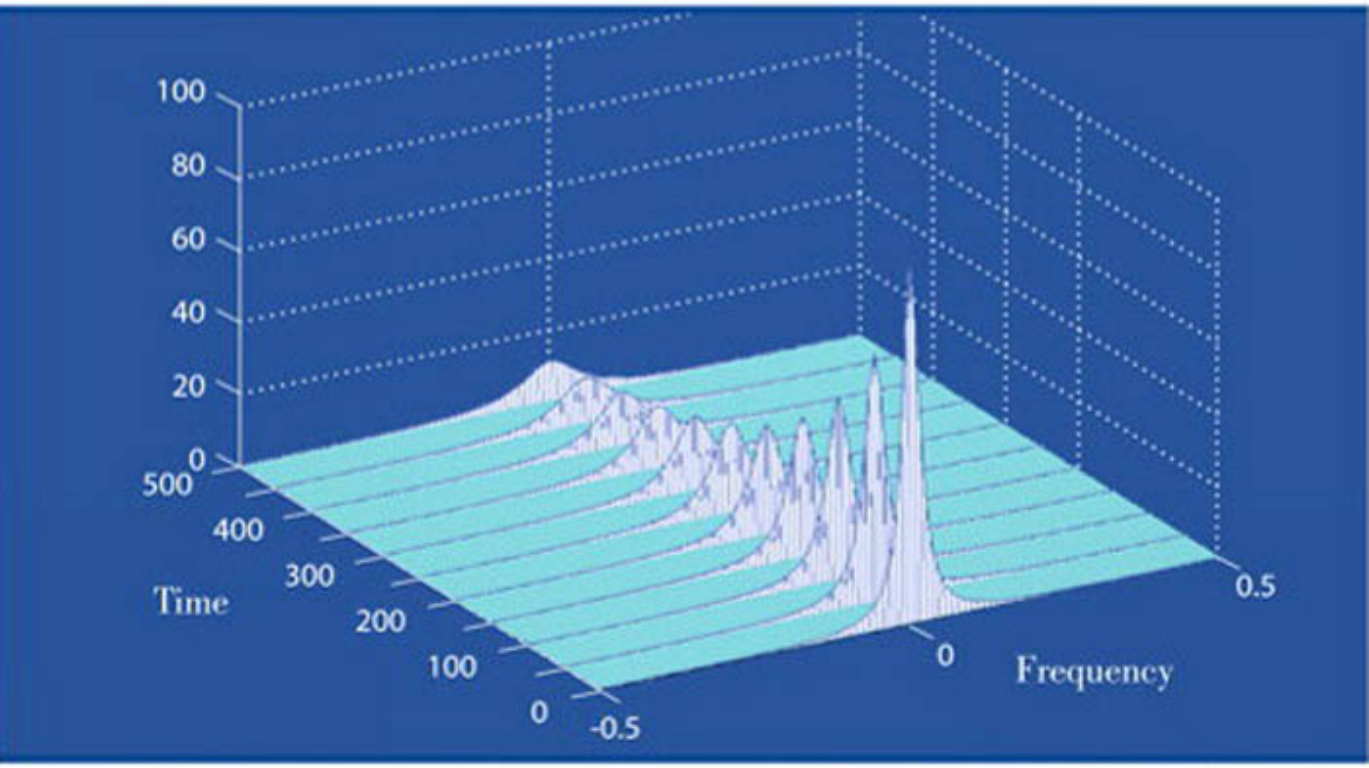
PRENTICE HALL

VOLUME III

FUNDAMENTALS OF STATISTICAL SIGNAL PROCESSING

PRACTICAL ALGORITHM DEVELOPMENT



STEVEN M. KAY

CD-ROM INCLUDED

FREE SAMPLE CHAPTER

SHARE WITH OTHERS

Fundamentals of Statistical Signal Processing, Volume III

This page intentionally left blank

Fundamentals of Statistical Signal Processing, Volume III

Practical Algorithm Development

Steven M. Kay

Upper Saddle River, NJ · Boston · Indianapolis · San Francisco
New York · Toronto · Montreal · London · Munich · Paris · Madrid
Capetown · Sydney · Tokyo · Singapore · Mexico City

Library of Congress Control Number: 92029495

ISBN-13: 978-0-13-280803-3
ISBN-10: 0-13-280803-X
Text printed in the United States on recycled paper at Courier Corporation in Westford, Massachusetts
First printing, March 2013.

To my wife
Cindy,
who is and will always be
my anchor

This page intentionally left blank

Contents

This page intentionally left blank

Preface

Fundamentals of Statistical Signal Processing: Practical Algorithm Development is the third volume in a series of textbooks by the same name. Previous volumes described the underlying theory of estimation and detection algorithms. In contrast, the current volume addresses the *practice* of converting this theory into software algorithms that may be implemented on a digital computer. In describing the methodology and techniques, it will not be assumed that the reader has studied the first two volumes, but of course, he/she is certainly encouraged to do so. Instead, the descriptions will focus on the general concepts using a minimum of mathematics but will be amply illustrated using MATLAB implementations. It is envisioned that the current book will appeal to engineers and scientists in industry and academia who would like to solve statistical signal processing problems through design of well-performing and implementable algorithms for real systems. These systems are typically encountered in many signal processing disciplines, including but not limited to communications, radar, sonar, biomedical, speech, optical, and image processing. Additionally, due to the emphasis on actual working algorithms, the material should be of use to the myriad of researchers in statistical signal processing who wish to obtain an overview of the state of the *practical* art. Those new to the field who are concerned with sorting the wheat from the chaff in the ever-exploding arsenal of signal processing algorithms will also benefit from the exposition.

The overall goal for this book is to allow the reader to develop his/her intuition and subsequent expertise into the *practice* of statistical signal processing. To accomplish this goal we have endeavored to

1. Describe the methodology, including mathematical modeling, computer simulation, and performance evaluation, used to develop algorithms.

2. Allow the reader to assimilate the important concepts by practicing with the tools typically available. These include useful analytical results and MATLAB implementations for design, evaluation, and testing.

3. Highlight the approaches and specific algorithms that work in practice, i.e., those that have stood the test of time.

4. Illustrate application areas by describing and solving real-world problems.

5. Introduce the reader to some extensions required in practice.

6. Translate a mathematical algorithm into MATLAB code and verify the integrity of the solution.

Pedagogically speaking, we believe that the strong reliance on MATLAB examples will aid in understanding the workings and subtleties of the various algorithms. The reader will then *learn by doing*. In the same vein, numerous analytical exercises have been embedded into the text for student practice. The full solutions are contained in the appendices of each chapter. MATLAB exercises are also given, with abbreviated solutions listed in the appendix of each chapter, and the full solutions, including executable MATLAB code, contained on the enclosed CD. At the end of many of the chapters is a section called "Lessons Learned". These conclusions are important observations that are intended to provide insight into the inner workings of the algorithms and rules of thumb that are routinely employed. These lessons learned are often critical to the development of successful algorithms. Most of the topics chosen for inclusion have been drawn from *Fundamentals of Statistical Signal Processing: Estimation Theory*, 1993, and *Fundamentals of Statistical Signal Processing: Detection Theory*, 1998, but we have also added much material from *Modern Spectral Estimation: Theory and Application*, 1988 (all books published by Prentice Hall), since the latter book contains many of the techniques required for data simulation and analysis. Finally, it is hoped that the current book will be useful for self-study. Although this volume can be used without MATLAB as a practice tool, much of the understanding that comes from that experience would be lost.

The background assumed for the reader is a knowledge of calculus, basic linear systems, including some digital signal processing, probability and introductory random processes, and some linear and matrix algebra. As previously mentioned, we have attempted to describe the techniques without heavy reliance upon mathematics and this background material. However, in the end the algorithms are by their nature mathematical and so it must be that this goal can only partially be attained.

The author would like to acknowledge the contributions of the many people who over the years have provided stimulating discussions of teaching and research problems and opportunities to apply the results of that research. Thanks are due to my colleagues L. Jackson, R. Kumaresan, L. Pakula, and P. Swaszek of the University of Rhode Island. A debt of gratitude is owed to all my current and former graduate students. They have contributed to the final manuscript through many hours of pedagogical and research discussions as well as by their specific comments and questions. In particular, Quan Ding and Naresh Vankayalapati have contributed specific comments and helped with the exercise solutions. Additionally, William Knight has provided valuable feedback on the manuscript. The author is indebted to the many agencies and program managers who have sponsored his research. These managers include Jon Davis, Darren Emge, James Kelly, Muralidhar Rangaswamy, Jon Sjogren, and Peter Zulch. The agencies include the Naval Undersea Warfare

Center, the Naval Air Warfare Center, the Air Force Office of Scientific Research, the Office of Naval Research, the Air Force Research Labs, and the Edgewood Chemical and Biological Center. The practical experience that the author has acquired from the numerous industrial firms for which he has consulted is also greatly appreciated. As always, the author welcomes comments and corrections, which can be sent to kay@ele.uri.edu.

—Steven M. Kay
University of Rhode Island
Kingston, RI

This page intentionally left blank

About the Author

Steven Kay was born in Newark, New Jersey. He received a B.E. from Stevens Institute of Technology, Hoboken, New Jersey, in 1972, an M.S. from Columbia University, New York, New York, in 1973, and a Ph.D. from Georgia Institute of Technology, Atlanta, Georgia, in 1980, all in electrical engineering. From 1972 to 1975, he was with Bell Laboratories, Holmdel, New Jersey, where he was involved with transmission planning for speech communications and simulation and subjective testing of speech processing algorithms.

From 1975 to 1977, he attended Georgia Institute of Technology to study communication theory and digital signal processing. From 1977 to 1980, he was with the Submarine Signal Division, Portsmouth, Rhode Island, where he engaged in research on autoregressive spectral estimation and the design of sonar systems. He is presently a Professor of Electrical Engineering at the University of Rhode Island, Kingston, and a consultant to numerous industrial concerns, the Air Force, the Army, and the Navy.

As a leading expert in statistical signal processing, he has been invited to teach short courses to scientists and engineers at government laboratories, including NASA and the CIA. He has written numerous journal and conference papers and is a contributor to several edited books. He is the author of the textbooks *Modern Spectral Estimation* (Prentice Hall, 1988), *Fundamentals of Statistical Signal Processing, Volume I: Estimation Theory* (Prentice Hall, 1993), *Fundamentals of Statistical Signal Processing, Volume II: Detection Theory* (Prentice Hall, 1998), and *Intuitive Probability and Random Processes using MATLAB* (Springer, 2005). His current interests are spectrum analysis, detection and estimation theory, and statistical signal processing.

Dr. Kay is a Fellow of the IEEE, and a member of Tau Beta Pi and Sigma Xi. He has been a distinguished lecturer for the IEEE signal processing society. He has been an associate editor for the IEEE *Signal Processing Letters* and the IEEE *Transactions on Signal Processing*. He has received the IEEE signal processing society education award for outstanding contributions in education and in writing scholarly books and texts.

Dr. Kay has recently been included on a list of the 250 most-cited researchers in the world in engineering.

This page intentionally left blank

Chapter 1

Introduction

1.1 Motivation and Purpose

Over the last forty years there has been a virtual explosion of ideas, approaches, techniques, and applications of digital signal processing (DSP) to commercial products and military systems. The primary journal devoted to digital signal processing, the *IEEE Transactions on Acoustics, Speech, and Signal Processing*, which was founded in 1974, was originally published bimonthly with each issue consisting of about 100 pages. Today the *IEEE Transactions on Signal Processing*, which is devoted solely to signal processing, is published monthly with a content of about 500 pages, reflecting a *tenfold increase* in papers. This does not even account for the other more specialized journals that have been spawned, such as the *IEEE Transactions on Audio, Speech, and Language Processing*, the *IEEE Transactions on Image Processing*, and others. The algorithm designer, who must choose and implement an approach, is now faced with a bewildering cornucopia of possible algorithms. Even surveying the open literature to glean a promising approach can be an overwhelming task. As a result, it is now more important than ever for the algorithm designer to have a tried and trusted arsenal at his/her disposal. These approaches may not solve the current problem in its entirety, but will at least provide a good starting point for algorithm development.

In addition to accumulating a suite of trusted algorithms, it is critical that we understand why they work as well as when they are likely *to fail*. DSP algorithms and more specifically *statistical signal processing* algorithms, being highly mathematical and stochastic in nature, do not yield their secrets easily. But as the designer begins to implement these algorithms and observe their behavior, his/her intuition grows along with chances for successful future algorithm choices. This intuition may only be gained through experience. We are fortunate today that it is not necessary to implement an algorithm in hardware to assess its performance. Software implementations are readily available and allow relatively painless assessments of performance. A popular and very versatile software language is MATLAB, and it

is this vehicle that we will use to implement our proposed algorithms and examine their performance. Its use allows us to "play" with the proposed algorithm as well as to provide us with a "first-cut" implementation in software. In fact, a MATLAB implementation frequently leads to an implementation on a DSP chip or on some specialized digital hardware. For these reasons we will rely heavily on MATLAB throughout this textbook.

The material contained herein are algorithms for *statistical signal processing*. On the other hand, for the processing of signals whose mathematical form is completely known and that are not subject to excessive noise, many standard techniques exist and have been found to be quite reliable. As examples, these typically include algorithms for the design of digital filters or for the computation of the discrete-time Fourier transform, i.e., the fast Fourier transform (FFT). Many excellent books describe these algorithms and their implementations [Ingle and Proakis 2007, Lyons 2009]. In contrast, our purpose is to describe algorithms that can be used to analyze and extract information from *random data*. For example, the specification that a signal whose Fourier spectrum is lowpass in nature should be filtered by a digital lowpass filter prior to further processing, naturally requires the design of a digital filter with a prescribed cutoff frequency. A slightly different specification might be to filter a bandpass signal whose center frequency is *unknown*. In the first case, the specification is complete. In the second case, it remains to determine how to center the filter so as to pass the signal but hopefully remove much of the noise. The former calls for deterministic signal processing while the latter requires *estimation of the center frequency*, preferably on-line, so that if the signal center frequency changes, our algorithm will still be able to provide the appropriately centered filter. When there is uncertainty in the signal characteristics, only a *statistical* approach is appropriate.

Algorithms for the analysis of random data are highly problem specific. This is to say that each real-world signal processing problem, although generically related to many others, is unique and requires a specialized approach. Because of the seemingly endless development of new electronic systems and devices, it is not possible to use "off-the-shelf" algorithms. However, all is not lost! There exist a suite of "core" algorithms that appear at the heart of most practical signal processing systems. It is our intention to describe and implement in MATLAB these approaches in this book. A general discussion of these algorithms is given next.

1.2 Core Algorithms

For signal processing problems requiring the detection of a signal and estimation of its parameters, there exist some statistically sound and consequently, well accepted approaches. As examples, we mention the *matched filter* for detection, the *maximum likelihood estimator* and its frequent implementation, the *least squares estimator*, for parameter estimation. It is these well accepted approaches that we intend to focus

on. Hopefully, with exposure to the techniques that work in practice, the signal processing algorithm designer will at least have a good starting point from which to proceed to an actual design. Many of the core approaches, in addition to more advanced but possibly not proven-in-practice techniques, have been described in detail in the first two volumes of *Fundamentals of Statistical Signal Processing* [Kay 1993, Kay 1998] and in *Modern Spectral Estimation: Theory and Application* [Kay 1988]. The latter book on spectral analysis is important for modeling of random signals and provides many useful algorithms for computer generation of these signals. The reader is encouraged to refer to these books for a fuller understanding of the theoretical underpinnings of these approaches. In this volume we

1. Describe the important algorithms used in practice,

2. Describe the assumptions required for their successful operation, and

3. Describe their performance and their limitations in practice.

This book is an attempt to accomplish these goals *without having had the exposure to the books referenced above.*

1.3 Easy, Hard, and Impossible Problems

Since our goal is to describe statistical signal processing algorithms that are widely used in practice, we may ask how these algorithms have attained this place of great honor. The reasons are two-fold. The first is that they "work" and the second is that they can be conveniently implemented in digital software/hardware. For an algorithm to "work", it must meet the specifications of the system. For example, it might be that a specification calls for a parameter to be estimated. The performance of the estimator should be a relative error of no more than 2.5%, as an example. Hence, whether an algorithm "works" or not depends upon what is expected of the algorithm. If the specifications are unreasonable, then a proposed approach *or any approach* may not work. It is therefore important to assess the *feasibility* of meeting the performance requirements. For the latter example, a commonly used method for feasible parameter estimation accuracy is the *Cramer-Rao lower bound* (see Section 8.2.1). It provides a lower bound on the variance of an unbiased estimator (i.e., one that on the average yields the correct result). If the specifications cannot be met in theory, then *there is no point in proceeding with a design.* Maybe we should require more accurate sensors, if this is a possibility, and/or more data. Given a signal model and a noise model (we will discuss this further in Chapters 3–6), signal processing is capable of providing implementable algorithms that *extract all the available information.* This information, we hope, is sufficient to yield the desired performance. However, *it cannot do the impossible*, although we may ask it to do so! As an example, suppose we wish to estimate the value A of a constant *discrete-time* signal $s[n]$, also referred to as a DC level signal (presumably a continuous-time

signal that has been sampled by an analog-to-digital (A/D) convertor). The signal is given as

$$s[n] = A \qquad n = 0, 1, \ldots, N-1$$

and is embedded in white Gaussian noise (WGN) $w[n]$ with power σ^2 (see Section 4.3). The observed data is then given by $x[n] = A + w[n]$ for $n = 0, 1, \ldots, N-1$ and from this data we wish to determine A as accurately as possible. It is well known that the optimal way to do this is to employ the estimator given by the *sample mean*

$$\hat{A}_N = \frac{1}{N} \sum_{n=0}^{N-1} x[n].$$

(The "hat" will always denote an estimator.) Suppose that $A = 10$ and that our specifications require for $N = 20$ and $\sigma^2 = 1$ that the estimate should fall within the interval $[A-0.25, A+0.25]$, which for this choice of parameter would be $[9.75, 10.25]$, a maximum relative error of 2.5%. A MATLAB computer simulation is shown in Figure 1.1, in which the estimate $\hat{A}_N$ is plotted versus the data record length N.[1] (We have connected the points $(\hat{A}_1, \hat{A}_2, \ldots)$ in the figure by straight lines to make the viewing easier.) Since our specification was for $N = 20$, the results seem to indicate that it has been met, with the estimate falling between the dashed lines at 9.75 and 10.25 at $N = 20$.

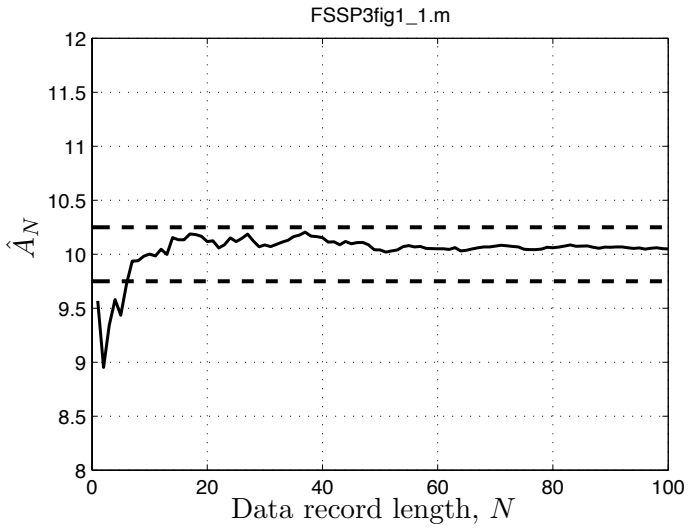


Figure 1.1: Estimate of DC level A versus data record length N.

[1]The MATLAB program used to produce the figure is listed at the top of the figure. These programs may be obtained from the author upon request.

However, this good performance may just be fortuitous in that if we repeat the experiment, whereby a different set of WGN samples $w[n]$ are generated, then we might obtain very different results. In Figure 1.2 the results of five different experiments, corresponding to five different WGN realizations, are shown. (Each realization of $N = 100$ noise samples will be different.)

Now the specification is not met for $N = 20$ data points and only appears to be met for more than about $N = 40$ points. In fact, if we require that the estimate fall within the dashed lines for $N = 20$ and *for 95.5% "of the time"*, then it can be shown that the variance $\text{var}(\hat{A}_N)$ of the estimator must satisfy

$$2\sqrt{\text{var}(\hat{A}_N)} \leq 0.25$$

or equivalently

$$\text{var}(\hat{A}_N) \leq \frac{1}{64}.$$

But the Cramer-Rao lower bound says that all (unbiased) estimators must have

$$\text{var}(\hat{A}_N) \geq \frac{\sigma^2}{N} \tag{1.1}$$

which for $\sigma^2 = 1$ and $N = 20$ produces a lower bound of $1/20$, and so this specification is impossible to meet. We will see later that the estimator $\hat{A}_N$, called the

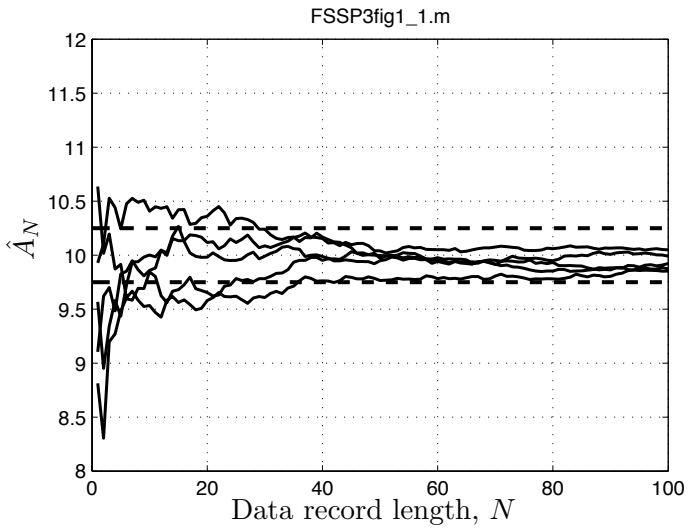


Figure 1.2: Five different realizations of the DC level estimator.

sample mean estimator, does indeed *attain* the bound and so its variance is given by equality in (1.1). Therefore, to meet the specification we would require

$$\frac{\sigma^2}{N} = \frac{1}{64}$$

which is to say that N must be at least 64 data samples for a noise power of $\sigma^2 = 1$. As a side note, the reader should heed the implicit lesson seen in Figures 1.1 and 1.2. Since the results depend upon the particular noise sequence generated, *it is imperative that the experiment be repeated many times.* No conclusions can be drawn from only a few realizations, and in fact, in Figure 1.2 many more realizations should be added.

Exercise 1.1 – Necessity of multiple realizations for performance analysis

a. Run the MATLAB program `FSSP3exer1_1.m` for 100 noise realizations and for $N = 64$. Do about 95 of the estimates fall within the specified interval $[9.75, 10.25]$?

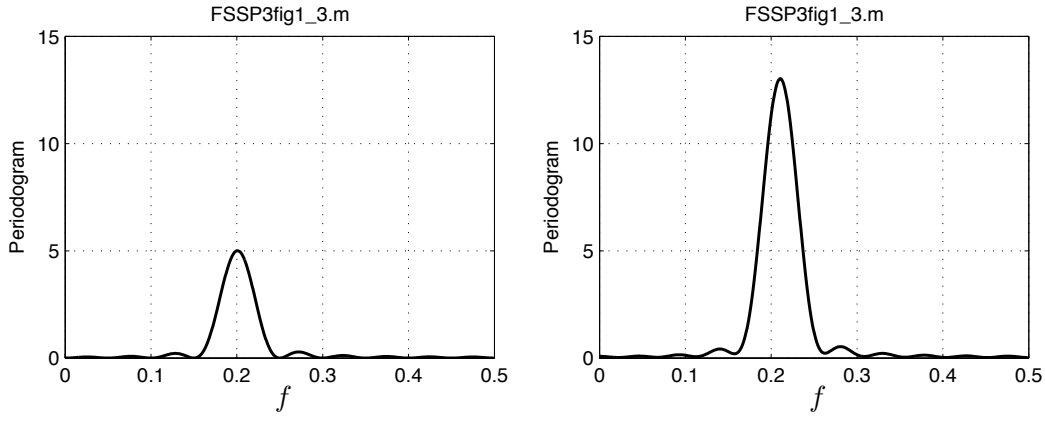
b. If the accuracy specification is increased to 1% maximum relative error, what should $\text{var}(\hat{A}_N)$ be? How long does the data record length N need to be to attain this? Finally, modify the code in `FSSP3exer1_1.m` to simulate the performance for this larger value of N. How many estimates meet the specification?

•

Some signal processing problems appear in many different fields. For example, it is of interest to be able to accurately determine the frequency of a sinusoidal signal when the signal is embedded in noise. A mathematically optimal approach is to use a *periodogram*, which is a type of spectral estimator, and pick the location of the maximum value as the estimate (see Algorithm 9.3). This works well in practice and has found widespread acceptance for numerous applications. Additionally, even if the underlying assumptions that need to be made to claim optimality are violated somewhat, the performance does not degrade radically. This is an example of a *robust algorithm*. In practice, robustness can be a critically important property of an algorithm. Few real-world data sets conform to a set of underlying theoretical assumptions. These assumptions are usually made for mathematical tractability so as to allow the derivation of an optimal approach. We might term the design of a frequency estimator of a single sinusoid as an *easy problem*, since the solution that performs well and is easily implemented (via an FFT) is the periodogram. Of course, to justify its use, the underlying assumption that we have a single sinusoid embedded in WGN, cannot be totally ignored. If other sinusoids or interfering signals are present and/or if the noise is highly correlated, then this approach may not perform as advertised. Specifically, if a second sinusoid is close in frequency to

the one of interest, then the frequency estimator can be severely biased, producing an inaccurate estimate. An example of this is given in Figure 1.3. In Figure 1.3a the periodogram of a single sinusoid with an amplitude of one at a frequency of 0.2 is shown. The maximum is seen to occur at the correct location in frequency. In Figure 1.3b the periodogram is shown for the sum of two sinusoids, the desired one at a frequency of 0.2 combined with an interfering sinusoid at a frequency of 0.22, also with an amplitude of one. It is seen that the peak is now biased away from the desired frequency of 0.2. To account for this possibility we must alter our thinking and acknowledge the potential for one or more interfering sinusoidal signals. This additional complication then leads to a *hard problem* [Kay 1988]. There may not be a good solution, especially if the frequencies of the interfering sinusoids are unknown. As the reader has no doubt surmised, properly designed algorithms work well (and some can be said to be optimal in performance) when the assumptions under which they were designed are satisfied. It is therefore critical that we be able to verify that *these assumptions hold in practice*. To prepare ourselves for disappointing performance results we need to assess not only the good properties of the algorithm but also it limitations.

It is often the case that a seemingly difficult or *hard problem* will yield to an easier one if viewed appropriately. It is well known that the *linear signal model* (see Sections 3.5 and 3.6.4 for a description) leads to optimally performing and easily implementable algorithms. In the real world not all signals can be said to conform to this model. Knowing, however, of the desirable properties of the linear model, it behooves us to try and transform our nonlinear model into a linear one. Continu-



(a) One real sinusoid at $f_0 = 0.2$ (b) Two real sinusoids at $f_0 = 0.2$ and $f_0 = 0.22$

Figure 1.3: Periodogram of a sinusoidal signal and also a sinusoidal signal plus an interfering signal.

ing with the sinusoidal signal example, we have its mathematical representation in discrete-time as

$$s[n] = A\cos(2\pi f_0 n + \phi) \qquad n = 0, 1, \ldots, N-1 \tag{1.2}$$

where A is an unknown amplitude with $A > 0$, f_0 is a known frequency with $0 < f_0 < 1/2$, and ϕ is an unknown phase with $-\pi \leq \phi < \pi$. We might wish to estimate the unknown amplitude and unknown phase. As it stands, however, the signal is nonlinear in the phase (since $A\cos(2\pi f_0 n + \phi_1 + \phi_2) \neq A\cos(2\pi f_0 n + \phi_1) + A\cos(2\pi f_0 n + \phi_2)$), and this will complicate the development of any estimation algorithm. To convert this problem into a more manageable one, we could use the trigonometric identity $\cos(C + D) = \cos(C)\cos(D) - \sin(C)\sin(D)$ to yield

$$s[n] = A\cos(\phi)\cos(2\pi f_0 n) - A\sin(\phi)\sin(2\pi f_0 n)$$

and then let $\alpha_1 = A\cos(\phi)$ and $\alpha_2 = -A\sin(\phi)$ (which is just a type of polar to Cartesian coordinate transformation) to produce

$$s[n] = \alpha_1 \cos(2\pi f_0 n) + \alpha_2 \sin(2\pi f_0 n). \tag{1.3}$$

The signal is now *linear* in the unknown transformed parameters α_1, α_2. We have transformed the original hard problem into a relatively easy one, and furthermore, into one *whose solution is well known*. It can be shown that a good estimator of amplitude and phase based on the observed data set $\{x[0], x[1], \ldots, x[N-1]\}$, which consists of the sinusoidal signal plus noise, is (see Section 3.5.4 and Algorithm 9.2)

$$\begin{aligned} \hat{A} &= \frac{2}{N}\left|\sum_{n=0}^{N-1} x[n]\exp(-j2\pi f_0 n)\right| \\ \hat{\phi} &= \arctan\left(\frac{-\sum_{n=0}^{N-1} x[n]\sin 2\pi f_0 n}{\sum_{n=0}^{N-1} x[n]\cos 2\pi f_0 n}\right). \end{aligned} \tag{1.4}$$

But to do so we had to have been familiar with the easy problem (the linear signal model) and then been able to choose the appropriate transformation. In practice, knowledge of easy problems for which we have already *known* solutions is *indispensable*. This book contains many of these well known solutions.

In the real world most of the signal processing problems are *hard* (if they were not, then someone would have already solved them!). But knowing the solutions to simpler problems and building upon the intuition that familiarity with these problems brings, can lead to good solutions for more difficult problems. For example, in (1.3) we might not know the frequency f_0. How could we then estimate the fre-

quency in addition to the amplitude and phase? It can be shown that the magnitude of the discrete-time Fourier transform of a time truncated sinusoid, i.e., of (1.2), is

$$|S(f)| = \left|\sum_{n=0}^{N-1} s[n]\exp(-j2\pi fn)\right| \tag{1.5}$$

$$\approx \frac{NA}{2}\left|\frac{\sin[N\pi(f-f_0)]}{N\sin[\pi(f-f_0)]}\right| \tag{1.6}$$

as long as f_0 is not very close to 0 or 1/2. You are asked to verify this in the next exercise (the solution is contained in Appendix 1A).

Exercise 1.2 – Derivation of discrete-time Fourier transform of truncated sinusoid

Verify the expression given in (1.6). To do so first break the real sinusoid given in (1.2) into its two complex conjugate components $\exp(j2\pi f_0 n)$ and $\exp(-j2\pi f_0 n)$, insert these into (1.5) and then use the complex geometric series result

$$\left|\sum_{n=0}^{N-1} z^n\right| = \left|\frac{1-z^N}{1-z}\right| = \left|\frac{z^{-N/2}-z^{N/2}}{z^{-1/2}-z^{1/2}}\right|$$

where the first equality is true for all complex z and the second equality is only true for $|z| = 1$, where $|\cdot|$ denotes the complex magnitude. Finally, discard the negative frequency contribution, which is negligible if we are evaluating the Fourier transform for $f > 0$ (and f_0 is not near 0 or 1/2).

•

A plot of $|S(f)|$ is shown in Figure 1.4 for $A = 1$, $N = 20$, and $f_0 = 0.2$. This suggests that we might be able to estimate the frequency by using the peak location in frequency as our estimator, and in fact forms the basis for the periodogram estimator previously mentioned and illustrated in Figure 1.3a. Although the effect of noise has not been included in our discussion, it does indeed turn out that even when noise is present, the estimator performs well (see Algorithm 9.3). This result is also intuitively clear in that for large data records, i.e., as $N \to \infty$, the Fourier transform becomes a Dirac impulse, and so should "stand out" above any noise background. The need for a good sense of intuition cannot be overemphasized. It prevents us from pursuing algorithmic approaches that result in dead ends and it explains why a good algorithm works well, apart from the mathematical justification—if there is one. How does one obtain this intuition? Answer: *Practice, practice, practice!*

1.4 Increasing Your Odds for Success—Enhance Your Intuition

To build up our intuition we must learn by doing. Fortunately, there are many software packages and hardware development kits that can provide us with adequate

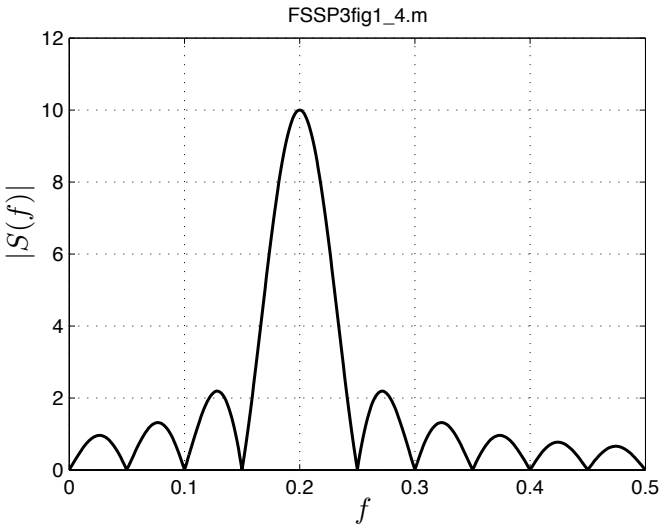


Figure 1.4: Magnitude of discrete-time Fourier transform of time truncated sinusoid.

practice. Since our aim is to design the algorithms in software, our discussion will be limited to this implementation. We have found the MathWorks software package MATLAB to be particularly well-suited to the types of signal processing problems encountered in practice. Therefore, throughout the book MATLAB will be our primary vehicle for developing and testing statistical signal processing algorithms. We have even used MATLAB to generate the textbook figures (the program used is always listed at the top of the graph and may be obtained from the author upon request) and to solve some of the exercises. The code for the exercise solutions as well as programs that implement the algorithms to be described and to perform various useful functions are provided on the enclosed CD.

Implementation of an algorithm in MATLAB is a testament to the understanding of the basic operation of the algorithm. Furthermore, testing the algorithm on controlled data sets generated within MATLAB, i.e., ones for which the signal and noise characteristics are known, *is essential* in verifying the efficacy of the algorithm. In observing the output of the MATLAB-coded algorithms, and comparing the output with the expected one, a strong sense of intuition can be attained. Finally, we can at times employ the same code used to assess the theoretical algorithm performance using a computer simulation to process actual field data, although usually not in a real-time operational sense. In all these cases, practice is essential in developing one's intuition. There will be ample opportunity to partake of this valuable tool for understanding in completing the exercises scattered throughout the book.

1.5 Application Areas

The statistical signal processing algorithms to be described find application in many fields. Some of these are:

1. Communications - transmission and reception of digital information [Proakis and Salehi 2007]

2. Radar and sonar - target detection, localization, and tracking [Richards 2005, Burdic 1984]

3. Biomedicine - heart arrhythmia detection, brain computer interfaces [Sörnmo and Laguna 2005, Sanei and Chambers 2007]

4. Image processing - medical diagnostic imaging, data compression [Gonzalez and Woods 2008]

5. Speech processing - speech recognition and synthesis [Rabiner and Schafer 2007]

6. Radio navigation - global positioning systems [Pany 2010].

This is but a small sampling of applications with many more being added every day. At first glance it may seem strange that these somewhat diverse fields employ nearly the same statistical signal processing algorithms. This was not always so. This confluence of approaches is mainly due to the use of the modern digital computer for implementation of signal processing. Whether the signal is electrical, as in radar, acoustic, as in sonar and speech, or optical, as in imaging, it ultimately is reduced to a set of numbers to be input and stored within a digital computer. The transformation from a physical signal, say speech, to a set of numbers is accomplished via a transducer, say a microphone, followed by an A/D convertor to produce a set of bytes to be read and stored for further processing within a digital computer. The only difference from a signal processing perspective is the character of the signal. It can be lowpass, having most of its energy at low frequencies, as in speech, or bandpass, having its energy within a given band, as in a radar signal. It can be one-dimensional in nature, as in an acoustic signal, or two-dimensional, as in an image. The sampling rate of the A/D convertor will therefore need to be tailored to the signal under consideration and will result in different amounts of data to process. However, remarkably similar, if not identical, algorithms are used to process these widely disparate types of signals. Matched filters are used in sonar, where the signal spectrum has frequencies in the KHz range, but also in radar, where the signal spectrum has higher frequencies, usually in the GHz range. FFTs are used for one-dimensional signals such as speech, but also the two-dimensional version of the FFT is used for image analysis. As a result, it is not unusual to design signal processing algorithms based solely on a mathematical description of the signal and

noise without reference to their origins. This would of course ignore any prior knowledge of the real-world constraints that the signal and noise must obey, and so any subsequent algorithm developed may have the potential for improved performance by imposing these constraints. Knowing, for example, that a signal evolved from a reflection from a moving target allows the radar designer to put a constraint on the Doppler frequency shift due to the constraint on maximum target speed.

1.6 Notes to the Reader

1.6.1 Types of Signals Considered

In describing the algorithms we will assume that the physical signal is a *real, lowpass signal* that has been sampled at an appropriate rate (at least twice the highest frequency, i.e., at least the Nyquist rate). For applications such as radar and sonar that process bandpass signals, it is customary to use complex demodulation, followed by sampling of the *in phase* and *quadrature* signals. This leads to a complex signal representation, which is slightly more complicated. The required extensions are described in Chapter 12. Also, because the signals are discrete-time in nature, having been sampled by some device and stored in a digital computer, we will always assume the received data has the form $x[n]$, which is a sequence of real numbers indexed by the integer n. Typically, we will use the index set $n = 0, 1, \ldots, N-1$ if the data record consists of N successive time samples. Note that we have referred to the samples as being indexed by time. However, the n index could equally well represent *spatial samples* such as would be obtained from N sensors equally spaced along a line with the spatial samples having been obtained by sampling all the sensor outputs *at a given and fixed time*.

1.6.2 Book Features and Notation

The MATLAB version used throughout the textbook is 7.8 (R2009A). Toolboxes are not required to run the MATLAB code, and where subprograms are called for, they are provided. A brief introduction to MATLAB is contained in Appendix B. A description of the code for all programs contained on the CD is given in Appendix C. In addition, a `readme.txt` file contained on the CD describes its contents. The "typewriter" font, such as used in `run_simulation.m`, indicates MATLAB program names and code.

Throughout the book there are exercises to provide the reader some practice in simple analytical manipulations and MATLAB algorithmic implementations. The solutions to the analytical exercises are contained in the corresponding chapter appendix. The MATLAB exercise solutions are only summarized, with more complete solutions found on the CD. Note the solutions were obtained using MATLAB version R2009A, and so future versions of MATLAB may produce slightly different results.

The reader is *strongly encouraged* to try the exercises as they form an important pathway to understanding the material.

At the end of each chapter there is a section entitled "Lessons Learned". These are important results, many of which have become "rules of thumb", and thus, should be committed to memory. For some applications they may form the basis on which an algorithm is either explored for its efficacy or otherwise rejected as not being suitable for the problem at hand.

The mathematical notation for all common symbols is summarized in Appendix A. The distinction between a continuous-time waveform and a discrete-time waveform or sequence is made through the symbolism $x(t)$ for continuous-time and $x[n]$ for discrete-time. Plots of discrete-time data such as $x[n]$, however, may appear continuous in time, the points having been connected by straight lines for easier viewing. An example of this is given in Figure 1.1. All vectors and matrices are **boldface**, with all vectors being *column* vectors. When a random variable needs to be contrasted with its value, we will use a capital letter, say X, to denote the random variable, and a lower case letter, say x, to denote its value. All other symbolism is defined within the context of the discussion. Also, the reader will frequently be warned of potential "pitfalls". Common misconceptions leading to design errors will be noted and described. The pitfall or caution symbol shown below should be heeded!!

1.7 Lessons Learned

The lessons listed next will be a major recurring theme of our discussions. We will have much more to say about these lessons as we examine algorithms and their performance throughout the book.

- Assess the feasibility of the algorithm requirements as the first step in the design. Typically, the Cramer-Rao lower bound is used for estimation, and the probability of detection of the Neyman-Pearson likelihood ratio test is used for detection. For classification, the maximum a posteriori (MAP) classifier is used. In all cases, the probability density functions must be known.

- Reassess the goals and/or require more accurate data if the specifications are not attainable.

- Signal processing cannot do the impossible—there must be a reasonable signal-to-noise ratio for it to succeed.

- In determining the performance of an algorithm via computer simulation, repeat the experiment many times, say 1000 or more. Make sure that your

performance metric is statistical in nature, i.e., variance for an estimator, probability of detection for a detector, probability of error for a classifier. Keep increasing the number of experiments until the evaluation of these metrics produces consistent results.

- Make sure that the algorithm is tested under varying operational conditions to assess robustness, also called *sensitivity*.
- Verify that the underlying algorithm assumptions hold in practice by analyzing real-world data.
- Try to transform the problem into a simpler one, for example, the linear signal model.
- Test the algorithm first in MATLAB by using controlled data sets generated within MATLAB. The results should agree with theoretical predictions, if available. The performance obtained under MATLAB controlled conditions should be an upper bound on that for field data.
- Before proposing an algorithm, scour the open literature for similar signal processing problems in other fields and the methods employed.

References

Burdic, W.S., *Underwater Acoustic Systems Analysis*, Prentice-Hall, Englewood Cliffs, NJ, 1984.

Gonzales, R.C., R.E. Woods, *Digital Image Processing*, Prentice Hall, Upper Saddle River, NJ, 2008.

Ingle, V.K., J.G. Proakis, *Digital Signal Processing Using MATLAB*, 2nd ed., Cengage Learning, Stamford, CT, 2007.

Kay, S., *Modern Spectral Estimation: Theory and Application*, Prentice-Hall, Englewood Cliffs, NJ, 1988.

Kay, S., *Fundamentals of Statistical Signal Processing: Estimation Theory, Vol. I*, Prentice-Hall, Englewood Cliffs, NJ, 1993.

Kay, S., *Fundamentals of Statistical Signal Processing: Detection Theory, Vol. II*, Prentice-Hall, Englewood Cliffs, NJ, 1998.

Lyons, R.G., *Understanding Digital Signal Processing*, Prentice Hall, Upper Saddle River, NJ, 2009.

Pany, T., *Navigation Signal Processing for GNSS Software Receivers*, Artech House, Norwood, MA, 2010.

Proakis, J., M. Salehi, *Digital Communications*, 5th ed., McGraw-Hill, NY, 2007.

Rabiner, L.R., R.W. Schafer, *Introduction to Digital Speech Processing*, Now, Boston, 2007.

Richards, M.A., *Fundamentals of Radar Signal Processing*, McGraw-Hill, NY, 2005.

Sanei, S., Chambers, J.A., *EEG Signal Processing*, Wiley-Interscience, NY, 2007.

Sörnmo, L., P. Laguna, *Bioelectrical Signal Processing in Cardiac and Neurological Applications*, Elsevier Academic Press, NY, 2005.

This page intentionally left blank

Appendix 1A Solutions to Exercises

To obtain the results described below initialize the random number generators to `rand('state',0)` and `randn('state',0)` at the beginning of any code. These commands are for the uniform and Gaussian random number generators, respectively.

1.1 For part a, run the code with `randn('state',0)` at the start of the program to intialize the random number generator. You should observe 92 estimates that meet the specifications, i.e., lie within the interval $[9.75, 10.25]$. For part b, we now require $2\sqrt{\text{var}(\hat{A}_N)} \leq 0.1$ so that from the CRLB $\text{var}(\hat{A}_N) = \sigma^2/N = 1/400$ and thus, we require that $N = 400$. Modifying the program and running it, produces 95 estimates that meet the specification, i.e., lie within the interval $[9.9, 10.1]$.

1.2

$$\begin{aligned}
S(f) &= \sum_{n=0}^{N-1} s[n] \exp(-j2\pi fn) \\
&= \sum_{n=0}^{N-1} A\cos(2\pi f_0 n + \phi)\exp(-j2\pi fn) \\
&= \sum_{n=0}^{N-1} \left[\frac{A}{2}\exp(j2\pi f_0 n + \phi) + \frac{A}{2}\exp(-j2\pi f_0 n - \phi)\right]\exp(-j2\pi fn) \\
&= \frac{A}{2}\exp(j\phi)\sum_{n=0}^{N-1}\exp[-j2\pi(f - f_0)n] \qquad \text{(1A.1)} \\
&\quad + \frac{A}{2}\exp(-j\phi)\sum_{n=0}^{N-1}\exp[-j2\pi(f + f_0)n]. \qquad \text{(1A.2)}
\end{aligned}$$

Now let $z = \exp[-j2\pi(f - f_0)]$ and note that $|z| = 1$. Dropping the second sum we have

$$S(f) = \frac{A}{2}\exp(j\phi)\sum_{n=0}^{N-1} z^n = \frac{A}{2}\exp(j\phi)\frac{1 - z^N}{1 - z}$$

and taking its complex magnitude

$$\begin{aligned}
|S(f)| &= \frac{A}{2}\left|\frac{1 - z^N}{1 - z}\right| \\
&= \frac{A}{2}\left|\frac{z^{N/2}(z^{-N/2} - z^{N/2})}{z^{1/2}(z^{-1/2} - z^{1/2})}\right| \\
&= \frac{A}{2}\left|\frac{z^{-N/2} - z^{N/2}}{z^{-1/2} - z^{1/2}}\right|.
\end{aligned}$$

Next letting $\alpha = -2\pi(f - f_0)$, we have $z = \exp(j\alpha)$ and

$$\begin{aligned} |S(f)| &= \frac{A}{2}\left|\frac{\exp[-j\alpha(N/2)] - \exp[j\alpha(N/2)]}{\exp[-j\alpha(1/2)] - \exp[j\alpha(1/2)]}\right| \\ &= \frac{A}{2}\left|\frac{\sin(N\alpha/2)}{\sin(\alpha/2)}\right|. \end{aligned}$$

Since $\alpha/2 = -\pi(f - f_0)$ we have finally that

$$|S(f)| = \frac{A}{2}\left|\frac{\sin[N\pi(f - f_0)]}{\sin[\pi(f - f_0)]}\right| = \frac{NA}{2}\left|\frac{\sin[N\pi(f - f_0)]}{N\sin[\pi(f - f_0)]}\right|.$$

Note that this is approximate since we have neglected the second term in the sum, i.e., the term given by (1A.2). Since this term is the mirror image of the positive frequency term (the one we retained) and is centered about $f = f_0$, its contribution will be small as long as it does not "interfere" with the positive frequency term. This will be the case if f_0 is not near 0 or 1/2. This is illustrated in Figure 1A.1 in which we have plotted the magnitude of the terms (1A.1) (positive frequency component) and (1A.2) (negative frequency

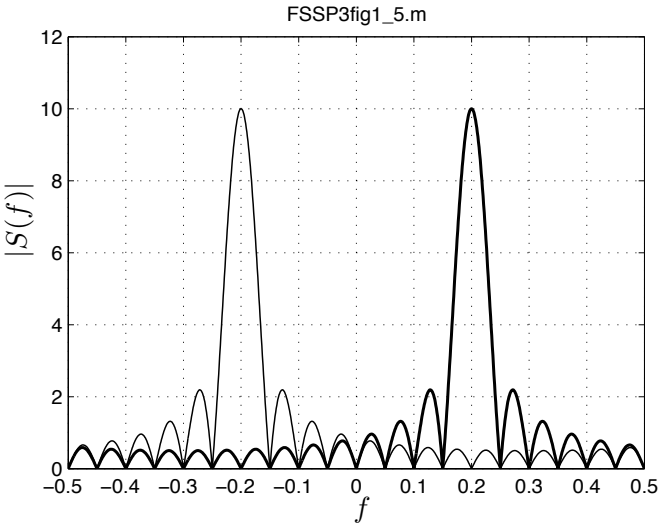


Figure 1A.1: Magnitude of discrete-time Fourier transform of frequency components with $f_0 = 0.2$ and $N = 20$. The heavier line indicates the Fourier transform magnitude of the positive frequency component given by (1A.1) while the lighter line is that for the negative frequency component given by (1A.2). Significant interactions between the two components occur when $0 < f_0 < 1/N = 0.05$ or $0.45 = 1/2 - 1/N < f_0 < 1/2$.

component) for the entire frequency band $-0.5 \leq f \leq 0.5$. The signal parameters are $N = 20$, $A = 1$, $\phi = 0$, and $f_0 = 0.2$. It is seen that there is little interference in this case.

This page intentionally left blank

This page intentionally left blank

Index