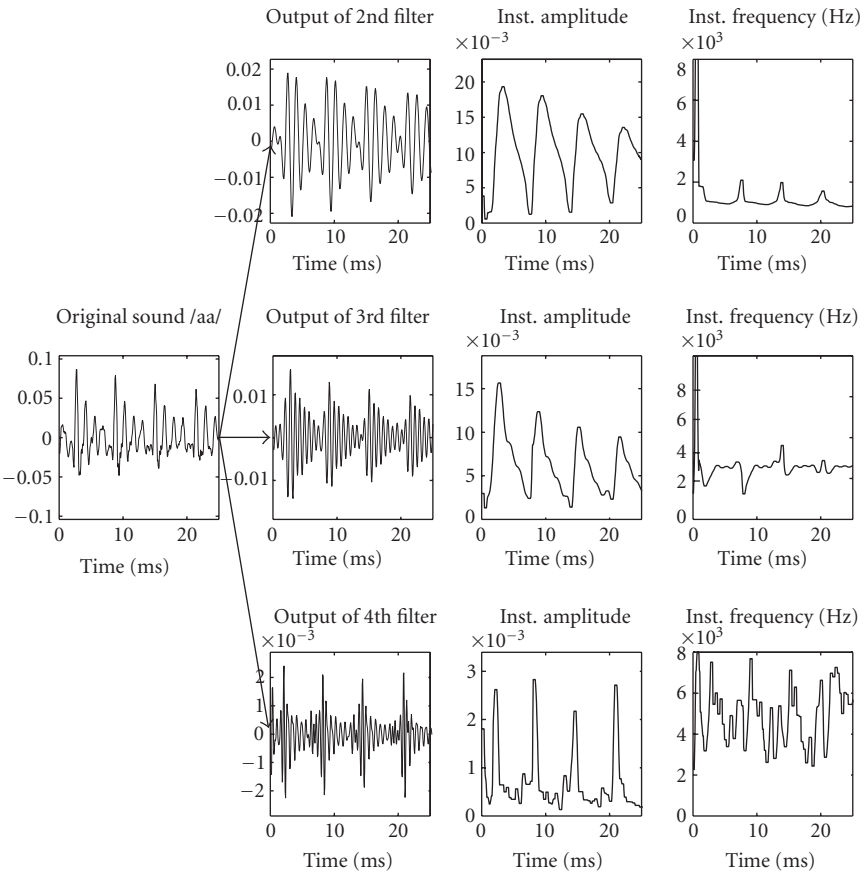



FIGURE 5.1. Demodulating a speech phoneme using a Gabor filterbank and the Spline-ESA.

proposed $1/f$ model for the instant phase was the fractional Brownian motions (FBMs), which are a popular fractal model for a subclass of $1/f$ noises [32]. In [11] this $1/f$ phase model was extended to include the class of *alpha-stable* processes. The method used in [11, 35] to solve the inverse problem, that is, that of extracting the phase modulation $p(t)$ from the speech resonance and modeling it as a $1/f$ noise, is summarized in the following steps.

(1) Isolate the resonance by Gabor bandpass filtering the speech signal.
(2) Use the ESA to estimate the AM and FM signals, $a(t)$ and $f(t)$.
(3) Median filter the FM signal for reducing some extreme spikes.
(4) Estimate the phase modulation signal $p(t)$ by integrating the instant frequency.
(5) Estimate the spectral exponent $\gamma$ of the phase modulation signal by using a statistically robust estimator of its power spectrum and least-squares fitting a line only on the part of the power spectrum not affected by low-pass filtering.

The efficiency of this method was successfully tested on artificial resonance signals with $1/f$ phase modulation signals.

Strong experimental evidence was also presented that certain classes of speech signals have resonances that can be effectively modeled as phase modulated $1/f$ signals. The validity of the model was demonstrated by confirming that its power spectrum obeys a spectral $1/f^{\gamma}$ power law.

Figure 5.2 demonstrates the application of the above described $1/f$ phase modulation model to a voiced fricative. In [11, 35] extensive similar experiments have been performed on real speech signals (from the TIMIT database), by following the same procedure: a strong speech resonance is located, possibly by using the iterative ESA method. Then the ESA is used to extract the phase modulation. (The phase modulations were also estimated via the Hilbert transform to make sure that the ESA does not introduce any artifacts.) The estimated phase is assumed to be a low-passed version of a $1/f^{\gamma}$ random process and the $\gamma$ exponent is estimated from the slope of the power spectrum. In all these experiments the conjecture in [35] that the phase modulation of random speech resonances has a $1/f^{\gamma}$ spectrum has always been verified.

Ongoing work in this area includes better estimation algorithms and a statistical study relating estimated exponents with types of sounds. Some advances can be found in [11].

### 5.2.2. Fractal methods

Motivated by Mandelbrot's conjecture that fractals can model multiscale structures in turbulence, Maragos [34] used the *short-time fractal dimension* of speech sounds as a feature to approximately quantify the degree of turbulence in them. Although this may be a somewhat simplistic analogy, the short-time fractal dimension of speech has been found in [34, 37] to be a feature useful for speech sound classification into phonetic classes, segmentation, and recognition. An efficient algorithm developed in [34] to measure it consists of using multiscale morphological filters that create geometrical covers around the graph of the speech signal, whose fractal dimension $D$ can then be found by

$$D = \lim_{s \to 0} \frac{\log\left[\text{Area of dilated graph by disks of radius } s/s^2\right]}{\log(1/s)}, \tag{5.7}$$

$D$ is between 1 and 2 for speech signals; the larger $D$ is, the larger the amount of geometrical fragmentation of the signal graph is. In practice, real-world signals do not have the same structure over all scales; hence, $D$ is computed by least-squares fitting a line to the log-log data of (5.7) over a small scale window that can move along the $s$ axis and thus create a profile of local *multiscale fractal dimensions* $D(s,t)$ at each time location $t$ of the short speech analysis frame. The function $D(s,t)$ is called a *fractogram*. The fractal dimension at the smallest scale ($s = 1$) can provide some discrimination among various classes of sounds such as vowels (very low $D$), unvoiced fricatives (very high $D$), and voiced fricatives (medium $D$). At higher

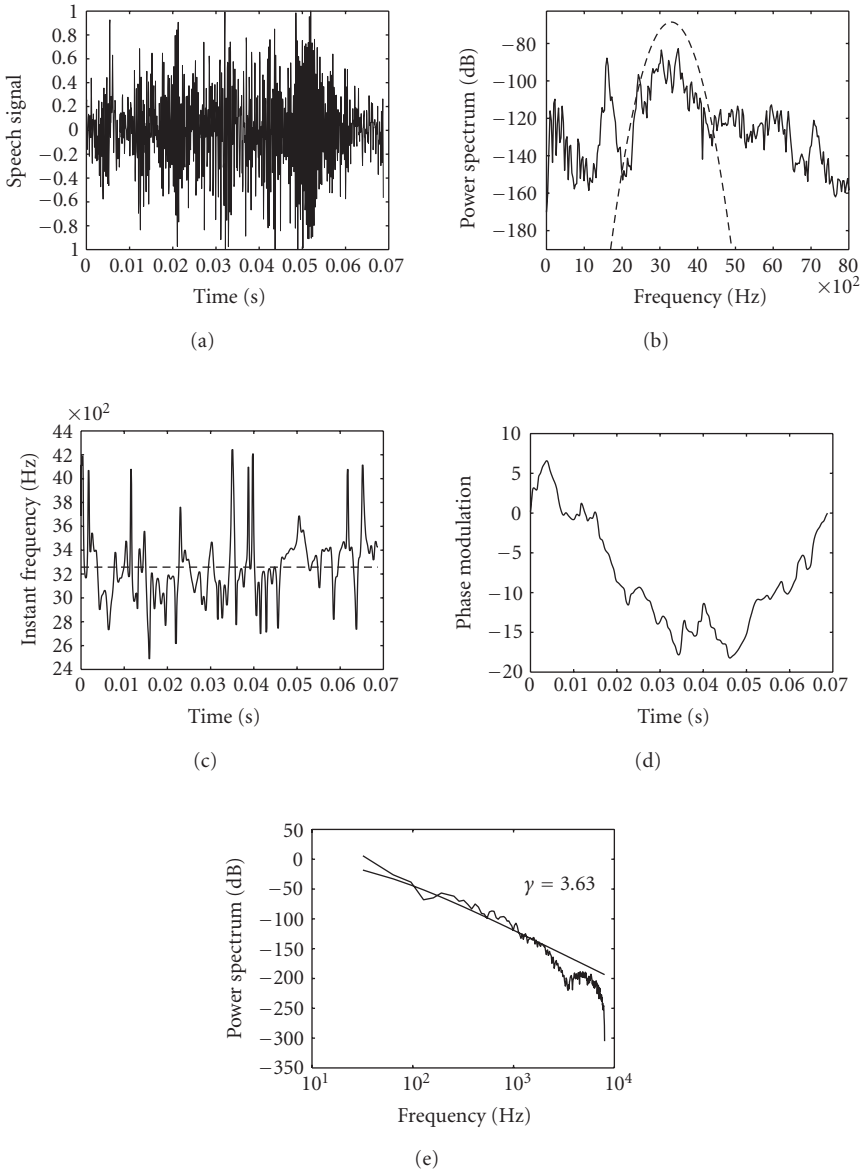



FIGURE 5.2. Experiments with phoneme /z/. (a) Speech signal $s(t)$, (b) PSD of $s(t)$ and Gabor filter, (c) instant frequency, (d) phase modulation $\hat{p}(t)$, and (e) PSD of $\hat{p}(t)$ and estimated slope.

scales, the fractogram multiscale fractal dimension profile can also offer additional information that helps in discriminating among speech sounds, see Figure 5.3.

Related to the Kolmogorov 5/3 law (5.3) is the fact that the variance between particle velocities at two spatial locations $X$ and $X + \Delta X$ varies $\propto |\Delta X|^{2/3}$. By

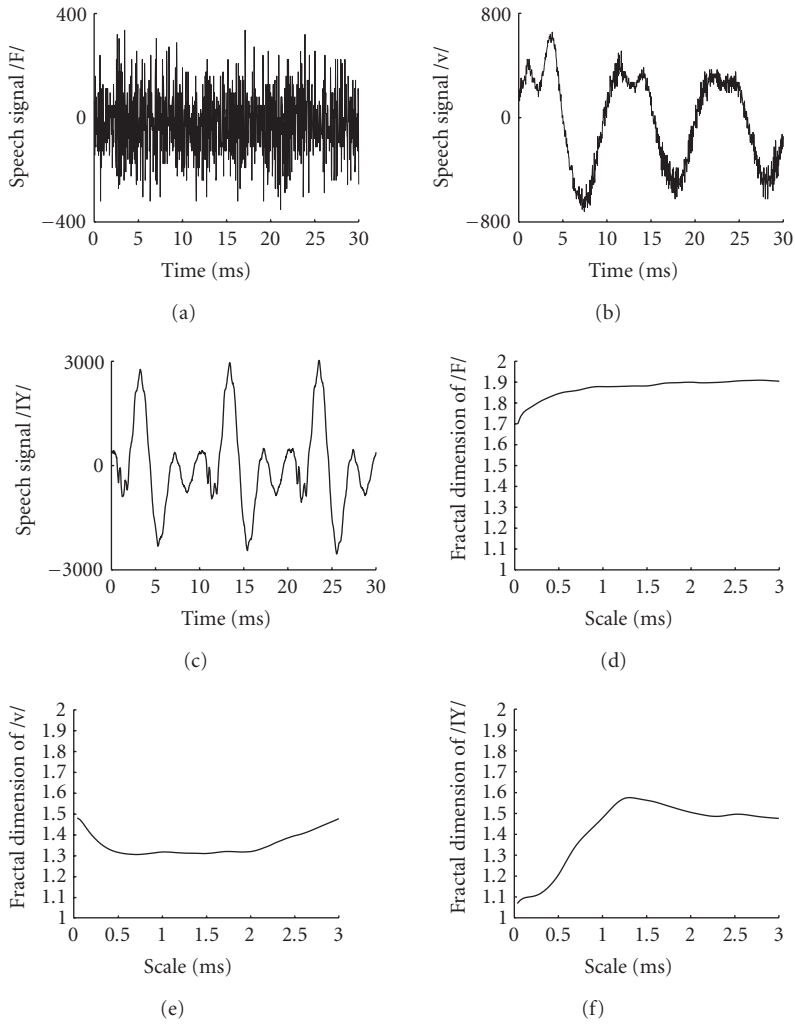



FIGURE 5.3. (a), (b), and (c) show waveforms from speech sounds sampled at 30 kHz. (d), (e), and (f) show their multiscale fractal dimensions estimated over moving windows of 10 scales.

linking this to similar scaling laws in FBMs, it was concluded in [34] that speech turbulence leads to fractal dimension of $D = 5/3$, which was often approximately observed during experiments with fricatives.

### 5.2.3. Poincaré maps and epoch marking

The section discusses how nonlinear techniques can be applied to pitch marking of continuous speech. We wish to locate the instants in the time domain speech signal at which the glottis is closed. A variety of existing methods can be employed

to locate the epochs. These are abrupt change detection [10], maximum likelihood epoch detection [9], and dynamic programming [68]. All of the above techniques are sound and generally provide good epoch detection. The technique presented here should not be viewed as a direct competitor to the methods outlined above. Rather it is an attempt to show the practical application of ideas from nonlinear dynamical theory to a real speech processing problem. The performance in clean speech is comparable to many of the techniques discussed above.

In nonlinear processing a $d$-dimensional system can be reconstructed in an $m$-dimensional state space from a single dimension time series by a process called embedding. Takens' theorem states that $m \geq 2d+1$ for an adequate reconstruction [67], although in practice it is often possible to reduce $m$. An alternative is the singular value decomposition (SVD) embedding [8], which may be more attractive in real systems where noise is an issue.

A Poincaré map is often used in the analysis of dynamical systems. It replaces the flow of an $n$th order continuous system with an $(n-1)$th order discrete-time map. Considering a three-dimensional attractor a Poincaré section slices through the flow of trajectories and the resulting crossings form the Poincaré map. Reexamining the attractor reconstructions of voiced speech shown above, it is evident that these three-dimensional attractors can also be reduced to two-dimensional maps.[1] Additionally, these reconstructions are pitch synchronous, in that one revolution of the attractor is equivalent to one pitch period. This has previously been used for cyclostationary analysis and synchronization [31]; here we examine its use for epoch marking.

The basic processing steps required for a waveform of $N$ points are as follows.

(1) Mark $y_{\mathrm{GCI}}$, a known glottal closure instant (GCI) in the signal.
(2) Perform an SVD embedding on the signal to generate the attractor reconstruction in 3D state space.
(3) Calculate the flow vector, $\mathbf{h}$, at the marked point $\mathbf{y}_{\mathrm{GCI}}$ on the attractor.
(4) Detect crossings of the Poincaré section, $\Sigma$, at this point in state space by signs changes of the scalar product between $\mathbf{h}$ and the vector $\mathbf{y}_i - \mathbf{y}_{\mathrm{GCI}}$ for all $1 \leq i \leq N$ points.
(5) Points on $\Sigma$ which are within the same portion of the manifold as $\mathbf{y}_{\mathrm{GCI}}$ are the epochs.

When dealing with real speech signals a number of practical issues have to be considered. The input signal must be treated on a frame-by-frame basis, within which the speech is assumed stationary. Finding the correct intersection points on the Poincaré section is also a difficult task due to the complicated structure of the attractor. Because of this, additional measures are used for locating the epoch points. The flow chart shown in Figure 5.4 illustrates the entire process. Two different data sets were used to test the performance of the algorithm, giving varying degrees of realistic speech and hence difficulty.

[1] Strictly these attractor reconstructions are discrete-time maps and not continuous flows. However it is possible to construct a flow vector between points and use this for the Poincaré section calculation.

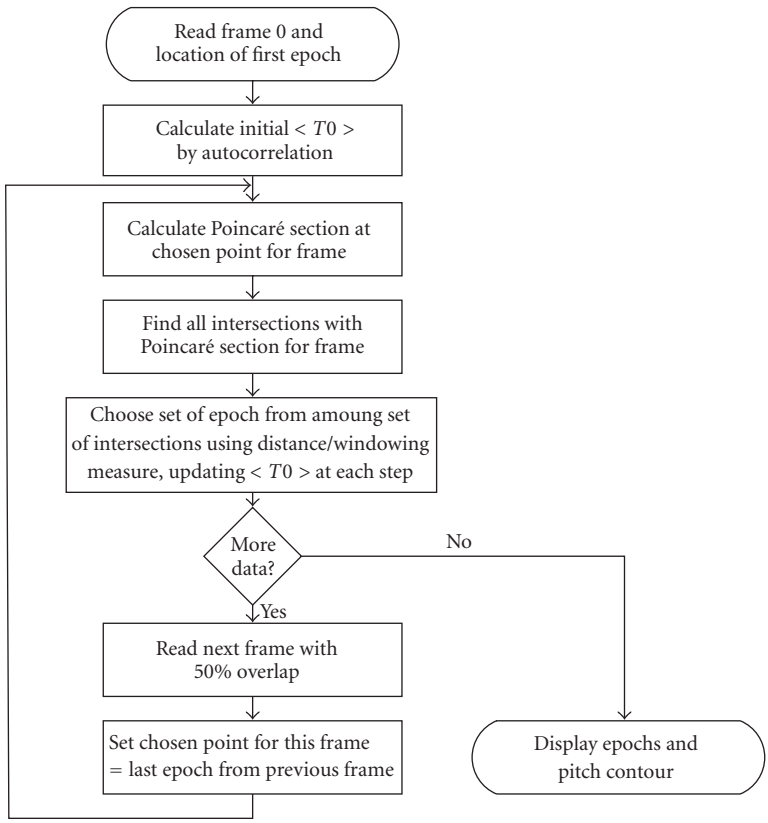



Figure 5.4. Schematic of the epoch marking algorithm.

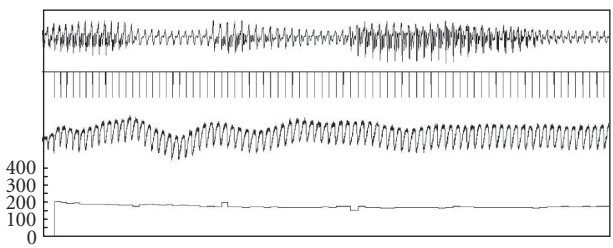



Figure 5.5. Results for the voiced section of "came along" from the Keele database for a female speaker. From top to bottom: the signal; the epochs as calculated by the algorithm; the laryngograph signal; and the pitch contour (Hz) resulting from the algorithm.

(1) Keele University pitch extraction database [49]. This database provides speech and laryngograph data from 15 speakers reading phonetically balanced sentences.

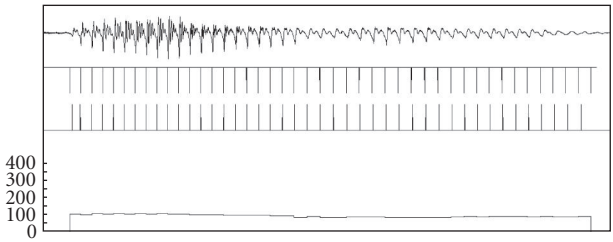



FIGURE 5.6. Results for the voiced section of "raining" from the BT Labs database for a male speaker. From top to bottom: the signal; the epochs as calculated by the algorithm; the processed laryngograph signal; and the pitch contour (Hz) resulting from the algorithm.

(2) BT Labs continuous speech. 2 phrases, spoken by 4 speakers, were processed manually to extract a data set of continuous voiced speech. Laryngograph data was also available.

The signals were up-sampled to 22.05 kHz, the BT data was originally sampled at 12 kHz, and the Keele signals at 20 kHz. All the signals had 16 bit resolution.

Figure 5.5 shows the performance of the algorithm on a voiced section taken from the phrase "a traveller came along wrapped in a warm cloak," spoken by a female speaker. There is considerable change in the signal, and hence in the attractor structure, in this example, yet the epochs are sufficiently well located when compared against the laryngograph signal.

In Figure 5.6, which is a voiced section from the phrase "see if it's raining" spoken by a male speaker, the epochs are well located for the first part of the signal, but some slight loss of synchronization can be seen in the latter part.

### 5.2.4. Pitch variations in LP synthesized vowels: using nonlinear methods

As was made clear in the previous section pitch modification is the key to many applications in speech. In this section we wish to consider the application of nonlinear methods to this problem. In an effort to simplify the problem, vowels generated by a linear prediction synthesizer are examined. The purpose of this exercise, which appears counter-intuitive when dealing with nonlinear synthesis, is to start with a simpler waveform, which can also be generated at any required fundamental frequency. If an analysis of this simple waveform leads to the development of a suitable algorithm for pitch modification, then this is a step towards implementing an algorithm for real speech signals.

Figure 5.7 shows the time domain waveforms and corresponding 2D projections of the 3D phase space structures for the linear prediction synthesized vowel /u/. The LP coefficients were calculated using the constant pitch /u/ vowel sound of the male speaker PB, taken from the Edinburgh 1996 database. The LP filter was then excited by a Dirac pulse train of appropriate period. This generated the stationary synthesized vowels shown, with fundamental frequencies of 70 Hz, 100 Hz, 130 Hz, and 160 Hz. These values of pitch would be typical for a male speaker.

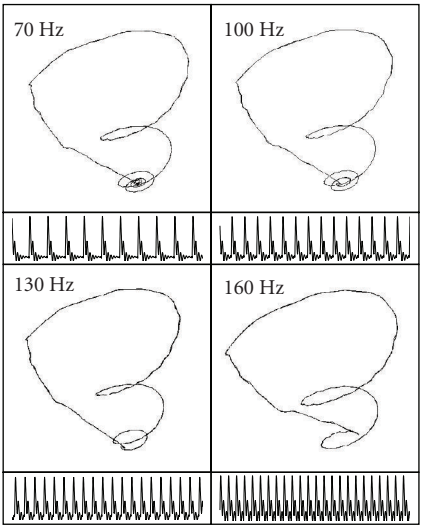



Figure 5.7. LP synthesized vowel /u/ at pitch values of 70 Hz, 100 Hz, 130 Hz, and 160 Hz.

All of these phase space reconstructions are characterized by a trajectory rising steeply out from a center point, which then returns towards this point in a series of downwards spirals. This equates to the excitation by the glottal pulse, followed by a slow decay of the waveform until the next impulse. At lower fundamental frequencies, corresponding to longer pitch periods, the number of spirals is greater, since there is a greater delay between epoch pulses. It is possible to divide each of the phase space reconstructions into two parts. The outer part consists of the outward pulse and the wide initial inward spiral, and is almost constant across all four structures. The inner part consists of the tight inner spirals close to the center point, and it is here that the variation between structures due to the pitch change can be observed. The number of inner spirals appears to be solely due to the length of the pitch period. This topological description, although applied to LP synthesized vowels, also has some similarities with the real speech signals shown previously. By a careful examination a form of outward trajectory followed by inwards spiraling can be seen.

Examining the LP phase space reconstructions from a nonlinear dynamical theory viewpoint, it would appear that there is a fixed point around which the trajectories evolve. Fixed points may take a number of forms. A more complex form is a saddle point, which has trajectories approaching the fixed point close to the inset (stable manifold) and diverging away near the outset (unstable manifold). Index 1 saddle points have an inset that is a surface, and an outset that is a curve, whereas the opposite is true of index 2 saddle points. Spiral saddle points have trajectories spiraling around the fixed point near the surface, towards or away from the saddle, for index 1 and index 2, respectively. Figure 5.8 shows an example of an index 1 spiral saddle point. Looking at the inner part of the LP phase space

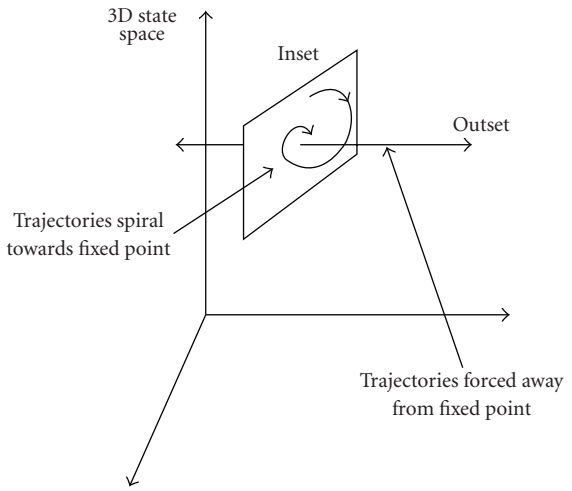



FIGURE 5.8. A spiral saddle point (index 1), with trajectories spiraling towards the fixed point near a surface and diverging away along a curve.

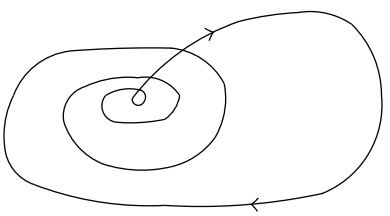


FIGURE 5.9. A Šilnikov-type orbit for an index 1 spiral saddle point.

reconstruction, particularly at low pitches, an index 1 spiral saddle point appears to be a good description.

A homoclinic trajectory occurs when a trajectory on the unstable manifold joins another on the stable manifold, and thus forms a single trajectory connecting the saddle to itself [46]. For spiral saddle points, this type of trajectory is also called a Šilnikov orbit, after Šilnikov's theorem [73], as shown in Figure 5.9. When the inset and outset intersect, then a so-called homoclinic intersection occurs [23]. This leads to the situation where a trajectory on the unstable manifold joins another trajectory on the stable manifold to form a single trajectory. This trajectory joins the saddle point to itself producing a homoclinic orbit.

Trajectories that come near the saddle point will approach the saddle close to the stable manifold and then be forced away from it near the unstable manifold. However, as they are pushed away by the outset, they will be recaptured by the inset and pulled back in towards the saddle point. This description captures very closely the behavior seen in all parts of the LP vowel state space reconstructions. The similarity between vowel phase space reconstructions and Šilnikov orbits has

also been noted by Tokuda et al. [71], in their nonlinear analysis of the Japanese vowel /a/.

#### 5.2.4.1. Attempted application of controlling chaos ideas

This analysis inspires a possible alternative approach to pitch modification, which operates entirely in the state space domain. It is based on concepts from the controlling chaos literature, and involves perturbing the trajectory of the speech signal in state space in order to affect a change in its orbit.

Examining once again the phase space reconstructions for different pitches of the vowel sound as produced by a linear prediction synthesizer (Figure 5.7), it appears that almost all of the information about the higher pitch sounds is contained in the lowest pitch vowel reconstruction. The effect of decreasing pitch, that is, increasing pitch period, is an increase in the number of spirals towards the center of the reconstruction, while the remainder of the phase space structure is approximately constant. Therefore it should be possible to modify the lowest pitch phase space reconstruction in some way, so that a higher pitch version can be produced. This may not be entirely realistic for real vowel sounds, but an algorithm capable of pitch modification of LP synthesized sounds would provide a stepping stone to pitch modification of real voiced speech.

#### 5.2.4.2. Controlling chaos

In the field of nonlinear dynamics, there has been a large amount of interest in the possibility of controlling systems which exhibit chaotic behavior, so as to improve their performance. The basic principle is to locate low-period unstable periodic orbits within the attractor, which mainly comprises a large number of uncontrolled chaotic orbits. Then, using small perturbations of some control parameter, the system is moved and stabilized onto one of the low-period orbits, which is chosen so that performance is optimized. This was first proposed by Ott et al. [44], and then further refined, allowing the technique to be used with time-delay embedding, by Dressler and Nitsche [15].

#### 5.2.4.3. Principle of pitch modification by small changes

Some of the concepts of this technique can be applied to the low-pitch vowel phase space reconstruction in order to modify the fundamental frequency. Assuming that the phase space reconstruction does have a Šilnikov-type orbit, with an index 1 saddle point at the center, then the trajectory spirals in towards the fixed point near the stable manifold (which is a plane), before being ejected out close to the unstable manifold. The process then repeats with the reinjection back towards the fixed point. The idea is to perturb the trajectory when it is close to the saddle point. Moving the trajectory closer to the plane of the stable manifold will cause it to spend longer in the region of the fixed point, thus increasing the pitch period and decreasing the pitch. Conversely, moving the trajectory away from the stable

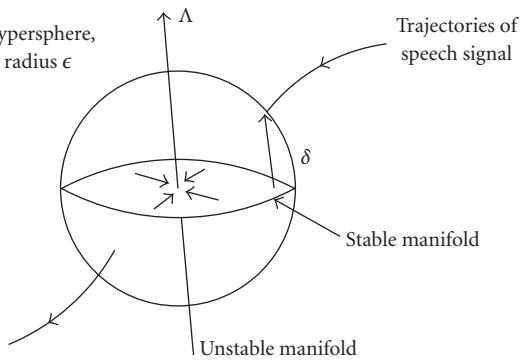



Figure 5.10. Principle of perturbing trajectory to modify pitch.

manifold, in the direction of the unstable manifold, will cause a faster ejection and therefore higher pitch.

An algorithm capable of perturbing the speech trajectories as described would need to perform the following operations.

(i) Embed the time series in 3D state space.

(ii) Locate the fixed point. The center of the phase space reconstruction will be close to the index 1 saddle point, where the two-dimensional stable manifold intersects with the one-dimensional unstable manifold. In practice, it will not be possible to locate the saddle point exactly, only the closest data point to it.

(iii) Find the direction of the stable and unstable manifolds. The stable manifold is expected to be a plane and the unstable manifold a curve.

(iv) Perturb the trajectory. Figure 5.10 shows the trajectory approaching the saddle point and entering a sphere of radius $\epsilon$. At the point of entry, it is a distance $\delta$ away from the stable manifold in the direction of the unstable manifold, whose magnitude is $\Lambda$. By perturbing the trajectory towards the stable manifold (i.e., decreasing $\delta$), the trajectory will spend longer near the fixed point, whereas moving the trajectory away from the manifold (an increase in $\delta$) will cause a faster ejection.

(v) Calculate the relationship between the size of the perturbation and the change in pitch, so that arbitrary pitch modifications can be made.

#### 5.2.4.4. Period modification in a Šilnikov flow

Before attempting to apply the above algorithm to the LP speech signal, modifying the period of a system which is completely specified, via a set of equations, will be examined. Consider the three coupled differential equations:

$$\begin{aligned} \dot{x} &= \alpha x - \beta y, \\ \dot{y} &= \beta x + \alpha y, \\ \dot{z} &= \gamma z. \end{aligned} \tag{5.8}$$

These define a Šilnikov flow in the region around the fixed point, although they do not model the reinjection that is characteristic of a homoclinic orbit. The system can be seen to have a fixed point at $(0, 0, 0)$, since the time derivatives go to zero at this point. Performing an eigenanalysis, the eigenvalues are found to be at

$$\begin{aligned} \lambda_1 &= \gamma, \\ \lambda_{2,3} &= \alpha \pm j\beta. \end{aligned} \tag{5.9}$$

The eigenvalue $\lambda_1 = \gamma$ has a corresponding eigenvector of $(0, 0, 1)$, and the eigenvectors of the complex conjugate eigenvalues are both in the $x$-$y$ plane. If $\alpha$ is negative and $\gamma$ is positive, then this defines an index 1 spiral saddle. Trajectories will spiral in towards the fixed point near the $x$-$y$ stable manifold and then will be ejected out near the $z$ unstable manifold.

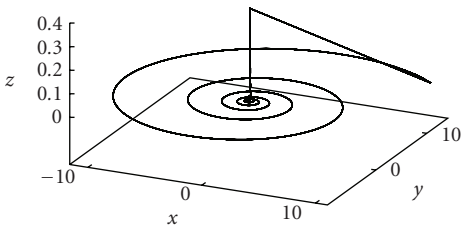
Choosing the values $\alpha = -0.1$, $\beta = 1.0$, and $\gamma = 0.08$, the equations were iterated using the fourth-order Runge-Kutta method [53], with a step size of 0.1. In order to simulate a homoclinic orbit, the trajectory was reset back to its start point after it had been ejected out along the unstable manifold a considerable distance from the fixed point (the threshold was set at $z > 0.4$). The resulting state space plot is shown in Figure 5.11(a). Plotting the variable $x$ against time, as seen in Figure 5.11(b), results in a periodic waveform. The reinjection to complete the orbit is clearly seen. Evidently this is not very realistic, but this is not relevant as it is only the inward spiral followed by ejection that is of interest. In these terms, (5.8) with reinjection generates a realistic homoclinic orbit.

Now the principle of perturbing the trajectory is applied. To reiterate, moving the trajectory towards the stable manifold should cause an increase in the period, whereas moving away from the stable manifold (in the direction of the unstable eigenvector) will decrease the period. When the trajectory enters a sphere of radius $\epsilon$ during the Runge-Kutta iteration, it is then modified. In the experiments presented here, $\epsilon$ was set at 0.1, which is approximately 1% of the spiral radius in the $x$-$y$ plane. In order to find the relationship between the period length and the modification factor, the distance from the stable manifold in the direction of the unstable manifold, $\delta$, and the corresponding period length, $n$, were recorded over a range of values. Plotting $n$ against $\log \delta$ results in a straight line, as seen in Figure 5.12. Therefore the period length $n$ can be expressed as

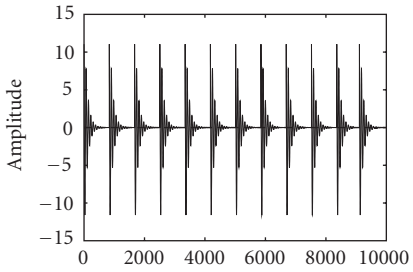
$$n = a \ln \delta + b, \tag{5.10}$$

where $a$ and $b$ are constants, which are easily calculated from simultaneous equations. Denoting the distance from the stable manifold as the trajectory enters the sphere before modification as $\delta_0$, then the multiplier, $R$, required to change the period length to $n_d$ samples is

$$R = \frac{e^{(n_d - b)/a}}{\delta_0}. \tag{5.11}$$




(a)




(b)

FIGURE 5.11. Šilnikov orbit with reinjection from (5.8): (a) 3D state space and (b) $x$ against time.

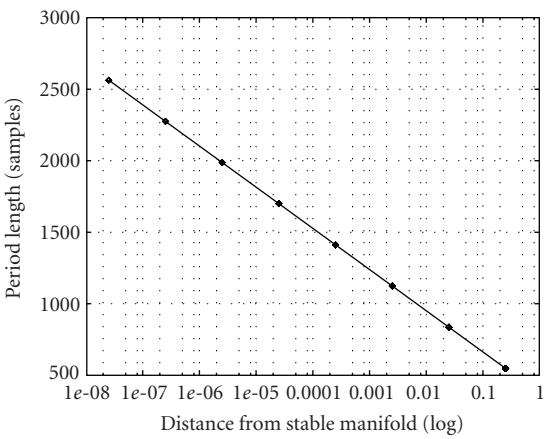



FIGURE 5.12. Log relationship between period length and distance from stable manifold.

Upon entering the sphere, the position vector $(x, y, z)$ is modified to $(x, y, Rz)$, causing a move towards or away from the stable manifold in the direction of the unstable eigenvector.

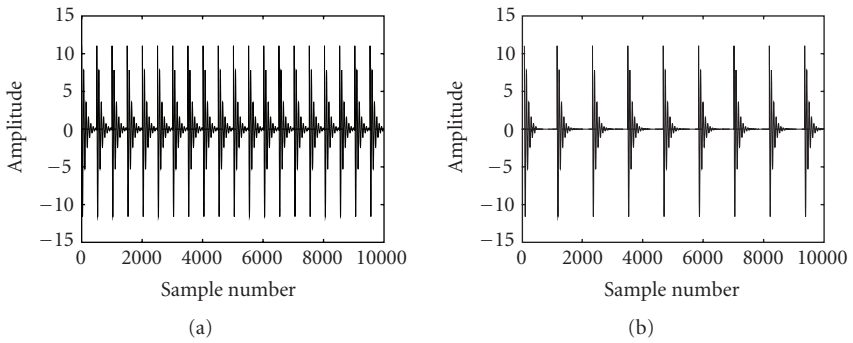



FIGURE 5.13. Period modification of the Šilnikov orbit by a factor of (a) 0.6 and (b) 1.4.

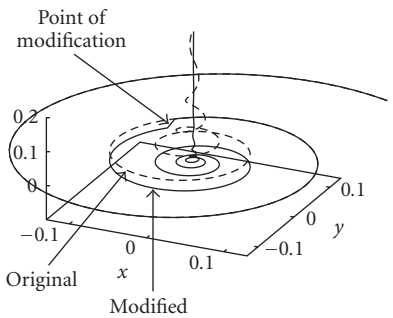



FIGURE 5.14. Zoom in on the fixed point, showing a comparison of the original and the modified trajectories.

The original orbit has a period length of 836 samples. In the following examples, the period is modified by 0.6 ($n_d$ = 502 samples) and 1.4 ($n_d$ = 1170 samples). $a$ and $b$ are calculated as −124.99 and 375.92, respectively. The values of $R$ are then found as 14.5 and 0.069. The resulting modified waveforms are shown in Figure 5.13. Figure 5.14 shows a magnified view of the state space plot around the fixed point demonstrating the modification taking place, for the case of period modification by a factor of 1.4.

This demonstrates the validity of the trajectory perturbation approach. A Šilnikov-type orbit, which has a periodic time domain structure quite reminiscent of a vowel, has been modified, allowing both extension and shortening of the period length. In theory, it should be possible to extend the period length by any required factor by moving close to the stable manifold. The limit on period length shortening, on the other hand, is governed by the size of the sphere, since no modification occurs until the trajectory has entered it. However, it should be noted that increasing the size of the sphere beyond a small radius about the fixed point could

introduce some discontinuity, and, when applied to speech, this would introduce audible artifacts.

#### 5.2.4.5. Application to LP speech

Performing trajectory modifications when the system model is derived from a data set, rather than a set of equations, will evidently be a more complex task. The stages of the algorithm outlined in Section 5.2.4.3 and the problems found are now discussed for the LP synthesized vowel /u/.

*Embedding.* The LP data first has a small amount of Gaussian white noise added, to give a signal-to-noise ratio of 20 dB. This adds some variation to the signal, thus making the manifold more than a single trajectory wide. The formation of local neighborhoods can then be made by selecting near points from adjacent trajectories. The data is then embedded in three dimensions using time-delay embedding with $\tau$ set at 12 samples, equal to the first minimum of the mutual information.

*Fixed point location.* Because of the asymptotic nature of the fixed point, the closer the trajectory comes to it, the greater the amount of time that will be spent in the region around it. Therefore, for sampled data, the Euclidean distance between subsequent samples will decrease as the trajectory moves towards the fixed point. The data sample which has the minimum distance between adjacent samples will be the closest data point to the fixed point. This is then used as the best known position of the fixed point, $\mathbf{x}_f$, in subsequent calculations.

*Direction of manifolds.* An index 1 saddle point has a two-dimensional stable manifold and a one-dimensional unstable manifold. The approximate directions of these manifolds are found by an eigenanalysis of the data trajectory about the fixed point. To do this, a tangent map can be formed. Taking $\mathbf{x}_f$, the closest data point to the fixed point, as the center, a neighborhood matrix is constructed from the $M$ points within a hypersphere around $\mathbf{x}_f$. The neighborhood matrix is then evolved forward $a$ samples and recalculated. The tangent map then defines the linear transformation between the two local neighborhood matrices. Its eigenvalues and eigenvectors are found by SVD. Figure 5.15 shows the eigenvectors found by this method for the LP vowel /u/. The saddle point is marked and the three eigenvectors are shown. The largest eigenvector corresponds to the unstable manifold, and the two smaller eigenvectors are in the plane of the stable manifold. The parameters used in this analysis were 25 local neighbors, with an evolve length of 10 samples.

*Perturbing trajectory.* It is now possible to consider perturbing the trajectory, as shown in Figure 5.10, by moving the trajectory either towards or away from the stable manifold in the direction of the unstable manifold, as it enters the sphere of radius $\epsilon$. However, an insurmountable problem is immediately encountered. Perturbing the trajectory implies moving it away from its existing course (defined by the locally linear tangent map calculated at each synthesis step) and into some

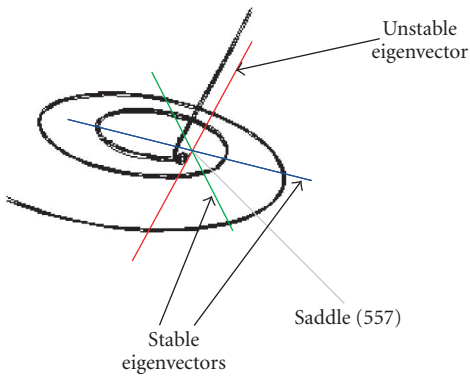



FIGURE 5.15. Eigenvectors around the saddle point for the LP vowel /u/.

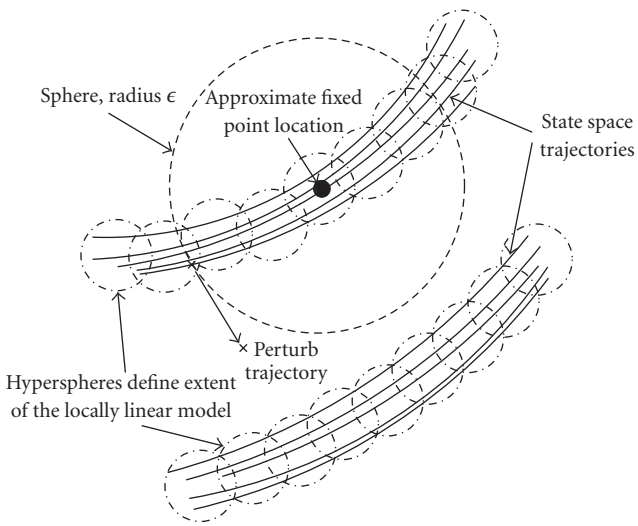



FIGURE 5.16. Stylized diagram of state space around the saddle point, showing two sets of data trajectories. Perturbing trajectory implies moving into an area of state space not covered by the locally linear model.

other area of state space. This part of state space will not contain any data, and hence is not covered by the locally linear model. Therefore perturbing the trajectory moves it into a region of state space that is completely unmodeled, as shown in Figure 5.16. There is no information available to indicate how it should continue to evolve (contrary to the previous example with the set of equations that defined all points in state space), and so continuing the synthesis process after perturbing the trajectory is impossible.

This means that pitch modification by perturbing the trajectory and locally linear synthesis are not compatible, and leaves the problem of realistic pitch modification unresolved.

### 5.2.5. Functional approximation methods

*Neural network synthesis background.* Kubin and Birgmeier reported an attempt made to use a radial basis function (RBF) network approach to speech synthesis. They propose the use of a nonlinear oscillator, with no external input and global feedback in order to perform the mapping

$$x(n) = \mathcal{A}(\mathbf{x}(n-1)), \tag{5.12}$$

where $\mathbf{x}(n-1)$ is the delay vector with nonunit delays, and $\mathcal{A}$ is the nonlinear mapping function [30].

The initial approach taken [4] used a Kalman-based RBF network, which has all of the network parameters trained by the extended Kalman filter algorithm. The only parameter that must be specified is the number of centers to use. This gives good prediction results, but there are many problems with resynthesis. In particular, they report that extensive manual fine-tuning of the parameters such as dimension, embedding delay, and number and initial positions of the centers is required. Even with this tuning, synthesis of some sounds with complicated phase space reconstructions does not work [30].

In order to overcome this problem, Kubin resorted to a technique that uses all of the data points in the training data frame as centers [30]. Although this gives correct resynthesis, even allowing the resynthesis of continuous speech using a frame-adaptive approach, it is unsatisfactory due to the very large number of varying parameters, and cannot be seen as actually learning the dynamics of the speech generating system.

Following their dynamical analysis of the Japanese vowel /a/, Tokuda et al. constructed a feed-forward neural network to perform synthesis [71]. Their structure has three layers, with five neurons in the input layer, forty neurons in the hidden layer, and one in the output layer. The time delay in the input delay vector is set at $\tau = 3$ and the weights are learnt by back propagation. Using global feedback, they report successful resynthesis of the Japanese vowel /a/. The signal is noisy, but preserves natural human speech qualities. No further results in terms of speech quality or resynthesis of other vowels are given.

An alternative neural network approach was proposed by Narasimhan et al. This involves separating the voiced source from the vocal tract contribution, and then creating a nonlinear dynamical model of the source [43]. This is achieved by first inverse filtering the speech signal to obtain the linear prediction (LP) residual. Next the residue waveform is low-pass filtered at 1 kHz, then normalized to give a unit amplitude envelope. This processed signal is used as the training data in a time-delay neural network with global feedback. The NN structure reported is extremely complex, consisting of a 30 tap delay line input and two hidden layers of 15 and 10 sigmoid activation functions, with the network training performed using back propagation through time. Finally, the NN model is used in free-running synthesis mode to recreate the voiced source. This is applied to a LP filter in order to synthesize speech. They show that the NN model successfully preserves the jitter of the original excitation signal.

*RBF network for synthesis.* A well-known nonlinear modeling approach is the radial basis function neural network. It is generally composed of three layers, made up of an input layer of source nodes, a nonlinear hidden layer, and an output layer giving the network response. The hidden layer performs a nonlinear transformation mapping the input space to a new space, in which the problem can be better solved. The output is the result of linearly combining the hidden space, multiplying each hidden layer output by a weight whose value is determined during the training process.

The general equation of an RBF network with an input vector $\mathbf{x}$ and a single output is

$$\mathcal{F}(\mathbf{x}(n)) = \sum_{j=1}^{P} w_j \phi(||\mathbf{x} - \mathbf{c}_j||), \tag{5.13}$$

where there are $P$ hidden units, each of which is weighted by $w_j$. The hidden units, $\phi(\|\mathbf{x} - \mathbf{c}_j\|)$, are radially symmetric functions about the point $\mathbf{c}_j$, called a center, in the hidden space, with $\|\cdot\|$ being the Euclidean vector norm [42]. The actual choice of nonlinearity does not appear to be crucial to the performance of the network. There are two distinct strategies for training an RBF network. The most common approach divides the problem into two steps. Firstly the center positions and bandwidths are fixed using an unsupervised approach, not dependent on the network output. Then the weights are trained in a supervised manner so as to minimize an error function.

Following from the work of Kubin et al., a nonlinear oscillator structure is used. The RBF network is used to approximate the underlying nonlinear dynamics of a particular stationary voiced sound, by training it to perform the prediction

$$x_{i+1} = \mathcal{F}(\mathbf{x}_i), \tag{5.14}$$

where $\mathbf{x}_i = \{x_i, x_{(i-\tau)}, \ldots, x_{(i-(m-1)\tau)}\}$ is a vector of previous inputs spaced by some delay $\tau$ samples, and $\mathcal{F}$ is a nonlinear mapping function. From a nonlinear dynamical theory perspective, this can be viewed as a time-delay embedding of the speech signal into an $m$-dimensional state space to produce a state space reconstruction of the original $d$-dimensional system attractor. The embedding dimension is chosen in accordance with Takens' embedding theorem [67] and the embedding delay, $\tau$, is chosen as the first minimum of the average mutual information function [18]. The other parameters that must be chosen are the bandwidth, the number and position of the centers, and the length of training data to be used. With these sets, the determination of the weights is linear in the parameters and is solved by minimizing a sum of squares error function, $E_S(\widehat{\mathcal{F}})$, over the $N$ samples of training data:

$$E_s(\widehat{\mathcal{F}}) = \frac{1}{2} \sum_{i=1}^{N} (\hat{x}_i - x_i)^2, \tag{5.15}$$

where $\hat{x}_i$ is the network approximation of the actual speech signal $x_i$. Incorporating (5.13) into the above and differentiating with respect to the weights, then setting the derivative equal to zero gives the least-squares problem [5], which can be written in matrix form as

$$(\mathbf{\Phi}^T\mathbf{\Phi})\mathbf{w}^T = \mathbf{\Phi}^T\mathbf{x}, \tag{5.16}$$

where $\mathbf{\Phi}$ is an $N \times P$ matrix of the outputs of the centers; $\mathbf{x}$ is the target vector of length $N$; and $\mathbf{w}$ is the $P$ length vector of weights. This can be solved by standard matrix inversion techniques.

Two types of center positioning strategy were considered.

(1) Data subset. Centers are picked as points from around the state space reconstruction. They are chosen pseudorandomly, so as to give an approximately uniform spacing of centers about the state space reconstruction.
(2) Hyperlattice. An alternative, data independent approach is to spread the centers uniformly over an $m$-dimensional hyperlattice.

*Synthesis.* From analysis, an initial set of parameters with which to attempt resynthesis was chosen. The parameters were set at the following values.

Bandwidth = 0.8 for hyperlattice, 0.5 for data subset; dimension = 7; number of centers = 128; hyperlattice size = 1.0; and training length = 1000.

For each vowel in the database, the weights were learnt, with the centers either on a 7D hyperlattice, or chosen as a subset of the training data. The global feedback loop was then put in place to allow free-running synthesis. The results gave varying degrees of success, from constant (sometimes zero) outputs, through periodic cycles not resembling the original speech signal and noise-like signals, to extremely large spikes at irregular intervals on otherwise correct waveforms [33].

These results implied that a large number of the mapping functions learnt by the network suffered from some form of instability. This could have been due to a lack of smoothness in the function, in which case regularization theory was the ideal solution. Regularization theory applies some prior knowledge, or constraints, to the mapping function to make a well-posed problem [21].

The selection of an appropriate value for the regularization parameter, $\lambda$, is done by the use of cross-validation [5]. After choosing all the other network parameters, these are held constant and $\lambda$ is varied. For each value of $\lambda$, the MSE on an unseen validation set is calculated. The MSE curve should have a minimum indicating the best value of $\lambda$ for generalization. With the regularization parameter chosen by this method, the 7D resynthesis gave correct results for all of the signals except KH /i/ and KH /u/ when using the data subset method of center selection. However, only two signals (CA /i/ and MC /i/) were correctly resynthesized by the hyperlattice method. It was found that $\lambda$ needed to be increased significantly to ensure correct resynthesis for all the signals when the hyperlattice was used. Achieving stable resynthesis inevitably comes at some cost. By forcing smoothness onto the approximated function there is the risk that some of the finer detail of the state space reconstruction will be lost. Therefore, for best results, $\lambda$ should be set at the smallest possible value that allows stable resynthesis. The performance of

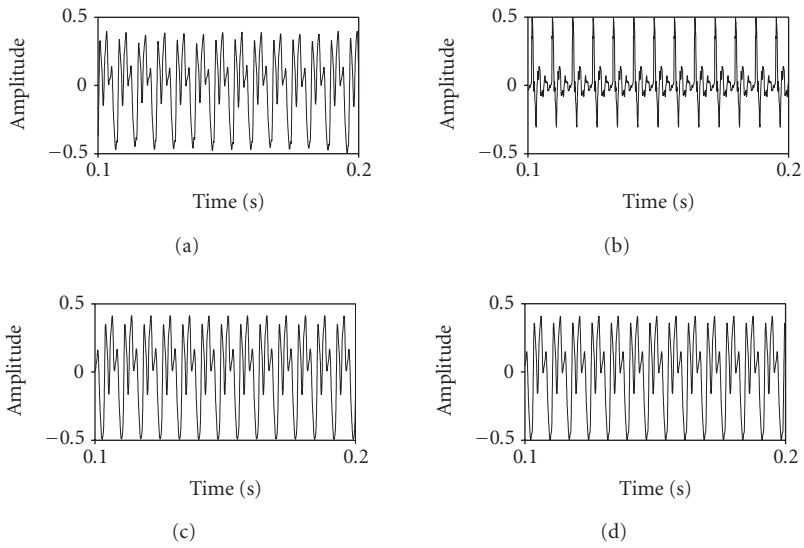
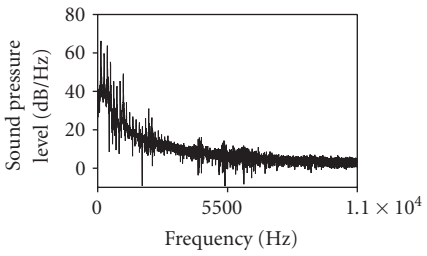


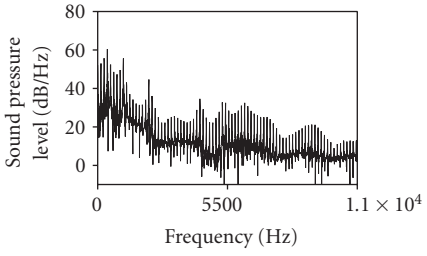

FIGURE 5.17. Time domain examples of the vowel /u/, speaker MC. (a) Original signal and (b) linear prediction synthesized signal; RBF network synthesized signal for (c) hyperlattice and (d) data subset. (Reprinted from Signal Processing, Vol.81, Lain Mann and Stephen McLaughlin, "Synthesising natural-sounding vowels using a nonlinear dynamical model," pages 1743–1756 © 2001 with permission Elsevier Science.)

the regularized RBF network as a nonlinear speech synthesizer is now measured by examining the time and frequency domains, as well as the dynamical properties. In addition to comparing the output of the nonlinear synthesizer to the original speech signal, the synthetic speech from a traditional linear prediction synthesizer is also considered. In this case, the LP filter coefficients were found from the original vowel sound (analogous to the training stage of the RBF network). The estimate $(F_s + 4)$ [56] was used to set the number of filter taps to 26. Then, using the source-filter model, the LP filter was excited by a Dirac pulse train to produce the desired length LP synthesized signal. The distance between Dirac pulses was set to be equal to the average pitch period of the original signal. In this way, the three vowel sounds for each of the four speakers in the database were synthesized.

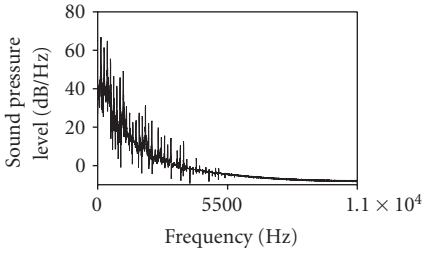
Figure 5.17 shows the time domain waveforms for the original signal, the LP synthesized signal and the two RBF synthesized signals, for the vowel /u/, speaker MC. Figure 5.18 shows the corresponding frequency domain plots of the signals, and the spectrograms are shown in Figure 5.19. In these examples, the regularization parameter $\lambda$ was set at 0.01 for the hyperlattice, and 0.005 for the data subset. In the linear prediction case, the technique attempts to model the spectral features of the original, hence results in the reasonable match seen in the spectrum (although the high frequencies have been overemphasized), but the lack of resemblance in the time domain. The RBF techniques, on the other hand, resemble the original in the time domain, since it is from this that the state space reconstruction



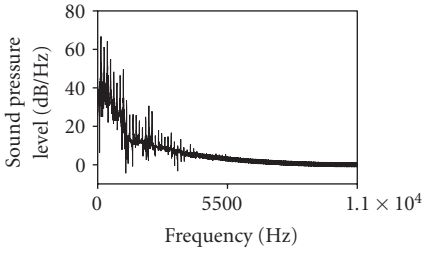

(a)




(b)




(c)




(d)

FIGURE 5.18. Spectrums for examples of the vowel /u/, corresponding to the signals in Figure 5.17. (a) Original signal, (b) linear prediction synthesized signal; RBF network synthesized signal for (c) hyperlattice, (d) data subset. (Reprinted from Signal Processing, Vol.81, Lain Mann and Stephen McLaughlin, "Synthesising natural-sounding vowels using a nonlinear dynamical model," pages 1743–1756 © 2001 with permission Elsevier Science.)

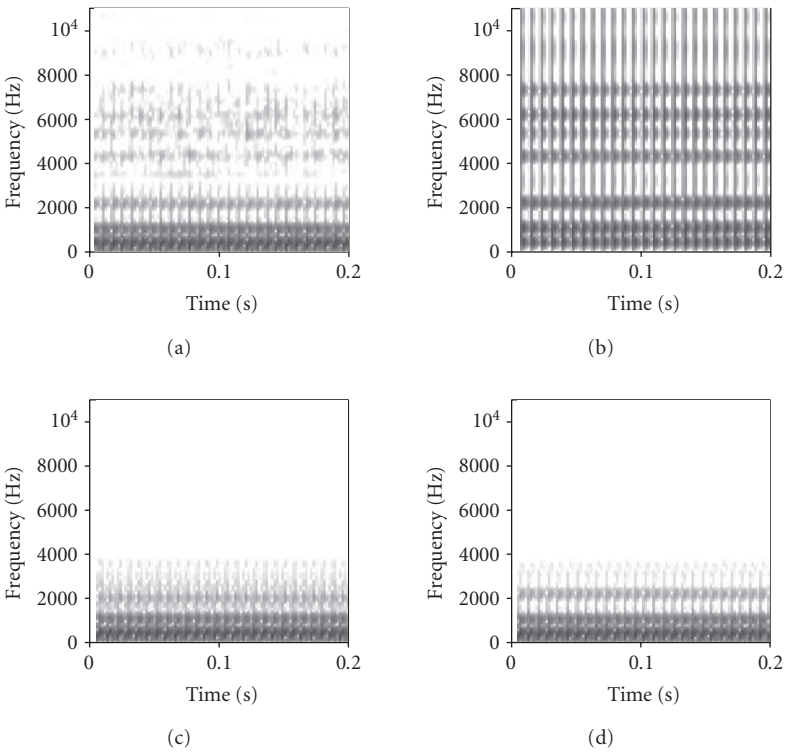



FIGURE 5.19. Wideband spectrograms for examples of the vowel /u/ corresponding to the signals in Figure 5.17. (a) Original signal, (b) linear prediction synthesized signal; RBF network synthesized signal for (c) hyperlattice, (d) data subset. (Reprinted from Signal Processing, Vol.81, Lain Mann and Stephen McLaughlin, "Synthesising natural-sounding vowels using a nonlinear dynamical model," pages 1743–1756 © 2001 with permission Elsevier Science.)

is formed, although the spectral plots show the higher frequencies have not been well modeled by this method. This is because the networks have missed some of the very fine variations of the original time domain waveform, which may be due to the regularization.

Further spectrogram examples for different vowels and speakers follow the same pattern, with the size of $\lambda$ being seen to influence the quality of the signal at high frequencies.

*Jitter and shimmer.* Jitter and shimmer measurements were made on all of the original and RBF synthesized waveforms, using epoch detection[2] over a 500 millisecond window. Jitter is defined as the variation in length of individual pitch periods and for normal healthy speech should be between 0.1% and 1% of the average pitch period [61]. Table 5.1 shows the results of the average pitch length variation, expressed as a percentage of the average pitch period length. Results

[2]Using entropic laboratory's ESPS epoch function.

Table 5.1. Percentage jitter and shimmer in original and synthesized waveforms (hyperlattice and data subset) averaged over the vowels /i/, /a/, and /u/ for each speaker, and as an average over the database. (Reprinted from Signal Processing, Vol.81, Lain Mann and Stephen McLaughlin, "Synthesising natural-sounding vowels using a nonlinear dynamical model," pages 1743–1756 © 2001 with permission Elsevier Science.)

| Data type | MC (male) | CA (female) | Average (female) |
|---|---|---|---|
| Hyperlattice jitter (%) | 0.470 | 1.14 | 0.697 |
| Data subset jitter (%) | 0.482 | 0.663 | 0.521 |
| Original jitter (%) | 0.690 | 0.685 | 0.742 |
| Hyperlattice shimmer (%) | 1.00 | 1.33 | 0.922 |
| Data subset shimmer (%) | 0.694 | 7.65 | 2.34 |
| Original shimmer (%) | 4.21 | 7.06 | 5.17 |

for both center placing techniques are presented, with the jitter measurements of the original speech data. The hyperlattice synthesized waveforms contain more jitter than the data subset signals, and both values are reasonable compared to the original.

Shimmer results (the variations in energy each pitch cycle) for the original and synthesized waveforms are also displayed in Table 5.1. It can be seen that in general there is considerably less shimmer on the synthesized waveforms as compared to the original, which will detract from the quality of the synthetic speech.

*Incorporating pitch into the nonlinear synthesis method.* The approach adopted here is to model the vocal tract as a forced nonlinear oscillator and to embed an observed scalar time series of a vowel with pitch information into a higher dimensional space. This embedding, when carried out correctly, will reconstruct the data onto a higher dimensional surface which embodies the dynamics of the vocal tract, (see, e.g., [63, 64] for issues regarding embedding).

Previous studies, discussed above, have successfully modeled stationary (i.e., constant pitch) vowel sounds using nonlinear methods, but these have very limited use since the pitch cannot be modified to include prosody information. The new approach described here resolves this problem by including pitch information in the embedding. Specifically, a nonstationary vowel sound is extracted from a database and, using standard pitch extraction techniques, a pitch contour is calculated for the time series so that each time domain sample has an associated pitch value. In the present study measurements of rising pitch vowel sounds, where the pitch rises through the length of the time series, have been used as the basis for modeling; see, for example, Figures 5.20 and 5.21.

The time series is then embedded in an $m$-dimensional space, along with the pitch contour, to form an $(m+1)$-dimensional surface. A mixed embedding delay between time samples (greater than unity) is used to capture the variable time scales present in the vowel waveform. The $(m+1)$-dimensional surface is modeled by a nearest neighbor approach, which predicts the next time series sample given

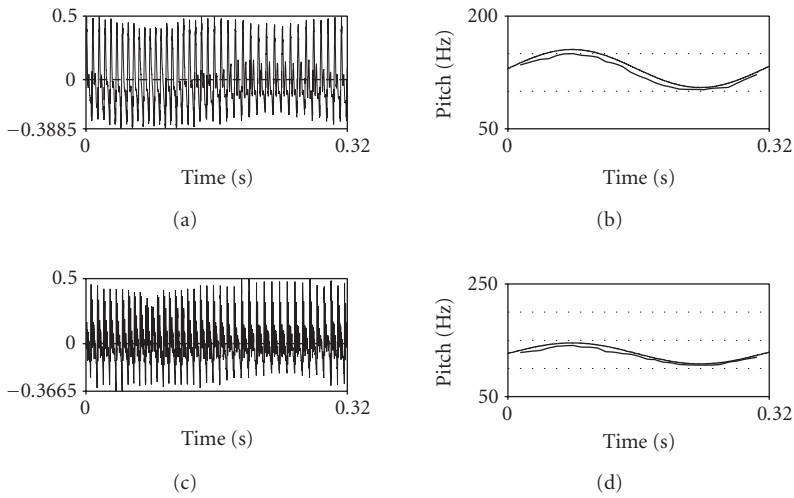



FIGURE 5.20. Synthesized vowel sounds together with desired and measured pitch profiles. (a) RV1, (c) RV4.

a vector of previous time samples and a pitch value (it is envisaged that more sophisticated modeling techniques will be incorporated at a later date).

Synthesis is then performed by a modification of the nonlinear oscillator approach [20], whereby the input signal is removed and the delayed synthesizer output is fed back to form the next input sample. In contrast to previous techniques, the required pitch contour is also passed into the model as an external forcing input. Our results show that this method allows the vowel sound to be generated correctly for arbitrary specified pitch contours (within the input range of pitch values), even though the training data is only made up of the rising vowel time series and its associated pitch contour. In addition, sounds of arbitrary duration can be readily synthesized by simply running the oscillator for the required length of time. Typical synthesis results are shown. It can be seen that the sinusoidal pitch contour of the synthesized sound is quite different from the rising pitch profile of the measured data; the duration of the synthesized data is also somewhat longer than that of the measured data. The small offset evident between desired and synthesized pitch contours is attributed to a minor calibration error.

The initial results presented here are encouraging. Indeed, perhaps somewhat surprisingly so since a limited measured pitch excitation data set, involving a simple rising pitch profile with a small number of data points at each specific pitch value, was used. Specifically, good synthesis results are obtained using a simple nearest neighbor embedding model with only sparse data (typically around 1000 data points embedded in a space of dimension 17, corresponding to a very low density of around only 1.5 data points per dimension).

*Nonlinear function approximation models for chaotic systems.* In their attempt to model and analyze nonlinear dynamics in speech signals, Kokkinos and Maragos

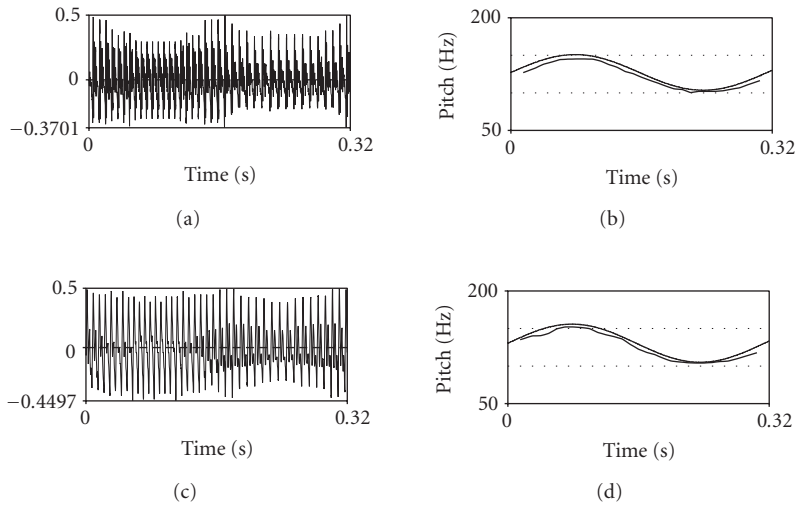



FIGURE 5.21. Synthesized vowel sounds together with desired and measured pitch profiles. (a) RV5, (c) RV6.

[29] have explored the applicability of nonlinear function approximation methods for the approximation of the speech production system dynamics; as in related work, the modeling is done not on the scalar speech signal, but on its reconstructed multidimensional attractor by embedding the scalar signal into a phase space. However, in contrast to the aforementioned approaches to nonlinear speech synthesis, the authors' focus has been on facilitating the application of the methods of chaotic signal analysis even when only a short-time series is available, like phonemes in natural speech utterances. This introduces an increased degree of difficulty that has been dealt with by resorting to sophisticated function approximation models that are appropriate for short data sets. A variety of nonlinear models have been explored, ranging from commonly used approximations based on global or local polynomials as well as approximations inspired from machine learning such as radial basis function networks, fuzzy logic systems and support vector machines.

Among the set of models explored, the authors opted for the use of the Takagi-Sugeno-Kang [66] model from the fuzzy logic literature, which can be seen as a special case of the probabilistic mixture of experts architecture used for function approximation. The expression used for the approximation $\widehat{\mathcal{F}}$ of the nonlinear function $\mathcal{F}$ can be written as

$$\widehat{\mathcal{F}}(X) = \frac{\sum_{i=1}^{M} \mu_i(X)\ell_i(X)}{\sum_{i=1}^{M} \mu_i(X)}, \tag{5.17}$$

where $\mu_i$ measures the degree of membership of $X$ in the $i$th fuzzy set and $\ell_i(X)$ is the local model of the system dynamics for the $i$th fuzzy set. The term $\mu_i$ is typically expressed as a radially symmetric function $\phi(X - C_i)$ centered around point $C_i$

and $\ell_i$ are first-order polynomials in $X$. This expression was found experimentally to give accurate approximations of complex functions using short data records, while its probabilistic interpretation leaves open an interesting perspective for the incorporation of probabilistic information in speech synthesis.

Using this model has enabled the computation of useful features, like Lyapunov exponents, that are used to assist in the characterization of chaotic systems. Specifically, in [29] promising experimental results are reported, demonstrating the usefulness of Lyapunov exponents in the classification of speech phonemes in broad phoneme classes.

### 5.2.6. Nonlinear methods in speech recognition

Despite many decades of research, the current automatic speech recognition (ASR) systems still fall short from the corresponding human cognitive abilities, especially in noisy environments, because of the limitations of their acoustic processing, pattern recognition, and linguistic subsystems. Thus, there is an industrial need to develop improved robust ASR systems. Further, the complexity of the problem requires a long-term vision.

For developing the front end of ASR systems in a way consistent with the nonlinear structure of speech, one direction of nonlinear speech signal processing research has been the work of Maragos, Potamianos, and their collaborators [13, 14, 37, 48, 50]. This consists of two goals: (1) development of new and robust acoustic speech representations of the nonlinear and time-varying type (modulations and fractals) based on improved models for speech production and hearing, and (2) integration/fusion of the perceptually important among the new speech representations and the standard linear ones (cepstrum) to develop improved acoustic processing front ends for general speech recognition systems.

The motivations for the above goals include the following. (i) Adding new information to the feature set such as nonlinear and instantaneous information and good formant tracks derived from the nonlinear model can model better the aerodynamics and time evolution of speech features. (ii) Robustness to large speaker population or large vocabularies with confusable words can be achieved by using speech processing models motivated by the physics of speech production and auditory perception. (iii) Feature specialization can be achieved by investigating which of the new nonlinear features and the standard linear features correspond to the various pieces of information conveyed by the speech waveform. Such a feature tuning can lead to feature economy with corresponding reduction of computation and better acoustic modeling.

Some of the first efforts on using fractal features for ASR include the experiments in [37] on recognition of spoken letters from the *E*-set of the ISOLET database. Incorporating the speech fractogram as additional features to the cepstral feature vector led to a moderate decrease in the recognition error. Generalized fractal features, extracted after embedding the speech signal in a multidimensional space, have given good classification results [48] in discriminating among various

sound classes, for example, fricatives, stops, vowels, and so forth, from the TIMIT database. Further, fractal-related features like the correlation dimension and the fractogram, extracted after a filtering of the nonlinear speech dynamics in the multidimensional embedding space, have yielded a notable decrease in recognition error on standard databases such as AURORA 2, especially in noisy conditions [12]. Finally, the AM-FM modulation features have given an even more significant decrease in recognition error on standard databases such as TIMIT-plus-noise and AURORA 3, as reported in [14]. It appears that these nonlinear features (of the modulation and fractal type) increase the robustness of speech recognition in noise. Ongoing work in this area deals with finding statistically optimal ways to determining the relative weights for *fusing* the nonlinear with the linear (cepstral) features.

### 5.3. Summary

In view of these observations, it seems likely that the data-based model of the vowel dynamics possesses an important degree of structure, perhaps reflecting physiological considerations, that requires further investigation. It is also clear that whilst encouraging there is still some way to go in overcoming the limitations of the approach. It is clear that speech is a nonlinear process and that if we are to achieve the holy grail of truly natural sounding synthetic speech than this must be accounted for. It is also clear that nonlinear synthesis techniques offer some potential to achieve this although a great deal of research work remains to be done. In the field of speech recognition, there is also strong experimental evidence that acoustic features representing various aspects of the nonlinear structure of speech can increase the robustness of recognition systems. However, more research is needed to find optimal ways for fusing the nonlinear with the linear speech features.

## Acknowledgments

S. McLaughlin acknowledges the contributions of his colleagues Iain Mann, Mike Banbrook, Justin Fackrell, and Drew Lowry to this work as well as numerous useful discussions with Professor Bernie Mulgrew and Professor Dave Broomhead. P. Maragos acknowledges the contributions of his collaborators D. Dimitriadis, A. Dimakis, I. Kokkinos, V. Pitsikalis, and A. Potamianos.

## Bibliography

[1] H. D. I. Abarbanel, *Analysis of Observed Chaotic Data*, Springer, New York, NY, USA, 1996.

[2] A. Barney, C. H. Shadle, and P. O. A. L. Davies, "Fluid flow in a dynamic mechanical model of the vocal folds and tract. I. measurements and theory," *The Journal of the Acoustical Society of America*, vol. 105, no. 1, pp. 444–455, 1999.

[3] M. C. Beutnagel, A. D. Conkie, H. J. Schroeter, Y. Stylianou, and A. K. Syrdal, "The AT&T next-gen TTS system," in *Proceedings of Joint Meeting of ASA, EAA, and DEGA*, Berlin, Germany, March 1999.

[4] M. Birgmeier, *Kalman-trained neural networks for signal processing applications*, Ph.D. thesis, Technical University of Vienna, Vienna, Austria, 1996.

[5] C. M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Oxford, UK, 1995.

[6] A. C. Bovik, P. Maragos, and T. F. Quatieri, "AM-FM energy detection and separation in noise using multiband energy operators," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3245–3265, 1993.

[7] D. M. Brookes and P. A. Naylor, "Speech production modelling with variable glottal reflection coefficient," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '88)*, vol. 1, pp. 671–674, New York, NY, USA, April 1988.

[8] D. S. Broomhead and G. P. King, "On the qualitative analysis of experimental dynamical systems," in *Nonlinear Phenonema and Chaos*, pp. 113–144, Adam Hilger, Bristol, UK, 1986.

[9] Y. M. Cheng and D. O'Shaughnessy, "Automatic and reliable estimation of glottal closure instant and period," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 37, no. 12, pp. 1805–1815, 1989.

[10] E. Moulines and R. J. Di Francesco, "Detection of the glottal closure by jumps in the statistical properties of the speech signal," *Speech Communication*, vol. 9, no. 5-6, pp. 401–418, 1990.

[11] A. G. Dimakis and P. Maragos, "Phase-modulated resonances modeled as self-similar processes with application to turbulent sounds," *IEEE Transactions on Signal Processing*, vol. 53, no. 11, pp. 4261–4272, 2005.

[12] D. Dimitriadis, N. Katsamanis, P. Maragos, G. Papandreou, and V. Pitsikalis, "Towards automatic speech recognition in adverse environments," in *Proceedings of 7th Hellenic European Conference on Research on Computer Mathematics and Its Applications (HERCMA '05)*, Athens, Greece, September 2005.

[13] D. Dimitriadis and P. Maragos, "Robust energy demodulation based on continuous models with application to speech recognition," in *Proceedings of 8th European Conference on Speech Communication and Technology (EUROSPEECH '03)*, pp. 2853–2856, Geneva, Switzerland, September 2003.

[14] D. Dimitriadis, P. Maragos, and A. Potamianos, "Robust AM-FM features for speech recognition," *IEEE Signal Processing Letters*, vol. 12, no. 9, pp. 621–624, 2005.

[15] U. Dressler and G. Nitsche, "Controlling chaos using time delay coordinates," *Physical Review Letters*, vol. 68, no. 1, pp. 1–4, 1992.

[16] M. Edgington, A. Lowry, P. Jackson, A. P. Breen, and S. Minnis, "Overview of current text-to-speech techniques: Part II—prosody and speech generation," *BT Technical Journal*, vol. 14, no. 1, pp. 84–99, 1996.

[17] G. Fant, *Acoustic Theory of Speech Production*, Mouton, The Hague, The Netherlands, 1960.

[18] A. M. Fraser and H. L. Swinney, "Independent coordinates for strange attractors from mutual information," *Physical Review A*, vol. 33, no. 2, pp. 1134–1140, 1986.

[19] B. Gabioud, "Articulatory models in speech synthesis," in *Fundamentals of Speech Synthesis and Speech Recognition*, pp. 215–230, John Wiley & Sons, New York, NY, USA, 1994.

[20] H. Haas and G. Kubin, "A multi-band nonlinear oscillator model for speech," in *Proceedings of 32nd IEEE Asilomar Conference on Signals, Systems & Computers*, vol. 1, pp. 338–342, Pacific Grove, Calif, USA, November 1998.

[21] S. Haykin and J. C. Principe, "Making sense of a complex world [chaotic events modeling]," *IEEE Signal Processing Magazine*, vol. 15, no. 3, pp. 66–81, 1998.

[22] G. C. Hegerl and H. Hoge, "Numerical simulation of the glottal flow by a model based on the compressible Navier-Stokes equations," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '91)*, vol. 1, pp. 477–480, Toronto, Ontario, Canada, May 1991.

[23] R. C. Hilborn, *Chaos and Nonlinear Dynamics: An Introduction for Scientists and Engineers*, Oxford University Press, New York, NY, USA, 1994.

[24] K. Ishizaka and J. L. Flanagan, "Synthesis of voiced sounds from a two-mass model of the vocal chords," *Bell System Technical Journal*, vol. 51, no. 6, pp. 1233–1268, 1972.

[25] J. F. Kaiser, "Some observations on vocal tract operation from a fluid flow point of view," in *Vocal Fold Physiology: Biomechanics, Acoustics, and Phonatory Control*, I. R. Titze and R. C. Scherer, Eds., pp. 358–386, Denver Center for the Performing Arts, Denver, Colo, USA, 1983.

[26] D. H. Klatt, "Software for a cascade/parallel formant synthesizer," *The Journal of the Acoustical Society of America*, vol. 67, no. 3, pp. 971–995, 1980.

[27] T. Koizumi, S. Taniguchi, and S. Hiromitsu, "Glottal source—vocal tract interaction," *The Journal of the Acoustical Society of America*, vol. 78, no. 5, pp. 1541–1547, 1985.

[28] T. Koizumi, S. Taniguchi, and S. Hiromitsu, "Two-mass models of the vocal cords for natural sounding voice synthesis," *The Journal of the Acoustical Society of America*, vol. 82, no. 4, pp. 1179–1192, 1987.

[29] I. Kokkinos and P. Maragos, "Nonlinear speech analysis using models for chaotic systems," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 6, pp. 1098–1109, 2005.

[30] G. Kubin, "Synthesis and coding of continuous speech with the nonlinear oscillator model," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '96)*, vol. 1, pp. 267–270, Atlanta, Ga, USA, May 1996.

[31] G. Kubin, "Poincaré section techniques for speech," in *Proceedings of IEEE Workshop on Speech Coding for Telecommunications Proceeding*, pp. 7–8, Pocono Manor, Pa, USA, September 1997.

[32] B. B. Mandelbrot, *The Fractal Geometry of Nature*, W. H. Freeman, New York, NY, USA, 1982.

[33] I. Mann, *An investigation of nonlinear speech synthesis and pitch modification techniques*, Ph.D. thesis, University of Edinburgh, Edinburgh, Scotland, UK, 1999.

[34] P. Maragos, "Fractal aspects of speech signals: dimension and interpolation," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '91)*, vol. 1, pp. 417–420, Toronto, Ontario, Canada, May 1991.

[35] P. Maragos, A. G. Dimakis, and I. Kokkinos, "Some advances in nonlinear speech modeling using modulations, fractals, and chaos," in *Proceedings of 14th IEEE International Conference on Digital Signal Processing (DSP '02)*, vol. 1, pp. 325–332, Santorini, Greece, July 2002.

[36] P. Maragos, J. F. Kaiser, and T. F. Quatieri, "Energy separation in signal modulations with application to speech analysis," *IEEE Transactions on Signal Processing*, vol. 41, no. 10, pp. 3024–3051, 1993.

[37] P. Maragos and A. Potamianos, "Fractal dimensions of speech sounds: computation and application to automatic speech recognition," *The Journal of the Acoustical Society of America*, vol. 105, no. 3, pp. 1925–1932, 1999.

[38] J. D. Markel and A. H. Gray, *Linear Prediction of Speech*, Springer, Berlin, Germany, 1976.

[39] R. McAulay and T. F. Quatieri, "Speech analysis/synthesis based on a sinusoidal representation," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 34, no. 4, pp. 744–754, 1986.

[40] R. S. McGowan, "An aeroacoustic approach to phonation," *The Journal of the Acoustical Society of America*, vol. 83, no. 2, pp. 696–704, 1988.

[41] E. Moulines and F. Charpentier, "Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones," *Speech Communication*, vol. 9, no. 5-6, pp. 453–467, 1990.

[42] B. Mulgrew, "Applying radial basis functions," *IEEE Signal Processing Magazine*, vol. 13, no. 2, pp. 50–65, 1996.

[43] K. Narasimhan, J. C. Principe, and D. G. Childers, "Nonlinear dynamic modeling of the voiced excitation for improved speech synthesis," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '99)*, vol. 1, pp. 389–392, Phoenix, Ariz, USA, March 1999.

[44] E. Ott, C. Grebogi, and J. A. Yorke, "Controlling chaos," *Physical Review Letters*, vol. 64, no. 11, pp. 1196–1199, 1990.

[45] J. H. Page and A. P. Breen, "The Laureate text-to-speech system-architecture and applications," *BT Technology Journal*, vol. 14, no. 1, pp. 57–67, 1996.

[46] T. S. Parker and L. O. Chua, *Practical Numerical Algorithms for Chaotic Systems*, Springer, New York, NY, USA, 1989.

[47] H.-O. Peitgen, H. Jürgens, and D. Saupe, *Chaos and Fractals: New Frontiers of Science*, Springer, New York, NY, USA, 1992.

[48] V. Pitsikalis, I. Kokkinos, and P. Maragos, "Nonlinear analysis of speech signals: generalized dimensions and lyapunov exponents," in *Proceedings of 8th European Conference on Speech Communication & Technology (EUROSPEECH '03)*, pp. 817–820, Geneva, Switzerland, September 2003.

[49] F. Plante, G. F. Meyer, and W. A. Ainsworth, "A pitch extraction reference database," in *Proceedings of 4th European Conference on Speech Communication and Technology (EUROSPEECH '95)*, vol. 1, pp. 837–840, Madrid, Spain, September 1995.

[50] A. Potamianos and P. Maragos, "Time-frequency distributions for automatic speech recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 9, no. 3, pp. 196–200, 2001.

[51] A. Potamianos and P. Maragos, "Speech formant frequency and bandwidth tracking using multiband energy demodulation," *The Journal of the Acoustical Society of America*, vol. 99, no. 6, pp. 3795–3806, 1996.

[52] A. Potamianos and P. Maragos, "Speech analysis and synthesis using an AM-FM modulation model," *Speech Communication*, vol. 28, no. 3, pp. 195–209, 1999.

[53] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge University Press, New York, NY, USA, 1992.

[54] T. F. Quatieri, *Discrete-Time Speech Signal Processing: Principles and Practice*, Prentice-Hall, Englewood Cliffs, NJ, USA, 2001.

[55] L. R. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1993.

[56] L. R. Rabiner and R. W. Schafer, *Digital Processing of Speech Signals*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1978.

[57] G. Richard, M. Liu, D. Snider, et al., "Numerical simulations of fluid flow in the vocal tract," in *Proceedings of 4th European Conference on Speech Communication and Technology (EUROSPEECH '95)*, pp. 1297–1300, Madrid, Spain, September 1995.

[58] J. Schoentgen, *Dynamic models of the glottal pulse*, Elsevier, Amsterdam, The Netherlands, 1995.

[59] J. Schoentgen, "Non-linear signal representation and its application to the modelling of the glottal waveform," *Speech Communication*, vol. 9, no. 3, pp. 189–201, 1990.

[60] J. Schoentgen, "Glottal waveform synthesis with Volterra shaping functions," *Speech Communication*, vol. 11, no. 6, pp. 499–512, 1992.

[61] J. Schoentgen and R. de Guchteneere, "An algorithm for the measurement of jitter," *Speech Communication*, vol. 10, no. 5-6, pp. 533–538, 1991.

[62] C. H. Shadle, A. Barney, and P. O. A. L. Davies, "Fluid flow in a dynamic mechanical model of the vocal folds and tract. II: implications for speech production studies," *The Journal of the Acoustical Society of America*, vol. 105, no. 1, pp. 456–466, 1999.

[63] J. Stark, "Delay embeddings for forced systems. I: deterministic forcing," *Journal of Nonlinear Science*, vol. 9, no. 3, pp. 255–332, 1999.

[64] J. Stark, D. S. Broomhead, M. E. Davies, and J. P. Huke, "Takens embedding theorems for forced and stochastic systems," in *Proceedings of 2nd World Congress of Nonlinear Analysts*, Athens, Greece, July 1996.

[65] I. Steinecke and H. Herzel, "Bifurcations in an asymmetric vocal-fold model," *The Journal of the Acoustical Society of America*, vol. 97, no. 3, pp. 1874–1884, 1995.

[66] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 15, no. 1, pp. 116–132, 1985.

[67] F. Takens, "Detecting strange attractors in turbulence," in *Proceedings of Symposium on Dynamical Systems and Turbulence*, D. A. Rand and L. S. Young, Eds., vol. 898 of *Lecture Notes in Mathematics*, pp. 366–381, Coventry, UK, 1980.

[68] D. Talkin, "Voicing epoch determination with dynamic programming," *The Journal of the Acoustical Society of America*, vol. 85, no. S1, p. S149, 1989.

[69] H. M. Teager and S. M. Teager, "Evidence for nonlinear sound production mechanisms in the vocal tract," in *Speech Production and Speech Modelling*, W. J. Hardcastle and A. Marchal, Eds., vol. 55 of *NATO Advanced Study Institute Series D*, pp. 241–261, Bonas, France, July 1989.

[70] T. J. Thomas, "A finite element model of fluid flow in the vocal tract," *Computer Speech & Language*, vol. 1, pp. 131–151, 1986.

[71] I. Tokuda, R. Tokunaga, and K. Aihara, "A simple geometrical structure underlying speech signals of the Japanese vowel /a/," *International Journal of Bifurcation and Chaos in Applied Sciences and Engineering*, vol. 6, no. 1, pp. 149–160, 1996.
[72] D. J. Tritton, *Physical Fluid Dynamics*, Oxford University Press, New York, NY, USA, 2nd edition, 1988.
[73] L. P. Šil'nikov, "A case of the existence of a denumerable set of periodic motions," *Soviet Mathematics Doklady*, vol. 6, pp. 163–166, 1965.

Steve McLaughlin: Institute for Digital Communications, School of Engineering and Electronics, University of Edinburgh, Edinburgh EH9 3JL, Scotland, UK

*Email*: sml@ee.ed.ac.uk

Petros Maragos: School of Electrical and Computer Engineering, National Technical University of Athens, Athens 157 73, Greece

*Email*: maragos@cs.ntua.gr

# 6 Communication system nonlinearities: challenges and some solutions

G. Tong Zhou, Hua Qian, and Ning Chen

## 6.1. Introduction

A power amplifier (PA) is an essential part of a transmitter, which in turn is necessary for all wireless communication systems, including wireless LAN devices, cellular phones, wireless base stations, radio and TV stations, satellites and so forth. Unfortunately, all PAs are nonlinear to a certain extent which creates nonlinear distortions [8, 17, 25].

For some basic understanding of the nonlinear phenomenon, let us examine a simple third-order nonlinear model [8, page 5],

$$\bar{y}(t) = \bar{a}_1\bar{x}(t) + \bar{a}_2\bar{x}^2(t) + \bar{a}_3\bar{x}^3(t), \tag{6.1}$$

where $\bar{x}(t)$ is real-valued input, $\bar{y}(t)$ is real-valued output, and $\{\bar{a}_i\}$ are real-valued coefficients.

Consider a two-tone input

$$\bar{x}(t) = \cos(\omega_1 t) + \cos(\omega_2 t), \tag{6.2}$$

where $\omega_1 = \omega_c + \omega_a$ and $\omega_2 = \omega_c + \omega_b$. Assume that $\omega_a < \omega_b$ (and thus $\omega_1 < \omega_2$), and that $|\omega_a|, |\omega_b| \ll \omega_c$ so $\bar{x}(t)$ can be viewed as a narrowband signal with $\omega_c$ as the carrier frequency.

Substituting (6.2) into (6.1), we can see from Table 6.1 that new frequencies beyond those already present in $\bar{x}(t)$ are created in the output—a hallmark of any nonlinear system. Since the contents around the zero frequency (DC), and the harmonics $(2\omega_c, 3\omega_c)$ are far away from the input carrier, they can be easily removed by bandpass filtering around $\omega_c$. However, the frequencies $2\omega_1 - \omega_2$ and $2\omega_2 - \omega_1$ are sufficiently close to the frequencies of interest that they may not be eliminated by filtering [17, page 2].

It is easy to see that $(2k)$th-order nonlinear terms generate frequencies that are near $0, 2\omega_c, 4\omega_c, \ldots$, up to $2k\omega_c$, which can be ignored due to the bandpass filtering. However, $(2k+1)$th-order nonlinearities always generate $(k+1)\omega_1 - k\omega_2$

Table 6.1. Frequency contents in (6.1).

| Term | Frequency content | |
|---|---|---|
| $\tilde{a}_1\tilde{x}(t)$ | Around $\omega_c$ : $\omega_1, \omega_2$ | |
| $\tilde{a}_2\tilde{x}^2(t)$ | Around DC | Around $2\omega_c$ |
| | $0, \omega_2 - \omega_1$ | $2\omega_1, 2\omega_2, \omega_1 + \omega_2$ |
| $\tilde{a}_3\tilde{x}^3(t)$ | Around $\omega_c$ | Around $3\omega_c$ |
| | $\omega_1, \omega_2,$ | $3\omega_1, 3\omega_2,$ |
| | $2\omega_1 - \omega_2, 2\omega_2 - \omega_1$ | $2\omega_1 + \omega_2, \omega_1 + 2\omega_2$ |

and $(k+1)\omega_2 - k\omega_1$ frequency terms that reside inside or near the wanted channel and thus cause concern. Nonlinearities can negatively impact communication systems performance in terms of spectral efficiency, interference to other users, bit error rate (BER), and so forth and must be carefully controlled.

RF PAs commonly employed in wireless systems have the following classical modes of operation: Class A, Class AB, Class B, Class C. Their linearity decreases, but their power efficiency increases, in the above order [8, Chapter 3] [1]. Power efficiency is defined as $\eta = P_1/P_{\text{DC}}$, where $P_1$ is the RF fundamental output power, and $P_{\text{DC}}$ is the power drawn from the DC source [8, page 52]. High efficiency leads to longer battery life and smaller battery size for the handheld device, or lower operating cost for the base station.

For a given PA, power efficiency also depends on the input level. The closer the PA is driven to saturation, the higher the power efficiency. Unfortunately, the PA is also the most nonlinear near saturation. Reducing the input power, that is, backing-off the PA, reduces nonlinear distortions, but also lowers the power efficiency. A back-off level of 10 dB means that an amplifier capable of 10 times the desired output power must be used; the power supplies, heatsinks and output devices must all be considerably larger than those designed according to the actual output power level [17, page 7].

We thus face a paradox in communication system applications—the opposing relationship between PA linearity and efficiency and the desire to achieve high levels of both.

In this chapter, we will first investigate characteristics and manifestations of nonlinear effects in communication systems in Section 6.2. A certain amount of nonlinearity is usually tolerated by the standards so quantitative analysis offered in Section 6.3 can be useful. In Section 6.4, we will attempt to algorithmically "linearize" a nonlinear PA so the resulting system can be both sufficiently linear and power efficient. Testbed measurement results will be shown to demonstrate the performance of the linearization algorithms.

## 6.2. Nonlinear communication system concepts

### 6.2.1. Passband versus baseband nonlinearities

Passband models such as (6.1) are often used in the literature, but it is sometimes more convenient to work with baseband quantities. Unfortunately, the confusion

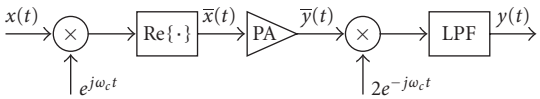



FIGURE 6.1. Block diagram of a passband system.

often arises regarding the correct form of the baseband nonlinear model [35, 45] [1, page 69] so it is helpful to elucidate here.

The block diagram in Figure 6.1 depicts the upconversion of the complex-valued baseband signal $x(t)$ to the passband, the amplification of the real-valued passband input $\bar{x}(t)$, and the down-conversion of the passband output $\bar{y}(t)$ into a complex-valued baseband signal $y(t)$. The relationship between the real-valued passband input signal $\bar{x}(t)$ and the complex-valued baseband input signal $x(t)$ is given by

$$\bar{x}(t) = \mathrm{Re}\left\{x(t)e^{j\omega_c t}\right\} \Longleftrightarrow x(t) = 2\mathrm{LPF}\left[\bar{x}(t)e^{-j\omega_c t}\right], \tag{6.3}$$

where $\omega_c$ is the carrier frequency, $\mathrm{LPF}[\cdot]$ denotes the lowpass filtering operation, and $\mathrm{Re}\{\cdot\}$ denotes the real-value part.

Consider a memoryless polynomial nonlinear PA model in the passband,

$$\bar{y}(t) = \sum_{\ell=1}^{L} \bar{a}_\ell \left[\bar{x}(t)\right]^\ell. \tag{6.4}$$

It can be shown [45] that the baseband equivalent model is

$$y(t) = x(t) \sum_{k=0}^{K} a_{2k+1} \left|x(t)\right|^{2k} \tag{6.5}$$

$$= \sum_{k=0}^{K} a_{2k+1} \left[x(t)\right]^{k+1} \left[x^*(t)\right]^k, \tag{6.6}$$

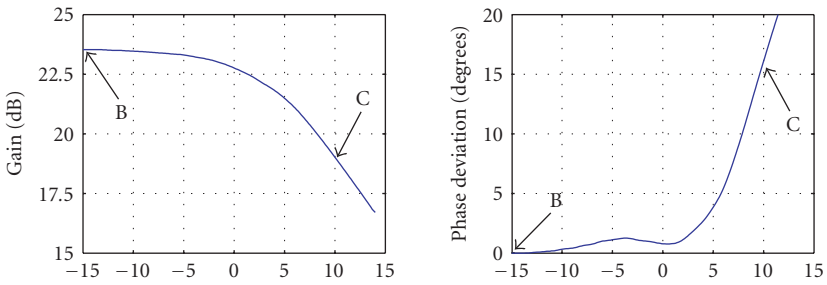
where

$$a_{2k+1} = \frac{1}{2^{2k}} \binom{2k+1}{k} \bar{a}_{2k+1}, \tag{6.7}$$

and $K = (L-1)/2$ if $L$ is odd or $K = L/2 - 1$ if $L$ is even.

Although the passband polynomial model in (6.4), or equivalently, the baseband polynomial model in (6.5) is quite simple, it has been shown to be accurate for modeling memoryless nonlinear PAs [17, Section 2.13], [44] below the saturation point. Nonlinear models with memory will be discussed later.

From this point on, we will be working with baseband quantities. We discuss next, standard characterizations of memoryless nonlinear PAs.




(a) AM/AM (b) AM/PM

FIGURE 6.2. AM/AM and AM/PM characteristics of an actual Class AB PA.

### 6.2.2. AM/AM, AM/PM characteristics

Recall that $x(t)$ is the baseband PA input and $y(t)$ is the baseband PA output. Usually, one would expect that the output amplitude $|y(t)|$ depends on both the input amplitude $|x(t)|$ and the input phase $\angle x(t)$. Interestingly however, for the so-called (quasi) memoryless[1] PAs, experimental evidence [17, page 64] shows that both the output amplitude $|y(t)|$ and the output phase deviation $\angle y(t) - \angle x(t)$ depend on the input amplitude $|x(t)|$ alone. The relationship between $|y(t)|$ and $|x(t)|$,

$$|y(t)| = f_a(|x(t)|), \tag{6.8}$$

or equivalently, the relationship between the gain function $|y(t)/x(t)|$ and $|x(t)|$, is referred to as the amplitude modulation to amplitude modulation (AM/AM) conversion. The relationship between $\angle y(t) - \angle x(t)$ and $|x(t)|$,

$$\angle y(t) - \angle x(t) = f_p(|x(t)|), \tag{6.9}$$

is referred to as the amplitude modulation to phase modulation (AM/PM) conversion [17, page 64].

A strictly memoryless PA has AM/AM conversion but no AM/PM conversion (i.e., $f_p(|x(t)|)$ is a constant). A quasimemoryless PA has both AM/AM and AM/PM conversions [35]. An ideal linear PA would have $f_a(|x(t)|) = G|x(t)|$ and $f_p(|x(t)|) = \theta$ for all $|x(t)|$ between 0 and the saturation point of the PA, where $G$ and $\theta$ are constants.

Figures 6.2(a) and 6.2(b) show the measured AM/AM and AM/PM characteristics, respectively, of an actual Class AB PA. Gain compression is seen at large input values, indicating the presence of nonlinear effects. Nonlinearity is also evident from the AM/PM curve.

[1] In a memoryless system, the output only depends on the input at the same time instant.

Nonlinearity is not a problem for constant envelope signals. Suppose that $x(t) = Ae^{j\phi_t}$ so $|x(t)| = A$ is constant. We infer from (6.8) and (6.9) that $|y(t)| = G|x(t)|$, and $\angle y(t) = \angle x(t) + \theta$ for some constants $G$ and $\theta$. Together, they give rise to

$$y(t) = |y(t)|e^{j\angle y(t)} = G|x(t)|e^{j(\theta+\angle x(t))} = (Ge^{j\theta})x(t). \tag{6.10}$$

Although the underlying PA is still nonlinear, the constant amplitude signal $x(t)$ only "sees" or "experiences" a single point on the AM/AM or AM/PM curve (e.g., point C in Figure 6.2). Thus, the PA "appears" linear to the constant amplitude signal as indicated by (6.10).

This explains why nonlinear PAs are routinely used for constant envelope signals such as continuous wave (CW), frequency modulation (FM), classical frequency-shift keying (FSK), and Gaussian minimum shift keying (GMSK) (used in Global System for Mobile Communications—GSM) without causing performance degradations [32].

### 6.2.3. Sensitivity of modern modulation formats to nonlinear effects

Denote by $x_n$ the Nyquist-rate sampled version of $x(t)$ and by $h_n$ the discrete-time equivalent pulse shaping filter. Even if the discrete-time source symbols $s_n$ have a constant amplitude $A$, the digital communication signal after pulse shaping ($*$ denotes convolution)

$$x_n = s_n * h_n = \sum_k s_k h_{n-k} \tag{6.11}$$

does not have constant amplitude any more.

The peakedness or the dynamic range of a signal can be characterized by the so-called crest factor, or the peak-to-average power ratio (PAR) which is the square of the crest factor [17, page 46]. If $s_n$ has zero mean and $|s_n| = A$, it can be shown that the worst case PAR for the above $x_n$ is

$$\frac{\left(\sum_n |h_n|\right)^2}{\sum_n |h_n|^2} \geq 1. \tag{6.12}$$

In contrast, the PAR of the constant amplitude $s_n$ is 1 (0 dB), so the PAR of $x_n$ has increased beyond that of $s_n$ due to filtering. For example, square-root-raised-cosine pulse shaping can cause the PAR to increase by 3–6 dB [32].

Orthogonal frequency division multiplexing (OFDM) is a popular transmission format that has been adopted by many standards including IEEE 802.11a, IEEE 802.11g, IEEE 802.16, HIPERLAN 2, digital audio broadcast, and digital video broadcast [16] [14, Section 1.2]. The Nyquist-rate sampled time-domain

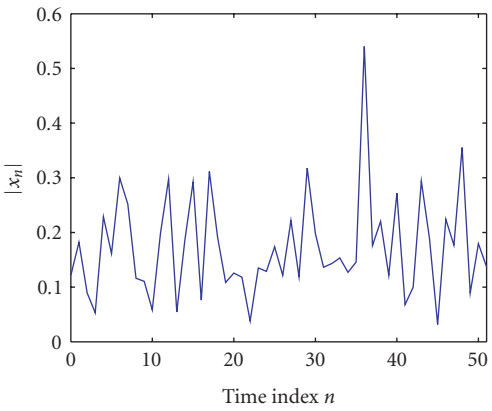



FIGURE 6.3. Amplitude of one realization of the OFDM signal with $N = 52$.

OFDM signal is given by [42]

$$x_n = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} X_k e^{j(2\pi kn/N)}, \quad 0 \le n \le N-1, \tag{6.13}$$

where $N$ is the OFDM block length, and $\{X_k\}_{k=0}^{N-1}$ is the frequency domain OFDM signal belonging to a known constellation.

If $X_k$ has constant modulus, it can be shown that the worst case PAR of the OFDM signal $x_n$ is $N$ [42]. However, worst case PAR values rarely happen. Since PAR is a random variable, an appropriate descriptor of the PAR is the complementary cumulative distribution function (CCDF), $\Pr(\text{PAR} > \gamma)$, where $\gamma$ is a given PAR threshold.

Figure 6.3 shows the amplitude of one realization of the OFDM signal $x_n$ with $N = 52$ as specified in the IEEE 802.11a standard [15]. Occasional large peaks are observed.

Figure 6.4 shows the CCDF curves for certain modern communication signals. The solid line in Figure 6.4(a) is the empirical CCDF of the PAR of the OFDM signal with $N = 52$. We infer for example that there is a 1% chance that the OFDM block will have a PAR value $\ge 9.2$ dB. The dashed line is the empirical CCDF of the PAR of the OFDM signal with $N = 256$. In the probabilistic sense, the PAR worsens as $N$ increases.

Another signal that has a large dynamic range is the forward-link (i.e., base-station to mobile) code-division multiple-access (CDMA) signal [22, 23]. Figure 6.4(b) shows the CCDF of the so-called instantaneous-to-average power ratio (IAR) [23] of a 25-channel (3 overhead channels plus 22 traffic channels) forward-link CDMA signal. We can see that 1% of the signal has IAR values in excess of 7.7 dB.

Now since $|x(t)|$ traverses a large region of the PA characteristic (e.g., region B-C in Figure 6.2), much of the inherent nonlinearity of the PA is now revealed to

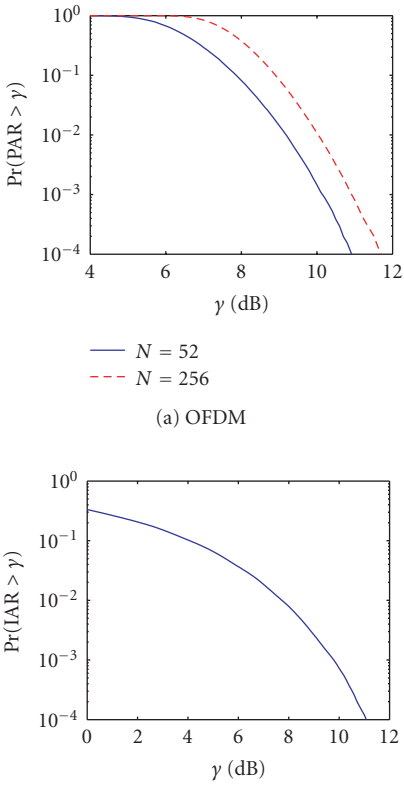



FIGURE 6.4. (a) CCDF of the PAR of OFDM signals; (b) CCDF of the IAR of a forward-link CDMA signal with 25 active channels.

the input signal. The larger the dynamic range of the signal, the more sensitive the signal is to the system nonlinearity.

### 6.2.4. Nonlinear effects with memory

Memoryless nonlinear PAs have been studied for decades. In the recent years, the problem of nonlinear PAs with memory has attracted a great deal of attention. Memory effects arise in high power amplifiers (such as those used in base station applications) and/or when wideband signals (such as wideband CDMA) are present at the PA input [3, 4, 38]. We are generally concerned with two types of memory effects: (i) electrical memory effects; (ii) thermal memory effects.

Electrical memory effects are caused by varying envelope, fundamental or second harmonic impedances at different modulation frequencies. With careful design, electrical memory effects can be limited to those caused by bias networks [43].

Thermal memory effects are caused by electro-thermal coupling, and these affect low modulation frequencies up to a few megahertz (MHz) [43].

Frequency dependent behavior of the PA can often be ignored for narrowband signals, but they must be accounted for when wideband signals are amplified.

Symptoms of the memory nonlinear effect include: (i) scattering and/or hysteresis of the AM/AM, AM/PM curves [20, 27] [9, page 99], (ii) asymmetry in the intermodulation distortion (IMD) products [5, 20]; (iii) IMD variation as a function of tone spacing in a two-tone test [20, 21, 43].

To understand the IMD behavior when memory effect is present, let us consider a baseband equivalent memoryless nonlinear system:

$$w(t) = \sum_{k} a_{2k+1}\big[x(t)\big]^{k+1}\big[x^*(t)\big]^{k}, \tag{6.14}$$

followed by a linear time-invariant (LTI) system with impulse response $h(t)$:

$$y(t) = w(t) * h(t) = \int_{-\infty}^{\infty} w(\tau)h(t-\tau)d\tau, \tag{6.15}$$

where $*$ denotes convolution. The overall system linking $x(t)$ to $y(t)$ is a nonlinear system with memory.

Suppose that a two-tone input

$$x(t) = Ae^{j\omega_a t} + Ae^{j\omega_b t} \tag{6.16}$$

is available and assume that $\omega_b > \omega_a$.

It is straightforward to show, by substituting (6.16) into (6.14), that both frequency components of $w(t)$ at

$$\begin{aligned} \mathrm{IMD}_{2k+1,L} &= (k+1)\omega_a - k\omega_b, \\ \mathrm{IMD}_{2k+1,H} &= (k+1)\omega_b - k\omega_a \end{aligned} \tag{6.17}$$

have equal amplitude $A^{2k+1}a_{2k+1}$ which does not depend on the tone spacing $\Delta\omega = \omega_b - \omega_a$. This explains empirical observations [9, Section 3.5] that IMD products from memoryless nonlinear devices appear symmetric and their magnitudes do not depend on the tone spacing.

When the signal further propagates through the LTI system in (6.15), the frequency component at $\mathrm{IMD}_{2k+1,L}$ will have amplitude

$$\mathcal{A}_{2k+1,L} = A^{2k+1}a_{2k+1}H(\omega_a - k\Delta\omega), \tag{6.18}$$

whereas the frequency component at $\mathrm{IMD}_{2k+1,H}$ will have amplitude

$$\mathcal{A}_{2k+1,H} = A^{2k+1}a_{2k+1}H\big(\omega_a + (k+1)\Delta\omega\big). \tag{6.19}$$

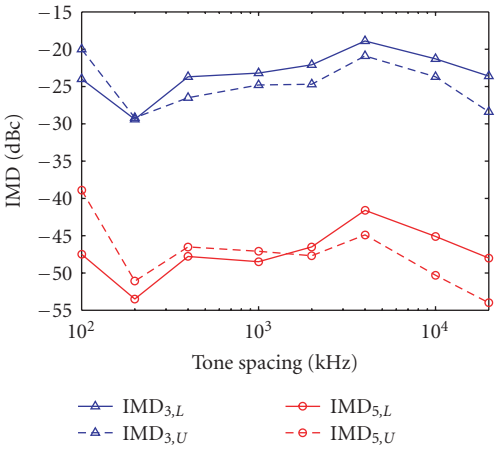



FIGURE 6.5. The IMD products versus the tone spacing for an Ericsson 45W PA.

In (6.18)-(6.19), $H(\omega)$ denotes the Fourier transform of $h(t)$, that is, $H(\omega)$ is the frequency response of the LTI system at frequency $\omega$.

From (6.18)-(6.19), we see that $|\mathcal{A}_{2k+1,L}| \neq |\mathcal{A}_{2k+1,H}|$ in general due to the frequency response of $h(t)$ at different frequencies. Moreover, the magnitudes of these IMD products vary with the tone spacing $\Delta\omega$ since $H(\omega)$ is generally nonflat.

The nonlinear system between $x(t)$ and $y(t)$ as described by (6.14)-(6.15) is a Hammerstein system [26]. However, the above described behaviors of the IMD products can also be observed if we have a general Volterra nonlinear system.

To provide an illustrative example, we conducted a two-tone experiment on an Ericsson 45 W PA (19D903800G1). In this experiment, we fixed the peak PA output power at 36.5 dBm but incrementally changed the tone spacing from 100 kHz to 20 MHz. Figure 6.5 shows the 3rd-order IMD product at the lower sideband of the main channel (IMD $_{3,L}$) and at the upper sideband of the main channel (IMD $_{3,U}$), as well as 5th-order IMD products IMD $_{5,L}$ and IMD $_{5,U}$, when the tone spacing is changed. We observe that the IMD products varied significantly with changes in the tone-spacing (please note that the vertical axis is on the dB scale), and appeared asymmetric between the lower and upper sidebands. In this example, IMD $_{3,L}$ and IMD $_{3,U}$ differed by as much as 5 dB, whereas IMD $_{5,L}$ and IMD $_{5,U}$ differed by as much as 9 dB. These results led us to believe that the Ericsson 45 W PA exhibited significant memory effects.

A $(2k+1)$th-order baseband Volterra model is given by

$$y(t) = \sum_{k} \int \cdots \int h_{2k+1}(\boldsymbol{\tau}_{2k+1}) \prod_{i=1}^{k+1} x(t-\tau_i) \prod_{i=k+2}^{2k+1} x^*(t-\tau_i)\, d\boldsymbol{\tau}_{2k+1}, \tag{6.20}$$

where $h_{2k+1}(\cdot)$ is the $(2k+1)$th-order Volterra kernel, $\boldsymbol{\tau}_{2k+1} = [\tau_1, \tau_3, \ldots, \tau_{2k+1}]^T$, and $d\boldsymbol{\tau}_{2k+1} = d\tau_1 d\tau_3 \cdots d\tau_{2k+1}$ [2]. It is a general nonlinear model with memory.

Two special cases of the Volterra model are worth mentioning. The Wiener model is a linear time-invariant block followed by a memoryless nonlinearity. In the memory polynomial model, the Volterra kernels are zero everywhere except along the main diagonals. Both the Wiener model [7] and the memory polynomial model [19] have been shown to be appropriate for certain nonlinear PAs with memory.

Time-delayed neural networks [24] and a frequency-dependent Saleh model [17, page 79] are other notable alternatives for modeling nonlinear PAs with memory effects.

Next, we will describe and quantify distortions encountered in nonlinear communication systems.

## 6.3. Nonlinear distortions

Digitally modulated signals occupy a continuum of frequencies within a given bandwidth $B_x$. Suppose in (6.16) that $-B_x/2 < \omega_a < \omega_b < B_x/2$ so both $\omega_a$ and $\omega_b$ appear in band. As we discussed earlier, a $(2k+1)$th-order nonlinearity generates IMD products in the form of $(k+1)\omega_a - k\omega_b$ that are of concern. These IMD products can appear in band or out of band. For example, if $\omega_a = B_x/4$, $\omega_b = B_x/3$, then $\text{IMD}_3$ at $2\omega_a - \omega_b = B_x/6$ resides in band, whereas if $\omega_a = -B_x/4$, $\omega_b = B_x/3$, then $2\omega_a - \omega_b = -5B_x/6$ appears in the adjacent channel.

Since $B_x$ is much smaller than the carrier frequency, even if the IMD products appear out of band, they may still be too close to the main channel to be removed by filtering. Certainly, those in band IMD products cannot be eliminated using filters.

In-band IMD components degrade the BER. The so-called error vector magnitude (EVM) is often measured based on which inference on the BER can be indirectly made. EVM measures the normalized difference (expressed as a percentage) between the reference waveform and the measured waveform [29]. Typical figures are in the range of 5%–15% for most mobile radio systems [17, pages 210–211].

Out of band IMDs cause adjacent channel interference which is measured by the adjacent channel power ratio (ACPR) [6, 17]

$$\text{ACPR} = \frac{\int_{f_o-0.5B_{\text{adj}}}^{f_o+0.5B_{\text{adj}}} S(f)df}{\int_{-0.5B_{\text{ch}}}^{0.5B_{\text{ch}}} S(f)df}. \tag{6.21}$$

The numerator in (6.21) is the total interference power in a specified adjacent channel bandwidth $B_{\text{adj}}$, at a given frequency offset $f_o$ from the carrier frequency. The denominator in (6.21) is the total main channel power in the specified channel bandwidth $B_{\text{ch}}$ [18].

As an example, we show the spectral emission limits for wideband CDMA (WCDMA) in Table 6.2 and the corresponding spectral mask[2] in Figure 6.6.

[2]The shape of the spectral mask (including the "notch") in Figure 6.6 is specific to WCDMA only.

Table 6.2. Spectrum emission limits [28].

| $\lvert\Delta f\rvert$ (MHz) | Minimum requirement | Measurement bandwidth |
|---|---|---|
| 2.5 – 3.5 | $-35 - 15 \cdot (\lvert\Delta f\rvert - 2.5)$ (dBc) | 30 kHz |
| 3.5 – 7.5 | $-35 - 1 \cdot (\lvert\Delta f\rvert - 3.5)$ (dBc) | 1 MHz |
| 7.5 – 8.5 | $-39 - 10 \cdot (\lvert\Delta f\rvert - 7.5)$ (dBc) | 1 MHz |
| 8.5 – 12.5 | −49 dBc | 1 MHz |

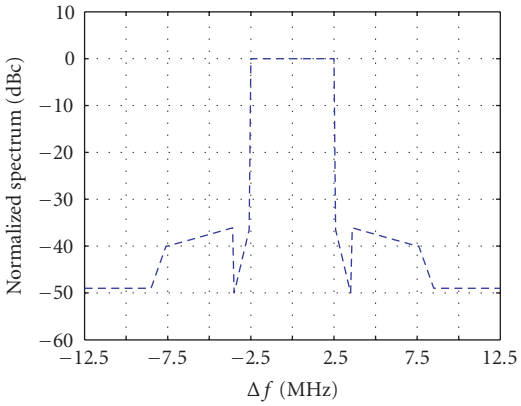



Figure 6.6. Spectral mask for WCDMA.

### 6.3.1. Spectral regrowth prediction

It would be useful to be able to predict the spectrum of the signal as it goes through a nonlinear PA. Elegant closed-form solutions have been found [46] for the "simple" case of wide-sense stationary Gaussian input and polynomial nonlinear PA.

When the input is non-Gaussian, one can resort to cumulants [44] to analyze the output autocovariance function, whose Fourier transform yields the power spectral density (PSD). However, the combination of a non-Gaussian process and a nonlinear system means that the complexity of the analysis can quickly get out of hand.

A digital communication signal can be represented as

$$x(t) = \sum_m s_m h(t - mT), \tag{6.22}$$

where $s_m$ is the $m$th symbol, $h(t)$ is the pulse shaping filter, and $T$ is the symbol period. Thus, $x(t)$ is strict-sense cyclostationary. Interestingly, it is shown in [46] that when the bandwidth of the pulse shaping filter is limited to $1/T$ (i.e., $h(t)$ has no excess bandwidth), $x(t)$ in (6.22) is wide-sense stationary, meaning that its autocovariance function does not depend on the time variable $t$. Furthermore,

when $x(t)$ is Gaussian, as is (approximately) the case with OFDM and forward-link CDMA signals, the output signal PSD can be accurately predicted as described below.

Theorem 6.1 (see [46]). *Assume that $x(t)$ is stationary, zero-mean, and circular complex Gaussian distributed. If the output $y(t)$ is related to the input $x(t)$ through* (6.6)*, then the autocovariance function of $y(t)$, defined as $c_{2y}(\tau) = E[y^*(t)y(t+\tau)]$, is related to that of $x(t)$ through*

$$c_{2y}(\tau) = \sum_{m=0}^{K} \alpha_{2m+1} \left| c_{2x}(\tau) \right|^{2m} c_{2x}(\tau), \tag{6.23}$$

*where the constant coefficient*

$$\alpha_{2m+1} = \frac{1}{m+1} \left| \sum_{k=m}^{K} a_{2k+1} \binom{k}{m} (k+1)! \left[c_{2x}(0)\right]^{k-m} \right|^2, \qquad \binom{k}{m} = \frac{k!}{m!(k-m)!}. \tag{6.24}$$

*The PSD of $y(t)$, defined as $S_{2y}(f) = \int_{-\infty}^{\infty} c_{2y}(\tau) e^{-j2\pi f\tau} d\tau$, is related to that of $x(t)$ through*

$$S_{2y}(f) = \sum_{m=0}^{K} \alpha_{2m+1} \underbrace{S_{2x}(f) * \cdots * S_{2x}(f)}_{m+1} * \underbrace{S_{2x}(-f) * \cdots * S_{2x}(-f)}_{m}, \tag{6.25}$$

*where $*$ denotes convolution.*

The above theorem subsumes the specific-order nonlinearity results published in [13, 40, 44].

We infer from (6.25) that the bandwidth of $y(t)$ is $2K+1$ times that of $x(t)$ due to spectral convolutions. The resulting spectral expansion causes adjacent channel interference which is tightly controlled by the regulatory bodies.

As an illustration of the results obtained in Theorem 6.1, we show in Figure 6.7, $S_{2x}(f)$ of an OFDM signal as dashed lines. The predicted output PSD $S_{2y}(f)$, as the result of passing through a 9th-order nonlinear PA model whose AM/AM, AM/PM characteristics closely match those depicted in Figure 6.2, is shown as solid lines in Figure 6.7. Simulation results [46] confirm that the formulas given in Theorem 6.1 are accurate.

When excess bandwidth is present in $x(t)$, spectral regrowth analysis becomes more complicated but preliminary results are available in [37].

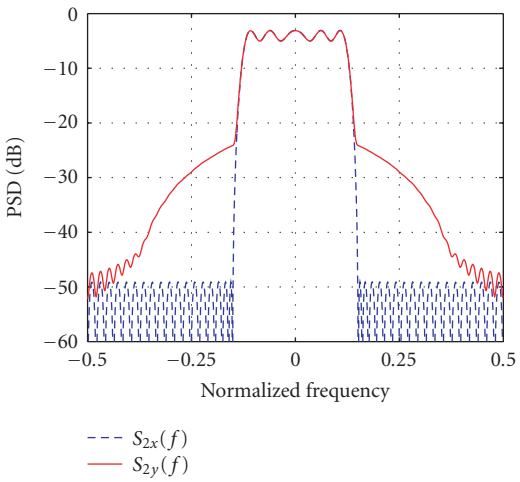



FIGURE 6.7. PSDs of the input and the output of the PA.

### 6.3.2. Signal-to-noise-and-distortion ratio

So far, we have analyzed nonlinear distortions in the frequency domain in the form of IMD or spectral regrowth. Next, we consider a completely different metric, the so-called signal-to-noise-and-distortion ratio (SNDR) [30, 31]. The SNDR concept is similar to the signal-to-noise ratio (SNR). The hope is to gain insight about the BER through the SNDR in the context of nonlinear systems.

Let us first examine a simple, third-order baseband nonlinearity

$$y(t) = a_1 x(t) + a_3 x^2(t) x^*(t) + v(t), \tag{6.26}$$

where $v(t)$ is zero-mean additive noise and is assumed to be independent of the signal $x(t)$.

Since we are interested in linear amplification of $x(t)$, we can regard the third-order term, $b(t) = a_3 x^2(t) x^*(t)$, as a form of distortion.

If $x(t)$ is zero-mean circular complex Gaussian distributed, that is, $E[x^2(t)] = 0$, $E[|x(t)|^2] = \sigma_x^2$, we can find an equivalent representation of (6.26),

$$y(t) = \mu x(t) + d(t) + v(t), \tag{6.27}$$

$$\mu = a_1 + 2a_3\sigma_x^2, \tag{6.28}$$

$$d(t) = a_3 x(t)\left[\,|x(t)|^2 - 2\sigma_x^2\right]. \tag{6.29}$$

In (6.27), we can view $\mu x(t)$ as the useful signal and $d(t)$ as the distortion. The advantage of the decomposition in (6.27) is that $d(t)$ can be shown to be uncorrelated with $x(t)$, that is, $E[x^*(t)d(t)] = 0$. In contrast, the $b(t)$ term defined earlier is correlated with $x(t)$, that is, $E[x^*(t)b(t)] \neq 0$.

Because $d(t)$ and $v(t)$ are both uncorrelated with $x(t)$, we can lump their energies together and define the SNDR as [30]

$$\text{SNDR} = \frac{|\mu|^2\sigma_x^2}{\varepsilon_d + \sigma_v^2}, \quad \varepsilon_d = E\Big[\,|\,d(t)\,|^2\Big], \tag{6.30}$$

and use it to assess the condition of the nonlinear system.

Generalizing, we consider the nonlinear system

$$y(t) = h\big(x(t)\big) + v(t), \tag{6.31}$$

where $h(\cdot)$ is a memoryless nonlinear mapping. We can always decompose (6.31) into

$$y(t) = \mu x(t) + d(t) + v(t), \tag{6.32}$$

$$\mu = \frac{E[x^* h(x)]}{\sigma_x^2}, \tag{6.33}$$

$$d(t) = h\big(x(t)\big) - \mu x(t). \tag{6.34}$$

The above decomposition ensures that $d(t)$ is uncorrelated with $x(t)$. The value of $\mu$ and thus $d(t)$ depend on $h(\cdot)$ and the probability density function (PDF) of $x(t)$.

Substituting (6.33)-(6.34) into (6.30), the SNDR expression becomes

$$\text{SNDR} = \frac{|\,E[x^* h(x)]\,|^2/\sigma_x^2}{E\Big[\,|\,h(x)\,|^2\Big] - |\,E[x^* h(x)]\,|^2/\sigma_x^2 + \sigma_v^2}. \tag{6.35}$$

It is therefore possible to analyze the SNDR for a given nonlinearity and the input signal PDF and make inference on the system performance.

We are interested in peak amplitude limited nonlinearities since physical devices (such as PAs) are almost always peak amplitude (or peak power) limited. Suppose that $|h(\cdot)| \le A$: this amplitude constraint itself implies that $h(\cdot)$ is nonlinear. It will be easier to work with normalized quantities so let us set $\gamma = |x|/\sigma_x$ and define $g(\gamma) = |h(x)|/A$ which implies $0 \le g(\cdot) \le 1$.

Some example $g(\cdot)$ functions are shown in Figure 6.8. An interesting question to ask is, what kind of $g(\cdot)$ gives rise to the highest SNDR? The answer lies in the following theorem.

Theorem 6.2 (see [34]). *Within the class of $g(\cdot)$ satisfying $0 \le g(\cdot) \le 1$, the following $g(\cdot)$ maximizes the SNDR expression in* (6.35)*:*

$$g(\gamma) = \begin{cases} \dfrac{\gamma}{\eta^\star}, & 0 \le \gamma < \eta^\star, \\ 1, & \gamma \ge \eta^\star, \end{cases} \tag{6.36}$$

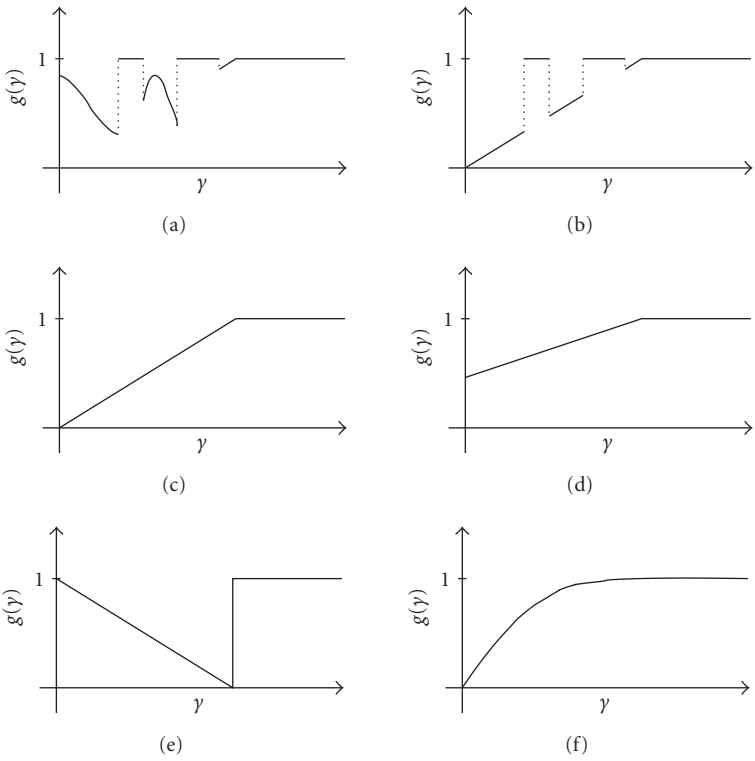



FIGURE 6.8. Nonlinear mappings $g(\cdot)$ that satisfy the $0 \le g(\cdot) \le 1$ constraint.

*where the threshold* $\eta^\star$ *is found from*[3] $\eta^\star = T^{-1}(A^2/\sigma_v^2)$, *with*

$$\begin{aligned} T(\eta) &= \frac{\eta}{C_1(\eta) - \eta C_0(\eta)}, \\ C_0(\eta) &= \int_\eta^\infty p(\gamma)d\gamma, \\ C_1(\eta) &= \int_\eta^\infty \gamma p(\gamma)d\gamma, \end{aligned} \tag{6.37}$$

*and* $p(\gamma)$ *is the PDF of* $\gamma$.

Theorem 6.2 establishes that the nonlinearity in the shape of Figure 6.8(c) is optimal.

If a given system nonlinearity $u(\cdot)$ is undesirable and it is possible to apply a predistortion mapping $f(\cdot)$, then according to Theorem 6.2, it is best to make $u(f(\cdot))$ equal to the $g(\cdot)$ function given in (6.36) (assume that $u(\cdot)$ is normalized

[3]We can show that $T(\cdot)$ is a monotonically increasing function and thus its inverse $T^{-1}(\cdot)$ exists.

to have a maximum amplitude of 1). This is the well-known linearization strategy [17]. However, what was little understood before, was the selection of the threshold $\eta$, or equivalently, the gain factor $1/\eta$. Theorem 6.2 says that the optimal (in terms of SNDR) $\eta^\star$ depends on the PDF of $\gamma$ and the peak signal-to-noise ratio PSNR $= A^2/\sigma_v^2$. This naturally leads us to the next topic, PA linearization.

## 6.4. Digital baseband predistortion linearization

As we explained earlier, nonconstant envelope modulation schemes are sensitive to the PA nonlinearity. In these applications, PA linearization is often pursued to limit nonlinear distortions and to improve the efficiency of the PA. For mobile terminals, increased efficiency means reduced battery drain, reduced battery size and weight, and increased battery life. Power efficiency is also of prime importance for base station applications for the purposes of reducing the equipment cost, size, and network operating costs. According to a study described in [17, page 13], application of PA linearization technologies is projected to yield annual power savings of 164 million kilowatt hours for a surveyed network consisting of approximately 10,000 base sites.

PA linearization has been investigated for decades. Available techniques include feedback, feedforward, predistortion approaches; see [9, 17, 41] for comprehensive reviews of various linearization methods. The objective here is to discuss recent developments that go beyond the established methods.

### 6.4.1. Baseband predistortion

We are interested in predistortion techniques as they offer good compromise between complexity, cost, and linearization performance. Figure 6.9 shows the block diagram[4] of the 2nd/3rd generation base transceiver station functionality. Predistortion is now considered an essential element of the baseband processing unit.

The concept of predistortion can be explained using Figures 6.10 and 6.11. When a varying signal envelope passes through a nonlinear PA (shown in Figures 6.10, 6.11 are baseband equivalent PAs), distortions occur to the signal waveform especially at larger amplitudes due to gain compression. In predistortion, we first pass the signal through a nonlinear block that is complementary to the PA response. The signal is distorted, but after subsequent distortion by the PA, a linearly amplified version of the original signal can be obtained.

Predistortion can be implemented at RF, IF, or baseband, and can employ analog components or digital techniques. We are interested in digital baseband predistortion (as pictured in Figure 6.9), since digital baseband processing is flexible and it performs well.

With "direct" inversion techniques, one first selects a PA model, estimates its parameters, and then constructs the predistorter (PD) as an (approximate) inverse of the PA response. The PA can be described by a variety of models such as the

[4] See, for example, http://www.ti.com.

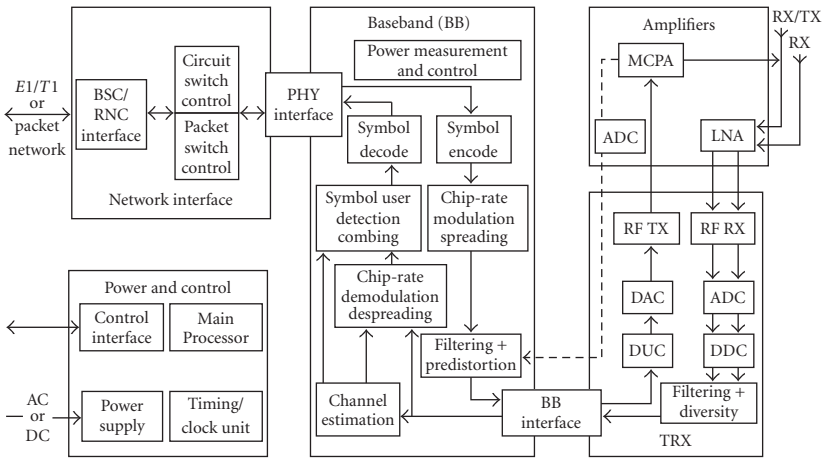



FIGURE 6.9. 2nd/3rd generation base transceiver station functionality.

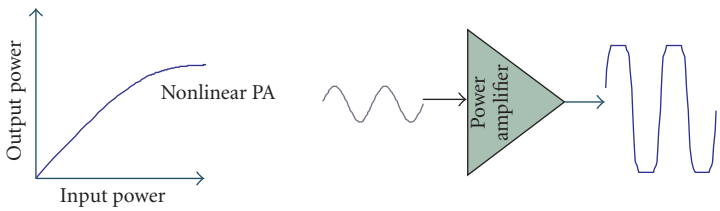



FIGURE 6.10. Baseband equivalent PA characteristic.

Saleh model [38], [17, page 79], the Volterra (or polynomial) model [26, 39] and so forth. For the latter, the so-called $p$th-order inverse [39, Chapter 7], [26, Chapter 9] has been well developed. Under the $p$th-order inverse framework, we try to find a Volterra (or polynomial) model for a given Volterra (or polynomial) model, so the concatenation of the two has a (dominant) linear term, skips a few low-order nonlinear terms, but faces some high-order nonlinear terms that, hopefully, have negligible power.

For a simple illustration of the concept, let us consider a 3rd-order polynomial PA model:

$$y(t) = b_1\, z(t) + b_3\, z^2(t) z^*(t), \tag{6.38}$$

and a 3rd-order polynomial PD model:

$$z(t) = x(t) + a_3\, x^2(t) x^*(t). \tag{6.39}$$

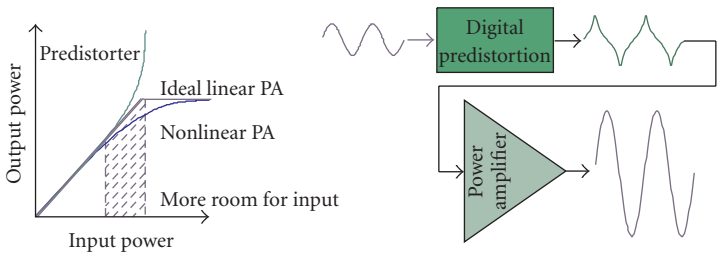



FIGURE 6.11. Baseband equivalent PD and PA characteristics.

Substituting (6.39) into (6.38), we find the input/output relationship for the PD-PA concatenation:

$$\begin{aligned} y(t) &= b_1 x(t) + (a_3 b_1 + b_3) x^2(t) x^*(t) + (2a_3 + a_3^*) b_3 x^3(t) [x^*(t)]^2 \\ &\quad + a_3 b_3 (a_3 + 2a_3^*) x^4(t) [x^*(t)]^3 + a_3^2 a_3^* b_3 \, x^5(t) [x^*(t)]^4. \end{aligned} \tag{6.40}$$

If we set $a_3 = -b_3/b_1$, then the 3rd-order nonlinear term vanishes from the $y(t)$ in (6.40). Although concatenation of two 3rd-order nonlinearities produces a 9th-order nonlinearity, the hope is that the 5th-, 7th-, and 9th-order nonlinear terms in (6.40) have diminishing powers, and thus do not contribute significantly to the overall system.

The challenges of such "direct" approaches are: (i) the PA model may not be accurate; (ii) the PA model parameters may not be straightforward to extract; (iii) the inverse of the nonlinear model may be difficult to find.

### 6.4.2. Indirect learning architecture

In the direct approaches described above, finding the PA model is only a means to the end, since the ultimate goal is to construct the PD. It makes sense then to go directly after the PD. We can possibly achieve this using the indirect learning architecture [12] described in Figure 6.12. The baseband predistorter input is denoted by $x(t)$, the predistorter output/baseband PA input is denoted by $z(t)$, and the baseband PA output is denoted by $y(t)$. The feedback path labeled "Predistorter Training (A)" has $y(t)/G$ as its input, where $G$ is the intended gain of the PA, and $\hat{z}(t)$ as its output. The actual predistorter (copy of A) is an exact copy of the predistorter training branch. Since when $y(t) = Gx(t)$, the error $e(t) = z(t) - \hat{z}(t)$ is 0, the predistorter parameters can be found by minimizing $\|e(t)\|^2$.

Suppose that the PD is modeled as $z(t) = f(x(t))$. The following steps describe the iterative process of finding $f(\cdot)$ using the indirect learning architecture. We use $\{t_n\}_{n=1}^N$ to denote the sampling instances (which do not have to be uniformly spaced); $N$ is the total number of samples.

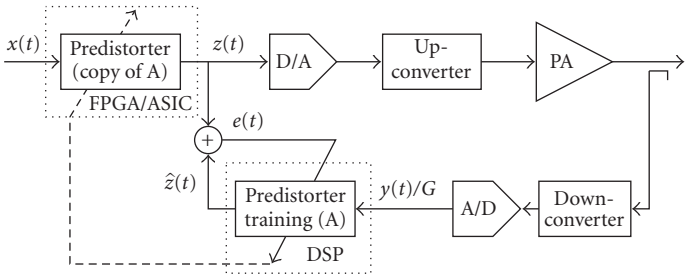



FIGURE 6.12. The indirect learning architecture for the predistorter.

*Step* 1. Initialize the predistorter model with $f^{(0)}(x(t_n)) = x(t_n)$ and set the iteration number $i = 1$.

*Step* 2. For a given block of input data $\{x^{(i)}(t_n)\}_{n=1}^{N}$, calculate the predistorted signal

$$z^{(i)}(t_n) = f^{(i-1)}\left(x^{(i)}(t_n)\right), \tag{6.41}$$

and measure the corresponding PA output $\{y^{(i)}(t_n)\}_{n=1}^{N}$.

*Step* 3. Solve the parameters in $f^{(i)}(\cdot)$ by treating $y^{(i)}(t_n)/G$ as its input and $z^{(i)}(t_n)$ as its output.

*Step* 4. Increase $i$ by one and go back to Step 2.

This iterative procedure ensures that the predistorter can adapt to slow changes in the PA characteristics due to biasing changes, ambient temperature variations, aging, and so forth.

If training is performed on-the-fly, $\{x^{(i)}(t_n)\}_{n=1}^{N}$ will be different from block to block. If training is performed off-line, $\{x^{(i)}(t_n)\}_{n=1}^{N}$ can be the same data—this will be helpful for speeding up the convergence.

It is advantageous to model the PD using basis functions. Let us write

$$z(t) = \sum_{k=1}^{K} c_k \phi_k(x(t)), \tag{6.42}$$

where $\{\phi_k(\cdot)\}_{k=1}^{K}$ are a set of $K$ known basis functions. In Step 3 of the indirect learning algorithm, we replace $x(t)$ by the measured $y(t)/G$ and solve for the $\{c_k\}_{k=1}^{K}$ coefficients using the linear least squares (LS) method described below. Such a simple solution is possible since $z(t)$ is linear in the unknowns $\{c_k\}_{k=1}^{K}$.

Let us define $N$-by-1 data vectors

$$\mathbf{y} = [y(t_1), \ldots, y(t_N)]^T, \qquad \mathbf{z} = [z(t_1), \ldots, z(t_N)]^T, \tag{6.43}$$

$N$-by-1 regressor vector

$$\boldsymbol{\phi}_k(\mathbf{y}/G) = [\phi_k(y(t_1)/G), \ldots, \phi_k(y(t_N)/G)]^T, \tag{6.44}$$

$N$-by-$K$ regressor matrix

$$\boldsymbol{\Phi} = \left[\boldsymbol{\phi}_1(\mathbf{y}/G) \quad \boldsymbol{\phi}_2(\mathbf{y}/G) \quad \cdots \quad \boldsymbol{\phi}_K(\mathbf{y}/G)\right], \tag{6.45}$$

and the $K$-by-1 parameter vector

$$\mathbf{c} = [c_1, c_2, \ldots, c_K]^T. \tag{6.46}$$

When the PD has converged to the true inverse of the PA, we will have

$$\mathbf{z} = \boldsymbol{\Phi}\mathbf{c}. \tag{6.47}$$

The LS solution for $\mathbf{c}$ is

$$\mathbf{c}_{\mathrm{LS}} = (\boldsymbol{\Phi}^H\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^H\mathbf{z}, \tag{6.48}$$

where $[\cdot]^H$ denotes the Hermitian transpose.

### 6.4.3. Orthogonal polynomials

A straightforward set of basis functions for (6.42) is the polynomial functions

$$\phi_k(x(t)) = |x(t)|^{k-1}x(t). \tag{6.49}$$

It is shown in [10] that by including even $k$ terms in (6.42), predistortion performance can be improved. Note that although the odd-order only baseband PA model is well justified, the PD model can still contain even-order nonlinear terms.

As pointed out in [33, 36], a numerical instability problem can be encountered when we invert the $K$-by-$K$ matrix $\boldsymbol{\Phi}^H\boldsymbol{\Phi}$ in (6.48). This is particularly the case if high $K$ values are used and/or quantization errors are present in the data.

To alleviate the numerical instability problem, we can replace $\phi_k(\cdot)$ by the following set of orthogonal polynomial basis functions:

$$\psi_k(x) = \sum_{l=1}^{k}(-1)^{l+k}\frac{(k+l)!}{(l-1)!(l+1)!(k-l)!}|x|^{l-1}x. \tag{6.50}$$

Table 6.3. Orthogonal polynomial basis functions $\psi_k(x)$ for $1 \le k \le 5$.

| |
|---|
| $\psi_1(x) = x$ |
| $\psi_2(x) = 4\lvert x\rvert x - 3x$ |
| $\psi_3(x) = 15\lvert x\rvert^2 x - 20\lvert x\rvert x + 6x$ |
| $\psi_4(x) = 56\lvert x\rvert^3 x - 105\lvert x\rvert^2 x + 60\lvert x\rvert x - 10x$ |
| $\psi_5(x) = 210\lvert x\rvert^4 x - 504\lvert x\rvert^3 x + 420\lvert x\rvert^2 x - 140\lvert x\rvert x + 15x$ |

These functions are orthogonal in the sense that

$$E\left[\psi_k^*(x)\psi_l(x)\right] = 0, \quad \forall k \neq l, \tag{6.51}$$

when $|x|$ is uniformly distributed in $[0, 1]$. The first five orthogonal polynomial basis functions are listed in Table 6.3.

Although in reality, the uniform amplitude distribution assumption does not hold for communication signals, the above basis functions can still serve to lower the condition number of the $\mathbf{\Psi}^H\mathbf{\Psi}$ matrix ($\mathbf{\Psi}$ is obtained similarly to $\mathbf{\Phi}$, with $\psi_k$ replacing $\phi_k$ as the basis function), which is necessary for solving for the PD parameters $\{\beta_k\}_{k=1}^K$ as in

$$z(t) = \sum_{k=1}^{K} \beta_k \psi_k(x(t)). \tag{6.52}$$

In practice, we do not require $|x|$ to be exactly in $[0, 1]$ in order for the orthogonal polynomial basis function $\psi_k(x)$ to be used. Details of the scaling operation can be found in [33].

### 6.4.4. Memory polynomial predistorter

Although the Volterra series is a general nonlinear model with memory [26, 39], its complexity is often prohibitive for real-time PD implementations. We resort to the memory polynomial model which has been shown to be a robust preinverse for a variety of nonlinear systems with memory [11]. The discrete-time memory polynomial PD model is given by

$$z(n) = \sum_{k=1}^{K} \sum_{q=0}^{Q} a_{kq} \left| x(n-q) \right|^{k-1} x(n-q), \tag{6.53}$$

where $K$ is the highest polynomial order and $Q$ is the largest delay tap. To improve modeling accuracy, both even- and odd-order terms are included in (6.53) [10].

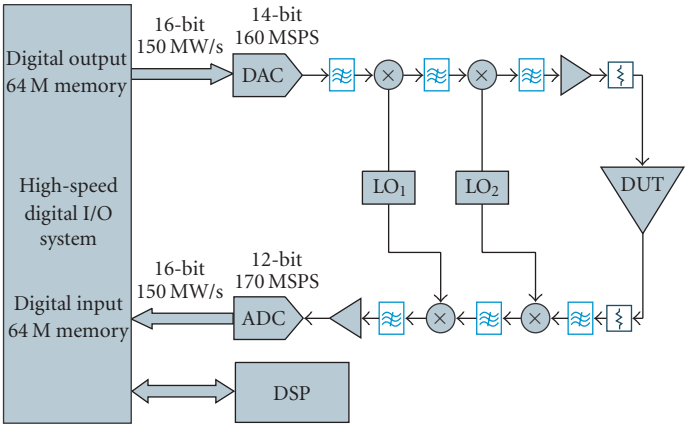



FIGURE 6.13. Schematic of the testbed.

Matrix inversion will be needed when we solve for the PD parameters $\{a_{kq}\}$ using LS. For better numerical stability, we suggest the use of the orthogonal polynomial basis in (6.50), that is,

$$z(n) = \sum_{k=1}^{K} \sum_{q=0}^{Q} \alpha_{kq} \psi_k(x(n-q)), \tag{6.54}$$

and solve for the parameters $\{\alpha_{kq}\}$.

### 6.4.5. Testbed and measurement results

We have carried out experiments to demonstrate the performance of our model-based predistortion algorithms.

We show in Figure 6.13 the configuration of our testbed. The high-speed digital I/O system has a 150 million samples per second (MSPS) 16-bit digital input/output capability. In the transmission mode, the digital I/O system first generates digital baseband data $x(t_n)$, predistorts it to yield $z(t_n)$, digitally upconverts $z(t_n)$ to an intermediate frequency (IF) of 30 MHz, and then sends out the 14-bit data stream to the digital-to-analog (D/A) converter at a sampling rate of 120 MSPS. In the acquisition mode, the digital I/O system acquires 12-bit digital IF data at the sampling rate of 120 MSPS from the analog-to-digital (A/D) converter. In the figure, $y(t_n)$ is obtained by converting the PA output to baseband and removing the time delay $\tau$ between the input and the output of the digital I/O system. Since the signal is modulated in the digital domain, any in-phase and quadrature imbalance problem in the quadrature modulator is obviated. Superheterodyne up-conversion and down-conversion chains are adopted to convert the digital IF signal to and from the radio frequency (RF). We have designed the RF

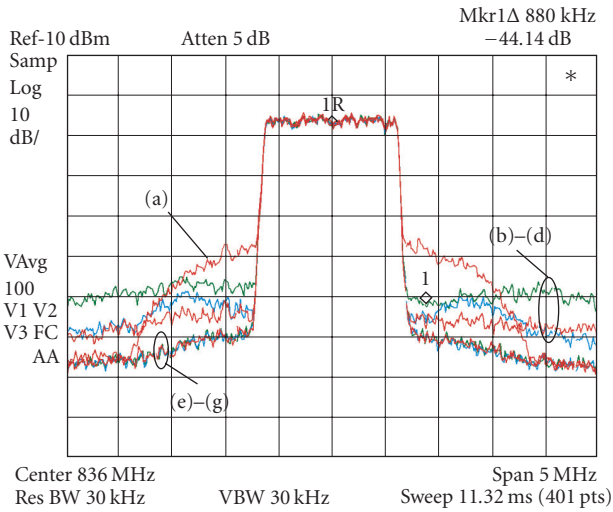



FIGURE 6.14. Measured PA output PSDs for a Siemens 1 W PA: (a) without predistortion; (b)–(d) with conventional memory polynomial predistortion at iteration numbers 3, 4, and 5; (e)–(g) with orthogonal memory polynomial predistortion at iteration numbers 3, 4, and 5. Both the conventional and the orthogonal polynomial predistorters used $K = 5$ and $Q = 4$.

transmit and receive chains to have a linear response over a wide bandwidth and a large dynamic range.

First, we would like to compare the performance of the conventional memory polynomial predistorter (6.53) with that of the orthogonal memory polynomial predistorter (6.54). In this experiment, the device under test (DUT) was a Siemens 1 W PA. The input signal was a 1.25 MHz bandwidth CDMA signal with a PAR of 6 dB. The peak envelope power (PEP) of the PA output was 28 dBm.

In Figure 6.14, line (a) is the PA output PSD without predistortion; lines (b)–(d) are the PA output PSDs with conventional memory polynomial predistortion at iteration numbers 3, 4, and 5; lines (e)–(g) are the PA output PSDs with orthogonal memory polynomial predistortion at iteration numbers 3, 4, and 5. In this experiment, we observed that the conventional memory polynomial predistorter performance was not stable (the lines fluctuated up and down). In contrast, the orthogonal memory polynomial predistorter showed stability and effectiveness; it successfully suppressed the IMD in the adjacent channel by 22 dB. In the remaining experiments, the orthogonal polynomial basis (6.50) is adopted.

Next, we compared the performances of the memoryless and memory polynomial predistorters. We first used the Siemens 1 W PA as the DUT. The input signal was a 1.2 MHz bandwidth 8-tone signal with 150 kHz tone spacing. The PEP of the PA output was 28 dBm.

Figure 6.15 shows the performance of the memory polynomial predistorter and the memoryless polynomial predistorter in terms of PSD of the PA output. In Figure 6.15, line (a) is the PA output PSD with memory polynomial predistortion; line (b) is the PA output PSD with memoryless polynomial predistortion; line

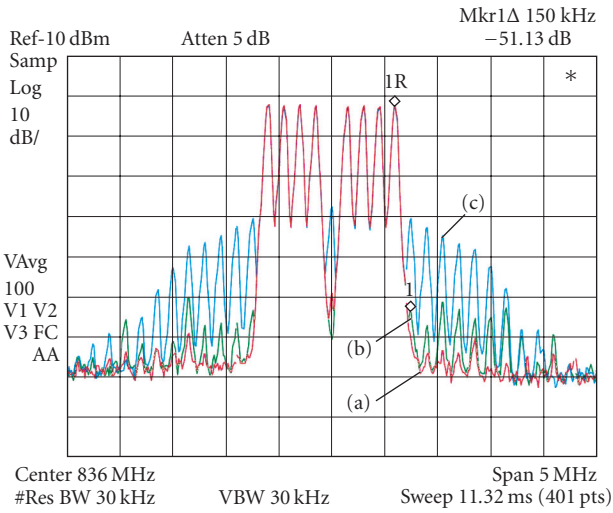



FIGURE 6.15. Measured PA output PSD for a Siemens 1 W PA: (a) with the $K = 5$, $Q = 9$ memory polynomial predistorter; (b) with the $K = 5$ memoryless predistorter; (c) without predistortion.

(c) is the PA output PSD without predistortion. In this experiment, we observed that the memory polynomial predistorter suppressed almost all the spectral regrowth. In the adjacent channel, the memory polynomial predistorter suppressed the nearest IMD product by 33 dB, while the memoryless polynomial predistorter gave 23 dB of IMD reduction. The memory polynomial predistorter outperformed the memoryless predistorter by about 10 dB.

Next, we used the Ericsson 45 W PA as the DUT. The input signal was a 2.5 MHz bandwidth 2-carrier CDMA signal with a PAR of 8.5 dB. The PEP of the PA output was 36.5 dBm.

Figure 6.16 shows the performances of the memory polynomial predistorter and the memoryless polynomial predistorter in terms of PSD of the PA output. In Figure 6.16, line (a) is the PA output PSD with memory polynomial predistortion; line (b) is the PA output PSD with memoryless polynomial predistortion; line (c) is the PA output PSD without predistortion. In this experiment, we observed that the memory polynomial predistorter suppressed the IMD in the adjacent channel by 15 dB, whereas the memoryless polynomial predistorter only gave 8 dB of IMD reduction. The results showed that the memory polynomial predistorter had a significant advantage over the memoryless polynomial predistorter for the PA with memory effects.

## 6.5. Summary

The PA is an important element in wireless transmission systems and is inherently nonlinear. Nonconstant envelope signals are particularly sensitive to the nonlinear

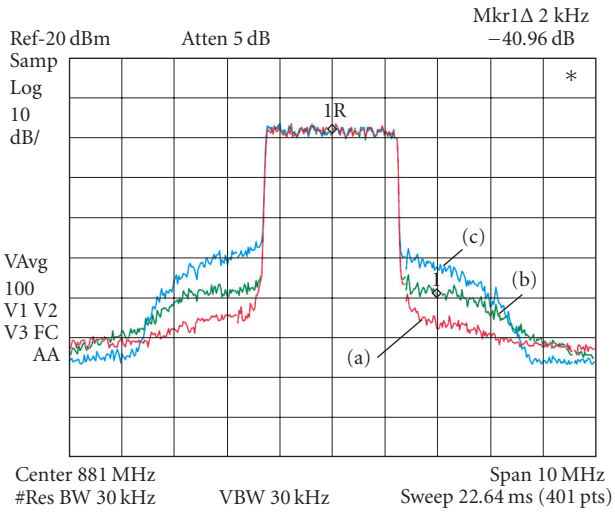



FIGURE 6.16. Measured PA output PSD for an Ericsson 45 W PA: (a) with the $K = 5$, $Q = 4$ memory polynomial predistorter; (b) with the $K = 5$ memoryless predistorter; (c) without predistortion.

effects in the PA. We describe here characteristics of the PA and prevailing black-box types of modeling approaches. Several nonlinear distortion descriptors—IMD, spectral regrowth (ACPR), SNDR, are introduced and analyzed for polynomial types of nonlinearities. PA linearization is often pursued in order to achieve higher levels of linearity and power efficiency. We introduce the indirect learning architecture and advocate the use of basis expansion models for the PD, especially the orthogonal polynomial basis for improved numerical stability. We discuss memory effects in the PA—origins, symptoms, models, and remedies. Testbed measurement results on real PAs are provided to support the claims.

## Acknowledgments

This work was supported in part by the US Army Research Laboratory Communications and Networks Collaborative Technology Alliance Program, the US National Science Foundation Grant no. 0218778, and the Texas Instruments DSP Leadership University Program. The authors would like to thank Dr. Lei Ding, Dr. Raviv Raich, and Dr. Stevenson Kenney for insightful discussions on these topics.

## Bibliography

[1] S. Benedetto and E. Biglieri, *Principles of Digital Transmission: With Wireless Applications*, Kluwer Academic, Norwell, Mass, USA, 1999.

[2] S. Benedetto, E. Biglieri, and R. Daffara, "Modeling and performance evaluation of nonlinear satellite links—a Volterra series approach," *IEEE Transactions on Aerospace and Electronic*, vol. 15, no. 4, pp. 494–507, 1979.

[3] R. Blum and M. C. Jeruchim, "Modeling nonlinear amplifiers for communication simulation," in *Proceedings of IEEE International Conference on Communications (ICC '89)*, vol. 3, pp. 1468–1472, Boston, Mass, USA, June 1989.

[4] W. Bosch and G. Gatti, "Measurement and simulation of memory effects in predistortion linearizers," *IEEE Transactions on Microwave Theory and Techniques*, vol. 37, no. 12, pp. 1885–1890, 1989.

[5] N. Borges de Carvalho and J. C. Pedro, "A comprehensive explanation of distortion sideband asymmetries," *IEEE Transactions on Microwave Theory and Techniques*, vol. 50, no. 9, pp. 2090–2101, 2002.

[6] K. Chang, I. Bahl, and V. Nair, *RF and Microwave Circuits and Component Design for Wireless Systems*, John Wiley & Sons, New York, NY, USA, 2002.

[7] C. J. Clark, G. Chrisikos, M. S. Muha, A. A. Moulthrop, and C. P. Silva, "Time-domain envelope measurement technique with application to wideband power amplifier modeling," *IEEE Transactions on Microwave Theory and Techniques*, vol. 46, no. 12, part 2, pp. 2531–2540, 1998.

[8] S. C. Cripps, *RF Power Amplifiers for Wireless Communications*, Artech House, Norwood, Mass, USA, 1999.

[9] S. C. Cripps, *Advanced Techniques in RF Power Amplifier Design*, Artech House, Norwood, Mass, USA, 2002.

[10] L. Ding and G. T. Zhou, "Effects of even-order nonlinear terms on power amplifier modeling and predistortion linearization," *IEEE Transactions on Vehicular Technology*, vol. 53, no. 1, pp. 156–162, 2004.

[11] L. Ding, G. T. Zhou, D. R. Morgan, et al., "A robust digital baseband predistorter constructed using memory polynomials," *IEEE Transactions on Communications*, vol. 52, no. 1, pp. 159–165, 2004.

[12] C. Eun and E. J. Powers, "A predistorter design for a memory-less nonlinearity preceded by a dynamic linear system," in *Proceedings of IEEE Global Telecommunications Conference (GLOBECOM '95)*, vol. 1, pp. 152–156, Singapore, November 1995.

[13] K. G. Gard, H. M. Gutierrez, and M. B. Steer, "Characterization of spectral regrowth in microwave amplifiers based on the nonlinear transformation of a complex Gaussian process," *IEEE Transactions on Microwave Theory and Techniques*, vol. 47, no. 7, part 1, pp. 1059–1069, 1999.

[14] L. Hanzo, M. Münster, B. J. Choi, and T. Keller, *OFDM and MC-CDMA for Broadband Multi-User Communications, WLANs and Broadcasting*, John Wiley & Sons, West Sussex, England, 2003.

[15] The Institute of Electrical and Electronics Engineers, New York, NY, USA. *Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements—Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications—Amendment 1: High-speed Physical Layer in the 5 GHz band*, 2003. IEEE Std 802.11a-1999(R2003).

[16] T. Keller and L. Hanzo, "Adaptive multicarrier modulation: a convenient framework for time-frequency processing in wireless communications," *Proceedings of The IEEE*, vol. 88, no. 5, pp. 611–640, 2000.

[17] P. B. Kenington, *High Linearity RF Amplifier Design*, Artech House, Boston, Mass, USA, 2000.

[18] J. S. Kenney, *The RF and Microwave Handbook*, chapter Nonlinear Microwave Measurement and Characterization, CRC Press, Boca Raton, Fla, USA, 2001.

[19] J. Kim and K. Konstantinou, "Digital predistortion of wideband signals based on power amplifier model with memory," *Electronics Letters*, vol. 37, no. 23, pp. 1417–1418, 2001.

[20] H. Ku and J. S. Kenney, "Behavioral modeling of nonlinear RF power amplifiers considering memory effects," *IEEE Transactions on Microwave Theory and Techniques*, vol. 51, no. 12, pp. 2495–2504, 2003.

[21] H. Ku, M. D. McKinley, and J. S. Kenney, "Quantifying memory effects in RF power amplifiers," *IEEE Transactions on Microwave Theory and Techniques*, vol. 50, no. 12, pp. 2843–2849, 2002.

[22] V. K. N. Lau, "On the analysis of peak-to-average ratio (PAR) for IS95 and CDMA 2000 systems," *IEEE Transactions on Vehicular Technology*, vol. 49, no. 6, pp. 2174–2188, 2000.

[23] J. S. Lee and L. E. Miller, "Analysis of peak-to-average power ratio for IS-95 and third generation CDMA forward link waveforms," *IEEE Transactions on Vehicular Technology*, vol. 50, no. 4, pp. 1004–1013, 2001.

[24] T. Liu, S. Boumaiza, and F. M. Ghannouchi, "Dynamic behavioral modeling of 3G power amplifiers using real-valued time-delay neural networks," *IEEE Transactions on Microwave Theory and Techniques*, vol. 52, no. 3, pp. 1025–1033, 2004.

[25] S. A. Maas, *Nonlinear Microwave Circuits*, IEEE Press, Piscataway, NJ, USA, 1997.

[26] V. J. Mathews and G. L. Sicuranza, *Polynomial Signal Processing*, John Wiley & Sons, New York, NY, USA, 2000.

[27] K. J. Muhonen and M. Kavehrad, "Amplifier linearization with memory for broadband wireless applications," in *Proc. 35th Asilomar Conference on Signals, Systems and Computers*, vol. 1, pp. 689–693, Pacific Grove, Calif, USA, November 2001.

[28] Technical Specification Group Radio Access Network. *User Equipment (UE) radio transmission and reception (FDD)*. 3rd Generation Partnership Project, Valbonne, France, June 2002. 3GPP TS 25.101.

[29] Technical Specification Group Radio Access Network. *Base Station (BS) conformance testing (FDD)*. 3rd Generation Partnership Project, Valbonne, France, March 2003. 3GPP TS 25.141.

[30] H. Ochiai and H. Imai, "Performance of the deliberate clipping with adaptive symbol selection for strictly band-limited OFDM systems," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 11, pp. 2270–2277, 2000.

[31] H. Ochiai and H. Imai, "Performance analysis of deliberately clipped OFDM signals," *IEEE Transactions on Communications*, vol. 50, no. 1, pp. 89–101, 2002.

[32] F. H. Raab, P. Asbeck, S. Cripps, et al., "Power amplifiers and transmitters for RF and microwave," *IEEE Transactions on Microwave Theory and Techniques*, vol. 50, no. 3, pp. 814–826, 2002.

[33] R. Raich, H. Qian, and G. T. Zhou, "Orthogonal polynomials for power amplifier modeling and predistorter design," *IEEE Transactions on Vehicular Technology*, vol. 53, no. 5, pp. 1468–1479, 2004.

[34] R. Raich, H. Qian, and G. T. Zhou, "Optimization of SNDR for amplitude limited nonlinearities," *IEEE Transactions on Communications*, vol. 53, no. 11, pp. 1964–1972, 2005.

[35] R. Raich and G. T. Zhou, "On the modeling of memory nonlinear effects of power amplifiers for communication applications," in *Proc. IEEE 10th Digital Signal Processing Workshop (DSP '02)*, pp. 7–10, Pine Mountain, Ga, USA, November 2002.

[36] R. Raich and G. T. Zhou, "Orthogonal polynomials for complex Gaussian processes," *IEEE Transactions on Signal Processing*, vol. 52, no. 10, part 1, pp. 2788–2797, 2004.

[37] R. Raich and G. T. Zhou, "Spectral analysis for bandpass nonlinearity with cyclostationary input," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '04)*, vol. 2, pp. 465–468, Montreal, Quebec, Canada, May 2004.

[38] A. M. Saleh, "Frequency-independent and frequency-dependent nonlinear models of TWT amplifiers," *IEEE Transactions on Communications*, vol. 29, no. 11, pp. 1715–1720, 1981.

[39] M. Schetzen, *The Volterra and Wiener Theories of Nonlinear Systems*, Krieger Publishing, Malabar, Fla, USA, 1980.

[40] S. P. Stapleton, G. S. Kandola, and J. K. Cavers, "Simulation and analysis of an adaptive predistorter utilizing a complex spectral convolution," *IEEE Transactions on Vehicular Technology*, vol. 41, no. 4, pp. 387–394, 1992.

[41] L. Sundström, *Digital RF power amplifier linearisers—analysis and design*, Ph.D. thesis, Department of Applied Electronics, Lund University, Lund, Sweden, 1995.

[42] J. Tellado, *Multicarrier Modulation with Low PAR: Applications to DSL and Wireless*, Kluwer Academic, New York, NY, USA, 2000.

[43] J. H. K. Vuolevi, T. Rahkonen, and J. P. A. Manninen, "Measurement technique for characterizing memory effects in RF power amplifiers," *IEEE Transactions on Microwave Theory and Techniques*, vol. 49, no. 8, pp. 1383–1389, 2001.

[44] G. T. Zhou and J. S. Kenney, "Predicting spectral regrowth of nonlinear power amplifiers," *IEEE Transactions on Communications*, vol. 50, no. 5, pp. 718–722, 2002.

[45] G. T. Zhou, H. Qian, L. Ding, and R. Raich, "On the baseband representation of a bandpass nonlinearity," *IEEE Transactions on Signal Processing*, vol. 53, no. 8, part 1, pp. 2953–2957, 2005.

[46] G. T. Zhou and R. Raich, "Spectral analysis of polynomial nonlinearity with applications to RF power amplifiers," *EURASIP Journal on Applied Signal Processing*, vol. 2004, no. 12, pp. 1831–1840, 2004, Special issue on Nonlinear Signal and Image Processing—Part I.

G. Tong Zhou: School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0250, USA

*Email*: gtz@ece.gatech.edu

Hua Qian: School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0250, USA

*Email*: qianhua@ece.gatech.edu

Ning Chen: School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0250, USA

*Email*: nchen@ece.gatech.edu

# 7 Nonlinear multichannel active noise control

Giovanni L. Sicuranza and Alberto Carini

## 7.1. Introduction

The principle of active noise control (ANC) has been well known for many years, as shown by the seminal contributions in [28, 33]. However, methods for ANC have been intensively studied only in the last three decades as a consequence of the developments in the area of digital technologies. Such studies have already provided a good comprehension of the theory and consequently useful applications in vibration and acoustic noise control tasks. Generally speaking, active control of vibrations and acoustic noises are based on the same principle of the destructive interference in a given location between a disturbance source and an appropriately generated signal with the same amplitude as the disturbance source but with an opposite phase. The main difference in the two situations is due to the nature of the actuators used to generate the interfering signal, which are mainly mechanical in the case of vibration control while they are electroacoustical devices, as loudspeakers, in the case of acoustic noise control.

The vibration control is also related to the problem of attenuating the corresponding sound generated by the vibrations in the auditory spectrum. Studies in this field are currently developed with reference to environmental and legal restrictions concerning people's safety and health. Among many examples, we may refer to the application of actively controlling the inputs to vibrating structures, such as airplane cockpits, to minimize the sound radiation.

On the other hand, in most cases the acoustic noise can be directly reduced by introducing a secondary noise source, such as a loudspeaker, which produces a sound field that destructively interferes with the original noise in a desired location where the so-called error microphone is placed. The secondary noise source is driven by the active noise controller that is adapted according to the error signal and the input signal collected by a reference microphone in proximity of the noise source. Successful implementations of this approach can be found in systems such as air conditioning ducts, to attenuate the low frequency noise due to the fans, or in transport systems, to reduce the noise generated by the engines inside propeller aircrafts, vehicles, and helicopters.

Physical limitations generally restrict the frequency range of ANC systems below a few hundred hertz. The main limitation is related to the wavelength of the acoustic waves in connection to the extension of the silenced area. At higher frequencies, it is still possible to cancel the sound in a limited zone around, for example, a listener's ear, and suitable systems have been already successfully implemented for active headsets. To attenuate disturbance frequencies above 1 kH, passive systems, based on the absorption and reflection properties of materials, are widely used and give good performances.

The initial activities on vibration and noise control originated in the field of control engineering [24, 32, 53], while in recent years the advantages offered by the signal processing methodologies have been successfully exploited [18, 19, 27]. The latter approach offers great opportunities for real-time implementations as a consequence of recent advances in electroacoustic transducers, digital signal processors, and adaptation algorithms.

Most of the studies presented in the literature refer to linear models and linear adaptive controllers represent the state of the art in the field. However, linear controllers can hardly provide any further improvements in terms of control performance. On the other hand, nonlinear modeling techniques may bring new insights and suggest new developments. In fact, it is often recognized that nonlinearities may affect actual applications. Nonlinear effects arise from the behavior of the noise source which rather than a stochastic process may be described as a nonlinear deterministic process, sometimes of chaotic nature [30, 45]. Moreover, the paths modeling the acoustic system may exhibit a nonlinear behavior, thus further motivating the use of a nonlinear controller [26].

The simplest and most frequently used ANC systems exploit a single-channel configuration where the primary path $P$ consists of the acoustic response from the noise source to an error microphone located at the canceling point. The noise source is sensed by a single reference microphone and an interfering signal is produced by a single loudspeaker. A single-channel active noise controller gives, in principle, attenuation of the undesired disturbance only in the proximity of the error microphone. Even though the silenced region may be relatively large at low frequencies, that is, in a region that is appropriately small with respect to the wavelength of the corresponding acoustical waves, to spatially extend the silenced region a multichannel approach may be preferred. In this case, sets of reference sensors, actuators, and error microphones are used increasing the complexity of the ANC system.

In this chapter, the main aspects of the active control of acoustic noises are reviewed from a signal processing point of view with particular attention to the nonlinear multichannel framework. While different nonlinear models for the controllers have been considered recently, polynomial or Volterra filters [29] have been shown to be particularly useful. Thus in this chapter we will refer to a class of nonlinear controllers that contains all the filters that are characterized by a linear relationship of the output with respect to the filter coefficients. Such a class includes in particular truncated Volterra filters [29] as well as functional-link artificial neural networks [35].

The main problems arising in the nonlinear controllers are the following.

(i) The increase in the computational load caused by the complexity of both the multichannel approach and the representation of nonlinearities.

(ii) The correlations existing among signals in multichannel schemes due to the similarities in the acoustic paths and their interaction.

(iii) The correlations existing, even when the input signals are white, between the elements of the input vectors to the Volterra filters which are given by products of samples [29, page 253].

All these problems have an impact on the efficiency and the accuracy of the adaptation algorithms used for the controllers. As a consequence, the main aspect to study is the derivation of fast and reliable adaptation algorithms for nonlinear multichannel controllers. It is worth noting that this research area represents a challenge for modern nonlinear signal processing.

In this chapter, a description of the ANC scenario is given first with particular reference to the so-called *filtered-X* adaptation algorithms for linear single-channel and multichannel schemes. Then, the nonlinear effects that may influence the behavior of an ANC system are reviewed. It is also shown how the filtered-X updating algorithms previously described can be profitably applied to the class of nonlinear active noise controllers whose filters are characterized by a linear relationship of the output with respect to the filter coefficients. A few experimental results for nonlinear multichannel active noise controllers belonging to this class are presented and commented. Finally, current work and future research lines are exposed.

## 7.2. The active noise control scenario

Actual ANC systems are essentially mixed analog-digital systems that require A/D and D/A converters, power amplifiers, loudspeakers, and transducers. In this section, the digital core of single-channel and multichannel ANC systems is briefly described, together with the corresponding algorithms used for adapting the controllers in the linear case. The feedforward adaptation algorithms are considered in more detail since they can be profitably exploited in nonlinear situations. The derivations that lead to the filtered-X least mean square (LMS) and the filtered-X affine projection (AP) algorithms for single-channel and multichannel schemes are summarized. All the derivations make use of a vector-matrix notation. Due to space limitations, in some cases only the rational of the adaptation algorithms is reported while the detailed derivations can be found in the specific papers referred to.

### 7.2.1. Single-channel and multichannel schemes for active noise control

A typical single-channel acoustic noise control scheme is shown in Figure 7.1.

The primary path $P$ consists of the acoustic response from the noise source to the error microphone located at the canceling point. The noise source is sensed by a reference microphone that collects the samples of the noise source $n_s(n)$ and

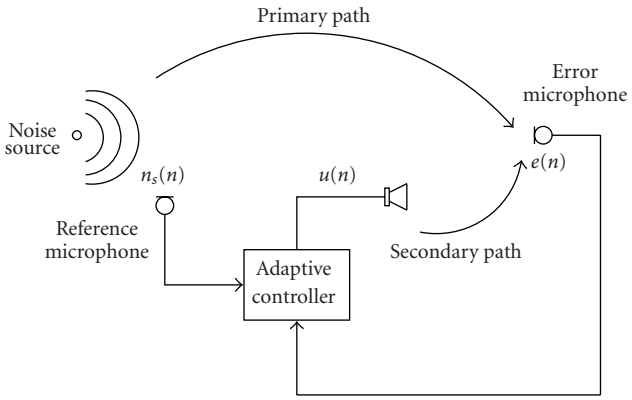



FIGURE 7.1. Principle of single-channel active noise control.

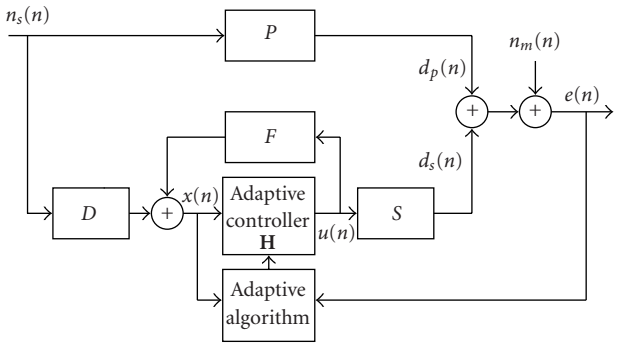



FIGURE 7.2. Block diagram of a single-channel active noise control system.

feeds them as input to the adaptive controller. The controller is adapted using $n_s(n)$ and the error signal $e(n)$ sent back by the error microphone. The signal $u(n)$, generated by the controller through a loudspeaker and traveling through the secondary path $S$, gives a signal which destructively interferes with the undesired noise at the error microphone location. A block diagram depicting this situation is shown in Figure 7.2. In this scheme, $P$ and $S$ represent the transfer functions of the primary and secondary paths, respectively, and $d_p(n)$ and $d_s(n)$ are the signals that destructively interfere at the location where the error microphone is placed. $D$ is the transfer function of the detector path, that is, the path between the noise source and the reference microphone, while $F$ is a transfer function modeling the acoustical feedback between the loudspeaker and the reference microphone. Finally, $n_m(n)$ is an added measurement noise, described as a white or colored zero mean noise process uncorrelated with the noise source. In general, all the various paths, and thus the corresponding transfer functions, are assumed to be linear, even though, as discussed below, there is the evidence that nonlinear distortions

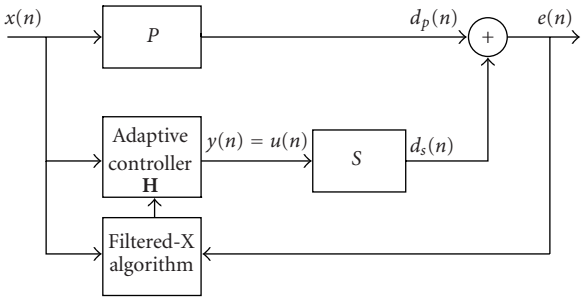



FIGURE 7.3. Feedforward scheme for active noise control.

may affect all these paths. This general block diagram includes the most relevant schemes that have been studied in the literature [23]. The first assumption that the acoustical feedback is negligible leads, in fact, to the widely used *feedforward* schemes [18]. By further neglecting the detector path and the measurement noise, the simplified scheme of Figure 7.3 is derived. In a linear and time-invariant environment, it is often assumed that the secondary path has been modeled by separate estimation procedures in the form of an FIR filter with impulse response $s(n)$. Then, the signal $d_s(n)$ is given as the linear convolution of $s(n)$ with the signal $y(n)$ and thus the error signal can be expressed as

$$e(n) = d_p(n) + d_s(n) = d_p(n) + s(n) * y(n), \tag{7.1}$$

where $*$ indicates the operation of linear convolution. This is the basis for the derivation of the so-called *filtered-X* algorithms [1, 18, 38] where the adaptive controller is modeled as an FIR filter. In case the acoustical feedback can not be neglected, the controller needs to be an IIR filter. Its coefficients are adapted using the so-called *filtered-U* algorithm [20], which is based on the pseudolinear regression method proposed by Feintuch for adaptive IIR filters [21].

The model of Figure 7.2 also includes the other broad class of active noise controllers, that is, the *feedback* schemes [18, 31]. In fact, assuming that the detector path equals the primary path and the intrinsic feedback equals the secondary path, with the condition that there is no output measurement noise, the scheme of Figure 7.4 is derived which coincides with the scheme of a feedback structure. The lack of a reference signal is characteristic for the feedback schemes and this arrangement is used when it is difficult to put a reference microphone or a transducer in proximity of the noise source. The most successful development of these systems is in active headsets where an actuator source (a loudspeaker) and an error sensor (a microphone) are located in very close positions within the earcup. Since the earcup cavity is small with respect to the sound wavelengths, the difficulties related to the extension of the silenced zone are alleviated. It was reported in 1995 [24] that in the market there were about a dozen of different competing headsets, offering sound reductions of more than 10 dB up to about 500 Hz.

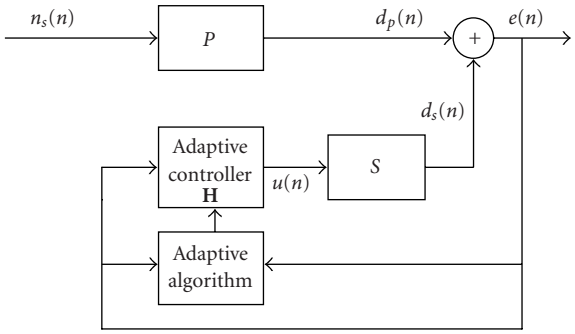



FIGURE 7.4. Feedback scheme for active noise control.

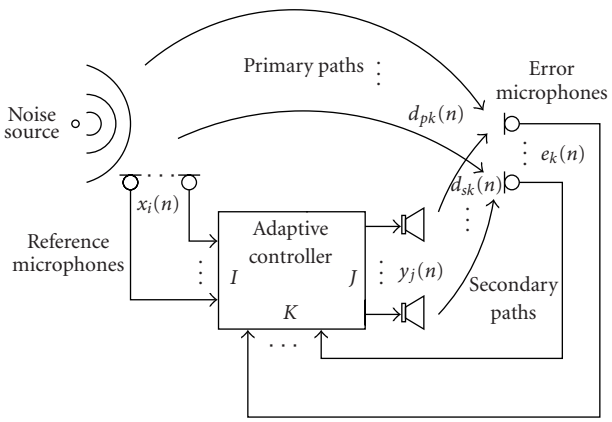



FIGURE 7.5. Multichannel active noise control.

However, to avoid the delays in the feedback loop of these systems, the first feedback controllers used analog circuits optimized to work well with a given disturbance spectrum. Experiments on active headphone sets in a digital context are reported in [31], where also some reference to nonlinear effects is made.

The single-channel ANC scheme gives, in principle, attenuation of the undesired disturbance in the proximity of the point where the error microphone collects the error signal. To spatially extend the silenced region a multichannel approach can be applied using sets of reference sensors, actuator sources, and error microphones [16, 17]. A general scheme describing this situation is shown in Figure 7.5, where $I$ input sensors are used to collect the corresponding input signals. In the controller, any input $i$, $1 \leq i \leq I$, is usually connected to the output $j$, $1 \leq j \leq J$, via an FIR filter, and thus the controller computes $J$ output signals which are propagated to the $K$ error sensors. As shown in [15], the main drawbacks of this approach are the complexity of the coefficient updates, the data storage requirements, and the slow convergence of the adaptive algorithms.

A commercial application of a multichannel controller has been developed, for example, by Saab for the reduction of the interior noise in a Saab 340 propeller aircraft [24]. The reference signals are generated from synchronization signals taken from the propeller shafts. The controller is implemented by means of FIR filters equipped with a multichannel version of the filtered-X algorithm. A system of 24 sources and 48 microphone errors are used and a noise attenuation of about 10 dB is reached. Other experiments are reported in [15] for the active control of the air compressor noise. Different multichannel structures have been simulated using air compressor data measured in an anechoic environment resulting in significant levels of attenuation.

### 7.2.2. Filtered-X LMS algorithm for linear single-channel controllers

With reference to Figure 7.3, we can express the output from the single-channel controller at time $n$ as

$$y(n) = \mathbf{h}^T(n)\mathbf{x}(n), \tag{7.2}$$

where $\mathbf{x}(n)$ is the vector of the last $N$ input samples from the reference microphone

$$\mathbf{x}(n) = \begin{bmatrix} x(n) & x(n-1) & \cdots & x(n-N+1) \end{bmatrix}^T \tag{7.3}$$

and $\mathbf{h}(n)$ is the vector of the $N$ coefficients of the FIR filter implementing the controller at time $n$

$$\mathbf{h}(n) = \begin{bmatrix} h(n;0) & h(n;1) & \cdots & h(n;N-1) \end{bmatrix}^T. \tag{7.4}$$

The signal collected at the error microphone is then expressed as

$$\begin{aligned} e(n) &= d_p(n) + d_s(n) \\ &= d_p(n) + s(n) * y(n) \\ &= d_p(n) + s(n) * \{\mathbf{h}^T(n)\mathbf{x}(n)\}, \end{aligned} \tag{7.5}$$

where the symbol $*$ indicates the operation of linear convolution between the finite-impulse response $s(n)$ modeling the secondary path and the signal at the output of the controller. The coefficients in the vector $\mathbf{h}(n)$ need to be updated so that the error function $E[e^2(n)]$ is minimized. By applying the steepest descent algorithm, the updating equation becomes

$$\mathbf{h}(n+1) = \mathbf{h}(n) - \alpha \frac{\partial E[e^2(n)]}{\partial \mathbf{h}(n)}. \tag{7.6}$$

According to the stochastic gradient approximation, the updating terms are computed as

$$\begin{aligned}\frac{\partial E[e^2(n)]}{\partial \mathbf{h}(n)} \simeq \frac{\partial(e^2(n))}{\partial \mathbf{h}(n)} &= 2e(n)\frac{\partial(d_p(n)+s(n)*\{\mathbf{h}^T(n)\mathbf{x}(n)\})}{\partial \mathbf{h}(n)} \\ &= 2e(n)(s(n)*\mathbf{x}(n)).\end{aligned} \tag{7.7}$$

The complete updating equation then becomes

$$\mathbf{h}(n+1) = \mathbf{h}(n) - \mu e(n)(s(n)*\mathbf{x}(n)), \tag{7.8}$$

where $\mu = 2\alpha$ is the step size, or adaptation constant, controlling the convergence properties of the algorithm. It is worth noting that the adaptation equation in (7.8) requires the a priori knowledge or estimation of the impulse response $s(n)$ of the secondary path.

### 7.2.3. Filtered-X LMS algorithm for linear multichannel controllers

The single-channel updating algorithm can be easily extended to the multichannel scheme. With reference to Figure 7.5, $I$ input signals enter into the controller whose outputs drive $J$ loudspeakers. The linear controller is modeled as a set of $J \cdot I$ FIR filters with memory length equal to $N_i$, $1 \le i \le I$, connecting each input $i$ to each output $j$. The adaptation algorithm exploits the $K$ signals fed back by the error microphones. Updating the notation used for the single-channel case, it is possible to write for each actuator $j$ of the multichannel scheme the following equations:

$$y_j(n) = \sum_{i=1}^{I} \mathbf{h}_{ji}^T(n)\mathbf{x}_i(n), \tag{7.9}$$

where $\mathbf{x}_i(n)$ is the vector of the last $N_i$ input samples from the reference microphone $i$,

$$\mathbf{x}_i(n) = \begin{bmatrix} x_i(n) & x_i(n-1) & \cdots & x_i(n-N_i+1) \end{bmatrix}^T \tag{7.10}$$

and $\mathbf{h}_{ji}(n)$ is the vector of the $N_i$ coefficients of the FIR filter connecting at time $n$ the input $i$ to the output $j$ of the controller

$$\mathbf{h}_{ji}(n) = \begin{bmatrix} h_{ji}(n;0) & h_{ji}(n;1) & \cdots & h_{ji}(n;N_i-1) \end{bmatrix}^T. \tag{7.11}$$

The signal collected at the $k$th error microphone is then expressed as

$$\begin{aligned} e_k(n) &= d_{pk}(n) + d_{sk}(n) = d_{pk}(n) + \sum_{j=1}^{J} s_{kj}(n) * y_j(n) \\ &= d_{pk}(n) + \sum_{j=1}^{J} s_{kj}(n) * \left\{ \sum_{i=1}^{I} \mathbf{h}_{ji}^T(n)\mathbf{x}_i(n) \right\}, \end{aligned} \tag{7.12}$$

where the signal $d_{sk}(n)$ takes into account all the contributions given by the $J$ actuator sources. The coefficients in each vector $\mathbf{h}_{ji}(n)$ are still updated so that the whole error function $E[\sum_{k=1}^{K} e_k^2(n)]$ is minimized. According to the stochastic gradient approximation, the updating terms for each of the vectors $\mathbf{h}_{ji}(n)$ are computed as

$$\begin{aligned} \frac{\partial\left(\sum_{k=1}^{K} e_k^2(n)\right)}{\partial \mathbf{h}_{ji}(n)} &= 2\sum_{k=1}^{K} e_k(n) \frac{\partial\left(d_{pk}(n) + \sum_{j=1}^{J} s_{kj}(n) * \left\{\sum_{i=1}^{I} \mathbf{h}_{ji}^T(n)\mathbf{x}_i(n)\right\}\right)}{\partial \mathbf{h}_{ji}(n)} \\ &= 2\sum_{k=1}^{K} e_k(n)\left(s_{kj}(n) * \mathbf{x}_i(n)\right). \end{aligned} \tag{7.13}$$

The complete updating equation for the FIR filter connecting the input $i$ to the output $j$ can be written as

$$\mathbf{h}_{ji}(n+1) = \mathbf{h}_{ji}(n) - \sum_{k=1}^{K} \mu_k e_k(n)\left(s_{kj}(n) * \mathbf{x}_i(n)\right), \tag{7.14}$$

where $\mu_k$, $k = 1, \ldots, K$, are the step sizes, or adaptation constants, controlling the convergence properties of the algorithm. It is worth noting that the coefficients of each filter in the controller are updated independently from those of the other filters using a weighted combination of the errors at the $K$ sensors. The adaptation equation in (7.14) requires the a priori knowledge or estimation of the impulse responses $s_{kj}(n)$ of the secondary paths.

### 7.2.4. Filtered-X AP algorithm for linear single-channel controllers

While the filtered-X LMS algorithm minimizes, according to the stochastic gradient approximation, the single error at time $n$, the aim of the filtered-X AP algorithm of order $L$ is to minimize the last $L$ errors at times $n, n-1, \ldots, n-L+1$ [34]. More specifically, the desired minimization is obtained by finding the minimum norm of the coefficient increments that set to zero the last $L$ *a posteriori* errors defined as

$$\epsilon_{n+1}(n-l+1) = d_p(n-l+1) + s(n-l+1) * \left\{\mathbf{h}^T(n+1)\mathbf{x}(n-l+1)\right\}, \tag{7.15}$$

for $1 \leq l \leq L$. A suitable error function is defined and then minimized by using a set of Lagrange's multipliers. The set of constraints for an $L$th-order filtered-X AP algorithm is given by the following equations:

$$\begin{aligned} d_p(n) + s(n) * \{\mathbf{h}^T(n+1)\mathbf{x}(n)\} &= 0, \\ d_p(n-1) + s(n-1) * \{\mathbf{h}^T(n+1)\mathbf{x}(n-1)\} &= 0, \\ &\vdots \\ d_p(n-L+1) + s(n-L+1) * \{\mathbf{h}^T(n+1)\mathbf{x}(n-L+1)\} &= 0. \end{aligned} \tag{7.16}$$

The function $\varepsilon(n)$ to be minimized is defined as

$$\begin{aligned} \varepsilon(n) = {} & \delta\mathbf{h}^T(n+1)\delta\mathbf{h}(n+1) \\ & + \sum_{l=1}^{L} \lambda_l (d_p(n-l+1) + s(n-l+1) * \{\mathbf{h}^T(n+1)\mathbf{x}(n-l+1)\}), \end{aligned} \tag{7.17}$$

where

$$\delta\mathbf{h}(n+1) = \mathbf{h}(n+1) - \mathbf{h}(n) \tag{7.18}$$

are the coefficient increments and $\lambda_l$ are the Lagrange's multipliers. By differentiating $\varepsilon(n)$ with respect to $\delta\mathbf{h}(n+1)$ the following set of $N$ equations is obtained:

$$2\delta\mathbf{h}(n+1) = -\sum_{l=1}^{L} \lambda_l s(n-l+1) * \mathbf{x}(n-l+1), \tag{7.19}$$

where the linearity of the convolution operation has been exploited. Equation (7.19) can be also rewritten as

$$2\delta\mathbf{h}(n+1) = -\mathbf{G}(n)\boldsymbol{\Lambda}, \tag{7.20}$$

where the $N \times L$ matrix $\mathbf{G}(n)$ is defined as

$$\mathbf{G}(n) = \begin{bmatrix} s(n) * \mathbf{x}(n) & s(n-1) * \mathbf{x}(n-1) & \cdots & s(n-L+1) * \mathbf{x}(n-L+1) \end{bmatrix}, \tag{7.21}$$

and the $L \times 1$ vector $\boldsymbol{\Lambda}$ is defined as

$$\boldsymbol{\Lambda} = \begin{bmatrix} \lambda_1 & \lambda_2 & \cdots & \lambda_L \end{bmatrix}^T. \tag{7.22}$$

By premultiplying (7.20) by $\mathbf{G}^T(n)$, the following equation is obtained:

$$\boldsymbol{\Lambda} = -\left(\mathbf{G}^T(n)\mathbf{G}(n)\right)^{-1}\mathbf{G}^T(n)2\delta\mathbf{h}(n+1). \tag{7.23}$$

The $L \times 1$ vector $\mathbf{G}^T(n)2\delta\mathbf{h}(n+1)$ can be also written as

$$\mathbf{G}^T(n)2\delta\mathbf{h}(n+1) = 2\begin{bmatrix} (s(n) * \mathbf{x}^T(n))\delta\mathbf{h}(n+1) \\ (s(n-1) * \mathbf{x}^T(n-1))\delta\mathbf{h}(n+1) \\ \vdots \\ (s(n-L+1) * \mathbf{x}^T(n-L+1))\delta\mathbf{h}(n+1) \end{bmatrix}. \tag{7.24}$$

Apart from the factor 2, the $l$th element of this vector ($l = 1, \ldots, L$) can be written, after some manipulations, as

$$\begin{aligned} &(s(n-l+1) * \mathbf{x}^T(n-l+1))\,(\mathbf{h}(n+1) - \mathbf{h}(n)) \\ &\quad = s(n-l+1) * \{\mathbf{h}^T(n+1)\mathbf{x}(n-l+1)\} \\ &\qquad - s(n-l+1) * \{\mathbf{h}^T(n)\mathbf{x}(n-l+1)\} \\ &\quad = -d_p(n-l+1) - s(n-l+1) * \{\mathbf{h}^T(n)\mathbf{x}(n-l+1)\} \\ &\quad = -e(n-l+1). \end{aligned} \tag{7.25}$$

In deriving this expression, the linearity of the convolution operation and the constraints given by (7.16) have been exploited. As a conclusion, (7.24) can be rewritten as

$$\mathbf{G}^T(n)2\delta\mathbf{h}(n+1) = -2\mathbf{e}(n), \tag{7.26}$$

where $\mathbf{e}(n)$ is the $L \times 1$ vector of the filtered-X a priori estimation errors

$$\mathbf{e}(n) = \begin{bmatrix} e(n) & e(n-1) & \cdots & e(n-L+1) \end{bmatrix}^T. \tag{7.27}$$

By combining (7.20), (7.23), and (7.26), the following relation is derived:

$$\delta\mathbf{h}(n+1) = -\mathbf{G}(n)\left(\mathbf{G}^T(n)\mathbf{G}(n)\right)^{-1}\mathbf{e}(n). \tag{7.28}$$

In conclusion, the filtered-X AP adaptation rule for a linear controller can be written as

$$\mathbf{h}(n+1) = \mathbf{h}(n) - \mu\mathbf{G}(n)\left[\mathbf{G}^T(n)\mathbf{G}(n)\right]^{-1}\mathbf{e}(n), \tag{7.29}$$

where $\mu$ is a parameter that controls both the convergence rate and the stability of the filtered-X AP algorithm. The $L \times L$ matrix $\mathbf{G}^T(n)\mathbf{G}(n)$ represents an estimate of the autocorrelation matrix of the input signal $x$ filtered by the transfer function $S$ of the secondary path, obtained using the last $L$ input vectors. The computation of its inverse is required at every time instant $n$. For low orders of AP, that is, with $L = 2, 3$, the direct inversion of the matrix is an affordable task. The only necessary

care in order to avoid possible numerical instabilities is to add a diagonal matrix $\delta\mathbf{I}$, where $\delta$ is a small positive constant, to the matrix $\mathbf{G}^T(n)\mathbf{G}(n)$ so that

$$\mathbf{h}(n+1) = \mathbf{h}(n) - \mu\mathbf{G}(n)\left[\mathbf{G}^T(n)\mathbf{G}(n) + \delta\mathbf{I}\right]^{-1}\mathbf{e}(n). \tag{7.30}$$

A general and efficient solution which can be applied to any order $L$ of AP is shown in [7] by resorting to a simpler and more stable estimate for the inverse of the matrix $\mathbf{G}^T(n)\mathbf{G}(n)$, based on the matrix inversion lemma, as usually done in classical RLS algorithms.

It is worth noting that for $L = 1$ the matrix $\mathbf{G}(n)$ reduces to

$$\mathbf{G}(n) = \left[s(n) * \mathbf{x}(n)\right] \tag{7.31}$$

and thus the updating rule becomes

$$\mathbf{h}(n+1) = \mathbf{h}(n) - \mu\frac{s(n) * \mathbf{x}(n)}{\left\|s(n) * \mathbf{x}(n)\right\|^2}\mathbf{e}(n). \tag{7.32}$$

This updating rule coincides with the normalized filtered-X LMS algorithm.

### 7.2.5. Filtered-X AP algorithm for linear multichannel controllers

As already noted, the peculiarity of the AP algorithms [34] is that a set of $L-1$ previous errors are considered in addition to the present one. To derive in the multichannel case an exact AP filtered-X algorithm, similar to the one given by (7.30) for the single-channel controller, it is necessary to arrange all the coefficients of the $J \cdot I$ FIR filters of the controller in a single vector $\mathbf{h}(n)$. To this purpose, let us first define the following vectors and matrices:

$$\mathbf{x}_i(n) = \begin{bmatrix} x_i(n) & x_i(n-1) & \cdots & x_i(n-N+1) \end{bmatrix}^T \tag{7.33}$$

is a column vector which collects the last $N$ samples at the input $i$, $1 \le i \le I$. For sake of simplicity, it has been assumed here that the memory of all the FIR filters in the controller is equal to $N$.

$$\mathbf{x}(n) = \begin{bmatrix} \mathbf{x}_1^T(n) & \mathbf{x}_2^T(n) & \cdots & \mathbf{x}_I^T(n) \end{bmatrix}^T \tag{7.34}$$

is a column vector with $I \times N$ elements, formed by arranging in sequence the input column vectors $\mathbf{x}_i(n)$.

$$\mathbf{y}(n) = \begin{bmatrix} y_1(n) & y_2(n) & \cdots & y_J(n) \end{bmatrix}^T \tag{7.35}$$

is the $J \times 1$ column vector of the outputs from the controller.

$$\mathbf{h}_{ji}(n) = \begin{bmatrix} h_{ji}(n;0) & h_{ji}(n;1) & \cdots & h_{ji}(n;N-1) \end{bmatrix}^T \tag{7.36}$$

is the $N \cdot 1$ column vector formed by the coefficients at time $n$ of each FIR filter, with $1 \leq j \leq J$, $1 \leq i \leq I$.

$$\mathbf{H}(n) = \begin{bmatrix} \mathbf{h}_{11}(n) & \mathbf{h}_{21}(n) & \cdots & \mathbf{h}_{J1}(n) \\ \mathbf{h}_{12}(n) & \mathbf{h}_{22}(n) & \cdots & \mathbf{h}_{J2}(n) \\ & & \vdots & \\ \mathbf{h}_{1I}(n) & \mathbf{h}_{2I}(n) & \cdots & \mathbf{h}_{JI}(n) \end{bmatrix} \tag{7.37}$$

is the $(I \cdot N) \times J$ matrix formed by the $J \cdot I$ FIR filters of the controller.

Using such notations, the input-output relationship of the controller is given by

$$\mathbf{y}(n) = \mathbf{H}^T(n)\mathbf{x}(n). \tag{7.38}$$

Then, the column vector $\mathbf{h}(n)$ collecting all the $N_{\text{tot}} = J \cdot I \cdot N$ coefficients of the FIR filters can be obtained by the equation

$$\mathbf{h}(n) = \text{vec}\,(\mathbf{H}(n)), \tag{7.39}$$

where $\text{vec}(\cdot)$ is the vector operator which casts in a single column all the columns of a matrix. It is easy to show that the updating rules of (7.29) and (7.30) are still valid for the multichannel case if the matrix $\mathbf{G}(n)$ is suitably redefined. To this purpose, let us introduce the following $K$ column vectors of dimensions $N_{\text{tot}} \times 1$ which collect the results of the convolutions among the secondary path impulse responses $s_{kj}(n)$ and the input signals $\mathbf{x}_i(n)$, for $1 \leq j \leq J$ and $1 \leq i \leq I$,

$$\boldsymbol{\chi}_k(n) = \begin{bmatrix} s_{k1}(n) * \mathbf{x}_1(n) \\ s_{k1}(n) * \mathbf{x}_2(n) \\ \vdots \\ s_{k1}(n) * \mathbf{x}_I(n) \\ s_{k2}(n) * \mathbf{x}_1(n) \\ s_{k2}(n) * \mathbf{x}_2(n) \\ \vdots \\ s_{k2}(n) * \mathbf{x}_I(n) \\ \vdots \\ \vdots \\ s_{kJ}(n) * \mathbf{x}_1(n) \\ s_{kJ}(n) * \mathbf{x}_2(n) \\ \vdots \\ s_{kJ}(n) * \mathbf{x}_I(n) \end{bmatrix}, \tag{7.40}$$

where $1 \leq k \leq K$. Then, the $N_{\text{tot}} \times (K \cdot L)$ matrix $\mathbf{G}(n)$ is defined as

$$\mathbf{G}(n) = \Big[\chi_1(n) \quad \chi_1(n-1) \quad \cdots \quad \chi_1(n-L+1) \quad \chi_2(n) \quad \chi_2(n-1) \quad \cdots \quad \chi_2(n-L+1) \quad \chi_K(n) \quad \chi_K(n-1) \quad \cdots \quad \chi_K(n-L+1)\Big]. \tag{7.41}$$

It is worth noting that in the present situation $\mathbf{G}^T(n)\mathbf{G}(n)$ is a symmetric matrix that plays the role of a normalization factor. As a consequence, the exact filtered-X AP algorithm takes into account the interactions among the different paths and filters in the adaptation rule, in contrast to the multichannel LMS algorithm in (7.14). The computational complexity of the algorithm clearly depends on the dimensions $(K \cdot L) \times (K \cdot L)$ of this matrix, which needs to be inverted at each iteration. Therefore, to reduce the implementation complexity, approximate AP algorithms need to be devised.

#### 7.2.5.1. Two approximate filtered-X AP algorithms

In order to derive the approximate filtered-X AP algorithms, it is convenient to subdivide the matrix $\mathbf{G}(n)$ in $K$ submatrices of dimensions $N_{\text{tot}} \times L$,

$$\mathbf{G}_k(n) = \Big[\chi_k(n) \quad \chi_k(n-1) \quad \cdots \quad \chi_k(n-L+1)\Big], \tag{7.42}$$

for $1 \leq k \leq K$, so that

$$\mathbf{G}(n) = \Big[\mathbf{G}_1(n) \quad \mathbf{G}_2(n) \quad \cdots \quad \mathbf{G}_K(n)\Big]. \tag{7.43}$$

Then, a first approximate filtered-X AP algorithm can be obtained by replacing the matrix $\mathbf{G}^T(n)\mathbf{G}(n)$ with the block-diagonal matrix

$$\widetilde{\mathbf{G}}^T(n)\widetilde{\mathbf{G}}(n) = \begin{bmatrix} \mathbf{G}_1^T(n)\mathbf{G}_1(n) & & & \\ & \mathbf{G}_2^T(n)\mathbf{G}_2(n) & & \\ & & \ddots & \\ & & & \mathbf{G}_K^T(n)\mathbf{G}_K(n) \end{bmatrix}. \tag{7.44}$$

The updating relationship then becomes

$$\mathbf{h}(n+1) = \mathbf{h}(n) - \mu\mathbf{G}(n)\big[\widetilde{\mathbf{G}}^T(n)\widetilde{\mathbf{G}}(n) + \delta\mathbf{I}\big]^{-1}\mathbf{e}(n). \tag{7.45}$$

In this case, the inversion of $K$ matrices $L \times L$ is required in place of the inversion of the $(K \cdot L) \times (K \cdot L)$ matrix of the exact algorithm. It has been shown in [9] that this algorithm has better convergence behavior than the LMS algorithm while the computational complexity is much lower than the updating rule of (7.30) and (7.41).

Another possibility, which has been actually considered first in [40], consists in additionally splitting the matrices $\mathbf{G}_k(n)$ in $J \cdot I$ submatrices $N \times L$, $\mathbf{G}_{kji}(n)$, so that

$$\mathbf{G}_k(n) = \begin{bmatrix} \mathbf{G}_{k11}(n) \\ \mathbf{G}_{k12}(n) \\ \vdots \\ \mathbf{G}_{k1I}(n) \\ \mathbf{G}_{k21}(n) \\ \mathbf{G}_{k22}(n) \\ \vdots \\ \mathbf{G}_{k2I}(n) \\ \vdots \\ \mathbf{G}_{kJ1}(n) \\ \mathbf{G}_{kJ2}(n) \\ \vdots \\ \mathbf{G}_{kJI}(n) \end{bmatrix}. \tag{7.46}$$

In this case, the coefficients of each filter in the controller are updated with a rule that, apart from the normalization matrices, is similar to the filtered-X LMS algorithm in (7.14). The complete derivations, given in [40] for linear and quadratic filters, lead to the following expression for the updating of the FIR filters involved in the linear controller

$$\mathbf{h}_{ji}(n+1) = \mathbf{h}_{ji}(n) - \sum_{k=1}^{K} \mu_k \mathbf{G}_{kji}(n) \Big[ \mathbf{G}_{kji}^T(n) \mathbf{G}_{kji}(n) + \delta \mathbf{I} \Big]^{-1} \mathbf{e}_k(n), \tag{7.47}$$

where

$$\mathbf{e}_k(n) = \Big[ e_k(n) \quad e_k(n-1) \quad \cdots \quad e_k(n-L+1) \Big] \tag{7.48}$$

with $e_k(n)$ defined in (7.12). It is worth noting that each filter in the controller is adapted independently as in (7.14). With respect to the previous approximate method, further partitioned matrices are used, and thus the inversion of $K \cdot J \cdot I$ matrices $L \times L$ are required. As a consequence, it is in general more computationally expensive than the previous method. Moreover, the correlations existing among the filtered-X signals, due to the similarities among the signals collected by the reference microphones and among the impulse responses of the secondary paths, can degrade the convergence behavior of the algorithm. To avoid possible

numerical instabilities, the use of a small leakage factor $\gamma$ is suggested in [40]. The final updating relationship then becomes

$$\mathbf{h}_{ji}(n+1) = (1-\gamma)\mathbf{h}_{ji}(n) - \sum_{k=1}^{K} \mu_k \mathbf{G}_{kji}(n)\left[\mathbf{G}_{kji}^{T}(n)\mathbf{G}_{kji}(n) + \delta\mathbf{I}\right]^{-1}\mathbf{e}_k(n). \tag{7.49}$$

## 7.3. Nonlinear active noise controllers

In this section we review the main nonlinear effects that may influence the behavior of an active control system and thus motivate the use of nonlinear active noise controllers.

### 7.3.1. Nonlinear reference noise

It has been recently noted [45, 49] that the noise generated by a dynamic system can often be modeled as a nonlinear and deterministic process, such as a chaotic process rather than a stochastic one, as usually assumed in the studies on adaptive filters. In particular, it has been shown in [30] that the noise of the fan in a duct is well modeled by a chaotic process. Three kinds of chaotic noises, that is, logistic, Lorenz, and Duffing noises, have been applied in [45] to test the nonlinear controller proposed there. Among those noises, the logistic chaotic noise offers an extremely useful test signal since it is a second-order white and predictable nonlinear process. It is usually generated by the following expression:

$$\xi(n+1) = \lambda\xi(n)(1-\xi(n)), \tag{7.50}$$

where $\lambda = 4$ and $\xi(0)$ is a real number in the range $(0, 1)$. The nonlinear process is, in general, normalized in order to have a unit signal power, that is,

$$x(n) = \frac{\xi(n)}{\sigma_\xi}, \tag{7.51}$$

with $\sigma_\xi = E[\xi^2(n)]$. A segment of 512 samples of a zero mean logistic noise with $\xi(0) = 0.8$ is shown in Figure 7.6 and the corresponding spectrum is shown in Figure 7.7.

The impact of this kind of noise on the controller performance becomes evident when the secondary path between the actuator source and the error sensor is modeled as a nonminimum-phase FIR filter. In fact, the transfer function of the controller that minimizes the square of the error signal is given by $-P \cdot S^{-1}$. If the secondary path is described by a linear and minimum-phase FIR filter, its inverse is represented by a stable causal IIR filter. This filter can be replaced, in general, by an approximating FIR filter of suitable length. However, if the secondary path is described by a nonminimum-phase FIR filter, its unstable IIR inverse requires an approximating noncausal FIR filter. In contrast to other applications where a linear

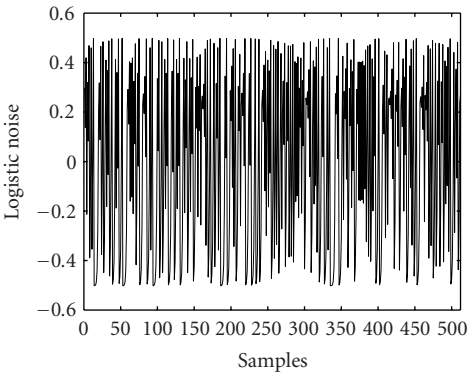



FIGURE 7.6. Logistic noise for $\lambda = 4$ and $\xi(0) = 0.8$.

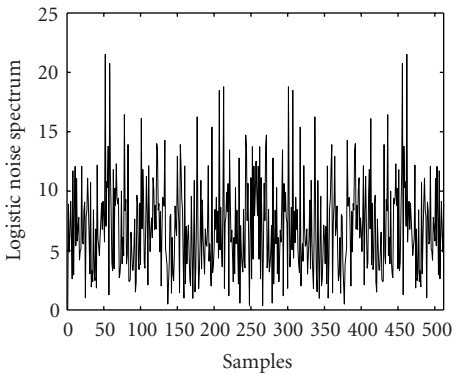



FIGURE 7.7. Spectrum of the logistic noise for $\lambda = 4$ and $\xi(0) = 0.8$.

inverse to a nonminimum-phase system can be provided by accepting a suitable delay, as in adaptive inverse control and adaptive equalization, in this application such an approach is not possible. Nevertheless, if the input signal is stochastic non-Gaussian or deterministic, a zero-delay nonlinear inverse exists. In fact, it has been shown in [45] that the minimum mean square error estimate of the disturbing noise at the error sensor is compensated by two terms involving the parameters of the controller and of the secondary path. The first term is a linear combination of present and past input samples (the causal part) while the second part is a linear combination of future input samples (the noncausal part). This second term can be viewed as a linear combination of multistep predictors acting on the input samples. Therefore, if the input signal is non-Gaussian and is not independent from sample to sample, then the optimal predictor is nonlinear. This predictor can be approximated in practice by using multilayer perceptrons, radial basis functions, or Volterra filters, as discussed in [45, 49].

### 7.3.2. Nonlinear primary path

Another situation in which a nonlinear model is required is when the primary path exhibits some nonlinear distortions. Evidence of this fact can be found, for example, in systems where the noise is propagating with high sound pressure [26] and the nonlinearity of the air is taken into account. It was reported in [26] that a sinusoidal wave at 500 Hz propagating in a duct at a sound pressure of 140 dB SPL produces harmonic distortions of about 1% after traveling a distance of 1 m.

A practical model commonly used to describe a nonlinear primary path is based on quadratic and cubic pointwise nonlinearities. If some additional memory effect described in general by a linear convolution model is present, then the result is a nonlinear model given in terms of quadratic and cubic kernels of the well-known Volterra series [29]. This kind of model demands a nonlinear controller described by a Volterra filter. It is worth mentioning that for these filters it is possible to derive efficient updating algorithms by exploiting the linearity of their output with respect to the kernel coefficients. The derivation of filtered-X algorithms for Volterra filters will be described in detail in the next sections.

Primary paths with cubic nonlinearities and reference noises modeled with a sinusoidal wave corrupted by a Gaussian noise, with a logistic chaotic noise, and with a white Gaussian and not predictable noise, respectively, have been tested in [49] in presence of minimum and nonminimum-phase secondary paths. Improvements of performance in terms of residual MSE have been demonstrated under all these conditions except when the logistic noise is applied and the secondary path is minimum phase. In such a case, in fact, an FIR controller achieves about the same MSE than a second-order or third-order Volterra controller. This result is expected since the reference noise itself contains second- and higher-order terms so that their linear combination due to the FIR filter can approximate a Volterra model. However, when the secondary path exhibits a nonminimum-phase behavior, the performance improvement due to the use of a Volterra controller is confirmed to be significant.

### 7.3.3. Nonlinear secondary path

The effects on nonlinearities in the secondary path have been also studied in the literature. In fact, actual ANC systems use in the secondary paths A/D and D/A converters, power amplifiers, loudspeakers, and transducers. Overdriving the electronics or the loudspeakers gives rise to nonlinear effects. An accurate study of the impact of a nonlinearity in the secondary path has been presented in [13] for the filtered-X LMS algorithm. The analysis is limited, however, to the presence of a zero-memory saturation nonlinearity. The degree of nonlinearity is defined by a steady-state parameter that describes the power limitation in the output of the secondary path due to the nonlinear distortion. Several examples are considered to illustrate the accuracy of the performed analysis for small, moderate, and large degrees of nonlinearity. The analytical and simulation results show that even a small nonlinearity may significantly affect the controller behavior.

Nonlinear paths and nonlinear distortions have been also considered in the framework of the feedback schemes. In particular, in [31] a frequency selective feedback structure has been described. This structure works essentially in the frequency domain and is considered as nonlinear since signal amplitudes and phases are adapted in place of the controller coefficients. It is also shown that in active headphone sets, which still contain microphones, amplifiers, and loudspeakers in the secondary path, nonlinear behaviors have been noticed [31]. In fact, the tests with three frequency components pointed out nonlinear modulation effects. Therefore, it has been concluded that it is necessary to control not only the fundamental harmonic of a disturbance frequency but also the first upper harmonic.

### 7.3.4. Nonlinear models for the controller

The presence of nonlinear effects in the source noise and in the primary and secondary paths motivates the use of a nonlinear controller. The models proposed in recent contributions can be grouped in the following categories.

(i) Neural networks (examples can be found in [2, 3, 12, 30, 35, 36, 44]).
(ii) Radial basis functions (examples can be found in [45]).
(iii) Volterra filters (examples can be found in [7, 40, 41, 45, 48, 49]).
(iv) Frequency selective filters, (examples can be found in [31, 47]).
(v) Fuzzy systems (examples can be found in [10, 11, 43]).

## 7.4. A class of nonlinear feedforward active noise controllers

The updating algorithms described in Section 7.2 can be easily adapted to deal with the class of nonlinear filters characterized by a linear relationship of the filter output with respect to the filter coefficients. Specifically, such a class is defined by the input-output relationship

$$y(n) = \mathbf{h}^T(n)\mathbf{x}(n), \tag{7.52}$$

where $\mathbf{h}(n)$ is the vector formed by the filter coefficients and $\mathbf{x}(n)$ is expressed as a vector function of the input samples, that is,

$$\mathbf{x}(n) = \begin{bmatrix} f_1\{x\} & f_2\{x\} & \cdots & f_M\{x\} \end{bmatrix}^T, \tag{7.53}$$

where $f_r\{\cdot\}$, $r = 1, \ldots, M$, is a time-invariant functional of its argument. In addition to linear filters, (7.52) and (7.53) include truncated Volterra filters of any order $p$ as well as structures based on trigonometric functional expansions [14] and other nonlinear functionals as, for example, saturating nonlinearities. It will be shown in this section that for this class of filters the updating rules presented in Section 7.2 require only a redefinition of the coefficient vectors and of the corresponding input vectors. Appropriate arrangements need to be found, according to the specific form of the input-output relationship. To this purpose, Volterra filters and structures based on functional expansions will be first briefly described

and then the formulations of the updating rules for nonlinear single-channel and multichannel controllers will be discussed.

### 7.4.1. Polynomial and Volterra filters

Discrete-time causal polynomial filters [29] are described by the input-output relationship

$$y(n) = \sum_{p=0}^{P} g_p\big(x(n), x(n-1), \ldots, x(n-N_x+1), y(n-1), \ldots, y(n-N_y+1)\big), \tag{7.54}$$

where $g_p(\cdot)$ is a polynomial of order $p$ in the variables within the parentheses. The linear filter is a particular case of polynomial filters since the relation in (7.54) becomes linear if $g_p(\cdot) = 0$ for all $p \neq 1$. With reference to (7.54), polynomial filters can be classified into recursive and nonrecursive filters. Recursive filters are characterized by possibly nonlinear feedback terms in their input-output relationships. They possess an infinite memory, as linear filters with infinite impulse response. A simple example of a recursive filter is the bilinear filter represented by the following equation:

$$y(n) = \sum_{i=0}^{N_1-1} a_i x(n-i) + \sum_{j=1}^{N_2-1} b_j y(n-j) + \sum_{i=0}^{N_3-1} \sum_{j=1}^{N_4-1} c_{ij} x(n-i) y(n-j). \tag{7.55}$$

In consideration of its infinite memory, a recursive polynomial filter admits, within some stability constraints, a convergent Volterra series expansion [29] of the form

$$\begin{aligned} y(n) = h_0 &+ \sum_{m_1=0}^{\infty} h_1(m_1) x(n-m_1) \\ &+ \sum_{m_1=0}^{\infty} \sum_{m_2=0}^{\infty} h_2(m_1, m_2) x(n-m_1) x(n-m_2) + \cdots \\ &+ \sum_{m_1=0}^{\infty} \sum_{m_2=0}^{\infty} \cdots \sum_{m_p=0}^{\infty} h_p(m_1, m_2, \ldots, m_p) \\ &\quad x(n-m_1) x(n-m_2) \cdots x(n-m_p) + \cdots, \end{aligned} \tag{7.56}$$

where $h_p(m_1, m_2, \ldots, m_p)$ denotes the $p$th-order *Volterra kernel* of the nonlinear filter.

Nonrecursive polynomial filters, commonly called Volterra filters, are characterized by input-output relationships that result from a double truncation of the Volterra series, that is, a memory truncation, by limiting the memory of the filters to a finite number of elements in the summations of (7.56) and an order truncation by limiting the number of Volterra kernels. As a consequence, a Volterra filter

of order $p$ is described by the equation

$$\begin{aligned} y(n) = h_0 &+ \sum_{m_1=0}^{N_1-1} h_1(m_1)x(n-m_1) \\ &+ \sum_{m_1=0}^{N_2-1}\sum_{m_2=0}^{N_2-1} h_2(m_1,m_2)x(n-m_1)x(n-m_2) + \cdots \\ &+ \sum_{m_1=0}^{N_p-1}\sum_{m_2=0}^{N_p-1}\cdots\sum_{m_p=0}^{N_p-1} h_p(m_1,m_2,\ldots,m_p) \\ &\quad x(n-m_1)x(n-m_2)\cdots x(n-m_p). \end{aligned} \tag{7.57}$$

A Volterra filter characterized by a single nonzero kernel is called a homogeneous Volterra filter.

The simplest nonlinear filter belonging to the class of filters in (7.57) is the quadratic filter obtained by limiting the terms to the linear and the second-order kernels. Using a vector-matrix representation, the filter output can be expressed as

$$y(n) = \mathbf{h}_1^T\mathbf{x}_1(n) + \mathbf{x}_2^T(n)\mathbf{H}_2\mathbf{x}_2(n), \tag{7.58}$$

where $\mathbf{x}_1(n)$ and $\mathbf{x}_2(n)$ are the vectors of the most recent $N_1$ and $N_2$ input samples, respectively, $\mathbf{h}_1$ is the vector formed by the $N_1$ coefficients of the linear filter and $\mathbf{H}_2$ is the $N_2 \times N_2$ matrix representing the second-order Volterra kernel. Without loss of generality, the second-order Volterra kernel can be assumed as symmetric and thus the matrix $\mathbf{H}_2$ is symmetric too. For a more efficient realization, $\mathbf{H}_2$ can be modified to assume the upper triangular form [29, page 35]. This representation is the basis of the implementation of the quadratic filter as a multichannel filter bank. This structure is frequently implemented in current applications and is also used in the field of nonlinear active noise control to derive efficient adaptation algorithms, as those shown in the next sections. It is essentially based on the so-called diagonal representation [37] that allows a truncated Volterra system of any order $p$ to be described by a "diagonal" arrangement of the entries of each one of its kernels. In fact, if the $p$th-order kernel is represented as a sampled hypercube of the same order, then the diagonal representation implies the change of the Cartesian coordinates to coordinates that are aligned along the diagonals of the hypercube. In this way, a homogeneous Volterra filter of any order $p$ can be implemented by a filter bank where each channel contains an FIR filter formed with the coefficients of the diagonals of the hypercube. The input signal is formed by the corresponding products of $p$ input samples. Figure 7.8 shows the multichannel implementation of a homogeneous quadratic filter. It is worth noting that this structure is computationally efficient since the products of input samples computed at the present time instant are then suitably delayed across the FIR structures. Then, a complete Volterra filter can be realized using the parallel connection

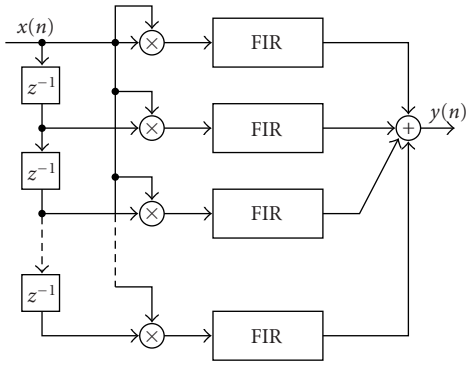



FIGURE 7.8. Multichannel implementation of a homogeneous quadratic filter.

of the filter banks implementing each kernel. This structure can be described in the form of (7.52) where the complete vector $\mathbf{h}$ of the filter coefficients is obtained by arranging in sequence the elements of the linear kernel, those of the diagonals of the quadratic kernel, and so on till the kernel of order $p$. The input vector is derived by using the corresponding samples and products of input samples. As an example, in the case of a quadratic filter, the complete input vector $\mathbf{x}(n)$ is a collection of samples and products of two samples of the input as shown by the following expression:

$$\begin{aligned}\mathbf{x}(n) = \Big[ & x(n) \quad x(n-1) \quad \cdots \quad x(n-N_1+1) \\ & x^2(n) \quad x^2(n-1) \quad \cdots \quad x^2(n-N_2+1) \\ & x(n)x(n-1) \quad \cdots \quad x(n-N_2+D_2)x(n-N_2+1)\Big]^T,\end{aligned} \tag{7.59}$$

where $N_1$ is the memory length of the linear kernel, $N_2$ is the memory length of the quadratic kernel, and $D_2$ is the number of implemented diagonals of the quadratic kernel. The number of elements in $\mathbf{x}(n)$ and $\mathbf{h}$ is $M = N_1 + D_2 \cdot N_2 - D_2 \cdot (D_2 - 1)/2$.

It is worth noting that another powerful tool for the analysis of Volterra filters has been recently introduced in [5], the so-called V-vector algebra. This algebra has been used in [5] to make feasible the extension of adaptation algorithms used for linear filters to Volterra filters. The main element of this algebra is the V-vector which is defined as a nonrectangular matrix whose rows are arranged in a decreasing length order. For what concerns the present discussion, it is sufficient to point out that the rows of the V-vectors simply correspond to the diagonals of the Volterra kernels and to the corresponding products of input samples, respectively, as in the diagonal representation. Therefore, the channels are again modeled as FIR filters but they are now hierarchically arranged according to the number of the respective filter coefficients [41].

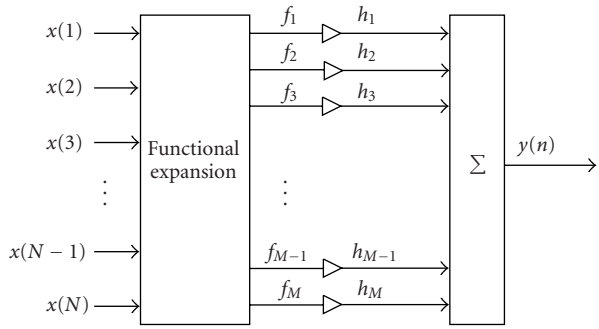



FIGURE 7.9. Schematic diagram of a FLANN.

### 7.4.2. Structures based on functional expansions

The functional-link artificial neural network (FLANN) [35] has been recently proposed in the field of neural networks as an alternative system capable of simplifying the learning algorithms and offering simple structures for hardware implementations. More recently, this structure has been applied in active noise control applications [14]. The schematic diagram of a FLANN is depicted in Figure 7.9. It is essentially based on the functional expansions $f_r\{\cdot\}$ of the input signal $x(n)$. Therefore, it is a member of the nonlinear class described by (7.52) and (7.53). The functional expansions suggested in [14] have been orthogonal polynomials, as for example Legendre, Chebyshev, and trigonometric polynomials. Among them, the trigonometric basis functions of order $P$, given by

$$\{x, \sin(\pi x), \cos(\pi x), \sin(2\pi x), \cos(2\pi x), \ldots, \sin(P\pi x), \cos(P\pi x)\}, \tag{7.60}$$

provide a compact and efficient representation in the mean square sense [14]. Therefore, if $N$ is the length of the input vector $\mathbf{x}(n)$ then the length of the vector representing its functional expansion is $M = N \cdot (2P + 1)$ since any input sample is expanded using (7.60).

As an example, let us now consider the simple case of a first-order trigonometric expansion, that is, $P = 1$. With reference to (7.53), the input signal is given by

$$\begin{aligned}\mathbf{x}(n) = \Big[ & x(n) \quad x(n-1) \quad \cdots \quad x(n-N+1) \\ & \sin(\pi x(n)) \quad \sin(\pi x(n-1)) \quad \cdots \quad \sin(\pi x(n-N+1)) \\ & \cos(\pi x(n)) \quad \cos(\pi x(n-1)) \quad \cdots \quad \cos(\pi x(n-N+1)) \Big]^T.\end{aligned} \tag{7.61}$$

The corresponding implementation is shown in Figure 7.10. The memory of the three FIR filters is equal to $N$ so that the total number of coefficients of the FLANN structure is $M = 3N$. This structure shares with that of Figure 7.8 the characteristic of being efficient since the elements of the functionals $f_r\{\cdot\}$ are computed using

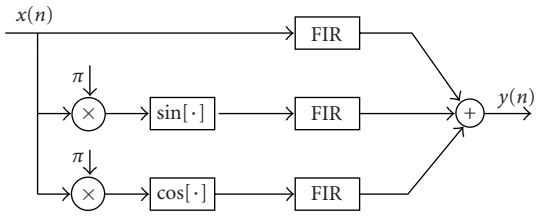



FIGURE 7.10. Implementation of a trigonometric functionally expanded network with $P = 1$.

the input sample at the present time instant and then the corresponding delayed terms are obtained through the FIR filters.

### 7.4.3. Single-channel active noise controllers

A single-channel nonlinear filtered-X LMS algorithm based on a Volterra model has been first proposed in [48, 49], where a multichannel implementation for a quadratic controller is analyzed, by exploiting the diagonal coordinate system proposed in [37]. As mentioned in [49], extensions of this approach to Volterra filters of any order $P$ are quite straightforward.

A single-channel nonlinear filtered-X AP algorithm based on a Volterra model has been first described in [7]. A description similar to the diagonal coordinate representation has been used for the so-called simplified Volterra filters (SVF). The homogeneous quadratic filter has been specifically considered. An SVF is still implemented as a filter bank but here the emphasis is on the fact that, according to the characteristics of many measured second-order kernels, it is possible to use a reduced number of active channels.

The extension of the filtered-X AP algorithms to higher-order Volterra filters has been discussed in [41] where the V-vector algebra [5] has been exploited. This algebra can directly lead to the implementation of Volterra filters of any order $P$ in the form of a multichannel filter bank. The channels are again modeled as FIR filters but they are now hierarchically arranged according to the number of the respective filter coefficients. In such a way, it is possible to devise models of reduced complexity by cutting the last and usually less relevant channels. The specific derivations and arrangements required in the single-channel case to illustrate the application of the diagonal or V-vector representations are deferred to the next subsection since they can be viewed as a particular case of the general multichannel configuration.

FLANN structures for single-channel ANC based on trigonometric expansions have been studied in [14] where the normalized LMS algorithm is used and a fast implementation is proposed too.

### 7.4.4. Multichannel active noise controllers

While linear multichannel filtered-X LMS and AP algorithms have been studied for many years [4, 15, 17], a nonlinear multichannel filtered-X AP algorithm for

controllers based on filters belonging to the class in (7.53) has been only recently proposed in [40]. A quadratic Volterra model has been used and an approximate AP algorithm has been studied. The main definitions used to derive the adaptation algorithm described in [40] are reported here since they are useful to illustrate the redefinitions that permit to extend the applicability of the adaptation algorithms described in Section 7.2 to nonlinear multichannel ANC schemes. These definitions illustrate in practice the concepts implied by the diagonal and V-vector representations.

We will start dealing with a multichannel scheme using $I$ reference microphones, $J$ actuators, and $K$ error microphones. It is worth noting that definitions and arrangements required in the single-channel case can be easily inferred by the multichannel case by assuming $I = 1$, $J = 1$, and $K = 1$. Let us first refer to a homogeneous quadratic filter described by means of its triangular representation [29, page 35]. Thus, using the filter bank structure of Figure 7.8 in each of the cross-paths of the adaptive controller in Figure 7.5, it is possible to write any output $y_j(n)$, $1 \le j \le J$, from the multichannel quadratic controller as

$$y_j(n) = \sum_{i=1}^{I} \sum_{m=1}^{M} y_{jim}(n). \tag{7.62}$$

In this equation $M$ is the number of channels actually used, with $M \le N$, where $N$ is the memory length of the quadratic filter connecting the input $i$ to the output $j$ of the controller. The variable $y_{jim}(n)$ represents the output from the channel $m$, $1 \le m \le M$ and is defined as

$$y_{jim}(n) = \sum_{r=0}^{N-m} h_{ji}(n; r, r+m-1) x_i(n-r) x_i(n-r-m+1), \tag{7.63}$$

where $h_{ji}(n; \cdot, \cdot)$ denotes the coefficients of the quadratic kernel at time $n$. Using a vector notation, (7.63) becomes

$$y_{jim}(n) = \mathbf{h}_{jim}^T(n) \mathbf{x}_{im}(n), \tag{7.64}$$

where $\mathbf{h}_{jim}(n)$ is the vector formed by the $N-m+1$ coefficients of the $m$th channel at time $n$,

$$\mathbf{h}_{jim}(n) = \begin{bmatrix} h_{ji}(n; 0, m-1) & h_{ji}(n; 1, m) & \cdots & h_{ji}(n; N-m, N-1) \end{bmatrix}^T. \tag{7.65}$$

The corresponding input vector $\mathbf{x}_{im}(n)$, again formed by $N - m + 1$ entries, is

defined as

$$\mathbf{x}_{im}(n) = \begin{bmatrix} x_i(n)x_i(n-m+1) \\ x_i(n-1)x_i(n-m) \\ \vdots \\ x_i(n-N+m)x_i(n-N+1) \end{bmatrix}. \tag{7.66}$$

Let us now define two vectors of $Q = \sum_{q=1}^{M}(N-q+1)$ elements

$$\begin{aligned} \mathbf{h}_{ji}(n) &= \begin{bmatrix} \mathbf{h}_{ji1}^T(n) & \mathbf{h}_{ji2}^T(n) & \cdots & \mathbf{h}_{jiM}^T(n) \end{bmatrix}^T, \\ \mathbf{x}_i(n) &= \begin{bmatrix} \mathbf{x}_{i1}^T(n) & \mathbf{x}_{i2}^T(n) & \cdots & \mathbf{x}_{iM}^T(n) \end{bmatrix}^T \end{aligned} \tag{7.67}$$

formed, respectively, with the vectors $\mathbf{h}_{jim}(n)$ and $\mathbf{x}_{im}(n)$ related to the single channels of the filter bank. Thus, the output of the homogeneous quadratic filter connecting the input $i$ to the output $j$ of the controller can be written as

$$y_j(n) = \sum_{i=1}^{I} \mathbf{h}_{ji}^T(n)\mathbf{x}_i(n). \tag{7.68}$$

The scalars $y_j(n)$, $1 \le j \le J$, and the vectors $\mathbf{x}_i(n)$ and $\mathbf{h}_{ji}(n)$, $1 \le i \le I$ and $1 \le j \le J$, defined above play now the role of the corresponding scalar and vectors used in Section 7.2.5 to derive the linear multichannel filtered-X AP algorithms. Therefore, by redefining all the other vectors and matrices used in Section 7.2.5 making use of these scalar and vectors, the same final updating rules given by (7.29) and (7.30) formally still apply. In addition, the two approximate relationships of (7.45) and (7.49) can be applied too. The updating rule of (7.49) has been explicitly derived in [40], where it has been also noted that it includes

(i) for $L = 1$ and $I = J = K = 1$, the single-channel filtered-X LMS algorithm for linear controllers given in [15], and that for quadratic controllers given in [49], apart from the normalization factor used here;
(ii) for $L > 1$ and $I = J = K = 1$, the single-channel filtered-X AP algorithm for quadratic controllers proposed in [7];
(iii) for $L = 1$ and $I > 1$ or $J > 1$ or $K > 1$, the multichannel filtered-X LMS algorithm for linear controllers given in [15], again apart from the normalization factor used here;
(iv) for $L \ge 1$ and $I > 1$ or $J > 1$ or $K > 1$, the multichannel filtered-X approximate AP algorithm for quadratic controllers.

Moreover, the exact AP updating rule and the approximate rule of (7.45) have been presented in [8, 9] together with a theoretical analysis of their transient and steady-state behaviors. In [8, 9] it has been proven that the filtered-X AP algorithms provide a biased estimate of the minimum mean square solution of multichannel ANC problem. However, since in many cases the bias is small, AP algorithms can be efficiently applied in nonlinear ANC problems.

In the case of a nonhomogeneous quadratic filter, the linear term can be considered as an additional channel. The vectors $\mathbf{h}_{ji}(n)$ and $\mathbf{x}_i(n)$ are modified by inserting on the top the coefficients of the linear filter and the samples of the input signal, respectively, as shown for a single filter in (7.59). Similar considerations apply to higher-order Volterra kernels, and thus to Volterra filters of any order $P$, again exploiting the principles of the diagonal coordinate or V-vector representations [41].

Finally, as noted in the single-channel case, the class of nonlinear filters described by (7.52) and (7.53) includes also in the multichannel case other useful structures, for example, those based on functional expansions. The study of these structures for multichannel ANC is presently under development. It can be anticipated that since they belong to the same class of nonlinear filters, adaptation algorithms of the type proposed for Volterra filters can be derived for these structures too, and in fact an example is given in the next section using an AP adaptation algorithm.

## 7.5. Simulation results

In this section, a couple of applications of nonlinear multichannel active noise controllers belonging to the class of nonlinear filters described by (7.52), (7.53) is presented and commented.

### 7.5.1. Experiment 1

As for single channel active noise controllers, a nonlinear controller can be required also in the multichannel case if the input noise is a nonlinear and deterministic process of a chaotic rather than stochastic nature. As mentioned in Section 7.3, such a noise can be, for example, modeled with the logistic noise that is a second-order white and predictable nonlinear process generated by (7.50). In the following example $\xi(0)$ is chosen equal to 0.9. In the multichannel case we are considering that the controller has one input ($I = 1$) and two outputs ($J = 2$) and two error microphones ($K = 2$) are used. The nonlinear process has been normalized in order to have unit signal power as in (7.51). A Gaussian noise with SNR $= -30$ dB has been added to the logistic noise at the input and a Gaussian noise with SNR $= -40$ dB has been added to the two signals sensed at the error microphones. The transfer functions of the primary and secondary paths are given as follows:

$$\begin{aligned} P_{11}(z) &= z^{-4} - 0.3z^{-5} + 0.2z^{-6}, \\ P_{21}(z) &= z^{-4} - 0.2z^{-5} + 0.1z^{-6}, \\ S_{11}(z) = z^{-2} + 1.5z^{-3} - z^{-4}, &\qquad S_{12}(z) = z^{-2} + 1.3z^{-3} - z^{-4}, \\ S_{21}(z) = z^{-2} + 1.3z^{-3} - z^{-4}, &\qquad S_{22}(z) = z^{-2} + 1.2z^{-3} - z^{-4}. \end{aligned} \tag{7.69}$$

The exact solution of the multichannel ANC problem requires the inversion of the $2 \times 2$ matrix $\mathbf{S}$ formed with the transfer functions $S_{kj}$ in (7.69). The inverse

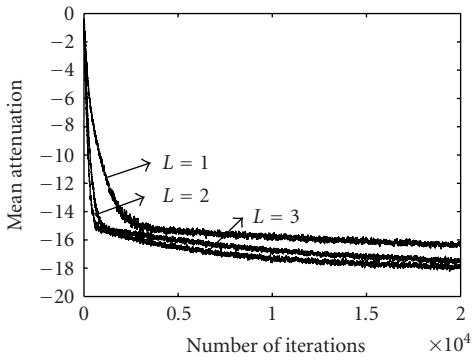



FIGURE 7.11. Mean attenuation at the error microphones in a multichannel active noise controller using quadratic filters.

matrix $\mathbf{S}^{-1}$ is thus formed with IIR transfer functions where the poles are given by the roots of the determinant of $\mathbf{S}$. It is easy to verify that in our example there is a root outside the unit circle so that the IIR transfer functions are unstable and the approximating FIR filters in the controller need to be noncausal. Nevertheless, as in the single-channel case, if the input noise is a nonlinear and deterministic process, a zero-delay nonlinear inverse matrix exists. The system is identified using two second-order Volterra filters with linear and quadratic parts of memory length $N = 8$. The quadratic kernels include only two channels ($M = 2$) corresponding to the principal diagonal and the adjacent one in the triangular representation. The adaptation algorithm used is the approximate rule of (7.49). In the experiment both the step sizes of the linear and quadratic parts have been fixed equal to $3 \times 10^{-3}$, the leakage factor $\gamma$ has been set equal to zero, and the constant $\delta$ has been set equal to $10^{-3}$. Figure 7.11 shows the ensemble average of the mean attenuation at the error microphones for 50 runs of the simulation system. The three curves refer to different values of the AP order $L$. The order $L = 1$ corresponds to an NLMS adaptation algorithm. For higher orders of AP the improvement can be easily appreciated.

To further validate these results, a few tests have been performed using only linear filters in the controller. If we use FIR filters of complexity equivalent to that of the quadratic filters, that is, with 23 coefficients, the diagrams shown in Figure 7.12 are obtained. These results are derived by assuming the same experimental conditions as before, but with an optimized step size set equal to $6 \times 10^{-3}$. The comparison of the diagrams in Figure 7.11 with those in Figure 7.12 demonstrates that second-order Volterra filters offer better performance than FIR filters of equivalent complexity in the experimental environment considered.

### 7.5.2. Experiment 2

The experimental setup is the same as in the previous example. The ensemble averages of the mean attenuation at the error microphones for 50 runs and $L = 1, 2, 3$

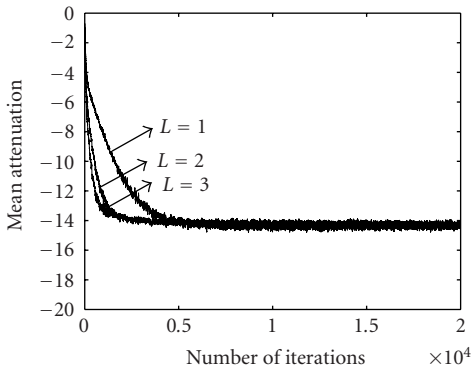



FIGURE 7.12. Mean attenuation at the error microphones in a multichannel active noise controller using linear filters.

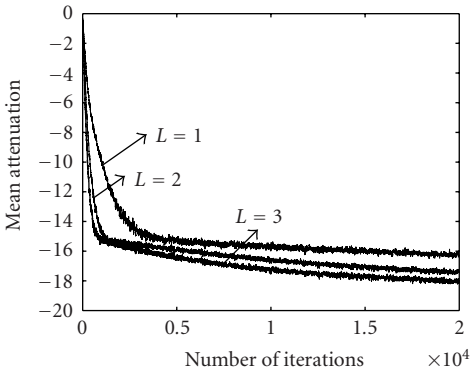



FIGURE 7.13. Mean attenuation at the error microphones in a multichannel active noise controller using quadratic filters.

are reported in Figure 7.13 for a controller using two quadratic filters with linear and quadratic parts of memory length $N = 8$. The quadratic kernels include only two channels ($M = 2$) corresponding to the principal diagonal and the adjacent one in the triangular representation. Therefore, the total number of coefficients is equal to 23. The adaptation algorithm used is the approximate AP rule of (7.45). The step sizes are equal to $5 \times 10^{-3}$ for both the linear and quadratic parts, and the constant $\delta$ has been set equal to $10^{-3}$. Again, the improvement offered by the AP method can be easily appreciated.

Figure 7.14 shows the ensemble averages of the mean attenuation at the error microphones for 50 runs and $L = 1, 2, 3$ for the same system using a trigonometric functional expansion with $P = 1$. The length of the FIR filters has been fixed equal to 8 so that the total number of coefficients is equal to 24. The step size has been fixed equal to $5 \times 10^{-3}$. The diagrams in Figure 7.14 show the good convergence behavior and the effect of the AP order.

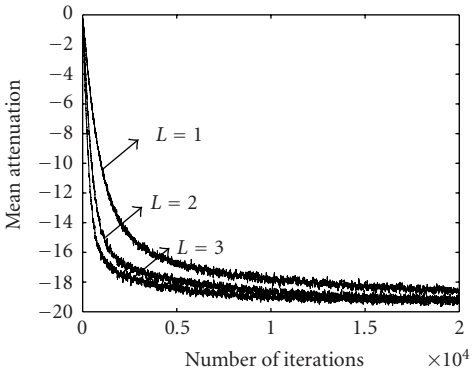



FIGURE 7.14. Mean attenuation at the error microphones in a multichannel active noise controller using a trigonometric functional expansion.

## 7.6. Current work and future developments

Current work includes the search for efficient implementations for nonlinear multichannel filtered-X LMS and AP algorithms. A recently considered strategy, applied in [42] to quadratic controllers, is based on the use of different methods for partial update. The so-called partial errors (PE) technique consists of using in the coefficient update only one of the $K$ error vectors $\mathbf{e}_k(n)$ at the time. In the block partial updates (PU) technique, each measurement of the error vectors $\mathbf{e}_k(n)$ is used to adapt a block of coefficients. The set membership filtering (SMF) concept is also adopted. In this case, the filter coefficients are updated sparsely in time, according to the following rule:

$$\mathbf{h}_{ji}(n+1) = \begin{cases} \text{Update } \mathbf{h}_{ji}(n) & \text{if } |e_{k,1}(n)| > e_b, \\ \text{No update} & \text{otherwise} \end{cases} \tag{7.70}$$

for a suitable choice of the threshold error $e_b$ [54]. All these methods have been studied and implemented in [42] for quadratic filters. Figures 7.15 and 7.16 show a few results for the experimental setup of the previous section. In particular, Figure 7.15 shows the ensemble average of the mean attenuation at the error microphones for 50 runs of the simulation system using the updating rule of (7.49) and the PE-PU-SMF techniques with an error bound $e_b = 0.1$. Figure 7.16 shows the corresponding results obtained for $e_b = 0.2$. The step sizes have been optimized and are chosen equal to $5 \times 10^{-2}$ for both the linear and quadratic parts.

In the PU technique, the coefficients of each quadratic filter in the controller have been subdivided in 3 blocks of 8, 8, and 7 coefficients, respectively. Comparing these results with those in Figure 7.11, it is easy to see that the convergence behavior is only slightly degraded while the number of operations is strongly reduced. In fact, using the PE-PU strategies, the computational load is approximately reduced by a factor $2 \times 6$, since $K = 2$ and 3 blocks of 8, 8, 7 coefficients are

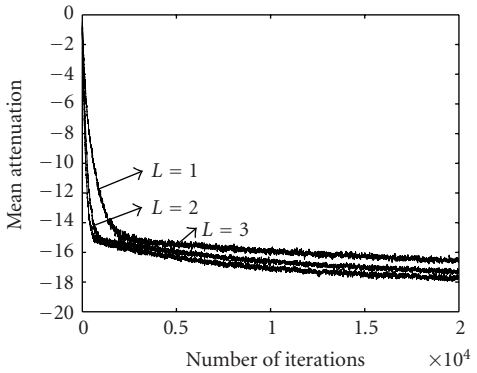



FIGURE 7.15. Mean attenuation at the error microphones with error bound $e_b = 0.1$.

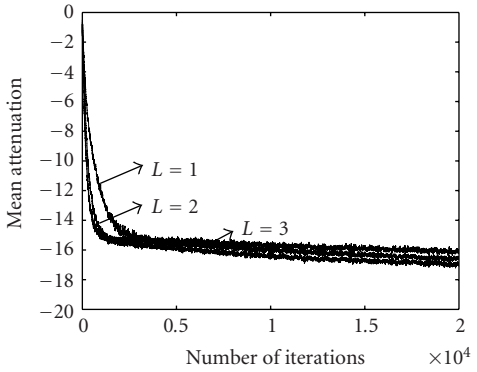



FIGURE 7.16. Mean attenuation at the error microphones with error bound $e_b = 0.2$.

used for each quadratic filter. Then, a further reduction ranging for $L = 3$ from 2 to 5 is obtained according to the SMF strategy, as shown in Table 7.1.

Another possibility for reducing the computational complexity is employed in [15] where a new efficient implementation of the single-channel filtered-X LMS algorithm is derived exploiting computational reduction techniques developed for fast projection algorithms [25, 50]. The same techniques could be applied to the derivation of efficient implementations of the algorithms presented in this chapter.

A topic recently investigated is the transient and steady-state analysis of exact and approximate multichannel filtered-X AP algorithms of the type presented in Section 7.2 for the class of nonlinear filters described in Section 7.4. Conditions for the stability of the multichannel adaptive filtered-X AP algorithms have been derived in [9]. Accurate expressions of the asymptotic mean square error and mean standard deviation of the filter coefficients have been derived too. One of the results is that filtered-X AP algorithms always provide a biased estimate of the minimum mean square solution. Nevertheless, in many cases the bias is small

Table 7.1. Average number of updates in 20 000 iterations for the PE-PU-SMF algorithm.

| $e_b$ | $L = 1$ | $L = 2$ | $L = 3$ |
|---|---|---|---|
| 0.1000 | 11 721 | 10 723 | 10 089 |
| 0.2000 | 4901 | 4287 | 3840 |

and thus these algorithms can be profitably applied to active noise control. The analysis developed in [9] is based on the methodology presented in [46] and can be adapted for studying the convergence properties of other filtered-X algorithms, allowing an accurate prediction of the actual behavior of the adaptive controllers.

Future perspectives in this area include further studies on

(i) the optimal step size control,
(ii) the analysis of the effects due to the correlations among channels,
(iii) the steady-state and transient analysis of adaptation rules using partial update methods,
(iv) the analysis of models with uncertain or not known secondary paths,
(v) the analysis of models with nonlinearities with memory in the secondary paths.

## 7.7. Summary

In this chapter, a description of the ANC scenario has been given with reference, in particular, to the feedforward schemes and the filtered-X adaptation algorithms. The nonlinear effects that may influence the behavior of ANC systems have been reviewed subsequently to motivate the use of nonlinear controllers. The filtered-X LMS and AP adaptation algorithms have been explicitly derived for the linear single-channel and multichannel schemes. Then, the point of view adopted throughout the rest of the chapter is to show how these algorithms can be profitably extended to the class of nonlinear active noise controllers whose filters are characterized by a linear relationship of their output with respect to the filter coefficients. This class of filters includes, of course in addition to linear filters, truncated Volterra filters of any order $p$, FLANN structures and other nonlinear functional expansions. It is shown that, basically, the adaptation algorithms remain the same as in the linear case and only a suitable redefinition of the coefficient vectors and of the corresponding input vectors is required. A few experimental results for nonlinear multichannel active noise controllers belonging to this class have been presented and commented. Finally, current work and future research lines have been briefly exposed.

## Bibliography

[1] E. Bjarnason, "Analysis of the filtered-X LMS algorithm," *IEEE Transactions on Speech and Audio Processing*, vol. 3, no. 6, pp. 504–514, 1995.

[2] M. Bouchard, B. Paillard, and C. T. L. Dinh, "Improved training of neural networks for the nonlinear active control of sound and vibration," *IEEE Transactions on Neural Networks*, vol. 10, no. 2, pp. 391–401, 1999.

[3] M. Bouchard, "New recursive-least-squares algorithms for nonlinear active control of sound and vibration using neural networks," *IEEE Transactions on Neural Networks*, vol. 12, no. 1, pp. 135–147, 2001.

[4] M. Bouchard, "Multichannel affine and fast affine projection algorithms for active noise control and acoustic equalization systems," *IEEE Transactions on Speech and Audio Processing*, vol. 11, no. 1, pp. 54–60, 2003.

[5] A. Carini, E. Mumolo, and G. L. Sicuranza, "V-vector algebra and its application to Volterra-adaptive filtering," *IEEE Transactions on Circuits and Systems—Part II: Analog and Digital Signal Processing*, vol. 46, no. 5, pp. 585–598, 1999.

[6] A. Carini, E. Mumolo, and G. L. Sicuranza, "V-vector algebra and Volterra filters," in *Advances in Imaging and Electronic Physics*, P. W. Hawkes, Ed., vol. 124, Academic Press, San Diego, Calif, USA, 2002.

[7] A. Carini and G. L. Sicuranza, "Filtered-X affine projection algorithms for active noise control using Volterra filters," *EURASIP Journal on Applied Signal Processing*, vol. 2004, no. 12, pp. 1841–1848, 2004, Special issue on Nonlinear Signal and Image Processing—Part I.

[8] A. Carini and G. L. Sicuranza, "Steady-state and transient analysis of multichannel filtered-X affine projection algorithms," in *Proceedings of IEEE International Conference on Acoustics, Speech, Signal Processing (ICASSP '05)*, Philadelphia, Pa, USA, March 2005.

[9] A. Carini and G. L. Sicuranza, "Transient and steady-state analysis of filtered-X affine projection algorithms," *IEEE Transactions on Signal Processing*, vol. 54, no. 2, pp. 665–678, 2006.

[10] C.-Y. Chang and K.-K. Shyu, "Active noise cancellation with a fuzzy adaptive filtered-X algorithm," *IEE Proceedings Circuits, Devices and Systems*, vol. 150, no. 5, pp. 416–422, 2003.

[11] J. M. Conchinha, C. A. Silva, J. M. Sousa, M. Ayala Botto, and J. M. G. Sá da Costa, "Acoustic noise modeling and identification using neural and fuzzy techniques," in *Proceedings of International Conference on Noise and Vibration Engineering (ISMA25)*, pp. 825–832, Leuven, Belgium, September 2000.

[12] J. M. Conchinha, M. Ayala Botto, J. M. Sousa, and J. M. G. Sá da Costa, "The use of neural network models in an active noise control applications," in *Proceedings of 4th European Conference on Noise Control (EURONOISE '01)*, pp. 1–13, Patras, Greece, January 2001.

[13] M. H. Costa, J. C. M. Bermudez, and N. J. Bershad, "Stochastic analysis of the filtered-X LMS algorithm in systems with nonlinear secondary paths," *IEEE Transactions on Signal Processing*, vol. 50, no. 6, pp. 1327–1342, 2002.

[14] D. P. Das and G. Panda, "Active mitigation of nonlinear noise processes using a novel filtered-s LMS algorithm," *IEEE Transactions on Speech and Audio Processing*, vol. 12, no. 3, pp. 313–322, 2004.

[15] S. C. Douglas, "Fast implementations of the filtered-X LMS and LMS algorithms for multichannel active noise control," *IEEE Transactions on Speech and Audio Processing*, vol. 7, no. 4, pp. 454–465, 1999.

[16] S. J. Elliott, I. Stothers, and P. A. Nelson, "A multiple error LMS algorithm and its application to the active control of sound and vibration," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 35, no. 10, pp. 1423–1434, 1987.

[17] S. J. Elliott and P. A. Nelson, "Multichannel active sound control using adaptive filtering," in *Proceedings of IEEE International Conference on Acoustics, Speech, Signal Processing (ICASSP '88)*, vol. 5, pp. 2590–2593, New York, NY, USA, April 1988.

[18] S. J. Elliott and P. A. Nelson, "Active noise control," *IEEE Signal Processing Magazine*, vol. 10, no. 4, pp. 12–35, 1993.

[19] S. J. Elliott, *Signal Processing for Active Control*, Academic Press, London, UK, 2000.

[20] L. J. Eriksson, "Development of the filtered-U algorithm for active noise control," *Journal of the Acoustical Society of America*, vol. 89, no. 1, pp. 257–265, 1991.

[21] P. L. Feintuch, "An adaptive recursive LMS filter," *Proceedings of the IEEE*, vol. 64, no. 11, pp. 1622–1624, 1976.

[22] A. Fermo, A. Carini, and G. L. Sicuranza, "Low complexity nonlinear adaptive filters for acoustic echo cancellation," *European Transactions on Telecommunications*, vol. 14, no. 2, pp. 161–169, 2003.

[23] R. Fraanje, M. Verhaegen, and N. Doelman, "Convergence analysis of the filtered-U LMS algorithm for active noise control in case perfect cancellation is not possible," *Signal Processing*, vol. 83, no. 6, pp. 1239–1254, 2003.
[24] C. R. Fuller and A. H. von Flotow, "Active control of sound and vibration," *IEEE Control Systems Magazine*, vol. 15, no. 6, pp. 9–19, 1995.
[25] S. L. Gay and S. Tavathia, "The fast affine projection algorithm," in *Proceedings of IEEE International Conference on Acoustics, Speech, Signal Processing (ICASSP '95)*, vol. 5, pp. 3023–3026, Detroit, Mich, USA, May 1995.
[26] W. Klippel, "Active attenuation of nonlinear sound," U.S. Patent 6 005 952, December 21, 1999.
[27] S. M. Kuo and D. R. Morgan, *Active Noise Control Systems: Algorithms and DSP Implementations*, John Wiley & Sons, New York, NY, USA, 1996.
[28] P. Leug, "Process of silencing sound oscillations," U.S. Patent no. 2,043,416, 1936.
[29] V. J. Mathews and G. L. Sicuranza, *Polynomial Signal Processing*, John Wiley & Sons, New York, NY, USA, 2000.
[30] T. Matsuura, T. Hiei, H. Itoh, and K. Torikoshi, "Active noise control by using prediction of time series data with a neural network," in *Proceedings of IEEE International Conference on Systems, Man and Cybernetics (SMC '95)*, vol. 3, pp. 2070–2075, Vancouver, BC, Canada, October 1995.
[31] T. Meurers, S. M. Veres, and S. J. Elliott, "Frequency selective feedback for active noise control," *IEEE Control Systems Magazine*, vol. 22, no. 4, pp. 32–41, 2002.
[32] P. A. Nelson and S. J. Elliott, *Active Control of Sound*, Academic Press, London, UK, 3rd edition, 1995.
[33] H. F. Olson and E. G. May, "Electronic sound absorber," *Journal of the Acoustical Society of America*, vol. 25, no. 6, pp. 1130–1136, 1953.
[34] K. Ozeki and T. Umeda, "An adaptive filtering algorithm using an orthogonal projection to an affine subspace and its properties," *Electronics and Communications in Japan*, vol. 67-A, no. 5, pp. 19–27, 1984.
[35] J. C. Patra, R. N. Pal, B. N. Chatterji, and G. Panda, "Identification of nonlinear dynamic systems using functional link artificial neural networks," *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics*, vol. 29, no. 2, pp. 254–262, 1999.
[36] D. Pavisic, L. Blondel, J.-P. Draye, G. Libert, and P. Chapelle, "Active noise control with dynamic recurrent neural networks," in *Proceedings of 3rd European Symposium on Artificial Neural Networks (ESANN '95)*, pp. 45–50, Brussels, Belgium, April 1995.
[37] G. M. Raz and B. D. Van Veen, "Baseband Volterra filters for implementing carrier based nonlinearities," *IEEE Transactions on Signal Processing*, vol. 46, no. 1, pp. 103–114, 1998.
[38] M. Rupp and A. H. Sayed, "Robust FxLMS algorithms with improved convergence performance," *IEEE Transactions on Speech and Audio Processing*, vol. 6, no. 1, pp. 78–85, 1998.
[39] H. Sakai and S. Miyagi, "Analysis of the adaptive filter algorithm for feedback-type active noise control," *Signal Processing*, vol. 83, no. 6, pp. 1291–1298, 2003.
[40] G. L. Sicuranza and A. Carini, "Filtered-X affine projection algorithm for multichannel active noise control using second-order Volterra filters," *IEEE Signal Processing Letters*, vol. 11, no. 11, pp. 853–857, 2004.
[41] G. L. Sicuranza and A. Carini, "A multichannel hierarchical approach to adaptive Volterra filters employing filtered-X affine projection algorithms," *IEEE Transactions on Signal Processing*, vol. 53, no. 4, pp. 1463–1473, 2005.
[42] G. L. Sicuranza and A. Carini, "Nonlinear multichannel active noise control using partial updates," in *Proceedings of IEEE International Conference on Acoustics, Speech, Signal Processing (ICASSP '05)*, Philadelphia, Pa, USA, March 2005.
[43] C. A. Silva, J. M. Sousa, and J. M. G. Sá da Costa, "Active noise control based on fuzzy models," in *Proceedings of 4th European Conference on Noise Control (EURONOISE '01)*, pp. 1–14, Patras, Greece, January 2001.
[44] S. D. Snyder and N. Tanaka, "Active control of vibration using a neural network," *IEEE Transactions on Neural Networks*, vol. 6, no. 4, pp. 819–828, 1995.
[45] P. Strauch and B. Mulgrew, "Active control of nonlinear noise processes in a linear duct," *IEEE Transactions on Signal Processing*, vol. 46, no. 9, pp. 2404–2412, 1998.

[46] A. H. Sayed, *Fundamentals of Adaptive Filtering*, John Wiley & Sons, New York, NY, USA, 2003.
[47] T. J. Sutton and S. J. Elliott, "Active attenuation of periodic vibration in nonlinear systems using an adaptive harmonic controller," *Journal of Vibration and Acoustics, Transactions of the ASME*, vol. 117, pp. 355–362, 1995.
[48] L. Tan and J. Jiang, "Filtered-X second-order Volterra adaptive algorithms," *Electronics Letters*, vol. 33, no. 8, pp. 671–672, 1997.
[49] L. Tan and J. Jiang, "Adaptive Volterra filters for active control of nonlinear noise processes," *IEEE Transactions on Signal Processing*, vol. 49, no. 8, pp. 1667–1676, 2001.
[50] M. Tanaka, Y. Kaneda, S. Makino, and J. Kojima, "Fast projection algorithm and its step size control," in *Proceedings of IEEE International Conference on Acoustics, Speech, Signal Processing (ICASSP '95)*, vol. 2, pp. 945–948, Detroit, Mich, USA, May 1995.
[51] O. J. Tobias and R. Seara, "Performance comparison of the FXLMS, nonlinear FXLMS and leaky FXLMS algorithms in nonlinear active control applications," in *Proceedings of 11th European Signal Processing Conference (EUSIPCO '02)*, vol. 1, pp. 155–158, Toulouse, France, September 2002.
[52] A. K. Wang and W. Ren, "Convergence analysis of the filtered-U algorithm for active noise control," *Signal Processing*, vol. 73, no. 3, pp. 255–266, 1999.
[53] G. E. Warnaka, "Active attenuation of noise—the state of the art," *Noise Control Engineering*, vol. 18, no. 3, pp. 100–110, 1982.
[54] S. Werner and P. S. R. Diniz, "Set-membership affine projection algorithm," *IEEE Signal Processing Letters*, vol. 8, no. 8, pp. 231–235, 2001.

Giovanni L. Sicuranza: Department of Electrical, Electronic, and Computer Engineering, University of Trieste, Via A. Valerio 10, 34127 Trieste, Italy
*Email*: sicuranza@univ.trieste.it

Alberto Carini: Information Science and Technology Institute, University of Urbino, Piazza della Repubblica 13, 61029 Urbino, Italy
*Email*: carini@sti.uniurb.it

# 8 Chaotic sequences for digital watermarking

Nikos Nikolaidis, Anastasios Tefas, and Ioannis Pitas

## 8.1. Introduction

The design of robust techniques for copyright protection and content verification of multimedia data became an urgent necessity in the last years due to the proliferation of digital media that made unauthorized use, distribution, and processing of content an easy task. This demand has been lately addressed by the emergence of a variety of watermarking methods. Such methods target towards hiding in the original data an imperceptible and undetectable signal which conveys information about the host medium (owner or authorized user, transaction or product ID, etc). For a review of existing schemes and a detailed discussion on the main requirements of a watermarking scheme, the interested reader may consult [5, 9].

The field of nonlinear dynamics which has its roots in the work of French mathematician H. Poincaré (1854–1912) is dealing with the study of the systems whose time evolution equations are of nonlinear nature. Under certain conditions, some nonlinear systems give rise to chaos, that is, a complex, aperiodic, and apparently random behavior. However, neither is the random and complex behavior of chaotic systems the result of uncontrolled external factors, namely, noise, nor is it necessarily attributed to the complexity of the system; most chaotic systems are deterministic in nature and can be described by very simple equations. Chaotic behavior can result, for example, from the iterative application of an appropriate nonlinear scalar map $f(\cdot)$:

$$x[n] = f(x[n-1]) = f^n(x[0]) = \underbrace{f(f(\cdots(f}_{n \text{ times}}(x[0]))\cdots)), \tag{8.1}$$

where $x[0]$ denotes the system initial condition and $f^n(x[0])$ denotes the $n$th application of the map. The logistic map is a well-known example of such a 1D iterated map that can exhibit the chaotic behavior for the appropriate values of the

parameter $A$:

$$x[n] = Ax[n-1](1-x[n-1]), \quad 0 \le x[n] \le 1,\ 0 \le A. \tag{8.2}$$

The points $x[0], x[1], x[2], \ldots$ generated by the recursion (8.1) form an orbit of the chaotic system which can be viewed as a 1D discrete-time chaotic sequence. Thus, a chaotic sequence $\mathbf{x}$ is fully described by the map $f(\cdot)$ and the initial condition $x[0]$. An important characteristic of chaotic systems is their sensitivity to initial conditions. Starting from two initial conditions that are very close to each other, a chaotic system will generate completely different orbits.

During the last decades, methods and principles of chaos and nonlinear dynamics have found numerous applications in a wide spectrum of scientific disciplines that include, but are not limited to, physics, economics, social science, and engineering. Chaos-based cryptography and chaotic spread spectrum communications are two prominent application examples of chaotic systems within the electrical engineering discipline. For a thorough treatment of the field of chaos and nonlinear dynamics, the reader can consult one of the many textbooks in this area, for example, [12, 23].

In this chapter, two applications of chaotic systems in digital watermarking are presented. The first involves chaotic systems as watermark signal generators in blind watermarking schemes employing correlation detection. Statistical properties of watermark sequences generated by piecewise linear Markov maps are exploited for both additive and multiplicative watermark embedding and the performance of such systems is theoretically evaluated. The major advantage of chaotic sequences in this context is their easily controllable spectral/correlation properties, a fact that makes them a good alternative to the widely used pseudorandom signals [29, 31]. Chaotic watermarks can be embedded either in the temporal/spatial domain or in a transform domain where their correlation/spectral properties can be exploited more efficiently for obtaining robust watermarking schemes. Such a transform-domain watermarking technique is exemplified. The scheme involves multiplicative embedding of high-frequency chaotic watermarks in the low frequencies of the discrete Fourier transform (DFT). The corresponding watermarking scheme guarantees robustness against lowpass attacks, along with enhancement of the detector reliability. The complete theoretical justification and statistical analysis of correlation-based additive and multiplicative schemes employing chaotic watermarks can be found in [11, 28, 29].

The second application deals with watermark generation by mixing (scrambling) a binary logo or encrypting other forms of messages using 2D chaotic maps. Mixing using chaotic maps provides excellent security and has many favorable cryptographic properties, stemming from the map sensitivity to initial conditions. An example of a system that utilizes such a chaotic mixing along with appropriate watermark embedding and detection functions is presented.

This chapter is organized as follows. Additive and multiplicative embedding correlation-based watermarking schemes employing Markov chaotic sequences are presented and analyzed in Section 8.2. Watermark generation by the chaotic

mixing is described in Section 8.3. Other applications of chaotic signals in watermarking are reviewed in brief in Section 8.4.

## 8.2. Correlation-based watermarking schemes employing Markov chaotic sequences

Watermarking schemes where watermark detection is performed through a correlation detector have been extensively used and studied in the corresponding literature. In this section, the use of sequences generated by chaotic systems as watermark signals in correlation-based watermarking schemes is examined and the performance of such schemes is compared against schemes utilizing pseudorandom sequences. Two different schemes are presented and analyzed. The first scheme involves additive watermark embedding and demonstrates the superior performance obtained when highpass chaotic signals are incorporated. Building on the analysis and the results derived for this scheme, a second scheme that incorporates multiplicative embedding of highpass chaotic signals in the low frequencies of the DFT domain in order to achieve robustness to lowpass manipulations is also presented and analyzed.

### 8.2.1. General model of a correlation-based watermarking scheme

A watermarking system encompasses three major functionalities, namely, watermark generation, watermark embedding, and watermark detection. The aim of watermark generation is to construct a sequence $\mathbf{w}, w[i] \in \mathcal{R}$, consisting of $N$ samples using an appropriate function $g$:

$$\mathbf{w} = g(K, N), \tag{8.3}$$

where $K$ denotes the watermark key. Watermark embedding aims at inserting within a host signal $\mathbf{f}_o$ the watermark signal $\mathbf{w}$ in a way that guarantees robustness in case of both intentional or unintentional attacks and imperceptibility of the distortions caused to the host signal. Watermark embedding can be performed in any suitable domain, that is, in the temporal/spatial domain or in a properly selected transform domain (DFT, DCT, wavelet transform, etc).

The aim of watermark detection is to verify whether or not a given watermark $\mathbf{w}_d$ resides in a test signal $\mathbf{f}_t$. Thus, watermark detection can be considered as a binary hypothesis test. The two hypotheses involved in this test are

(i) $H_0$: the test signal $\mathbf{f}_t$ hosts the watermark $\mathbf{w}_d$,

(ii) $H_1$: the test signal $\mathbf{f}_t$ does not host the watermark $\mathbf{w}_d$ or it hosts a different watermark $\mathbf{w}_e \neq \mathbf{w}_d$ than the one under investigation.

A test statistic that has been frequently employed in the watermarking literature for examining whether the signal $\mathbf{f}_t$ hosts a watermark $\mathbf{w}_d$ or not is the correlation between $\mathbf{f}_t$ and the watermark:

$$c = \frac{1}{N} \sum_{n=0}^{N-1} f_t[n] w_d[n]. \tag{8.4}$$

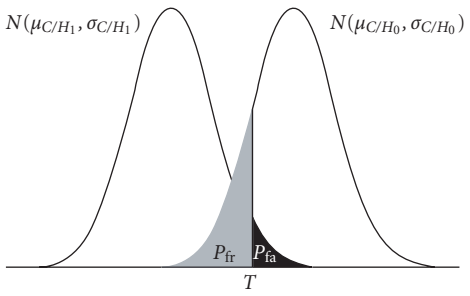



FIGURE 8.1. Conditional pdf's of the correlator output under hypotheses $H_0$, $H_1$.

Such a detection scheme is usually called a correlation detector. The decision on the valid hypothesis is taken by comparing $c$ against a properly selected threshold $T$. A decision to adopt hypothesis $H_0$ is taken when $c > T$. Obviously such a scheme is a zero-bit scheme in the sense that the detection procedure can provide only information on whether a specific watermark is indeed embedded in the host image or not.

The detection performance of a zero-bit watermarking scheme can be measured using the probability of false alarm $P_{\text{fa}}(T)$, which is defined as the probability to detect a watermark in a signal that is not watermarked or hosts a different watermark, and the probability of false rejection $P_{\text{fr}}(T)$, defined as the probability of failing to detect a watermark in a signal that is watermarked, for a certain threshold $T$,

$$P_{\text{fa}}(T) = \text{Prob}\{c > T \mid H_1\} = \int_T^{\infty} f_{c|H_1}(t)dt, \tag{8.5}$$

$$P_{\text{fr}}(T) = \text{Prob}\{c < T \mid H_0\} = \int_{-\infty}^{T} f_{c|H_0}(t)dt, \tag{8.6}$$

where $f_{c|H_0}, f_{c|H_1}$ are the conditional probability density functions of the correlation $c$ in (8.4) under the hypotheses $H_0$ and $H_1$, respectively (Figure 8.1).

By solving both (8.5) and (8.6) for $T$ and equating the results, one can express $P_{\text{fr}}$ as a function of $P_{\text{fa}}$. The plot of $P_{\text{fa}}$ versus $P_{\text{fr}}$ is called the receiver operating characteristic (ROC) curve of the corresponding watermarking system. This curve conveys all the information required in order to judge the detection performance of a such a system. Using the ROC curve one can evaluate various performance indicators, including the equal error rate (EER) point, that is, the point on the curve where $P_{\text{fa}} = P_{\text{fr}}$. EER can be used as a simple scalar metric of a watermarking scheme's performance.

### 8.2.2. Use of Markov chaotic sequences as watermark signals

In most watermarking schemes, the construction of the watermark signal that will be embedded in the host data involves a function that generates pseudo-random numbers. Samples generated by such functions can be safely modeled as

identically distributed independent random variables following a uniform distribution. Zero-mean, pseudorandom sequences distributed in the interval $[-0.5, 0.5]$ are used in most cases and will be the subject of comparison with chaotic sequences in this chapter. Watermark signals obeying Gaussian distributions are also frequently encountered in the corresponding literature.

Sequences generated by chaotic maps (8.1) can be used as an efficient alternative to watermarking sequences generated by pseudorandom number generators. Chaotic sequences $\mathbf{x}$ are usually generated in the interval $[0, 1]$. A chaotic watermark sequence whose values reside in a different interval can be constructed as

$$\mathbf{w} = \mathbf{x} - d\mathbf{1}, \tag{8.7}$$

where $d$ is a constant that controls the range of the watermark sequence and $\mathbf{1} = [1, 1, \ldots, 1]^T$.

By imposing certain constraints on the map or the initial condition, sequences of infinite period can be obtained. Thus, if one considers two sequences $\mathbf{x}$, $\mathbf{y}$ of finite length generated by the recursive application of the same chaotic map $f$ on two different initial conditions $x[0]$, $y[0]$, that belong to the same chaotic orbit, there will always be an integer $k > 0$ such that the following expressions hold:

$$x[0] = f^k(y[0]) \qquad \text{or} \qquad y[0] = f^k(x[0]). \tag{8.8}$$

Therefore, the samples $x[n]$, $y[n]$ are associated through the following expression for a properly selected value of $k > 0$:

$$y[n] = f^n(y[0]) = f^n\left(f^k(x[0])\right) = x[n+k] \qquad \text{or} \qquad x[n] = y[n+k]. \tag{8.9}$$

From now on, the constant $k$ will be called sequence shift.

A class of 1D chaotic systems that lend themselves to analysis are the eventually expanding, piecewise linear Markov maps. A map $\mathcal{M} : [0, 1] \to [0, 1]$ is an eventually expanding, piecewise linear Markov map if it satisfies the following conditions.

(1) The map is a piecewise linear one, that is, there exists a set of points $0 = \alpha_0 < \alpha_1 < \cdots < \alpha_M = 1$ (called partition points) such that, when the map is restricted to each of the intervals $(\alpha_{i-1}, \alpha_i)$ (called partition elements), it is affine.

(2) The map satisfies the Markov property, that is, the partition points are mapped to partition points:

$$\forall i \in [0, \ldots, M], \quad \exists j \in [0, \ldots, M] : \mathcal{M}(\alpha_i) = \alpha_j. \tag{8.10}$$

(3) The map is eventually expanding, that is, there exists an integer $r > 0$ such that

$$\inf_{x \in [0,1]} \left| \frac{d}{dx} \mathcal{M}^r(x) \right| > 1. \tag{8.11}$$

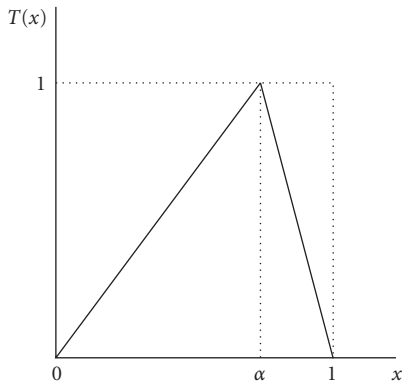



FIGURE 8.2. The skew tent map.

For brevity, maps satisfying the above definition will be referred to simply as Markov maps when no risk of ambiguity arises. The Markov sequences that will be used in the sequel have exponential autocorrelation function.

A piecewise linear Markov map with very interesting properties is the skew tent map [25]. This map is illustrated in Figure 8.2 and can be expressed as

$$\mathcal{T} : [0,1] \longrightarrow [0,1], \quad \mathcal{T}(x) = \begin{cases} \dfrac{1}{\alpha}x, & 0 \le x \le \alpha, \\ \dfrac{1}{\alpha - 1}x + \dfrac{1}{1-\alpha}, & \alpha < x \le 1, \end{cases} \quad \alpha \in (0,1). \tag{8.12}$$

A chaotic orbit $t[k]$ can be obtained by iterating this map

$$t[k] = \mathcal{T}\left(t[k-1]\right) = \mathcal{T}^k(t[0]). \tag{8.13}$$

Another interesting member of this family is the $m$-way Bernoulli shift which is defined as

$$\mathcal{B}_m : [0,1] \longrightarrow [0,1], \quad \mathcal{B}_m(x) = mx \,(\mathrm{mod}1). \tag{8.14}$$

Sequences generated by Markov maps posses properties that make them superior to pseudorandom signals when used as watermarks in correlation-based schemes. This will become obvious in the following sections that provide a theoretical and experimental performance analysis of such schemes.

### 8.2.3. Additive embedding correlation-based schemes employing Markov chaotic sequences

Additive watermark embedding is one of the most frequently used embedding rules:

$$\mathbf{f}_w = \mathbf{f}_o + p\mathbf{w}, \tag{8.15}$$

where $\mathbf{f}_w$ is the watermarked signal and $p$ is a constant that controls the watermark embedding power, which will be called hereafter watermark embedding factor and is closely related to the watermark perceptibility. Additive watermark embedding can be performed in any suitable domain. In the following, we will assume spatial domain embedding. However, readers should bear in mind that the analysis presented below can be applied, with certain adaptations, for other embedding domains as well.

Assuming that the test signal has not been subject to manipulations, the binary hypothesis test involved in watermark detection in case of additive embedding can be expressed as follows:

(i) $H_0$: the test signal $\mathbf{f}_t$ hosts the watermark $\mathbf{w}_d$, that is, $\mathbf{f}_t = \mathbf{f}_o + p\mathbf{w}_d$,

(ii) $H_1$: the test signal $\mathbf{f}_t$ does not host the watermark $\mathbf{w}_d$, that is, $\mathbf{f}_t = \mathbf{f}_o$ or it hosts a different watermark $\mathbf{w}_e \neq \mathbf{w}_d$ than the one under investigation.

The following formula can be used to summarize both events mentioned above:

$$\mathbf{f}_t = \mathbf{f}_o + p\mathbf{w}_e. \tag{8.16}$$

In this formula, the watermark $\mathbf{w}_d$ is indeed embedded in the signal if $p \neq 0$ and $\mathbf{w}_e = \mathbf{w}_d$ (event $H_0$), and it is not embedded in the signal if $p = 0$ (the signal hosts no watermark, an event that will be denoted as $H_{1a}$) or $\mathbf{w}_e \neq \mathbf{w}_d$ (the signal hosts a different watermark, denoted as event $H_{1b}$).

#### 8.2.3.1. Theoretical performance analysis

In an additive embedding scheme the correlation detector can be expressed as

$$c = \frac{1}{N}\sum_{n=0}^{N-1} f_t[n]w_d[n] = \frac{1}{N}\sum_{n=0}^{N-1}\left(f_o[n]w_d[n] + pw_e[n]w_d[n]\right). \tag{8.17}$$

Although samples of Markov chaotic watermarks are correlated for small $k > 0$, since they possess exponential autocorrelation function and $\mathbf{w}_d$ is a shifted version of $\mathbf{w}_e$ (cf. (8.9)), the central limit theorem for random variables with small dependency [3] may be used in order to establish that the correlation $c$ in (8.17) attains a Gaussian distribution, even in the case of event $H_{1b}$ (assuming that $N$ is sufficiently large). Therefore, for sequences generated by piecewise linear Markov maps, $f_{c|H_0}$, $f_{c|H_1}$ are normal distributions. The same can be shown for pseudorandom sequences. Thus, $f_{c|H_0}$, $f_{c|H_1}$ can be fully determined in terms of their means $\mu_{c|H_0}$, $\mu_{c|H_1}$ and variances $\sigma^2_{c|H_0}$, $\sigma^2_{c|H_1}$. As a consequence, the performance of an additive embedding correlation-based watermarking scheme for this type of watermark signals depends only on those four parameters and the ROC curve can be described through the following expression:

$$P_{\text{fa}} = \frac{1}{2}\left[1 - \text{erf}\left[\frac{\sqrt{2}\sigma_{c|H_0}\,\text{erf}^{-1}\left(2P_{\text{fr}} - 1\right) + \mu_{c|H_0} - \mu_{c|H_1}}{\sqrt{2}\sigma_{c|H_1}}\right]\right]. \tag{8.18}$$

A close look at Figure 8.1 reveals that the system performance improves (i.e., both the probability of false alarm and the probability of false rejection decrease for a certain value of the threshold $T$) when the two distributions move apart, that is, when the difference $\mu_{c|H_0} - \mu_{c|H_1}$ increases. In a similar manner, the performance improves when the variances of the two distributions $\sigma^2_{c|H_0}$, $\sigma^2_{c|H_1}$ decrease.

Expressions for the mean and variance of the correlation $c$ can be easily obtained:

$$
\begin{aligned}
\mu_c &= E[c] = \frac{1}{N}\sum_{n=0}^{N-1} E[f_o[n]]E[w_d[n]] + \frac{1}{N}\sum_{n=0}^{N-1} pE[w_e[n]w_d[n]],\\
\sigma_c^2 &= E[c^2] - E[c]^2 = E\left[\left(\frac{1}{N}\sum_{n=0}^{N-1}\left(f_o[n]w_d[n] + pw_e[n]w_d[n]\right)\right)^2\right] - \mu_c^2\\
&= \frac{1}{N^2}E\Bigg[\sum_{n=0}^{N-1}\left(f_o[n]w_d[n] + pw_e[n]w_d[n]\right)^2\\
&\qquad + \sum_{n=0}^{N-1}\sum_{\substack{m=0,\\ m\neq n}}^{N-1}\left(f_o[n]w_d[n] + pw_e[n]w_d[n]\right)\\
&\qquad \times \left(f_o[m]w_d[m] + pw_e[m]w_d[m]\right)\Bigg] - \mu_c^2\\
&= \frac{1}{N^2}\Bigg[\sum_{n=0}^{N-1}\left(E[f_o^2[n]]E[w_d^2[n]] + p^2E[w_d^2[n]w_e^2[n]]\right.\\
&\qquad \left. + 2pE[f_o[n]]E[w_e[n]w_d^2[n]]\right)\\
&\qquad + \sum_{n=0}^{N-1}\sum_{\substack{m=0,\\ m\neq n}}^{N-1}\left(E[f_o[n]f_o[m]]E[w_d[n]w_d[m]]\right.\\
&\qquad\qquad + pE[f_o[n]]E[w_d[n]w_e[m]w_d[m]]\\
&\qquad\qquad + pE[f_o[m]]E[w_e[n]w_d[m]w_d[n]]\\
&\qquad\qquad \left. + p^2E[w_e[n]w_e[m]w_d[n]w_d[m]]\right)\Bigg] - \mu_c^2.
\end{aligned}
\tag{8.19}
$$

The above expressions hold for both events $H_0$ ($\mathbf{w}_d = \mathbf{w}_e$) and $H_1$ ($\mathbf{w}_d \neq \mathbf{w}_e$ or $p = 0$). The statistical independence between the host signal $\mathbf{f}_o$ and both watermarks $\mathbf{w}_e$, $\mathbf{w}_d$ has been exploited in the derivation of the formulas above.

Obviously, several moments need to be evaluated if $\mu_c$, $\sigma_c^2$ are to be computed using (8.19). In order to proceed with such an evaluation, one has to make certain assumptions about the statistical properties of the host signal. In our case, the host

signal was assumed to be wide-sense stationary, thus

$$\begin{aligned} E[f_o[n]] &= \mu_{f_o} \quad \forall n,\ n = 0,\ldots,N-1, \\ E[f_o[n]f_o[n+k]] &= R_{f_o}[k] \quad \forall n,\ n = 0,\ldots,N-1. \end{aligned} \tag{8.20}$$

Furthermore, the host signal was assumed to follow a first-order exponential autocorrelation function model:

$$R_{f_o}[k] = \mu_{f_o}^2 + \sigma_{f_o}^2 \beta^k, \quad k \geq 0,\ |\beta| \leq 1. \tag{8.21}$$

In the previous equation, $\beta$ is the parameter of the autocorrelation function and $\sigma_{f_o}^2$ is the host signal variance:

$$\sigma_{f_o}^2 = E[f_o^2[n]] - E[f_o[n]]^2. \tag{8.22}$$

This model has been chosen because of its simplicity and tractability. Moreover, it can model sufficiently well the autocorrelation function of image scanlines and speech signals [8, 14, 19]. Despite their simplicity, the assumptions mentioned above have enabled us to derive theoretical results that are very close to the experimental results, when audio signals were considered as host signals (see, Section 8.2.3.2).

By substituting (8.7) in (8.19) while taking into account the fact that $w_d[n] = w_e[n+k]$, according to (8.9), one can derive general expressions for the mean value and the variance of the correlation $c$. These expressions are rather complex and lengthy and thus will not be replicated here. The interested reader can consult [29]. Much simpler expressions can be obtained if the constant value $d$ in (8.7) is chosen so that watermarks have zero mean value (which according to [19] results in better system performance) and the test signal mean value is subtracted prior to detection (an action which was shown in [29] to result in a decrease of the variance of the correlation, thus resulting in better system performance):

$$\mu_c = p(R_x[k] - \mu_x^2), \tag{8.23}$$

$$\begin{aligned} \sigma_c^2 = {} & \frac{p^2}{N^2} \sum_{m=0}^{N-1} (N-m)(2-\delta(m)) \\ & \times \{\mu_x^2(2R_x[m] + R_x[m+k] + R_x[k-m]) - \mu_x(R_x[k,m] + R_x[m,m+k] \\ & \quad + R_x[k,k-m] + R_x[k,m+k]) + R_x[m,k,m+k]\} \\ & + \frac{1}{N^2} \sum_{m=0}^{N-1} (N-m)(2-\delta(m))(R_x[m] - \mu_x^2)R_{f_o}[m] - p^2(R_x[k] - 2\mu_x^2)^2, \end{aligned} \tag{8.24}$$

where $\delta(m)$ is the Dirac delta function and $R_x[\mathbf{k}]$ is a statistic of the form

$$R_x[\mathbf{k}] = R_x[k_1, k_2, \ldots, k_r] = E[x[n]x[n+k_1]x[n+k_2]\cdots x[n+k_r]]. \tag{8.25}$$

This statistic will be called hereafter $r$th-order correlation statistic of a wide-sense stationary signal $x$.

Similar to expressions (8.19), expressions (8.23) and (8.24) can encompass all three events $H_0$, $H_{1a}$, $H_{1b}$, provided that piecewise linear Markov maps are used to generate the watermark sequence. In other words, event $H_{1a}$ can be represented by setting $p$ to zero, event $H_0$ can be represented by using a positive watermark embedding factor and $k = 0$, whereas event $H_{1b}$ can be represented by a positive embedding factor and $k > 0$.

When event $H_{1b}$ holds, that is, under the worst case assumption, both $\mu_c$ (8.23) and $\sigma_c^2$ (8.24) converge to constant values for large values of $k$. In such a case, $P_{\text{fa}|H_{1b}}$ substitutes $P_{\text{fa}|H_1}$ since this is the worst case. $P_{\text{fa}|H_{1b}}$ can be estimated using the limit values of $\mu_c$ and $\sigma_c^2$ when $k$ tends to infinity. The values of $P_{\text{fr}}$ can be estimated using the values of $\mu_c$ and $\sigma_c^2$ for $k = 0$ (which corresponds to event $H_0$). Subsequently, ROC curves can be evaluated using (8.18).

A close examination of (8.23) reveals that the correlation mean value depends on the embedding factor and the variance of the watermark but is independent of the type of the watermark signal generator (chaotic or pseudorandom) that has been used and the spectral properties of the watermark signal (see [29] for a more detailed explanation). Thus the performance of a watermarking system that involves watermark signals of the same embedding factor and the same variance is affected only by the variance of the correlation detector. The performance of such a system improves as long as the variance of the correlation for events $H_0$ and $H_1$ decreases. This fact implies that in order to improve the performance one has to design watermarks that result in small correlation variance. According to (8.24), such a design objective can be achieved by employing watermark signals with suitable first-, second-, and third-order correlation statistics. Chaotic watermark signals posses such desirable properties as will be shown in the rest of this section.

In order to gain further insight on (8.23), (8.24), we rewrite them below so that the two events $H_0$ and $H_{1a}$ are expressed separately:

$$\mu_c = \begin{cases} 0, & p = 0 (H_{1a}), \\ p\sigma_x^2, & k = 0,\ p \neq 0 (H_0), \end{cases} \tag{8.26}$$

$$\sigma_c^2 = \begin{cases} \dfrac{1}{N}\sigma_x^2\sigma_{f_o}^2 + \dfrac{2}{N^2}\displaystyle\sum_{m=1}^{N-1}(N-m)\left(R_x[m]-\mu_x^2\right)R_{f_o}[m], & p = 0 (H_{1a}), \\ \dfrac{1}{N}\sigma_x^2\sigma_{f_o}^2 + \dfrac{p^2}{N}\left(4\mu_x^2 R_x[0] - 4\mu_x R_x[0,0] + R_x[0,0,0]\right) & \\ \quad -p^2\left(R_x[0]-2\mu_x^2\right)^2 + \dfrac{2p^2}{N^2}\displaystyle\sum_{m=1}^{N-1}(N-m) & \\ \quad \times\left\{4\mu_x^2 R_x[m] - 2\mu_x\left(R_x[0,m]+R_x[m,m]\right)+R_x[0,m,m]\right\} & \\ \quad +\dfrac{2}{N^2}\displaystyle\sum_{m=1}^{N-1}(N-m)\left(R_x[m]-\mu_x^2\right)R_{f_o}[m], & k=0,\ p\neq 0(H_0). \end{cases} \tag{8.27}$$

In case of event $H_{1a}$, one can easily observe that the correlation variance depends only on the autocorrelation function $R_x[m]$ of the watermark signal which in turn is directly associated with its power spectral density (PSD) through the following formula:

$$S_x(\omega) = \sum_{k=-\infty}^{\infty} R_x[k]e^{-j\omega k} = R_x[0] + \sum_{k=1}^{\infty} R_x[k]\left(e^{-j\omega k} + e^{j\omega k}\right). \tag{8.28}$$

As a consequence, the variance of the correlation for the event $H_{1a}$ is determined by the spectral characteristics of the watermark signal in use. Furthermore, having in mind that Markov chaotic sequences have an exponential autocorrelation function such as that presented in (8.21), it can be easily concluded that the correlation variance (8.27) depends on the sum of the samples of the autocorrelation function, evaluated over the interval $[0, N-1]$. This sum is minimized for $\beta \to -1$, and maximized for $\beta \to 1$. Through (8.28), one can conclude that these two cases correspond to the most highpass and the most lowpass signals that can be generated with an exponential autocorrelation function. The discussion above can lead to the conclusion that highpass watermarks result in smaller correlation variance than lowpass ones when no attacks on the watermarked signal are considered and thus attain a better performance. In case of event $H_{1b}$, the correlation variance $\sigma^2_{c|H_{1b}}$ still depends only on the spectrum of the watermark signal, as was shown in [29].

Based on the previous analysis, another extremely interesting conclusion can be also drawn: two sequences having the same variance, embedding factor, and spectral properties might exhibit different performance in a watermarking scheme. In case of event $H_1$, the correlation mean and variance of such sequences would have the same value. However, the variances of the correlation for the event $H_0$ might differ, since in this case the variance depends also on the second- and third-order correlation statistics $R_x[0, m]$, $R_x[m, m]$, and $R_x[0, m, m]$ of the watermark. As a result, one can generate, for example, white chaotic watermark sequences that perform better than white sequences generated by a pseudorandom generator. This fact will be exemplified in Section 8.2.3.2.

Obviously, in order to proceed with the performance analysis of watermarking systems based on chaotic sequences, expressions for the correlation statistics, that are involved in expressions (8.23) and (8.24), must be derived. A linear operator, which is referred to as the Frobenius-Perron (FP) operator [17], can be defined so that

$$p_n(\cdot) = P_f\{p_{n-1}(\cdot)\} = P_f^n\{p_0(\cdot)\}, \tag{8.29}$$

where $p_n(\cdot)$ denotes the probability density function of the $n$th iterate $x[n]$ in (8.1). The FP operator describes the evolution in time of the density $p_n(\cdot)$ for a specific chaotic map. In the general case, the probability density functions at distinct iterates $n$ will differ. However, there can be certain choices of $p_0(\cdot)$ such

that the densities of subsequent iterates do not change over time, that is,

$$p(\cdot) = P_f^n\{p(\cdot)\} \quad \forall n. \tag{8.30}$$

A density $p(\cdot)$ that satisfies this property is called the invariant density of the map $f(\cdot)$. A map might have more than one invariant density. The invariant density plays an important role in the computation of time-averaged statistics of time series generated by nonlinear dynamical systems. When $p_0(\cdot)$ is chosen to be an invariant density, one can verify that the resulting stochastic process is stationary and, subject to certain constraints on the map, ergodic [17].

The Markov maps described in Section 8.2.2 possess a number of useful properties. All Markov maps posses invariant densities and are ergodic under easily verifiable conditions [4]. Moreover, sequences generated by Markov maps, when suitably quantized, are equivalent to Markov chains. As already mentioned, the Markov sequences that were used in this study attain exponential autocorrelation function given by (8.21), where $\beta$ is an eigenvalue of the corresponding FP matrix [16]. The higher-order correlation statistics of Markov maps can also be determined in closed form. A general strategy for computing these statistics was developed in [13]. The results obtained by applying this strategy on the Markov maps are detailed in [29] and will not be repeated here due to lack of space. Readers interested in obtaining more details on this analysis can consult this publication.

The analysis techniques presented above can be exemplified using the skew tent map, which, as already mentioned, is a piecewise linear Markov map. The invariant density of this map can be shown to be uniform. Following the methodology described in brief above, the statistical properties of sequences produced using the skew tent map can be derived as was shown in [29]. Among the formulas derived in this publication were analytical expressions for the first-, second-, and third-order correlation statistics (8.25) required for evaluating the performance of watermarking schemes based on the skew tent map. For example, the first-order correlation statistic, namely, the autocorrelation function, was shown to be

$$R_t[k] = \frac{1}{4} + \frac{1}{12}e_2^k = \frac{1}{4} + \frac{1}{12}(2\alpha - 1)^k, \tag{8.31}$$

where $e_2 = 2\alpha - 1$ is an eigenvalue of the FP matrix $\mathbf{P}_3$ (for more details, consult [13, 29]). According to this expression, the autocorrelation function of the skew tent map depends only on its parameter $\alpha$. Therefore, by controlling $\alpha$, one can generate sequences that have any desirable exponential autocorrelation function. Using (8.28) and (8.31), the PSD of sequences generated by the skew tent map can be easily evaluated:

$$S_t(\omega) = \frac{1 - e_2^2}{12(1 + e_2^2 - 2e_2\cos\omega)}. \tag{8.32}$$

Thus, highpass ($\alpha < 0.5$) or lowpass ($\alpha > 0.5$) sequences can be generated. The value $\alpha = 0.5$ corresponds to the symmetric tent map. Sequences generated by this

map have a Dirac delta autocorrelation function and thus possess a white spectrum. The control over the spectral properties of watermarking sequences provided by chaotic functions is very useful, since the spectral characteristics of the watermark sequence are related to watermark robustness against common attacks, such as filtering and compression.

Using the analytical expressions for the correlation statistics of skew tent sequences derived in [29] one can derive the mean value and the variance of the correlation detector for this map:

$$\mu_c = \begin{cases} 0, & p = 0 (H_{1a}), \\ \dfrac{p}{12}, & k = 0,\ p \neq 0 (H_0), \end{cases}$$

$$\sigma_c^2 = \begin{cases} \dfrac{\sigma_{f_o}^2}{12N^2} \dfrac{N - 2\beta e_2 - N\beta^2 e_2^2 + 2(\beta e_2)^{N+1}}{(1 - \beta e_2)^2}, & p = 0 (H_{1a}), \\ \dfrac{p^2}{180N^2} \dfrac{N - 2e_1 - Ne_1^2 + 2e_1^{N+1}}{(1 - e_1)^2} & \\ \quad + \dfrac{\sigma_{f_o}^2}{12N^2} \dfrac{N - 2\beta e_2 - N\beta^2 e_2^2 + 2(\beta e_2)^{N+1}}{(1 - \beta e_2)^2}, & k = 0,\ p \neq 0 (H_0), \end{cases} \tag{8.33}$$

where $e_1$, $e_2$ are eigenvalues of the FP matrix $\mathbf{P}_3$ and $\beta$ is the host signal autocorrelation function parameter in (8.21). Using these expressions, one can derive an analytical expression for the ROC curve of a watermarking system that employs sequences generated by the skew tent map.

The methodology presented above can be also used to derive the statistics required for the performance analysis of a watermarking system utilizing $m$-way Bernoulli maps. A detailed study on the performance of correlation-based watermarking schemes based on Bernoulli maps, using a different methodology than the one introduced in [29] and described in this section, can be found in [31]. For $m$-way Bernoulli sequences, the first-order correlation statistic can be shown to be

$$R_b[k] = E[b[n]b[n+k]] = \frac{1}{4} + \frac{1}{12m^k}. \tag{8.34}$$

By comparing (8.31), (8.34) one can conclude that for $\alpha = (m+1)/2m$ the tent map generates sequences that have the same spectral properties with the corresponding Bernoulli sequences. It is worth noting here that Bernoulli maps can generate only lowpass sequences. As $m$ tends to infinity, Bernoulli sequences tend towards obtaining a white spectrum. The most lowpass sequence that can be generated using Bernoulli maps is the one obtained for $m = 2$. Thus, Bernoulli maps lack in flexibility. On the contrary, tent chaotic watermarks can generate sequences with exponential autocorrelation function and any desirable spectral characteristics.

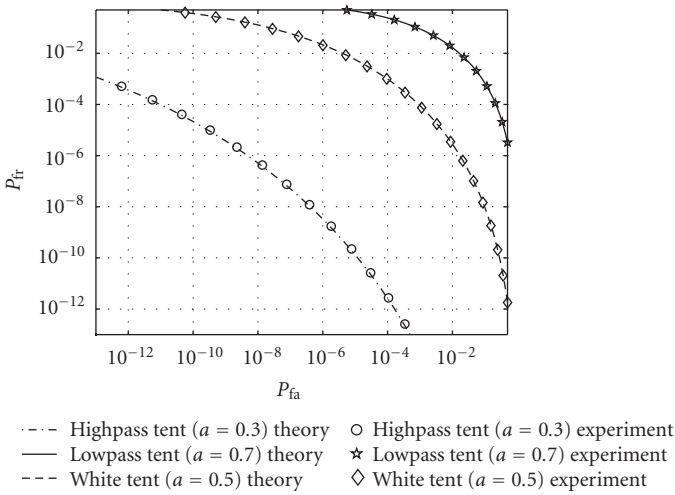
#### 8.2.3.2. Experimental performance evaluation

The results obtained through the theoretical analysis presented in the previous sections were verified experimentally. In order to do so, a watermarking system based on correlation detection was used to watermark music audio signals. In this section we will present results for such a signal of 1-second duration (sampling frequency 44.1 kHz, 16 bits per sample). The audio signal was assumed to comply with the signal model of (8.21) and the value of the audio signal autocorrelation parameter $\beta$ was experimentally evaluated to be equal to 0.97. The performance of the system was measured for three classes of watermark signals, that is, chaotic watermarks generated by tent maps with different spectral properties, pseudorandom white watermarks, and lowpass watermarks generated by Bernoulli chaotic sequences. A total of 10 000 keys for each class of signals were used in all the experiments. In all cases, a watermark embedding factor $p$ chosen so that the resulting watermarked signals had an SNR equal to 30 dB has been used. In the analysis below, ROC curve evaluation is performed under the worst case assumption for $P_{\text{fa}}$ evaluation (event $H_{1b}$). In other words, experiments performed for the evaluation of $P_{\text{fa}}$ were conducted upon signals being watermarked by a watermark different than the one used in detection.

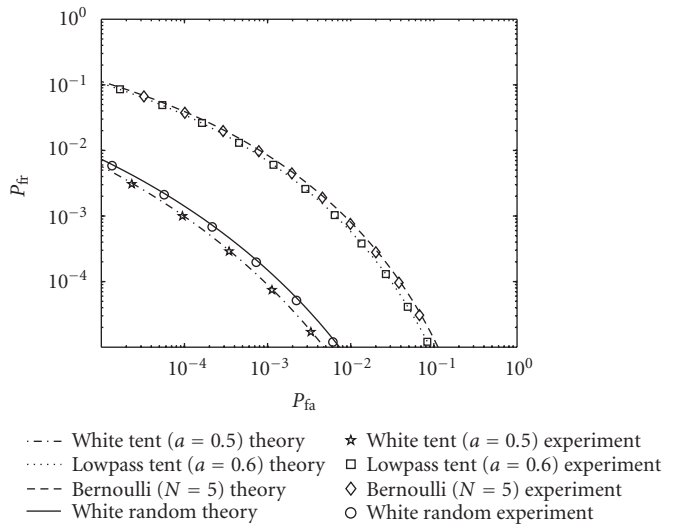
The statistical and spectral properties of the tent chaotic watermarks were experimentally evaluated and found to be in agreement with the analytical expressions presented above. Moreover, the experiments verified that the spectrum of tent chaotic watermarks can be easily controlled by the parameter $\alpha$: it is highpass for small values of $\alpha$ (i.e., values close to zero), becomes white for $\alpha = 0.5$, and tends to lowpass as $\alpha$ tends to one.

Next, the influence of the map parameter $\alpha$ on the performance of the watermarking system was evaluated in a set of experiments. ROC curves were evaluated both theoretically and experimentally for lowpass ($\alpha = 0.7$), white ($\alpha = 0.5$), and highpass ($\alpha = 0.3$) tent chaotic watermarks. The results verified the superior performance of the highpass tent chaotic watermarks, as can be observed in Figure 8.3(a). A considerably inferior performance is obtained when white tent watermarks are used, whereas lowpass watermarks result in the worst performance. Furthermore, the validity of the theoretical derivations was verified by the good agreement between the theoretical and experimental results.

The fact that schemes employing watermark signals of exactly the same spectral characteristics may exhibit different performance was also illustrated experimentally. In other words, it was verified that, in the case of event $H_1$, two watermarking schemes using sequences of the same spectral characteristics attain the same correlation mean and variance, but can exhibit different correlation variance when event $H_0$ holds, since in that case the variance depends also on the second- and third-order correlation statistics (8.27). For this purpose, white tent chaotic watermarks (i.e., tent watermarks generated using $\alpha = 0.5$) were compared against pseudorandom white watermarks. Moreover, lowpass tent watermarks were compared against lowpass Bernoulli watermarks, possessing exactly the same spectral characteristics.




(a)




(b)

FIGURE 8.3. (a) ROC curves for watermarking schemes based on highpass, lowpass, and white skew tent chaotic watermarks. (b) Comparison between sequences with the same spectral characteristics. (Reproduced from A. Tefas et al. "Performance analysis of correlation-based watermarking schemes employing Markov chaotic sequences," IEEE Transactions on Signal Processing, Vol. 51, no. 7, pages 1979–1994, © 2003 IEEE.)

Experimental results showed that sequences produced by white tent maps have superior performance when compared to white pseudorandom watermarks. Similarly, lowpass sequences generated by tent maps outperform lowpass Bernoulli

sequences of the same spectral characteristics. The experimentally and theoretically evaluated ROC curves for white tent ($\alpha = 0.5$), pseudorandom white, lowpass tent ($\alpha = 0.6$), and lowpass Bernoulli ($m = 5$) watermarks are plotted in Figure 8.3(b). In addition to the superiority of the tent maps, the figure verifies that theoretical and experimental ROCs are indeed in accordance.

### 8.2.4. Multiplicative embedding transform domain correlation-based scheme employing Markov chaotic sequences

According to the additive, correlation-based system analysis presented in Sections 8.2.3.1 and 8.2.3.2 above, when no distortions are imposed on the signal, highpass chaotic watermarks achieve superior performance when compared to those exhibiting lowpass or white spectral characteristics. However, in order to meet the requirement of robustness to common signal distortions and unauthorized intentional manipulations, the watermark should be embedded in the perceptually most significant components of the host data, which are usually the low-frequency regions [6, 18]. By doing so, an attacker would have to distort the fundamental data components, and thus cause severe quality degradation in his attempt to remove the watermark. Moreover, most signal processing techniques tend to leave such components intact. Therefore, a methodology that exploits the desirable properties of the high-frequency chaotic watermarks presented in the previous sections and in the same time performs watermark embedding in the perceptually significant low-frequency region of the signal should be devised. The method described in this section achieves this by embedding a highpass watermark in the low-frequency subbands of the DFT domain, where most of the host signal energy is concentrated in a way that does not impose significant distortions to the signal. Thus the method guarantees imperceptibility, improved detection reliability, and robustness to manipulations. For a more detailed description of the proposed system the reader can consult [11, 28].

Without restriction of the generality, 1D discrete-time signals will be considered. Let $s(n)$, $n = 0, 1, \ldots, N_s - 1$, be samples of the original 1D signal of length $N_s$ and let $S(k)$, $k = 0, 1, \ldots, N_s - 1$, be the DFT coefficients of $s(n)$. Watermark embedding is performed by modifying the magnitude $F(n) = |S(k)|$ of the DFT coefficients. In other words, the magnitude $F(n)$ will be considered hereafter as the host signal. The DC coefficient is considered to be located at $S(0)$ and thus indices of low-frequency DFT coefficients are close to zero or $N_s - 1$. Furthermore, the DFT magnitude is symmetric with respect to the coefficient $N_s/2$ since $s(n)$ is a real signal.

A basic watermark signal $W_o(i) \in \mathcal{R}$ is utilized for the construction of the final watermark $W(n)$. $W_o(i)$ consists of $N$ samples and is generated through an appropriate function $g$ (8.3). $W_o$ is embedded in the low-frequency components around zero. Due to the fact that the DFT magnitude is symmetric, a reflected version of $W_o$ is also embedded in the low-frequency components around $N_s - 1$. Since the final watermark signal $W(n)$ should affect a specific low-frequency

subband of the host signal, it can be represented by the following formula:

$$W(n) = \begin{cases} W_o(n - aN_s), & \text{if } aN_s \le n \le bN_s, \\ W_o(N_s - n - aN_s), & \text{if } (1-b)N_s \le n \le (1-a)N_s, \\ 0, & \text{otherwise,} \end{cases} \tag{8.35}$$

where $n = 0, 1, \ldots, N_s - 1$ and the coefficients $a$ and $b$ ($0 < a < b \le 0.5$) control the subband (frequency coefficients) that will be modified. Obviously, the length of the basic watermark is $N = \lceil (b-a)N_s \rceil$. Zero-mean watermarks that guarantee better system performance, as shown in [19], will be treated throughout this section.

A multiplicative superposition rule is used for the watermark embedding because it incorporates a simple perceptual masking effect by modifying coefficients proportionally to their magnitude:

$$F'(n) = F(n) + pW(n)F(n), \tag{8.36}$$

where $F'(n)$ is the watermarked signal and $p$ is the corresponding watermark embedding factor. Obviously $p$ is closely related to the watermark perceptibility as in the case of the additive watermark presented in the previous sections.

For a multiplicatively embedded watermark, the expression for the correlation detector (8.4) takes the following form:

$$c = \frac{1}{N}\sum_{n=0}^{N-1} F_t(n)W_d(n) = \frac{1}{N}\sum_{n=0}^{N-1} \left(F_o(n)W_d(n) + pW_e(n)F_o(n)W_d(n)\right). \tag{8.37}$$

For a test signal $F_t(n)$, and a watermark $W_d(n)$ whose existence in $F_t(n)$ is to be checked, the watermark detection procedure in the case of multiplicative embedding involves the following hypotheses:

(i) $H_0$: the test signal $F_t(n)$ contains the watermark $W_d(n)$, that is, $F_t(n) = F_o(n) + pW_d(n)F_o(n)$, $F_o(n)$ being the host signal, that is, the DFT magnitude,

(ii) $H_1$: the test signal $F_t(n)$ does not host the watermark $W_d(n)$.

Similar to the additive embedding case presented in Section 8.2.3, event $H_1$ can be further broken down into the events $H_{1a}$, corresponding to the case where the test signal is not watermarked, that is, $F_t(n) = F_o(n)$ and $H_{1b}$, corresponding to the case where the test signal is watermarked with a different watermark $W'_d(n) \neq W_d(n)$, that is, $F_t(n) = F_o(n) + pW'_d(n)F_o(n)$.

All these events can be combined in the following expression:

$$F_t(n) = F_o(n) + pW_e(n)F_o(n), \tag{8.38}$$

where the watermark $W_d(n)$ is indeed embedded in the signal if $p \neq 0$ and $W_e(n) = W_d(n)$ (event $H_0$), and it is not embedded in the signal if $p = 0$ (no watermark is present, event $H_{1a}$) or $p \neq 0$ and $W_e(n) \neq W_d(n)$ (a different watermark is present, event $H_{1b}$).

#### 8.2.4.1. Theoretical performance analysis

Using a similar justification as in the additive case, one can safely assume that, for sequences generated by piecewise linear Markov maps and pseudorandom sequences, the correlation $c$ in (8.37) obeys a Gaussian distribution and thus $f_{c|H_0}$, $f_{c|H_1}$ can be fully determined in terms of their means and variances whereas the ROC curve describing such a system is given by (8.18).

Using expression (8.37), the mean value of the correlation detector $c$ can be evaluated as follows:

$$\mu_c = E[c] = \frac{1}{N}\left(\sum_{n=0}^{N-1} E[F_o(n)]E[W_d(n)] + \sum_{n=0}^{N-1} pE[F_o(n)]E[W_e(n)W_d(n)]\right). \tag{8.39}$$

The corresponding expression for the variance $\sigma_c^2 = E[c^2] - E^2[c]$ of the correlation detector is given by

$$\begin{aligned}\sigma_c^2 = \frac{1}{N^2}\Bigg[&\sum_{n=0}^{N-1} \left(E[F_o^2(n)]E[W_d^2(n)] + p^2E[F_o^2(n)]E[W_d^2(n)W_e^2(n)]\right.\\ &\left.+ 2pE[F_o^2(n)]E[W_e(n)W_d^2(n)]\right)\\ &+ \sum_{n=0}^{N-1}\sum_{\substack{m=0,\\ m\neq n}}^{N-1} \left(E[F_o(n)F_o(m)]E[W_d(n)W_d(m)]\right.\\ &\qquad + pE[F_o(n)F_o(m)]E[W_e(m)W_d(n)W_d(m)]\\ &\qquad + pE[F_o(n)F_o(m)]E[W_e(n)W_d(n)W_d(m)]\\ &\qquad \left.+ p^2E[F_o(n)F_o(m)]E[W_e(n)W_e(m)W_d(n)W_d(m)]\right)\Bigg] - \mu_c^2.\end{aligned} \tag{8.40}$$

The above formulas are general and can be used to describe all three events, $H_0$, $H_{1a}$ and $H_{1b}$. In order to proceed with the theoretical analysis and derive closed-form expressions for $\mu_c$ and $\sigma_c^2$, all moments that appear in (8.39) and (8.40) should be estimated. To do so, the host signal, that is, the magnitude of the DFT coefficients, was assumed to be wide-sense stationary, following an exponential autocorrelation function like the one described in Section 8.2.3.1. Although these assumptions are rather simplistic, the validation of the theoretical results using experiments on audio signals in Section 8.2.4.2, verified that the assumptions hold in a great extent.

Before proceeding with the theoretical analysis, we will adopt the subtraction of the mean value $E[F_t(n)]$ from the test signal $F_t(n)$ prior to detection. This procedure, as mentioned in Section 8.2.3.1 and detailed in [29], improves the detection reliability for watermarks embedded in an additive way. It can be easily proven

that subtraction of the signal mean value improves the performance of multiplicative embedding watermarking schemes too. By utilizing zero-mean watermarks (8.7) and since for watermarks generated by chaotic maps $W_d[n] = W_e[n+k]$ (see (8.9)), $\sigma_c^2$ and $\mu_c$ in (8.40) and (8.39) can be shown to be

$$\begin{aligned}
\sigma_c^2 = {} & \frac{1}{N}\big(R_x[0,k,k] - 2\mu_x R_x[0,k] + 2\mu_x^2 R_x[0] \\
& \quad - 2\mu_x R_x[k,k] + 4\mu_x^2 R_x[k] - 3\mu_x^4\big)p^2 R_{F_o}[0] \\
& + \frac{1}{N}\big(2pR_x[k,k] - 4p\mu_x R_x[k] + (1-2p\mu_x)R_x[0] + 4p\mu_x^3 - \mu_x^2\big)\sigma_{F_o}^2 \\
& + \frac{2}{N^2}\Bigg[\sum_{m=1}^{N-1}(N-m)\big(R_{F_o}[m] - \mu_{F_o}^2\big)R_x[m] \\
& \qquad + \big(4p\mu_x^3 - \mu_x^2\big)\sum_{m=1}^{N-1}(N-m)\big(R_{F_o}[m] - \mu_{F_o}^2\big) \\
& \qquad - p\mu_x \sum_{m=1}^{N-1}(N-m)\big(R_{F_o}[m] - \mu_{F_o}^2\big) \\
& \qquad \times\big(2R_x[m] + 2R_x[k] + R_x[m+k] + R_x[k-m]\big) \\
& \qquad + p\sum_{m=1}^{N-1}(N-m)\big(R_{F_o}[m] - \mu_{F_o}^2\big)\big(R_x[k,k-m] + R_x[k,m+k]\big) \\
& \qquad + \sum_{m=1}^{N-1}(N-m)R_{F_o}[m] \\
& \qquad \times\{p^2 R_x[m,k,m+k] \\
& \qquad\quad - p^2\mu_x\big(R_x[m,k] + R_x[m,m+k] + R_x[k,k-m] + R_x[k,m+k]\big) \\
& \qquad\quad + p^2\mu_x^2\big(2R_x[m] + 2R_x[k] + R_x[k-m] + R_x[m+k]\big)\} \\
& \qquad - 3p^2\mu_x^4\sum_{m=1}^{N-1}(N-m)R_{F_o}[m]\Bigg] - \mu_c^2,
\end{aligned} \tag{8.41}$$

$$\mu_c = p\mu_{F_o}\big(R_x[k] - \mu_x^2\big). \tag{8.42}$$

The evaluation of expressions (8.42) and (8.41) requires the evaluation of several moments of the chaotic watermark $x[n]$ which can be derived using the methodology presented in Section 8.2.3.1 and detailed in [13, 26, 29].

The above equations continue to be general and encompass all three events, $H_0$, $H_{1a}$, $H_{1b}$. In detail, expressions for the event $H_o$ can be obtained by setting the sequence shift $k$ equal to zero and using an embedding factor $p > 0$. The event $H_{1a}$ can be described by an embedding factor $p$ having zero value whereas expressions for the event $H_{1b}$ can be obtained by using $k > 0$ and $p \neq 0$.

A close examination of (8.42) leads to the conclusion that the mean value becomes zero in the case of event $H_{1a}$. For the event $H_{1b}$, the mean value of the correlation detector output converges quickly to zero as $k$ increases. Furthermore, in the case of the correct watermark presence (event $H_0$), the mean value depends not only on the embedding factor and the variance of the watermark like in the additive watermark case (see (8.26)), but also on the mean value $\mu_{F_o}$ of the host signal (where $F_o$ refers to the DFT coefficients magnitude for the original signal). However, it is independent from the watermark spectrum. Therefore, for watermarks of the same embedding factor, the system performance is affected only by the variance of the correlation detector which in turn depends on the correlation (or, equivalently, spectral) characteristics of the watermark sequence. This remark justifies the use of sequences generated by chaotic maps as watermark signals, since one can exploit their easily controllable correlation/spectral characteristics to achieve improved system performance, as will be shown in Section 8.2.4.2.

Equations (8.42) and (8.41) can be modified so that they describe the two events $H_0$ and $H_{1a}$ in a simplified form:

$$\mu_c = \begin{cases} 0, & p = 0 (H_{1a}), \\ p\mu_{F_o}\sigma_x^2, & k = 0,\ p \neq 0 (H_0), \end{cases}$$

$$\sigma_c^2 = \begin{cases} \dfrac{1}{N}\sigma_x^2\sigma_{F_o}^2 + \dfrac{2}{N^2}\displaystyle\sum_{m=1}^{N-1}(N-m)\left(R_x[m]-\mu_x^2\right)\left(R_{F_o}[m]-\mu_{F_o}^2\right), & p = 0 (H_{1a}), \\ \dfrac{1}{N}\left(R_x[0,0,0] - 4\mu_x R_x[0,0] + 6\mu_x^2 R_x[0] - 3\mu_x^4\right) p^2 R_{F_o}[0] \\ \quad + \dfrac{1}{N}\left(2pR_x[0,0] + (1-6p\mu_x)R_x[0] + 4p\mu_x^3 - \mu_x^2\right)\sigma_{F_o}^2 \\ \quad + \dfrac{2}{N^2}\left[\displaystyle\sum_{m=1}^{N-1}(N-m)\left(R_{F_o}[m]-\mu_{F_o}^2\right)\left(R_x[m]+4p\mu_x^3-\mu_x^2\right)\right. \\ \qquad -2p\mu_x\displaystyle\sum_{m=1}^{N-1}(N-m)\left(R_{F_o}[m]-\mu_{F_o}^2\right)\left(2R_x[m]+R_x[0]\right) \\ \qquad +p\displaystyle\sum_{m=1}^{N-1}(N-m)\left(R_{F_o}[m]-\mu_{F_o}^2\right)\left(R_x[0,m]+R_x[m,m]\right)+\sum_{m=1}^{N-1}(N-m)R_{F_o}[m] \\ \qquad \times\left\{p^2R_x[0,m,m]\right. \\ \qquad\quad -2p^2\mu_x\left(R_x[0,m]+R_x[m,m]\right)+2p^2\mu_x^2\left(2R_x[m]+R_x[0]\right)\} \\ \qquad \left.-3p^2\mu_x^4\displaystyle\sum_{m=1}^{N-1}(N-m)R_{F_o}[m]\right] - \mu_c^2, \quad k = 0,\ p \neq 0 (H_0). \end{cases} \tag{8.43}$$

By inspecting the previous formula, it is obvious that the system performance depends on the watermark autocorrelation function $R_x[m]$, which, as mentioned in Section 8.2.3.1, is closely related to its power spectral density. As demonstrated in [29], highpass watermarks lead to reduced correlation variance for additive

embedding and, thus, better performance when no attacks are inflicted on the host signal. In the multiplicative embedding model under investigation, highpass watermarks still outperform lowpass and white spectrum watermarks as will be shown in Section 8.2.4.2. This fact is exploited by embedding highpass watermarks in the low-frequency subbands in order to take advantage of the correlation properties of such watermarks and, at the same time, achieve robustness to attacks of lowpass nature.

By substituting the correlation statistics of the skew tent sequences (presented in detail in [29]) in (8.43), the mean value and the variance of the correlation detector for multiplicative embedding watermarking systems employing such sequences can be derived:

$$\mu_c = \begin{cases} 0, & p = 0 (H_{1a}), \\ \dfrac{p\mu_{F_o}}{12}, & k = 0,\ p \neq 0 (H_0), \end{cases}$$

$$\sigma_c^2 = \begin{cases} \dfrac{\sigma_{F_o}^2}{12N^2} \dfrac{N - 2\beta e_2 - N\beta^2 e_2^2 + 2(\beta e_2)^{N+1}}{(1-\beta e_2)^2}, & p = 0 (H_{1a}), \\ \dfrac{\sigma_{F_o}^2}{N}\left(\dfrac{1}{12} + \dfrac{p^2}{80}\right) + \dfrac{p^2\mu_{F_o}^2}{180N} \\ \quad + \dfrac{2\sigma_{F_o}^2}{N^2}\left[\dfrac{3a-2-ap}{12(3a-2)} \dfrac{(\beta e_2)^{N+1} - N\beta^2 e_2^2 + N\beta e_2 - \beta e_2}{(1-\beta e_2)^2}\right. \\ \qquad + \dfrac{p^2}{144} \dfrac{\beta^{N+1} - N\beta^2 + N\beta - \beta}{(1-\beta)^2} \\ \qquad \left. + \dfrac{15ap + 3ap^2 - 2p^2}{180(3a-2)} \dfrac{(\beta e_1)^{N+1} - N\beta^2 e_1^2 + N\beta e_1 - \beta e_1}{(1-\beta e_1)^2}\right] \\ \quad + \dfrac{p^2\mu_{F_o}^2}{90N^2} \dfrac{e_1^{N+1} - Ne_1^2 + Ne_1 - e_1}{(1-e_1)^2}, & k = 0,\ p \neq 0 (H_0), \end{cases} \tag{8.44}$$

where $e_1 = 3\alpha^2 - 3\alpha + 1$, $e_2 = 2\alpha - 1$ are eigenvalues of the FP matrix $\mathbf{P}_3$ and $\beta$ is the parameter of the host signal autocorrelation function given by (8.21).

### 8.2.4.2. Experimental performance evaluation

The multiplicative embedding transform domain watermarking model was tested on a number of mono music audio signals. The DFT coefficients of these signals were used as host signals. Results will be presented below for such a signal of approximately 5.94-second duration, sampled at 44.1 kHz with 16 bits per sample. In the subsequent analysis, experiments were performed by employing chaotic watermark signals generated by the skew tent map. All sets of experiments were conducted using 10 000 keys and the system detection performance was measured

in terms of the ROC curves. In all experiments, a skew tent map parameter $\alpha = 0.3$ has been used, leading to highpass watermarks.

In the first set of experiments, the just-noticeable distortion for watermark embedding in different frequency bands of the host signal was evaluated. For this purpose, six overlapping frequency subbands of lengths equal to the 10% of the entire host signal were used. For each of these bands, their symmetric band was also utilized, and thus 20% of the host signal samples were watermarked. The selected subbands varied from very low frequencies ($a = 0.01$, $b = 0.11$) to middle frequencies ($a = 0.15$, $b = 0.25$) and high frequencies ($a = 0.30$, $b = 0.40$), with $a$, $b$ being the coefficients in (8.35). For each of these frequency bands, the embedding factor $p$ was increased gradually and a number of listeners were asked to verify that signal distortions could not be perceived. The point, where any additional increase of $p$ would lead to perceptible distortions, was considered as the maximum watermark embedding factor $p$ tolerated by this band, namely, the just noticeable distortion level. Results proved that watermark embedding in the lowest frequency band ($a = 0.01$, $b = 0.11$) provides great perceptual capacity, enabling the use of low SNR values without resulting in perceptual distortion of the host signal.

In the second set of experiments, the influence of the frequency band selection on the detection performance, both with and without attacks was investigated. To do so, the method under investigation was tested against lossy MPEG audio-I layer III compression at 64 kbps. The ROC curves for the undistorted and the compressed signal can be seen in Figure 8.4. Inspection of the curves reveals that embedding in the lowest subband ($a = 0.01$, $b = 0.11$) attains by far the highest robustness to compression. High-frequency bands ($a = 0.20$, $b = 0.30$ and $a = 0.30$, $b = 0.40$) exhibit better performance than other subbands when no distortions occur on the signal (Figure 8.4(a)). In fact, it was not possible to plot the ROC curve for the subband with $a = 0.30$, $b = 0.40$ in Figure 8.4(a) within the selected axis range. However, the content of these bands is severely affected by the compression algorithm and thus these bands do not provide sufficient robustness to attacks of this nature. The lowest frequency band ($a = 0.01$, $b = 0.11$) was ranked third when no distortions were imposed on the signal. The results of this and the previous experiments prove that this band is the best choice for embedding as it combines superior robustness to attacks of lowpass nature, sufficient performance in the case of distortion-free signals and can tolerate significant watermark-induced distortions. For these reasons, the lowest frequency subband was selected for embedding.

In order to compare the performance of the presented watermarking scheme against the performance of alternative techniques, experiments were conducted for two competitive embedding schemes. For all methods, watermarks that are just below the just noticeable distortion have been incorporated. For all three schemes, a correlation detector, applied in the appropriate domain, was used. The first alternative embedding scheme involved white pseudorandom watermark sequences ($w(i) \in \{-1, 1\}$) that were embedded multiplicatively in the same low-frequency DFT subband ($a = 0.01$, $b = 0.11$), producing watermarked signals

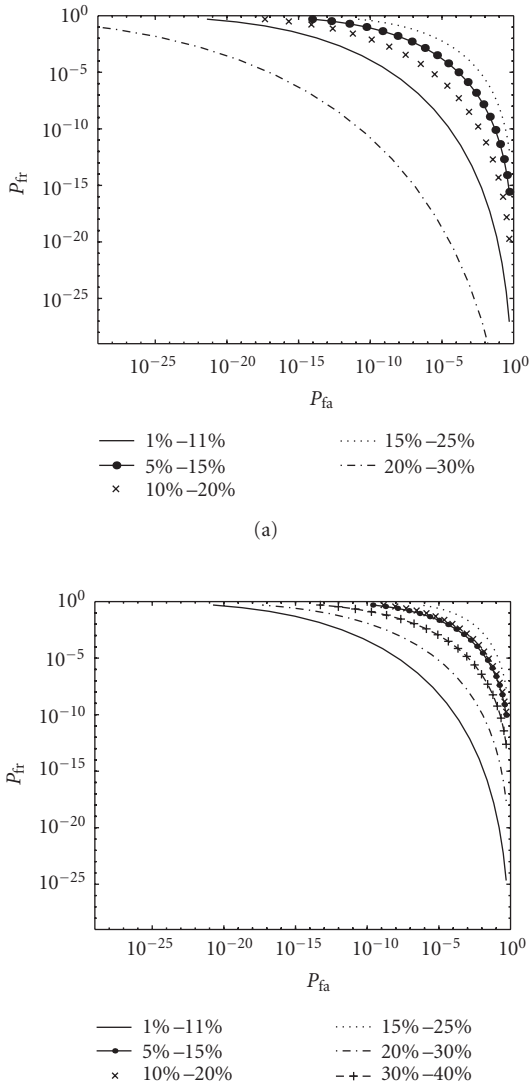



FIGURE 8.4. Experimental ROC curves for various frequency bands (skew tent sequences): (a) five frequency bands without attacks, (b) six frequency bands after MPEG compression at 64 kbps. (Figure 8.4(b) is reproduced from A. Tefas et al. "Enhanced transform-domain correlation-based audio watermarking," 2005 IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol. 2, pages 1049–1052, Philadelphia, © 2005 IEEE.)

with SNR = 23 dB. The second scheme was based on the time-domain audio watermarking scheme proposed in [2]. In this technique, a bipolar white pseudorandom watermark $w(n)$ ($w(n) \in \{-1, 1\}$) was modulated according to the

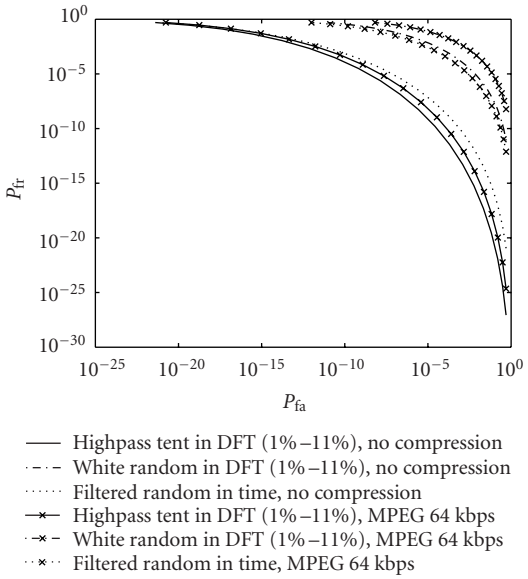



FIGURE 8.5. ROC curves for the three watermarking schemes (tent and white watermarks in the DFT domain and prefiltered watermarks in the time domain) in distortion-free signals and after MPEG compression at 64 kbps. (Reproduced from A. Tefas et al. "Enhanced transform-domain correlation-based audio watermarking," 2005 IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol. 2, pages 1049–1052, Philadelphia, © 2005 IEEE.)

amplitude of the original audio signal $m(n)$ using a multiplicative law:

$$w'(n) = p\,|\,m(n)\,|\,w(n), \tag{8.45}$$

with $p$ being the embedding factor. In the next stage, a 25th-order lowpass Hamming filter with cut-off frequency of 2205 Hz was applied on $w'(n)$ in order to improve imperceptibility and robustness to lowpass attacks. The resulting filtered watermark signal $w''(n)$ was embedded in the time domain of the original signal $m(n)$:

$$m_w(n) = m(n) + w''(n), \tag{8.46}$$

producing the watermarked signal $m_w(n)$ with SNR = 22 dB. Figure 8.5 illustrates the superior performance of highpass tent watermarks embedded in the low frequencies of the DFT transform against the two techniques described above both when no distortions occur and when 64 kbps MPEG-I layer III encoding is applied on the signal. In addition, the proposed method exhibited better performance than the other methods in a number of other attacks that included mean and median filtering and resampling. Details can be found in [28].

(a) (b)

FIGURE 8.6. (a) A sample image that can be used as logo for the watermark generation. (b) The image Lenna to be watermarked by using this logo.

## 8.3. Watermark generation by chaotic mixing

The watermarking systems described in Section 8.2 utilize zero-bit watermarks. In certain applications however, multiple-bit watermarks, that is, watermarks that can encode a number of information bits, are required. Watermarks that can embed an image logo within the host medium are an example of multiple-bit watermarks. A logo $l(\mathbf{x}) \in \{1, 2\}$ is usually a binary image of dimensions $N_1 \times N_2$ (Figure 8.6(a)). Using a logo as the information to be embedded is of interest in cases where a visually meaningful detected watermark whose integrity can be verified by visual inspection is desired. In such a case, a mechanism for constructing a watermark signal starting from the binary logo is required. This mechanism should be able to scramble or mix the logo and produce a watermark signal with the desired characteristic. Scrambling a logo is needed in order to prevent statistical estimation of the final watermark by an attacker. The same need arises when a nonvisually meaningful multiple-bit message has to be encrypted prior to embedding in order to increase security. Such a mixing procedure can be implemented efficiently by using chaotic mixing systems [10].

Chaotic maps used for chaotic mixing provide excellent security and have many desired cryptographic properties. They are simple to implement and achieve high encryption speed. Their basic property that makes them suitable for information encryption is their sensitivity to initial conditions. When a chaotic map is iteratively applied to two closely positioned points, their positions diverge quickly. Thus, local logo image correlation is destroyed. Another advantage of chaotic mixing is that it can lead to a private/public key watermarking scheme. That is, using chaotic mixing, one can have a certain key (private key) for watermark embedding and a different key (public key) for watermark detection. In that case, a potential attacker can not use the parameters of the public key for embedding, since this will result in a different watermark.

In the following sections, a description of a watermarking system that incorporates chaotic mixing will be provided. Readers interested in obtaining additional details can consult [30, 33].

### 8.3.1. Watermark generation

A 2D chaotic mixing can be considered as spatial transformation of planar regions. It is represented by a map

$$\mathcal{A} : U^2 \longrightarrow U^2, \quad U = [0,1) \subset \mathcal{R}, \tag{8.47}$$

and is defined by the formula

$$\mathbf{x}_{n+1} = (\mathbf{A}\mathbf{x}_n) \bmod 1, \quad \mathbf{x} \in U^2, \tag{8.48}$$

where $\mathbf{A}$ is a square matrix having integer elements, $\det \mathbf{A} = 1$, and mod 1 operator is the modulo operator that gives the decimal part of every element of $\mathbf{A}\mathbf{x}_n$.

Successive applications of the map $\mathcal{A}$ on a point $\mathbf{x}_0$ form a dynamical system described by the following iterative process:

$$\mathbf{x}_n = (\mathbf{A}^n \mathbf{x}_0) \bmod 1. \tag{8.49}$$

Although these systems are strongly chaotic, they possess a set of periodic orbits. An orbit is periodic if it is finite, that is, there exists a number $T$ of iterations such that $\mathbf{x}_T = \mathbf{x}_0$. The necessary and sufficient condition for an orbit to be periodic is for the initial position $\mathbf{x}_0$ to have rational coordinates [1]. The interested reader may refer to [1, 24] for details on chaotic mixing and to [33] for an application of chaotic mixing to copyright protection.

Chaotic maps can be easily applied on discrete lattices (e.g., images) of dimensions $N \times N$, $N \in \mathcal{Z}$, by modifying (8.48) to

$$\mathbf{x}_{n+1} = (\mathbf{A}\mathbf{x}_n) \bmod N, \quad \mathbf{x} \in \mathcal{F} = [0, N-1] \times [0, N-1] \subset \mathcal{Z}^2. \tag{8.50}$$

In the case of images, the lattice location $\mathbf{x}$ is equivalent to the position of a pixel. The least common multiple of the periods of all orbits in an image is called recurrence time. It is obvious that

$$(\mathbf{A}^R \mathbf{x}) = \mathbf{x}, \quad \forall \mathbf{x} \in \mathcal{F}, \tag{8.51}$$

where $R$ indicates the recurrence time of the image. The recurrence time is determined by the dimension $N$ of the square lattice and the type of chaotic map that is used for mixing.

Considering that in chaotic maps $\det \mathbf{A} = 1$, the following formula can be used to define a two-parameter family of maps:

$$\mathbf{A} = \begin{pmatrix} 1 & a \\ b & ab+1 \end{pmatrix}, \tag{8.52}$$

where $a, b \in \mathcal{Z}$. The above map can be used for mixing an image logo $l(\mathbf{x})$ in order to create the final watermark $w(\mathbf{x})$. First, the logo is superimposed at the position

**m** of a larger region (i.e., a region equal in size to the host image) to form the ternary unmixed watermark $w'(\mathbf{x})$:

$$w'(\mathbf{x}) = \begin{cases} l(\mathbf{x}-\mathbf{m}) & \text{if } \mathbf{m} \le \mathbf{x} \le \mathbf{m} + (N_1, N_2), \\ 0 & \text{otherwise.} \end{cases} \tag{8.53}$$

If the number of pixels in the image logo is smaller or bigger than the number of the host image pixels that we want to affect through watermarking, the logo image can be resized or repeated in different positions so that its pixel number is close to the desirable number. The above procedure results in an initial ternary unmixed watermark $w'(\mathbf{x}) \in \{0, 1, 2\}$.

Subsequently, the watermark $w'(\mathbf{x})$ is mixed using the chaotic mixing procedure described above to generate the final watermark $w(\mathbf{x}) \in \{0, 1, 2\}$ which is subsequently used for image watermarking. Instead of a single chaotic mixing step, a series of such steps with different parameters can be applied on $w'(\mathbf{x})$ for additional security and uncorrelated spatial spreading of the initial logo. In that case, the position of a pixel after $i+1$ mixing steps is given by

$$\mathbf{x}_{i+1} = (\mathbf{A}_i^{n_i}\mathbf{x}_i) \bmod N_i, \quad \mathbf{x}_i \in \mathcal{F} = [0, N_i - 1] \times [0, N_i - 1] \subset \mathcal{Z}^2, \tag{8.54}$$

where $\mathbf{x}_i$ is the position of the pixel after $i$ mixing procedures. Accordingly, the inverse procedure can be applied iteratively to the mixed image resulting in the initial image:

$$\mathbf{x}_i = (\mathbf{A}_i^{R_i - n_i}\mathbf{x}_{i+1}) \bmod N_i, \quad \mathbf{x}_{i+1} \in \mathcal{F} = [0, N_i - 1] \times [0, N_i - 1] \subset \mathcal{Z}^2, \tag{8.55}$$

where $R_i$ is the recurrence time for a given map $\mathbf{A}_i$ and dimension $N_i$.

All the parameters of the mixing procedure except for the parameters $n_i$ in (8.54) are generated using a pseudorandom sequence having as a seed the owner's key $k$. If the system is to be used within a private/public key framework, the private key is comprised of the owner's key $k$ and the parameters $n_i$ in (8.54). For example, if three mixing steps are used $K_{\text{pr}} = \{k, n_1, n_2, n_3\}$. The public key is produced after the embedding and is comprised of the owner's key $k$ and the parameters $R_i - n_i$ in (8.55), that is, for three mixing steps $K_{\text{pub}} = \{k, R_1 - n_1, R_2 - n_2, R_3 - n_3\}$.

The initial unmixed watermark with the logo superimposed in several randomly selected positions is illustrated in Figure 8.7(a). The result of the watermark mixing using three steps with different parameters according to (8.54) is shown in Figures 8.7(b), 8.7(c), 8.7(d). The inverse procedure for recovering the embedded logo according to (8.55) is illustrated in Figure 8.8.

The chaotic mixing watermark generation procedure described in this section can be applied to any watermarking scheme for generating the watermark sequence. In the two sections that follow, an embedding and a detection scheme that can be combined with this generation procedure will be presented.

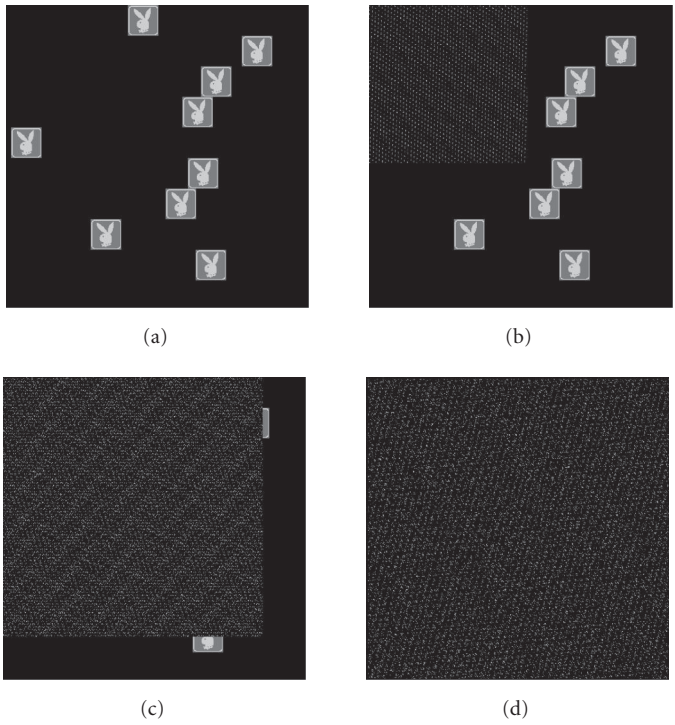


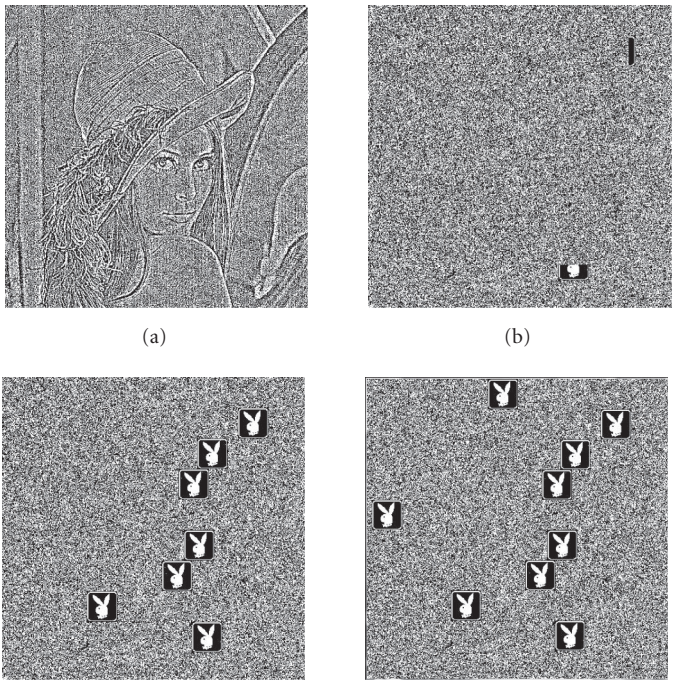
FIGURE 8.7. The watermark generation procedure by using chaotic mixing. (a) Initial random positions of the logo. (b) First mixing $M_1$ with parameters generated from the owner's watermark key. (c) Second mixing $M_2$ with parameters generated from the owner's watermark key. (d) Final watermark image generated by a third mixing $M_3$. (Reproduced from A. Tefas and I. Pitas, "Image authentication using chaotic mixing systems," 2000 IEEE International Symposium on Circuits and Systems, Vol. 1, pages 216–219, Geneva, Switzerland, © 2000 IEEE.)

### 8.3.2. Watermark embedding

Watermarks generated through the procedure described in the previous section can be embedded in images by altering the intensity of the pixels of the host image $f(\mathbf{x})$ according to the following formula:

$$f_w(\mathbf{x}) = \begin{cases} f(\mathbf{x}) & \text{if } w(\mathbf{x}) = 0, \\ g_1\big(f(\mathbf{x}), \mathcal{N}(\mathbf{x})\big) & \text{if } w(\mathbf{x}) = 1, \\ g_2\big(f(\mathbf{x}), \mathcal{N}(\mathbf{x})\big) & \text{if } w(\mathbf{x}) = 2, \end{cases} \tag{8.56}$$

where $g_1, g_2$ are appropriate embedding functions and $\mathcal{N}(\mathbf{x})$ is a local operator that involves pixels in a neighborhood around $\mathbf{x}$. The computation of $\mathcal{N}(\mathbf{x})$ does not include the pixel $\mathbf{x}$. The functions $g_1, g_2$ are selected so that suitable detection functions $G(f_w(\mathbf{x}), \mathcal{N}(\mathbf{x}))$—that is, functions that, when applied on the watermarked



(a) (b)

(c) (d)

FIGURE 8.8. The watermark detection (logo recovering) procedure by using chaotic mixing. (a) Initial detection image by applying the detection function in the image. (b) First inverse mixing $M_3^{-1}$ with parameters generated from the owner's watermark key. (c) Second inverse mixing $M_2^{-1}$ with parameters generated from the owner's watermark key. (d) Final detection image generated by the third inverse mixing $M_1^{-1}$. (Reproduced from A. Tefas and I. Pitas, "Image authentication using chaotic mixing systems," 2000 IEEE International Symposium on Circuits and Systems, Vol. 1, pages 216–219, Geneva Switzerland, © 2000 IEEE.)

image $f_w(\mathbf{x})$, give the watermark $w(\mathbf{x})$—can be devised:

$$G\big(f_w(\mathbf{x}), \mathcal{N}(\mathbf{x})\big) = w(\mathbf{x}). \tag{8.57}$$

Obviously, several embedding functions along with the corresponding detection function can be designed, thus resulting in different watermarking schemes. One such function is based on a superposition of real quantities on the intensities of the pixels which are to be altered by the watermark:

$$\begin{aligned} g_1\big(f(\mathbf{x}), \mathcal{N}(\mathbf{x})\big) &= \mathcal{N}(\mathbf{x}) \otimes \alpha_1 f(\mathbf{x}), \\ g_2\big(f(\mathbf{x}), \mathcal{N}(\mathbf{x})\big) &= \mathcal{N}(\mathbf{x}) \otimes \alpha_2 f(\mathbf{x}), \end{aligned} \tag{8.58}$$

where $\alpha_1$, $\alpha_2$ are suitably chosen constants. The operator $\otimes$ is a generalized superposition operator which includes appropriate data truncation and quantization,

if needed. For example, the values that can be used for the parameters of the system are $a_1 = -a_2 = 0.03$. For $\mathcal{N}(\mathbf{x})$, a location estimation operator can be used. The sign of $\alpha_1$, $\alpha_2$ is used in the detection function (see next section) and their magnitude determines the watermark embedding factor.

### 8.3.3. Watermark detection

The first step of the detection procedure is the generation of the watermark $w(\mathbf{x})$ according to the procedure described in Section 8.3.1. The detection function resulting from (8.58) is defined by

$$G\big(f_w(\mathbf{x}), \mathcal{N}(\mathbf{x})\big) = \begin{cases} 1 & \text{if } f_w(\mathbf{x}) - \mathcal{N}(\mathbf{x}) > 0, \\ 2 & \text{if } f_w(\mathbf{x}) - \mathcal{N}(\mathbf{x}) < 0. \end{cases} \tag{8.59}$$

The detection function (8.59) is valid if $\alpha_1 > 0$ and $\alpha_2 < 0$ in (8.58). By employing the detection function in the watermarked image, a bivalued detection image $d(\mathbf{x})$ is produced:

$$d(\mathbf{x}) = G\big(f_w(\mathbf{x}), \mathcal{N}(\mathbf{x})\big). \tag{8.60}$$

By comparing the watermark $w(\mathbf{x})$ and the detection image $d(\mathbf{x})$, the false detection image is formed:

$$e_w(\mathbf{x}) = \begin{cases} 1 & \text{if } w(\mathbf{x}) \neq 0,\ w(\mathbf{x}) \neq d(\mathbf{x}), \\ 0 & \text{otherwise.} \end{cases} \tag{8.61}$$

The false detection image has value 1 if a watermarked pixel is falsely detected and 0 otherwise. The detection ratio is given by the ratio of the correctly detected pixels to the sum of the watermarked pixels in the image:

$$D_w = 1 - \frac{\text{card}\,\{e_w(\mathbf{x})\}}{\text{card}\,\{w(\mathbf{x})\}}. \tag{8.62}$$

The embedding functions are designed in such way so that the probability of a pixel to be detected as signed with $g_1$ or $g_2$ for an unwatermarked image is $p = 0.5$. Thus, the detection ratio in an unwatermarked image forms a binomial distribution. In order to reach a decision on whether the image under investigation is watermarked or not, the watermark detection ratio of the image is compared against a predefined threshold $T$.

### 8.4. Other applications of chaotic systems in digital watermarking

Watermarking techniques based on chaotic systems appeared in the literature already in the first years of watermarking research [32]. An overview of early chaotic watermarking techniques can be found in [22]. In [27] the authors proposed an informed embedding blind audio watermarking scheme that incorporates differential chaos shift keying (DCSK) for watermark embedding. According to the DCSK principle which is used in chaotic wideband communication systems [15], every watermark bit is represented by two chaotic functions that are successive in time. The first function acts as a reference while the second carries the bit information. Bit 1 is represented by a reference chaotic signal followed by a copy of this signal, whereas bit 0 is represented by the reference chaotic signal followed by an inverted copy of the same signal. Watermark detection is accomplished by a differential coherent detector that delays the received signal by half the bit duration and evaluates the correlation (over half the bit duration) between the received signal and its delayed copy. A bit 0 (bit 1) is detected if the correlation is found to be negative (positive) and its absolute value is approximately equal to the signal energy over half the bit period. A logistic map is used in [7] in order to select in a random way the positions of the $8 \times 8$ blocks of the image that will form a subimage (equal in size to the 1/4th of the original image) to be used subsequently during embedding. A logistic map is also used (along with appropriate thresholding) to construct the binary watermark that is embedded in an additive way in the DWT coefficients of the subimage. The system employs a correlation-based detector. In [34] the authors use a binary signal generated by thresholding a chaotic map (namely, the ICMIC map) in order to encrypt (scramble) a binary image before embedding it in a host image. Encryption is performed by XORing the binary signal with the binary image. In addition, another binary signal generated from a chaotic map is used in order to split host image pixels into two sets that are used in the proposed watermark embedding and detection scheme that very much resembles the algorithm proposed in [21]. A chaotic dynamical system is used in [20] as part of a steganographic scheme that uses images as the host medium for the transmission of secret messages. The so-called chaotic asymmetric steganography scheme utilizes a 2D chaotic map (toral automorphism) to diffuse (scramble) a message and decide on the positions of the host image where the encrypted message bits are to be embedded. Embedding is done by changing the pixel intensity values in the neighborhood of a selected pixel so that a certain condition is imposed. The system is an asymmetric one, involving both a public and a private key.

### 8.5. Summary

Two applications of chaotic systems in digital watermarking were presented in this chapter. The first deals with the use of sequences generated by chaotic systems as watermark signals in schemes employing correlation-based detection. Two different alternatives were presented and analyzed both theoretically and experimentally: additive watermark embedding and multiplicative embedding in the low

frequencies of the DFT domain in order to achieve robustness to lowpass manipulations. Since the system performance depends on the spectral properties of the watermark signal, the controllable spectral/correlation properties of Markov chaotic watermarks was proven beneficial in this context. Highpass chaotic watermarks proved to perform better than white pseudorandom ones. Moreover, Markov maps having appropriate second- and third-order correlation statistics, like the skew tent map, were found to perform better than sequences with the same spectral properties generated by either Bernoulli or pseudorandom number generators.

The second application considered chaotic mixing as a means of encrypting an image logo prior to its embedding on a host image. two-dimensional chaotic maps were shown to provide a computationally efficient and secure way of performing this task.

## Acknowledgment

The authoring of this chapter has been supported in part by the European Commission through the IST Programme under Contract IST-2002-507932 ECRYPT.

## Bibliography

[1] D. K. Arrowsmith and C. M. Place, *An Introduction to Dynamical Systems*, Cambridge University Press, Cambridge, UK, 1990.

[2] P. Bassia, I. Pitas, and N. Nikolaidis, "Robust audio watermarking in the time domain," *IEEE Transactions on Multimedia*, vol. 3, no. 2, pp. 232–241, 2001.

[3] P. Billingsley, *Probability and Measure*, John Wiley & Sons, New York, NY, USA, 1995.

[4] A. Boyarsky and M. Scarowsky, "On a class of transformations which have unique absolutely continuous invariant measures," *Transactions of the American Mathematical Society*, vol. 255, pp. 243–262, 1979.

[5] I. J. Cox, M. Miller, and J. Bloom, *Digital Watermarking: Principles and Practice*, Morgan Kaufmann, San Francisco, Calif, USA, 2001.

[6] I. J. Cox, J. Kilian, F. T. Leighton, and T. Shamoon, "Secure spread spectrum watermarking for multimedia," *IEEE Transactions Image Processing*, vol. 6, no. 12, pp. 1673–1687, 1997.

[7] Z. Dawei, C. Guanrong, and L. Wenbo, "A chaos-based robust wavelet-domain watermarking algorithm," *Chaos, Solitons & Fractals*, vol. 22, no. 1, pp. 47–54, 2004.

[8] G. Depovere, T. Kalker, and J.-P. Linnartz, "Improved watermark detection reliability using filtering before correlation," in *Proceedings of International Conference on Image Processing (ICIP '98)*, vol. 1, pp. 430–434, Chicago, Ill, USA, October 1998.

[9] B. Macq Ed., "Special issue on identification & protection of multimedia information," *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1059–1304, 1999.

[10] J. Fridrich, A. C. Baldoza, and R. J. Simard, "Symmetric ciphers based on 2d maps," in *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics (SMC '97)*, pp. 1105–1110, Orlando, Fla, USA, October 1997.

[11] A. Giannoula, A. Tefas, N. Nikolaidis, and I. Pitas, "Improving the detection reliability of correlation-based watermarking techniques," in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME '03)*, vol. 1, pp. 209–212, Baltimore, Md, USA, July 2003.

[12] R. C. Hilborn, *Chaos and Nonlinear Dynamics: An Introduction for Scientists and Engineers*, Oxford University Press, Oxford, UK, 1994.

[13] S. H. Isabelle and G. W. Wornell, "Statistical analysis and spectral estimation techniques for one-dimensional chaotic signals," *IEEE Transactions on Signal Processing*, vol. 45, no. 6, pp. 1495–1506, 1997.
[14] N. S. Jayant and P. Noll, *Digital Coding of Waveforms: Principles and Applications to Speech and Video*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1984.
[15] M. P. Kennedy and G. Kolumbán, "Digital communications using chaos," *Signal Processing*, vol. 80, no. 7, pp. 1307–1320, 2000.
[16] T. Kohda, H. Fujisaki, and S. Ideue, "On distributions of correlation values of spreading sequences based on Markov information sources," in *Proceedings of International Symposium on Circuits and Systems (ISCAS '00)*, vol. 5, pp. 225–228, Geneva, Switzerland, May 2000.
[17] A. Lasota and M. C. Mackey, *Probabilistic Properties of Deterministic Systems*, Cambridge University Press, Cambridge, UK, 1985.
[18] J. Liang, P. Xu, and T. D. Tran, "A robust dct-based low frequency watermarking scheme," in *Proceedings of 34th Annual Conference in Information Systems and Science (CISS '00)*, vol. I, pp. 1–6, Princeton, NJ, USA, March 2000.
[19] J.-P. Linnartz, T. Kalker, and G. Depovere, "Modeling the false alarm and missed detection rate for electronic watermarks," in *Proceedings of 2nd International Workshop on Information Hiding (IHW '98)*, pp. 329–343, Portland, Ore, USA, April 1998.
[20] D.-C. Lou and C.-H. Sung, "A steganographic scheme for secure communications based on the chaos and Euler theorem," *IEEE Transactions on Multimedia*, vol. 6, no. 3, pp. 501–509, 2004.
[21] N. Nikolaidis and I. Pitas, "Robust image watermarking in the spatial domain," *Signal Processing*, vol. 66, no. 3, pp. 385–403, 1998, Special issue on copyright protection and access control.
[22] N. Nikolaidis, S. Tsekeridou, A. Nikolaidis, A. Tefas, V. Solachidis, and I. Pitas, "Applications of chaotic signal processing techniques to multimedia watermarking," in *Proceedings of IEEE Workshop on Nonlinear Dynamics in Electronic Systems (NDES '00)*, pp. 1–7, Catania, Italy, May 2000.
[23] E. Ott, *Chaos in Dynamical Systems*, Cambridge University Press, Cambridge, UK, 1993.
[24] I. Percival and F. Vivaldi, "Arithmetical properties of strongly chaotic motions," *Physica D*, vol. 25, no. 1–3, pp. 105–130, 1987.
[25] S. N. Rasband, *Chaotic Dynamics of Nonlinear Systems*, John Wiley & Sons, New York, NY, USA, 1990.
[26] T. Schimming, M. Gotz, and W. Schwarz, "Signal modeling using piecewise linear chaotic generators," in *Proceedings of 9th European Signal Processing Conference (EUSIPCO '98)*, pp. 1377–1380, Rhodes, Greece, September 1998.
[27] G. C. M. Silvestre, N. J. Hurley, G. S. Hanau, and W. J. Dowling, "Informed audio watermarking scheme using digital chaotic signals," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '01)*, vol. 3, pp. 1361–1364, Salt Lake City, Utah, USA, May 2001.
[28] A. Tefas, A. Giannoula, N. Nikolaidis, and I. Pitas, "Enhanced transform-domain correlation-based audio watermarking," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '05)*, vol. 2, pp. 1049–1052, Philadelphia, Pa, USA, March 2005.
[29] A. Tefas, A. Nikolaidis, N. Nikolaidis, V. Solachidis, S. Tsekeridou, and I. Pitas, "Performance analysis of correlation-based watermarking schemes employing Markov chaotic sequences," *IEEE Transactions Signal Processing*, vol. 51, no. 7, pp. 1979–1994, 2003.
[30] A. Tefas and I. Pitas, "Image authentication using chaotic mixing systems," in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS '00)*, vol. 1, pp. 216–219, Geneva, Switzerland, May 2000.
[31] S. Tsekeridou, V. Solachidis, N. Nikolaidis, A. Nikolaidis, A. Tefas, and I. Pitas, "Statistical analysis of a watermarking system based on Bernoulli chaotic sequences," *Signal Processing*, vol. 81, no. 6, pp. 1273–1293, 2001, Special issue on information theoretic issues in digital watermarking.
[32] G. Voyatzis and I. Pitas, "Chaotic mixing of digital images and applications to wateramrking," in *Proceedings of European Conference on Multimedia Applications, Services and Techniques (ECMAST '96)*, vol. 2, pp. 687–694, Louvain-la-Neuve, Belgium, May 1996.
[33] G. Voyatzis and I. Pitas, "Digital image watermarking using mixing systems," *Computers & Graphics*, vol. 22, no. 4, pp. 405–416, 1998.

[34] D.-J. Wang, L.-G. Jiang, and G.-R. Feng, "Novel blind non-additive robust watermarking using 1-D chaotic map," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '04)*, vol. 3, pp. 417–420, Montreal, Quebec, Canada, May 2004.

Nikos Nikolaidis: Department of Informatics, Aristotle University of Thessaloniki, 54124 Thessaloniki, Greece

*Email*: nikolaid@aiia.csd.auth.gr

Anastasios Tefas: Department of Informatics, Aristotle University of Thessaloniki, 54124 Thessaloniki, Greece

*Email*: tefas@aiia.csd.auth.gr

Ioannis Pitas: Department of Informatics, Aristotle University of Thessaloniki, 54124 Thessaloniki, Greece

*Email*: pitas@aiia.csd.auth.gr

# 9 Modeling of evolving textures using granulometries

A. J. Gray, S. Marshall, and J. McKenzie

## 9.1. Introduction

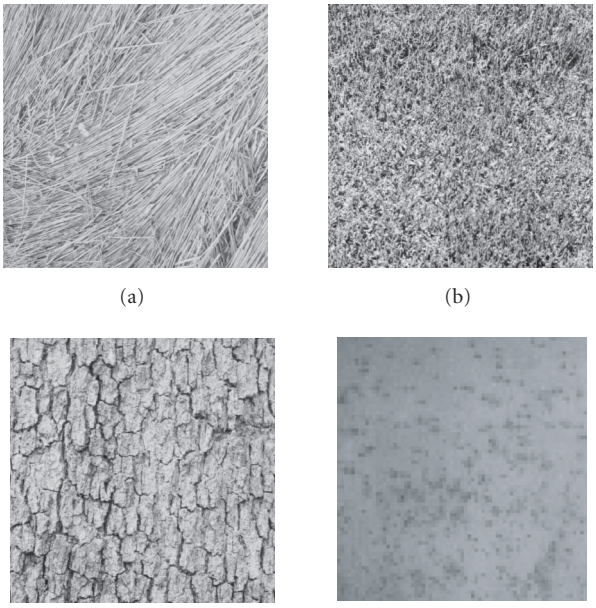
This chapter describes a statistical approach to classification of dynamic texture images, called parallel evolution functions (PEFs). Traditional classification methods predict texture class membership using comparisons with a finite set of predefined texture classes and identify the closest class. However, where texture images arise from a dynamic texture evolving over time, estimation of a time state in a continuous evolutionary process is required instead. The PEF approach does this using regression modeling techniques to predict time state. It is a flexible approach which may be based on any suitable image features. Many textures are well suited to a morphological analysis and the PEF approach uses image texture features derived from a granulometric analysis of the image.

The method is illustrated using both simulated images of Boolean processes and real images of corrosion. The PEF approach has particular advantages for training sets containing limited numbers of observations, which is the case in many real-world industrial inspection scenarios and for which other methods can fail or perform badly.

## 9.2. Textures and texture analysis

It is natural to describe textures qualitatively, in terms of measures such as coarseness, smoothness, granulation, or randomness. However, texture analysis is concerned with the extraction and use of quantitative descriptors of texture.

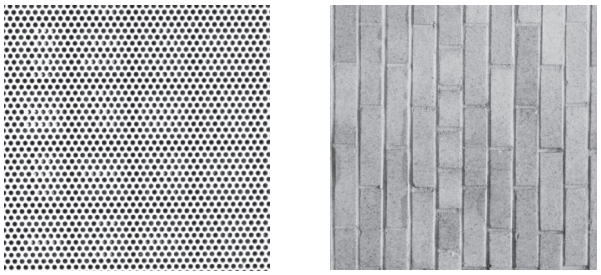
There are two main categories of textural feature [36], namely *statistical* and *structural* (and some which overlap both categories). Statistical textures include grass, canvas, cork, sand, marble, and corrosion. In statistical textures the location of the texture primitive(s) (i.e., meaningful regions) is random and irregular (Figure 9.1). Structural textures consist of repetitions of a basic texture element(s) (texels) or primitive(s). The texels are repeated regularly or with some degree of freedom (Figure 9.2).



(a) (b)

(c) (d)

FIGURE 9.1. Some examples of statistical textures: (a)–(c) from Brodatz [6]; (d) is an image of corrosion.



(a) (b)

FIGURE 9.2. Some examples of structural textures from Brodatz [6].

There are three main areas of texture analysis, namely, classification of an unknown texture region to one of a set of texture classes, modeling and describing a given texture region, and segmentation (of distinct textures within an image). Texture classification has been used in many different applications. Some examples are medical image processing, for detecting cancerous cells [51]; automated inspection of machinery, for detection of corrosion [50] and segmentation of landscape features from satellite images [7, 87]. An ideal robust texture classifier should be invariant to changes in rotation, lighting, and translation, and be able to detect

both very fine and fairly large textures, as well as irregular textures, such as many natural textures, including corrosion (of which rust is a common example).

### 9.2.1. Approaches to texture analysis

We briefly review approaches to the analysis of texture images and the advantages of a morphological approach [59, 80]. There are three main approaches to texture analysis, namely, *statistical*, *structural*, and *model-based*. Structural methods analyze the texture primitives present in an image and characterize regular textures in terms of arrangements of these texture primitives. The texture primitives can be used as image features for classification purposes. We do not consider structural methods in more detail here, as our main application of interest is corrosion images, which are statistical textures.

#### 9.2.1.1. Statistical methods

Statistical methods represent the texture by feature variables obtained through measurement. Some of the most common statistical texture analysis techniques [82, 83] are mathematical morphology [19, 78, 79] (Section 9.2.3); spatial gray-level co-occurrence or dependence [37], [36, Chapter 9]; gray-level run-length distributions [31]; digital transforms [91]; and the autocorrelation function [18, 89].

Spatial gray-level co-occurrence (SGLC) matrices [34, 36] characterize texture by the relative frequency of co-occurrence of gray-levels of pairs of pixels separated by a given distance and orientation. These are often averaged or summed over different angles. An alternative approach to remove dependence on angle is to compute neighboring gray-level dependence matrices [41]. Using regions with certain properties as the primitives instead of pixels has also been suggested. Either the matrices themselves, or more commonly, features computed from them are used for classification.

SGLCs have been very widely used and provide a robust, powerful, and general method for texture analysis, segmentation, and classification [15, 31, 37, 87, 94]. For a wide class of images, SGLCs capture much of the texture data, such as degree of coarseness and directionality. Vickers and Modestino [94] achieved 95% accurate classification of various texture types. However, co-occurrence matrices do not contain any shape or size information about texture grains. Some texture features derived from the SGLC [37], including angular second-order moment, entropy and correlation measures, do have the advantage of invariance to monotonic gray-level transformation. Gray-level difference vectors [65, 91] are used similarly to SGLC matrices, in that they compute relative frequency of absolute difference in gray-level between pairs of pixels at a given distance for a given orientation, and similar summary measures are used as texture features. Gray-level run-length distributions (GLRDs) represent frequency of runs of consecutive pixels with a given gray-level in a specified direction [91]. Coarse textures will have a high percentage of longer runs, while fine textures will have more short runs. Galloway [31] achieved 100% classification accuracy using GLRDs for aerial terrain images.

Tang [100] found that GLRDs (98–100% accurate) were slightly better than SGLCs on Brodatz [6] and VisTex images (MIT Media Library) [95].

An alternative approach is to view texture in the frequency domain. The Fourier spectrum [91] is found from the discrete Fourier transform of the image. Texture coarseness and directionality are indicated by peaks in 1D summations of the spectrum or the power spectrum, expressed in polar co-ordinates [4]. Very large images are required to obtain stable values, and local properties cannot be represented, therefore this is unsuitable for texture segmentation or detection of small local areas of texture. Wavelets (Section 9.2.1.2), in contrast, represent both spatial and frequency information. Also, the Fourier transform is not invariant to monotonic gray-level transformations, in common with most spatial frequency approaches [39, 96]. Conners and Harlow [15] found the SGLC to perform better than digital transforms. The autocorrelation function (ACF) is the Fourier transform of the power spectrum (spectral density function) [36]. It compares an image with a translated version of itself, thus detecting periodicities in the original image as peaks in the ACF [89]. The ACF is robust to noise and has been widely used in feature recognition. For example, Kurita and Otsu [49] found that it classified Brodatz textures with over 90% accuracy.

#### 9.2.1.2. Texture models

Textures can be synthesized from a texture model [43, 52, 99], or the parameters of a texture model can be used to characterize an observed texture and enable segmentation [74] or classification. Modeling approaches include random fields [46, 102], especially the Markov random field (MRF) [46], and the autoregression (AR) model [9], as well as wavelets [39, 102], and filter-based representations [66, 70, 71], including Gabor filters [42, 44].

MRFs and AR models both generalize the time-series autoregressive moving average models [102] to 2D. Image models are verified as suitable to represent the textures of interest using the training set of texture classes. The model parameter values are chosen to give the best model fit to each observed texture. Texture classification then compares model parameters for an observed image with those for a given texture class. The AR model [91] specifies pixel gray-level as a linear combination of the gray-levels in a neighborhood of specified dimensions. The coefficients of this linear combination characterize the texture. Performance depends on neighborhood size, which can be chosen to minimize prediction error.

Random field models represent the texture images as realizations of a random field with a probability distribution concentrated over the most likely images. The joint probability distribution for the image is specified through the conditional probabilities of pixel gray-level (or texture type) for a given pixel given the rest of the image [18]. These are assumed to depend only on gray-levels (texture types) of neighboring pixels (the Markov property). A suitable form must be specified for the neighborhood and nature of the probabilities. Commonly the Gibbs' random field model is used, in which the joint probability has an exponential form, or the Gauss-Markov random field [12, 48]. However, parameter estimation to fit

the probabilities to the images can be difficult. Such models also typically include only nearest neighbor interactions and do not realistically capture higher-order structural information in the images. Compared to other techniques, the MRF can achieve very high classification accuracy (100% for many textures) but for a few textures its performance can be very poor [1].

Wavelet transforms and Gabor filters both provide multiresolution spatial frequency decompositions of an image. Wavelets [55, 56, 63, 99] are a set of orthogonal basis functions which provide a complete representation of the image at different scales. The coefficients of the basis functions provide the multiresolution texture features. If the number of coefficients is very big, the dimension of the feature vector is decreased. There are many choices of basis function. The Fourier series expansion is one example, using sine and cosine curves. Wavelets are popular but do not always perform well for texture analysis [68, 93, 100]. Results in [93] suggest that the discrete wavelet approach (99% accurate overall) should perform better than most traditional single resolution approaches such as SGLC.

Gabor filters [30, 38, 42] are also widely used. The Gabor functions are used in a similar way to wavelets, as basis sets, though their non-orthogonality makes computation of the coefficients non-trivial. The image is represented as a linear superposition of Gaussian modulated complex sinusoids (whereas the Fourier representation [75] uses only sinusoids). Various studies have compared the Gabor filter with other texture classifiers and segmentation schemes, mainly MRF [38] variations and wavelet methods [67]. Haley and Manjunath [38] found that the Gabor filter outperformed MRF methods, while Manjunath and Ma [57] found that the Gabor filter and multiresolution simultaneous AR models outperformed wavelets. However it did not perform well in a comparison of filtering methods by Randen and Husoy [71]. Multichannel filters based on the Gabor filter, but with much lower computational cost, can perform almost as well [70]. Gabor filtering has various desirable properties over other multiresolution methods [58, 102] and the advantage of strong parallels with the low-level visual processing of humans and other mammals.

Finally, texture analysis of color images [27] is a higher dimensional problem than for gray-level textures. Transforming an image from RGB space to a different color space, such as LUV space [81], can be useful for producing a more perceptually consistent image segmentation [61] when combined with morphological operators. Köppen et al. [47] proposed a generalization of gray-scale morphology to color images. Monadjemi et al. [62] proposed Walsh transforms for color texture classification, while Chindaro et al. [12] used color space fusion for classifier combination.

### 9.2.2. Evolving textures

For evolving textures, such as what occurs in a corrosion process where a few small "spots" gradually increase in size and number until they cover the surface of the material (see Figure 9.8), or in diseased tissues imaged over time, the classification problem is one of relating a given sample of texture to a specific time within a

texture evolution process rather than to one of a series of unconnected texture classes.

Most conventional texture classifiers compare characteristics of the texture with those of a known class and place it to the nearest class, according to some measure of closeness, for example, the maximum likelihood (ML) or Euclidean distance criteria. For example, Liu et al. [51] use a quadratic discriminant function; Kyvelidis et al. [50] use fuzzy logic to classify defects (the results are used on a series of corrosion images to examine the growth process); García-Sevilla and Petrou [32] use the nearest-neighbor rule with various distance measures; Clausi [13] uses a 2-class Fisher linear discriminant rule to project an $n$-dimensional class feature set to a one-dimensional vector providing an optimal separation of 2 classes, and a series of ML classifications is then used to assign an observation to the class with the highest number of hits; and Rajesh et al. [69] use unsupervised classification based on the ISODATA algorithm [28], which minimizes a least-squares objective function.

The PEF method differs in that, instead of using a series of comparisons with predefined classes, it computes a set of parameters directly related to the class or state of evolution on a continuous scale, from the statistical texture characteristics. The removal of the direct comparison stage enables the classification of images that are not from any of the training classes, but are from classes intermediate to these, as long as the growth pattern is sufficiently well represented by the training examples. More importantly, the use of all instances of the texture over time to train the classifier means that for smaller training sets there is more information available for training. Chantler and Stoner [8] have shown that the use of combined sequences of data measurements can greatly decrease the error rate of feature classification over methods using each measurement in isolation.

### 9.2.3. Advantages of morphological analysis

Series of evolving texture images often consist of images which initially contain very little detail, then the texture builds up through time by the placement of more and/or larger objects in the image. This is the case with images of corroding metal, for example, considered in Section 9.5, in which the metal begins as a smooth surface, then becomes damaged and a small seed or grain of corrosion appears. Further grains may appear at other positions in the image and these grains increase and enlarge over time to cover the image plane. Alternatively an evolving texture may coarsen over time, as in images of soil becoming wetter. In either case such images are well suited to a morphological analysis, which extracts size and shape information from texture elements present in the image at a given time.

Mathematical morphology [59, 80] is a methodology for shape-based nonlinear image filtering using set operations. It has a very broad spectrum of applications and can be applied to localized textures, pixel by pixel [25, 26], particularly where a texture is composed of well-defined grains or where variation in the grains is important. This approach allows for good discrimination between

the texture of interest and any background clutter which does not share similar shape properties. In particular, although the granulometry (Section 9.3) is a statistical method, it can be related to the 3D physical properties of a material and its surface texture. While covariance-based methods, such as autoregression, difference statistics, Fourier transforms and SGLC, only describe the texture process up to second-order characteristics, morphological methods can capture higher-order properties of the spatial random processes [3] and no other approach directly uses shape information in the image. Morphological tools can also be directly translated to Boolean algebra, and therefore are easily implemented in hardware for efficient integration into digital systems for faster processing [2].

## 9.3. Granulometries

Granulometries are well-known morphological tools [19, 78] which consist of an iterative sequence of morphological operations. The simplest granulometry consists of a series of structural openings [19, 59], which we use here.

A morphological opening, *SoB*, of an image *S* by a structuring element (SE) *B*, is the image consisting of the union of every translation $x$ of $B$ that is totally contained within the image set in $S$ (or which fits beneath the image surface in the gray-level case), giving a coarse representation of the original image:

$$SoB = \cup\{B + x : B + x \subset S\}. \tag{9.1}$$

For gray-scale images, opening with a 2D SE is faster than using a 3D SE, however the latter can also extract information on the gradient and contours of the surface, as well as surface-shape information.

An opening-granulometry uses a series of SEs of increasing size $r$, that remove all image particles within which the SE cannot fit. Grains of smaller size are removed from the image by small-scale SEs and larger grains are removed sequentially as the SE is increased in size until no image content remains. The volume $\Omega(r)$ of the opened image is recorded at each step, that is the sum of all pixel intensity values in the image (or a sum of remaining pixels in the binary case). These volumes are then normalized using the initial volume $\Omega(0)$, to define a cumulative probability distribution function, the *size distribution*,

$$\phi(r) = 1 - \frac{\Omega(r)}{\Omega(0)}, \quad r = 0, \ldots, R, \tag{9.2}$$

which increases monotonically from 0 to 1 as $r$ increases. This is the proportion of the volume removed up to and including step $r$. The differences between the normalized values define the *pattern spectrum* of the image (Figure 9.3):

$$\Phi(r) = \phi(r) - \phi(r-1), \quad r = 1, \ldots, R, \tag{9.3}$$

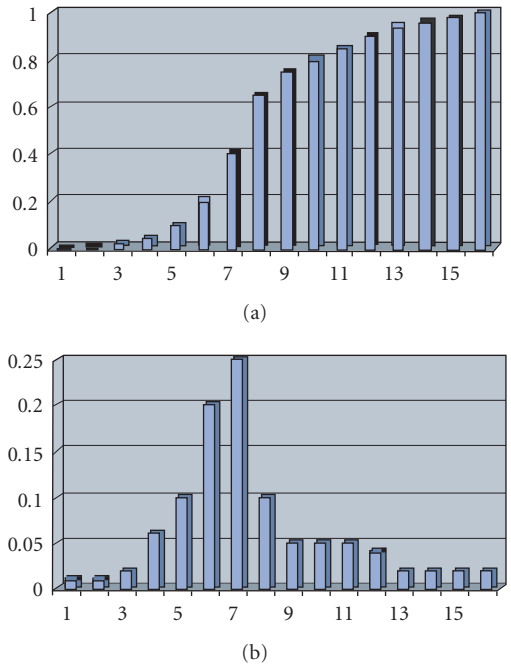



FIGURE 9.3. A size distribution (a) and its corresponding pattern spectrum (b).

that is, the proportion removed at stage $r$, which approximates the derivative of $\phi(r)$ and is a probability mass function.

The pattern spectrum profile captures the size and shape information from the foreground texture detail at differing scales. The more "peaked" the profile, the more closely the shape of the SE fits the texture structure, as sharp drops indicate a large number of grains of a given size and similar shape to the SE. Smooth graphs indicate a more even spread of sizes of grains, or grains which are not similar in shape to that of the SE. Figures 9.4 and 9.5 show examples for a simple gray-scale case and a real texture image. In Figure 9.4, the image contains 4 squares of different sizes and an ellipse, and the SE is square. The pattern spectrum contains several separate spikes, one for each of the 4 squares (which drop out one after the other as complete objects, as they are the same shape as the SE), and several contributed to by the ellipse (which differs in shape from the SE, and so is gradually eroded to become more rectangular in shape before disappearing completely). Figure 9.5 shows a 3D (gray-scale) image and the SE used was a 2D square. There is no clear separation of the grains by the background, and the use of an SE with dissimilar structural shape to that of the grains in the image results in a pattern spectrum which consists of a slowly decreasing curve, rather than a series of spikes corresponding to complete objects disappearing.

As the pattern spectrum is a probability mass function, its statistical moments can be calculated and used as texture feature descriptors [24, 53, 59]. These are

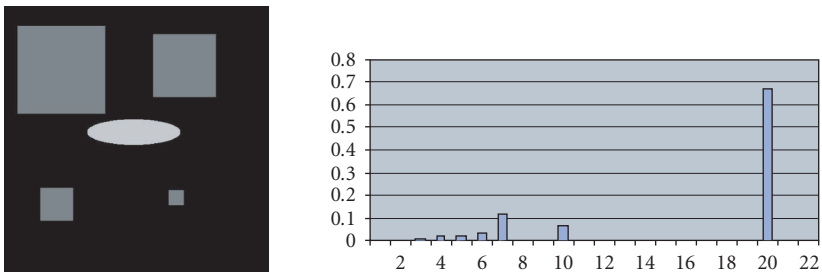



FIGURE 9.4. (a) Image consisting of differently shaped grains, and (b) its pattern spectrum obtained using a square SE; the horizontal axis represents size $r$ and the vertical axis $s$ represents the proportion of image volume removed.

referred to as the *granulometric moments* [19]. The $i$th order moment is

$$M_i = E(r^i) = \int_r r\Phi(r)dr \approx \sum r^i \Phi(r), \tag{9.4}$$

in which $r$ is a measure of size of the SE.

More general granulometries that do not necessarily use an opening can be defined [21]. In each case there is a parameterized set of operators $\{\psi_r\}$ and the volume of the image operated upon by $\{\psi_r\}$ is computed to form the size distribution. In our application $\{\psi_r\}$ is an opening by a single SE $rB$, where $B$ is a fixed primitive (the *generator* of the granulometry) and $r$ is a scaling factor. Another frequently used granulometry takes $\{\psi_r\}$ as a union of openings from different SEs, $rB_1, rB_2, \ldots, rB_m$ [5].

Use of the Euclidean granulometry [23] ensures that the granulometry acts strictly as a sieve, in which grains or image areas are removed depending only on their size and shape relative to the SE(s). The Euclidean granulometry is a family of operations $\{\psi_r(S) = SorB\}$ which is *anti-extensive* (i.e., $\psi_r(S) \subseteq S$); *increasing* ($A \subset B \implies \psi_r(A) \subset \psi_r(B)$, i.e., the sequence of processed images has the same order as the original images); *fulfilling the mesh property* ($\psi_r\psi_q = \psi_q\psi_r = \psi_{\max(r,q)}$, $r, q > 0$, i.e., use of several SEs is equivalent to using only the largest); *scale invariant* ($\psi_r(S) = r\psi_1(S/r)$, i.e., scaling $S$ by $1/r$, sieving by $\psi_1$, then rescaling by $r$ is the same as sieving $S$ by $\psi_r$); and *translation invariant* ($\psi_r(S_x) = (\psi_r(S))_x$, where $S_x$ is the image $S$ translated by $x$) [20]. These five properties limit the type of operators and SEs that can be used [20, 23, 79]. In particular, the SE used for opening must be convex.

Note that opening of the image foreground (the objects) and the background gives complementary information. Opening the foreground provides information on size and shape, while opening the background provides information on spatial distribution of the objects in the image. Use of the pattern spectrum moments from both foreground and background is therefore recommended.

It has been shown [76, 85] that the granulometric moments for an image containing disjoint grains are asymptotically normally distributed with increasing

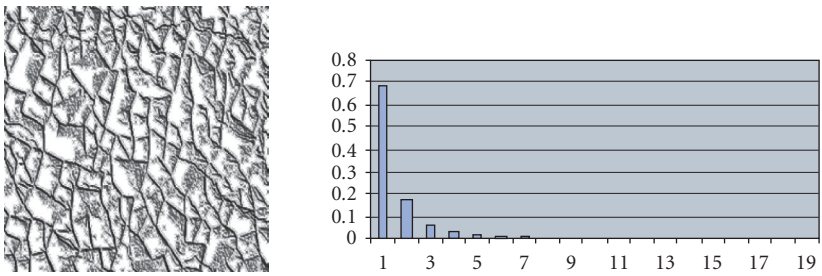



FIGURE 9.5. (a) A texture image of dried paint [6], and (b) its pattern spectrum from a square SE. (Reproduced from Proceedings of VIE 2005, International Conference on Visual Information Engineering, 4–6 April 2005, Glasgow, UK, ISBN 0-86341-507-5, A. J. Gray, J. Mckenzie, and S. Marshall, "Texture classification of grey scale corrosion images," pages 219–225, © 2005, with permission from the Publishing Department of the Institution of Electrical Engineers, London, UK.)

numbers of grains. For a limited amount of grain overlap, the distribution can be assumed to be approximately asymptotically normal. This asymptotic normality is one reason why the granulometry has proven to work well for texture classification in conjunction with a Gaussian ML classifier [76, 77]. We use this as a benchmark for the PEF approach.

The theoretical properties of granulometries and their properties in local texture classification are well researched. They have achieved classification accuracy in the high 90% range using granulometric features in a Gaussian ML classifier [11, 25] on Brodatz textures [6], even in the presence of noise and visually similar textures. Other morphological tools have also been used, with a fuzzy hit-or-miss texture transform achieving 96.9% accuracy for Brodatz textures in [101]. The granulometric approach has proved successful in varied applications, including blood cell analysis and chromosome overlap detection [21]; detecting changes in trabecular bone with an ML classifier [10]; analysis of electrophotographic (toner particle) images [24]; segmentation of digital mammograms [3], and counting white blood cells [90].

Ayala and Domingo [1] proposed new descriptors for binary and gray-scale images based on new spatial-size distributions, that generalize the usual granulometric size distribution. These combine a granulometric analysis of the image with a comparison between the geometric covariograms for binary images, or, for gray-scale images, the ACF (autocorrelation function) of the original image and its granulometric transformation. The new descriptor was better where a finer description of the image was required. The ACF on its own did not perform as well as using granulometry. Ayala and Domingo [1] compared their new method using a Bayesian classifier with some of the most common existing methods, including an opening-granulometry performed only on the foreground (thereby losing shape distribution information, which the new method included). There was no undisputed best classifier, however overall they concluded that the granulometry with added spatial data was at least as good as any of the other methods

(MRFs, SGLCs, fractal dimensions [16, 45, 92] and Gabor filters). Performance increased dramatically when spatial distribution data was included with the foreground granulometric moments. Even a foreground granulometry performed at least as well overall as a simple SGLC, for example.

Sivakumar and Goutsias [84] developed morphologically constrained Gibbs' random fields and applied them to texture synthesis and analysis. It was shown that under certain conditions the ML estimators of the morphologically constrained GRF parameters could be approximated using the granulometric pattern spectrum, providing a much easier and faster implementation of GRFs. A computationally simple morphological Bayes' classifier then achieved 98% correct classification on Brodatz textures.

### 9.4. Parallel evolution functions

The parallel evolution function (PEF) approach [53] to time classification of evolving textures is now described. It may be used with any calculable descriptors that characterize the image at different times. Here it makes use of the granulometric pattern spectrum moments as image features. These are calculated, as described above, from a granulometric opening of the texture image.

The PEF approach is a statistical method which places an image from an evolving process directly to a position on the evolution time scale of the process, rather than comparing the observed features with all available time classes and allocating to the closest class. It relates the evolution parameters of the model assumed to generate the texture images to the observable granulometric moments over a series of time steps, using multiple linear regression (MLR) of the moments, as the dependent variable, on the evolution parameters, as the explanatory variables. (Note that the moments are *random* functions of the *fixed* parameters of the evolution process). Using a training set of simulated images for which the underlying evolution parameters are known, a multiple regression model is fitted for each chosen moment, giving an "evolution function" relating the (mean over the available observations at each time step of the) moments of the evolving textures to the parameters controlling the evolution. The result is a set of these evolution functions; one for each chosen moment, hence the term "parallel." Once the PEFs are fitted, for any new image, for which the time state is unknown, the granulometric moments of the image are calculated. Each of the PEFs could be used singly to predict the evolution parameters and then the time of evolution, however each of the PEFs involves the same parameters. Therefore the PEFs are combined as described below.

At each of $n+1$ times $t = 0, \ldots, n$, there are $l$ images, giving $l \times (n+1)$ moments of a given order $i$, $i = 1, \ldots, m$. Let $M_i(t)$ denote the training sample mean (averaged over the $l$ images) of the $i$th moment at time $t$ for times $t = 0, 1, \ldots, n$, giving an $(n+1)$-vector of moments of order $i$, $\mathbf{M_i} = (M_i(0), M_i(1), \ldots, M_i(n))^T$. $\mathbf{P_j}$ is the $(n+1)$-vector $\mathbf{P_j} = (P_j(0), P_j(1), \ldots, P_j(n))^T$, where $P_j(t)$ is the $j$th evolution parameter, $j = 0, \ldots, J$, at time $t$ (used as an explanatory regressor variable).

The least-squares MLR model is given by

$$\mathbf{M_i} = \sum_{j=0}^{J} b_{ij}\mathbf{P_j} + r_i, \quad i = 1,\ldots,m, \tag{9.5}$$

that is,

$$\begin{pmatrix} M_i(0) \\ M_i(1) \\ \vdots \\ M_i(n) \end{pmatrix} = \begin{pmatrix} P_0(0) & P_1(0) & \cdots & P_J(0) \\ P_0(1) & P_1(1) & \cdots & P_J(1) \\ \vdots & \vdots & \ddots & \vdots \\ P_0(n) & P_1(n) & \cdots & P_J(n) \end{pmatrix} \begin{pmatrix} b_{i0} \\ b_{i1} \\ \vdots \\ b_{iJ} \end{pmatrix} + \begin{pmatrix} r_i(0) \\ r_i(1) \\ \vdots \\ r_i(n) \end{pmatrix}, \tag{9.6}$$

where the $(n+1)$-vector $\mathbf{P_0} = (1,1,\ldots,1)^T$, and $\mathbf{r_i}$ is an $(n+1)\times 1$ column vector of random errors or disturbance terms representing the deviations of the $i$th order moments in the vector $\mathbf{M_i}$ from the expected moments given by $\sum_{j=0}^{J} b_{ij}\mathbf{P_j}$. The coefficient vector $\mathbf{b_i} = (b_{i0}, b_{i1},\ldots,b_{iJ})^T$ is estimated by minimizing the error sum of squares [60]:

$$\sum_{t=0}^{n}\left[M_i(t) - \sum_{j=0}^{J} b_{ij}P_j(t)\right]^2, \tag{9.7}$$

giving

$$\hat{\mathbf{b}_\mathbf{i}} = \left(\mathbf{P}^T\mathbf{P}\right)^{-1}\mathbf{P}^T\mathbf{M_i}. \tag{9.8}$$

We have such a regression model $\mathbf{M_i} = \mathbf{P}\mathbf{b_i} + \mathbf{r_i}$ for each moment used ($i = 1,\ldots,m$). The goal is to estimate the evolution parameters from the moments, and having acquired these parameter estimates, to use them to derive the time status of the image. The same evolution parameters are involved in each regression equation, which we use as follows. Let $\mathbf{M}$, $\mathbf{B}$, and $\mathbf{P}$ be defined as

$$\mathbf{M} = \begin{pmatrix} M_1(0) & M_2(0) & \cdots & M_m(0) \\ M_1(1) & M_2(1) & \cdots & M_m(1) \\ \vdots & \vdots & \ddots & \vdots \\ M_1(n) & M_2(n) & \cdots & M_m(n) \end{pmatrix} = (\mathbf{M_1} \mid \mathbf{M_2} \mid \cdots \mid \mathbf{M_m}),$$

$$\mathbf{B} = \begin{pmatrix} b_{10} & b_{20} & \cdots & b_{m0} \\ b_{11} & b_{21} & \cdots & b_{m1} \\ \vdots & \vdots & \ddots & \vdots \\ b_{1J} & b_{2J} & \cdots & b_{mJ} \end{pmatrix} = (\mathbf{b_1} \mid \mathbf{b_2} \mid \cdots \mid \mathbf{b_m}),$$

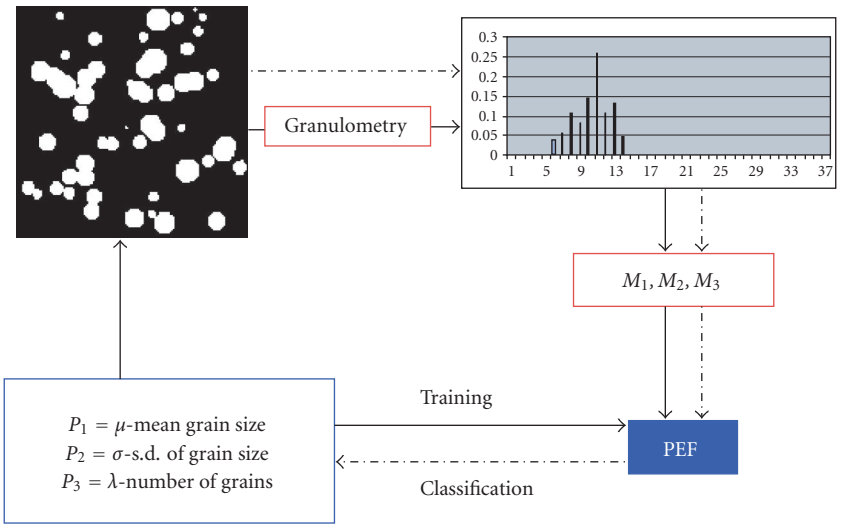



FIGURE 9.6. The training and classification process for the PEF approach, using parameters $P_1$–$P_3$ and moments $M_1$–$M_3$; solid and dotted lines indicate training and classification steps, respectively.

$$\mathbf{P} = \begin{pmatrix} P_0(0) & P_1(0) & \cdots & P_J(0) \\ P_0(1) & P_1(1) & \cdots & P_J(1) \\ \vdots & \vdots & \ddots & \vdots \\ P_0(n) & P_1(n) & \cdots & P_J(n) \end{pmatrix} = (\mathbf{P_0} \mid \mathbf{P_1} \mid \cdots \mid \mathbf{P_J}). \tag{9.9}$$

Then the MLR model can be summarized by $\mathbf{M} = \mathbf{PB} + \mathbf{R}$, where $\mathbf{R} = (\mathbf{r}_1 \mid \mathbf{r}_2 \mid \cdots \mid \mathbf{r_m})$. Then $\widehat{\mathbf{M}} = \mathbf{P}\widehat{\mathbf{B}}$ where $\widehat{\mathbf{B}} = (\widehat{\mathbf{b}_1} \mid \widehat{\mathbf{b}_2} \mid \cdots \mid \widehat{\mathbf{b_m}})$ contains all the least-squares estimates of the coefficients from the regressions for each of the moments ($i = 1, \ldots, m$) separately.

Given a new image, the evolution parameters $\mathbf{P}$ are then estimated from the observed moments as $\widehat{\mathbf{P}} = \mathbf{M}\widehat{\mathbf{B}}^{+}$ where $\widehat{\mathbf{B}}^{+}$ is the pseudo-inverse of $\widehat{\mathbf{B}}$ and where $\widehat{\mathbf{B}}^{+} = \widehat{\mathbf{B}}^{-1}$ if $\mathbf{B}$ is non-singular [18]. The assumed growth model equations relating evolution parameters to time are used to convert estimated evolution parameters to estimated evolution time (Section 9.6.1). The process is illustrated in Figure 9.6.

### 9.4.1. Model fitting issues

Use of ordinary least-squares (OLS) regression, as described, assumes that in each of the $m$ regression models, the errors $r_i(0), \ldots, r_i(n)$ are uncorrelated and have the same variance $\tau_i^2$, $i = 1, \ldots, m$, and the optimal estimators of the regression coefficients are given as in (9.8). If these assumptions are invalid, these estimated coefficients will be suboptimal. Weighted least-squares regression allows for differing

error variances within a given regression model, while generalized least-squares (GLS) regression allows for correlated errors also, as are likely to be present in data collected through time. GLS assumes that $\mathrm{cov}(\mathbf{r_i}) = \tau_i^2 \mathbf{W_i}$, $i = 1, \ldots, m$, where $\mathbf{W_i}$ is an arbitrary positive definite $(n+1) \times (n+1)$ variance-covariance matrix, rather than $\mathrm{cov}(\mathbf{r_i}) = \tau_i^2 \mathbf{I_{n+1}}$, a constant multiple of the $(n+1) \times (n+1)$ identity matrix, and the minimum variance linear unbiased estimators of the regression coefficients are then [72, 98]

$$\widehat{\mathbf{b_i}} = \left(\mathbf{P}^T \mathbf{W_i}^{-1} \mathbf{P}\right)^{-1} \mathbf{P}^T \mathbf{W_i}^{-1} \mathbf{M_i}. \tag{9.10}$$

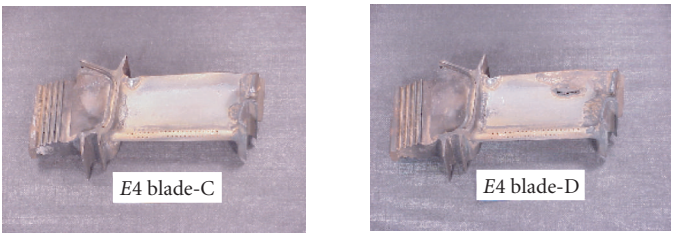
However, estimating the covariance matrix means that the estimated coefficients $\widehat{\mathbf{b_i}}$ no longer have minimum variance and may have greater variance than the OLS estimates. Reliable estimation of individual variances and covariances requires large amounts of data and unreliable estimation is likely to lead to estimates with poorer model fit than with OLS. For both the simulated and real images used below, the sample variances of the moments were in general not constant over time, however we use OLS regression modeling, owing to limited availability of data. This may adversely affect model fit.

It will be seen (Section 9.6.1) that in our model all evolution parameters increase with time, and hence there are likely to be collinear relationships between the explanatory variables [72]. The likelihood of the fitting of the regression coefficients becoming unstable (i.e., small changes in the data or explanatory variables leading to large changes in the estimated coefficients) due to collinearity was tested, using eigenanalysis [72] of the covariance matrix of the explanatory variables as well as altering the order of the chosen explanatory variables. Near-zero eigenvalues indicate linear dependencies between the explanatory variables. For both the simulated and real data, collinearity was found to be present, especially for the background data, however this did not appear to cause model instability. Nevertheless, centering the explanatory variables, or using principal components regression [72] may be helpful.

Assessing model fit using statistical tests requires that the moments (or residuals) in the regression model are normally distributed [72, 98]. For the simulated images, normal probability plots and histograms indicated that the distributions of the residuals were reasonably symmetric but heavy tailed. The plots for the background moments of the real images were closest to a normal distribution.

## 9.5. Application to corrosion images

Industrial inspection is a well-established field, essential to the safe and economical running of any business involved in manufacturing or requiring to maintain equipment. Component damage or defects often consists of cracks, discoloration, or variations in size or texture [24, 50, 81, 88]. Mechanical parts, such as aircraft engine components [73], showing signs of damage may be routinely replaced at regular intervals, whereas optimal replacement times may be different. Having the




(a) (b)

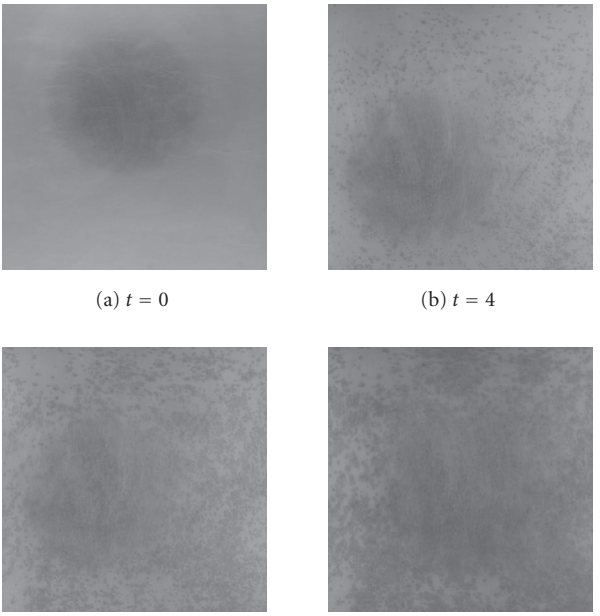
FIGURE 9.7. Turbine blades showing different degrees of corrosion: some damage visible on C and severe damage on D.

ability to relate degree of damage to weakening of a component is therefore important.

The motivation is to develop a method for automatic detection and classification of corrosion of metal parts according to its seriousness in terms of component weakening or loss of functionality. Figure 9.7 shows samples of turbine blades with varying degrees of corrosion. It would clearly be desirable to remove blades as corroded as that in D, before they reached this degree of damage.

Figure 9.8 shows 4 images of a mild steel plate, sprayed regularly with a dilute saline solution over a period of 9 days, as it corrodes over time. The resulting series of 10 images (at times $t = 0$–9 days) shows the evolution, from no corrosion to almost complete oxidation of the surface of the plate. The plate changed from gray to brown during the experiment, however there was insufficient variation in color for color itself to be useful for prediction of time state. The 10 images of $1400 \times 1400$ resolution were converted to gray-scale (as shown) for gray-scale granulometry to be used. The blob-like textures observed in these images are well suited to a morphological analysis, as provided by granulometry.

Corrosion appears as an irregular texture, which evolves according to an underlying chemical process controlled by many factors, such as environmental humidity and acidity, initial surface roughness, and type of material. The detection of corrosion is seldom easy, as any image of a component will contain shading and shapes that are not the result of corrosion but which may look similar to the effects of corrosion. The PEF method assumes that the evolution of the underlying chemical process results in a parallel evolution of the visual texture, and approximates this relationship in order to classify an observed texture to a precise point on the evolution scale. The texture process is modeled as a time-dependent stochastic spatial point pattern described by random variables whose value depends stochastically on parameters which are functions of time. These underlying parameters can be used to quantify component weakening, through mechanical testing or comparison with components that have been assessed by a human expert. However, these parameters describing the evolutionary state of corrosion cannot be observed directly from new corrosion images. The pattern spectrum moments of the corrosion image are therefore used as a means of estimating the underlying hidden parameters. These estimates can then be used as a measure of the state of



(a) $t = 0$ (b) $t = 4$

(c) $t = 7$ (d) $t = 9$

FIGURE 9.8. Gray-scale corrosion images at times $t = 0$, 4, 7, and 9 days.

evolution or severity of corrosion, by finding the equivalent length of time taken, under set conditions, for the corrosion to form and therefore the degree of weakening caused (if this is known for the training data).

## 9.6. Modeling the texture

### 9.6.1. Image simulation

The methodology is illustrated first on binary, then gray-scale, simulated images. Simulation permits fitting of the regression models relating the underlying model parameters to the observable granulometric moments, as the values of the parameters are known for a given evolution time of the texture. The choice of model and parameters controlling the evolution was based on observations of corrosion forming under natural conditions. This appears as a few small spots, which grow slowly at first, then size, and then also the number of spots increase rapidly, until the entire surface is covered. Therefore the model assumes corrosion to be a spatial point pattern growing from randomly distributed seed points [64].

A Boolean point process model [17, 40, 78], a random process in which random shapes ("grains") are positioned according to an independent point process [35], is used to represent the image textures, although any suitable model can be used. Grains are scaled to random size and placed randomly (according to a Uniform distribution) in the image plane with grains simulated according to a Poisson

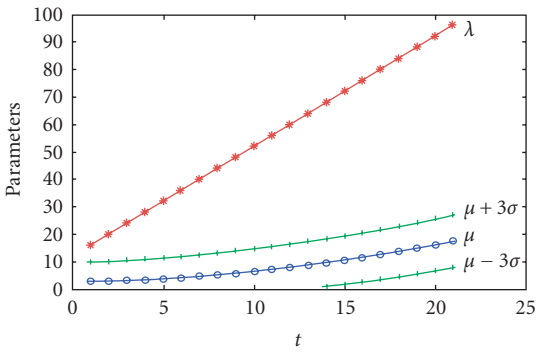


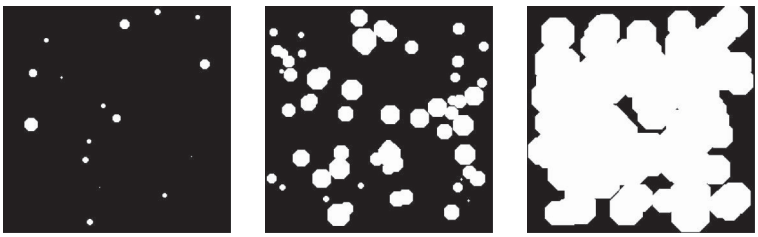

FIGURE 9.9. Assumed evolution path of the corrosion model parameters using $\mu(t) = 0.03t^2 + 0.133t + 2.61$, $\sigma(t) = 0.0416t + 2.349$, and $\lambda(t) = 4t + 16$. Note that some small grains are present at time 0.

point process with grain intensity $\lambda(t)$ (number of events per unit area) controlling the number of grains in the image. The Poisson process assumes that the grain positions are disjoint but grains may overlap. The values of the parameters of the model (the intensity and the grain size parameters), the "evolution parameters," are functions of the evolution time $t$.

The parameters were chosen empirically to give a series of realistic-looking textures, beginning with no coverage of the image background, increasing rapidly to almost complete coverage (at time $t = 20$). Texture growth is achieved by increasing grain intensity $\lambda(t)$ and mean grain size $\mu(t)$ with time. The variance of the grain dimensions also increases with time, to allow for new seed points forming once larger grains are already formed. An increase in the rate of corrosion with time can be explained as being due to the increased surface area exposed by surface roughening [97]. Grain size at time $t$ is taken as a folded Normal random variable (i.e., absolute grain size is used), with mean $\mu(t)$ and standard deviation (s.d.) $\sigma(t)$. The grain intensity (no. of grains) $\lambda(t)$ and size standard deviation $\sigma(t)$ are assumed to increase linearly, while mean grain size $\mu(t)$ increases quadratically with time, as illustrated in Figure 9.9. This results in grains that both grow and overlap to a greater extent as time increases.

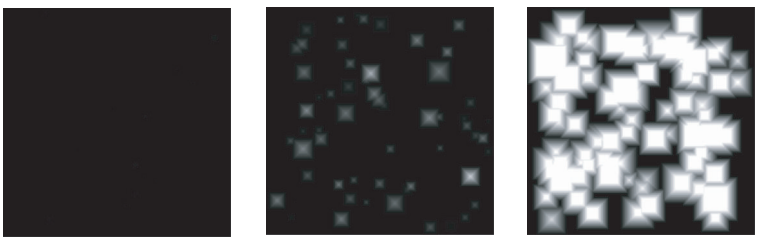
For a new image, once the parameter values are estimated from the moments, each of the equations relating the parameters to time is inverted to predict time $t$ for that image and these various predictions are averaged. Alternatively, a single equation may be chosen on the basis of minimizing classification error for the training images, giving a single predicted time.

Figures 9.10 and 9.11 show example textures using binary octagonal grains and square-based gray-scale pyramidal grains (clipped to a maximum gray-level of 255). These simulated images were generated using new seed points in every instance and not grown from one set over time. Both models use parameter evolution as in Figure 9.9. For the octagons (approximation to a disc) the grain size is the radius. For the pyramids, the mean base length is taken as $2\mu(t) + 1$, for



(a) $t = 0$ (b) $t = 11$ (c) $t = 20$

FIGURE 9.10. Binary octagonal textures at times $t = 0$, $t = 11$, and $t = 20$. (Reproduced from International Journal of Pattern Recognition and Artifical Intelligence, Vol. 17(2), J. Mckenzie, S. Marshall, A. J. Gray, and E. R. Dougherty, "Morphological texture analysis using the texture evolution function," pages 167–185, © 2003, with permission from World Scientific Publishing Co. Pte. Ltd, Singapore.)



(a) $t = 0$ (b) $t = 11$ (c) $t = 20$

FIGURE 9.11. Gray-scale pyramid images at times $t = 0$, $t = 11$, and $t = 20$. (Reproduced from International Journal of Pattern Recognition and Artifical Intelligence, Vol. 17(2), J. Mckenzie, S. Marshall, A. J. Gray, and E. R. Dougherty, "Morphological texture analysis using the texture evolution function," pages 167–185, © 2003, with permission from World Scientific Publishing Co. Pte. Ltd, Singapore.)

comparability with the octagons, and the mean step size is also set to $\mu(t)$, so that gradient and overall gray-level increase with time.

### 9.6.2. PEF modeling

We use the first three moments of the pattern spectrum, $M_i(t)$, $i = 1, 2, 3$, at each time $t$ (see Section 9.3), so there are three regression equations. We chose an MLR model depending on a subset of the first five powers of the Boolean model parameters, $\mu, \sigma, \lambda, \mu^2, \sigma^2, \lambda^2, \ldots, \mu^5, \sigma^5, \lambda^5$, which results in a set of 15 candidate explanatory regressor variables $p_1, p_2, \ldots, p_{15}$ (and 16 coefficients if an intercept is included). This was done to increase the chances of good model fit. The "best" three MLR regressor variables of these 15 candidates for each regression were chosen as those giving the lowest classification error on the training images. This is very time consuming, and a stepwise variable selection procedure may be more useful in practice [72]. Note that these were not necessarily those variables giving best fit of the regression models. It was generally found that the best results were obtained by restricting the number of regressor variables and using separate

regressions for the foreground and background moments. Three variables were used here so that **B** is a square matrix when modeling 3 moments, but more could be used.

The evolution time is estimated as the (nearest integer to the) mean of the estimated times from the foreground and background regressions.

### 9.6.3. Benchmark methods

The most commonly used classifier is the Gaussian maximum likelihood (ML) classifier [29], however (both for this and the optimal Bayes' classifier assuming multivariate normality of the feature variables), the number of observations needs to be larger than the number of variables used, for the covariance matrix of the variables to be non-singular [33]. This severely limits the usefulness of these methods in this context, given the very small number of observations often likely to be available in practice. The ML classifier was therefore used with the pooled covariance matrix from all time classes (the MLP method). In some cases MLP performed worse than ML using individual covariance matrices and in other cases it was better, but it always produces a result for 2 or more training observations. In our application, ML fails completely for fewer than 7 training observations, as it was used with 6 feature variables (i.e., the three pattern spectrum moments from each of the foreground and background images).

We use the ML and MLP classifiers and the minimum distance (MD) classifier [84] (which can be used regardless of sample size), as benchmark comparisons for the PEF. Unlike ML, MLP, and PEF, which use the pattern spectrum moments as feature variables, the MD method is applied to the discrete pattern spectrum itself (and all pattern spectra are filled out with zeroes to the size of the largest pattern spectrum observed, to make this possible). MD chooses class $k$ such that

$$k = \underset{t}{\operatorname{argmin}} \left\{ \sum_{r=0}^{R} \left[ \Phi_{\text{obs}}(r) - \Phi_t(r) \right]^2 \right\}, \tag{9.11}$$

where $r$ indexes the scale of the SE used in opening and $R$ is the largest such scale, and $\Phi_{\text{obs}}(r)$ and $\Phi_t(r)$ are the values of the pattern spectrum at scale $r$ for the observed image and the $t$th class, respectively. In practice we have a sum of squared distances, as in (9.11), for both the foreground and background pattern spectra and we classify using the total of these sums. See [84] for optimal properties of the MD classifier.

### 9.6.4. Results for simulated images

The results below use an octagonal SE for the opening of the binary octagonal grain images, and a pyramidal SE calculated using the fast algorithm of [86] for the gray-scale pyramid images. The gradient of the slope of the pyramid SE was decreased from very steep steps of 31 to steps of 2 (almost flat), with height being

Table 9.1. Type 0, type 1, type 2, and type 3 errors for binary octagonal textures and gray-scale pyramid textures, using 30 and 5 training images.

| Training set size | 30 images | | | | 5 images | | | |
|---|---|---|---|---|---|---|---|---|
| Error type | 0 | 1 | 2 | 3 | 0 | 1 | 2 | 3 |
| Images | Binary octagons | | | | | | | |
| Method | Error (%) | | | | | | | |
| MLP | 39 | 5.9 | 1.3 | 0.3 | 45.6 | 8.7 | 1.9 | 0.5 |
| ML | 38.7 | 3.5 | 0.6 | 0.2 | — | — | — | — |
| MD | 49 | 11 | 1.9 | 0.2 | 57 | 16 | 5.4 | 1.1 |
| PEF | 49 | 5.4 | 0.6 | 0 | 47.6 | 5.6 | 0.6 | 0 |
| Images | Gray-scale pyramids | | | | | | | |
| Method | Error (%) | | | | | | | |
| MLP | 42.3 | 16.9 | 7.8 | 2.6 | 46.1 | 17.1 | 8.6 | 3.6 |
| ML | 50.1 | 5.3 | 0.4 | 0.0 | — | — | — | — |
| MD | 61.0 | 23.0 | 9.4 | 4.4 | 66.3 | 26.5 | 11.4 | 5.3 |
| PEF | 60.4 | 11.4 | 2.1 | 0.4 | 60.0 | 11.4 | 2.3 | 0.4 |

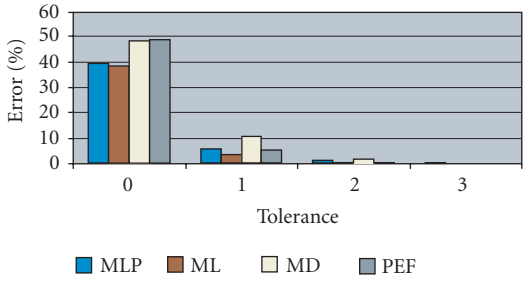
kept at approximately 255 by increasing base size as the gradient decreases. In this case, in calculation of the moments in (9.4), $r$ represents the gradient.

Performance is assesssed through several error measures, referred to as type 0, type 1, type 2, and type 3 errors. Type $i$ error is defined as

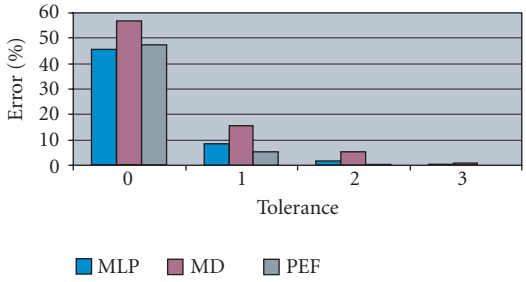
$$\frac{1}{n}\sum_{j=1}^{n} I_i(C_{\text{est}}^j - C_{\text{obs}}^j), \quad i = 0,\ldots,3, \tag{9.12}$$

where $j$ indexes the $n$ images to be classified, $C_{\text{est}}^j$ and $C_{\text{obs}}^j$ are, respectively, the estimated and actual class/state of image $j$, and $I_i(x) = 1$ if $|x| > i$, else $I_i(x) = 0$. Hence type $i$ error records the proportion of images not classified to within $i$ units of their actual time state. These error measures reflect the nature of the corrosion application. For applications where safety or damage limitation is critical, the maximum allowable chance of a component or system failing is set deliberately low to allow for uncertainty in safety status and ensure safe operation. It is more important always to predict the safety/damage status close to the correct class, than to be exactly correct most of the time but have the occasional large error. Where some overlap between classes is also possible, such as in the simulated images, for which the evolution parameters controlling the grain size and density are random variables, classification may not be exactly correct, but still effective for assessing evolution state. In assessing corrosion, deciding whether corrosion is too advanced will depend on being "close" to the time classes representing advanced corrosion in the training data. Again, allowing some tolerance before recording an error is sensible.

Table 9.1 and Figure 9.12 show results for the binary octagons, for training sets of size 30 and 5 images at each of times $t = 0,\ldots,20$. The smaller number of training images was chosen as in many real-world quality control situations very




(a) 30 training observations




(b) 5 training observations

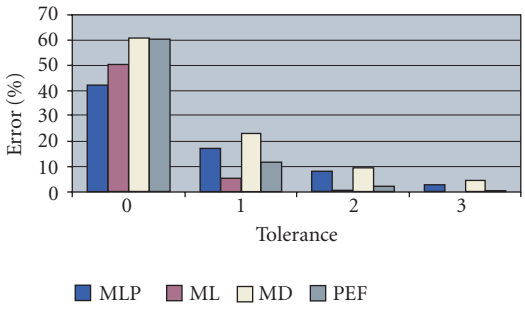
FIGURE 9.12. Percentage error versus error type (tolerance) for binary octagons.

limited training samples are available. The available data is often limited by difficulties in process monitoring, tracking of parts, record keeping, and various other factors. In both cases, an independent test set of 30 images was used at each time step to find the error rates, so that $n = 630$ in (9.12). The smaller sample size is too small for the unpooled ML method to be feasible. For the gray-scale pyramids, the errors have been averaged over 10 sets of training observations of a given size, for greater stability with smaller training set sizes. Table 9.1 and Figure 9.13 give the results.

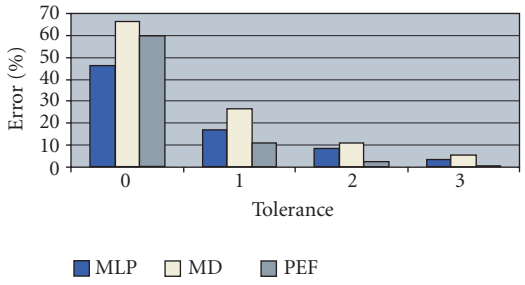
No method achieves a low type 0 error, and classifying exactly correctly is clearly difficult. For both octagons and pyramids, PEF is better than MD for all error types, especially for the smaller training set, and better than MLP for all except type 0 error. PEF is barely affected by a reduction in training set size, whereas MD and MLP both perform better for the larger training set.

For the octagons, and 30 training observations, ML is better than PEF for type 0 and 1 errors, but ML is unavailable for the smaller training set sizes, and PEF and ML are very similar for error types 2 and 3. For the pyramids, ML is always better than PEF for the larger training sample, but again cannot be used for smaller samples.

Figures 9.14 and 9.15 show type 0 and type 2 errors as a function of training set size. For type 0 error, for the octagons PEF is best for very small numbers of observations, beyond which MLP is best, whereas PEF is poorer than the other




(a) 30 training observations




(b) 5 training observations

FIGURE 9.13. Percentage error versus error type (tolerance) for gray-scale pyramids.

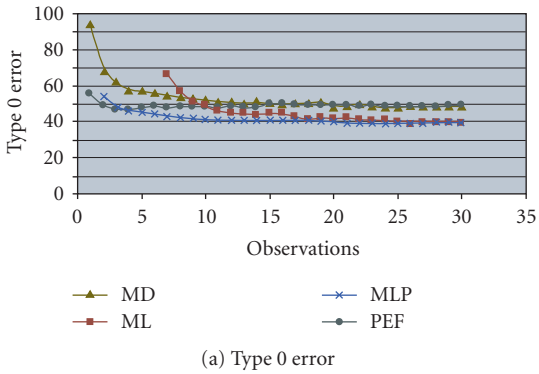
methods for the pyramids. Where some is allowed, PEF is clearly the best and most consistent choice regardless of sample size.

Further results, for triangular grains and mixed grain types, are given in [53, 54]. Robustness of the PEF, when trained on one set of textures and applied to textures in which only the grain shape differed but the generative model followed the same evolution process, was also investigated in [53, 54]. Achieving acceptable results depends on the relationships of the moments with time being similar for the two processes.

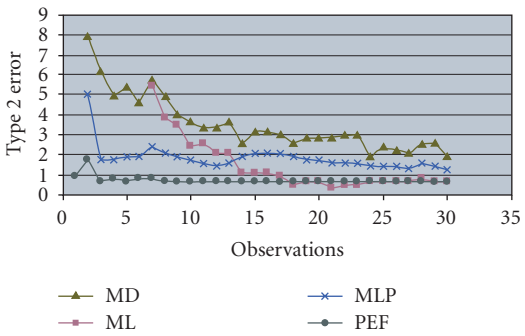
### 9.6.5. Results for corrosion images

The granulometric opening of the corrosion images here uses gray-scale pyramid structuring elements of increasing base size as 3D probing functions, applied as in Section 9.6.4.

For the simulated training images generated from a model, the evolution parameters used in the PEF approach are known. The model for the simulated images is used as an estimate of the model linking the parameters to time for the corrosion images. It is not necessary that the parameters relate to precise physical structures in the real images. They are instead "virtual" parameters which are used to encode time.




(a) Type 0 error

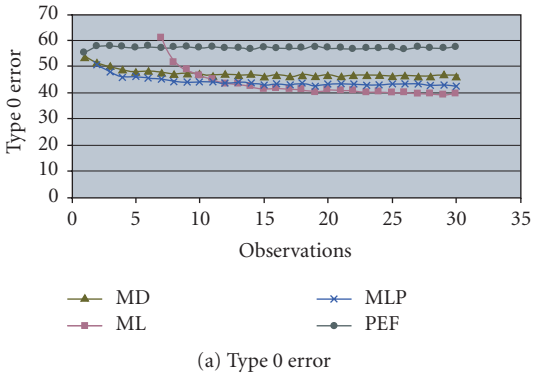



(b) Type 2 error

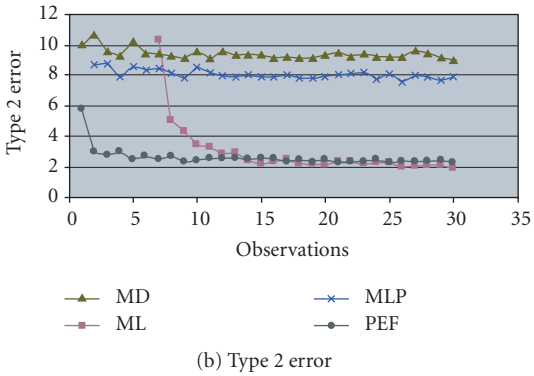
FIGURE 9.14. Percentage error versus training set size for binary octagons.

The ten $1400 \times 1400$ corrosion images (Figure 9.8) were each cut into 49 subimages of size $200 \times 200$. The first 7 such images (from the top row) for each time were used as the training data, reflecting the small training set sizes typically available in practice for such applications, and another 7 images from the right-hand side of the plate were considered as independent test data. The limited number, and the position, of the selected images owes to the reflection of the camera on the polished surface (seen in Figure 9.8) being harder to remove than the gradual light variations. However, there is substantial variation within the corrosion images at a given time $t$, and obtaining sufficiently similar training and test images was difficult. Figure 9.16 ilustrates this variation, showing a $200 \times 200$ training image taken from the top left of the original image at day 5, and a test image taken from the bottom right. In this application, clearly larger training images are desirable to encompass the variation found in these stochastic textures at a given point in time. The results shown are therefore for the dependent data.

Figures 9.17 and 9.18 show percentage errors for 1 to 7 training images. Given the small training set sizes, the ML method is not used and MLP is inapplicable




(a) Type 0 error

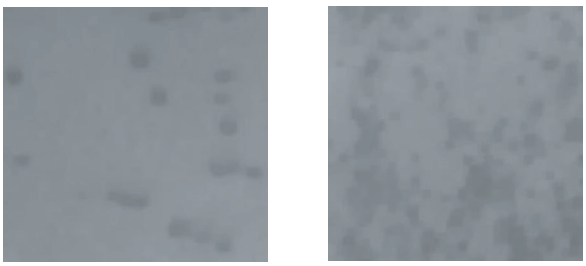



(b) Type 2 error

FIGURE 9.15. Percentage error versus training set size for gray-scale pyramids.

for only 1 training image. The error rates shown are averages. The training observations were chosen randomly from the available set of training images, and up to 10 such choices were made, depending on the training set size. Averaging the resulting error rates smooths out fluctuations caused by the particular choice of small training set. Results are shown for the combined PEF method, as previously, as well as for PEF applied to the image foreground (FRONT) and background (BACK) separately. The results are compared with those of MD and MLP.

For type 0 and type 1 errors, MD is best for 1 or 2 observations. For type 0 error, MLP is clearly best for more than 2 observations and PEF, FRONT, and BACK methods are poorer. For type 1 errors, MLP and PEF are equally good choices for more than 2 observations. For type 2 and type 3 errors, MD is best only for 1 observation, and beyond that PEF and FRONT are generally best. BACK does not perform as well, and of the combined PEF, FRONT, and BACK, the combined PEF is the most consistent of the three methods. These results are not quite as good as for the simulated images, however PEF is still the best method overall when a small tolerance is allowed, where there is more than 1 training observation. MD and MLP are more accurate with no tolerance.



(a) (b)

FIGURE 9.16. Reproduced from Training image (a) and independent image (b) from day 5. (Proceedings of VIE 2005, International Conference on Visual Information Engineering, 4–6 April 2005, Glasgow, UK, ISBN 0-86341-507-5, A. J. Gray, J. Mckenzie, and S. Marshall, "Texture classification of grey scale corrosion images," pages 219–225, © 2005, with permission from the Publishing Department of the Institution of Electrical Engineers, London, UK.)

PEF is more sensitive to changes in texture than MLP and MD, which is generally an advantage, although it does require that the moments and evolution functions are similar for the training data and the process generating the textures to be classified. The poorer performance of PEF for the real data with only 1 or 2 training observations is a result of the larger variation in the moments within a given time state requiring more data to fit the regression model reliably.

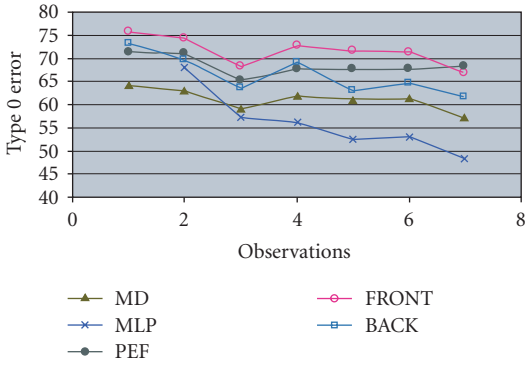
## 9.7. Summary

Morphological methods have the advantage, over other approaches to texture analysis, of directly evaluating shape and size information in the texture image. Granulometric analysis applies morphological operators in sequence at a series of different scales. The resulting granulometric pattern spectrum and its statistical summaries characterize the proportions of the image containing objects/texture elements of different sizes and shapes. Evolving shape-based textures, arising from a process in which size and/or shape of texture elements present changes with time, are therefore intrinsically well suited to a granulometric approach to texture analysis.

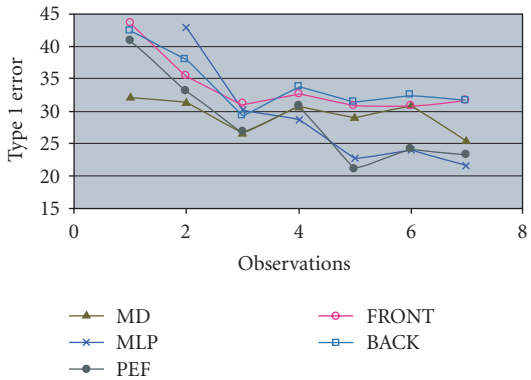
Furthermore, most texture classification techniques allocate images to one of a discrete set of unrelated static textures, rather than placing an image directly to a point on an underlying continuous scale on the basis of a set of evolving textures related by time.

The PEF method uses both granulometry and the dynamic nature of the textures to relate the values of the hidden parameters of an underlying generative image model to the observable pattern spectrum moments, using a statistical model. It then uses this model to estimate directly the evolution status of a new image.

The results shown clearly indicate that using the relationship between the moments and evolution parameters, rather than granulometric information alone,




(a) Type 0 error

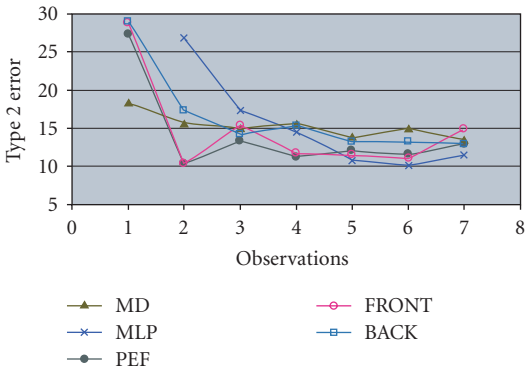



(b) Type 1 error

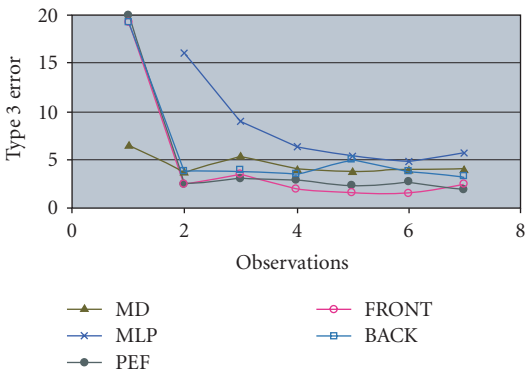
FIGURE 9.17. Percentage error versus number of training observations for corrosion images. (Reproduced from Proceedings of VIE 2005, International Conference on Visual Information Engineering, 4–6 April 2005, Glasgow, UK, ISBN 0-86341-507-5, A. J. Gray, J. Mckenzie, and S. Marshall, "Texture classification of grey scale corrosion images," pages 219–225, © 2005, with permission from Publishing Department of the Institution of Electrical Engineers, London, UK.)

improves classification accuracy for small to moderate training samples, and gives consistent performance for larger training samples.

The PEF method has been shown to perform better than other classifiers for small training sets when the observed data is visually similar to the training data or is controlled by a similar random growth process. This is especially the case when the main priority is for the predicted class to be close to the correct class, if not exactly correct all of the time. PEF performs well when only a few training observations are available, due to its use of information from all time states, and, unlike the maximum likelihood (ML) method, it does not require a minimum number of observations. Compared to both the minimum distance (MD) [84] and pooled maximum likelihood (MLP) [29] classifiers, it performs well, hardly




(a) Type 2 error




(b) Type 3 error

FIGURE 9.18. Percentage error versus number of training observations for corrosion images. (Figure 9.18(a) is reproduced from Proceedings of VIE 2005, International Conference on Visual Information Engineering, 4–6 April 2005, Glasgow, UK, ISBN 0-86341-507-5, A. J. Gray, J. Mckenzie, and S. Marshall, "Texture classification of grey scale corrosion images," pages 219–225, © 2005, with permission from the Publishing Department of the Institution of Electrical Engineers, London, UK.)

ever deviating far from the correct class. This is a great advantage in quality control applications.

As it predicts directly to a time state on a continuous scale, PEF also has the advantage of being able to predict intermediate time states between those represented in the training set. Since it involves multiple linear regression, reliable prediction of time state does require that the training data covers the period of time or severity of interest, to avoid extrapolation [98].

While granulometries are nonlinear, the PEF modeling linearly relates the granulometric moments to the evolution parameters. This facilitates inversion in order to estimate the evolution parameters for a new image. It is assumed that any departures from this model or from the required conditions for a multiple linear

regression will not seriously impact on the accuracy of prediction. However, more generally, nonlinear regression models [72] or time series methods [14] could be used for modeling. Time series models are likely to make back-prediction, as used in the PEF approach, difficult, but are worth considering as the best form of model may depend on the application.

Possible improvements to PEF include using a different point process, that is, more varied and irregular shapes of grain to build the evolution model, and/or different grain density distributions, and different growth functions for the mean grain size and its variance. Regarding the point process, it is likely that new sites of corrosion will depend on the existing corrosion rather than being randomly distributed. A suitable model should result in model parameters being more closely related to the granulometric moments of the real data and therefore more accurate prediction of time state. Optimizing choice of SE(s) [22] or use of multiple SEs in a more complex granulometry [5] are also worth considering.

## Acknowledgments

We gratefully acknowledge the inspiration, encouragement, and advice of Professor Edward R. Dougherty of the Department of Electrical Engineering, Texas A&M University. We are also grateful to the Engineering and Physical Sciences Research Council and Rolls Royce plc, Derby, England, for the CASE studentship which supported the Ph.D. studies of Jennifer McKenzie.

## Bibliography

[1] G. Ayala and J. Domingo, "Spatial size distributions: applications to shape and texture analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 12, pp. 1430–1442, 2001.

[2] J. A. Bangham and S. Marshall, "Image and signal processing with mathematical morphology," *IEE Electronics & Communication Engineering Journal*, vol. 10, no. 3, pp. 117–128, 1998.

[3] S. Baeg, S. Batman, E. R. Dougherty, et al., "Unsupervised morphological granulometric texture segmentation of digital mammograms," *Journal of Electronic Imaging*, vol. 8, no. 1, pp. 65–75, 1999.

[4] R. Bajcsy, "Computer description of textured surfaces," in *Proceedings 3rd International Joint Conference on Artificial Intelligence (IJCAI '73)*, pp. 572–579, Standford, Calif, USA, August 1973.

[5] S. Batman and E. R. Dougherty, "Size distributions for multivariate morphological granulometries: texture classification and statistical properties," *Optical Engineering*, vol. 36, no. 5, pp. 1518–1529, 1997.

[6] P. Brodatz, *Textures: A Photographic Album for Artists and Designers*, Dover, New York, NY, USA, 1966, images available at http://www.ux.his.no/~tranden/brodatz/.

[7] J. R. Carr and F. P. de Miranda, "The semivariogram in comparison to the co-occurrence matrix for classification of image texture," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 36, no. 6, pp. 1945–1952, 1998.

[8] M. J. Chantler and J. P. Stoner, "Automatic interpretation of sonar image sequences using temporal feature measures," *IEEE Journal of Oceanic Engineering*, vol. 22, no. 1, pp. 47–56, 1997.

[9] R. Chellappa and R. Kashyap, "Texture synthesis using 2-D noncausal autoregressive models," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 33, no. 1, pp. 194–203, 1985.

[10] Y. Chen, E. R. Dougherty, S. M. Totterman, and J. P. Hornak, "Classification of trabecular structure in magnetic resonance images based on morphological granulometries," *Magnetic Resonance in Medicine*, vol. 29, no. 3, pp. 358–370, 1993.

[11] Y. Chen and E. R. Dougherty, "Gray-scale morphological granulometric texture classification," *Optical Engineering*, vol. 33, no. 8, pp. 2713–2722, 1994.

[12] S. Chindaro, K. Sirlantzis, and M. C. Fairhurst, "Component based feature space partition and combination in multiple colour spaces for texture classification," in *Proceedings IEE International Conference on Visual Information Engineering (VIE '05)*, pp. 211–218, Glasgow, Scotland, UK, April 2005.

[13] D. A. Clausi, "Comparison and fusion of co-occurrence, Gabor, and MRF texture features for classification of SAR sea ice imagery," *Atmosphere & Oceans*, vol. 39, no. 3, pp. 183–194, 2001.

[14] W. S. Cleveland, *Visualizing Data*, Hobart Press, Summit, NJ, USA, 1993.

[15] R. W. Conners and C. A. Harlow, "A theoretical comparison of texture algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 2, no. 3, pp. 204–222, 1980.

[16] T. J. Dennis and N. G. Dessipris, "Fractal modelling in image texture analysis," *IEE Proceedings. F, Radar and Signal Processing*, vol. 136, no. 5, pp. 227–235, 1989.

[17] P. J. Diggle, "Binary mosaics and the spatial pattern of heather," *Biometrics*, vol. 37, pp. 531–539, 1981.

[18] E. R. Dougherty, *Random Processes for Image Signal Processing*, SPIE/IEEE Series on Imaging Science & Engineering, Wiley-IEEE Press, New York, NY, USA, 1998.

[19] E. R. Dougherty and J. T. Astola, *An Introduction to Nonlinear Image Processing*, vol. TT 16 of *Tutorial Texts in Optical Engineering*, SPIE Press, Bellingham, Wash, USA, 1994.

[20] E. R. Dougherty and J. T. Astola, Eds., *Nonlinear Filters for Image Processing*, SPIE/IEEE Series on Imaging Science & Engineering, SPIE Press, Bellingham, Wash, USA, 1999.

[21] E. R. Dougherty and Y. Chen, "Logical structural filters," *Optical Engineering*, vol. 37, no. 6, pp. 1668–1676, 1998.

[22] E. R. Dougherty and R. P. Loce, "Efficient design strategies for the optimal binary digital morphological filter: probabilities, constraints, and structuring-element libraries," in *Mathematical Morphology in Image Processing*, chapter 2, Marcel Dekker, New York, NY, USA, 1992.

[23] E. R. Dougherty and R. A. Lotufo, *Hands-on Morphological Image Processing*, SPIE Press, Bellingham, Wash, USA, 2003.

[24] E. R. Dougherty and J. B. Pelz, "Morphological granulometric analysis of electrophotographic images-size distribution statistics for process control," *Optical Engineering*, vol. 30, no. 4, pp. 438–445, 1991.

[25] E. R. Dougherty, J. T. Newell, and J. B. Pelz, "Morphological texture-based maximum-likelihood pixel classification based on local granulometric moments," *Pattern Recognition*, vol. 25, no. 10, pp. 1181–1198, 1992.

[26] E. R. Dougherty, J. B. Pelz, F. Sand, and A. Lent, "Morphological image segmentation by local granulometric size distributions," *Journal of Electronic Imaging*, vol. 1, no. 1, pp. 46–60, 1992.

[27] A. Drimbarean and P. F. Whelan, "Experiments in colour texture analysis," *Pattern Recognition Letters*, vol. 22, no. 10, pp. 1161–1167, 2001.

[28] R. O. Duda and P. E. Hart, *Pattern Recognition and Scene Analysis*, John Wiley & Sons, New York, NY, USA, 1973.

[29] B. S. Everitt and G. Dunn, *Applied Multivariate Data Analysis*, Edward Arnold, London, UK, 1991.

[30] S. Fogel and D. Sagi, "Gabor filters as texture discriminators," *Biological Cybernetics*, vol. 61, pp. 103–113, 1989.

[31] M. M. Galloway, "Texture analysis using gray level run lengths," *Computer Graphics and Image Processing*, vol. 4, pp. 172–179, 1975.

[32] P. García Sevilla and M. Petrou, "Analysis of irregularly shaped texture regions," *Computer Vision and Image Understanding*, vol. 84, no. 1, pp. 62–76, 2001.

[33] R. Gnanadesikan, *Methods for Statistical Data Analysis of Multivariate Observations*, John Wiley & Sons, New York, NY, USA, 1977.

[34] J. F. Haddon and J. F. Boyce, "Co-occurrence matrices for image analysis," *IEE Electronics & Communication Engineering Journal*, vol. 5, no. 2, pp. 71–83, 1993.
[35] J. C. Handley and E. R. Dougherty, "Maximum-likelihood-estimation for the two-dimensional discrete boolean random set and function models using multidimensional linear samples," *CVGIP: Graphical Models and Image Processing*, vol. 59, no. 4, pp. 221–231, 1997.
[36] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision, Vol. 1*, Addison-Wesley, Reading, Mass, USA, 1992.
[37] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural features for image classification," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 3, no. 6, pp. 610–621, 1973.
[38] G. M. Haley and B. S. Manjunath, "Rotation-invariant texture classification using a complete space-frequency model," *IEEE Transactions on Image Processing*, vol. 8, no. 2, pp. 255–269, 1999.
[39] E. Hernández and G. L. Weiss, *A First Course on Wavelets*, CRC Press, New York, NY, USA, 1996.
[40] G. W. Horgan, "Mathematical morphology for analysing soil structure from images," *European Journal of Soil Science*, vol. 49, no. 2, pp. 161–173, 1998.
[41] G. W. Horgan, C. A. Reid, and C. A. Glasbey, "Biological image processing and enhancement," in *Image Processing and Analysis: A Practical Approach*, R. Baldock and J. Graham, Eds., pp. 37–67, Oxford University Press, Oxford, UK, 2000.
[42] B. B. Hubbard, *The World According to Wavelets: The Story of a Mathematical Technique in the Making*, A. K. Peters, Wellesley, Mass, USA, 1995.
[43] H. Iversen and T. Lonnestad, "An evaluation of stochastic models for analysis and synthesis of gray-scale texture," *Pattern Recognition Letters*, vol. 15, no. 6, pp. 575–585, 1994.
[44] A. K. Jain and F. Farrokhnia, "Unsupervised texture segmentation using Gabor filters," *Pattern Recognition*, vol. 24, no. 12, pp. 1167–1186, 1991.
[45] T. Jossang and J. Feder, "The fractal characterization of rough surfaces," *Physica Scripta*, vol. T44, pp. 9–14, 1992.
[46] A. K. Katsaggelos and T. Chun-Jen, "Iterative image restoration," in *Handbook of Image and Video Processing*, A. Bovik, Ed., pp. 208–209, Academic Press, London, UK, 2000.
[47] M. Köppen, C. H. Nowack, and G. Rösel, "Pareto-morphology for color image processing," in *Proceedings 11th Scandinavian Conference on Image Analysis (SCIA '99)*, vol. 1, pp. 195–202, Kangerlussuaq, Greenland, June 1999.
[48] S. Krishnamachari and R. Chellappa, "Multiresolution Gauss-Markov random field models for texture segmentation," *IEEE Transactions on Image Processing*, vol. 6, no. 2, pp. 251–267, 1997.
[49] T. Kurita and N. Otsu, "Texture classification by higher order local autocorrelation features," in *Proceedings 1st Asian Conference on Computer Vision (ACCV '93)*, pp. 175–178, Osaka, Japan, November 1993.
[50] S. T. Kyvelidis, L. Lykouropoulos, and N. Kouloumbi, "Digital system for detecting, classifying and fast retrieving corrosion generated defects," *Journal of Coatings Technology*, vol. 73/915, pp. 67–73, 2001.
[51] Y. Liu, T. Zhao, and J. Zhang, "Learning multispectral texture features for cervical cancer detection," in *Proceedings 1st IEEE International Symposium on Biomedical Imaging: Macro to Nano (ISBI '02)*, pp. 169–172, Washington, DC, USA, July 2002.
[52] G. McGunnigle and M. J. Chantler, "Modelling deposition of surface texture," *Electronics Letters*, vol. 37, no. 12, pp. 749–750, 2001.
[53] J. McKenzie, S. Marshall, A. J. Gray, and E. R. Dougherty, "Morphological texture analysis using the texture evolution function," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 17, no. 2, pp. 167–185, 2003.
[54] J. McKenzie, *Classification of dynamically evolving textures using evolution functions*, Ph.D. thesis, University of Strathclyde, Glasgow, Scotland, UK, 2004.
[55] S. G. Mallat, "Multiresolution approximations and wavelet orthonormal bases of $L^2(R)$," *Transactions of the American Mathematical Society*, vol. 315, no. 1, pp. 69–87, 1989.
[56] S. G. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, pp. 674–693, 1989.
[57] B. S. Manjunath and W. Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 837–842, 1996.

[58] B. S. Manjunath, G. M. Haley, and W. Y. Ma, "Multiband techniques for texture classification and segmentation," in *Handbook of Image and Video Processing*, A. Bovik, Ed., pp. 367–381, Academic Press, London, UK, 2000.

[59] G. Matheron, *Random Sets and Integral Geometry*, Wiley Series in Probability and Mathematical Statistics, John Wiley & Sons, New York, NY, USA, 1975.

[60] J. Miles and M. Shevlin, *Applying Regression and Correlation: A Guide for Students and Researchers*, SAGE Publications, London, UK, 2001.

[61] M. Mirmehdi and M. Petrou, "Segmentation of color textures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 2, pp. 142–159, 2000.

[62] A. Monadjemi, B. T. Thomas, and M. Mirmehdi, "Speed v. accuracy for high resolution colour texture classification," in *Proceedings 13th British Machine Vision Conference (BMVC '02)*, pp. 143–152, BMVA Press, Cardiff, UK, September 2002.

[63] P. Moulin, "Multiscale image decompositions and wavelets," in *Handbook of Image and Video Processing*, A. Bovik, Ed., pp. 289–300, Academic Press, London, UK, 2000.

[64] W. M. Mullins, E. J. Shumaker, and G. J. Tyler, "Stochastic kinetics of corrosion and fractal surface evolution," *The Journal of Corrosion Science and Engineering*, vol. 1, paper 7, 1997, at http://www.jcse.org/Volume1/paper7/v1p7.html.

[65] T. Ojala, M. Pietikainen, and D. Harwood, "A comparative study of texture measures with classification based on feature distributions," *Pattern Recognition*, vol. 29, no. 1, pp. 51–59, 1996.

[66] M. Petrou and G. Lazaridis, "Texture segmentation using local Walsh coefficients," in *Proceedings IEE International Conference on Visual Information Engineering (VIE '05)*, pp. 189–194, Institution of Electrical Engineers, Glasgow, Scotland, UK, April 2005.

[67] O. Pichler, A. Teuner, and B. J. Hosticka, "A comparison of texture feature extraction using adaptive Gabor filtering, pyramidal and tree structured wavelet transforms," *Pattern Recognition*, vol. 29, no. 5, pp. 733–742, 1996.

[68] R. Porter and N. Canagarajah, "A robust automatic clustering scheme for image segmentation using wavelets," *IEEE Transactions on Image Processing*, vol. 5, no. 4, pp. 662–665, 1996.

[69] K. Rajesh, C. V. Jawahar, S. Sengupta, and S. Sinha, "Performance analysis of textural features for characterization and classification of SAR images," *International Journal of Remote Sensing*, vol. 22, no. 8, pp. 1555–1569, 2001.

[70] T. Randen and J. H. Husoy, "Multichannel filtering for image texture segmentation," *Optical Engineering*, vol. 33, no. 8, pp. 2617–2625, 1994.

[71] T. Randen and J. H. Husoy, "Filtering for texture classification: a comparative study," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 4, pp. 291–310, 1999.

[72] J. O. Rawlings, *Applied Regression Analysis: A Research Tool*, Statistics/Probability Series, Wadsworth and Brooks/Cole, Belmont, Calif, USA, 1988.

[73] S. Rebbapragada, M. J. Palakal, R. M. Pidaparti, and C. R. Jones, "Corrosion detection and quantification using image processing for aging aircraft panels," in *Proceedings 3rd Joint FAA/DoD/NASA Conference on Aging Aircraft*, Albuquerque, NM, USA, September 1999.

[74] T. R. Reed and J. M. H. du Buf, "A review of recent texture segmentation and feature extraction techniques," *CVGIP: Image Understanding*, vol. 57, no. 3, pp. 359–372, 1993.

[75] A. Rosenfeld, "Multi-resolution image representation," in *Digital Image Analysis*, E. S. Levadi, Ed., pp. 22–25, Pitman Press, London, UK, 1984.

[76] F. Sand and E. R. Dougherty, "Asymptotic normality of the morphological pattern-spectrum moments and orthogonal granulometric generators," *Journal of Visual Communication and Image Representation*, vol. 3, no. 2, pp. 203–214, 1992.

[77] F. Sand and E. R. Dougherty, "Asymptotic granulometric mixing theorem: morphological estimation of sizing parameters and mixture proportions," *Pattern Recognition*, vol. 31, no. 1, pp. 53–61, 1998.

[78] J. Serra, *Image Analysis and Mathematical Morphology*, Academic Press, New York, NY, USA, 1982.

[79] J. Serra, *Image Analysis and Mathematical Morphology. Vol. 2: Theoretical Advances*, Academic Press, London , UK, 1988.

[80] J. Serra and G. Verchery, "Mathematical morphology applied to fibre composite materials," *Film Science and Technology*, vol. 6, pp. 141–158, 1973.

[81] L. Shafarenko, M. Petrou, and J. Kittler, "Automatic watershed segmentation of randomly textured color images," *IEEE Transactions on Image Processing*, vol. 6, no. 11, pp. 1530–1544, 1997.

[82] M. Singh and S. Singh, "Spatial texture analysis: a comparative study," at http://citeseer.ist.psu.edu/507622.html.

[83] S. Singh and M. Sharma, "Texture analysis experiments with Meastex and Vistex benchmarks," in *Proceedings 2nd International Conference on Advances in Pattern Recognition (ICAPR '01)*, vol. 2013 of *Lecture Notes in Computer Science*, pp. 417–424, Rio de Janeiro, Brazil, March 2001, http://citeseer.ist.psu.edu/528275.html.

[84] K. Sivakumar and J. Goutsias, "Morphologically constrained GRFs: applications to texture synthesis and analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 2, pp. 99–113, 1999.

[85] K. Sivakumar, Y. Balagurunathan, and E. R. Dougherty, "Asymptotic joint normality of the granulometric moments," *Pattern Recognition Letters*, vol. 22, no. 14, pp. 1537–1543, 2001.

[86] K. Sivakumar, M. J. Patel, N. Kehtarnavaz, Y. Balagurunathan, and E. R. Dougherty, "A constant-time algorithm for erosions/dilations with applications to a morphological texture feature computation," *Real Time Imaging*, vol. 6, no. 3, pp. 223–239, 2000.

[87] L.-K. Soh and C. Tsatsoulis, "Texture analysis of SAR sea ice imagery using gray level co-occurrence matrices," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 37, no. 2, pp. 780–795, 1999.

[88] K. Y. Song, J. Kittler, and M. Petrou, "Defect detection in random colour textures," *Image and Vision Computing*, vol. 14, no. 9, pp. 667–683, 1996.

[89] J. Teuber, *Digital Image Processing*, Series in Acoustics, Speech and Signal Processing, Prentice-Hall, Englewood Cliffs, NJ, USA, 1993.

[90] N. Theera-Umpon and P. D. Gader, "Counting white blood cells using morphological granulometries," *Journal of Electronic Imaging*, vol. 9, no. 2, pp. 170–177, 2000.

[91] F. Tomita and S. Tsuji, *Computer Analysis of Visual Textures*, Kluwer Academic, Dordrecht, The Netherlands, 1990.

[92] K. R. Trethewey and P. R. Roberge, "The characterization of surface profiles created by localized corrosion with stochastic and fractal analysis techniques," in *Proceedings Localized Damage III. Computer-Aided Assessment and Control*, vol. 3, pp. 323–330, Udine, Italy, June 1994.

[93] M. Unser, "Texture classification and segmentation using wavelet frames," *IEEE Transactions on Image Processing*, vol. 4, no. 11, pp. 1549–1560, 1995.

[94] A. L. Vickers and J. W. Modestino, "A maximum likelihood approach to texture classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 4, no. 1, pp. 61–68, 1982.

[95] VisTex Database, MIT Media Library, at http://vismod.media.mit.edu/vismod/imagery/VisionTexture/vistex.html.

[96] J. S. Walker, *Fourier Analysis*, Oxford University Press, New York, NY, USA, 1988.

[97] C. Weiping and X. Chenghui, "Fractal structure of uniform corrosion surface for Fe-based alloys," *Journal of Materials Science Letters*, vol. 16, no. 2, pp. 113–114, 1997.

[98] G. B. Wetherill, P. Duncombe, P. Kenward, M. Kollerstrom, S. R. Paul, and B. J. Bowden, *Regression Analysis with Applications*, Monographs on Statistics and Applied Probability, Chapman & Hall, London, UK, 1986.

[99] D. S. Wickramanayake, E. A. Edirisinghe, and H. E. Bez, "A wavelet based image quilting approach to fast, multiresolution texture synthesis," in *Proceedings IEE International Conference on Visual Information Engineering (VIE '05)*, pp. 93–100, Glasgow, Scotland, UK, April 2005.

[100] X. Tang, "Texture information in run-length matrices," *IEEE Transactions on Image Processing*, vol. 7, no. 11, pp. 1602–1609, 1998.

[101] Y.-G. Lee, J.-H. Lee, and Y.-C. Hsueh, "Genetic-based fuzzy hit-or-miss texture spectrum for texture analysis," *Electronics Letters*, vol. 31, no. 23, pp. 1986–1988, 1995.

[102] J. Zhang, P. Fieguth, and D. Wang, "Random field models," in *Handbook of Image and Video Processing*, A. Bovik, Ed., pp. 301–312, Academic Press, London, UK, 2000.

A. J. Gray: Department of Statistics and Modeling Science, University of Strathclyde, Livingstone Tower, 26 Richmond Street, Glasgow G1 1XH, UK

*Email*: alison@stams.strath.ac.uk

S. Marshall: Department of Electrical and Electronic Engineering, University of Strathclyde, Royal College, 204 George Street, Glasgow G1 1XW, UK

*Email*: s.marshall@eee.strath.ac.uk

J. McKenzie: Department of Electrical and Electronic Engineering, University of Strathclyde, Royal College, 204 George Street, Glasgow G1 1XW, UK

*Email*: jennym@spd.eee.strath.ac.uk

# 10 Multichannel weighted medians

Yinbo Li and Gonzalo R. Arce

Weighted medians over multichannel signals are not uniquely defined. Due to their simplicity, Astola et al.'s vector median (VM) has received considerable attention particularly in image processing applications. In this chapter, we show that the VM and its direct extension, the weighted VM, are limited as they do not fully utilize the cross-channel correlation. In fact, VM treats all subchannel components independently of each other. By revisiting the principles of maximum-likelihood estimation of location in a multivariate signal space, we describe a set of new and simple multichannel weighted median filters which can capture cross-channel information effectively both in real domain and complex domain. Their optimal filter derivations are also presented, followed by a series of simulations from color image denoising to array signal processing, where the efficiency of these filtering structures is illustrated.

## 10.1. Introduction

Multichannel signal processing [1, 22] has undergone rapid developments in the past decade, primarily due to its importance to multispectrum imaging, array processing, multimedia, and medical signal processing [19]. Noise in color imaging is often impulsive, and since edges and details are of paramount importance, it is natural to extend weighted medians and order statistic filters [2, 31] from the scalar domain into the multichannel space. The extension of the weighted median (WM) for use with multidimensional (multichannel) signals is however not straightforward. Sorting multicomponent (vector) values and selecting the middle value is not well defined as in the scalar case, see Barnett (1976) [4]. In consequence, the weighted median filtering operation of a multidimensional signal can be achieved in a number of ways among which the most well known are marginal medians of orthogonal coordinates by Hayford (1902) [10], $L_1$-norm median, by Gini and Galvani (1929) [5] and Haldane (1948) [9] that minimizes the sum of distances to all samples, the halfplane median by Tukey (1975) [29] that minimizes the maximum number of samples on a halfplane, convex hull median by Barnett (1976) [4] and Shamos (1976) [26] that results from the continuous "peeling" off pairs of

extreme samples, the simplex median by Oja (1983) [20] that minimizes the sum of the volumes of all simplices formed by the output and some subsets of the data, the simplicial median of Liu (1990) [16] that maximizes the number of simplices that contain it, and the hyperplane median by Rousseeuw and Hubert (1999) [23] that maximizes the *hyperplane depth*. For historical reviews on multivariate medians, see Small (1990) [27] and Aloupis et al. (2001) [3]. Other approaches can be found in [8, 13, 21], and [28].

A problem with many definitions of multivariate medians is that they have more conceptual meaning than practical use because of their high computational complexities. The algorithms used to compute the $L_1$ median often involve gradient techniques or iterations that can only provide numerical solutions as shown by Groß and Strempel (1998) [7], and even the fastest algorithm up-to-date for the Oja median is about $O(N^3 \log N)$ in time, see Aloupis et al. (2001) [3]. Moreover, they usually have difficulties with their extension to structures admitting weights and the optimal design of these weights.

Unlike the above-mentioned approaches, the so-called vector median (VM), proposed by Astola et al. [1] in 1990, has since then received considerable attention in signal processing research. Relaxed on its mathematical rigorousness but focused on its practical use, the basic idea of VM is to confine the filter output to be one of the $N$ vector-valued input samples which minimizes the sum of $L_1$ distances from this output to all other samples in the observation window. Denote $\{\vec{X}_i\}$ as the sample set. Astola's VM is defined as

$$\vec{Y} = \arg\min_{\vec{X} \in \{\vec{X}_i\}} \sum_{i=1}^{N} \left|\left|\vec{X} - \vec{X}_i\right|\right|_p, \tag{10.1}$$

where $\|\cdot\|_p$ denotes $L_p$ norm, where $\|\vec{X}\|_p = (\sum |X_i|^p)^{1/p}$. An otherwise full multivariate space search is thus replaced by at most $N$ calculations of the cost function. The computational complexity is thus greatly reduced. The vector median is a selection type of robust filter, a property also possessed by univariate median filters. In order to expand the capabilities of the vector median, the weighted vector median (WVM) was introduced as a direct extension [30],

$$\vec{Y} = \arg\min_{\vec{X} \in \{\vec{X}_i\}} \sum_{i=1}^{N} W_i \left|\left|\vec{X} - \vec{X}_i\right|\right|_p, \tag{10.2}$$

where a weighted cost function is defined as such that the $L_1$ distances are first weighted by nonnegative scalars before they are summed together.

Although the WVM finds its immediate applications in color imaging, and has several optimization algorithms in existence [17, 18, 25], the principles of parameter estimation reveal that the very structure of weighted vector medians are limited and as such they are not suitable for a broad class of problems in multichannel signal processing. In fact, Astola's vector median is derived from the maximum likelihood (ML) estimation of location for independent and identically distributed (i.i.d.) vector-valued samples obeying a Laplacian distribution.

The weighted vector median, in turn, emerges from the location estimate of independent (but not identically distributed) vector-valued samples, where only the scale of each input vector sample varies. The multichannel components of each sample are, however, still considered mutually independent in both cases. Even though their selection-type filter outputs preserve the cross correlation in the data, weighted vector medians are still limited since the cross-channel correlation structure, inherently present in most multichannel applications, are not fully exploited. Weights in weighted VM filtering are applied treating each individual vector sample in the processing window differently, hence they utilize the spatial correlation. Such cross-channel weighting is not used so as to treat channel components differently within each vector sample, thus, the weighted VM cannot fully exploit the cross correlation between channels. In this sense, the weighted vector median in [30] is cross-channel blind, and the vector median in [1] is both spatially and cross-channel blind.

## 10.2. Multichannel weighted median filtering structures

The notation used hereafter is clarified first. Let $M$ represent the dimension of the multivariate data, $N$ the filter length. Throughout the paper, the vectors in the time (or spatial) domain with length $N$ are denoted using boldface letters like $\mathbf{X}$, vectors in the spectral domain with length $M$ are represented in regular font but with an arrow on top like $\vec{X}_i$, matrices in spectral domain will be represented using blackboard bold font like $\mathbb{W}$. Vector multiplications can only happen between two vectors that are in the same domain. The transpose superscript $^T$ is used in both domains.

### 10.2.1. Multichannel weighted median filter I

As in the scalar case, the multivariate filtering structure is derived from the maximum likelihood estimation of location, this time in a multivariate signal space. Consider a set of independent but not identically distributed vector-valued samples, each obeying a joint Gaussian distribution with the same location parameter $\vec{\beta}$,

$$f\left(\vec{X}_i\right) = \frac{1}{(2\pi)^{M/2}\left|\mathbb{C}_i\right|^{1/2}} e^{-(1/2)\left(\vec{X}_i-\vec{\beta}\right)^T \mathbb{C}_i^{-1}\left(\vec{X}_i-\vec{\beta}\right)}, \tag{10.3}$$

where $\vec{X}_i$ and $\vec{\beta}$ are all $M$-variate column vectors, and $\mathbb{C}_i^{-1}$ is the inverse of the $M \times M$ covariance matrix of the sample $\vec{X}_i$. The maximum likelihood estimation of location $\vec{\beta}$ can be derived as

$$\vec{\beta} = \left(\sum_{i=1}^{N} \mathbb{C}_i\right)\left(\sum_{i=1}^{N} \mathbb{C}_i^{-1}\vec{X}_i\right). \tag{10.4}$$

A general multivariate filtering structure results from (10.4) as

$$\vec{Y} = \sum_{i=1}^{N} \mathbb{W}_i^T \vec{X}_i = \begin{bmatrix} \sum_{i=1}^{N} \sum_{j=1}^{M} W_i^{j1} X_i^j \\ \vdots \\ \sum_{i=1}^{N} \sum_{j=1}^{M} W_i^{jM} X_i^j \end{bmatrix}, \tag{10.5}$$

where $\mathbb{W}_i^T = (\sum_{i=1}^{N} \mathbb{C}_i)\mathbb{C}_i^{-1}$. The operation in (10.5) is referred to as the complete multichannel linear filter (CML). Obviously, the sample set within each channel is the same $\{X_i^j |_{i=1}^{N} |_{j=1}^{M}\}$, but the corresponding weights $\{W_i^{jl} |_{i=1}^{N} |_{j=1}^{M}\}$ are all different for different channel $l = 1, \ldots, M$. Hence the optimal CML filter can be considered as $M$ independent univariate linear filters each with $NM$ samples and weights. The total number of the weights is $NM^2$. Its corresponding median version, a.k.a the complete multichannel weighted median filter (CMWM), can thus be expressed as

$$\vec{Y} = \begin{bmatrix} \text{MEDIAN}\left( |W_i^{j1}| \diamond \text{sgn}(W_i^{j1}) X_i^j |_{j=1}^{M} |_{i=1}^{N} \right) \\ \vdots \\ \text{MEDIAN}\left( |W_i^{jM}| \diamond \text{sgn}(W_i^{jM}) X_i^j |_{j=1}^{M} |_{i=1}^{N} \right) \end{bmatrix}. \tag{10.6}$$

The weighted median in each channel is a real-valued operation with signs of the weights coupled into the corresponding samples. See [2] for its definition and computation.

An example of an optimal filter design algorithm for (10.5) is shown by Robinson (1983) [24]. The optimization of (10.6) can be implemented by $M$ independent weighted median filters each as outlined in [2]. Unfortunately, these structures are often impractical due to the overwhelming size of the weight set. For instance, even using a $3 \times 3$ window to filter a 3-channel color image requires the optimization of 81 weights. Alternate filter structures requiring lesser weights yet preserving its capability of capturing cross-channel information are needed for both linear and median cases. The approach described next provides one solution to this problem [14].

In most multichannel applications, signals from subchannels are correlated. Very often the cross-channel correlation in these cases has some inherent structure. The simplest structure would be that the covariance matrix is a constant matrix throughout the entire signal. This is true when the multivariate samples are independent and identically distributed. A slightly more complicated structure would be that the inverse covariance matrices at different time indices $\mathbb{C}_i^{-1}$ though may not hold constant but are all proportional to each other, and differ only by scale factors. Mathematically this is expressed as

$$\mathbb{C}_i^{-1} = q_i \mathbb{C}^{-1}. \tag{10.7}$$

This can be seen as each multivariate sample is drawn from the same multivariate distribution but with a different variance parameter. The corresponding MLE is then

$$\vec{\beta} = \left( \sum_{i=1}^{N} q_i \mathbb{C}^{-1} \right)^{-1} \left( \sum_{i=1}^{N} q_i \mathbb{C}^{-1} \vec{X}_i \right), \tag{10.8}$$

where $(\sum_{i=1}^{N} q_i \mathbb{C}^{-1})^{-1}$ is a normalization constant and $\sum_{i=1}^{N} q_i \mathbb{C}^{-1} \vec{X}_i$ provides the filtering structure. Removing the normalization constant, the filtering structure can be formulated as

$$\vec{Y} = \sum_{i=1}^{N} V_i \mathbb{W}^T \vec{X}_i \tag{10.9}$$

$$= \begin{bmatrix} \sum_{i=1}^{N} V_i \left( \sum_{j=1}^{M} W^{j1} X_i^j \right) \\ \vdots \\ \sum_{i=1}^{N} V_i \left( \sum_{j=1}^{M} W^{jM} X_i^j \right) \end{bmatrix}, \tag{10.10}$$

where $V_i$ is the time-dependent weight applied to the $i$th vector sample in the observation window and $W^{jk}$ is the cross-channel weight exploiting the correlation between the $j$th and $k$th components of a sample. The idea behind this structure with two sets of weights is clear: the cross-channel weight matrix $\mathbb{W}$ reflects the inherent correlation between subchannels, whilst the set of time-dependent weights $V_i$ is responsible for capturing the time correlation between the neighboring samples. In the remainder of this chapter, we will refer to this linear structure as the multichannel linear (ML) filter I. The filter thus consists of $M^2 + N$ weights. In the example of a RGB image with a $3 \times 3$ window, the number of weights would be reduced from 81 to 18.

Even though it is mathematically intractable to derive a similar result as in (10.10) from a multivariate Laplace distribution,[1] it is still possible to define a nonlinear multichannel filter by direct analogy by replacing the summations in (10.10) with median operators. This filter is referred to as the multichannel weighted median (MWM) filter I and is defined as follows [14]:

$$\vec{Y} = \begin{bmatrix} \text{MEDIAN} \left( |V_i| \diamond \text{sgn}(V_i) Q_i^1 |_{i=1}^{N} \right) \\ \vdots \\ \text{MEDIAN} \left( |V_i| \diamond \text{sgn}(V_i) Q_i^M |_{i=1}^{N} \right) \end{bmatrix}, \tag{10.11}$$

where $Q_i^l = \text{MEDIAN}(|W^{jl}| \diamond \text{sgn}(W^{jl}) X_i^j |_{j=1}^{M})$ for $l = 1, \ldots, M$.

[1] The multivariate Laplace distribution is expressed by the Bessel function [12], it is thus intractable to derive a simple closed form solution for the ML estimate.

### 10.2.2. Multichannel weighted median filter II

There are some applications where the initial assumption stated in (10.7) may not be appropriate. In these cases, the inherent cross-channel correlation structure is more complicated, and the constraint on $\mathbb{C}_i$ is too strong. Notice that, in the ML filter I structure, the time-dependent weight $V_i$ is equally applied to all subchannels of the sample $\vec{X}_i$. This inspires us to replace (10.7) by

$$\mathbb{C}_i^{-1} = \operatorname{diag}(\vec{q}_i)\mathbb{C}^{-1} = \begin{bmatrix} q_i^1 C^{11} & \cdots & q_i^1 C^{1M} \\ \vdots & \ddots & \vdots \\ q_i^M C^{M1} & \cdots & q_i^M C^{MM} \end{bmatrix}, \tag{10.12}$$

where $\vec{q}_i = [q_i^1 \quad \cdots \quad q_i^M]^T$, $\operatorname{diag}(\vec{q}_i)$ is a matrix with $\vec{q}_i$ elements on its diagonal and zeros elsewhere. In this case, the cross-channel correlation is no longer considered stationary, and the $q_i^j$ represent the correlation between components of different samples in the observation window. The ML filter II is thus defined as

$$\vec{Y} = \sum_{i=1}^{N} \operatorname{diag}(\vec{V}_i)\mathbb{W}^T\vec{X}_i = \begin{bmatrix} \sum_{i=1}^{N} V_i^1 \left( \sum_{j=1}^{M} W^{j1} X_i^j \right) \\ \vdots \\ \sum_{i=1}^{N} V_i^M \left( \sum_{j=1}^{M} W^{jM} X_i^j \right) \end{bmatrix}, \tag{10.13}$$

where $\vec{V}_i$ is an $M$-variate weight vector, $V_i^l$ is the weight reflecting the influence of the $l$th component of the $i$th sample in the $l$th component of the output. The weights $W^{ij}$ have the same meaning as in the MWM filter I.

Applying the same analogy used in the previous case, a more general multichannel weighted median filter (MWMII) structure can be defined as

$$\vec{Y} = \begin{bmatrix} \operatorname{MEDIAN}\left( |V_i^1| \diamond \operatorname{sgn}(V_i^1) Q_i^1 |_{i=1}^{N} \right) \\ \vdots \\ \operatorname{MEDIAN}\left( |V_i^M| \diamond \operatorname{sgn}(V_i^M) Q_i^M |_{i=1}^{N} \right) \end{bmatrix}, \tag{10.14}$$

where $Q_i^l$ is the same as in (10.11). The number of weights increases compared to the MWM filter I, but is still significantly smaller compared to the number of weights required by the complete version of the filter in (10.5). For the color image filtering example, the number of weights will be $M \times (N + M) = 36$.

In the following section, optimal adaptive algorithms for the structures in (10.11) and (10.14) are described.

Table 10.1. Optimal filtering structures for multichannel signals.

| Case | $\mathbb{C}_i^{-1}$ | Optimal linear filtering structure | Median equivalence | No. of weights |
|---|---|---|---|---|
| 1 | $\sigma^2\mathbb{I}$ | $\vec{Y} = \sum_i W\vec{X}_i$ | VM (10.1) | 1 |
| 2 | $\sigma_i^2\mathbb{I}$ | $\vec{Y} = \sum_i W_i\vec{X}_i$ | WVM (10.2) | $N$ |
| 3 | $\sigma^2\,\mathrm{diag}(\gamma_1,\ldots,\gamma_M)$ | $\vec{Y} = \sum_i \mathrm{diag}(W^1,\ldots,W^M)\vec{X}_i$ | — | $M$ |
| 4 | $\sigma_i^2\,\mathrm{diag}(\gamma_1,\ldots,\gamma_M)$ | $\vec{Y} = \sum_i V_i\,\mathrm{diag}(W^1,\ldots,W^M)\vec{X}_i$ | — | $N+M$ |
| 5 | $\mathbb{C}^{-1}$ | $\vec{Y} = \sum_i \mathbb{W}^T\vec{X}_i$ | — | $M^2$ |
| 6 | $q_i\mathbb{C}^{-1}$ | $\vec{Y} = \sum_i V_i\mathbb{W}^T\vec{X}_i$ | MWMI (10.11) | $N+M^2$ |
| 7 | $\mathrm{diag}(q_i^1,\ldots,q_i^M)\mathbb{C}^{-1}$ | $\vec{Y} = \sum_i \mathrm{diag}(V_i^1,\ldots,V_i^M)\mathbb{W}^T\vec{X}_i$ | MWMII (10.14) | $NM+M^2$ |
| 8 | $\mathbb{C}_i^{-1}$ | $\vec{Y} = \sum_i \mathbb{W}_i^T\vec{X}_i$ | CMWM (10.6) | $NM^2$ |

### 10.2.3. Optimal filtering structures for multichannel signals

A general strategy for finding the optimal filtering structure for a particular multichannel signal can be deducted from previous discussions. The key to the problem holds in characterization of the covariance matrices $\mathbb{C}_i$ (or its inverse). In general, the less structured the covariance matrix is, the more parameters are needed to describe it, and the more filter weightings are required. "Optimal" in this sense means that the number of weights the filter has should be exactly the same as the number of parameters the covariance matrix has, and the filter and the covariance matrix should be matched in structure. A summary of various structures of the covariance matrix and corresponding optimal filtering structures is presented in Table 10.1.

## 10.3. Filter optimization

Assume that the observed process $\vec{X}(n)$ is statistically related to a desired process $\vec{D}(n)$ of interest, typically considered a transformed or corrupted version of $\vec{D}(n)$. The filter input vector at time $n$ is

$$\mathbf{X}(n) = \left[\vec{X}_1(n) \quad \vec{X}_2(n) \quad \cdots \quad \vec{X}_N(n)\right]^T, \tag{10.15}$$

where $\vec{X}_i(n) = [X_i^1(n) \quad X_i^2(n) \quad \cdots \quad X_i^M(n)]^T$. The desired signal is $\vec{D}(n) = [D^1(n) \quad D^2(n) \quad \cdots \quad D^M(n)]^T$. Denote $\vec{e} = \vec{D} - \hat{\vec{D}}$ as the multivariate error signal, $e^l = D^l - \hat{D}^l$ its $l$th element. Also, the cross-channel weight matrix is

$$\mathbb{W} = \begin{bmatrix} W^{11} & \cdots & W^{1M} \\ \vdots & \ddots & \vdots \\ W^{M1} & \cdots & W^{MM} \end{bmatrix}. \tag{10.16}$$

Optimization for the ML filter I and II is fairly simple, and can be found in [14].

### 10.3.1. Optimization for the MWM filter I

Denote $Q_i^l = \text{MEDIAN}(|W^{jl}| \diamond \text{sgn}(W^{jl})X_i^j|_{j=1}^M)$ for $l = 1,\ldots,M$, then the output of the MWM can be defined as

$$\hat{\vec{D}} = \begin{bmatrix} \hat{D}^1 & \hat{D}^2 & \cdots & \hat{D}^M \end{bmatrix}^T, \tag{10.17}$$

where

$$\hat{D}^l = \text{MEDIAN}\left(|V_i| \diamond \text{sgn}(V_i)Q_i^l|_{i=1}^N\right), \quad l = 1,\ldots,M. \tag{10.18}$$

Applying the real-valued threshold decomposition technique as shown in [2], we can rewrite (10.18) to be analyzable as follows:

$$\begin{aligned} \hat{D}^l &= \frac{1}{2}\int \text{MEDIAN}\left(|V_i| \diamond \text{sgn}\left(\text{sgn}(V_i)Q_i^l - p^l\right)|_{i=1}^N\right)dp^l \\ &= \frac{1}{2}\int \text{sgn}\left(\mathbf{V}_a^T\mathbf{G}^{p^l}\right)dp^l, \end{aligned} \tag{10.19}$$

where

$$\begin{aligned} \mathbf{V}_a &= \begin{bmatrix} |V_1| & |V_2| & \cdots & |V_N| \end{bmatrix}^T, \\ \mathbf{G}^{p^l} &= \begin{bmatrix} \text{sgn}\left(\text{sgn}(V_1)Q_1^l - p^l\right) & \cdots & \text{sgn}\left(\text{sgn}(V_N)Q_N^l - p^l\right) \end{bmatrix}^T. \end{aligned} \tag{10.20}$$

Similarly, by defining

$$\begin{aligned} \vec{W}_a^l &= \begin{bmatrix} |W^{1l}| & |W^{2l}| & \cdots & |W^{Ml}| \end{bmatrix}^T, \\ \vec{S}_i^{q_i^l} &= \begin{bmatrix} \text{sgn}\left(\text{sgn}(W^{1l})X_i^1 - q_i^l\right) & \cdots & \text{sgn}\left(\text{sgn}(W^{Ml})X_i^M - q_i^l\right) \end{bmatrix}^T, \end{aligned} \tag{10.21}$$

the inner weighted medians will have the following thresholded representation:

$$\begin{aligned} Q_i^l &= \frac{1}{2}\int \text{MEDIAN}\left(|W^{jl}| \diamond \text{sgn}\left(\text{sgn}(W^{jl})X_i^j - q_i^l\right)|_{j=1}^M\right)dq_i^l \\ &= \frac{1}{2}\int \text{sgn}\left(\left(\vec{W}_a^l\right)^T\vec{S}_i^{q_i^l}\right)dq_i^l. \end{aligned} \tag{10.22}$$

Under the least mean absolute (LMA) error criterion, the cost function to minimize is

$$J_1(\mathbf{V},\mathbb{W}) = E\{||\vec{D} - \hat{\vec{D}}||_1\} \tag{10.23}$$

$$= E\left\{\sum_{l=1}^{M} |D^l - \hat{D}^l|\right\}. \tag{10.24}$$

Substitute (10.19) in (10.24) to obtain

$$J_1(\mathbf{V},\mathbb{W}) = E\left\{\frac{1}{2}\sum_{l=1}^{M}\left|\int \operatorname{sgn}(D^l - p^l) - \operatorname{sgn}(\mathbf{V}_a^T\mathbf{G}^{p^l})dp^l\right|\right\}. \tag{10.25}$$

Since the integrals in (10.25) act on strictly positive or strictly negative functions, the absolute value operators and the integral operators can thus be interchanged, leading to

$$J_1(\mathbf{V},\mathbb{W}) = E\left\{\frac{1}{2}\sum_{l=1}^{M}\int |\operatorname{sgn}(D^l - p^l) - \operatorname{sgn}(\mathbf{V}_a^T\mathbf{G}^{p^l})|\,dp^l\right\}. \tag{10.26}$$

Due to the linearity of the expectation, the summation, and the integration operations, (10.26) can then be rewritten as

$$J_1(\mathbf{V},\mathbb{W}) = \frac{1}{2}\sum_{l=1}^{M}\int E\{|\operatorname{sgn}(D^l - p^l) - \operatorname{sgn}(\mathbf{V}_a^T\mathbf{G}^{p^l})|\}dp^l. \tag{10.27}$$

Furthermore, since the absolute value operators inside the expectations in (10.27) can only take values in the set $\{0,2\}$, they can be replaced by a properly scaled square operator resulting in

$$J_1(\mathbf{V},\mathbb{W}) = \frac{1}{4}\sum_{l=1}^{M}\int E\left\{(\operatorname{sgn}(D^l - p^l) - \operatorname{sgn}(\mathbf{V}_a^T\mathbf{G}^{p^l}))^2\right\}dp^l. \tag{10.28}$$

Taking the derivative of the above equation with respect to $\mathbf{V}$ results in

$$\frac{\partial}{\partial\mathbf{V}}J_1(\mathbf{V},\mathbb{W}) = -\frac{1}{2}\sum_{l=1}^{M}\int E\left\{e^{p^l}\frac{\partial}{\partial\mathbf{V}}\operatorname{sgn}(\mathbf{V}_a^T\mathbf{G}^{p^l})\right\}dp^l, \tag{10.29}$$

where $e^{p^l} = \operatorname{sgn}(D^l - p^l) - \operatorname{sgn}(\mathbf{V}_a^T\mathbf{G}^{p^l})$. For mathematical convenience, the nondifferentiable sign function is approximated by the hyperbolic tangent function

$\operatorname{sgn}(x) \approx \tanh(x) = (e^x - e^{-x})/(e^x + e^{-x})$. Since its derivative $(d/dx)\tanh(x) = \operatorname{sech}^2(x) = 4/(e^x + e^{-x})^2$, it follows that

$$\frac{\partial}{\partial \mathbf{V}} \operatorname{sgn}\left(\mathbf{V}_a^T \mathbf{G}^{p^l}\right) \approx \operatorname{sech}^2\left(\mathbf{V}_a^T \mathbf{G}^{p^l}\right) \begin{bmatrix} \operatorname{sgn}(V_1) G_1^{p^l} \\ \vdots \\ \operatorname{sgn}(V_N) G_N^{p^l} \end{bmatrix}, \tag{10.30}$$

where $G_i^{p^l} = \operatorname{sgn}(\operatorname{sgn}(V_i) Q_i^l - p^l)$ for $i = 1, \ldots, N$. Substituting (10.30) in (10.29) leads to the updates for the $V_i$:

$$\begin{aligned} V_i(n+1) &= V_i(n) + 2\mu_v \left\{ -\frac{\partial}{\partial V_i} J_1(\mathbf{V}, \mathbb{W}) \right\} \\ &= V_i(n) + \mu_v \left( \sum_{l=1}^{M} \int E\left\{ e^{p^l} \operatorname{sech}^2\left(\mathbf{V}_a^T \mathbf{G}^{p^l}\right) \operatorname{sgn}(V_i) G_i^{p^l} \right\} dp^l \right), \end{aligned} \tag{10.31}$$

where $\mu_v$ is the step-size parameter for $\mathbf{V}$ updates.

Using the instantaneous estimate for the gradient, and applying an approximation similar to the one in [2], we obtain the adaptive algorithm for the time-dependent weight vector $\mathbf{V}$ of the MWM filter as follows:

$$V_i(n+1) = V_i(n) + \mu_v \operatorname{sgn}(V_i(n)) \{ \vec{e}^T(n) \vec{G}_i^{\hat{D}}(n) \}, \tag{10.32}$$

where $\vec{G}_i^{\hat{D}} = [G_i^{\hat{D}^1} \quad \cdots \quad G_i^{\hat{D}^M}]^T$ and $G_i^{\hat{D}^l} = \operatorname{sgn}(\operatorname{sgn}(V_i) Q_i^l - \hat{D}^l)$ for $l = 1, \ldots, M$.

To derive the updates for $\mathbb{W}$, it is easy to verify that

$$\frac{\partial}{\partial W^{st}} J_1(\mathbf{V}, \mathbb{W}) \approx -\frac{1}{2} \sum_{l=1}^{M} \int E\left\{ e^{p^l} \operatorname{sech}^2\left(\mathbf{V}_a^T \mathbf{G}^{p^l}\right) \mathbf{V}_a^T \left( \frac{\partial}{\partial W^{st}} \mathbf{G}^{p^l} \right) \right\} dp^l, \tag{10.33}$$

$$\frac{\partial}{\partial W^{st}} \mathbf{G}^{p^l} \approx \begin{bmatrix} \operatorname{sech}^2\left( \operatorname{sgn}(V_1) Q_1^l - p^l \right) \operatorname{sgn}(V_1) \dfrac{\partial}{\partial W^{st}} Q_1^l \\ \vdots \\ \operatorname{sech}^2\left( \operatorname{sgn}(V_N) Q_N^l - p^l \right) \operatorname{sgn}(V_N) \dfrac{\partial}{\partial W^{st}} Q_N^l \end{bmatrix}, \tag{10.34}$$

$$\begin{aligned} \frac{\partial}{\partial W^{st}} Q_i^l &\approx -\frac{1}{2} \int \operatorname{sech}^2\left( (\vec{W}_a^l)^T \vec{S}_i^{q_i^l} \right) \frac{\partial}{\partial W^{st}} \left( (\vec{W}_a^l)^T \vec{S}_i^{q_i^l} \right) dq_i^l \\ &\approx \begin{cases} \operatorname{sgn}(W^{st}) \operatorname{sgn}\left( \operatorname{sgn}(W^{st}) X_i^s - q_i^t \right), & l = t, \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{10.35}$$

Initialization

$\mathbf{V} = \mathbf{1}$; $\mathbb{W} = \mathbb{I}$

For Loop

For $i = 1, \ldots, N$ and $l = 1, \ldots, M$

$$Q_i^l = \text{MEDIAN}\left(|W^{jl}| \diamond \text{sgn}(W^{jl}) X_i^j \,|_{j=1}^{M}\right)$$

$$\hat{\vec{D}}(n) = \begin{bmatrix} \text{MEDIAN}\left(|V_i| \diamond \text{sgn}(V_i) Q_i^1 \,|_{i=1}^{N}\right) \\ \vdots \\ \text{MEDIAN}\left(|V_i| \diamond \text{sgn}(V_i) Q_i^M \,|_{i=1}^{N}\right) \end{bmatrix}$$

$$\vec{e}(n) = \vec{D}(n) - \hat{\vec{D}}(n)$$

For $i = 1, \ldots, N$

$$V_i(n+1) = V_i(n) + \mu_v \,\text{sgn}(V_i(n)) \{\vec{e}^T(n) \vec{G}_i^{\hat{D}}(n)\}$$

where $\vec{G}_i^{\hat{D}} = \left[\text{sgn}\left(\text{sgn}(V_i) Q_i^l - \hat{D}^l\right) |_{l=1}^{M}\right]^T$

For $s, t = 1, \ldots, M$

$$W^{st}(n+1) = W^{st}(n) + \mu_w \,\text{sgn}\left(W^{st}(n)\right) e^t(n) \{\mathbf{V}^T(n+1) \mathbf{A}^{st}(n)\}$$

where $\mathbf{A}^{st} = \left[\delta\left(\text{sgn}(V_i) Q_i^t - \hat{D}^t\right) \text{sgn}\left(\text{sgn}(W^{st}) X_i^s - Q_i^t\right) |_{i=1}^{N}\right]^T$

Algorithm 10.1. Summary of the LMA algorithm for the MWM filter I.

Notice that in (10.35), the derivative that introduces one more $\text{sech}^2$ term is omitted since it is insignificant compared to the other one.

After some mathematical manipulations and similar arguments as in [2], the adaptive algorithm for the cross-channel weight matrix $\mathbb{W}$ can be simplified as follows:

$$W^{st}(n+1) = W^{st}(n) + \mu_w \,\text{sgn}\left(W^{st}(n)\right) e^t(n) \{\mathbf{V}^T(n) \mathbf{A}^{st}(n)\}, \tag{10.36}$$

where $\mathbf{A}^{st} = [A_1^{st} \quad A_2^{st} \quad \cdots \quad A_N^{st}]^T$, and $A_i^{st} = \delta(\text{sgn}(V_i) Q_i^t - \hat{D}^t)\, \text{sgn}(\text{sgn}(W^{st}) X_i^s - Q_i^t)$ for $i = 1, \ldots, N$, where $\delta(x) = 1$ for $x = 0$ and $\delta(x) = 0$ otherwise. $\mu_w$ is the step-size parameter for $\mathbb{W}$ updates. Algorithm 10.1 summarizes the LMA algorithm for the MWM filter I.

One note on the initialization of the weight matrix $\mathbb{W}$. Although it is set to be the identity matrix in Algorithm 10.1, in simulations each element should be added a small value so that the off-diagonal elements can be updated through (10.36), otherwise they will be held 0 all the time. This is also true for Algorithm 10.2.

### 10.3.2. Optimization for the MWM filter II

The optimization process for the second MWM filtering structure is very similar to the one shown above. Assume that the time-dependent weight vector and the

Initialization

$\mathbf{V} = \mathbf{1}$; $\mathbb{W} = \mathbb{I}$

For Loop

For $i = 1, \ldots, N$ and $l = 1, \ldots, M$

$$Q_i^l = \text{MEDIAN}\left( |W^{jl}| \diamond \text{sgn}\left(W^{jl}\right) X_i^j \,|_{j=1}^{M} \right)$$

$$\hat{\vec{D}}(n) = \begin{bmatrix} \text{MEDIAN}\left( |V_i^1| \diamond \text{sgn}\left(V_i^1\right) Q_i^1 \,|_{i=1}^{N} \right) \\ \vdots \\ \text{MEDIAN}\left( |V_i^M| \diamond \text{sgn}\left(V_i^M\right) Q_i^M \,|_{i=1}^{N} \right) \end{bmatrix}$$

$$\vec{e}(n) = \vec{D}(n) - \hat{\vec{D}}(n)$$

For $i = 1, \ldots, N$ and $t = 1, \ldots, M$

$$V_i^t(n+1) = V_i^t(n) + \mu_v \,\text{sgn}\left(V_i^t(n)\right) e^t(n) G_i^{\widehat{D}^t}(n)$$

where $G_i^{\widehat{D}^t} = \text{sgn}\left(\text{sgn}\left(V_i^t\right) Q_i^t - \widehat{D}^t\right)$

For $s, t = 1, \ldots, M$

$$W^{st}(n+1) = W^{st}(n) + \mu_w \,\text{sgn}\left(W^{st}(n)\right) e^t(n) \left\{ \left(\mathbf{V}^t\right)^T (n+1) \mathbf{A}^{st}(n) \right\}$$

where $\mathbf{A}^{st} = \left[ \delta\left(\text{sgn}\left(V_i^t\right) Q_i^t - \hat{D}^t\right) \text{sgn}\left(\text{sgn}\left(W^{st}\right) X_i^s - Q_i^t\right) |_{i=1}^{N} \right]^T$

Algorithm 10.2. Summary of the LMA algorithm for the MWM filter II.

cross-channel weight matrix are

$$\mathbf{V} = \begin{bmatrix} \vec{V}_1 \\ \vdots \\ \vec{V}_N \end{bmatrix}, \qquad \mathbb{W} = \begin{bmatrix} W^{11} & \cdots & W^{1M} \\ \vdots & \ddots & \vdots \\ W^{M1} & \cdots & W^{MM} \end{bmatrix}, \tag{10.37}$$

where $\vec{V}_i = [V_i^1 \quad V_i^2 \quad \cdots \quad V_i^M]^T$.

If $Q_i^l$ and $\vec{S}_i^{\,l}$ are defined as in the previous case, the output of the filter can be written as

$$\hat{\vec{D}} = \left[ \hat{D}^1 \quad \hat{D}^2 \quad \cdots \quad \hat{D}^M \right]^T, \tag{10.38}$$

where for $l = 1, \ldots, M$,

$$\begin{aligned} \hat{D}^l &= \text{MEDIAN}\left( |V_i^l| \diamond \text{sgn}\left(V_i^l\right) Q_i^l \,|_{i=1}^{N} \right) \\ &= \frac{1}{2} \int \text{MEDIAN}\left( |V_i^l| \diamond \text{sgn}\left(V_i^l\right) Q_i^l \,|_{i=1}^{N} \right) dp^l \\ &= \frac{1}{2} \int \text{sgn}\left( \left(\mathbf{V}_a^l\right)^T \mathbf{G}^{p^l} \right) dp^l, \end{aligned} \tag{10.39}$$

where

$$\begin{aligned}\mathbf{V}_a^l &= \left[\,|\,V_1^l\,| \quad |\,V_2^l\,| \quad \cdots \quad |\,V_N^l\,|\,\right]^T,\\ \mathbf{G}^{p^l} &= \left[\operatorname{sgn}\left(\operatorname{sgn}\left(V_1^l\right)Q_1^l - p^l\right) \quad \cdots \quad \operatorname{sgn}\left(\operatorname{sgn}\left(V_N^l\right)Q_N^l - p^l\right)\right]^T.\end{aligned} \tag{10.40}$$

Under the least mean absolute (LMA) criterion, the cost function to minimize will be just like (10.28).

Taking the derivative of the above equation with respect to a particular $V_i^t$ and using similar approximations to the ones used in the previous case results in

$$\frac{\partial}{\partial V_i^t} J_1(\mathbf{V},\mathbb{W}) = -\frac{1}{2}\int E\left\{e^{p^t}\operatorname{sech}^2\left(\left(\mathbf{V}_a^l\right)^T\mathbf{G}^{p^t}\right)\operatorname{sgn}\left(V_i^t\right)G_i^{p^t}\right\}dp^t, \tag{10.41}$$

where $e^{p^t} = \operatorname{sgn}(D^t - p^t) - \operatorname{sgn}((\mathbf{V}_a^l)^T\mathbf{G}^{p^t})$. Using instantaneous estimates for the expectation the updates for $\mathbf{V}$ result in

$$\begin{aligned}V_i^t(n+1) &= V_i^t(n) + \mu_v \operatorname{sgn}\left(V_i^t(n)\right)e^t(n)\operatorname{sgn}\left(\operatorname{sgn}\left(V_i^t(n)\right)Q_i^t(n) - \widehat{D^t}(n)\right)\\ &= V_i^t(n) + \mu_v \operatorname{sgn}\left(V_i^t(n)\right)e^t(n)G_i^{\widehat{D^t}}(n).\end{aligned} \tag{10.42}$$

On the other hand, the updates for $\mathbb{W}$ are given by

$$W^{st}(n+1) = W^{st}(n) + \mu_w \operatorname{sgn}\left(W^{st}(n)\right)e^t(n)\left\{\left(\mathbf{V}^t\right)^T(n)\mathbf{A}^{st}(n)\right\} \tag{10.43}$$

that is basically the same as (10.36) with the only difference that every $V_i$ is replaced by $V_i^t$. The LMA algorithm for the MWM filter II is summarized in Algorithm 10.2.

## 10.4. Complex multichannel WMs and their optimization

### 10.4.1. Complex weighted median

The weighting strategy is essential to filtering operations. Many communications-related applications, such as matched filtering, equalization, beamforming, and so forth, require filtering structures admitting complex-valued weights. For linear filters, there is no difficulties in obtaining the optimal weights. However, due to the nonlinear nature of the median operation, the optimal complex weight design for median-type filters has not been explored until the paper [11], and even the meaning of the complex weighting itself is vague.

The simplest approach to attain complex WM filtering is to perform marginal operations directly, where the real component of the weights $\{W_R|_{i=1}^N\}$ affects the

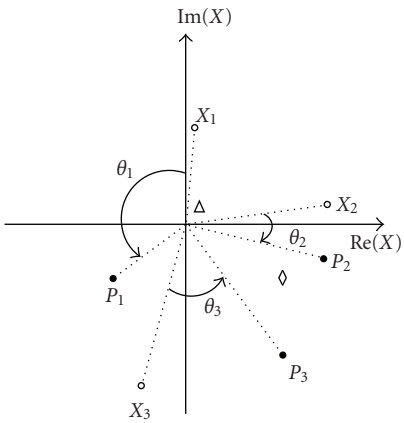



FIGURE 10.1. Marginal phase-coupled CWM illustration, "$\circ$": original samples, "$\cdot$": phase-coupled samples, "$\triangle$": marginal median output, "$\diamondsuit$": marginal phase-coupled median output.

real part of the samples $\{X_R|_{i=1}^N\}$ and the imaginary component of the weights $\{W_I|_{i=1}^N\}$ affects the imaginary part of the samples $\{X_I|_{i=1}^N\}$. This approach referred to as *marginal* complex WM filter outputs:

$$\begin{aligned}\hat{\beta}_{\text{marginal}} = {} & \text{MEDIAN}\left(|W_{R_i}| \diamondsuit \operatorname{sgn}(W_{R_i}) X_{R_i}|_{i=1}^N\right) \\ & + j\,\text{MEDIAN}\left(|W_{I_i}| \diamondsuit \operatorname{sgn}(W_{I_i}) X_{I_i}|_{i=1}^N\right).\end{aligned} \tag{10.44}$$

The definition in (10.44) assumes that the real and imaginary components of the input samples are independent. On the other hand, if the real and imaginary domains are correlated, better performance is attainable by mutually *coupling* the real and imaginary components of the signal and weights. According to [11], the complex weighted median we will refer to throughout the chapter is defined in the following manner:

$$\begin{aligned}\tilde{\beta} = {} & \text{MEDIAN}\left(|W_i| \diamondsuit \operatorname{Re}\{\Phi(W_i^*) X_i\}|_{i=1}^N\right) \\ & + j\,\text{MEDIAN}\left(|W_i| \diamondsuit \operatorname{Im}\{\Phi(W_i^*) X_i\}|_{i=1}^N\right),\end{aligned} \tag{10.45}$$

where $\operatorname{Re}\{\cdot\}$ and $\operatorname{Im}\{\cdot\}$ denote real and imaginary parts, respectively, $|W_i|$ represents the modulus of $W_i$, $\Phi(W_i)$ represents the unit direction vector $e^{j\angle W_i}$. The operation that the phases of the complex weights are incorporated into samples as in $\Phi(W_i^*)X_i$ is called phase coupling.

To help understand this definition better, a simple example is given in Figure 10.1. Three complex-valued samples $X_1, X_2, X_3$ and three complex-valued weights

on the unit circle $W_1$, $W_2$, $W_3$ are arbitrarily chosen. The phase-coupled samples $P_1$, $P_2$, $P_3$ are plotted to show the effect of phase coupling. The weights are not directly shown in the figure, but their phases $\theta_1$, $\theta_2$, $\theta_3$ are shown as the angles between the original and altered samples.

Due to the rotation of the samples, the real and imaginary parts of the signal are coupled together before further processing, hence the intrinsic correlation in the signal is preserved. In addition, the complex WM in (10.45) clearly has a structural advantage over the marginal complex WM in (10.44). In the above example, the real part of the marginal complex WM is confined to be chosen from $\{X_{R_i}|_{i=1}^3, -X_{R_i}|_{i=1}^3\}$; similarly, the imaginary part is confined to be chosen from $\{X_{I_i}|_{i=1}^3, -X_{I_i}|_{i=1}^3\}$. While on the other hand, the output of the complex WM can be any point on the complex plane inside a circle whose radius is the largest magnitude among all three samples. This enables the capability of designing a more powerful robust filter.

### 10.4.2. Complex differentiation

The previous subsection solves the problem of how to mathematically represent the median filter in the complex domain. However in order to obtain an optimal design of such filters, complex differentiation over this mathematical expression has to be executed so that the gradient technique can be used to attain desired filter configuration. Moreover, due to phase coupling, this complex differentiation incurs even more difficulties. To better understand the properties of the complex weighted median, and to develop its optimization algorithm, the differentiation of a complex function of $f(z) = |z|^2$ kind has to be exercised. Unfortunately in complex analysis, function $f(z)$ is differentiable only at the origin. It does not hold the Cauchy-Riemann equations and thus is not analytic. In other words, it has no derivative in the strict complex sense. To obviate this mathematical difficulty, we adopt the so-called complex differential operators [6].

For $z \in \mathcal{C}$, $x, y \in \mathcal{R}$, denote

$$z = x + jy, \qquad z^* = x - jy. \tag{10.46}$$

Two complex differential operators in Cartesian coordinates are defined as

$$\partial_z = \frac{1}{2}\left(\frac{\partial}{\partial x} - j\frac{\partial}{\partial y}\right), \qquad \partial_{z^*} = \frac{1}{2}\left(\frac{\partial}{\partial x} + j\frac{\partial}{\partial y}\right). \tag{10.47}$$

One immediate derivation of these operators would be

$$\begin{aligned} \partial_z(z) &= 1, \qquad \partial_z(z^*) = 0, \\ \partial_{z^*}(z) &= 0, \qquad \partial_{z^*}(z^*) = 1. \end{aligned} \tag{10.48}$$

A couple of properties which will be extensively used in later optimal filter derivation are listed here without proof (see [15] for details):

$$\begin{aligned}
\partial_z |z|^2 &= \partial_z |z^*|^2 = z^*, \\
\partial_{z^*} |z^*|^2 &= \partial_{z^*} |z|^2 = z, \\
\partial_z |z| &= \partial_z |z^*| = \frac{1}{2}\Phi(z^*), \\
\partial_{z^*} |z^*| &= \partial_{z^*} |z| = \frac{1}{2}\Phi(z), \\
\partial_z \Phi(z) &= \frac{1}{2|z|}, \qquad \partial_{z^*} \Phi(z^*) = \frac{1}{2|z^*|}, \\
\partial_{z^*} \Phi(z) &= -\frac{1}{2|z|}\Phi(z^2), \qquad \partial_z \Phi(z^*) = -\frac{1}{2|z^*|}\Phi(z^{*2}).
\end{aligned} \tag{10.49}$$

Since the complex differential operators are defined only by the partial derivatives over $x$ and $y$, it greatly relaxes the requirements for differentiability of the given complex function.

### 10.4.3. Multichannel WM filters in the complex domain

Given the filter input vector at time $n$ is

$$\mathbf{X}(n) = \left[\vec{X}_1(n) \quad \vec{X}_2(n) \quad \cdots \quad \vec{X}_N(n)\right]^T, \tag{10.50}$$

where $\vec{X}_i(n) = [X_i^1(n) \quad X_i^2(n) \quad \cdots \quad X_i^M(n)]^T$ and each element of $\vec{X}_i(n)$ is complex. Also, the time/spatial-dependent weight vector and the cross-channel weight matrix are

$$\mathbf{V} = \left[\vec{V}_1 \quad \cdots \quad \vec{V}_N\right], \qquad \mathbb{W} = \begin{bmatrix} W^{11} & \cdots & W^{1M} \\ \vdots & \ddots & \vdots \\ W^{M1} & \cdots & W^{MM} \end{bmatrix}, \tag{10.51}$$

where $\vec{V}_i = [V_i^1 \quad V_i^2 \quad \cdots \quad V_i^M]^T$, and each element of them is complex-valued too. Then applying the complex weighted median definition in (10.45), the multichannel WM filter II in (10.14) can be extended to complex domain as the following:

$$\vec{Y} = \left[\begin{array}{l} \text{MEDIAN}\left(\left|\left(V_i^l\right)^*\right| \diamondsuit \operatorname{Re}\left\{P_i^l\right\}\Big|_{i=1}^N\right) \\ \quad + j\,\text{MEDIAN}\left(\left|\left(V_i^l\right)^*\right| \diamondsuit \operatorname{Im}\left\{P_i^l\right\}\Big|_{i=1}^N\right) \end{array}\Bigg|_{l=1}^M\right]^T, \tag{10.52}$$

where $P_i^l = \Phi((V_i^l)^*)Q_i^l$. The intermediate variable $Q_i^l$ is defined by another complex weighted median

$$\begin{aligned} Q_i^l = {} & \text{MEDIAN}\left(\left|\,(W^{jl})^*\,\right| \diamondsuit \text{Re}\,\{S_i^{jl}\}\,\Big|_{i=1}^{N}\right) \\ & + j\,\text{MEDIAN}\left(\left|\,(W^{jl})^*\,\right| \diamondsuit \text{Im}\,\{S_i^{jl}\}\,\Big|_{i=1}^{N}\right), \end{aligned} \tag{10.53}$$

where $S_i^{jl} = \Phi((W^{jl})^*)X_i^l$.

If we consider the complex MWM filter I as a special case of (10.52) where $V_i^l = V_i$ for all $l$, then the complex version of (10.11) can be expressed as

$$\vec{Y} = \left[\begin{array}{l} \text{MEDIAN}\left(\left|\,V_i^*\,\right| \diamondsuit \text{Re}\,\{P_i^l\}\,\Big|_{i=1}^{N}\right) \\ + j\,\text{MEDIAN}\left(\left|\,V_i^*\,\right| \diamondsuit \text{Im}\,\{P_i^l\}\,\Big|_{i=1}^{N}\right) \end{array}\Bigg|_{l=1}^{M}\right]^T, \tag{10.54}$$

where $P_i^l = \Phi(V_i^*)Q_i^l$ and $Q_i^l$ is the same as in (10.53).

### 10.4.4. Filter optimization

Assume that the observed process $\vec{X}(n)$ is statistically related to a desired multichannel complex process $\vec{D}(n) = [D^1(n) \quad D^2(n) \quad \cdots \quad D^M(n)]^T$ of interest, typically considered a transformed or corrupted version of $\vec{D}(n)$. Denote $\vec{e} = \vec{D} - \vec{Y}$ as the multivariate error signal, $e^l = D^l - Y^l$ its $l$th element.

Under the least mean absolute (LMA) criterion, the cost function to minimize is

$$J_1(\mathbf{V}, \mathbb{W}) = E\{||\vec{D} - \vec{Y}||_1\} = E\left\{\sum_{l=1}^{M} |\,D^l - Y^l\,|\right\}. \tag{10.55}$$

After numerous mathematical manipulations and similar arguments as in [11], the updates for $\mathbf{V}$ are given by

$$\begin{aligned} V_i^t(n+1) = {} & V_i^t(n) + \mu_v \Phi(V_i^t(n)) \Big\{ e_R^t \,\text{sgn}\,(P_{R_i}^t - Y_R^t) + e_I^t \,\text{sgn}\,(P_{I_i}^t - Y_I^t) \\ & + 2je_R^t P_{I_i}^t \delta(P_{R_i}^t - Y_R^t) - 2je_I^t P_{R_i}^t \delta(P_{I_i}^t - Y_I^t) \Big\} \end{aligned} \tag{10.56}$$

for $s = 1, \ldots, N$, $t = 1, \ldots, M$. For $s, t = 1, \ldots, M$, we have

$$\begin{aligned} W^{st}(n+1) = {} & W^{st}(n) + \mu_w \Phi(W^{st}(n)) \\ & \times \left\{ e_R^t \sum_{i=1}^{N} ((V_i^t)^* A_i^{st} + V_i^t B_i^{st}) \delta(P_{R_i}^t - Y_R^t) \right. \\ & \left. - je_I^t \sum_{i=1}^{N} ((V_i^t)^* A_i^{st} - V_i^t B_i^{st}) \delta(P_{I_i}^t - Y_I^t) \right\}, \end{aligned} \tag{10.57}$$

where

$$A_i^{st} = \operatorname{sgn}\left(S_{R_i}^{st} - Q_{R_i}^{t}\right) + j2S_{I_i}^{st}\delta\left(S_{R_i}^{st} - Q_{R_i}^{t}\right) + j\operatorname{sgn}\left(S_{I_i}^{st} - Q_{I_i}^{t}\right) + 2S_{R_i}^{st}\delta\left(S_{I_i}^{st} - Q_{I_i}^{t}\right), \tag{10.58}$$

$$B_i^{st} = \operatorname{sgn}\left(S_{R_i}^{st} - Q_{R_i}^{t}\right) + j2S_{I_i}^{st}\delta\left(S_{R_i}^{st} - Q_{R_i}^{t}\right) - j\operatorname{sgn}\left(S_{I_i}^{st} - Q_{I_i}^{t}\right) - 2S_{R_i}^{st}\delta\left(S_{I_i}^{st} - Q_{I_i}^{t}\right). \tag{10.59}$$

As a special case of the complex MWM filter II, the optimization for the complex MWM filter I can be easily derived as

$$\begin{aligned}
V_i(n+1) &= V_i(n) \\
&\quad + \mu_v \Phi\left(V_i(n)\right) \sum_{l=1}^{M} \Big\{ e_R^l\left(\operatorname{sgn}\left(P_{R_i}^l - Y_R^l\right)\right) + e_I^l\left(\operatorname{sgn}\left(P_{I_i}^l - Y_I^l\right)\right) \\
&\qquad + 2je_R^l P_{I_i}^l \delta\left(P_{R_i}^l - Y_R^l\right) - 2je_I^l P_{R_i}^l \delta\left(P_{I_i}^l - Y_I^l\right) \Big\}, \\
W^{st}(n+1) &= W^{st}(n) + \mu_w \Phi\left(W^{st}(n)\right) \\
&\quad \times \left\{ e_R^t \sum_{i=1}^{N} \left(V_i^* A_i^{st} + V_i B_i^{st}\right) \delta\left(P_{R_i}^t - Y_R^t\right) \right. \\
&\qquad \left. - je_I^t \sum_{i=1}^{N} \left(V_i^* A_i^{st} - V_i B_i^{st}\right) \delta\left(P_{I_i}^t - Y_I^t\right) \right\},
\end{aligned} \tag{10.60}$$

where $A_i^{st}$ and $B_i^{st}$ are the same as in (10.58).

## 10.5. Simulations

### 10.5.1. Real-valued signals

In order to evaluate the performance improvement of the described filter structures in the previous sections over the existing approaches, two types of simulations are executed. A color image denoising is set up mainly to test the validity and efficiency of the MWM filter I, while a three-sensor array processing is formulated to focus on the performance of the MWM filter II.

First, a RGB color image contaminated with 20% independent salt-and-pepper noise in each channel is processed by six multichannel filters, including the marginal vector median filter (MARG) where three univariate weighted median filters run through their correspondent channels independently, the WVM filter which uses the greedy algorithm summarized in [25], the multichannel linear filter I (MLI) in (10.10), the MWM filter I and II developed in Section 10.2, and the CMWM filter expressed in (10.6). The observation window is set to $3 \times 3$. The optimal weights for each filter are obtained by running the filters through a training image with similar noise contamination. The uncorrupted version of the training

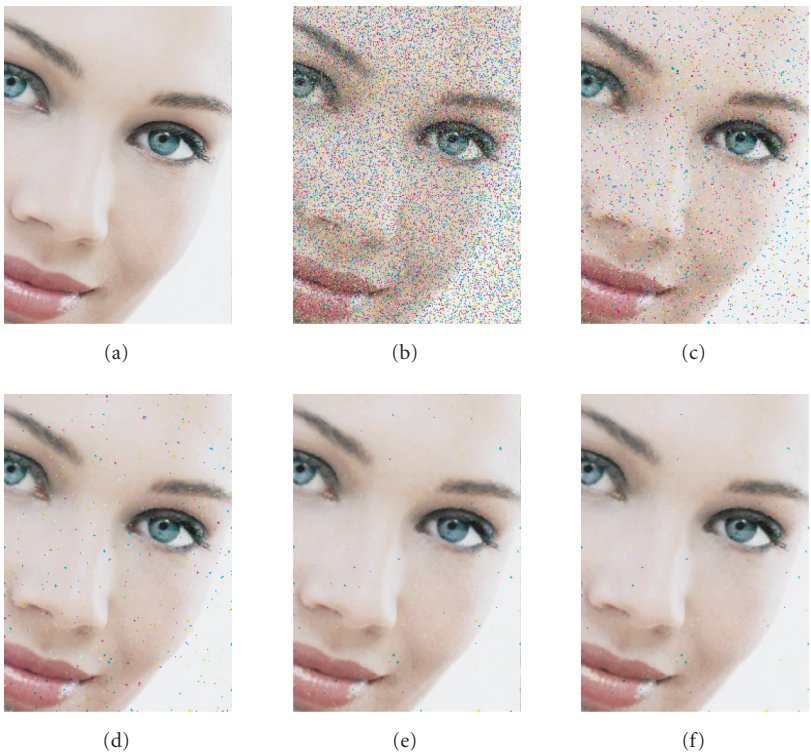



FIGURE 10.2. Multichannel medians for color image denoising in salt-and-pepper noise, $3 \times 3$ window, $\mu_v, \mu_w = 0.001$ for MWMI, and $\mu = 0.01$ for WVM, MARG, and CMWM. (a) Noiseless image, (b) contaminated image, (c) WVM, (d) marginal, (e) MWMI, and (f) CMWM.

image is known and is used as the desired signal. In the simulation, we choose this training image to be a $150 \times 150$ subsection of the test image for all filters. To guarantee that the weight training for every filter converges, the step-size parameter (or parameters in the two MWM cases) for each filter has to be cautiously set. The resulting weights are then passed to the corresponding filters to denoise the entire image. The outputs of the four median-related filters are depicted in Figure 10.2. The output of the MWMII filter is not shown because it closely resembles that of MWMI. The result of the MLI filter is also not shown due to its blurry quality and loss of details which obviously is the direct consequence of averaging over outliers.

As a measure of the effectiveness of the filters, the mean absolute error of the outputs was calculated for each filter. Peak signal-to-noise ratio (PSNR) was also used to evaluate the fidelity of the filtered images. The results of all six filters are summarized in Table 10.2 along with their corresponding number of weights.

The visual outputs in Figure 10.2 and the performance metrics in Table 10.2 show that in general as the number of weights used by a filter structure increases, the respective filter output performs better in terms of MAE, PSNR, and visual

Table 10.2. Multichannel median filters comparison on color image denoising.

| Filter | No. of weights | MAE | PSNR (dB) |
|---|---|---|---|
| Noisy signal | — | 0.2970 | 11.74 |
| WVM | 9 | 0.1231 | 17.89 |
| MARG | 27 | 0.0343 | 25.89 |
| MLI | 18 | 0.2740 | 19.27 |
| MWMI | 18 | 0.0359 | 29.51 |
| MWMII | 36 | 0.0359 | 29.51 |
| CMWM | 81 | 0.0305 | 30.80 |

quality. As expected, the MLI filter performs the worst both in measurements and visual perception, simply because the linear structure is susceptible to outliers. Among the five median-related multichannel filters, the WVM filter has the least number of weights and its performance is the poorest. On the other hand, the CMWM filter shows its superiority in both categories since it has the most number of weights. However, due to the efficient structural characterization of the strong cross-channel correlation of the signal, the MWMI filter can combine useful information from other channels into its filtering process, while in MARG such information is completely ignored. As a result, MWMI outperforms the marginal vector median filter with higher PSNR, visually less unfiltered outliers, and comparable MAE even though it has one third less weights than MARG. Thus, MWMI will always perform better than marginal filtering unless the signals from subchannels are totally uncorrelated.

Table 10.3 compares the optimized weights obtained from MWMI and II to distinquish the two structures. The cross-channel weight matrices $\mathbb{W}$ are column-wise normalized with the magnitude of the diagonal element in that column. This is done to emphasize how information from other channels are weighted into the filtering process. The space-dependent weight vector $\mathbf{V}$ in MWMI case is normalized with its biggest element. In MWMII case, the normalization is done within each channel. All these space-dependent weights are not shown in their original $3 \times 3$ matrix form for convenience of tabulation.

It is not surprising that the two cross-channel weight matrices are almost the same, since they all reflect the intrinsic cross-channel correlation structure of the color image. The weight vector $\mathbf{V}$ on the other hand, reflects the spatial correlation within each channel. Due to the special characteristics of color images, the spatial correlation within subchannels are very much alike, hence the three columns shown in Table 10.3 under MWMII category are very similar. If they are replaced by one of the columns, the MWMII filter reduces to another implementation of a MWMI filter which differs with the one in the table only slightly on the weighting of the center sample. The filter output thus will not show much difference. This explains why the MWMII filter will not gain anything from the MWMI filter in color imaging applications. Lastly, since MWMI and II are just slightly behind CMWM on measurements and visual perception yet they have significantly less weights,

Table 10.3. Comparison on optimized weights of MWMI and MWMII.

| | MWMI | | | MWMII | | |
|---|---|---|---|---|---|---|
| | 1.0000 | 0.9627 | 0.3232 | 1.0000 | 0.9624 | 0.3230 |
| $\mathbb{W}$ | 0.6656 | 1.0000 | 0.6754 | 0.6655 | 1.0000 | 0.6756 |
| | 0.3340 | 1.0071 | 1.0000 | 0.3341 | 1.0071 | 1.0000 |
| | | | | ch 1 | ch 2 | ch 3 |
| | | 0.6936 | | 0.8886 | 0.8752 | 0.8562 |
| | | 0.6988 | | 0.8963 | 0.8772 | 0.8523 |
| | | 0.6593 | | 0.8825 | 0.8647 | 0.8302 |
| | | 0.7280 | | 0.8741 | 0.8907 | 0.8973 |
| $\mathbf{V}$ | | 1.0000 | | 1.0000 | 1.0000 | 1.0000 |
| | | 0.7127 | | 0.8831 | 0.8918 | 0.8680 |
| | | 0.6622 | | 0.8327 | 0.8674 | 0.8806 |
| | | 0.6911 | | 0.8468 | 0.8825 | 0.8898 |
| | | 0.6701 | | 0.8409 | 0.8765 | 0.8726 |

Table 10.4. MAE of the output signals on array processing. (Reproduced from Y. Li, J. B. Rodriguez, and G. R. Arce, "Weighted median filters for multichannel signals," IEEE Transactions on Signal Processing © 2006 IEEE.)

| Filter | MAE |
|---|---|
| Noisy signal | 3.8031 |
| WVM | 1.7998 |
| MWMI | 1.4752 |
| MLII | $\infty$ |
| MARG | 0.6417 |
| MWMII | 0.6178 |
| CMWM | 0.4227 |

they can be considered as promising alternatives when excessive weighting is not suitable.

To test the effectiveness of the MWM filter II, a simple array processing problem with real-valued signals is used. The system shown in Figure 10.3 is implemented. It consists of a 3-element array and 3 sources in the farfield of the array transmitting from different directions and at different frequencies as indicated in the figure. The goal is to separate the signals from all sources using the receiver array in the presence of independent alpha stable noise.

In order to do so, WVM, MARG, MLII, MWMI and II, and CMWM are put to test all with a window size of 25. The filters are optimized using the techniques described earlier in this section, with a reference signal whose components are noiseless versions of the signals received at the sensors. The results obtained are summarized in Figure 10.4 and Table 10.4.

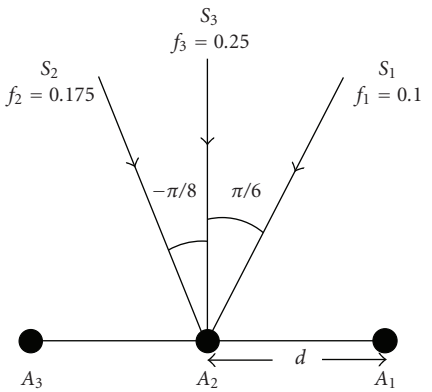



FIGURE 10.3. Three-element sensor array used for the simulation. Frequencies are normalized, and $d = \lambda_3/4$. (Reproduced from Y. Li, J. B. Rodriguez, and G. R. Arce, "Weighted median filters for multichannel signals," IEEE Transactions on Signal Processing © 2006 IEEE.)

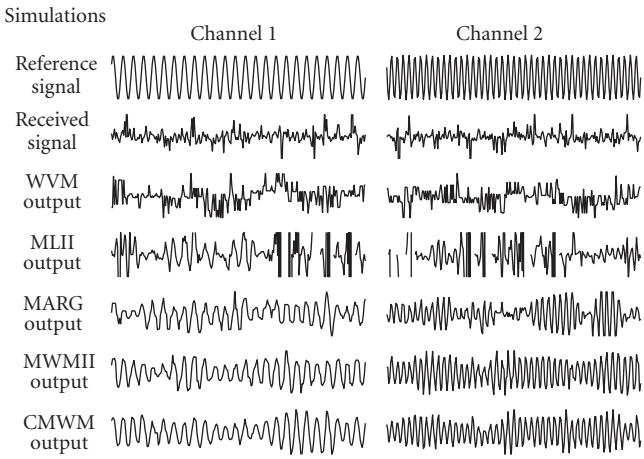



FIGURE 10.4. Input and output signals for array processing with multichannel medians. Each column corresponds to a channel (only channel one and two are shown). The output of MWMI is not included in the figure. To better illustrate, MARG, MWMII, and CMWM are plotted using the same scale, others are not. (Reproduced from Y. Li, J. B. Rodriguez, and G. R. Arce, "Weighted median filters for multichannel signals," IEEE Transactions on Signal Processing © 2006 IEEE.)

To extract different frequency components from the same multichannel signal, different time weighting structures should be applied at different channels. As clearly shown in Figure 10.4, MARG, MWMII, and CMWM perform fairly successfully in retrieving the desired signals in strongly impulsive noise, since they all have enough time-dependent weights for each frequency component. WVM and MWMI on the other hand are unable to do so because they only have one set of time-dependent weights and cannot reflect the time correlation structure of

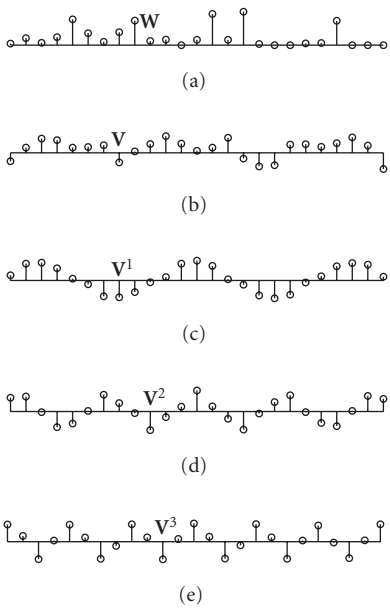



FIGURE 10.5. Optimized weights for the multichannel medians in the array processing example: (a) **W** from WVM, (b) **V** from MWMI, (c) channel 1 of **V** from MWMII, (d) channel 2 of **V**, and (e) channel 3 of **V**. (Reproduced from Y. Li, J. B. Rodriguez, and G. R. Arce, "Weighted median filters for multichannel signals," IEEE Transactions on Signal Processing © 2006 IEEE.)

three frequency components at the same time. In consequence, their outputs bear no resemblance of the desired signals as seen in Figure 10.4. MLII gets diverged very often due to the impulsiveness of the noise even with weight reset mechanism. MWMII outperforms MARG and closely approaches the performance of CMWM which is the best in the simulation. The improvement from marginal filtering to MWMII filtering is clearly achieved by combining signals from other channels. The final weights from several filters of interest are shown in Figure 10.5 and Table 10.5.

Figure 10.5(a) shows why the WVM filter is not able to obtain a good result for this problem. The optimal weights are not reflecting any frequency component in the signal due to the inadequate of the structure. A similar conclusion can be reached for the weights of the MWMI filter in Figure 10.5(b). The outer weights of the MWM filter II are shown in Figures 10.5(c)–10.5(e), each one corresponding to a different channel of the signals. It can be seen how the weights show a certain periodicity with frequencies related to the ones of the signals we want to extract in each channel.

The inner weights for the MWM filter I and the MWM filter II are shown in Table 10.5. The normalization of these weight matrices is done the same way as in Table 10.3. Notice that the signs of these weights are not normalized out because they are responsible for modifying corresponding samples.

Table 10.5. Optimized inner weights $\mathbb{W}$ from MWMI and MWMII.

| MWMI | | | MWMII | | |
|---|---|---|---|---|---|
| 1.0000 | −0.6989 | −2.3043 | 1.0000 | 0.6634 | 0.6013 |
| 1.0787 | 1.0000 | −2.3760 | 0.8098 | 1.0000 | 0.4499 |
| −0.1263 | −0.3402 | −1.0000 | 1.1263 | 0.6392 | 1.0000 |

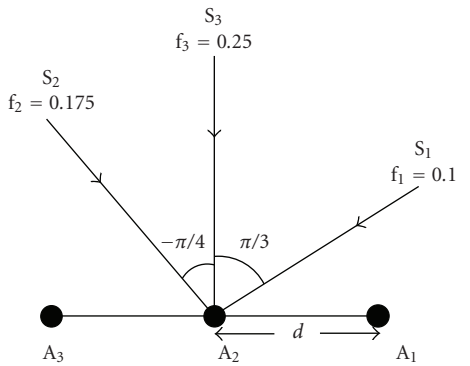



Figure 10.6. Three-element sensor array used for the simulation. Frequencies are normalized, and $d = \lambda_3/2$.

It can be seen that the weight matrix from MWMII better reflects the cross-channel correlation of the signals while the weight matrix from MWMI is kind of eccentric mainly because of its inadequacy in structure. When the cross-channel weight matrix is fixed to be an identity matrix which means the subchannels are totally independent of each other, the MWM filter II reduces to a marginal vector median filter. Thus given the same noise environment, the more correlated the subchannels are, the better MWMII will perform.

### 10.5.2. Complex-valued signals

To test the effectiveness of the complex MWM filter II, a simple array processing problem with complex-valued signals is used. The system shown in Figure 10.6 is implemented. The system consists of a 3-element array and 3 sources in the farfield of the array transmitting from different directions and at different frequencies. The goal is to separate the signals from all sources using the array in the presence of alpha stable noise.

A complex marginal vector median filter (MARGc) where $M$ univariate complex weighted median filters run through their correspondent channels independently, and the forementioned complex multichannel WM filter II (MWMIIc) are put to test all with a window size of 25. The filters are optimized with a reference signal whose components are noiseless versions of the signals received at the sensors.

The results obtained are summarized in Figure 10.7 and Table 10.6.

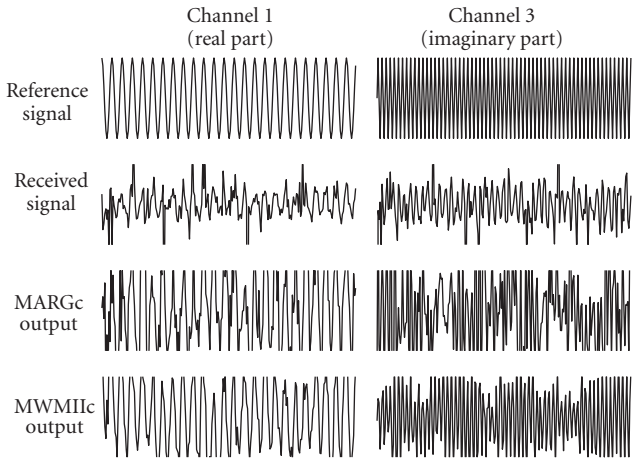



FIGURE 10.7. Input and output signals for array processing with multivariate medians. Each column corresponds to a channel (only the real part of channel one and the imaginary part of channel three are shown).

TABLE 10.6. Average MAE of the output signals.

| Filter | MAE |
|---|---|
| Noisy signal | 29.6263 |
| MARG | 2.5947 |
| MWMIIc | 1.6636 |

Figure 10.7 shows that MWMIIc outperforms MARGc and the improvement is achieved mainly because it combines signals from other channels into filtering.

A considerable improvement MWMIIc has over MWMII is that, since each single complex weight in the cross-channel weight matrix $\mathbb{W}$ has the capability of compensating the phase discrepancy between two channels, MWMIIc theoretically has no limitation on the incident angles. On the contrary, MWMII has to rely on the magnitude similarity (or correlation) between channels to gain performance boost over marginal processing. Thus the narrower the incident angles are, the better the performance will be. Compare Figures 10.3 and 10.6 to see the difference.

## 10.6. Summary

Using the concept of the multivariate maximum likelihood estimation of location, two new multivariate weighting structures are described in this chapter to bridge the gap between the "scalar weighting" extensively used today and the very complicated "complete weighting" structure for multichannel signal processing. This is made possible by exploiting the intrinsic correlation property between channels or in other words the covariance matrix. When the covariance matrix assumes

finely structured form, the optimal filter can be chosen accordingly in the exactly same structure. In this way, excessive weighting can be avoided reducing the system complexity.

The optimizations on the new multichannel linear filters as well as the multichannel weighted median filters exploiting these two weighting strategies are developed. Moreover, these weighting structures can be extended onto the complex domain making them more suitable to real-world array processing. Simulations on color image denoising and array signal processing show that the multichannel weighted median filters in impulsive environments outperform the scalar-weighted median, the marginal multichannel implementation, and other weighted median filtering methods used today, yet require a much less weights than the complete-weighted multichannel median filter.

## Bibliography

[1] J. Astola, P. Haavisto, and Y. Neuvo, "Vector median filters," *Proceedings of IEEE*, vol. 78, no. 4, pp. 678–689, 1990.

[2] G. R. Arce, "A general weighted median filter structure admitting negative weights," *IEEE Transactions on Signal Processing*, vol. 46, no. 12, pp. 3195–3205, 1998.

[3] G. Aloupis, M. Soss, and G. Toussaint, "On the computation of the bivariate median and a Fermat-Torricelli problem," Tech. Rep. SOCS-01.2, School of Computer Science, McGill University, Montreal, Quebec, Canada, February 2001.

[4] V. Barnett, "The ordering of multivariate data," *Journal of the Royal Statistical Society Series A*, vol. 139, no. 3, pp. 318–355, 1976.

[5] C. Gini and L. Galvani, "Di talune estensioni dei concetti di media ai caratteri qualitativi," *Metron*, vol. 8, pp. 3–209, 1929, Partial English translation in *Journal of the American Statistical Association*, vol. 25, pp. 448–450.

[6] M. O. Gonzalez, *Classical Complex Analysis*, Marcel Dekker, New York, NY, USA, 1991.

[7] C. Groß and T. Strempel, "On generalizations of conics and on a generalization of the Fermat-Torricelli problem," *American Mathematical Monthly*, vol. 105, no. 8, pp. 732–743, 1998.

[8] R. C. Nardie and G. R. Arce, "Ranking in $R^p$ and its use in multivariate image estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 1, no. 2, pp. 197–209, 1991.

[9] J. B. S. Haldane, "Note on the median of a multivariate distribution," *Biometrika*, vol. 35, pp. 414–415, 1948.

[10] J. F. Hayford, "What is the center of an area, or the center of a population?" *Journal of the American Statistical Association*, vol. 8, no. 58, pp. 47–58, 1902.

[11] S. Hoyos, Y. Li, J. Bacca, and G. R. Arce, "Weighted median filters admitting complex-valued weights and their optimization," *IEEE Transactions on Signal Processing*, vol. 52, no. 10, part 1, pp. 2776–2787, 2004.

[12] W. J. Krzanowski and F. H. C. Marriott, *Multivariate Analysis. Part 1: Distributions, Ordination and Inference*, Edward Arnold, London, UK, 1994.

[13] V. Koivunen, "Nonlinear filtering of multivariate images under robust error criterion," *IEEE Transactions on Image Processing*, vol. 5, no. 6, pp. 1054–1060, 1996.

[14] Y. Li and G. R. Arce, "Weighted median filters for multichannel signals," to appear in *IEEE Transactions on Signal Processing*, 2005.

[15] Y. Li, "Nonlinear signal processing in the complex domain and higher dimensions," Ph.D. dissertation, Department of Electrical and Computer Engineering, University of Delaware, Newark, Del, USA, 2005.

[16] R. Y. Liu, "On a notion of data depth based on random simplices," *The Annals of Statistics*, vol. 18, no. 1, pp. 405–414, 1990.

[17] R. Lukac, K. N. Plataniotis, B. Smolka, and A. N. Venetsanopoulos, "Weighted vector median optimization," in *Proceedings of 4th EURASIP Conference focused on Video/Image Processing and Multimedia Communications (EC-VIP-MC '03)*, vol. 1, pp. 227–232, Zagreb, Croatia, July 2003.
[18] L. Lucat, P. Siohan, and D. Barba, "Adaptive and global optimization methods for weighted vector median filters," *Signal Processing*, vol. 17, no. 7, pp. 509–524, 2002.
[19] R. Lukac, B. Smolka, K. Martin, K. N. Plataniotis, and A. N. Venetsanopoulos, "Vector filtering for color imaging," *IEEE Signal Processing Magazine*, vol. 22, no. 1, pp. 74–86, 2005.
[20] H. Oja, "Descriptive statistics for multivariate distributions," *Statistics & Probability Letters*, vol. 1, no. 6, pp. 327–332, 1983.
[21] I. Pitas and S. Vougioukas, "LMS order statistic filters adaptation by backpropagation," *Signal Processing*, vol. 25, no. 3, pp. 319–335, 1991.
[22] K. N. Plataniotis and A. N. Venetsanopoulos, *Color Image Processing and Applications*, Springer, New York, NY, USA, 2000.
[23] P. J. Rousseeuw and M. Hubert, "Depth in an arrangement of hyperplanes," *Discrete & Computational Geometry*, vol. 22, no. 2, pp. 167–176, 1999.
[24] E. A. Robinson, *Multichannel Time Series Analysis*, Goose Pond Press, Houston, Tex, USA, 1983.
[25] Y. Shen and K. Barner, "Fast optimization of weighted vector median filters," to appear in *IEEE Transactions on Signal Processing*, 2004.
[26] M. I. Shamos, "Geometry and statistics: problems at the interface," in *Algorithms and Complexity: New Directions and Recent Results*, J. F. Traub, Ed., pp. 251–280, Academic Press, New York, NY, USA, 1976.
[27] C. G. Small, "A survey of multidimensional medians," *International Statistical Review*, vol. 58, no. 3, pp. 263–277, 1990.
[28] P. E. Trahanias, D. Karakos, and A. N. Venetsanopoulos, "Directional processing of color images: theory and experimental results," *IEEE Transactions on Image Processing*, vol. 5, no. 6, pp. 868–880, 1996.
[29] J. W. Tukey, "Mathematics and the picturing of data," in *Proceedings of International Congress of Mathematicians*, vol. 2, pp. 523–531, Vancouver, BC, Canada, 1975.
[30] T. Viero, K. Oistamo, and Y. Neuvo, "Three-dimensional median related filters for color image sequence filtering," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 4, no. 2, pp. 129–142, 1994.
[31] L. Yin, R. Yang, M. Gabbouj, and Y. Neuvo, "Weighted median filters: a tutorial," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 43, no. 3, pp. 157–192, 1996.

Yinbo Li: Electrical & Computer Engineering Department, University of Delaware, 140 Evans Hall, Newark, DE 19716, USA

*Email*: yli@ee.udel.edu

Gonzalo R. Arce: Electrical & Computer Engineering Department, University of Delaware, 140 Evans Hall, Newark, DE 19716, USA

*Email*: arce@ee.udel.edu

# 11 Color image processing: problems, progress, and perspectives

E. R. Davies and D. Charles

## 11.1. Introduction

Image processing has evolved over a period of some forty years, and it has now attained a fair measure of maturity. It has spawned a good many branches, such as image enhancement, image restoration, image matching, image compression, and image recognition, not to mention the closely allied subjects of image analysis, computer vision, and automated visual inspection—together with some apparently less related subjects such as virtual reality.

In many ways, image processing is a narrow subject, its basic definition being the study of how one image may be converted into another: traditionally, this means suppression of noise and artefacts, enhancement to improve clarity, or "deconvolution" to eliminate mathematically modeled distortions. Effectively, this definition means applying a transform to convert an image, pixel by pixel, into an improved image of the same sort (though here we should not exclude changing the resolution to match the immediate purpose for which the images will be used). This definition of image processing is motivated by the need to present human operators with improved images, which will be easier for them to examine: for example, doctors may require X-radiographs to be enhanced and to have lower levels of noise so that diagnosis will be made easier.

However, a separate motivation for image processing is to improve or adapt images so that the computer will itself be able to analyze them more easily and effectively. Image processing can then be considered as a preprocessing stage on the way to analysis and recognition. A further aspect is that image processing can be used to present a completely transformed image exhibiting edge signal instead of image intensity, or to present a corner- or other feature-enhanced image, which (to go one stage further) need only to be thresholded to indicate points that will be of immediate relevance for matching, recognition, 3D disparity measurement, and so on.

There are also higher level possibilities, such as transforming an image to a different viewpoint in 3D, taking several images, and reconstructing whole scenes,

or in some way, automatically removing undesired objects from photographs (a recent case in point is automatically removing the effects of blooming in digital cameras by deconvolution using inverses of rather complex blooming functions [1]).

In this chapter, we largely ignore this last set of complexities, and indeed all complexities arising from 3D aspects of the real world. Instead we concentrate on fairly general image processing aspects pertaining to preprocessing and the needs of subsequent image analysis; and we develop this topic to cover color processing as well as standard gray-scale processing. Color has recently been increasing in importance, with the advent of digital television, digitally generated films, and ubiquitous digital cameras—which have also become very much the sine qua non of the latest generation of mobile phones. In addition, we must not forget the huge increase in the use of surveillance—not just from earth-circling satellites, but from cameras above motorways and in urban environments including supermarkets, shops, and railway stations. Indeed, the digital camera and the webcam have fallen in cost so far that price has become almost negligible relative to the cost of computers and computer software, and, furthermore, it is actually becoming difficult to find cameras that present black and white rather than color images. Added to this, color is almost always preferred by humans (whether acting as amateurs or as professional operators), and it clearly gives far more information that may be crucial for matching and recognition tasks. Note also that the arrival of color in a convenient form and at competitive prices is very recent relative to the time span of the whole subject area, and to a certain extent we have been caught unprepared for its being provided so conveniently without behest.

It used to be possible to argue in many cases that color was a waste of resources. For example, in a simple inspection application, a gray-scale image could be thresholded to give a binary image that provided all the shape information needed to inspect metal flanges or other flat objects, and to check that all holes, corners, and other features were intact. That was a time when color seemed to add little, and indeed, one of the few inspection applications where color seemed to be crucial was in the inspection of fruit for ripeness and freedom from bruises [2].

On the other hand, we have now moved into the era of surveillance and crime prevention and detection, and it has become necessary to use full-color images to ascertain exactly what is going on in an outdoor environment: coincidentally, the September 11 scenario happened at a time when it was becoming increasingly possible to detect criminal and terrorist activity from surveillance cameras. This chapter will pay special attention to the color question, and will examine the extent to which color image processing has progressed in comparison with its gray-scale counterpart.

The aspects of color processing that will be covered in this chapter include how to manage the considerable additional information it brings; the concept of color bleeding—how color bleeding arises and how to cope with it; the contrast between linear and nonlinear filtering; the mode filter—its value and its implementation; the role of modern switched noise suppression filters; filters with adjustable parameters; image distortions produced by grayscale and color filters; and

a general review of recent color work. Further extensions and clarifications will appear in the immediately following chapter, which covers nonlinear edge detection in color images.

## 11.2. The color problem

Gray-scale images reflect a systematic change away from binary images, partly in respect of logic values giving way to intensity values, but also in respect of 1-bit arithmetic precision giving way to 8-bit (or even 12-bit) precision. In the latter case, we are merely moving along a scale of precision that can proceed as far as necessary, and the change is monotonic in nature. However, the change from gray-scale to color is distinctly different: three separate channels (or more in the case of multispectral, e.g., satellite, data) are involved, so the data characteristics have undergone a quantum leap rather than a monotonic change. Furthermore, it is not always obvious how to handle and combine the information from the different independent channels. If the images arose in a fixed application, a single channel could perhaps be produced by a simple linear weighting and addition operation applied to the three intensities, so, for example, apples could be graded on a single green-red scale. More complicated nonlinear functions could employ a lookup table to convert from color to gray-scale for the same basic purpose. However, the problem is that in a good many applications, the color is necessary, and the means of processing it to produce optimal meaningful results are likely to be complex and nontrivial. One standard approach [3] is to apply principal components analysis (PCA) to find the main information-carrying channels, and move from there to more systematic informed processing of the data. In fact, these aspects will not be broached in this chapter, as our aim here is to concentrate on the basic image processing: but the fact that there will be three independent channels rather than a single one will form a vibrant theme in the analysis.

## 11.3. Linear versus nonlinear processing

Image processing is commonly carried out in small neighborhoods or "windows," and in the past, $3 \times 3$ pixels was the most commonly used, though windows as large as $21 \times 21$ pixels are not uncommon now that computers have become more powerful. Linear processing is simple, useful, and straightforward to analyze and understand mathematically. When a linear operation is applied uniformly across the whole image, the mathematical process is actually a convolution—a very well understood process, and one that is exceptionally powerful. Convolution forms the basis of correlation-type template matching, and this provides maximum sensitivity for detection of features via the (spatial) matched filtering process, which came into prominence in the development of radar [4]. Thus it is no surprise that convolution forms the basis of most types of edge detector,[1] and has also been employed for corner detection, line segment detection, and sometimes detection of whole

[1]This also applies in the case of color; see the account given in the following chapter.

objects. The overall procedure is to use convolutions to produce an image whose pixels present the local signal for the feature in question, and then to apply nonmaximum suppression and/or thresholding—or more sophisticated methods—to perform the actual feature detection. For example, the Sobel edge detector and the Plessey corner detector apply convolutions that differentiate the image locally in the $x$ and $y$ directions, giving $g_x$ and $g_y$ differential gradient images, from which the following formulae lead to edge and corner signal values [5, 6]:

$$\begin{aligned} g &= \left(g_x^2 + g_y^2\right)^{1/2}, \\ c &= \frac{\langle g_x^2\rangle\langle g_y^2\rangle - \langle g_x g_y\rangle^2}{\langle g_x^2\rangle + \langle g_y^2\rangle}, \end{aligned} \tag{11.1}$$

the latter being deduced from the rotationally invariant determinant and trace of the locally defined matrix:

$$A = \begin{bmatrix} \langle g_x^2\rangle & \langle g_x g_y\rangle \\ \langle g_x g_y\rangle & \langle g_y^2\rangle \end{bmatrix}. \tag{11.2}$$

Several other corner detectors employ second-order local derivatives, $g_{xx}$, $g_{xy}$, $g_{yy}$—again derived by convolution masks—and these are applied within more complex formulae to estimate the local corner signal [5].

Lastly, we note that mean and Gaussian image smoothing filters also apply convolution operators, typified by the following $3 \times 3$ convolution masks:

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \qquad \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}. \tag{11.3}$$

Overall, considering that, even in a $3 \times 3$ window, the general convolution operator has nine adjustable parameters, its high range of utility and power is undeniable. Nevertheless, its power is limited. First, two convolutions applied one after another remain a convolution, not a more powerful operator (though an $n \times n$ and an $m \times m$ convolutions applied in sequence will be equivalent to a larger $p \times p$ convolution, where $p = n + m - 1$), that is, it is still linear. Second, several of the linear processes mentioned above are normally followed by a highly nonlinear operation such as thresholding to actually detect the features. Third, the image smoothing operators are well known to have the disadvantage of smoothing the signal as well as any noise, so it is generally impossible to use them for this purpose. (However, Gaussian smoothing can be used for the special purpose of progressively cutting down image detail to create a hierarchical image structure or data pyramid, leading on to scale-space representations of images—though we will not pursue this possibility further here.)

These considerations show that in general we cannot do without nonlinear processing of the image. In fact, a more important question is not whether *linear*

processing is powerful enough to be used on its own but whether *nonlinear* processing is powerful enough to be applied in this way. However, even this question is misguided. Why constrain image processing in any such way? In fact, we should use a method that is appropriate for the particular type of data and the particular task to be carried out. Again, in image smoothing, it is well known that mean filtering is optimal for images that are corrupted by Gaussian noise, while median filtering is optimal for suppressing impulse noise. (Mathematically, median filtering is optimal for double exponential noise, which has wide wings matching well the incidence of outliers [5, 7].)

## 11.4. Color filtering

The remarks in the previous section were restricted to gray-scale processing. However, it is now necessary to enquire how color processing differs from gray-scale processing. We start by taking the well-known example of the median filter. In fact, this provides an immediate problem, as the median is only applicable to a single channel (such as gray-scale) whose entities are placed in numerical order and the median element in the ordered set is selected as the output value. In the multichannel case, there is no intrinsic ordering, and the median is hence undefined. Much work has been carried out to allow a suitable ordering to be applied in the multichannel (vectorial signal) case, but the problem has no universal solution, and again, any solutions that are found will tend to be of limited applicability: as remarked in the previous section (in the case of linear and nonlinear processing), the particular choice must depend on the data and the task.

In spite of these remarks, we can deal with the median filter problem by an alternative approach that does not require an ordering procedure [7]. We merely apply an alternative formulation—that the generalized median is the result of forming the following minimum sum of distances in color space:[2]

$$M = \min_i \sum_j |d_{ij}| \quad (i \neq j), \tag{11.4}$$

where

$$d_{ij} = \left[ \sum_{k=1}^{3} (I_{i,k} - I_{j,k})^2 \right]^{1/2}, \tag{11.5}$$

$\mathbf{I}_i$, $\mathbf{I}_j$ being RGB vectors, and $I_{i,k}, I_{j,k}$ $(k = 1, \ldots, 3)$ being the color components. Thus the generalized median vector is the one that has the smallest total distance to all the other vectors. Clearly, an outlier would have a rather large total distance

[2] A more precise statement is that $\mathbf{I}_i = \arg(\min_i \sum_j |d_{ij}|)$, the value of $i$ for which the minimum occurs being used to point back to the appropriate input vector. (It may be helpful to note that the mathematical expression $\arg\min_i$ is intended to be read "the sample vector giving the minimum value of ....")

to all the other vectors, and this total distance would become smaller as the chosen vector became more central: so this is an intuitively correct solution. More particularly, in one dimension, the formulation gives an identical result to that for the standard median: there will be equal numbers of points on either side of the chosen value. The only problem in the 1D case arises when there is an even number of points. This particular formulation asserts that the proper answer is the midpoint between the two central points, whereas that was only a convenient convention in the normal median case (it was inconvenient to say that there were two medians or alternatively none, so workers chose to say that the true median was the midpoint: but the new formulation forces us to make the median the midpoint).

Finally, the generalized median is one of the original vectors, and this corresponds to the median being one of the original data points in the 1D case. From now on, we follow normal practice in calling the generalized median, defined in this way, the vector median (though we will sometimes revert to calling it the median when it is understood that we are dealing with a particular 3D manifold).

Before proceeding further, it is worth noting that the generalized mean in color space is readily defined using the same color space metric (e.g., the Euclidean distance) as for the median. There is no necessity for the result to be one of the initial data points, and indeed very little likelihood that the mean would be the same as one of the initial points, either in 1D or in the multichannel case.

So far, there has been no indication whether or not it is desirable for the output signal to be identical, in any dimensionality, to one of the input signals. It seemed that in the median case, it had to be identical, whereas in the mean case it was most unlikely to be identical. Yet in the latter case, restricting the result to one of the input values would significantly limit the accuracy of the final result. Indeed, Davies [8] has shown that this situation also applies for the single channel median filter. We will consider the situation in more detail below.

## 11.5. Color bleeding

As has already been remarked, one of the advantages of median filtering is its capability for eliminating outliers. However, when an outlier occurs near an edge, the effect can be to pull the edge closer to the outlier as well as eliminating the outlier. This is seen from the 1D examples shown in Figure 11.1, for all of which a 3-element median has been applied to the data.

This behavior could be serious in the case of multichannel data. For example, in a 3-color RGB image, if we applied the median separately to the three channels and an outlier occurred in just one of them, the result would be that the outlier would be eliminated, but the edge would be shifted in one channel, and this would result in an erroneous change of color at that location [7] (Figure 11.2). The effect might sometimes be more serious than the improvement resulting from elimination of the outlier—though "seriousness" may be at least partly a human subjective response governed by the prominence of any new color that is introduced.

There is also the opportunity for erroneous colors to arise in other situations. While edges tend to demarcate pairs of regions in an image, there will be places

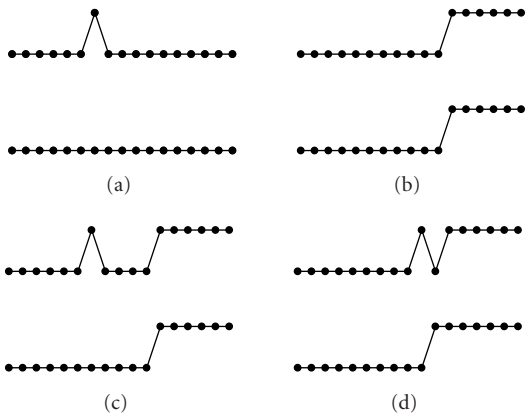



FIGURE 11.1. Application of a 1D median filter. Here a 3-element median filter is applied to four 1D signals: (a) an isolated noise point, (b) a single edge, (c) a noise point far from an edge, (d) a noise point near an edge. In (d), notice how the edge has been shifted left by one pixel.

FIGURE 11.2. Effect of scalar component-wise filtering. This figure shows the effect of mode filtering on a brightly colored image with the filter applied to each color channel independently. Color bleeding is most noticeable as isolated pink pixels and a number of green pixels around the yellow sweets. (Reproduced from D. Charles and E. R. Davies, "Mode filters and their effectiveness for processing color images," Imaging Science Journal © 2004 RPS.)

where three regions come together, and at these points there will often be three totally different colors, each of which can be regarded as an outlier to the other two.[3] Depending on the particular geometry existing at the joining regions, and the local variabilities of the channel intensities, substantial "bleeding" of one region into another can occur. The problem is exacerbated by the numbers of different colors that can result when different intensities of each color channel are brought in.

[3]It is important to remember that outliers may arise not only as a result of noise but also in the form of background clutter.

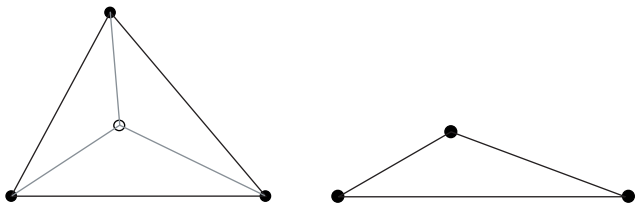



FIGURE 11.3. Geometry of three color data points in 2D. In (a), the minimum total distance point (MTDP) lies near the centroid of the triangle. In (b), the MTDP is the uppermost data point.

The value of the vector median approach now becomes particularly apparent: the output pixel vector is severely restricted by the fact that it must be the same as that of a particular pixel in the window. Thus no new color combination is generated, and the output pixel vector will in the case of a $3\times3$ window be identical to that of one of the nine pixels [7]. In larger windows, the restriction will not be so marked. However, in a $p \times p$ window, the maximum number of available colors will be $p^2$, compared with $(p^2)^3 = p^6$ when the input to any color channel may be combined with the input of any other two color channels in 3D color space.

It should next be questioned whether any disadvantage can result from application of the vector median procedure. Here we consider just one problem. We illustrate the problem by reference to a 2D color space containing three data points (the case of two data points is essentially a 1D problem). Let us find the minimum total distance point (MTDP) given by (11.4) for several sets of 2D vector data. Figure 11.3 suggests that the MTDP will, in most cases, occur in the space between the data points. Exceptions arise when the points are arranged in a rather squat triangle. In fact, for nonsquat triangles, the MTDP will lie near the centroid of the space, where the lines joining the MTDP to the three data points are at angles of 120° to each other. (An intuitive proof of this result involves first considering the points at the corners of an equilateral triangle, and then noticing that moving them, arbitrary distances from the MTDP in the same directions would not produce any movement of the MTDP.)

For four data points in 2D, there are again two possibilities [9]: one is that the MTDP is at the space near the centroid of the points; the other is that the MTDP is situated at one of the points. In the first of these cases, the actual MTDP is on the crossing of the diagonals of the quadrilateral formed from the four data points (Figure 11.4): to prove this, we merely note that moving in any direction from the crossing point changes one or other diagonal, or both, to two sides of a triangle, which must have greater total length. The second case corresponds to a concave quadrilateral (i.e., one point lies inside the triangle formed from the other three points): to prove that the central point $C$ has smaller total distance to the other points, first move to a point $T$ a short distance in the direction of the point $P$ (Figure 11.5); this will not change the sum of the distances to $C$ and to $P$, but will increase the distances to the other two points $Q$ and $R$; the MTDP is at $C$.

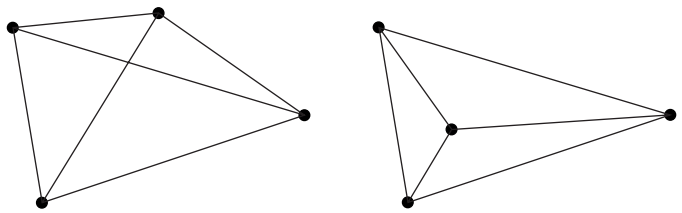



FIGURE 11.4. Geometry of four color data points in 2D. Case where (a) none and (b) one of the points lie within the triangle formed from the other three points. In (a), the MTDP is at the crossing of the diagonals, whereas in (b), it is at the central data point.

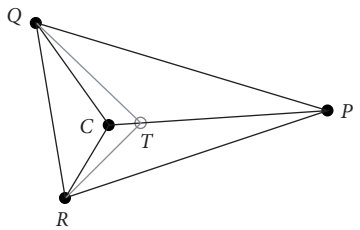



FIGURE 11.5. Geometry for proving the position of the MTDP for four color data points in 2D. In this case, the MTDP for data points $P$, $Q$, $R$, $C$ cannot be at $T$, because this would extend the distances from $Q$ and $R$, but not the sum of the distances from $P$ and $C$.

Coupled with the 1D case, these two examples show that there is something like 50% probability that the MTDP will lie in free space, and not at one of the initial data points—and this situation does not seem likely to be different in a 3D color space, or in higher dimensional spaces [9]. Thus the basic mathematical equivalent of the median ordering formulation, (11.4), would appear to state that for minimum error, we must always use the MTDP: it does not insist that the median be taken to be the data point that is closest to the MTDP. Contrariwise, restricting the output multichannel median to one of the input vector samples seems likely to introduce substantial error.

Understanding the situation is not trivial. One view is that if a purely numeric solution is needed, the MTDP must give the correct solution. On the other hand, if a true color interpretation or color recognition process is to ensue, then the vector median solution must be used, as misinterpretation of the color would constitute a major error. Another factor is that if the noise is Gaussian, the MTDP will be better (in the sense of being numerically more accurate), whereas if the noise is impulse noise, the vector median will be better (in the sense of not being open to misinterpretation). In any case, it should always be borne in mind that using the vector median procedure may not represent the most accurate solution, and that the decision about whether to use it must be task dependent.

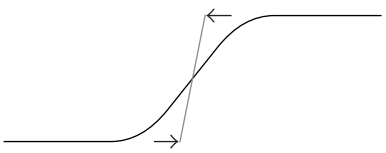


FIGURE 11.6. Action of the mode filter. Here the mode filter is applied to a 1D edge intensity profile. The filter outputs the majority value of the intensities within each window and thus pushes this intensity towards the edge boundary from each side. The result is a profile more like that indicated by the gray line.

## 11.6. The mode filter

As indicated above, median filters are widely used for suppressing noise in images, and are particularly important in view of the fact that they do not at the same time blur the image signals. In fact, there are many other types of image filter. One other type of filter based on a commonly known statistic is the mode filter. This takes the main mode of the local intensity distribution (in the case of gray-scale images) and uses it to provide the output value for each pixel. There is much sense in using this particular statistic, as its value ought to represent the most probable intensity in the local intensity distribution. However, the result is not exactly what might a priori have been expected—for the mode filter *enhances* the image rather than merely eliminating noise (though it does achieve a considerable degree of noise reduction, as the mode is generally well away from the extremes of the distribution and thus it is good at eliminating outliers). The enhancement property is achieved since the filter ignores minor modes and therefore gives a majority view in the vicinity of edge points: just inside an edge it gives the inside view, ignoring the outside view, and just outside an edge it gives the outside view, ignoring the inside view. The result is that it pushes the local majority intensity inwards from both sides of the edge—thereby giving a much more crisp boundary to the edge [10] (Figure 11.6).

Unfortunately, there are distinct problems in estimating the mode, not merely because of noise, but also because the local intensity distribution is sparse; as a result, while the highest point is technically the mode, it is not necessarily the underlying mode, which is the one that needs to be used as the filter output. To overcome this problem, Algorithm 11.1 was devised (see also Figure 11.7) [10].

The rationale of this algorithm is the following: if we knew the position of the underlying mode $U$, and located the nearer end $E$ of the distribution, and then moved from $E$ to $U$ and an equal distance further on, this would be a reasonable position to truncate the distribution. Next, noting that the median of the distribution is normally on the far side of the underlying mode relative to $E$, using the median as an estimator of $U$ is safe in the sense that it will never truncate more than the ideal proportion of distribution (Figure 11.7). While iterating the procedure can in principle give a better estimate of $U$, it has been found that this is not necessary in practice. Indeed, the mode filter has been found to produce very good

(1) Find the median $M$ of the distribution.
(2) Find the end $E$ of the distribution that is nearer to $M$.
(3) Measure the distance $d$ from $E$ to $M$.
(4) Move a further distance $d$ from $M$, to point $T$.
(5) Truncate the distribution from $T$ to the other end $F$ of the distribution.
(6) Find the median $U$ of the remaining portion of the distribution.
(7) Use the median value $U$ as an estimator of the underlying mode.

Algorithm 11.1. Mode filter algorithm for gray-scale images.

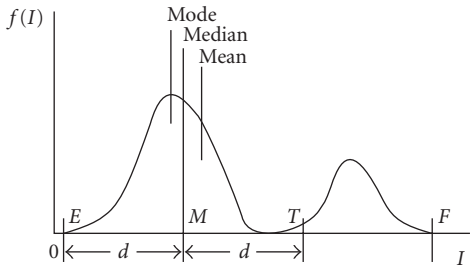



Figure 11.7. Method of truncating the local distribution. The diagram shows the ordering of the three means for a bimodal distribution. It also shows how the intensity histogram is truncated at a specified distance from the median. The rationale for this procedure is indicated by the idealized situation that would exist if the position of the mode was known initially. (Reproduced from D. Charles and E. R. Davies, "Mode filters and their effectiveness for processing color images," Imaging Science Journal © 2004 RPS.)

results with a minimum of algorithmic complexity and to be reasonably resistant to the effects of sparsity of the local intensity distribution [10]. One final aspect that is of interest here: in stage 7 of the algorithm, the median can be replaced by the mean with little effect on performance, and without introducing blurring. This is because blurring is normally introduced by a mean filter as a result of mixing together the intensities from parts of the window inside and outside the edge, but here this is prevented as the smaller of these parts has been eliminated by truncating the distribution.

### 11.6.1. The situation in color images

When applying the mode filter to color images, all the basic components of the algorithm can be replicated without problem. Specifically, the median can be replaced by the vector median, and the truncation can be carried out without difficulty. Nevertheless, determining exactly where to perform the truncation involves slight difficulties and the gray-scale algorithm has to be adapted carefully.

The first requirement is to find the vector median of the sample points in the 3D color space. Then we need to consider where the closest part on the boundary of the distribution might be. However, this is difficult to do robustly, as it is necessary to distinguish the closest points on the boundary from those internal to

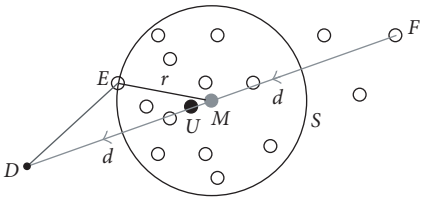


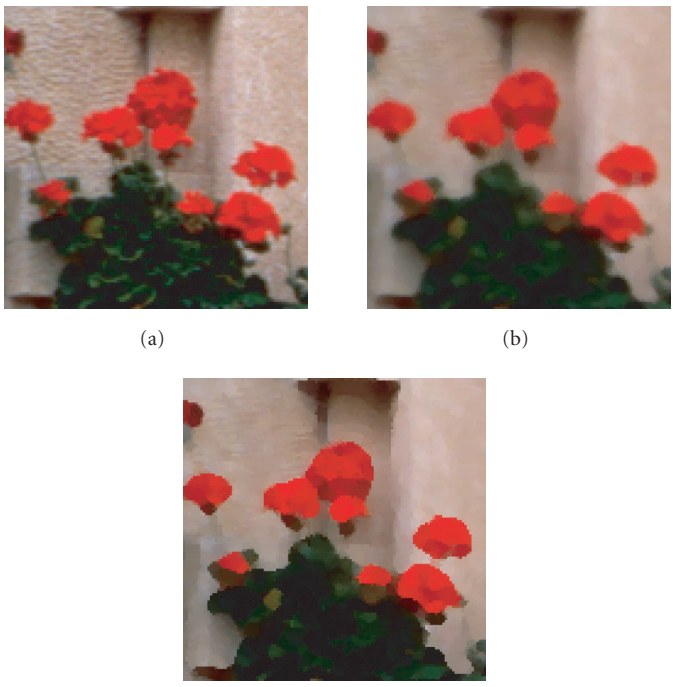

FIGURE 11.8. Method of truncating the local distribution in 2D, extendible to 3D color spaces. The circles are samples, the shaded circle is the vector median $M$ of the input distribution, and the black circle $U$ is the vector median of the truncated distribution. The dot $D$ is not a sample: it is the location in color space found by moving from sample $F$ to sample $M$ and then the same vector distance **d** further on. (Reproduced from D. Charles and E. R. Davies, "Properties of the mode filter when applied to color images," IEE Conference Publication, © 2003 IEE.)

(1) Find the vector median $M$ of the distribution.
(2) Find the end $F$ of the distribution that is furthest from $M$.
(3) Find the position $D$ that is (relative to $M$) diametrically opposite the end $F$.
(4) Find the point $E$ that is closest to $D$.
(5) Construct a sphere $S$ centered at $M$ and passing through $E$.
(6) Truncate the parts of the distribution lying outside $S$.
(7) Find the vector median $U$ of the remaining portion of the distribution.
(8) Use the vector median $U$ as an estimator of the underlying mode.

ALGORITHM 11.2. Mode filter algorithm for color images.

the distribution. In fact, it is easiest to start by determining the *furthest* part of the boundary, that is, the position and direction of the furthest outlier $F$. Once these have been located, we move to the median and then an equal vector distance **d** further on, to $D$; next, we find the sample that is closest to $D$ [11, 12] (Figure 11.8). The resulting algorithm is shown in Algorithm 11.2: it is essentially the same as in the gray-scale case. However, it is important to note that stage 8 now has to employ the vector median rather than the mean or the MTDP, as the final position selected must be one of the original data points, so that color bleeding will be suppressed.

An interesting aspect of mode filters in color images is that in addition to their enhancement properties (Figure 11.9), they turn out to have remarkable capabilities for eliminating large amounts of impulse noise [12]. In fact, up to 70% of randomly corrupted pixels containing random colors (each color channel having 70% of the intensities uniformly distributed over the entire range 0 to 255) were largely eliminated with very little distortion of the final images (Figure 11.10). This capability for elimination of high levels of noise was not matched by the median filter in spite of the fact that the median filter ought to be more resistant to outliers, as theory and earlier work on gray-scale images had suggested. It is likely that this degree of improvement resulted from using larger windows than used to be customary with these relatively computation intensive filters, thereby allowing the



(a) (b)

(c)

FIGURE 11.9. Effect of the mode filter on a color image. (a) Original image. (b) Effect of vector median filtering. (c) Effect of vector mode filtering. Both of the filters are applied in a $5 \times 5$ window. (Reproduced from D. Charles and E. R. Davies, "Mode filters and their effectiveness for processing color images," Imaging Science Journal © 2004 RPS.)

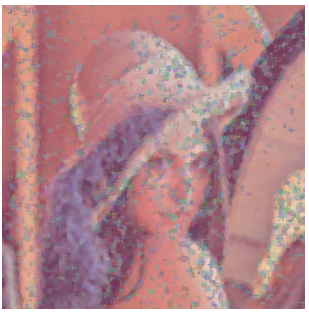
mode filter to lock on to and make full use of the remaining 30% of unaffected pixels. Another factor is the considerable discriminating power available when color is taken into account.

## 11.7. Modern "switched" noise suppression filters

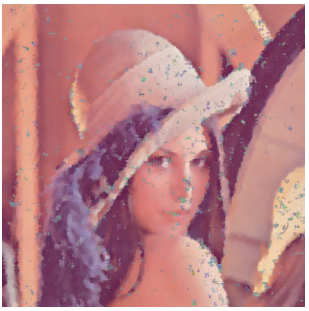
In the past few years, several filters that are substantially better than the median at eliminating noise have been devised (see, e.g., [13–16]), and some comment on the mode filter's noise performance vis-à-vis these new filters is appropriate.

Typically, these new filters obtain their improvements by analyzing the image contents more carefully than the median does, and acting accordingly. Rather than merely sorting the pixel intensity values and taking no account of where in the window each intensity originated, the new filters adopt cleverer strategies. In particular, they have several means of calculating the output intensity, and select amongst (or "switch" between) these values according to deductions made about the situation at and around each pixel. For instance, they retain the option of using the output of a standard median filter or leaving the pixel intensity unchanged

(a)



(b)



(c)

FIGURE 11.10. Filtering in the presence of large amounts of impulse noise. (a) Lena image with 70% random impulse noise. (b) Effect of vector median filtering. (c) Effect of vector mode filtering. Both of the filters are applied in a $9 \times 9$ window. (Reproduced from D. Charles and E. R. Davies, "Mode filters and their effectiveness for processing color images," Imaging Science Journal © 2004 RPS.)

if it is judged not to be modified by impulse noise. In the case of the Eng and Ma [13] noise adaptive soft-switching median (NASM) and "ideal" filters,[4] selection of output values is also made on the basis of any local edges that are detected in the image and whether more than one noise impulse is found at or near a given pixel. These particular filters also consider the optimum size of the filter window, larger windows being selected when the noise density is greater.

Tests made by Charles and Davies [12] confirmed that the two filters proposed by Eng and Ma were very effective at suppressing high levels of impulse noise. These tests employed gray-scale images, as the Eng and Ma filters only provided this capability. In comparison with the median and mode filters, the peak signal-to-noise ratios achieved for an impulse noise density of 70% were

(i) median 17 dB,
(ii) mode 24 dB,
(iii) NASM 28 dB,
(iv) ideal 32 dB.

[4]The "ideal" filter is a simulated idealized filter in which the positions of the noise impulses are taken to be known, and to this extent it does not provide a realistic comparison.

Clearly, then, the mode filter is not as effective as the NASM filter at suppressing high densities of impulse noise, though in this respect it is much better than the median filter. What is interesting is how well the mode filter performs despite its simplicity (the C++ code is only about 30 lines long) and lack of specific mechanisms for adapting to the local conditions.

As remarked above, the two switched types of filter outlined above were applied only to gray-scale images, and it is of interest here whether they could be applied successfully to color images. The critical aspects of these filters include the use of a median filter and switching its output on and off depending on local conditions in the image. None of these aspects precludes their use with color images, the only proviso being that a vector median filter needs to be used in place of a median filter. Further developments in this area are therefore to be expected in the near future.

## 11.8. Filters with adjustable parameters

Many image filters such as the median filter have no adjustable parameters other than the crude one of deciding the size of window in which they are to be applied. Hence there is virtually no scope for optimization to the conditions occurring in different images—such as varying levels of noise. Thus there has been a trend towards the inclusion of adjustable parameters within filter algorithms. As we will see below, these parameters often take the form of weights.

The concept of weighted median filters is now well established in the literature, and a number of workers have given much attention to the development of weighted filters for gray and color image noise removal and enhancement (Sun and Neuvo [17]; Fotopoulos et al. [18]; Sangwine and Horne [19]; Marshall [20]). In one definition (Sun and Neuvo [17]), the weighting is achieved by duplication of samples inside the filter, such that with input sample set $\mathbf{x} = [x_1, x_2, \ldots, x_n]$ and corresponding weights $\mathbf{w} = [w_1, w_2, \ldots, w_n]$, the output $y$ of the filter is

$$y = \text{median}[w_1 \diamondsuit x_1, w_2 \diamondsuit x_2, \ldots, w_n \diamondsuit x_n], \tag{11.6}$$

where $\diamondsuit$ denotes weighting by using each $x_i$ data point $w_i$ times. In this case, the $w_i$ are necessarily integral. In contrast, the minimum weighted distance function defined by Fotopoulos et al. [18] (leading to the result $\mathbf{x}_i$) is

$$M = \min_i w_i \sum_j |d_{ij}| \quad (i \neq j), \tag{11.7}$$

where the weights $w_i$ need not be integral.

In an alternative approach, Charles and Davies [21] introduced a filter that incorporates a window distance weighting function. In particular, (11.4) is modified by introducing a weighting function $f(r_i)$ dependent on the Euclidean distance $r_i$

between pixel $i$ and the center of the window:

$$M = \min_i f(r_i) \sum_j |d_{ij}| \quad (i \neq j), \tag{11.8}$$

where

$$r_i = \left[ (x_i - x_0)^2 + (y_i - y_0)^2 \right]^{1/2}. \tag{11.9}$$

Here $x_0$ and $y_0$ represent the location of the center pixel and $f(r)$ is the weighting function chosen according to the application.

The weighting functions $f(r)$ are necessarily increasing functions which are largest at large distances. This is useful for continuity reasons, so that the limitations of the window support region do not have a too great effect on the properties of the operation: indeed, if the function value went to infinity at the boundary of the support region, increasing the size of the window would give no observable effect. (It is of course necessary to remember that the most relevant function values are those for small $r$, as (11.8) takes the minimum value as the output.)

The value of this approach is that one or two crucial carefully selected parameters can be employed instead of a plethora of arbitrary weights. In fact, in [21], only a single parameter $\alpha$ reflecting the radial extent of the function was used:

$$f(r) = 1 + \frac{r}{\alpha}. \tag{11.10}$$

Clearly, the weighting function and the amount of weighting applied will vary with the type of application and desired results, but even in its basic form, this type of weighted median filter moves continuously from the detail-preserving properties of a small operator to the greater noise-averaging properties of a large operator by adjustment of the single parameter $\alpha$.

Figure 11.11 shows the shape of the normalized mean-square error (NMSE) optimization curve when applying the distance-weighted filter for a range of parameter values. (In such cases, tests are made by adding known amounts of noise and then measuring how much of the noise is eliminated by the algorithm.) Note that the horizontal asymptote shows the situation when a standard vector median filter is employed—that is, this is a limiting, suboptimal case of the distance-weighted filter.

## 11.9. Distortions produced by median and other filters

The previous sections of this chapter have discussed image processing and filtering, and have gone to some trouble to consider how color modifies the algorithms and to assess their effectiveness. At this point it is useful to consider the extent to which they may produce distortions in the images as a by-product of their desired action, whether this be noise elimination or some other effect such as enhancement.

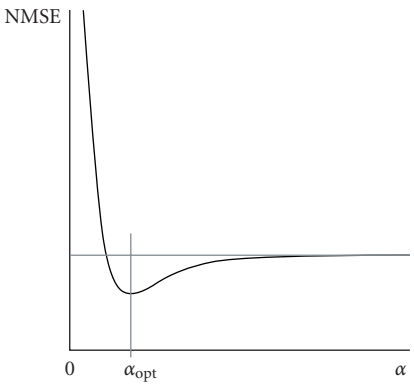



FIGURE 11.11. Shape of the normalized mean-square error (NMSE) optimization curve when applying the distance weighted filter for a range of parameter values. The horizontal gray line shows the NMSE for the standard vector median filtered image.

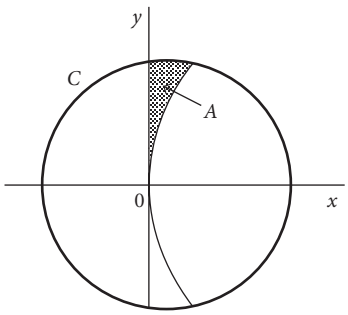



FIGURE 11.12. Geometry for calculation of contour shifts using the median filter. The area *A* is shown shaded. The origin on the median contour (which bisects the area of the circular window *C*) is not at the center of *C*. (Reproduced from E. R. Davies, "Image distortions produced by mean, median, and mode filters," IEE Proceedings-Vision Image and Signal Processing, © 1999 IEE.)

In fact, median and other filters produce distinct edge shifts, but only for curved edges. For straight edges, symmetry dictates that there will be no shifts (any straight edge will be shifted laterally by the same amount in both directions). To discuss the situation for curved edges, consider first the case of binary images—or, equivalently, gray-scale images containing step edges. Here the median contour in a window will be the one with equal areas (equal numbers of pixels) on either side of the contour. The same applies when mean or mode images are being produced. Interestingly, the same result applies again in the case of the median filter when the edge profiles present linear rather than step variations. We will return later to certain other possibilities.

Fortunately, it is quite simple to see what happens when a circular boundary divides the region in a circular window into equal areas [22]. Examining Figure 11.12, we can see that the circular boundary cannot pass through the center of the window, so when its intensity is placed at the output of the filter, it will appear at

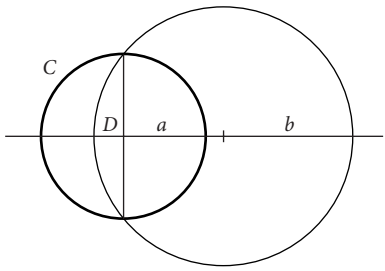



FIGURE 11.13. Geometry for the calculation of contour shifts using the mode filter. (Reproduced from E. R. Davies, "Image distortions produced by mean, median, and mode filters," IEE Proceedings-Vision Image and Signal Processing, © 1999 IEE.)

the center position in the output image—that is, there will be a distinct shift inwards. We can find an approximate value for the shift by taking a circular contour, centered at $(b, 0)$ and passing through the origin (Figure 11.12)

$$(x - b)^2 + y^2 = b^2. \tag{11.11}$$

After simplifying and approximating to large $b$, we find

$$x \approx \frac{y^2}{2b}. \tag{11.12}$$

The part of the area in the positive quadrant lying to the left of the contour is

$$A \approx \int_0^a x\,dy = \frac{1}{2b}\int_0^a y^2 dy = \frac{1}{2b}\left[\frac{y^3}{3}\right]_0^a = \frac{1}{6b}a^3 = \frac{1}{6}\kappa a^3, \tag{11.13}$$

where $a$ is the radius of the circular window; the result is an approximation which is only valid if the curvature $\kappa(= 1/b)$ is small and applies because $A$ is actually bounded by the curved boundary of the window rather than a horizontal line distant $a$ from the $x$-axis. For a median contour, $A$ must be equal to $aD$. Hence

$$D = \frac{1}{6}\kappa a^2. \tag{11.14}$$

This proves the result for a median filter for all shapes of edge profile, and for the mean and mode filters when there is a step edge profile. Moving on to the mean filter when there is a linear edge profile, we find that we have a similar calculation to perform with similar approximations, except that in this case we have to find the mean intensity in the window using a polar coordinate representation, and then deduce for what contour the mean intensity would arise. This leads to the formula

$$D = \frac{1}{8}\kappa a^2. \tag{11.15}$$

For details of the calculation, the reader is referred to [23, 24].

TABLE 11.1. Summary of shifts predicted on continuum model. (Reproduced from E. R. Davies, "Image distortions produced by mean, median, and mode filters," IEE Proceedings-Vision Image and Signal Processing, © 1999 IEE.)

| Edge type | Mean | Median | Mode |
|---|---|---|---|
| Step | $\frac{1}{6}\kappa a^2$ | $\frac{1}{6}\kappa a^2$ | $\frac{1}{6}\kappa a^2$ |
| Intermediate | $\sim \frac{1}{7}\kappa a^2$ | $\frac{1}{6}\kappa a^2$ | $\frac{1}{2}\kappa a^2$ |
| Linear | $\frac{1}{8}\kappa a^2$ | $\frac{1}{6}\kappa a^2$ | $\frac{1}{2}\kappa a^2$ |

In the case of the mode filter when there is a step edge profile, the situation turns out to be far simpler and no integration is required [24, 25]. This is because the contour with the mode intensity must be the longest one, and is therefore the one which spans the longest diameter of the window (Figure 11.13). Thus we merely have to ask how far this contour is from the center of the window. The answer is given by setting $x = D$ and $y = a$ in (11.11) and (11.12), leading to the formula

$$D = \frac{1}{2}\kappa a^2. \tag{11.16}$$

All the above results are summarized in Table 11.1. Table 11.1 also includes a set of estimates of shifts for intermediate intensity patterns—namely those whose profiles lie between a linear edge and a step edge. Note, however, that a full explanation of the variations in the shifts for the mode filter is quite complex [24].

### 11.9.1. Using a discrete model to explain median shifts

The explanation of edge shifts given above was completely analog in nature, as it effectively dealt with spatial densities of pixels in various parts of the filter window. While it leads to good estimates of the shifts for large windows, it is inaccurate for small windows and thus a new discrete model had to be developed to make accurate predictions of the shifts [22, 26].

To produce a discrete model, we need to recognize explicitly the positions of the pixels within an $n \times n$ window. We approximate by assuming that the intensity of any pixel is the mean intensity over the whole pixel and is represented by a sample positioned at the center of the pixel. We start by examining the case of a $3 \times 3$ window, and proceed by taking the underlying analog intensity variation to have contours of curvature $\kappa$.

First, zero shift occurs for $\kappa = 0$. Next, if $\kappa$ is even minutely greater than zero, the center pixel will not necessarily be the median pixel. Consider the case when the circular median intensity contour passes close to the center of the window at a small angle $\theta$ to the positive $x$-axis (Figures 11.14 and 11.15(a)). In that case, the

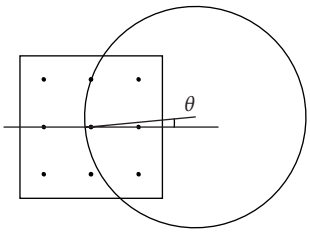



FIGURE 11.14. Geometry for the calculation of median shifts on the discrete model. (Reproduced from E. R. Davies, "Image distortions produced by mean, median, and mode filters," IEE Proceedings-Vision Image and Signal Processing, © 1999 IEE.)

filter will produce a definite shift, whose value is

$$D_\theta \approx \frac{1}{2}\kappa a_0^2 - a_0\theta = \frac{1}{2}\kappa - \theta, \tag{11.17}$$

where $a_0$ is the interpixel separation, which is taken as unity in this chapter. (The factor 1/2 in the first term arises following (11.12) when $y$ is replaced by $a_0$.) If $\theta$ is close to 45°, the shift will have the value

$$D_\phi \approx \frac{1}{2}\kappa(\sqrt{2}a_0)^2 - (\sqrt{2}a_0)\phi = \kappa - \sqrt{2}\phi, \tag{11.18}$$

where

$$\phi = \frac{\pi}{4} - \theta \tag{11.19}$$

and the relevant pixel separation is $\sqrt{2}a_0$ rather than $a_0$ (see Figure 11.15(c)).

As indicated above, (11.17) and (11.18) are approximate. However, the above derivations and solutions provide a useful insight into the situation. In particular, (11.17) and (11.18) show that at the ends of the range $0 \le \theta \le \pi/4$, $D_\theta$ varies in proportion to $\Delta\theta$. The next problem is understanding what happens when $D_\theta$ falls to zero at intermediate values of $\theta$. In fact, $D_\theta$ remains at zero in this range, the reason being that the median contour reverts to passing through the central pixel in the window (Figure 11.15(b)). The resulting approximately piecewise-linear variation in $D_\theta$ is far from what would be expected on the continuum model. To make a realistic comparison, we must average over all $\theta$. In that case we obtain the result

$$\begin{aligned} D &\approx \frac{\left[\int_0^{\kappa/2}(\kappa/2-\theta)\,d\theta + \int_0^{\kappa/\sqrt{2}}(\kappa-\sqrt{2}\phi)\,d\phi\right]}{(\pi/4)} \\ &= \frac{4}{\pi}\left\{\left[-\frac{1}{2}\left(\frac{\kappa}{2}-\theta\right)^2\right]_0^{\kappa/2} + \left[-\frac{1}{2\sqrt{2}}(\kappa-\sqrt{2}\phi)^2\right]_0^{\kappa/\sqrt{2}}\right\} \\ &= \frac{1+2\sqrt{2}}{2\pi}\kappa^2 \approx 0.61\kappa^2. \end{aligned} \tag{11.20}$$

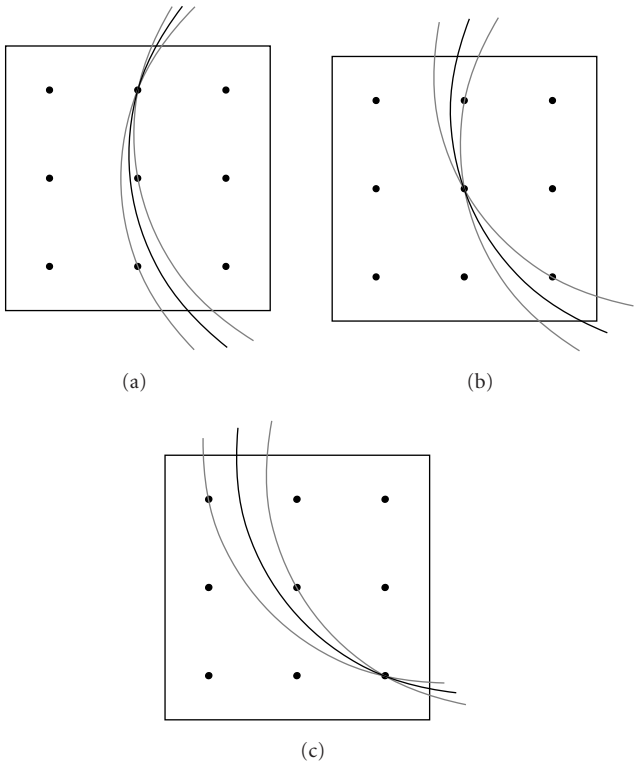



FIGURE 11.15. Geometry for the calculation of median shifts at low $\kappa$. These three diagrams show the positions of the median pixels and the ranges of orientations of circular intensity contours for which they apply: (a) for low $\theta$, (b) for intermediate $\theta$, and (c) for high $\theta$. (Reproduced from E. R. Davies, "Image distortions produced by mean, median, and mode filters," IEE Proceedings-Vision Image and Signal Processing, © 1999 IEE.)

This shows that $D$ follows a square law rather than a linear law at low values of $\kappa$, unlike the situation for the continuum model. However, for increasing values of $\kappa$, the variation reverts ever more closely to a linear model: this occurs when $\kappa$ reaches such high values that the range of values of $\theta$ for which $D_\theta = 0$ falls to zero.

### 11.9.2. The situation for color images

Interestingly, the continuum and discrete shift models given above apply without change for color images. It is clear that they apply for the individual color channels. However, why should they apply when the colors are considered together? The reason is that the models are rather special in considering single idealized circular edges, with no complicating factors arising from other edge fragments (such as might arise from shadows or image clutter) or from noise. The models were very

carefully set up to consider what happened to the underlying signal. Other work had already been carried out on the effects of noise and how noise can introduce edge shifts (as has already been considered in Section 11.5) [27, 28]. Thus the idea of the two types of the model was to provide a framework on which noise aspects could be considered properly.

As a check on the two models and the theories of edge shifts they gave rise to, careful experimental measurements were made by Charles and Davies [12] for median filters: no detectable difference between the color and gray-scale shifts was found. Exactly the same situation occurred for color and gray-scale mode filters (though at certain curvature values the mode shifts suddenly rose sharply relative to the median shifts—a result for which a reasonable explanation was found [12]).

Overall, intrinsic shifts have been found to occur for binary, gray-scale, and color images when median, mean, and mode filters are applied. It has been found possible to explain these shifts using continuum models and, to a limited extent, (but where this is possible, more accurately) using discrete models.[5] As the shifts are intrinsic rather than being due to noise, they are unavoidable, and in practical applications, they can be dealt with in the following ways:

(1) by avoiding the use of filters wherever possible;
(2) by applying special types of filter such as the hybrid median filter [29] or the neural network (trained) filter [30];
(3) by correcting for the shifts using the models outlined in this section.

## 11.10. Review of other color work

In the preceding sections, color arose in a natural way as the simultaneous representation and use of three color channels—normally RGB—and no interaction between them was envisaged. However, it was noted that the results of processing using obvious types of filter could appear unnatural to the human eye. It is also the case that image analysis needs to make best use of the image colors if objects such as faces and trees are to be segmented or recognized easily. Indeed, one does not have to look far in the literature of color vision (human or machine) before questions emerge about the most appropriate color spaces.

If color filtering was solely an abstract process, it would make little difference which of many orthogonal rotations were made in a color space from the original RGB axes. However, human perceptual problems, and the complex task of machine recognition, place a substantial gloss on this question. But the situation is far more complex than this. Spaces such as HSI (hue, saturation, intensity) reexpress the relevant variables nonlinearly, in such a way that hue is an angle and saturation is the argument of the hue, to which varying degrees of whiteness are added. Thus hue is periodic; furthermore, its value is meaningless when $S = 0$, a situation that

[5]Note that the mathematical analysis of discrete models is not trivial, and so far has only been achieved for median filters. Hence continuum models have by no means been superseded: this is why both types of model have been examined in the present chapter.

arises both when $I = 0$ and when $I = 255$;[6] this reflects the "double-cone" structure of color space (the vertices of the cones lie at $(0, 0, 0)$ and $(255, 255, 255)$).

The motivation for the HSI representation is supposedly that of theoretical relevance, but the factors just mentioned make it difficult to work with. As a result, many workers have searched for more appropriate representations within their own spheres of relevance. Normalized RGB (or rgb) is a useful representation, and is also useful as a paradigm for comparison:

$$\begin{aligned} r &= \frac{R}{(R+G+B)}, \\ g &= \frac{G}{(R+G+B)}, \\ b &= \frac{B}{(R+G+B)}. \end{aligned} \tag{11.21}$$

It will be impossible here to do justice to the plethora of color spaces that have been developed. Instead it will be useful to outline a few choices workers have recently made in different areas of application. For further examples and insights into color spaces, see Chapter 12.

### 11.10.1. Recognizing human skin tones

When finding human skin regions, Chen and Grecos [31] found that skin-tone color pixels are most compact (and therefore most easily segmented) in normalized RGB space, and recalled previous work showing that normalized RG color space performs almost as well as RGB space. Working with this space, they cut down computation and also avoided a color affine transformation. Hsu et al. [32] tackled the same problem, stating that normalized RG space is not the best choice for face detection, and adopting $YC_bC_r$ space since it is "perceptually uniform" and also gives compact skin clusters in color space.

### 11.10.2. Extraction of multicolored objects

Bucha et al. [33], working on extraction of multicolored cartographic objects, stated that it is difficult to design color image processing algorithms in RGB space because of the high correlation between channels, and asserted that HSI allows more robust measures of the differences between two colors to be made; they then reiterated the difficulties with HSI mentioned earlier. They therefore combined RGB and HSI to obtain more reliable distance measures, following earlier work by Gevers and Smeulders [34]. The latter presented several new color models and gave theoretical justification for choice of color spaces depending on whether the

[6] When $S \approx 0$, a similar situation arises, especially if noise is present.

illumination is controlled, whether highlights are present, and whether the objects to be recognized are multicolored.

### 11.10.3. Mathematical morphology in color spaces

Hanbury and Serra [35] approach the problem of mathematical morphology in color spaces. In fact, color morphology suffers from exactly the same problems as median (and other) filtering, in that an ordering of the pixels within the window is needed, and in color images it is a priori undefined. Thus an ordering has to be imposed. Lexicographic ordering (applying a dictionary type of ordering, such as ordering by $R$, then by $G$ and then by $B$—effectively ordering by the value of $256^2R + 256G + B$) is an obvious one, but is arbitrary: in fact, any such ordering may be appropriate in some situations but poor in others. The chapter aimed to find a theoretically more suitable ordering: specifically, it examined how the HLS space could best be adapted for the purpose (a priori, hue should be taken first, then saturation and finally luminance). They found that because HLS space has a double-cone shape, it is far better to weight hue by saturation, and only then to prioritize using hue, saturation, and luminance (note that hue had to be applied on its own to avoid introducing false colors). Another aspect of this work was the need to choose an origin of hue, as this permitted ordering on the hue unit circle—hue being a periodic function. However, this imparted a degree of arbitrariness to the methodology, which would make it less general but at the same time more adaptable to different recognition tasks. Discussions of some alternative schemes are given by Hanbury and Serra [36].

J. Li and Y. Li [37] took a rather different approach to ordering in color space by concentrating on principal component analysis (PCA) to provide a suitable ordering procedure. As a result, the method is able to preserve colors in the same way as the vector median filter.

### 11.10.4. Effect of varying illumination and shadows

All the cases listed so far provide fixed color spaces with perhaps one or two parameters that are adjusted to optimize for any given application. The approach made by Finlayson et al. [38, 39] looks instead at achieving color constancy—that is, resistance to natural variations in illumination on objects. In this work, two images of a scene are taken, in one of which a color filter is placed in front of the camera. This gives additional information which allows the color to be normalized. Specifically, the additional information is of such a type that it can be used to cancel out misleading information that arises as the background illumination changes. The work is founded on rigorous theory and is practically impressive; for details, the reader is referred to [38, 39]. In related work, Finlayson et al. have found how to eliminate shadows from color images [40]: only a single image is needed once the camera has been calibrated. Both of these sets of work can be called "filtering," though they are some way away from conventional filtering as described in the literature as, to some degree, they embody global understanding of image content.

## 11.11. Summary

This chapter has presented a short survey of image processing, color filtering, and their inter-relationship. There is clearly much more to be said on this topic, given its marked overlap with image analysis and recognition. One point that has emerged strongly is that image processing does not happen in a vacuum: it is needed as an aid both for human interaction and for computer interpretation. In either, case divergences and preferences exist. What humans regard as divisive color bleeding may be largely irrelevant to a computer recognition algorithm, though in the latter case it could introduce inaccuracies of measurement.

At first, color bleeding appeared to be an accidental occurrence in image filtering. However, its incidence is deeper, and it applies equally to mathematical morphology, which is a widely used, rather generalized approach to the processing and analysis of images. Intrinsic to bleeding is the need to order image data points, and in a multichannel context, this is fraught with difficulty. Some approaches to multichannel ordering, such as lexicographic ordering, are obviously arbitrary, and there is no reason why they would work well in practice (except with restricted data sets and algorithm settings). More systematic approaches aim to identify relevant color representations, but these will often be applicable only to restricted types of data. Methods based on PCA have more right to be called general, but (as in many other applications of PCA) they may still be suboptimal and at this stage they do not appear to be fully proven experimentally in the color context.

Color is a complex topic: while it is perceptually simple for humans, it is difficult to program and equally difficult to be sure of optimality. In fact, the computer vision community has not had anything like as much experience with color as with gray-scale images, but with the advent of very cheap color cameras, this situation seems certain to change quite quickly. Nevertheless, the multichannel ordering problem is a hard one, and much more work is required before the theoretical backcloth is fully prepared for guaranteeing optimal algorithms. The following chapter confirms that the problems recounted in the present chapter are not parochial, but are manifest for a good many aspects of color processing. For further reading, a number of useful articles on color image processing appear in [41]: in particular, the article by Lukac et al. on vector filtering contains valuable additional topics such as vector directional filters which could not be covered adequately in the present chapter [42].

## Acknowledgments

Professor E. R. Davies is grateful to Research Councils UK for Basic Technology Grant GR/R87642/02. Table 11.1, Figures 11.8, 11.12–11.15, and some of the text are reproduced from [11, 22] with permission from the IEE; Figures 11.2, 11.7, 11.9, 11.10, and some of the text are reproduced from [12] with permission from the Royal Photographic Society.

## Bibliography


[1] G. Ward, "High dynamic range imaging," in *Invited Paper at the IEE International Conference on Visual Information Engineering (VIE '05)*, Glasgow, Scotland, UK, April 2005.

[2] P. H. Heinemann, Z. A. Varghese, C. T. Morrow, H. J. Sommer III, and R. M. Crassweller, "Machine vision inspection of 'Golden Delicious' apples," *Applied Engineering in Agriculture*, vol. 11, no. 6, pp. 901–906, 1995.

[3] Y. Ohta, *Knowledge-Based Interpretation of Outdoor Natural Color Scenes*, Pitman, Marshfield, Mass, USA, 1985.

[4] E. R. Davies, *Electronics, Noise and Signal Recovery*, Academic Press, London, UK, 1993.

[5] E. R. Davies, *Machine Vision: Theory, Algorithms, Practicalities*, Morgan Kaufmann, San Francisco, Calif, USA, 3rd edition, 2005.

[6] E. R. Davies, "Using an edge-based model of the Plessey operator to determine localisation properties," in *Proceedings of IEE International Conference on Visual Information Engineering (VIE '05)*, pp. 385–391, Glasgow, Scotland, UK, April 2005.

[7] J. Astola, P. Haavisto, and Y. Neuvo, "Vector median filters," *Proceedings of IEEE*, vol. 78, no. 4, pp. 678–689, 1990.

[8] E. R. Davies, "A remanent noise problem with the median filter," in *Proceedings of 11th IAPR International Conference on Pattern Recognition (ICPR '92)*, vol. 3, pp. 505–508, The Hague, The Netherlands, August–September 1992.

[9] E. R. Davies, "Accuracy of multichannel median filter," *Electronics Letters*, vol. 36, no. 25, pp. 2068–2069, 2000.

[10] E. R. Davies, "On the noise suppression and image enhancement characteristics of the median, truncated median and mode filters," *Pattern Recognition Letters*, vol. 7, no. 2, pp. 87–97, 1988.

[11] D. Charles and E. R. Davies, "Properties of the mode filter when applied to colour images," in *Proceedings of IEE International Conference on Visual Information Engineering (VIE '03)*, IEE Conference Publication, no. 495, pp. 101–104, Guildford, UK, July 2003.

[12] D. Charles and E. R. Davies, "Mode filters and their effectiveness for enhancing colour images," *Imaging Science Journal*, vol. 52, no. 1, pp. 3–25, 2004.

[13] H.-L. Eng and K.-K. Ma, "Noise adaptive soft-switching median filter," *IEEE Transactions on Image Processing*, vol. 10, no. 2, pp. 242–251, 2001.

[14] T. Chen and H. R. Wu, "Application of partition-based median type filters for suppressing noise in images," *IEEE Transactions on Image Processing*, vol. 10, no. 6, pp. 829–836, 2001.

[15] R. Lukac, "Adaptive vector median filtering," *Pattern Recognition Letters*, vol. 24, no. 12, pp. 1889–1899, 2003.

[16] X. Xu, E. L. Miller, D. Chen, and M. Sarhadi, "Adaptive two-pass rank order filter to remove impulse noise in highly corrupted images," *IEEE Transactions on Image Processing*, vol. 13, no. 2, pp. 238–247, 2004.

[17] T. Sun and Y. Neuvo, "A simple synthesis method for weighted median filters," in *Proceedings of 6th European Signal Processing Conference (EUSIPCO '92)*, pp. 1405–1408, Brussels, Belgium, August 1992.

[18] S. Fotopoulos, D. Sindoukas, N. Laskaris, and G. Economou, "Image enhancement using color and spatial information," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '97)*, vol. 4, pp. 2581–2584, Munich, Germany, April 1997.

[19] S. J. Sangwine and R. E. N. Horne, Eds., *The Colour Image Processing Handbook*, Chapman & Hall, London, UK, 1998.

[20] S. Marshall, "New direct design method for weighted order statistic filters," *IEE Proceedings on Vision, Image and Signal Processing*, vol. 151, no. 1, pp. 1–8, 2004.

[21] D. Charles and E. R. Davies, "Distance-weighted median filters and their application to colour images," in *Proceedings of IEE International Conference on Visual Information Engineering (VIE '03)*, IEE Conference Publication, no. 495, pp. 117–120, Guildford, UK, July 2003.

[22] E. R. Davies, "Image distortions produced by mean, median and mode filters," *IEE Proceedings on Vision, Image and Signal Processing*, vol. 146, no. 5, pp. 279–285, 1999.

[23] E. R. Davies, "Median and mean filters produce similar shifts on curved boundaries," *Electronics Letters*, vol. 27, no. 10, pp. 826–828, 1991.

[24] E. R. Davies, "An analysis of the geometric distortions produced by median and related image processing filters," *Advances in Imaging and Electron Physics*, vol. 126, pp. 93–193, 2003.

[25] E. R. Davies, "Shifts produced by mode filters on curved intensity contours," *Electronics Letters*, vol. 33, no. 5, pp. 381–382, 1997.

[26] E. R. Davies, "Formulation of an accurate discrete theory of median shifts," *Signal Processing*, vol. 83, no. 3, pp. 531–544, 2003.

[27] G. J. Yang and T. S. Huang, "The effect of median filtering on edge location estimation," *Computer Graphics and Image Processing*, vol. 15, no. 3, pp. 224–245, 1981.

[28] A. C. Bovik, T. S. Huang, and D. C. Munson Jr., "The effect of median filtering on edge estimation and detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, no. 2, pp. 181–194, 1987.

[29] A. Nieminen, P. Heinonen, and Y. Neuvo, "A new class of detail-preserving filters for image processing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, no. 1, pp. 74–90, 1987.

[30] D. Greenhill and E. R. Davies, "Relative effectiveness of neural networks for image noise suppression," in *Pattern Recognition in Practice IV: Multiple Paradigms, Comparative Studies and Hybrid Systems*, E. S. Gelsema and L. N. Kanal, Eds., pp. 367–378, Elsevier Science, Amsterdam, The Netherlands, 1994.

[31] L. Chen and C. Grecos, "A fast skin region detector for colour images," in *Proceedings of IEE International Conference on Visual Information Engineering (VIE '05)*, pp. 195–201, Glasgow, Scotland, UK, April 2005.

[32] R.-L. Hsu, M. Abdel-Mottaleb, and A. K. Jain, "Face detection in color images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 696–706, 2002.

[33] V. Bucha, S. Ableameyko, and T. Pridmore, "Semi-automatic extraction and vectorisation of multicoloured cartographic objects," in *Proceedings of IEE International Conference on Visual Information Engineering (VIE '05)*, pp. 115–120, Glasgow, Scotland, UK, April 2005.

[34] T. Gevers and A. W. M. Smeulders, "Color-based object recognition," *Pattern Recognition*, vol. 32, no. 3, pp. 453–464, 1999.

[35] A. G. Hanbury and J. Serra, "Mathematical morphology in the HLS colour space," in *Proceedings of 12th British Machine Vision Conference (BMVC '01)*, vol. 2, pp. 451–460, Manchester, UK, September 2001.

[36] A. G. Hanbury and J. Serra, "Morphological operators on the unit circle," *IEEE Transactions on Image Processing*, vol. 10, no. 12, pp. 1842–1850, 2001.

[37] J. Li and Y. Li, "Multivariate mathematical morphology based on principal component analysis: initial results in building extraction," *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 35, no. B7, pp. 1168–1173, 2004.

[38] G. D. Finlayson and P. Morovic, "Human visual processing: beyond 3 sensors," in *Proceedings of IEE International Conference on Visual Information Engineering (VIE '05)*, pp. 1–7, Glasgow, Scotland, UK, April 2005.

[39] G. D. Finlayson, S. D. Hordley, and P. Morovic, "Colour constancy using the chromagenic constraint," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR '05)*, vol. 1, pp. 1079–1086, San Diego, Calif, USA, June 2005.

[40] G. D. Finlayson, S. D. Hordley, and M. S. Drew, "Removing shadows from images," in *Proceedings of 7th European Conference on Computer Vision-Part IV (ECCV '02)*, pp. 823–836, Copenhagen, Denmark, May 2002.

[41] H. J. Trussell, E. Saber, and M. Vrhel, "Color image processing [basics and special issue overview]," *IEEE Signal Processing Magazine*, vol. 22, no. 1, pp. 14–22, 2005.

[42] R. Lukac, B. Smolka, K. Martin, K. N. Plataniotis, and A. N. Venetsanopoulos, "Vector filtering for color imaging," *IEEE Signal Processing Magazine*, vol. 22, no. 1, pp. 74–86, 2005.

E. R. Davies: Machine Vision Group, Department of Physics, Royal Holloway, University of London, Egham, Surrey TW20 0EX, UK

*Email*: e.r.davies@rhul.ac.uk

D. Charles: Machine Vision Group, Department of Physics, Royal Holloway, University of London, Egham, Surrey TW20 0EX, UK
*Current address*: Fuji Film Electronic Imaging, Rickmansworth WD3 1RE, UK

*Email*: derekcharles@hotmail.com

# 12 Nonlinear edge detection in color images

Adrian N. Evans

## 12.1. Introduction

The widespread use of low-cost color digital cameras and the increasing availability of satellite imagery have brought about the situation where color and multispectral images are increasingly popular digital image formats. As a consequence, many of the basic image processing operations that have been developed for gray-scale images are now being extended to multichannel images. Edge detection is one of the most important tasks in image processing and its application to color images is a subject of much current interest [1, 2]. In gray-scale images, edge detection often performs a preprocessing role and is used in combination with other computer vision algorithms. For example, feature extraction techniques such as the Hough transform and active contour models require edge information as input. Alternatively, image segmentation can be achieved by applying the watershed transformation to a gradient function.

From an information theory perspective, as edge points typically constitute less than 5% of the total pixels, they are very rich in information. In this context edge detection can be viewed as an information filter that greatly reduces the number of pixels that have to be considered with little impact on the information content. As the volume of data for color and multispectral images is $m$ times that of gray-scale images, where $m$ is the number of channels, it can be argued that edge detection provides a greater advantage here than for the gray-scale case.

A naive approach to detecting color edges is to simply apply existing gray-scale techniques to the luminance component. However, this overlooks the fact that color edges cannot simply be modeled as discontinuities in intensity. For example, an edge can occur in a region of constant intensity when the chromaticity changes. Novak and Shafer [3] found that 90% of the edge points in a color image are also gray-scale edge points, and therefore, to detect the remaining 10%, a color edge detection scheme is required. The use of color edge detectors also better reflects the human visual system (HVS) where color edges play an important role in scene interpretation.

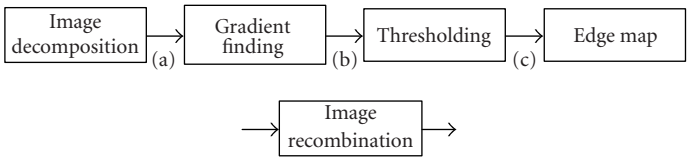



FIGURE 12.1. Classification of color edge detection algorithms after [4]. Inserting the image recombination step at positions (a), (b), or (c) gives rise to vector, multidimensional gradient, and output fusion methods, respectively.

This chapter first reviews the current state of the art in color edge detection. Nonlinear approaches to color edge detection typically treat color images as two-dimensional multichannel vector images and calculate their outputs based on vector differences. In addition, vector-based edge detectors may employ an explicit or implicit vector ordering prior to applying a distance metric. Both the ordering and differencing operations depend on the color space and the distance metric applied. Some of choices for these are discussed. The difference vector and vector gradient operators are then described. These operators calculate vector differences between color pixels according to their positions within a mask. An alternative approach is to select the color pixels to be differenced by using their position in an ordered list. In this class, color edge detectors based on vector order statistics (VOS) have been cited as one of the most compelling vector methods [4] and feature in several recent review papers [1, 5]. Their operation is discussed in detail in Section 12.5. However, despite their well-known qualities, they have a tendency to produce a bimodal response to a single edge and a new analysis of this phenomenon is presented. Color morphological gradient operators are a new class of color edge detectors that employ a pairwise vector ordering and can be viewed as an extension of the morphological gradient to color images. Their performance is evaluated both objectively and subjectively, in comparison with the VOS edge detectors.

## 12.2. Color edge detection

A useful classification of color edge detectors is given in [4], which groups color edge detection techniques into output fusion methods, multidimensional gradient methods, or vector methods, according to the position of the image recombination step in the processing chain, see Figure 12.1. Output fusion methods combine the results produced by applying gray-scale edge detectors to the individual color channels. Operations used in the combination process include the logical OR [6], effectively stating that a color edge exists if an edge is present in at least one of the color channels. Other authors have proposed a summation [7] or some form of weighted sum [8, 9].

Multidimensional gradient methods combined the gradients from the individual channels before the thresholding stage. Robinson determined the directional derivatives for each neighbor in each channel, giving $8 \times 3 = 24$ gradients,

and then selected the one with the largest magnitude. A more advanced approach first computes the derivatives of the gradient in the horizontal and vertical directions [10]. The products are then used to form a $2 \times 2$ matrix in each component, the matrices summed over all channels and the edge magnitude, and direction given by the principal eigenvalue and its eigenvector. Cumani [11], Chapron [12], and Alshatti and Lambert [13] have all used variations of this approach; for a comparison of these and other simpler techniques see [14].

Vector methods do not have to consider the problem of how to combine the channels to produce a final edge map as all their gradient finding operation is performed in the vector space. As such they are the most attractive of the three approaches. To determine the rate and direction of maximum change at each pixel, differential geometry can be employed [15, 16]. Using Bayesian theory, the maximum a posteriori criterion can be used to detect and locate edges in vector images [17]. Another approach makes use of probability distributions. For example, in [18] relative entropy is used as a dissimilarity measure between the probability distributions found locally and in a homogenous region. Alternatively, edge points can be defined by the location of density minima in multidimensional feature space, while the regions either side of the edge form clusters [19]. The compass edge detector uses vector quantization to produce a compact color signature that, unlike histograms, does not enforce an arbitrary partitioning of the color space [4]. In this impressive work, a circular window is split in half and the difference between color signatures on either side measured using the earth mover's distance. Repeating this operation at multiple orientations produces a maximum response and associated edge direction. Prefiltering can be employed to reduce noise prior to edge detection. Russo and Lazzari propose a multipass nonlinear filtering stage before a noise-protected edge detector based on fuzzy models [20].

## 12.3. Color spaces and distance measures

The output of color edge detectors that calculate vector differences depends on both the distance metric and the color space in which it is applied, and these areas have been the subject of recent research. This brief review of color spaces presented is focussed on color representations for edge detection and is not in any way intended to be comprehensive. For a more complete treatment see [2, Chapter 1], [21], and the references therein.

Cones found in the human eye provide the basis for photopic vision. They can be classified into three types according to the wavelength of light they are most sensitive to, providing the basis for the red, green, and blue (RGB) color space. As most color images are captured using a sensor array that responds to the primary colors, the RGB color space is widely used, despite the fact that this representation neglects the perceptual aspects of color. The direct use of the RGB space also avoids the need to perform color space conversion which can be computationally expensive. However, its color channels are highly correlated and it does not describe colors in a form well suited for human interpretation.

Color models that specify a color in terms of its hue, saturation, and brightness give a color description that corresponds well with human perception. Many such models have been proposed including hue, saturation, and intensity (HSI); hue, lightness, and saturation (HLS); and HSV. The conversion from RGB to any of these is achieved by a transformation from a rectangular to a polar coordinate system. Three-dimensional polar representations decouple the luminance from the chrominance components and are compatible with human intuition. However, as discussed in Section 11.10, many of them were not designed for image processing operations and are therefore difficult representations to work with. More recently, a family of improved HLS (IHLS) representations have been proposed and its compatibility with a number of color gradients has been assessed [22]. The opponent color spaces such as YUV and YIQ used in television and video applications also separate the chrominance and intensity information. One such space is the $xyY$ color space, designed for use with a color filtering scheme that is nonlinear in the $x$ and $y$ chrominance components and linear in the $Y$ component [23].

The CIE $L^*a^*b^*$ and $L^*u^*v^*$ color spaces were designed to be perceptually uniform and to separate the perceived lightness $L^*$ from the chrominance, given by the $a^*b^*$ or the $u^*v^*$ components, respectively. Although the two spaces differ in formulation, in both the Euclidean distance between two colors provides a good estimate of the perceptual distance between them. An important caveat to this is given in [4], namely, that the equivalence between the perceptual and Euclidean distances in the $L^*a^*b^*$ space only holds for small distances. Therefore the Euclidean distance at edges consisting of two disparate colors can classify the colors as different but the distance is not well quantified perceptually.

Many color edge detectors have been developed and evaluated in the RGB color space although few, if any, are restricted to this representation. Wesolkowski et al. have evaluated the performance of a number of vector edge detectors and multichannel implementations of the Roberts and Sobel operators in multiple color spaces [24]. The operators evaluated in this study include the vector gradient edge detector of [25] that was originally developed for use in the $L^*u^*v^*$ color space and is described in Section 12.4.3.

A separate issue from the color space used is the choice of metric to apply in a given space. Much of the work in this area is concerned with the development of distance measures that give improved performance in the RGB space, thus avoiding the need for expensive color space conversion. The most popular metric to assess the distance between two vectors $\mathbf{X}^i$ and $\mathbf{X}^j$ is the $L_p$ norm

$$\|\mathbf{X}\|_p = \left[ \sum_k |\mathbf{X}_k^i - \mathbf{X}_k^j|^p \right]^{1/p}, \tag{12.1}$$

where $p = 1, 2$ and $\infty$ are most commonly used, giving the city-block, euclidean, and chess-board distances, respectively.

Equation (12.1) assesses the difference between two vectors in the magnitude domain. An alternative is to consider differences in terms of orientation and the

angular difference $\theta$ between two vectors can be found by considering its cosine,

$$\cos\theta = \frac{\mathbf{X}^i \mathbf{X}^{jT}}{||\mathbf{X}^i|| ||\mathbf{X}^j||}. \tag{12.2}$$

The vector angle is employed by the chromaticity-preserving vector directional filters [26, 27] and can be combined with a magnitude-based measure to produce a generalized set of measures for assessing the similarity between vectors [28]. As a brief aside, [29] reports that neither $\theta$ nor $\cos\theta$ are suitable for direct use as a gradient measure and, instead, $\sin\theta$ can be used, given by a modified form of (12.2),

$$\sin\theta = \left(1 - \frac{\left(\mathbf{X}^i \mathbf{X}^{jT}\right)^2}{\mathbf{X}^i \mathbf{X}^{iT} \mathbf{X}^j \mathbf{X}^{jT}}\right)^{1/2}. \tag{12.3}$$

Combined distance measures have the advantage of responding to differences in both magnitude and angle. Color edge detectors that employ intensity-based and saturation-based approaches to combining angle and distance metrics are presented in [30]. In this work, the intensity-based approach favors the angle difference when the intensity of both pixels is high and the Euclidean distance when either or both pixels have low intensity. Similarly, when both pixels have a high saturation the vector angle is used, and the Euclidean distance is used when one pixel is low in saturation. Another combined measure that is also applied directly in the RGB color space is the combined difference $\text{Diff}_{\text{com}}$ given by

$$\text{Diff}_{\text{com}} = 1 - \left[1 - \frac{2}{\pi}\cos^{-1}\left(\frac{\mathbf{X}^i \cdot \mathbf{X}^j}{||\mathbf{X}^i|| ||\mathbf{X}^j||}\right)\right]\left[1 - \frac{||\mathbf{X}^i - \mathbf{X}^j||}{\sqrt{3 \cdot 255^2}}\right], \tag{12.4}$$

where the terms in square brackets assess the difference in angle and magnitude, respectively [31]. This measure was originally proposed for color image retrieval and includes normalization factors relating to the maximum angle and magnitude.

## 12.4. Color edge detectors based on vector differences

Many vector methods of color edge detection form their output from some combination of the vector differences between pixels contained within a local mask. These techniques use two approaches to the problem of determining which vectors from the mask are used to calculate the output. In the first, the vectors are chosen according to their position in the mask while, in the second, the vectors are selected according to their positions in a ranked list. Color edge detectors belonging to the first category have the advantage that, in addition to edge magnitude, they are able to estimate edge direction. These detectors are reviewed below. An estimate of edge direction is advantageous as it allows the nonmaxima suppression stage of the Canny edge detector [32] to be applied, producing thinner edges. One of the most widely reported classes of edge detectors belonging to the second

category are those based on VOS. These are described in detail in Section 12.5. Although the color morphological gradient (CMG) operators are also based on vector ordering and therefore belong to the latter class of approaches, they are also able to provide an estimate of the edge direction. The CMG edge detectors are described in Section 12.6.

### 12.4.1. Directional operators

Directional operators for color edge detection that, conceptually, are an extension of the well-known Prewitt edge operators for gray-scale images to color images were proposed in [33]. The Prewitt edge detector employs two orthogonal masks $P_H$ and $P_V$ that respond to horizontal and vertical gradients, respectively,

$$P_H = \frac{1}{3}\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}, \qquad P_V = \frac{1}{3}\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}. \tag{12.5}$$

The Prewitt edge detector therefore finds the average intensity of the regions marked +1 and −1 in (12.5) and then calculates their differences. Noting that the operators are not restricted to a $3 \times 3$ mask, the corresponding directional operators for color images first find the average color vectors $\mathbf{H}_-$, $\mathbf{H}_+$, $\mathbf{V}_-$, and $\mathbf{V}_+$ for the +1 and −1 regions in the horizontal and vertical masks. For a window size $(2w+1) \times (2w+1)$ located at pixel $(x_0, y_0)$ these are given by

$$\begin{aligned}
\mathbf{H}_-(x_0, y_0) &= \frac{1}{w(2w+1)} \sum_{y=y_0-w}^{y_0+w} \sum_{x=x_0-1}^{x_0-w} \mathbf{f}(x, y), \\
\mathbf{H}_+(x_0, y_0) &= \frac{1}{w(2w+1)} \sum_{y=y_0-w}^{y_0+w} \sum_{x=x_0+1}^{x_0+w} \mathbf{f}(x, y), \\
\mathbf{V}_-(x_0, y_0) &= \frac{1}{w(2w+1)} \sum_{y=y_0-1}^{y_0-w} \sum_{x=x_0-w}^{x_0+w} \mathbf{f}(x, y), \\
\mathbf{V}_+(x_0, y_0) &= \frac{1}{w(2w+1)} \sum_{y=y_0+1}^{y_0+w} \sum_{x=x_0-w}^{x_0+w} \mathbf{f}(x, y),
\end{aligned} \tag{12.6}$$

where $\mathbf{f}(x, y)$ is the color $(r, g, b)$. The directional operators are then given by

$$\begin{aligned}
\mathbf{H}(x_0, y_0) &= \mathbf{H}_+(x_0, y_0) - \mathbf{H}_-(x_0, y_0), \\
\mathbf{V}(x_0, y_0) &= \mathbf{V}_+(x_0, y_0) - \mathbf{V}_-(x_0, y_0),
\end{aligned} \tag{12.7}$$

and the change in color in the horizontal and vertical directions by the Euclidean norms $\|\mathbf{H}(x_0, y_0)\|$ and $\|\mathbf{V}(x_0, y_0)\|$, respectively. From these two scalars the edge

magnitude $M$ is found in an identical manner to that of the Prewitt operator by

$$M = \sqrt{||\mathbf{H}(x_0, y_0)||^2 + ||\mathbf{V}(x_0, y_0)||^2}. \tag{12.8}$$

The estimation of the direction is not quite so straightforward, because the scalar values $\|\mathbf{H}(x_0, y_0)\|$ and $\|\mathbf{V}(x_0, y_0)\|$ do not have any signs associated with them. Signed values of color contrast $H_s(x_0, y_0)$ and $V_s(x_0, y_0)$ can be found by considering whether the change in luminous energy is positive or negative such that

$$\begin{aligned} H_s(x_0, y_0) &= \begin{cases} ||\mathbf{H}(x_0, y_0)|| & \text{if } ||\mathbf{H}_+(x_0, y_0)|| \geq ||\mathbf{H}_-(x_0, y_0)||, \\ -||\mathbf{H}(x_0, y_0)|| & \text{otherwise,} \end{cases} \\ V_s(x_0, y_0) &= \begin{cases} ||\mathbf{V}(x_0, y_0)|| & \text{if } ||\mathbf{V}_+(x_0, y_0)|| \geq ||\mathbf{V}_-(x_0, y_0)||, \\ -||\mathbf{V}(x_0, y_0)|| & \text{otherwise,} \end{cases} \end{aligned} \tag{12.9}$$

and the edge direction $\theta$ is estimated by

$$\theta = \arctan\left(\frac{V_s(x_0, y_0)}{H_s(x_0, y_0)}\right) + k\pi, \tag{12.10}$$

where $k$ is an integer.

### 12.4.2. Difference vector operators

Difference vector (DV) operators calculate the maximum gradient across the central pixel. They work by dividing the window into subwindows at four orientations: horizontal, vertical, positive, and negative diagonals [34, 35]. The vector difference between the two subwindows provides a measure of the gradient at each orientation and the maximum vector provides the gradient magnitude and the associated direction. For a $3 \times 3$ window located at pixel $(x_0, y_0)$ the DV gradients in four directions are defined by

$$\begin{aligned} \mathrm{DV}_{0^\circ}(x_0, y_0) &= ||\mathbf{f}(x_0 + 1, y_0) - \mathbf{f}(x_0 - 1, y_0)||, \\ \mathrm{DV}_{45^\circ}(x_0, y_0) &= ||\mathbf{f}(x_0 + 1, y_0 - 1) - \mathbf{f}(x_0 - 1, y_0 + 1)||, \\ \mathrm{DV}_{90^\circ}(x_0, y_0) &= ||\mathbf{f}(x_0, y_0 + 1) - \mathbf{f}(x_0, y_0 - 1)||, \\ \mathrm{DV}_{135^\circ}(x_0, y_0) &= ||\mathbf{f}(x_0 - 1, y_0 - 1) - \mathbf{f}(x_0 + 1, y_0 + 1)||. \end{aligned} \tag{12.11}$$

The output difference vector is the maximum response such that

$$\mathrm{DV} = \max\left(\mathrm{DV}_{0^\circ}, \mathrm{DV}_{45^\circ}, \mathrm{DV}_{90^\circ}, \mathrm{DV}_{135^\circ}\right) \tag{12.12}$$

and the edge direction is that associated with the maximum response.

To improve the performance of the operators in the presence of impulsive and Gaussian noise, the window size can be increased although, for computational and localization reasons, a $5 \times 5$ window is preferred [35]. This gives 11 vectors in each subwindow if the center pixel is included in all subwindows, as recommended in [34]. A filter function can then be applied to the subwindows to produce the two vectors required by (12.11). Zhu et al. have investigated various nonlinear filters including the vector median, the vector mean, and the adaptive nearest-neighbor filter [35]. Instead of filtering the subwindows, a more computationally attractive approach is to prefilter the entire image using a $3 \times 3$ window and then apply (12.11) and (12.12), again using a $3 \times 3$ mask. In addition, this approach was shown to have a better noise suppression performance that of the subwindow-based methods. However, as discussed in Section 11.9, care must be taken when applying the median and other filters to color images as they can introduce edge shifts.

A further variation that reduces the thickness of diagonal edges and slightly reduces the computational complexity is to calculate only the difference vectors in the horizontal and vertical directions, giving

$$\mathrm{DV}_{hv} = \max\left(\mathrm{DV}_{0^\circ}, \mathrm{DV}_{90^\circ}\right). \tag{12.13}$$

A probabilistic evaluation has shown this approach to detect fewer false edges while finding virtually the same number of true edge points [2, 35]. In addition, its peak SNR performance on real images corrupted with mixed noise is only slightly worse than that produced by calculating the vector differences in all four directions.

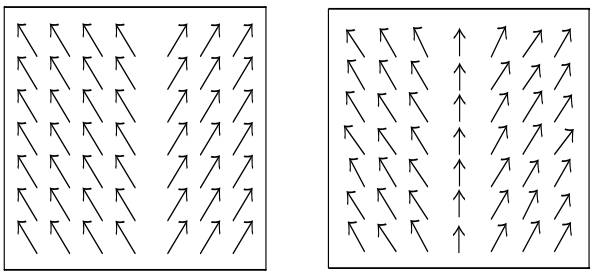
### 12.4.3. Vector gradient operators

The vector gradient (VG) edge detector calculates the maximum difference between the center pixel and its eight-connected neighbors. The VG at pixel $(x_0, y_0)$ is given by

$$\mathrm{VG}(x_0, y_0) = \max_{i \in \mathcal{N}(x_0, y_0)} \{||\mathbf{f}_i(x, y) - \mathbf{f}(x_0, y_0)||\}, \tag{12.14}$$

where $\mathcal{N}(x_0, y_0)$ is the set of neighbors of pixel $(x_0, y_0)$. This operator differs from the difference vector (DV) operator in that it calculates a noncentered difference. However, it still allows edge direction to be estimated from the direction of maximum gradient.

The technique was originally proposed for use with the $L^*u^*v^*$ color space that was specifically designed so that the similarity between any two colors is quantified by the Euclidean distance [25]. It has also been applied in the RGB color space using combined Euclidean distance and vector angle metrics [30], and a study of its performance in multiple color spaces was undertaken in [24].



(a) (b)

FIGURE 12.2. Ideal color edges consisting of vectors $V_1 = (-3, -5)$, $V_2 = (3, -5)$, and $V_3 = (0, -5)$: (a) ideal step edge and (b) ideal ramp edge. (Reproduced from Adrian N. Evans and Xin U. Liu, "A morphological gradient approach to color edge detection," IEEE Transactions on Image Processing, © 2006 IEEE.)

## 12.5. Vector order statistics color edge detectors

Color edge detectors based on VOS were proposed by Trahanias and Venetsanopoulos [26, 27] and represent an extension of the morphological edge operators for gray-scale images of [36] to multichannel images. VOS edge detectors employ reduced or aggregate ordering ($R$-ordering) to sort a set of $n$ vectors $\mathbf{X}^1, \mathbf{X}^2, \ldots, \mathbf{X}^n$ according to their aggregate distances, $d_i s$. Each $d_i$ is given by

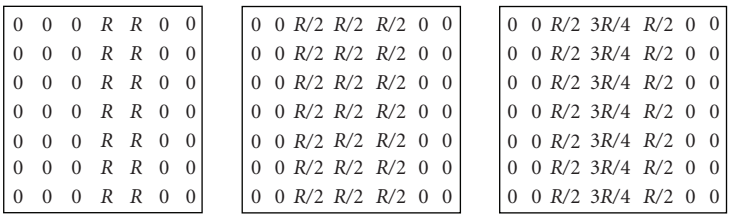
$$d_i = \sum_{k=1}^{n} \left\| \mathbf{X}^i - \mathbf{X}^k \right\|_p, \quad i = 1, 2, \ldots, n, \tag{12.15}$$

where $\| \cdot \|_p$ is a vector norm. When the vectors are ordered according to their $d_i s$ such that $d_{(1)} \leq d_{(2)} \leq \cdots \leq d_{(n)}$, the result is the ordered sequence $\mathbf{X}^{(1)} \leq \mathbf{X}^{(2)} \leq \cdots \leq \mathbf{X}^{(n)}$ in which $\mathbf{X}^{(1)}$ is the vector median [37] and $\mathbf{X}^{(n)}$ is the vector extremum. This ordering avoids the difficulty of distinguishing between the minimum and maximum vectors as both are simply high-ranking vectors and are termed extrema.

The simplest VOS edge detector is the vector range (VR) defined by

$$\text{VR} = \left\| \mathbf{X}^{(n)} - \mathbf{X}^{(1)} \right\|_p, \tag{12.16}$$

which provides a quantitative measure of the distance from the vector median to the highest-ranking vector. The VR is the underlying distance measure for the VOS edge detectors and its operation can be illustrated by considering its response to the ideal color step and ramp edges shown in Figure 12.2, using a $3 \times 3$ mask. The response of the VR to the ideal step edge shown in Figure 12.3(a) is 2 pixels wide and is unbiased. For the color ramp edge, the VR response in Figure 12.3(b) extends across 3 pixels and does not differentiate between the response of the true edge pixels and those of their neighbors [27]. This can be explained by reducing the VR to a single channel form where it can be seen that it measures the difference



(a) (b) (c)

FIGURE 12.3. Response of VOS edge detectors to ideal color edges using a $3 \times 3$ mask, where $R = \|V_1 - V_2\|_2 = 6$ and $R/2 = \|V_1 - V_3\|_2 = 3$: (a) VR response to step edge, (b) VR response to ramp edge, and (c) MVD response to ramp edge with $k = 3$ and $l = 6$.

between the median and either the highest or the lowest intensity depending on which is furthest from the median. For a ramp edge this is a noncentered difference and therefore may not always provide an accurate estimate of the true gradient.

As the output of the VR depends on the extremum vector, it is very sensitive to noise, and will respond to a single noisy pixel. To improve the robustness to exponential and impulsive noise, the VR can be modified by calculating the distances from the vector median to the $k$ highest-ranked vectors and then selecting the minimum of these, giving rise to the minimum vector range (MVR) color edge detector,

$$\text{MVR} = \min_j \left\{ \left\| \mathbf{X}^{(n-j+1)} - \mathbf{X}^{(1)} \right\|_p \right\}, \quad j = 1, 2, \ldots, k;\ k < n. \tag{12.17}$$

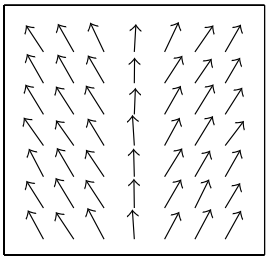
The MVR therefore rejects all but one of the $k$ highest-ranked vectors. In practice the value of $k$ can be selected by considering the number of pixels belonging to the minority population at an edge; for example, $k$ should be $\leq 10$ for a $5 \times 5$ mask.

Finally, a VOS edge detector that is also robust to short-tailed or Gaussian noise is obtained by replacing the vector median in (12.16) by the $\alpha$-trimmed mean of the $l$ lowest-ranked vectors. The resultant operator is known as the vector dispersion (VD) edge detector which, when combined with the MVR of (12.17), results in the minimum vector dispersion (MVD) color edge detector given by

$$\text{MVD} = \min_j \left\{ \left\| \mathbf{X}^{(n-j+1)} - \sum_{i=1}^{l} \frac{\mathbf{X}^{(i)}}{l} \right\|_p \right\}, \quad j = 1, 2, \ldots, k;\ k, l < n. \tag{12.18}$$

Although no formal definition exists for the parameter $l$, an intuitive interpretation is that it is the number of pixels in the majority population contained by the mask. This is analogous to the interpretation of $k$ for the MVR.

A further advantage of the color edge detectors employing the $\alpha$-trimmed mean is an improved performance for color ramp edges resulting from an increased response at the true edge pixel [27]. For example, when the MVD edge detector is applied to the ideal ramp edge of Figure 12.2(b) the true edge pixel has



| | | | | |
|---|---|---|---|---|
| 0.40 | 3.05 | 3.11 | 2.68 | 0.50 |
| 0.28 | 2.69 | 3.05 | 2.85 | 0.55 |
| 0.33 | 2.83 | 3.25 | 2.85 | 0.46 |
| 0.38 | 2.85 | 3.22 | 2.91 | 0.54 |
| 0.47 | 2.93 | 3.16 | 2.80 | 0.45 |

| | | | | |
|---|---|---|---|---|
| 0.33 | 3.12 | 2.69 | 2.85 | 0.48 |
| 0.37 | 2.85 | 2.83 | 2.98 | 0.53 |
| 0.41 | 2.94 | 2.83 | 2.94 | 0.43 |
| 0.43 | 2.94 | 2.72 | 3.06 | 0.55 |
| 0.44 | 2.91 | 2.89 | 2.98 | 0.44 |

(a) (b) (c)

FIGURE 12.4. Response of MVD edge detector to ramp edge corrupted by Gaussian noise ($\sigma = 0.3$): (a) noisy edge; (b) and (c) MVD response to central $5 \times 5$ pixels from (a) with parameters $k = 3$, $l = 3$, and $l = 6$, respectively.

a raised response for $4 \leq l \leq 6$. The maximum MVD response to the true edge occurs with parameters $1 \leq k \leq 3$ and $l = 6$, when the output at the edge is $\|V_2 - (3V_1 + 3V_3)/6\| = 3R/4$, where $R = \|V_1 - V_2\|_2 = 6$; this is illustrated in Figure 12.3(c).

Extensive experimentation has shown the MVD to be the best performing VOS edge detector in the presence of noise [26, 27]. However, it has also been observed that it can often produce a double response to a single ramp edge [38]. The $\alpha$-trimmed mean of the $l$ lowest-ranked vectors used by the VD and MVD edge detectors requires the vectors on each side of the edge to be ranked in adjacent positions if it is to produce a raised response. For the ideal ramp edge of Figure 12.2(b), the maximum MVD response at the true edge position of $3R/4$ is achieved when $l = 6$ and the ordered sequence is $[V_3, V_3, V_3, V_1, V_1, V_1, V_2, V_2, V_2]$. However, when the edge is corrupted by a small amount of Gaussian noise, such as in Figure 12.4(a), the fragile ordering of the 6 highest-ranking vectors from the noise-free case is lost. As a consequence, the MVD can produce a lower response at the true edge position than at the neighboring positions.

Figure 12.4 illustrates this point and shows the MVD response for the central $5 \times 5$ pixels of Figure 12.4(a) for $l = 3$ and $l = 6$, with $k$ set to 3. When $l = 3$, the response in Figure 12.4(b) exhibits a slightly raised response to the true edge in line with expectations. However, the situation in Figure 12.4(c) shows that when $l = 6$ the response at the true edge positions is reduced from the noise-free value of $3R/4 = 4.5$ to values in approximate range of 2.7 to 2.9. The true edge positions now have a lower response than their neighbors and when a threshold is applied this can produce false edge points on either side of missed true edge points.

Finally, it should be noted that in contrast to the DV and VG operators of the multidimensional gradient methods of [25, 35], the MVD edge detector only provides edge magnitude and not edge direction as there is no physical position in the mask associated with the $\alpha$-trimmed mean. Even the simpler VR and MVR edge detectors are unable to provide a reliable estimate of edge orientation as the direction from the vector median to a high-ranked vector will not necessarily coincide with the true edge direction.

### 12.6. Color morphological gradient operators

A different approach to vector ordering is provided by the color morphological gradient (CMG) edge detectors [38, 39]. These operators were inspired by the classic morphological gradient operator for gray-scale images, given by the difference between a dilation and an erosion [40]. For a structuring element $B$, the morphological gradient (MG) is given by

$$\mathrm{MG} = \delta_B(f) - \epsilon_B(f). \tag{12.19}$$

As $\delta_B(f)$ and $\epsilon_B(f)$ are given by the maximum and minimum pixels, respectively, (12.19) cannot be applied directly to color images. However, if the MG is expressed by the alternative form

$$\begin{aligned} \mathrm{MG} &= \max_{x\in B}\{f(x)\} - \min_{y\in B}\{f(y)\} \\ &= \max\left(\left|f(x) - f(y)\right|\right) \quad \forall x, y \in B, \end{aligned} \tag{12.20}$$

the MG of color images can easily be found by replacing the absolute operation by a norm. The CMG is therefore given by

$$\mathrm{CMG} = \max_{i,j\in\mathcal{X}} \left(\left|\left|\mathbf{X}^i - \mathbf{X}^j\right|\right|_p\right), \tag{12.21}$$

where $\mathcal{X} = [\mathbf{X}^1, \mathbf{X}^2, \ldots, \mathbf{X}^n]$ is the set of $n$ vectors contained within the structuring element $B$ and $\|\cdot\|_p$ is an $L_p$ norm. The CMG therefore identifies the maximum and minimum vectors as the two vectors that are furthest apart but does not distinguish between them.

Removing the two vectors associated with the CMG and applying (12.21) to the remaining vectors produces another gradient estimate. Repeatedly applying this process results in a pairwise vector ordering scheme that is the basis of the other CMG operator, namely, the robust color morphological gradient (RCMG). The pairwise vector ordering scheme uses the distances

$$d_s = \left|\left|\mathbf{X}_1^{(s)} - \mathbf{X}_2^{(s)}\right|\right|_p, \quad s = 0, 1, \ldots, \frac{n}{2} - 1, \tag{12.22}$$

where $\mathbf{X}_1^{(s)}$ and $\mathbf{X}_2^{(s)}$ are the two vectors associated with the maximum distance in (12.21) after $s$ pairs of vectors have been removed. Using the distances $d_0 \geq d_1 \geq \cdots \geq d_{(n/2)-1}$, the pairwise vector ordering can be defined by

$$\mathbf{X}^{(1)} \leq \mathbf{X}^{(2)} \leq \cdots \leq \mathbf{X}^{(n-1)} \leq \mathbf{X}^{(n)}, \tag{12.23}$$

where $\mathbf{X}^{(s+1)}$ and $\mathbf{X}^{(n-s)}$ are the two vectors associated with distance $d_s$ and are interchangeable. The vector norms are then ordered by

$$\left|\left|\mathbf{X}^{(1)} - \mathbf{X}^{(n)}\right|\right|_p \geq \left|\left|\mathbf{X}^{(2)} - \mathbf{X}^{(n-1)}\right|\right|_p \geq \cdots \geq \left|\left|\mathbf{X}^{(n/2)} - \mathbf{X}^{((n/2)+1)}\right|\right|_p. \tag{12.24}$$

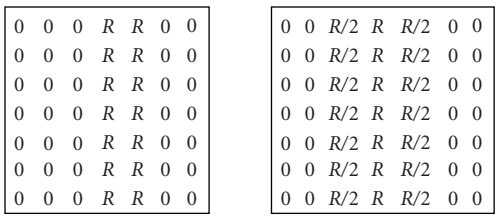



FIGURE 12.5. CMG response to ideal color edges of Figure 12.2: (a) step edge and (b) ramp edge. (Figure 12.5(b) is reproduced from Adrian N. Evans and Xin U. Liu, "A morphological gradient approach to color edge detection," IEEE Transactions on Image Processing, © 2006 IEEE.)

Equations (12.22)–(12.24) can be used to produce a pairwise vector ordering for $n$ vectors, where $n$ is even. However, like the standard median, for even $n$ there are two mid-ranked vectors and if the median vector is required, recourse to the somewhat unsatisfactory convention of choosing the midpoint between the two central vectors is necessary. When $n$ is odd there is one unpaired vector remaining after the other $n-1$ vectors are ordered. This vector is assigned the distance $d_{(n+1)/2} = \|\mathbf{X}^{(n+1)/2} - \mathbf{X}^{(n+1)/2}\|_p = 0$ and ranked in position $(n+1)/2$. It therefore follows that, for $n$ odd, the final unpaired vector is the median vector.

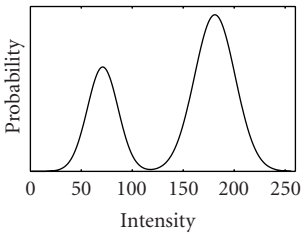
The CMG of (12.21) is the simplest operator to employ this pairwise ordering, and Figure 12.5 presents its response to the ideal color step and ramp edges of Figure 12.2 using a $3 \times 3$ mask. The CMG response to the step edge is unbiased and identical to that of the VOS edge detectors. The ramp edge response is 3 pixels wide but is raised from $R/2$ to $R$ at the true edge position; this compares favorably with the MVD response of Figure 12.3(c) which has a maximum raised response of $3R/4$. This improved ramp edge performance can be explained by considering the single channel form of the CMG, which is the difference between the maximum and minimum pixels and, for the ramp edge, measures the centered difference. The CMG performs exactly the same operation on vector-valued images and defines the maximum and minimum vectors as the two vectors that are furthest apart without uniquely identifying which is which.

Although the CMG produces a favorable response on ideal edges, its output depends on outlying vectors and is sensitive to image noise. This problem is overcome by the RCMG that outputs the distance $\|\mathbf{X}^{(s+1)} - \mathbf{X}^{(n-s)}\|_p$ for $s = 0, 1, \ldots, \lfloor (n/2) - 1 \rfloor$, a process equivalent to removing $s$ outlying vector pairs from $\mathcal{X}$ and finding the CMG of the remainder. The RCMG is therefore defined by

$$\text{RCMG} = \max_{i,j \in \{\mathcal{X} - \mathcal{R}_s\}} \left\{ \left\| \mathbf{X}^i - \mathbf{X}^j \right\|_p \right\}, \tag{12.25}$$

where $\mathcal{R}_s$ is the set of $s$ pairs of vectors removed. In this context, the CMG operator of (12.21) can bethought of as a special case of the RCMG for $s = 0$.

The performance of the RCMG operator in noise is now considered. Figure 12.6(a) presents the intensity distribution of a single channel step edge corrupted




(a)

| | | | | |
|---|---|---|---|---|
| 0.53 | 3.51 | 5.99 | 3.21 | 1.11 |
| 0.68 | 3.38 | 6.02 | 3.39 | 0.97 |
| 0.82 | 3.21 | 6.35 | 3.49 | 0.83 |
| 0.87 | 3.24 | 6.42 | 3.38 | 1.05 |
| 0.76 | 3.24 | 6.42 | 3.21 | 0.63 |

(b)

FIGURE 12.6. (a) Distribution of intensities at step edge corrupted by Gaussian noise and (b) response of RCMG to central $5 \times 5$ pixels from Figure 12.4(a) using a $3 \times 3$ mask with $s = 1$.

by Gaussian noise. It has two populations on either side of the edge centered on intensity values of 70 and 180, giving an ideal difference of 110. The response of the RCMG will be closest to this distance when its output is formed from one pixel from each of the two modes of the distribution. To approximate this, the approach presented in [39] is to set the parameter $s$ so that the output is the median of the centered differences across the edge. The number of centered differences is determined by the number of pixels in the minority population. For example, with a $3 \times 3$ mask there are 3 centered differences and setting $s = 1$ selects their median. The RCMG response to the central $5 \times 5$ pixels from the noise corrupted ramp edge of Figure 12.4(a) with $s = 1$ is given in Figure 12.6(b) and shows the responses at the true edge positions to be close to the noise-free response of $R = 6$. Compared with the noise-free case, the RCMG responses on either side of the edge are slightly increased and the responses away from the edge are raised by a greater amount. Overall, the differences between the responses at and away from the edge compare favorably with those of the MVD.

The RCMG is also robust to impulsive noise as outlying vectors are high ranking and are typically contained in the vector pairs removed. If a nonoutlying vector is removed, the output will only be slightly affected as other, similar vectors will exist. To find the median centered difference in the presence of impulsive or exponential noise, it is necessary to remove some outlying pairs of vectors before finding the median of the remaining centered differences across the edge. This can be achieved by increasing the value of the parameter $s$ and hence the number of pairs of vectors removed. However, for the RCMG operator to work as designed, $s$ should not be greater than the number of pixels in the minority population. For a $5 \times 5$ mask this restricts $s$ to the range $4 \leq s \leq 9$ [39].

The performance of many edge detectors can be improved by applying the nonmaximal suppression and thresholding with hysteresis stages of the Canny edge detector [32]. For these steps to be appropriate for the CMG operators, two criteria must be met. Firstly, the peak response of the color edge detector should occur at the true edge position and, secondly, the operators must be able to provide an estimate of the edge direction. From the theoretical analysis above, the CMG and RCMG operators meet the first criterion and an evaluation to determine that this is also the case for natural images is undertaken in Section 12.7.2 below.

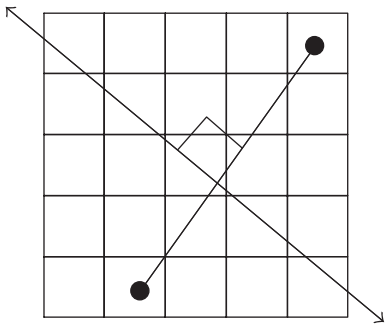


FIGURE 12.7. CMG operators edge direction is given by the normal to the line joining the pixel positions of the vector pair which are furthest apart.

As the CMG operators select their vectors according to their differences in a ranked list, it is not immediately obvious that they are able to estimate an edge's direction as well as its magnitude. However, an approximate estimate of edge direction can be found by considering the fact that the two pixels that give rise to the centered difference come from either side of the edge. Edge direction can then be estimated by first constructing a line joining the two pixels that give rise to the RCMG output and then defining the edge direction as the normal to this line, see Figure 12.7. Provided the two pixels used to define the gradient lie on opposite sides of the edge, the error in the direction estimate will be bounded by 45°. Therefore, although the edge direction is only approximate, it is suitable for use with nonmaximal suppression schemes where only a quantized edge direction is required.

## 12.7. Results and evaluation

In this section, the performance of the color edge detectors that employ vector ordering is evaluated. The operators compared are those based on VOS (Section 12.5) and the CMG operators that employ the pairwise ordering described in Section 12.6. The performance of the color edge detectors is evaluated in application to both simulated and natural images using the Euclidean distance and the RGB color space. A simulated image is useful as it contains known edges and can therefore be used to provide quantitative performance information, for example, in the presence of artificial noise. As a subjective evaluation in application to natural color test images remains an important test procedure, this is also undertaken. A mask size of $5 \times 5$ was used for both edge detectors as this represents a good compromise between performance and computational complexity. For all experiments the parameter value of $s = 8$ suggested in [39] is used for the RCMG. In [26] the suggested ranges for the MVD parameter values are $7 \leq k \leq 10$ and $10 \leq l \leq 15$. More recently, Androutsos et al. recommended $k = 6$ and $l = 10$ as the best MVD parameter settings [34]. For the experiments presented here, $k$ was set to 7 and $l$ to 10 as this provides a good compromise.

(a) (b)

FIGURE 12.8. Simulated color test image: (a) original image and (b) ideal edge response.

### 12.7.1. Robustness to noise

The performance of the edge detectors in the presence of noise is investigated by using a simulated image in conjunction with a measure of edge deviation. To this end, the $(128 \times 128)$-pixel color test image shown in Figure 12.8(a) was corrupted by Gaussian and impulsive noise. The noise was additionally classified as either independent or correlated, with a correlation factor $\rho = 0.5$. Pratt's figure of merit (FOM) [41] is used to provide a quantitative measure of the quality of the edges and is given by

$$\text{FOM} = \frac{1}{\max\{I_D, I_I\}} \sum_{k=1}^{I_D} \frac{1}{1 + \alpha(d_k)^2}, \tag{12.26}$$

where $I_D$ and $I_I$ are the number of detected and ideal edge points, $d_k$ is the distance from the $k$th detected edge point to the nearest true edge point and $\alpha$ $(> 0)$ is a scaling constant. The ideal edge map required by the FOM is shown in Figure 12.8(b) and was generated by marking all pixels in Figure 12.8(a) with any differently colored neighbors as edge points. This contrasts with other approaches where the response of color edge detectors to a noise-free image is used. Using a theoretical response provides a sterner test as even when no noise is present, edge detectors can produce a nonideal response, particularly at corners.

Figure 12.9 presents the FOM results for Gaussian and impulsive noise with $\alpha$ set to 0.2. The FOM values for each edge detector were obtained by adjusting a global threshold until the maximum FOM value resulted. Only the MVD and RCMG operators were evaluated as the VR and CMG operators are not designed to have any noise robustness. The FOM performance of both edge detectors for Gaussian noise is similar, particularly around 12 dB for independent noise and 14.5 dB for correlated noise. Away from these values the FOM for the RCMG is

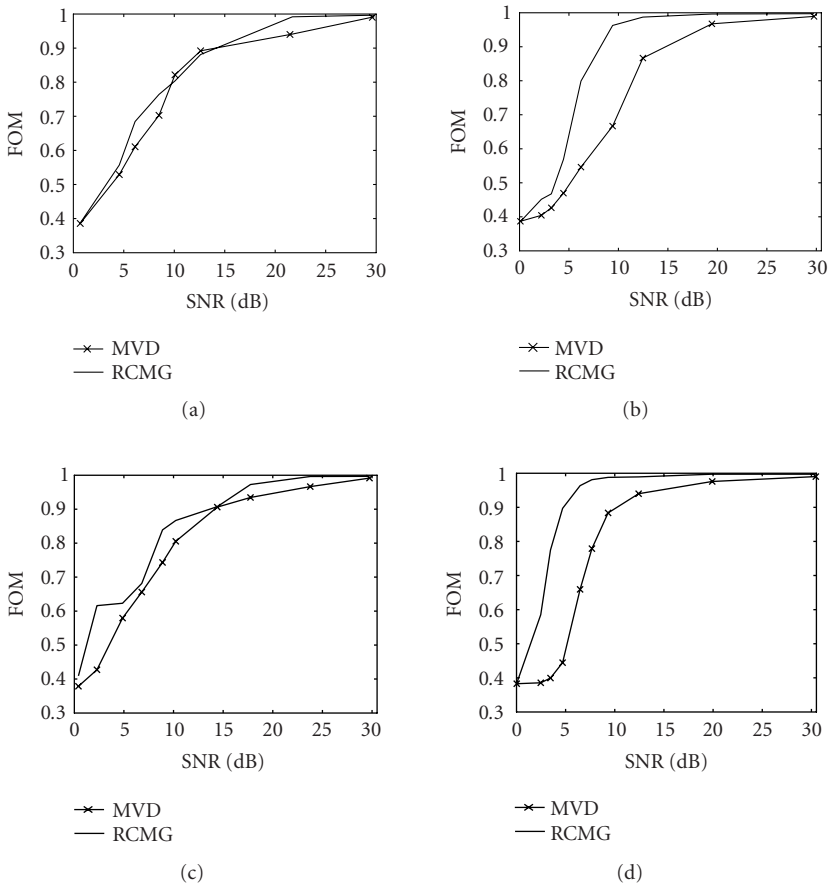



FIGURE 12.9. FOM results for RCMG and MVD color edge detectors: (a) Gaussian independent noise, (b) impulsive independent noise, (c) Gaussian correlated noise, and (d) impulsive correlated noise. (Reproduced from Adrian N. Evans and Xin U. Liu, "A morphological gradient approach to color edge detection," IEEE Transactions on Image Processing, © 2006 IEEE.)

above that of the MVD. For impulsive noise the RCMG has an FOM that is significantly higher than that of the MVD over the entire 0–30 dB range for both the independent and correlated cases. To illustrate this difference in FOM performance, the edge detection results for 8% and 15% of correlated impulsive noise are shown in Figure 12.10. At a noise level of 8% both edge detectors are able to find the true edges while effectively suppressing noise. However, the RCMG result has thinner, better-defined edges and fewer noise responses. When the noise level is increased to 15% the robustness of the RCMG is clearly demonstrated by the edge result, which shows little additional degradation above that of the 8% noise result. This is confirmed by the corresponding FOMs of 0.9933 and 0.9810 for 8% and 15% noise, respectively. In contrast, the MVD result for 15% noise exhibits thicker edges, noise points, and missing true edge points, and its FOM is reduced

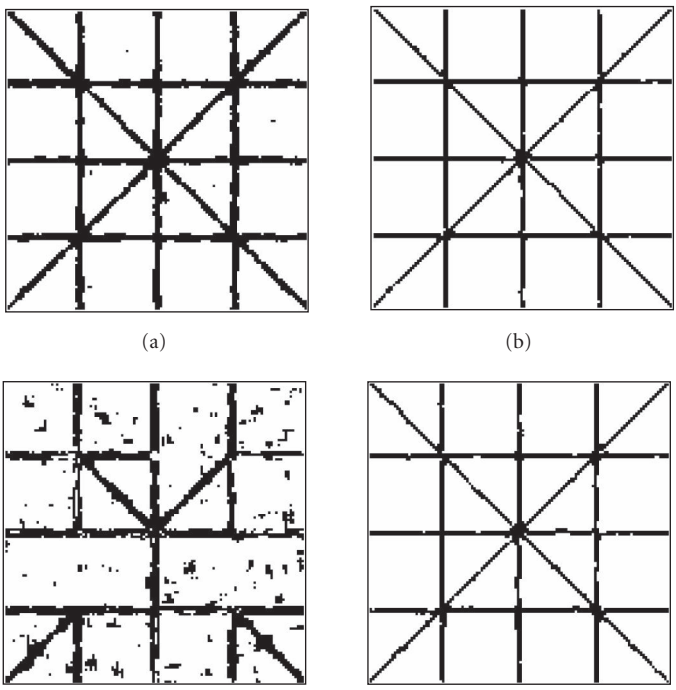


FIGURE 12.10. Color edge detector results for simulated image corrupted by correlated ($\rho = 0.5$) impulsive noise with corresponding FOM values shown bracketed: (a) MVD for 8% noise (0.9202), (b) RCMG for 8% noise (0.9933), (c) MVD for 15% noise (0.7789), and (d) RCMG for 15% noise (0.9810). (Figure 12.10(c) and Figure 12.10(d) are reproduced from Adrian N. Evans and Xin U. Liu, "A morphological gradient approach to color edge detection," IEEE Transactions on Image Processing, © 2006 IEEE.)

to 0.7789. The MVD 15% correlated impulsive noise result also has some instances where the edge center is missed while its neighboring pixels are marked as edges.

### 12.7.2. Edge localization

If the nonmaximal suppression stage of the Canny edge detector is to be applied to produce single-pixel thickness edges, it is important that the peak response occurs at the edge center. This is investigated for natural images by studying the responses produced by the color edge detectors for cross-sections across ramp edges. Numerous edges from a variety of test images have been investigated and the results for an edge position from the Airplane test image shown in Figure 12.11 are representative of those obtained.

Figure 12.11(b) shows the response of the CMG and VR edge detectors for the marked portion of column 63 from Figure 12.11(a). This location was chosen as it is a relatively strong edge and produces a typical response. A $5 \times 5$ mask was used for both edge detectors, and to help interpretation the luminance values are

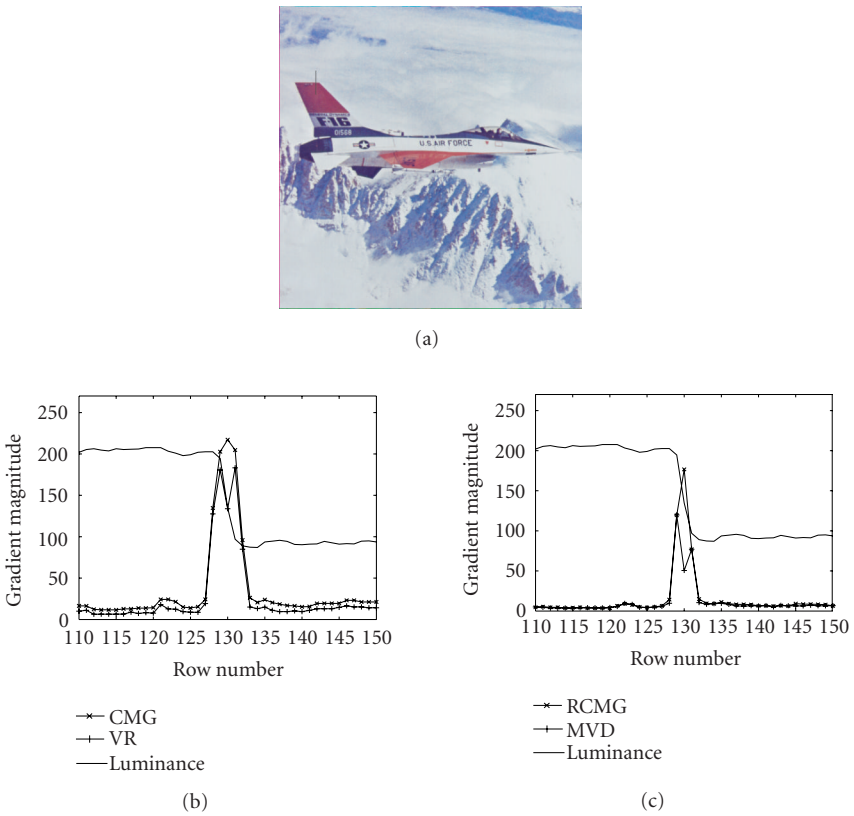



FIGURE 12.11. Behavior of color edge detectors at step edges. (a) Airplane with edge position marked, (b) CMG and VR results, and (c) RCMG and MVD results.

also shown. Away from the edges both edge detectors produce similar responses, with the magnitude of the CMG above that of the VR as would be expected from the theory. Both edge detectors show an increased response at the edge. However, the VR has a local minimum at the edge center showing that, for this mask size, its noncentered difference does not provide a good estimate of the true gradient. In contrast, the CMG response across the edge is well behaved, showing a single peak located at the center of the edge.

The responses of the RCMG and the MVD at the same edge position are shown in Figure 12.11(c), using the parameters detailed in Section 12.7.1 above. Compared with Figure 12.11(b), the MVD and RCMG responses away from the edge are reduced in magnitude and are very similar. This shows that the RCMG and MVD produce comparable results in homogeneous regions. The response of both detectors to the edge is also slightly reduced and is more localized. Here, the RCMG response is well behaved and marks the edge center with its peak response. In contrast the response of the MVD is still bimodal with a local minimum at the

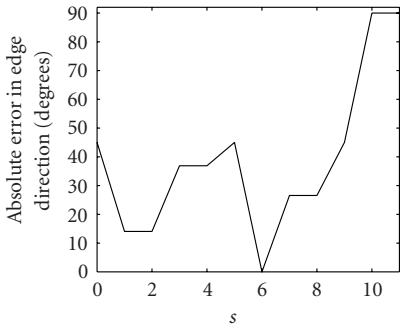
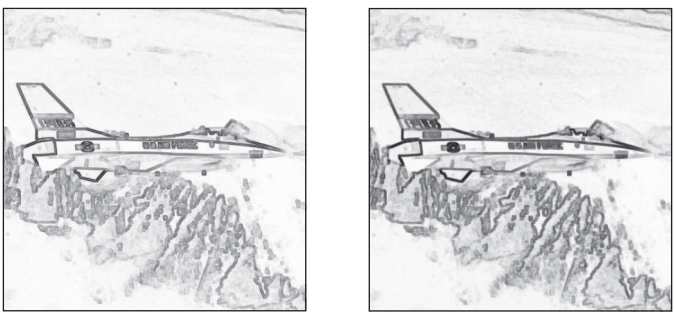



FIGURE 12.12. Variation of edge direction error with $s$ for the edge position marked in Figure 12.11(a).



(a) (b)

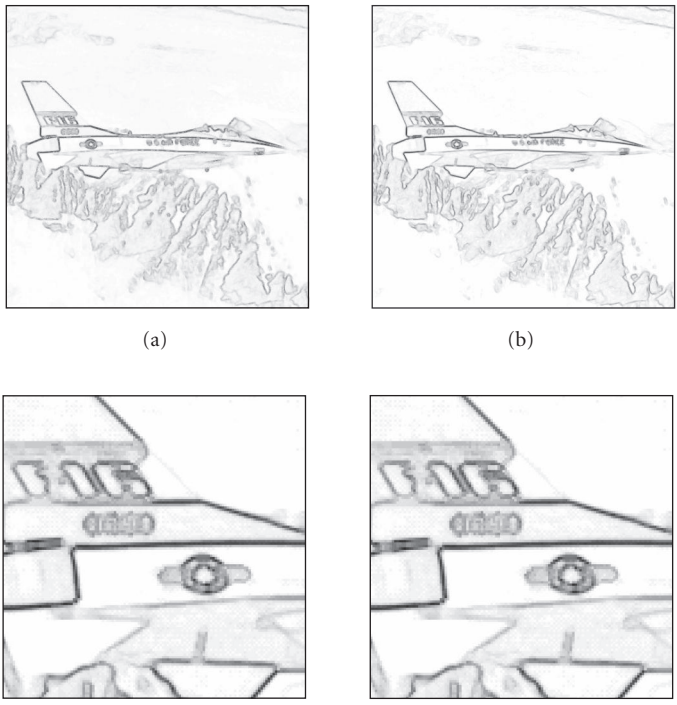
FIGURE 12.13. Color edge results for Airplane image: (a) VR result and (c) CMG result.

edge center, either because the $5 \times 5$ mask used is too small a scale or as a result of the effect described in Section 12.5, or a combination of both.

The performance of the edge direction estimation for the RCMG is also studied. Figure 12.12 shows the variation in edge direction error with $s$ for the edge marked in Figure 12.11(a). The result shows that for $0 \leq s \leq 9$ the absolute error is $\leq 45°$. This conforms with expectations as up to $s = 9$, at least one vector from the minority population within a $5 \times 5$ mask remains, and a reasonable estimate of edge direction can be made. When $s \geq 10$ the RCMG no longer calculates a centered difference and the direction estimates are unreliable. A more comprehensive test of the accuracy of the edge direction estimates is to assess the edge direction estimates in conjunction with a nonmaximal suppression scheme, and this is evaluated in the next section.

### 12.7.3. Natural images

To provide an illustration of the qualitative performance of the VOS and CMG edge detectors, they are applied to the Airplane and Peppers color test images,



(a) (b)

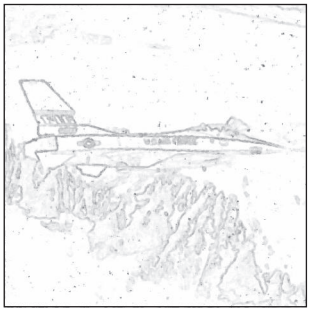
(c) (d)

FIGURE 12.14. Color edge results for Airplane image: (a) MVD result, (b) RCMG result, (c) and (d) close-ups of (a) and (b), respectively.

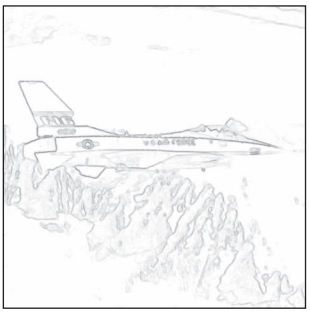
as these images are representative of those evaluated. Figure 12.13 shows the results produced by the VR and MVD edge detectors for the Airplane image of Figure 12.11(a) using a $5 \times 5$ mask. The edge images are normalized to a 0–255 range and show that the CMG has a stronger edge response, while the VR produces a double response at many of the edge positions. Both edge operators produce edge responses that are several pixels wide in which it is difficult to identify the true edge position. This situation is improved by the MVD and RCMG operators, which produce thinner, better-defined edges, see Figure 12.14. However, close-ups of a $128 \times 128$ pixels region towards the rear of the plane in Figure 12.14(c) show that the MVD operator often produces a low response at the true edge position with stronger edge responses on either side. In contrast, the RCMG result in Figure 12.14(d) shows thinner, more continuous edges.

To study the behavior of the color edge detectors in the presence of noise, the Airplane image is corrupted with 10% correlated ($\rho = 0.5$) impulsive noise, see Figure 12.15(a). Comparison of the MVD and RCMG results, given in Figures 12.15(b) and 12.15(c), respectively, with those of Figure 12.14 shows the performance of the RCMG to be only slightly affected by this noise level. In contrast, the

(a)



(b)



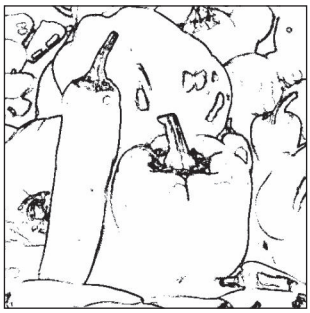
(c)

FIGURE 12.15. Color edge results for Airplane image corrupted by 10% correlated ($\rho = 0.5$) impulsive noise: (a) original image, (b) MVD result, and (c) RCMG result.

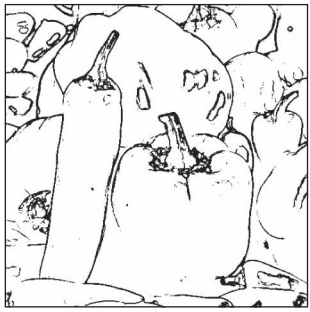
MVD result shows a degraded response at edges and many spurious edge points resulting from noise.

Thresholded edge maps are evaluated in application to the Peppers test image. Figure 12.16 shows the original image and the results of the MVD and RCMG operators, obtained by adjusting a global threshold to produce the best subjective results. Although both edge detectors produce similar edge maps, the MVD result in Figure 12.16(b) generally has thicker, less-continuous edges than the RCMG response of Figure 12.16(c). The advantages of using the nonmaximal suppression and thresholding with hysteresis stage of the Canny edge detector are illustrated for the RCMG edge detector in Figure 12.17, using the definition of edge direction described in Section 12.6. The RCMG edge result in Figure 12.17(a) shows the operator to perform well at the majority of edges in the image. The only exception is in the low-intensity regions, for example, the bottom left of the central green pepper. When the combined distance measure of (12.4) is used with the RCMG the edge map is improved to that shown in Figure 12.17(b). Here, the challenging edge points at the bottom of the central green pepper have been detected with little adverse impact elsewhere in the image. The continuity of the edges in Figure 12.17 confirms that the method of estimating edge direction described in Section 12.6

(a)



(b)



(c)

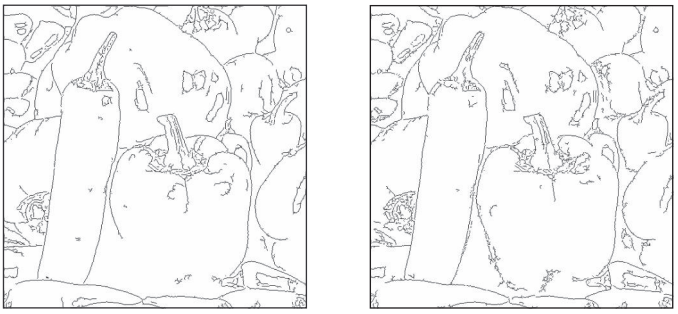
FIGURE 12.16. Thresholded color edge results for Peppers image: (a) original image, (b) MVD result, and (c) RCMG result. (Figure 12.16(c) is reproduced from Adrian N. Evans and Xin U. Liu, "A morphological gradient approach to color edge detection," IEEE Transactions on Image Processing, © 2006 IEEE.)

produces results that are sufficiently accurate to allow the nonmaximal suppression stage to work successfully.

## 12.8. Summary

A review of nonlinear edge detection techniques for application to color images has been presented. Compared with the large volume of work addressing the problem of edge detection in gray-scale images, color edge detection has received limited attention. This review has focussed on vector methods of color edge detection that treat color images as two-dimensional vector fields and thus avoid the problems associated with combining the results from individual channels after the gradient finding or thresholding stages. In particular, techniques that assess the color gradient by applying a metric or distance measure to vectors contained within a local window have been examined.

One class of approaches selects the vectors to be differenced according to their positions within the mask, either calculating a centered or noncentered difference according to whether the center pixel is included or not. This approach has the



(a) (b)

FIGURE 12.17. RCMG edge maps for Peppers image after nonmaximal suppression and thresholding with hysteresis: (a) using Euclidean distance and (b) using combined distance metric of (12.4). (Reproduced from Adrian N. Evans and Xin U. Liu, "A morphological gradient approach to color edge detection," IEEE Transactions on Image Processing, © 2006 IEEE.)

advantage of also being able to provide an estimate of edge direction. Alternatively, the vectors can be ordered and their difference can be assessed according to their rank. Here, the most widely reported techniques are those based on VOS. Section 12.5 describes the VOS operators and also includes a new analysis that attempts to explain the undesirable tendency of the operators to produce a double response to a single edge.

A new approach to ranking vectors, termed pairwise ordering, has been introduced and its use in the development of color edge detectors has been described. The resultant CMG operators identify the maximum and minimum vectors but do not distinguish between them. However, as a measure of gradient is given by the difference between maximum and minimum values, the inability to unambiguously identify the maximum and minimum vectors is not important; in edge detection it is the difference between them that matters. The RCMG edge detector based on this ordering has been demonstrated to be robust to Gaussian and impulsive noise and also has the advantage of being able to estimate edge direction. CMG operators can be applied in the RGB color space and have good edge localization properties. As such, they offer good performance with relatively low computation cost.

With the increased importance of color imagery and image analysis, the requirement for high-performance, computationally efficient color edge detection algorithms is undoubtedly on the increase. The techniques presented in this chapter provide a snapshot of the current state of development and may well prove to be the starting point for future research effort.

## Acknowledgments

Dr. A. N. Evans gratefully acknowledges the contribution made by Xin U. Liu to the results presented in Section 12.7. He is also grateful for the permission from the IEEE to reprint some figures from [39].

## Bibliography


[1] A. Koschan and M. Abidi, "Detection and classification of edges in color images," *IEEE Signal Processing Magazine*, vol. 22, no. 1, pp. 64–73, 2005.

[2] K. N. Plataniotis and A. N. Venetsanopoulos, *Color Image Processing and Applications*, Springer, New York, NY, USA, 2000.

[3] C. L. Novak and S. A. Shafer, "Color edge detection," in *Proceedings of DARPA Image Understanding Workshop*, vol. 1, pp. 35–37, Los Angeles, Calif, USA, February 1987.

[4] M. A. Ruzon and C. Tomasi, "Edge, junction, and corner detection using color distributions," *IEEE Transactions on Pattern Analaysis and Machine Intelligence*, vol. 23, no. 11, pp. 1281–1295, 2001.

[5] R. Lukac, B. Smolka, K. Martin, K. N. Plataniotis, and A. N. Venetsanopoulos, "Vector filtering for color imaging," *IEEE Signal Processing Magazine*, vol. 22, no. 1, pp. 74–86, 2005.

[6] J. Fan, D. K. Y. Yau, A. K. Elmagarmid, and W. G. Aref, "Automatic image segmentation by integrating color-edge extraction and seeded region growing," *IEEE Transactions on Image Processing*, vol. 10, no. 10, pp. 1454–1466, 2001.

[7] M. Hedley and H. Yan, "Segmentation of color images using spatial and color space information," *Journal of Electronic Imaging*, vol. 1, no. 4, pp. 374–380, 1992.

[8] T. Carron and P. Lambert, "Color edge detector using jointly hue, saturation and intensity," in *Proceedings of IEEE International Conference on Image Processing (ICIP '94)*, vol. 3, pp. 977–981, Austin, Tex, USA, November 1994.

[9] T. Carron and P. Lambert, "Fuzzy color edge extraction by inference rules: quantitative study and evaluation of performances," in *Proceedings of IEEE International Conference on Image Processing (ICIP '95)*, vol. 2, pp. 181–184, Washington, DC, USA, October 1995.

[10] S. Di Zenzo, "A note on the gradient of a multi-image," *Computer Vision Graphics and Image Processing*, vol. 33, no. 1, pp. 116–125, 1986.

[11] A. Cumani, "Edge detection in multispectral images," *CVGIP: Graphical Models and Image Processing*, vol. 53, no. 1, pp. 40–51, 1991.

[12] M. Chapron, "A color edge detector based on statistical rupture tests," in *Proceedings of IEEE International Conference on Image Processing (ICIP '00)*, vol. 2, pp. 820–823, Vancouver, BC, Canada, September 2000.

[13] W. Alshatti and P. Lambert, "Using eigenvectors of a vector field for deriving a second directional derivative operator for color images," in *Proceedings of 5th International Conference on Computer Analysis of Images and Patterns (CAIP '93)*, vol. 719 of *Lecture Notes in Computer Science*, pp. 149–156, Budapest, Hungary, September 1993.

[14] A. Koschan, "A comparative study on color edge detection," in *Proceedings of 2nd Asian Conference on Computer Vision (ACCV '95)*, vol. 3, pp. 574–578, Singapore, December 1995.

[15] M. Chapron, "A chromatic contour detector based on abrupt change techniques," in *Proceedings of IEEE International Conference on Image Processing (ICIP '97)*, vol. 3, pp. 18–21, Santa Barbara, Calif, USA, October 1997.

[16] D. Zugaj and V. Lattuati, "A new approach of color images segmentation based on fusing region and edge segmentations outputs," *Pattern Recognition*, vol. 31, no. 2, pp. 105–113, 1998.

[17] P. M. Djuric and J.-K. Fwu, "On the detection of edges in vector images," *IEEE Transactions on Image Processing*, vol. 6, no. 11, pp. 1595–1601, 1997.

[18] A. Fotinos, G. Economou, and S. Fotopoulos, "Use of relative entropy in color edge detection," *Electronics Letters*, vol. 35, no. 18, pp. 1532–1534, 1999.

[19] G. Economou, "Detecting edges using density value," *Electronics Letters*, vol. 40, no. 24, pp. 1528–1530, 2004.

[20] F. Russo and A. Lazzari, "Color edge detection in presence of Gaussian noise using nonlinear prefiltering," *IEEE Transactions on Instrumentation and Measurement*, vol. 54, no. 1, pp. 352–358, 2005.

[21] G. Sharma and H. J. Trussell, "Digital color imaging," *IEEE Transactions on Image Processing*, vol. 6, no. 7, pp. 901–932, 1997.

[22] J. Angulo and J. Serra, "Color segmentation by ordered mergings," in *Proceedings of IEEE International Conference on Image Processing (ICIP '03)*, vol. 2, pp. 125–128, Barcelona, Catalonia, Spain, September 2003.

[23] L. Lucchese and S. K. Mitra, "Filtering color images in the xyY color space," in *Proceedings of IEEE International Conference on Image Processing (ICIP '00)*, vol. 3, pp. 500–503, Vancouver, BC, Canada, September 2000.

[24] S. Wesolkowski, M. E. Jernigan, and R. D. Dony, "Comparison of color image edge detectors in multiple color spaces," in *Proceedings of IEEE International Conference on Image Processing (ICIP '00)*, vol. 2, pp. 796–799, Vancouver, BC, Canada, September 2000.

[25] L. Shafarenko, M. Petrou, and J. V. Kittler, "Automatic watershed segmentation of randomly textured color images," *IEEE Transactions on Image Processing*, vol. 6, no. 11, pp. 1530–1544, 1997.

[26] P. E. Trahanias and A. N. Venetsanopoulos, "Color edge detection using vector order statistics," *IEEE Transactions on Image Processing*, vol. 2, no. 2, pp. 259–264, 1993.

[27] P. E. Trahanias and A. N. Venetsanopoulos, "Vector order statistics operators as color edge detectors," *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, vol. 26, no. 1, pp. 135–143, 1996.

[28] K. N. Plataniotis, D. Androutsos, and A. N. Venetsanopoulos, "Adaptive fuzzy systems for multichannel signal processing," *Proceedings of the IEEE*, vol. 87, no. 9, pp. 1601–1622, 1999.

[29] R. D. Dony and S. Wesolkowski, "Edge detection on color images using RGB vector angles," in *Proceedings of IEEE Canadian Conference on Electrical and Computer Engineering*, vol. 2, pp. 687–692, Edmonton, Alberta, Canada, May 1999.

[30] S. Wesolkowski and E. D. Jernigan, "Color edge detection in RGB using jointly euclidean distance and vector angle," in *Proceedings of IAPR Vision Interface Conference*, pp. 9–16, Trois Rivieres, Quebec, Canada, May 1999.

[31] D. Androutsos, K. N. Plataniotis, and A. N. Venetsanopoulos, "Distance measures for color image retrieval," in *Proceedings of IEEE International Conference on Image Processing (ICIP '98)*, vol. 2, pp. 770–774, Chicago, Ill, USA, October 1998.

[32] J. F. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analaysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698, 1986.

[33] J. Scharcanski and A. N. Venetsanopoulos, "Edge detection of color images using directional operators," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 2, pp. 397–401, 1997.

[34] P. Androutsos, D. Androutsos, K. N. Plataniotis, and A. N. Venetsanopoulos, "Color edge detectors: an overview and comparison," in *Proceedings of IEEE Canadian Conference on Electrical and Computer Engineering*, vol. 2, pp. 607–610, St. John's, Newfoundland, Canada, May 1997.

[35] S.-Y. Zhu, K. N. Plataniotis, and A. N. Venetsanopoulos, "Comprehensive analysis of edge detection in color image processing," *Optical Engineering*, vol. 38, no. 4, pp. 612–625, 1999.

[36] J. Lee, R. Haralick, and L. Shapiro, "Morphologic edge detection," *IEEE Transactions on Robotics and Automation*, vol. 3, no. 2, pp. 142–156, 1987.

[37] J. Astola, P. Haavisto, and Y. Neuvo, "Vector median filters," *Proceedings of the IEEE*, vol. 78, no. 4, pp. 678–689, 1990.

[38] A. N. Evans, "Morphological gradient operators for color images," in *Proceedings of IEEE International Conference on Image Processing (ICIP '04)*, vol. 5, pp. 3089–3092, Singapore, October 2004.

[39] A. N. Evans and X. U. Liu, "A morphological gradient approach to color edge detection," *IEEE Transactions on Image Processing*, vol. 15, no. 6, pp. 1454–1463, 2006.

[40] J.-F. Rivest, P. Soille, and S. Beucher, "Morphological gradients," *Journal of Electronic Imaging*, vol. 2, no. 4, pp. 326–336, 1993.

[41] I. E. Abdou and W. K. Pratt, "Quantitative design and evaluation of enhancement/thresholding edge detectors," *Proceedings of the IEEE*, vol. 67, no. 5, pp. 753–763, 1979.

Adrian N. Evans: Department of Electronic and Electrical Engineering, University of Bath, Bath BA2 7AY, UK

*Email*: a.n.evans@bath.ac.uk

# Index