mobile communications series

SOFTWARE-DEFINED RADIO for ENGINEERS

TRAVIS F. COLLINS
ROBIN GETZ
DI PU
ALEXANDER M. WYGLINSKI

Software-Defined Radio for Engineers

For a listing of recent titles in the *Artech House Mobile Communications*, turn to the back of this book.

Software-Defined Radio for Engineers

Travis F. Collins
Robin Getz
Di Pu
Alexander M. Wyglinski

Library of Congress Cataloging-in-Publication Data
A catalog record for this book is available from the U.S. Library of Congress.

British Library Cataloguing in Publication Data
A catalog record for this book is available from the British Library.

ISBN-13: 978-1-63081-457-1

Cover design by John Gomes

10 9 8 7 6 5 4 3 2 1

Dedication

To my wife Lauren
—Travis Collins

To my wonderful children, Matthew, Lauren, and Isaac, and my patient wife, Michelle—sorry I have been hiding in the basement working on this book. To all my fantastic colleagues at Analog Devices: Dave, Michael, Lars-Peter, Andrei, Mihai, Travis, Wyatt and many more, without whom Pluto SDR and IIO would not exist.
—Robin Getz

To my lovely son Aidi, my husband Di, and my parents Lingzhen and Xuexun
—Di Pu

To my wife Jen
—Alexander Wyglinski

Contents

Preface xiii

CHAPTER 1
Introduction to Software-Defined Radio 1

1.1 Brief History 1
1.2 What is a Software-Defined Radio? 1
1.3 Networking and SDR 7
1.4 RF architectures for SDR 10
1.5 Processing architectures for SDR 13
1.6 Software Environments for SDR 15
1.7 Additional readings 17
References 18

CHAPTER 2
Signals and Systems 19

2.1 Time and Frequency Domains 19
2.1.1 Fourier Transform 20
2.1.2 Periodic Nature of the DFT 21
2.1.3 Fast Fourier Transform 22
2.2 Sampling Theory 23
2.2.1 Uniform Sampling 23
2.2.2 Frequency Domain Representation of Uniform Sampling 25
2.2.3 Nyquist Sampling Theorem 26
2.2.4 Nyquist Zones 29
2.2.5 Sample Rate Conversion 29
2.3 Signal Representation 37
2.3.1 Frequency Conversion 38
2.3.2 Imaginary Signals 40
2.4 Signal Metrics and Visualization 41
2.4.1 SINAD, ENOB, SNR, THD, THD + N, and SFDR 42
2.4.2 Eye Diagram 44
2.5 Receive Techniques for SDR 45
2.5.1 Nyquist Zones 47
2.5.2 Fixed Point Quantization 49

2.5.3 Design Trade-offs for Number of Bits, Cost, Power, and So Forth 55
2.5.4 Sigma-Delta Analog-Digital Converters 58
2.6 Digital Signal Processing Techniques for SDR 61
2.6.1 Discrete Convolution 61
2.6.2 Correlation 65
2.6.3 Z-Transform 66
2.6.4 Digital Filtering 69
2.7 Transmit Techniques for SDR 73
2.7.1 Analog Reconstruction Filters 75
2.7.2 DACs 76
2.7.3 Digital Pulse-Shaping Filters 78
2.7.4 Nyquist Pulse-Shaping Theory 79
2.7.5 Two Nyquist Pulses 81
2.8 Chapter Summary 85
References 85

CHAPTER 3
Probability in Communications 87
3.1 Modeling Discrete Random Events in Communication Systems 87
3.1.1 Expectation 89
3.2 Binary Communication Channels and Conditional Probability 92
3.3 Modeling Continuous Random Events in Communication Systems 95
3.3.1 Cumulative Distribution Functions 99
3.4 Time-Varying Randomness in Communication Systems 101
3.4.1 Stationarity 104
3.5 Gaussian Noise Channels 106
3.5.1 Gaussian Processes 108
3.6 Power Spectral Densities and LTI Systems 109
3.7 Narrowband Noise 110
3.8 Application of Random Variables: Indoor Channel Model 113
3.9 Chapter Summary 114
3.10 Additional Readings 114
References 115

CHAPTER 4
Digital Communications Fundamentals 117
4.1 What Is Digital Transmission? 117
4.1.1 Source Encoding 120
4.1.2 Channel Encoding 122
4.2 Digital Modulation 127
4.2.1 Power Efficiency 128
4.2.2 Pulse Amplitude Modulation 129

4.2.3 Quadrature Amplitude Modulation 131
4.2.4 Phase Shift Keying 133
4.2.5 Power Efficiency Summary 139
4.3 Probability of Bit Error 141
4.3.1 Error Bounding 145
4.4 Signal Space Concept 148
4.5 Gram-Schmidt Orthogonalization 150
4.6 Optimal Detection 154
4.6.1 Signal Vector Framework 155
4.6.2 Decision Rules 158
4.6.3 Maximum Likelihood Detection in an AWGN Channel 159
4.7 Basic Receiver Realizations 160
4.7.1 Matched Filter Realization 161
4.7.2 Correlator Realization 164
4.8 Chapter Summary 166
4.9 Additional Readings 168
References 169

CHAPTER 5
Understanding SDR Hardware 171
5.1 Components of a Communication System 171
5.1.1 Components of an SDR 172
5.1.2 AD9363 Details 173
5.1.3 Zynq Details 176
5.1.4 Linux Industrial Input/Output Details 177
5.1.5 MATLAB as an IIO client 178
5.1.6 Not Just for Learning 180
5.2 Strategies For Development in MATLAB 181
5.2.1 Radio I/O Basics 181
5.2.2 Continuous Transmit 183
5.2.3 Latency and Data Delays 184
5.2.4 Receive Spectrum 185
5.2.5 Automatic Gain Control 186
5.2.6 Common Issues 187
5.3 Example: Loopback with Real Data 187
5.4 Noise Figure 189
References 190

CHAPTER 6
Timing Synchronization 191
6.1 Matched Filtering 191
6.2 Timing Error 195
6.3 Symbol Timing Compensation 198

6.3.1 Phase-Locked Loops 200
6.3.2 Feedback Timing Correction 201
6.4 Alternative Error Detectors and System Requirements 208
6.4.1 Gardner 208
6.4.2 Müller and Mueller 208
6.5 Putting the Pieces Together 209
6.6 Chapter Summary 212
References 212

CHAPTER 7
Carrier Synchronization 213
7.1 Carrier Offsets 213
7.2 Frequency Offset Compensation 216
7.2.1 Coarse Frequency Correction 217
7.2.2 Fine Frequency Correction 219
7.2.3 Performance Analysis 224
7.2.4 Error Vector Magnitude Measurements 226
7.3 Phase Ambiguity 228
7.3.1 Code Words 228
7.3.2 Differential Encoding 229
7.3.3 Equalizers 229
7.4 Chapter Summary 229
References 230

CHAPTER 8
Frame Synchronization and Channel Coding 231
8.1 O Frame, Where Art Thou? 231
8.2 Frame Synchronization 232
8.2.1 Signal Detection 235
8.2.2 Alternative Sequences 239
8.3 Putting the Pieces Together 241
8.3.1 Full Recovery with Pluto SDR 242
8.4 Channel Coding 244
8.4.1 Repetition Coding 244
8.4.2 Interleaving 245
8.4.3 Encoding 246
8.4.4 BER Calculator 251
8.5 Chapter Summary 251
References 251

CHAPTER 9
Channel Estimation and Equalization 253
9.1 You Shall Not Multipath! 253

9.2 Channel Estimation 254
9.3 Equalizers 258
9.3.1 Nonlinear Equalizers 261
9.4 Receiver Realization 263
9.5 Chapter Summary 265
References 266

CHAPTER 10
Orthogonal Frequency Division Multiplexing 267
10.1 Rationale for MCM: Dispersive Channel Environments 267
10.2 General OFDM Model 269
10.2.1 Cyclic Extensions 269
10.3 Common OFDM Waveform Structure 271
10.4 Packet Detection 273
10.5 CFO Estimation 275
10.6 Symbol Timing Estimation 279
10.7 Equalization 280
10.8 Bit and Power Allocation 284
10.9 Putting It All Together 285
10.10 Chapter Summary 286
References 286

CHAPTER 11
Applications for Software-Defined Radio 289
11.1 Cognitive Radio 289
11.1.1 Bumblebee Behavioral Model 292
11.1.2 Reinforcement Learning 294
11.2 Vehicular Networking 295
11.3 Chapter Summary 299
References 299

APPENDIX A
A Longer History of Communications 303
A.1 History Overview 303
A.2 1750–1850: Industrial Revolution 304
A.3 1850–1945: Technological Revolution 305
A.4 1946–1960: Jet Age and Space Age 309
A.5 1970–1979: Information Age 312
A.6 1980–1989: Digital Revolution 313
A.7 1990–1999: Age of the Public Internet (Web 1.0) 316
A.8 Post-2000: Everything comes together 319
References 319

APPENDIX B
Getting Started with MATLAB and Simulink 327
B.1 MATLAB Introduction 327
B.2 Useful MATLAB Tools 327
B.2.1 Code Analysis and M-Lint Messages 328
B.2.2 Debugger 329
B.2.3 Profiler 329
B.3 System Objects 330
References 332

APPENDIX C
Equalizer Derivations 333
C.1 Linear Equalizers 333
C.2 Zero-Forcing Equalizers 335
C.3 Decision Feedback Equalizers 336

APPENDIX D
Trigonometric Identities 337

About the Authors 339

Index 341

CHAPTER 4

Digital Communications Fundamentals

In this chapter, we will provide an overview of several key fundamental concepts employed in the transmission of digital data. Starting with an understanding of how binary data can be used to manipulate the physical characteristics of electromagnetic waveforms, we will then look at the basic anatomy of a digital communication system before concentrating our attention on several different approaches for modulating electromagnetic waveforms using binary data. Once we have established how a digital data transmission process operates, we will then explore one of the key analytical tools for assessing the quantitative performance of such systems: the bit error rate. Finally, this chapter will conclude with an introduction to the design of digital receivers via a signal vector space perspective.

4.1 What Is Digital Transmission?

A digital transceiver is a system composed of a collection of both digital and analog processes that work in concert with each other in order to handle the treatment and manipulation of binary information. The purpose of these processes is to achieve data transmission and reception across some sort of medium, whether it is a twisted pair of copper wires, a fiber optic cable, or a wireless environment. At the core of any digital transceiver system is the *binary digit* or *bit*, which for the purposes of this book is considered to be the fundamental unit of information used by a digital communication system.

Therefore, a digital transceiver is essentially responsible for the translation between a stream of digital data represented by bits and electromagnetic waveforms possessing physical characteristics that uniquely represent those bits. Since electromagnetic waveforms are usually described by sine waves and cosine waves, several physical characteristics of electromagnetic waveforms commonly used to represent digital data per time interval T include the amplitude, phase, and carrier frequency of the waveform, as shown in Figure 4.1. Notice how different combinations of bits represent different amplitude levels or different phase values or different carrier frequency values, where each value uniquely represents a particular binary pattern. Note that in some advanced mapping regimes, binary patterns can potentially be represented by two or more physical quantities.

However, there is much more going on in a digital transceiver than just a mapping between bits and waveforms, as shown in Figure 4.2. In this illustration of the basic anatomy for a digital transceiver, we observe that there are several functional blocks that constitute a communication system. For instance, the mapping between bits and electromagnetic waveform characteristics is represented by the modulation and demodulation blocks. Additionally, there are the source

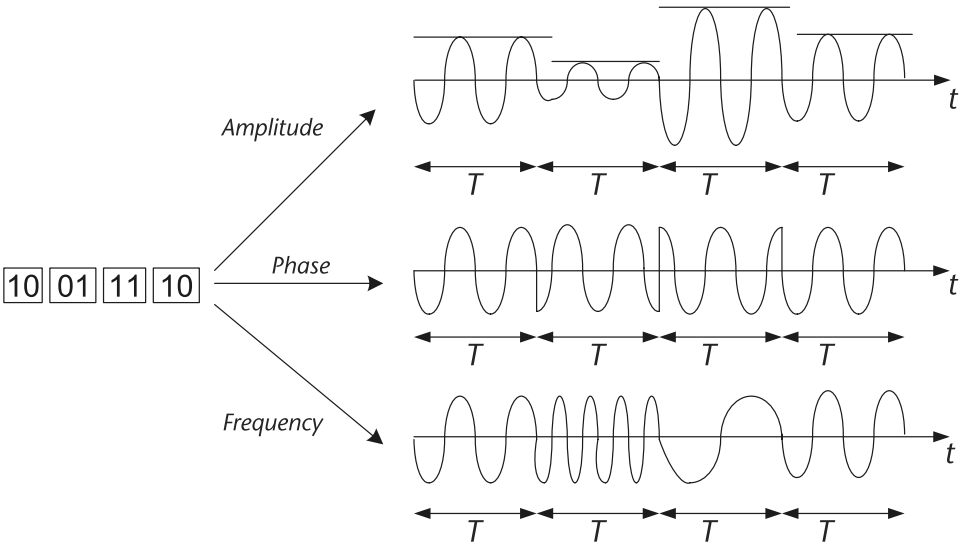

Figure 4.1 Possible mappings of binary information to EM wave properties.

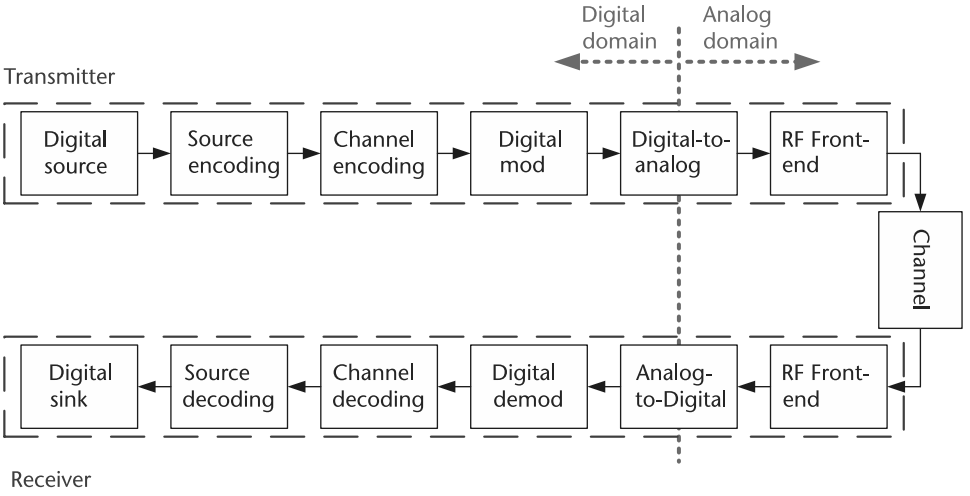

Figure 4.2 Generic representation of a digital communication transceiver.

encoding and source decoding blocks that handle the removal of redundant information from the binary data, channel encoding and channel decoding blocks that introduce a controlled amount of redundant information to protect the transmission for potential errors, and the radio frequency front end (RFFE) blocks that handle the conversation of baseband waveforms to higher carrier frequencies.

One may ask the question, Why do we need all these blocks in our digital communication system? Notice in Figure 4.2 the presence of a channel between the transmitter and the receiver of the digital transmission system. The main reason why the design of a digital communication system tends to be challenging, and that so many blocks are involved in its implementation, is due to this channel. If the channel was an ideal medium where the electromagnetic waveforms from the transmitter are clearly sent to the receiver without any sort of distortion or disturbances, then the design of digital communication systems would be trivial. However, in reality a channel introduces a variety of random impairments to a digital transmission that

can potentially affect the correct reception of waveforms intercepted at the receiver. For instance, a channel may introduce some form of noise that can obfuscate some of the waveform characteristics. Furthermore, in many real-world scenarios many of these nonideal effects introduced by the channel are time-varying and thus difficult to deal with, especially if they vary rapidly in time.

Thus, under real-world conditions, the primary goal of any digital communication system is to transmit a binary message $m(t)$ and have the reconstructed version of this binary message $\hat{m}(t)$ at the output of the receiver to be equal to each other. In other words, our goal is to have $P(\hat{m}(t) \neq m(t))$ as small as needed for a particular application. The metric for quantitatively assessing the error performance of a digital communication system is referred to as the probability of error or BER, which we define as $Pe = P(\hat{m}(t) \neq m(t))$. Note that several data transmission applications possess different P_e requirements due in part to the data transmission rate. For instance, for digital voice transmission, a BER of $Pe \sim 10^{-3}$ is considered acceptable, while for an average data transmission application a BER of $Pe \sim 10^{-5} - 10^{-6}$ is deemed sufficient. On the other hand, for a very high data rate application such as those that employ fiber-optic cables, a BER of $Pe \sim 10^{-9}$ is needed since any more errors would flood a receiver given that the data rates can be extremely high.

To help mitigate errors that may occur due to the impairments introduced by the channel, we will briefly study how source encoding and channel encoding works before proceeding with an introduction to modulation.

Hands-On MATLAB Example: Communication systems convey information by manipulating the physical properties of an electromagnetic signal before it is broadcasted across a medium. Signal properties such as the amplitude, phase, and/or frequency are manipulated over time in such a manner that the receiver can interpret the message being conveyed by the transmitter. These electromagnetic broadcasts often use sine wave signals, which makes them relatively straightforward to manipulate for the purposes of conveying information. In the MATLAB script below, we generate three sine wave-based transmissions, where information is embedded in them via their amplitude levels (amplitude shift keying), phase characteristics (phase shift keying), or frequency values (frequency shift keying). In this script, we generate random binary data using the `rand` function and then round it to the nearest integer (one or zero), and then map those binary values to a corresponding amplitude, phase, or frequency value to be used by a sine wave signal. Note that since sine wave signals are continuous waveforms, we have to approximate this situation by using a large number of discrete points to model the sine wave signal as continuous.

The mapping of these random binary values to the physical attributes of a signal wave signal are shown in Figure 4.3, where we can readily observe how the amplitude (Figure 4.3[b]), phase (Figure 4.3[c]), and frequency (Figure 4.3[d]) values change over time in order to represent the binary values being transmitted to the receiver (see Figure 4.3[a]). In all three case, we use the exact same sine wave signal as the basis for communicating this information, but the sine wave characteristics are changing as a function of time.

Code 4.1 Sending Binary Data via Sinusoidal Signal Manipulation: **chapter4.m**

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Sending binary data via sinusoidal signal manipulation

% Parameters
sig_len = 1000; % Signal length (in samples)
sampl_per_bin = 100; % Samples per binary representation
bin_data_len = sig_len/sampl_per_bin;
          % Length of binary stream is a multiple of signal length
bin_data = round(rand(1,bin_data_len));

% Create sinusoidal carriers
sig_carrier_base = sin(2*pi*(0:(1/sampl_per_bin):
          (1-(1/sampl_per_bin)))); % Baseline carrier
sig_carrier_freq = sin(2*2*pi*(0:(1/sampl_per_bin):
          (1-(1/sampl_per_bin)))); % Double frequency
sig_carrier_phase = sin(2*pi*(0:(1/sampl_per_bin):
          (1-(1/sampl_per_bin)))+(pi/4)); % Phase shifted by 45 degrees

% Modulate sinusoidal carrier via amplitude, phase, and frequency
% manipulations
sig_bin = []; % Binary waveform
sig_ask = []; % Amplitude modulated
sig_psk = []; % Phase modulated
sig_fsk = []; % Frequency modulated
for ind = 1:1:bin_data_len,
    if (bin_data(ind)==1)
        sig_bin = [sig_bin ones(1,sampl_per_bin)];
        sig_ask = [sig_ask sig_carrier_base];
        sig_psk = [sig_psk sig_carrier_base];
        sig_fsk = [sig_fsk sig_carrier_base];
    else
        sig_bin = [sig_bin zeros(1,sampl_per_bin)];
        sig_ask = [sig_ask 0.5*sig_carrier_base];
        sig_psk = [sig_psk sig_carrier_phase];
        sig_fsk = [sig_fsk sig_carrier_freq];
    end;
end;
```

4.1.1 Source Encoding

One of the goals of any communication system is to efficiently and reliably communicate information across a medium from a transmitter to a receiver. As a result, it would be ideal if all the redundant information from a transmission could be removed in order to minimize the amount of information that needs to be sent across the channel, which would ultimately result in a decrease in the amount of time, computational resources, and power being expended on the transmission. Consequently, source encoding is a mechanism designed to remove redundant information in order to facilitate more efficient communications.

The way source encoding operates is by taking a sequence of source symbols $\underline{u}$ and mapping them to a corresponding sequence of source encoded symbols $\underline{v}$, $v_i \in \underline{v}$ as close to random as possible and the components of $\underline{v}$ are uncorrelated (i.e., unrelated). Thus, by performing this source encoding operation we hopefully achieve some level of redundancy minimization in $v_i \in \underline{v}$, thus limiting the amount

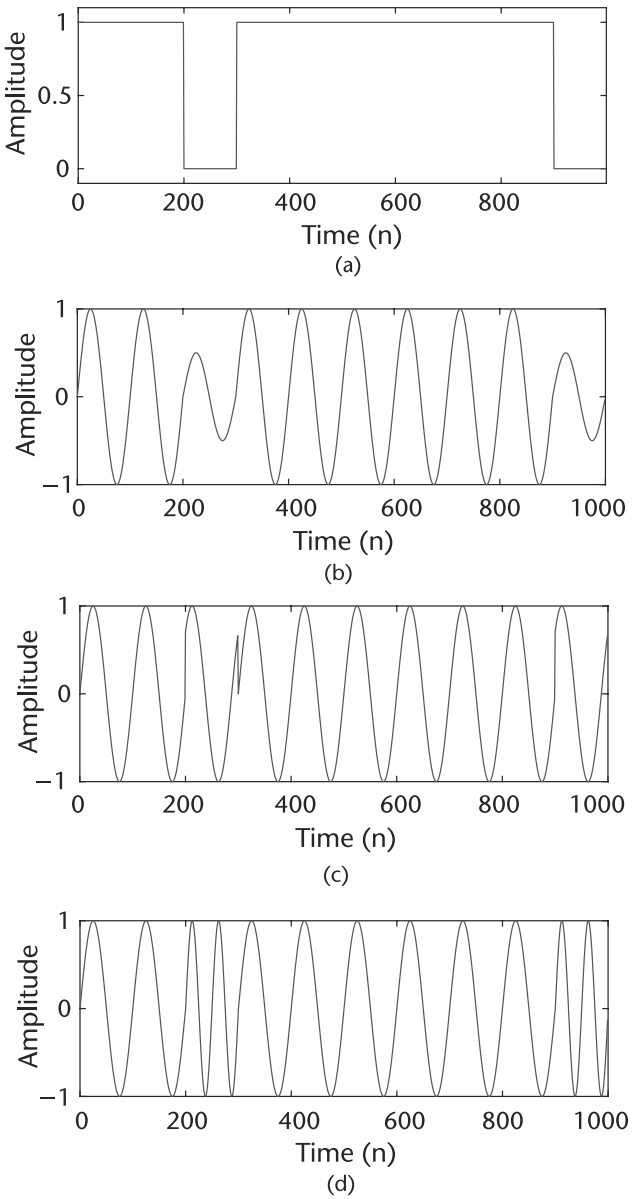

Figure 4.3 Examples of amplitude, phase, and frequency representations of binary transmissions. These sine wave properties are the building blocks for most commonly used modulation schemes used in digital communication systems. (a) Binary signal, (b) amplitude shift keying, (c) phase shift keying, and (d) frequency shift keying.

of wasted radio resources employed in the transmission of otherwise predictable symbols in $\underline{u}$. In other words, a source encoder removes redundant information from the source symbols in order to realize efficient transmission. Note that in order to perform source encoding, the source symbols need to be digital.

Hands-On MATLAB Example: Source coding exploits redundancy in a collection of data by removing it and replacing that redundancy with a short

A single analog television channel occupies 6 MHz of frequency bandwidth. On the other hand, up to eight digitally encoded television channels can fit within the same frequency bandwidth of 6 MHz.

codeword, thus reducing the overall amount of information. In the following MATLAB script, we will illustrate how source coding works when it is applied to data possessing different amounts of redundancy. Using a combination of the `rand` and `round` functions, we generate two binary data vectors, with one possessing an equal amount of one and zero values while the other vector possesses approximately 90% one values and 10% zero values. To compress the data in these vectors, we use an encoding technique where we take all continuous strings of ones in each vector and replace it with a decimal value representing the length of these strings of one. For example, if a binary vector existed, and 15 one values exist betwen two zero values, we would replace those 15 one values with a decimal value (or equivalent codeword) indicating that 15 one values exist in that part of the binary vector.

Based on the implemented encoding scheme, our intuition would dictate that the binary vector with the 90/10 ratio of one to zero values would be compressed more relative to the binary vector with the 50/50 ratio since the former would have a greater likelihood of having long strings of one values to compress. Referring to Figure 4.4, our intuition is confirmed, with a significant reduction in size for the compressed 90/10 binary data stream. On the other hand, we observe that the 50/50 binary vector does not even compress but rather grow in size. This is due to the fact that the amount of overhead needed to replace every string of one values with a corresponding codeword actually takes up more information than the original binary sequence. Consequently, when performing source coding, it is usually worth our while to compress data streams that possess obvious amounts of redundancy, otherwise we can actually make the situation even more inefficient.

4.1.2 Channel Encoding

To protect a digital transmission from the possibility of its information being corrupted, it is necessary to introduce some level of controlled redundancy in order to reverse the effects of data corruption. Consequently, channel encoding is designed to correct for channel transmission errors by introducing controlled redundancy into the data transmission. As opposed to the redundancy that is removed during the source encoding process, which is random in nature, the redundancy introduced by

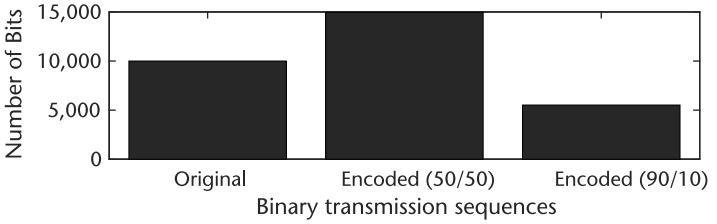

Figure 4.4 Impact of source coding on binary transmissions.

Code 4.2 Reducing Amount of Data Transmission Using Source Coding: **chapter4.m**

```
% Define parameters
len = 10000; % Length of binary data stream

% Have two binary sources, one 50/50 and the other 90/10 in terms of ones
% and zeros
bin1 = round(rand(1,len)); % 50/50 binary
bin2 = round(0.5*rand(1,len)+0.45); %90/10 binary

% Encode strings of ones in terms of the length of these strings
enc_bin1 = [];
enc_bin2 = [];
for ind = 1:1:len,
    if (bin1(ind) == 1)  % Encoding 50/50
        if (ind == 1)
            enc_bin1 = 1;
        else
            enc_bin1(end) = enc_bin1(end)+1;
        end;
    else
        enc_bin1 = [enc_bin1 0];
    end;
    if (bin2(ind) == 1)  % Encoding 90/10
        if (ind == 1)
            enc_bin2 = 1;
        else
            enc_bin2(end) = enc_bin2(end)+1;
        end;
    else
        enc_bin2 = [enc_bin2 0];
    end;
end;

% Find size of encoded binary streams
% (assume all one string length values possess the same number of bits)
ind1 = find(enc_bin1 ~= 0);
ind2 = find(enc_bin2 ~= 0);
[largest_ebin1,ind_largest_ebin1] = max(enc_bin1(ind1));
[largest_ebin2,ind_largest_ebin2] = max(enc_bin2(ind2));
numbits1 = length(dec2bin(largest_ebin1)-'0');
numbits2 = length(dec2bin(largest_ebin2)-'0');
total_size_ebin1 = length(ind1)*numbits1 + length(find(enc_bin1 == 0));
total_size_ebin2 = length(ind2)*numbits2 + length(find(enc_bin2 == 0));
```

a channel encoding is specifically designed to combat the effects of bit errors in the transmission (i.e., the redundancy possesses a specific structure known to both the transmitter and receiver).

In general, channel encoding operates as follows: Each vector of a source encoded output of length K; namely, v_l where $l = 1, 2, ..., 2^K$, is assigned a unique *codeword* such that the vector $v_l = (101010\ldots)$ is assigned a unique codeword $c_l \in \mathbb{C}$ of length N, where $\mathbb{C}$ is a *codebook*. During this process, the channel encoder has introduced $N-K = r$ controlled number of bits to the channel encoding process. The *code rate* of a communications system is equal to the ratio of the number of information bits to the size of the codeword (i.e., the code rate is equal to k/N).

When designing a codebook for a channel encoder, it is necessary to quantitatively assess how well or how poorly the codewords will perform in a situation involving data corruption. Consequently, the *Hamming distance* is often used to determine the effectiveness of a set of codewords contained within a codebook by evaluating the relative difference between any two codewords. Specifically, the Hamming distance $d_H(c_i, c_j)$ between any two codewords, say c_i and c_j, is equal to the number of components in which c_i and c_j are different. When determining the effectiveness of a codebook design, we often are looking for the minimum Hamming distances between codewords; that is,

$$d_{H,\min} = \min_{c_i, c_j \in \mathbb{C},\ i \neq j} d_H(c_i, c_j), \tag{4.1}$$

since our goal is to maximize the minimum Hamming distance in a given codebook to ensure that the probability of accidentally choosing a codeword other than the correct codeword is kept to a minimum. For example, suppose we have a codebook consisting of $\{101, 010\}$. We can readily calculate the minimum Hamming distance to be equal to $d_{H,min} = 3$, which is the best possible result. On the other hand, a codebook consisting of $\{111, 101\}$ possesses a minimum Hamming distance of $d_{H,min} = 1$, which is relatively poor in comparison to the previous codebook example.

In the event that a codeword is corrupted during transmission, *decoding spheres* (also known as *Hamming spheres*) can be employed in order to make decisions on the received information, as shown in Figure 4.5, where codewords that are corrupted during transmission are mapped to the nearest eligible codeword. Note that when designing a codebook, the decoding spheres should not overlap in order to enable the best possible decoding performance at the receiver (i.e., $\rightarrow d_{H,\min} = 2t + 1$).

Hands-On MATLAB Example: Let us apply repetition coding to an actual vector, using a range of repetition rates, and observe its impact when the binary vector is exposed to random bit-flips. The MATLAB script below takes the original binary vector `bin_str` and introduces controlled redundancy into it in the form of repeating these binary values by a factor of N. For example, instead of transmitting `010`, applying a repetition coding scheme with a repetition factor of $N = 3$ would yield an output of `000111000`. The reason this is important is that in the event a bit is flipped from a one to a zero or from a zero to a one, which is considered an error, the other repeated bits could be used to nullify the error at the receiver;

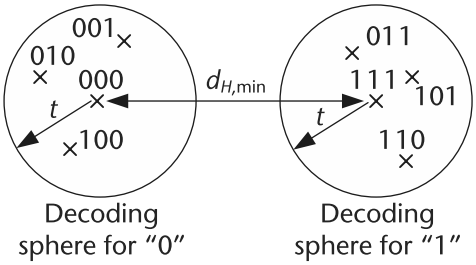

Figure 4.5 Example of decoding spheres.

A rate 1/3 *repetition code* with no source encoding would look like:

$$1 \rightarrow 111 = c_1 \text{ (1st codeword)}$$
$$0 \rightarrow 000 = c_2 \text{ (2nd codeword)}$$
$$\therefore C = \{000, 111\}$$

What are the Hamming distances for the codeword pairs $d_H(111,000)$ and $d_H(111,101)$?

that is, if a zero value is flipped to a one, the other two zero values will inform the receiver that one of the bits is in error and discard that value when decoding the incoming binary vector. Intuitively, we would assume that with the more repetition applied to a binary vector, the more secure it is from corruption; for example, there are circumstances = where more than one bit is corrupted, which could still yield an error unless there are even more repeated bits to inform the receiver otherwise.

Code 4.3 Protect Data Using Simple Repetition Channel Coding: **`chapter4.m`**

```
% Define parameters
len = 100000; % Length of original binary data stream
N1 = 3; % First repetition factor; should be odd to avoid tie
N2 = 5; % Second repetition factor; should be odd to avoid tie
N3 = 7; % Third repetition factor; should be odd to avoid tie

% Generate binary data stream
bin_str = round(rand(1,len));

% Employ repetition code with repetition factors N1, N2, N3
chcode1_bin_str = zeros(1,N1*len);
chcode2_bin_str = zeros(1,N2*len);
chcode3_bin_str = zeros(1,N3*len);
for ind = 1:1:max([N1 N2 N3]),
    if (ind<=N1)
        chcode1_bin_str(ind:N1:(N1*(len-1)+ind))=bin_str;
    end;
    if (ind<=N2)
        chcode2_bin_str(ind:N2:(N2*(len-1)+ind))=bin_str;
    end;
    if (ind<=N3)
        chcode3_bin_str(ind:N3:(N3*(len-1)+ind))=bin_str;
    end;
end;

% Corrupt both binary strings with zero-mean unit variance Gaussian
% noise followed by rounding (creates "bit flipping" errors)
noisy_bin_str = bin_str + randn(1,len);
rx_bin_str0 = zeros(1,len);
ind0 = find(noisy_bin_str >= 0.5);
rx_bin_str0(ind0) = 1;
noisy_chcode1_bin_str = chcode1_bin_str + randn(1,N1*len);
rx_chcode1_bin_str = zeros(1,N1*len);
ind1 = find(noisy_chcode1_bin_str >= 0.5);
rx_chcode1_bin_str(ind1) = 1;
```

```
noisy_chcode2_bin_str = chcode2_bin_str + randn(1,N2*len);
rx_chcode2_bin_str = zeros(1,N2*len);
ind2 = find(noisy_chcode2_bin_str >= 0.5);
rx_chcode2_bin_str(ind2) = 1;
noisy_chcode3_bin_str = chcode3_bin_str + randn(1,N3*len);
rx_chcode3_bin_str = zeros(1,N3*len);
ind3 = find(noisy_chcode3_bin_str >= 0.5);
rx_chcode3_bin_str(ind3) = 1;

% Decode three encoded binary sequences
dec1_bin = (vec2mat(rx_chcode1_bin_str,N1)).';
dec2_bin = (vec2mat(rx_chcode2_bin_str,N2)).';
dec3_bin = (vec2mat(rx_chcode3_bin_str,N3)).';
ind11 = find(((sum(dec1_bin,1))/N1) >= 0.5);
ind12 = find(((sum(dec2_bin,1))/N2) >= 0.5);
ind13 = find(((sum(dec3_bin,1))/N3) >= 0.5);
rx_bin_str1 = zeros(1,len);
rx_bin_str1(ind11) = 1;
rx_bin_str2 = zeros(1,len);
rx_bin_str2(ind12) = 1;
rx_bin_str3 = zeros(1,len);
rx_bin_str3(ind13) = 1;

% Calculate bit error rate
ber0 = sum(abs(bin_str - rx_bin_str0))/len;
ber1 = sum(abs(bin_str - rx_bin_str1))/len;
ber2 = sum(abs(bin_str - rx_bin_str2))/len;
ber3 = sum(abs(bin_str - rx_bin_str3))/len;
```

Based on the MATLAB script, we can see the impact of repetition coding on a binary vector being corrupted by bit-flipping in an error-prone environment (see Figure 4.6). As we introduce more controlled redundancy in the form of larger repetition rates, the amount of bit errors present in the transmission decreases gradually. This makes sense since as we are introducing more resources into the transmission to make it more reliable in an error-prone environment, the rate at which errors occur will start to decrease. However, one should note that there is a cost-benefit trade-off here since we are introducing more resources into the transmission but this may or may not linearly correlate with the benefits we are obtaining.

4.1.2.1 Shannon's Channel Coding Theorem

In digital communications, it is sometimes necessary to determine the upper limit of the data rate for a specific digital transceiver design. Consequently, in 1949

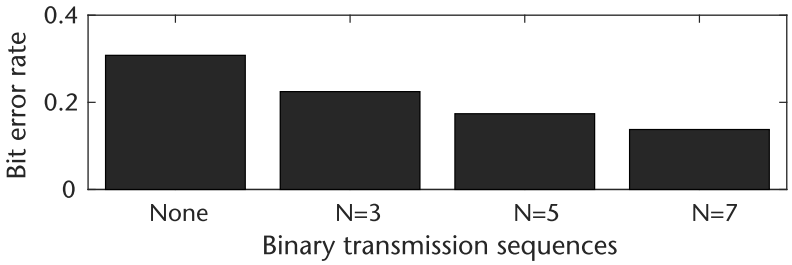

Figure 4.6 Impact of repetition coding on binary transmissions for different repetition factors N.

Claude Shannon published his seminar paper that addressed this topic, entitled "Communication in the Presence of Noise" [1]. In this paper, he defined a quantitative expression that described the limit on the data rate, or capacity, of a digital transceiver in order to achieve error-free transmission.

Suppose one considers a channel with capacity C and we transmit data at a fixed code rate of K/N, which is equal to R_c (a constant). Consequently, if we increase N, then we must increase K in order to keep R_c equal to a constant. What Shannon states is that a code exists such that for $R_c = K/N < C$ and as $N \to \infty$, we have the probability of error $P_e \to 0$. Conversely, for $R_c = K/N \geq C$, Shannon indicated that no such code exists. Hence, C is the limit in rate for reliable communications (i.e., C is the *absolute limit* that you cannot go any faster than this amount without causing errors).

So why is the result so important? First, the concept of reliability in digital communications is usually expressed as the probability of bit error, which is measured at the output of the receiver. As a result, it would be convenient to know what this capacity is given the transmission bandwidth, B, the received SNR using mathematical tools rather than empirical measurements. Thus, Shannon derived the information capacity of the channel, which turned out to be equal to

$$C = B\log_2(1 + SNR) \qquad \text{[b/s]}, \tag{4.2}$$

where this information capacity tells us the achievable data rate. Note that Shannon only provided us with the theoretical limit for the achievable capacity of a data transmission, but he does not tell us how to build a transceiver to achieve this limit.

Second, the information capacity of the channel is useful since this expression provides us with a bound on the achievable data rate given bandwidth B and received SNR, employed in the ratio $\eta = R/C$, where R is the signaling rate and C is the channel capacity. Thus, as $\eta \to 1$, the system becomes more efficient. Therefore, the capacity expression provides us with a basis for trade-off analysis between B and SNR, and it can be used for comparing the noise performance of one modulated scheme versus another.

4.2 Digital Modulation

In analog modulation schemes, the analog message signal modulates a continuous wave prior to transmission across a medium. Conversely, digital modulation involves having a digital message signal modulating a continuous waveform. As we have seen earlier in this chapter, this can be accomplished by uniquely manipulating the amplitude and phase information of a signal during each symbol period T based on a specific pattern of bits. However, most digital modulation techniques possess an intermediary step that we will focus on in this section, where collections of b bits forming a binary message m_b are mapped to a symbol, which is then used to define the physical characteristics of a continuous waveform in terms of amplitude and phase. In particular, for each of the possible 2^b values of m_b, we need a unique signal $s_i(t)$, $1 \leq i \leq 2^b$ that can then be used to modulate the continuous waveform, as shown in Figure 4.7.

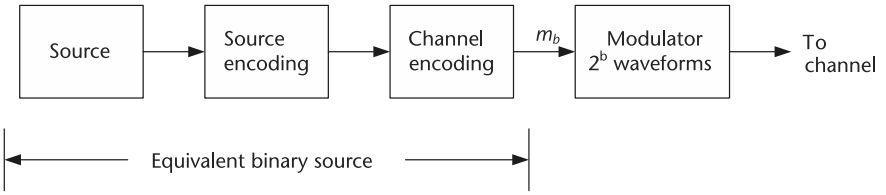

Figure 4.7 Modulation process of equivalent binary data.

In this section, we will study several different families of approaches for mapping binary data into symbols that can then be used to modulate continuous waveforms. These modulation scheme families are defined by which physical characteristic or combination of characteristics are manipulated by the mapping process in order to uniquely represent a specific bit pattern. However, there exist various trade-offs between these different families, including how efficiently a bit is mapped to a symbol in terms of the transmit power expended. Consequently, we will first explore how we assess this trade-off before studying three major families of digital modulation schemes and how they compare to each other.

4.2.1 Power Efficiency

In order to assess the effectiveness of mapping a bit to a symbol in terms of the transmit power expended per symbol, we can employ the *power efficiency* metric. Suppose we define the energy of a symbol $s(t)$ as

$$E_s = \int_0^T s^2(t)dt, \tag{4.3}$$

where T is the period of the symbol. Then, for a modulation scheme consisting of M symbols, we can define the average symbol energy via the following weighted average:

$$\bar{E}_s = P(s_1(t)) \cdot \int_0^T s_1^2(t)dt + \cdots + P(s_M(t)) \cdot \int_0^T s_M^2(t)dt, \tag{4.4}$$

where $P(s_i(t))$ is the probability that the symbol $s_i(t)$ occurs. Furthermore, if we would like to calculate the average energy per bit, we can approximate this using $\bar{E}_s$ and dividing this quantity by $b = \log_2(M)$ bits per symbol, yielding

$$\bar{E}_b = \frac{\bar{E}_s}{b} = \frac{\bar{E}_s}{\log_2(M)}. \tag{4.5}$$

To quantitatively assess the similarity between two symbols in terms of their physical characteristics, we define the *Euclidean distance* as

$$d_{ij}^2 = \int_0^T (s_i(t) - s_j(t))^2 dt = E_{\Delta s_{ij}}, \tag{4.6}$$

where $\Delta s_{ij}(t) = s_i(t) - s_j(t)$. Since we are often interested in the worst-case scenario when assessing the performance of a modulation scheme, we usually compute the

minimum Euclidean distance; namely:

$$d_{min}^2 = \min_{s_i(t), s_j(t), i \neq j} \int_0^T (s_i(t) - s_j(t))^2 dt. \tag{4.7}$$

Thus, the power efficiency of a signal set used for modulation is given by the expression

$$\varepsilon_p = \frac{d_{\min}^2}{\bar{E}_b}. \tag{4.8}$$

Suppose we would like to find the ε_p given the following waveforms:

$$s_1(t) = A \cdot [u(t) - u(t - T)] = s(t)$$
$$s_2(t) = -A \cdot [u(t) - u(t - T)] = -s(t)$$

where $u(t)$ is the unit step function. Compute the following:

- The minimum Euclidean distance $d_{\min}^2$.
- The average bit energy $\bar{E}_b$.
- The power efficiency ε_P.

4.2.2 Pulse Amplitude Modulation

Of the various physical characteristics of a signal waveform that can be manipulated in order to convey digital information, the most obvious choice is the signal amplitude level. Leveraging this physical characteristic, *pulse amplitude modulation* (PAM) is a digital modulation scheme where the message information is encoded in the amplitude of a series of signal pulses. Furthermore, demodulation of a PAM transmission is performed by detecting the amplitude level of the carrier at every symbol period.

The most basic form of PAM is binary PAM (B-PAM), where the individual binary digits are mapped to a waveform $s(t)$ possessing two amplitude levels according to the following modulation rule:

- "$1''$ $\rightarrow$ $s_1(t)$
- "$0''$ $\rightarrow$ $s_2(t)$

where $s_1(t)$ is the waveform $s(t)$ possessing one unique amplitude level while $s_2(t)$ is also based on the waveform $s(t)$ but possesses another unique amplitude level. Note that the waveform $s(t)$ is defined across a time period T and is zero otherwise. Since the duration of the symbols is equivalent to the duration of the bits, the bit rate for a B-PAM transmission is defined as $R_b = 1/T$ bits per second.

The energy of the waveform $s(t)$ is defined as

$$E_s = \int_0^T s^2(t) dt \quad \text{(Joules)}. \tag{4.9}$$

Suppose we define $s(t)$ as a rectangular waveform; namely,

$$s(t) = A \cdot [u(t) - u(t - T)], \tag{4.10}$$

where $u(t)$ is the unit step function and A is the signal amplitude. Furthermore, suppose that the bit "1" is defined by the amplitude A while the bit "0" is defined by the amplitude $-A$. We can subsequently write our modulation rule to be equal to

- "1" $\rightarrow s(t)$
- "0" $\rightarrow -s(t)$

Therefore, the symbol energy is given by $E_s{=}E_{-s}{=}A^2T{=}\frac{A^2}{R_b}$. From this result, we can define the energy per bit for a B-PAM transmission as

$$\bar{E}_b = P(1) \cdot \int_0^T s_1^2(t)dt + P(0) \cdot \int_0^T s_2^2(t)dt, \tag{4.11}$$

where $P(1)$ is the probability that the bit is a "1," and $P(0)$ is the probability that the bit is a "0." Thus, if we define $s_1(t) = s(t)$ and $s_2(t) = -s(t)$, then the average energy per bit is equal to

$$\bar{E}_b = E_s\{P(1) + P(0)\} = E_s = \int_0^T s^2(t)dt = A^2T. \tag{4.12}$$

Calculating the minimum Euclidean distance, we get

$$d_{min}^2 = \int_0^T (s(t) - (-s_(t)))^2 dt = \int_0^T (2s(t))^2 dt = 4A^2T, \tag{4.13}$$

which is then plugged into (4.8) in order to yield a power efficiency result for a B-PAM transmission of

$$\varepsilon_p = \frac{d_{\min}^2}{\bar{E}_b} = \frac{4A^2T}{A^2T} = 4. \tag{4.14}$$

As we will observe throughout the rest of this section, a power efficiency result of 4 is the best possible result that you can obtain for any digital modulation scheme when all possible binary sequences are each mapped to a unique symbol.

Suppose we now generalize the B-PAM results obtained for the average bit energy, the minimum Euclidean distance, and the power efficiency and apply them to the case when we try mapping binary sequences to one of M possible unique signal amplitude levels, referred to as M-ary pulse amplitude modulation (M-PAM). First, let us express the M-PAM waveform as

$$s_i(t) = A_i \cdot p(t), \text{ for } i = 1, 2, \ldots, M/2 \tag{4.15}$$

where $A_i = A(2i - 1)$, $p(t) = u(t) - u(t - T)$, and $u(t)$ is the unit step function. Graphically speaking, the M-PAM modulation scheme looks like the signal constellation shown in Figure 4.8.

In order to compute the power efficieny of M-PAM, $\varepsilon_{p,\mathrm{M-PAM}}$, we select the $d_{\min}^2$ pair $s_1(t) = A \cdot p(t)$ and $s_2(t) = -A \cdot p(t)$ since this pair of signal waveforms are the closest to each other in terms of shape. Thus, using this selection of waveforms, we can solve for the difference between them:

$$\Delta s(t) = 2A \cdot p(t), \tag{4.16}$$

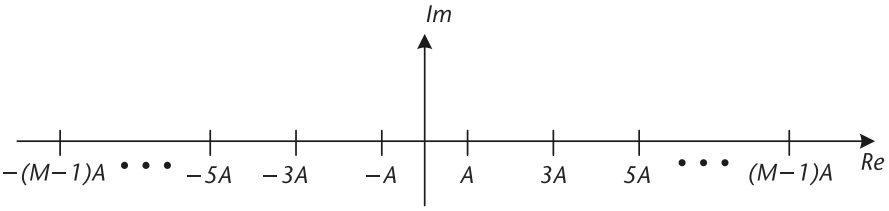

Figure 4.8 M-PAM signal constellation.

which yields a minimum Euclidean distance of

$$d_{\min}^2 = 4A^2T. \tag{4.17}$$

In order to calculate the average symbol energy, $\bar{E}_s$, we can simplify the mathematics by exploiting the symmetry of signal constellation, which yields

$$\begin{aligned} \bar{E}_s &= \frac{2}{M}A^2T\sum_{i=1}^{M/2}(2i-1)^2 \\ &= A^2T\frac{(M^2-1)}{3} \quad \text{which is simplified via tables} \\ &\rightarrow \bar{E}_b = \frac{\bar{E}_s}{\log_2(M)} = \frac{A^2T(2^{2b}-1)}{3b}. \end{aligned} \tag{4.18}$$

Finally, solving for the power efficiency yields

$$\varepsilon_{p,\text{M-PAM}} = \frac{12b}{2^{2b}-1}. \tag{4.19}$$

4.2.3 Quadrature Amplitude Modulation

Similar to PAM, quadrature amplitude modulation (QAM) implies some sort of amplitude modulation. However, QAM modulation is a two-dimensional signal modulation scheme as opposed to PAM modulation. The two dimensions of the QAM modulation; namely, the in-phase and quadrature components, are orthogonal to each other, which implies that one can essentially double the transmission data rate for free. Furthermore, rectangular QAM can be thought of as two orthogonal PAM signals being transmitted simultaneously.

Mathematically, if a rectangular QAM signal constellation consists of M unique waveforms, this could potentially be represented as $\sqrt{M}$-PAM transmissions operating simultaneously in orthogonal dimensions. Note that QAM signal constellations could also take the form of nested circles (called circular QAM), or any other geometric pattern that involves orthogonal modulators. Rectangular QAM is a popular modulation scheme due to its relatively simple receiver structure, as shown in Figure 4.9, where each dimension employs a $\sqrt{M}$-ary PAM detector.

In order to determine the power efficiency of M-QAM, let us first define the mathematical representation of a signal waveform belonging to this form of modulation:

$$s_{ij}(t) = A_i \cdot \cos(\omega_c t) + B_j \cdot \sin(\omega_c t), \tag{4.20}$$

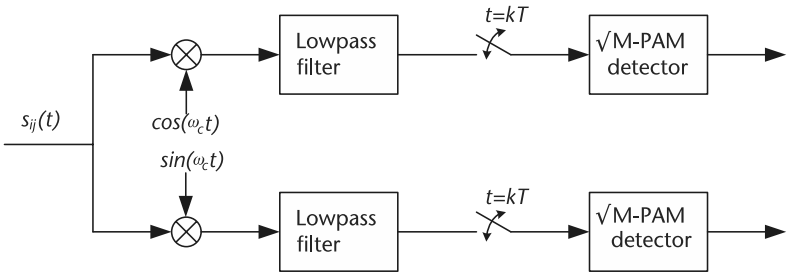

Figure 4.9 M-QAM receiver structure.

where ω_c is the carrier frequency, and A_i and B_j are the in-phase and quadrature amplitude levels. Notice how the cosine and sine functions are used to modulate these amplitude levels in orthogonal dimensions. Visualizing M-QAM as a signal constellation, the signal waveforms will assume positions in both the real and imaginary axes, as shown in Figure 4.10.

To compute the power efficiency of M-QAM, $\varepsilon_{p,\text{M-QAM}}$, we first need to calculate the minimum Euclidean distance, which becomes

$$d_{\min}^2 = \int_0^T \Delta s^2(t)dt = 2A^2T, \tag{4.21}$$

where we have selected the following signal waveforms without loss of generality:

$$\begin{aligned} s_1(t) &= A \cdot \cos(\omega_c t) + A \cdot \sin(\omega_c t) \\ s_2(t) &= 3A \cdot \cos(\omega_c t) + A \cdot \sin(\omega_c t). \end{aligned} \tag{4.22}$$

In order to derive the average symbol energy, $\bar{E}_s$, we leverage the expression from M-ary PAM by replacing M with $\sqrt{M}$ such that

$$\bar{E}_s = A^2T\frac{M-1}{3}, \tag{4.23}$$

which can then be used to solve

$$\bar{E}_b = \frac{\bar{E}_s}{\log_2(M)} = A^2T\frac{2^b-1}{3b}. \tag{4.24}$$

Thus, the power efficiency is equal to

$$\varepsilon_{p,\text{M-QAM}} = \frac{3!b}{2^b-1}. \tag{4.25}$$

Hands-On MATLAB Example: QAM modulation is a very useful technique for sending information efficiently within the same symbol period. As mentioned previously, it exploits both the in-phase and quadrature domains in order to transmit information in parallel down both channels orthogonally. In the MATLAB script above, we implement a QAM transmitter and receiver that takes binary vectors `samp_I` and `samp_Q`, modulates them to the in-phase and quadrature channels of a QAM transmission, and then extracts these binary vectors using QAM

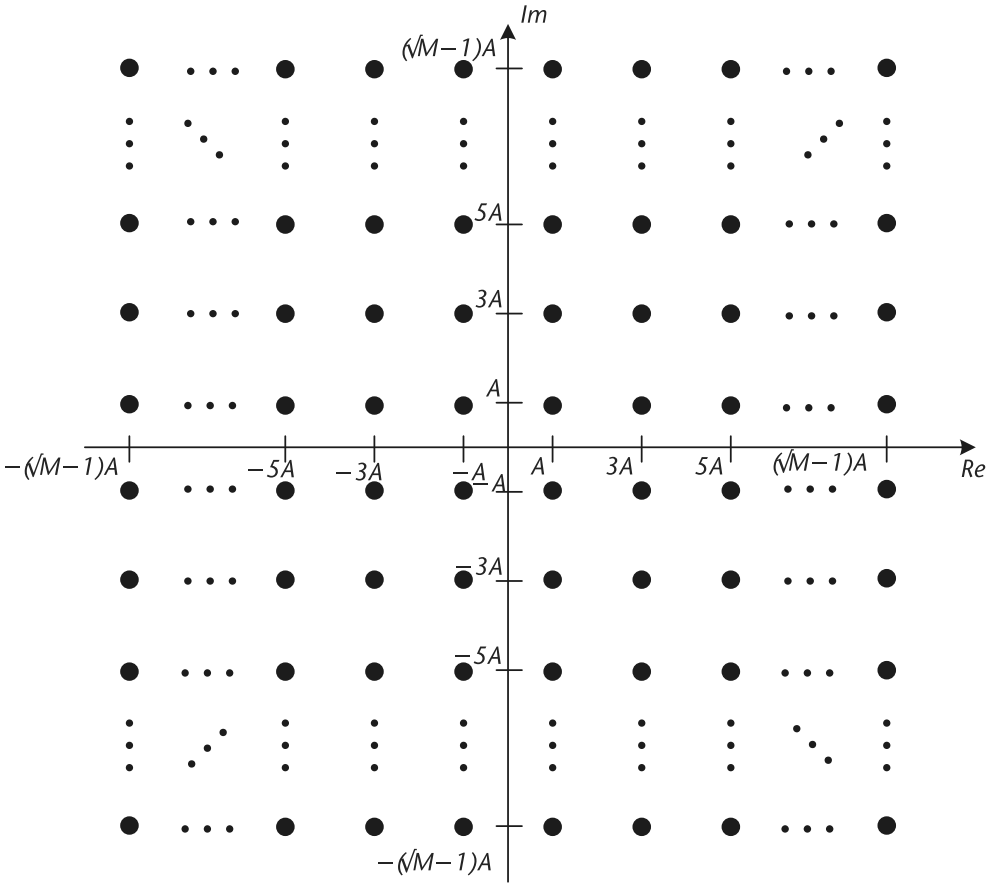

Figure 4.10 M-QAM signal constellation.

demodulation. At the core of QAM modulation are the sine and cosine functions, which are mathematically orthogonal. As we can see in the script, the in-phase and quadrature data is modulated onto the cosine and sine wave signals at the transmitter. At the receiver, the in-phase and quadrature information is separated out of the received signal by exploiting this orthogonality and using trigonometric properties.

In Figure 4.11, we can observe the usage cosine and sine waves as carriers of information, where this information can be transmitted simultaenously and recovered perfectly. By exploiting two dimensions for the transmission of information, we make more efficient use of each symbol period that we use when broadcasting data.

4.2.4 Phase Shift Keying

Phase shift keying (PSK) is a digital modulation scheme that conveys data by changing or modulating the phase of a reference signal (i.e., the carrier wave). Any digital modulation scheme uses a finite number of distinct signals to represent digital data. PSK uses a finite number of phases, each assigned a unique pattern of binary digits. Usually, each phase encodes an equal number of bits. Each pattern of bits forms the symbol that is represented by the particular phase. The demodulator,

Code 4.4 Decoding QAM Waveform Using I/Q Receiver: **chapter4.m**

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Decoding QAM waveform using I/Q receiver

% Define parameters
N_samp = 1000; % Number of samples per symbol
N_symb = 10; % Number of symbols in transmission
cfreq = 1/10; % Carrier frequency of cosine and sine carriers

% Generate inphase and quadrature channels with 2-PAM waveforms
chI = 2*round(rand(1,N_symb))-1;
chQ = 2*round(rand(1,N_symb))-1;
samp_I = [];
samp_Q = [];
for ind = 1:1:N_symb,
    samp_I = [samp_I chI(ind)*ones(1,N_samp)];
    samp_Q = [samp_Q chQ(ind)*ones(1,N_samp)];
end;

% Apply cosine and sine carriers to inphase and quadrature components,
% sum waveforms together into composite transmission
tx_signal = samp_I.*cos(2.*pi.*cfreq.*(1:1:length(samp_I)))
             + samp_Q.*sin(2.*pi.*cfreq.*(1:1:length(samp_Q)));

% Separate out inphase and quadrature components from composite
```

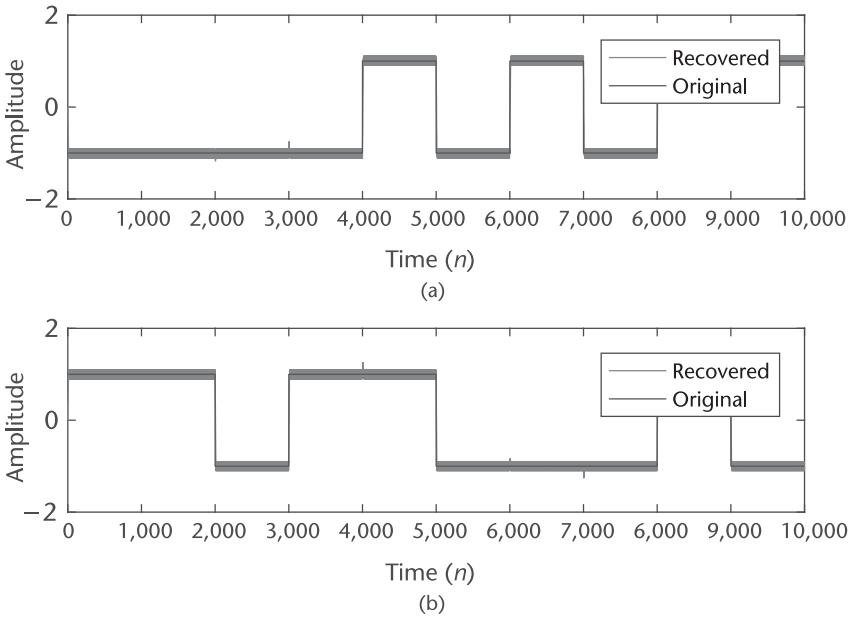

Figure 4.11 Example of quadrature amplitude modulation waveform using an in-phase/quadrature receiver. (a) In-phase signal component, and (b) quadrature signal component.

which is designed specifically for the symbol set used by the modulator, determines the phase of the received signal, and maps it back to the symbol it represents, thus recovering the original data. This requires the receiver to be able to compare the phase of the received signal to a reference signal. Such a system is termed coherent.

PSK characterizes symbols by their phase. Mathematically, a PSK signal waveform is represented by

$$s_i(t) = A\cos(2\pi f_c t + (2i-1)\frac{\pi}{m}), \quad \text{for} \quad i = 1, ..., \log_2 \text{m}, \tag{4.26}$$

where A is the amplitude, f_c is carrier frequency, and $(2i-1)\frac{\pi}{m}$ is the phase offset of each symbol. PSK presents an interesting set of trade-offs with PAM and QAM. In amplitude modulation schemes, channel equalization is an important part of decoding the correct symbols. In PSK schemes, the phase of the received signal is much more important than the amplitude information.

There are several types of PSK modulation schemes based on the number of M possible phase values a particular PSK waveform can be assigned. One of the most popular and most robust is binary PSK, or B-PSK, modulation, whose signal constellation is illustrated in Figure 4.12. In general, the modulation rule for B-PSK modulation is the following:

$$\begin{aligned} \text{“1”} \rightarrow s_1(t) &= A \cdot \cos(\omega_c t + \theta) \\ \text{“0”} \rightarrow s_2(t) &= -A \cdot \cos(\omega_c t + \theta) \\ &= A \cdot \cos(\omega_c(t) + \theta + \pi) \\ &= -s_1(t). \end{aligned} \tag{4.27}$$

In other words, the two signal waveforms that constitute a B-PSK modulation scheme are separated in phase by θ.

In order to derive the power efficiency of a B-PSK modulation scheme, $\varepsilon_{p,\text{BPSK}}$, we first need to compute the minimum Euclidean distance $d^2_{\min}$ by employing the definition and solving for

$$\begin{aligned} d^2_{\min} &= \int_0^T (s_1(t) - s_2(t))^2 dt \\ &= 4A^2 \int_0^T \cos^2(\omega_c t + \theta) dt \end{aligned}$$

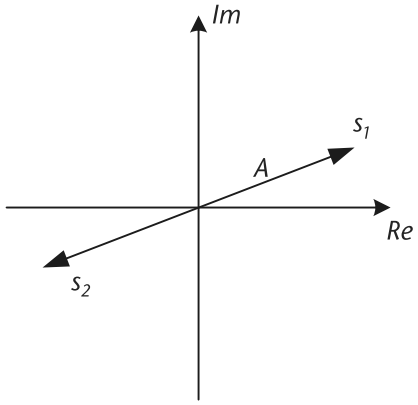

Figure 4.12 BPSK signal constellation.

Notice how in (4.28) how the second term disappeared from the final result. This is due to the fact that the second term possessed a carrier frequency that was twice that of the original signal. Since the carrier signal is a periodic sinusoidal waveform, integrating such a signal possessing such a high frequency would result in the positive portion of the integration canceling out the negative portion of the integration, yielding an answer than is close to zero. Consequently, we refer to this term as a *double frequency term*. Also note that many communication systems filter their received signals, which means the probability of filtering out the double frequency term is also quite high.

$$= \frac{4A^2T}{2} + \frac{4A^2}{2}\int_0^T \cos(2\omega_c t + 2\theta)dt$$
$$= 2A^2T. \tag{4.28}$$

Note that another way for computing $d^2_{\min}$ is to use the concept of correlation, which describes the amount of similarity between two different signal waveforms. In this case, we can express the minimum Euclidean distance as

$$d^2_{\min} = \int_0^T (s_2(t) - s_1(t))^2 dt = E_{s_1} + E_{s_2} - 2\rho_{12} \tag{4.29}$$

where the symbol energy for symbol i, E_{s_i}, and the correlation between symbols 1 and 2, ρ_{12}, are given by

$$E_{s_i} = \int_0^T s_i^2(t)dt \quad \text{and} \quad \rho_{12} = \int_0^T s_1(t)s_2(t)dt.$$

Employing the definition for $\bar{E}_b$, we can now solve for the average bit energy of the B-PSK modulation scheme by first solving for the symbol energies of the two signal waveforms and then averaging them; that is,

$$\begin{aligned} E_{s_1} &= \int_0^T s_1^2(t)dt = A^2\int_0^T \cos^2(\omega_c t + \theta)dt \\ &= \frac{A^2T}{2} + \frac{A^2}{2}\int_0^T \cos(2\omega_c t + 2\theta)dt \\ &= \frac{A^2T}{2} \\ E_{s_2} &= \frac{A^2T}{2} \\ \bar{E}_b &= P(0)\cdot E_{s_2} + P(1)\cdot E_{s_1} = \frac{A^2T}{2}. \end{aligned} \tag{4.30}$$

Note that since the number of bits represented by a single symbol is equal to one, and both the bit energy and symbol energy are equivalent.

Finally, applying the definition for the power efficiency, we get the following expression:

$$\varepsilon_{p,\text{BPSK}} = \frac{d_{\min}^2}{\bar{E}_b} = 4. \tag{4.31}$$

This is supposed to be the largest possible value for ε_p for a modulation scheme employing all possible signal representations; that is, $M = 2^b$ waveforms. Notice when using the correlation approach to calculate the minimum Euclidean distance, in order to get a large ε_p, we need to maximize $d_{\min}^2$, which means we want $\rho_{12} < 0$. Thus, to achieve this outcome, we need the following situation:

$$E_{s_1} = E_{s_2} = E = A^2T/2, \tag{4.32}$$

which means $d_{\min}^2 = 2(E - \rho_{12})$ and consequently $\rho_{12} = -E$.

Show that for the following signal waveforms:

$$s_1(t) = A \cdot \cos(\omega_c t + \theta)$$
$$s_2(t) = 0$$

the power efficiency is equal to $\varepsilon_p = 2$.

Show that for the following signal waveforms:

$$s_1(t) = A \cdot \cos(\omega_c t + \theta)$$
$$s_2(t) = A \cdot \sin(\omega_c t + \theta)$$

the power efficiency is equal to $\varepsilon_p = 2$.

So far we have studied a PSK modulation scheme that consists of only just one of two waveforms. We will now expand our PSK signal constellation repertoire to include four distinct waveforms per modulation scheme. In quadrature PSK (QPSK) modulation, a signal waveform possesses the following representation:

$$s_i(t) = \pm A \cdot \cos(\omega_c t + \theta) \pm A \cdot \sin(\omega_c t + \theta), \tag{4.33}$$

where each signal waveform possesses the same amplitude but one of four possible phase values. This kind of phase modulation is illustrated by the signal constellation diagram shown in Figure 4.13, where each waveform is located at a different phase value.

In order to derive the power efficiency of QPSK, $\varepsilon_{p,\text{QPSK}}$, we first need to solve for the minimum Euclidean distance, $d_{\min}^2$, which is equal to

$$d_{\min}^2 = \int_0^T \Delta s^2(t)dt = 2A^2T. \tag{4.34}$$

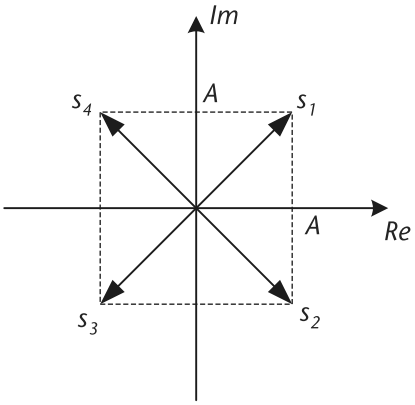

Figure 4.13 QPSK signal constellation.

Next, we would like to find $\bar{E}_b$, which requires us to average over all the signal waveforms. Consequently, this is equal to

$$\bar{E}_b = \frac{(E_{s_1} + E_{s_2} + E_{s_3} + E_{s_4})/4}{\log_2(M)} = \frac{A^2T}{2}, \tag{4.35}$$

where the symbol energy of all four symbols is equal to $E_{s_1} = E_{s_2} = E_{s_3} = E_{s_4} = A^2T$. Finally, solving for the power efficiency using (4.8), we get

$$\varepsilon_{p,\text{QPSK}} = \frac{d^2_{\min}}{\bar{E}_b} = 4, \tag{4.36}$$

which is the same as BPSK but with 2 bits per symbol, making this a fantastic result!

Finally, let us study the general case when a PSK modulation scheme has a choice of M possible phase values, where the distance of a signal constellation point to the origin is always a constant and the signal constellation consists of M equally spaced points on a circle. Referred to as M-PSK, a signal waveform can be mathematically represented as

$$s_i(t) = A \cdot \cos\left(\omega_c t + \frac{2\pi i}{M}\right), \text{ for } i = 0, 1, 2, \ldots, M-1. \tag{4.37}$$

Note that there are several advantages and disadvantages with this modulation scheme. For instance, as M increases the spacing between signal constellation points decreases, thus resulting in a decrease in error robustness. Conversely, having the information encoded in the phase results in constant envelope modulation, which is good for nonlinear power amplifiers and makes the transmission robust to amplitude distortion channels.

Regarding the derivation of the power efficiency for an M-PSK modulation scheme, $\varepsilon_{p,\text{M-PSK}}$, suppose we define two adjacent M-PSK signal waveforms as $s_1(t) = A \cdot \cos(\omega_c t)$ and $s_2(t) = A \cdot \cos(\omega_c t + 2\pi/M)$. Calculating the minimum Euclidean distance using

$$d^2_{\min} = E_{s_1} + E_{s_2} - 2\rho_{12} \tag{4.38}$$

where we define the symbol energy as

$$E_{s_i} = \int_0^T s_i^2(t)dt = \frac{A^2T}{2}, \text{ for } i = 1, 2, \tag{4.39}$$

and the correlation between the two signal waveforms as

$$\rho_{12} = \int_0^T s_1(t)s_2(t)dt = \frac{A^2T}{2}\cos\left(\frac{2\pi}{M}\right), \tag{4.40}$$

this yields

$$d_{\min}^2 = A^2T\left(1 - \cos\left(\frac{2\pi}{M}\right)\right). \tag{4.41}$$

The average bit energy $\bar{E}_b$ is equal to $\bar{E}_b = \frac{\bar{E}_s}{\log_2(M)} = \frac{\bar{E}_s}{b}$, where $\bar{E}_s = A^2T/2$. Using the definition for the power efficiency from (4.8), we see that

$$\varepsilon_{p,\text{M-PSK}} = 2b\left(1 - \cos\left(\frac{2\pi}{M}\right)\right) = 4b\sin^2\left(\frac{\pi}{2^b}\right). \tag{4.42}$$

4.2.5 Power Efficiency Summary

After examining the power efficiency performance of several different modulation schemes, it is important to assess the trade-offs between the different schemes such that we can make the appropriate design decisions in the future when implementing a digital communication system. To determine how much power efficiency we are losing relative to $\varepsilon_{p,\text{QPSK}}$, which possesses the best possible result, we use the following expression:

$$\delta\text{SNR} = 10 \cdot \log_{10}\left(\frac{\varepsilon_{p,\text{QPSK}}}{\varepsilon_{p,\text{other}}}\right). \tag{4.43}$$

Using this expression, we created a table of δSNR values for the modulation schemes studied in this chapter, as shown in Table 4.1.

From Table 4.1, notice how the two-dimensional modulation schemes perform better than the one-dimensional modulation schemes. Furthermore, notice how all of the modulation schemes studied are linear modulation schemes, which means they possess a similar level of receiver complexity. Given these insights on the power efficiency performance of these modulation schemes, we now turn our attention to the robustness of a modulation technique in the presence of noise.

Table 4.1 δSNR Values of Various Modulation Schemes

M	b	*M-ASK*	*M-PSK*	*M-QAM*
2	1	0	0	0
4	2	4	0	0
8	3	8.45	3.5	—
16	4	13.27	8.17	4.0
32	5	18.34	13.41	—
64	6	24.4	18.4	8.45

Hands-On MATLAB Example: Noise introduced by a transmission medium can potentially result in symbols being decoded in error. In the following MATLAB script, we will examine the behavior of how the introduction of noise can obfuscate the true identity of an intercepted symbol. Specifically, we will compare the originally transmitted symbols and the noisy received symbols using a visual representation referred to as a signal constellation diagram, which plots the locations of symbols across a 2-axis plot with an in-phase axis and a quadrature axis. Notice that we are considering three different waveforms in this example: 4-PAM, 4-QAM, and QPSK. For each of these waveforms, we generated an alphabet of different symbols that each can produce. The randomly generated binary data streams representing in-phase and quadrature information are mapped to these different waveform symbols for each modulation scheme. Then, we introduce Gaussian noise to the three transmissions using the `randn` function.

The before-and-after signal constellation plots for the 4-PAM, 4-QAM, and QPSK modulated transmissions are shown in Figure 4.14. The original symbols are

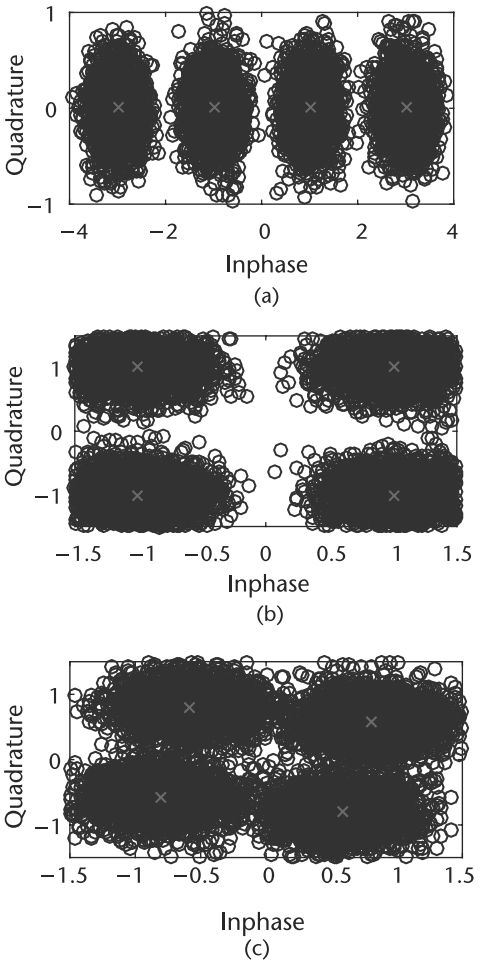

Figure 4.14 Examples of four-level pulse amplitude modulation, quadrature amplitude modulation, and quadrature phase shift keying waveforms. (a) Four-level pulse amplitude modulation, (b) four-level quadrature amplitude modulation, and (c) four-level quadrature phase shift keying.

Code 4.5 Generating Four Level Pulse Amplitude Modulation, Quadrature Amplitude Modulation, and Quadrature Phase Shift Keying Waveforms: **chapter4.m**

```
% Define parameters
len = 10000; % Length of binary string
nvar = 0.15; % Noise variance

% Generate the random bit streams that have already been demultiplexed
% from a single high speed data stream
bin_str1 = round(rand(1,len)); % Inphase data stream
bin_str2 = round(rand(1,len)); % Quadrature data stream

% Perform mapping of binary streams to symbols
ind_wavefm = 2.*bin_str2 + 1.*bin_str1; % Create waveform indices
wavefm_4pam = zeros(1,len); % 4-PAM
wavefm_4qam = zeros(1,len); % 4-QAM
wavefm_qpsk = zeros(1,len); % QPSK
symb_4pam = [-3 -1 3 1];
symb_4qam = [-1+i 1+i -1-i 1-i];
symb_qpsk = [exp(i*(pi/5+pi/2)) exp(i*(pi/5+pi)) exp(i*(pi/5+0))
        exp(i*(pi/5+3*pi/2)) ];
for ind = 1:1:4,
    wavefm_4pam(find(ind_wavefm == (ind-1))) = symb_4pam(ind);
    wavefm_4qam(find(ind_wavefm == (ind-1))) = symb_4qam(ind);
    wavefm_qpsk(find(ind_wavefm == (ind-1))) = symb_qpsk(ind);
end;

% Add complex zero-mean white Gaussian noise
noise_signal = (1/sqrt(2))*sqrt(nvar)*randn(1,len)
        + i*(1/sqrt(2))*sqrt(nvar)*randn(1,len);
rx_wavefm_4pam = wavefm_4pam + noise_signal;
rx_wavefm_4qam = wavefm_4qam + noise_signal;
rx_wavefm_qpsk = wavefm_qpsk + noise_signal;
```

shown as little red cross marks in the center of a cloud of corrupted received symbols after the noise is added to them. This is occurring since whenever a transmission is occurring over a noisy channel, the symbols that are sent over this channel are being displaced from their original coordinates in the in-phase/quadrature plane by the complex Gaussian noise. This displacement is what makes it difficult for the receiver to decode these symbols without any error since the noise might sufficiently displace these symbols closer to another nearby symbol location that is part of the signal constellation. For all of these modulation schemes shown in Figure 4.14(a) (4-PAM), Figure 4.14(b) (4-QAM), and Figure 4.14(c) (QPSK), there is a nonneglible probability that these symbols have been moved closer to another point in the overall signal constellation, which would result in an error in decode that would translate into a bit error.

4.3 Probability of Bit Error

One of the most commonly used quantitative metrics for measuring the performance of a digital communication system is the probability of BER, which is the probability that a bit transmitted will be decoded incorrectly. This metric is very important when assessing whether the design of a digital communication system meets the

specific error robustness requirements of the application to be supported (e.g., voice, multimedia, or data). Furthermore, having a metric that quantifies error performance is helpful when comparing one digital communication design with another. Consequently, in this section we will provide a mathematical introduction to the concept of BER.

Suppose that a signal $s_i(t)$, $i = 1, 2$ was transmitted across an AWGN channel with noise signal $n(t)$, and that a receiver intercepts the signal $r(t)$. The objective of the receiver is to determine whether either $s_1(t)$ or $s_2(t)$ was sent by the transmitter. Given that the transmission of either $s_1(t)$ or $s_2(t)$ is a purely random event, the only information that the receiver has about what was sent by the transmitter is the observed intercepted signal $r(t)$, which contains either signal in addition to some noise introduced by the AWGN channel.

Given this situation, we employ the concept of *hypothesis testing* [2] in order to set up a framework by which the receiver can decide on whether $s_1(t)$ or $s_2(t)$ was sent based on the observation of the intercepted signal $r(t)$. Thus, let us employ the following hypothesis testing framework:

$$\mathcal{H}_1 : r(t) = s_1(t) + n(t),\ 0 \leq t \leq T$$
$$\mathcal{H}_0 : r(t) = s_2(t) + n(t),\ 0 \leq t \leq T$$

where $\mathcal{H}_0$ and $\mathcal{H}_1$ are *Hypothesis 0* and *Hypothesis 1*.

Leveraging this framework, we next want to establish a *decision rule* at the receiver such that it can select which waveform was sent based on the intercept signal. Suppose we assume that $s_1(t)$ was transmitted. In general, we can determine the level of correlation between two signals $x(t)$ and $y(t)$ over the time interval $0 \leq t \leq T$ using the expression

$$\int_0^T x(t)y(t)dt.$$

Consequently, our decision rule on whether $s_1(t)$ or $s_2(t)$ was transmitted given that we observe $r(t)$ is defined as

$$\int_0^T r(t)s_1(t)dt \geq \int_0^T r(t)s_2(t)dt, \tag{4.44}$$

where we assume that $s_1(t)$ was transmitted. Recall that correlation tells us how similar one waveform is to another waveform. Therefore, if the receiver knows the appearance of $s_1(t)$ and $s_2(t)$, we can then determine which of these two waveforms is more correlated to $r(t)$. Since $s_1(t)$ was assumed to be transmitted, ideally the received signal $r(t)$ should be more correlated to $s_1(t)$ than $s_2(t)$.

On the other hand, what happens if some distortion, interference, and/or noise is introduced in the transmission channel such that the transmitted signal waveforms are corrupted? In the situation where a transmitted signal waveform is sufficiently corrupted such that it appears to be more correlated to another possible signal waveform, the receiver could potentially select an incorrect waveform, thus yielding an error event. In other words, assuming $s_1(t)$ was transmitted, an error event occurs

when

$$\int_0^T r(t)s_1(t)dt \leq \int_0^T r(t)s_2(t)dt. \tag{4.45}$$

Since $r(t) = s_1(t) + n(t)$, we can substitute this into the error event in order to obtain the decision rule

$$\int_0^T s_1^2(t)dt + \int_0^T n(t)s_1(t)dt \leq \int_0^T s_1(t)s_2(t)dt + \int_0^T n(t)s_2(t)dt$$

$$E_{s_1} - \rho_{12} \leq \int_0^T n(t)(s_2(t) - s_1(t))dt$$

$$E_{s_1} - \rho_{12} \leq z.$$

From this expression, we observe that both E_{s_1} and ρ_{12} are deterministic quantities. On the other hand, z is based on the noise introduced by the transmission channel, and thus it is a random quantity that requires some characterization. Since $n(t)$ is a Gaussian random variable, then z is also a Gaussian random variable. This is due to the fact that the process of integration is equivalent to a summation across an infinite number of samples, and since we are summing up Gaussian random variables, the result in is also a Gaussian random variable. With $z \sim \mathcal{N}(0, \sigma^2)$, we now need to calculate the variance of z, σ^2, which can be solved as follows:

$$\begin{aligned}\sigma^2 = E\{z^2\} &= \frac{N_0}{2}\int_0^T (s_1(t) - s_2(t))^2 dt \\ &= \frac{N_0}{2}(E_{s_1 f:ch4_4 mod s_q psk} + E_{s_2} - 2\rho_{12}) \rightarrow \text{Assume } E_{s_1} = E_{s_2} = E \\ &= N_0(E - \rho_{12}),\end{aligned}$$

where $E = E_i = \int_0^T s_i^2(t)dt$ and $\rho_{12} = \int_0^T s_1(t)s_2(t)dt$. Note that we are assuming that the channel is introducing zero-mean noise, which means the sum of these noise contributions; that is, z will also be zero-mean.

With both deterministic and random quantities characterized, we can now proceed with the derivation for the probability of bit error. The probability of an error occurring given that a "1" was transmitted; that is, $P(e|1)$ is equal to

$$\begin{aligned}P(z \geq E - \rho_{12}) &= Q\left(\frac{E - \rho_{12}}{\sigma}\right) \rightarrow \begin{array}{l}\text{Since } z \sim \mathcal{N}(0, \sigma^2) \\ \text{and } E - \rho_{12} \text{ is constant}\end{array} \\ &= Q\left(\sqrt{\frac{(E - \rho_{12})^2}{\sigma^2}}\right) \rightarrow \text{Use } \sigma^2 = N_0(E - \rho_{12}) \\ &= Q\left(\sqrt{\frac{E - \rho_{12}}{N_0}}\right),\end{aligned}$$

where the Q-function is defined as

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_{x}^{\infty} e^{-t^2/2} dt. \tag{4.46}$$

The next step is to optimize the probability of bit error by optimizing the probability of error by minimizing $P(e|1)$, which can be achieved by optimizing the correlation term ρ_{12}. Intuitively, the best choice we can make is when $s_2(t) = -s_1(t)$, which gives us $\rho_{12} = -E$. Consequently, this result yields

$$P(e|1) = Q\left(\sqrt{\frac{2\bar{E}_b}{N_0}}\right). \tag{4.47}$$

Note that when $E_{s_1} \neq E_{s_2}$, we can then use $d^2_{\min} = E_{s_1} + E_{s_2} - 2\rho_{12}$, which yields the following expression for the probability of bit error:

$$P_e = Q\left(\sqrt{\frac{d^2_{\min}}{2N_0}}\right). \tag{4.48}$$

Show that the total probability of bit error is equal to:

$$P_e = P(e|1)P(1) + P(e|0)P(0) = Q\left(\sqrt{\frac{E - \rho_{12}}{N_0}}\right) \tag{4.49}$$

When dealing with a large number of signal waveforms that form a modulation scheme, the resulting probability of error, P_e, is expressed as a sum of pairwise error probabilities; that is, the probability of one received symbol being another specific received symbol. The pairwise error probability of $s_i(t)$ being decoded when $s_j(t)$ was transmitted is given as

$$Q\left(\frac{d^2_{ij}}{2N_0}\right), \tag{4.50}$$

where N_0 is the variance of the noise. An important note here is that we are assuming the noise is AWGN, since Q functions apply specifically to Gaussian random variables. Therefore, the complete expression for P_e can be expressed as

$$Q\left(\frac{d^2_{\min}}{2N_0}\right) \leq P_e \leq Q\left(\frac{d^2_{1j}}{2N_0}\right) + \ldots + Q\left(\frac{d^2_{Mj}}{2N_0}\right), \quad i \neq j, \tag{4.51}$$

where the second half of the relationship is the summation of every single pairwise error probability.

Hands-On MATLAB Example: We have previously observed the impact of a noisy channel on a received signal constellation, where the signal constellation

points corresponding to specific symbols of the modulation scheme are potentially shifted to other nearby signal constellation points and risk being incorrectly decoded at the receiver. In the following MATLAB script, we will translate these shifts of signal constellation points into actual BER values in order to quantify the actual severity of the noise being introduced by the channel. The code below employs a nearest neighbor approach for decoding symbols at the receiver, where a received signal constellation point that has been corrupted by noise is decoded by mapping it to the nearest originally transmitted signal constellation point. Using the Euclidean distance, we can calculate the distance between a received signal constellation point with all possible signal constellation point options, map it to the one with the shortest Euclidean distance, and then decode the symbol into the corresponding binary word.

Using this MATLAB script and the Euclidean distance approach for deciding on the nearest neighbors, we can decode the received messages sent across the noisy channel. However, since there are instances where the noise is significant enough that it can move a symbol much closer to another signal constellation point, we should anticipate that there might be several symbols that have been incorrectly decoded. Figure 4.15 presents the BER results for our nearest neighbor decoding scheme for 4-PAM, 4-QAM, and QPSK modulation. Although the first two modulation schemes do not possess a substantial amount of error, the QPSK modulation scheme possesses a large amount of incorrect decisions. This is due to the fact of how the signal constellation points are spaced out, with the QPSK signal constellation points being closer together relative to the other two modulation schemes. As a result, for the same amount of noise, the QPSK modulation will perform noticeably worse compared to the other two schemes.

4.3.1 Error Bounding

Computing each pairwise error probability is not always practical. It is possible to create an upper and lower bound on P_e by computing only the pairwise errors of points that are within one degree of the point of interest. Consider the behavior of the Q function $Q(.)$. As the input to $Q(.)$ increases, the resulting output of the Q function approaches zero. You will find that computing the pairwise error probability of points farther away yields negligible contributions to the total P_e, but can save a significant amount of time as well as cycles. Thus, an accurate estimate of $P(e)$ can be computed from the following bounds.

These upper and lower bounds can be expressed as

$$Q\left(\frac{d_{min}^2}{2N_0}\right) \leq P(e) \leq \sum_{i \in I} Q\left(\frac{d_{ij}^2}{2N_0}\right), \tag{4.52}$$

where I is the set of all signal waveforms within the signal constellation that are immediately adjacent to the signal waveform j. In order to accurately assess the performance of a communications system, it must be simulated until a certain number of symbol errors are confirmed [3]. In most cases, 100 errors will give a 95% confidence interval, which should be employed later on in this book in order to characterize the bit error rate of any digital communication system under evaluation.

Code 4.6 Decode Messages from Previous Example Using Euclidean Distance: **chapter4.m**

```
% Go through every received waveform and determine Euclidean distance
% between received waveform and the available waveforms
eucl_dist_4pam = zeros(4,len);
eucl_dist_4qam = zeros(4,len);
eucl_dist_qpsk = zeros(4,len);
for ind = 1:1:4,
    eucl_dist_4pam(ind,1:1:len) = abs(symb_4pam(ind).*ones(1,len)
                                    - rx_wavefm_4pam);
    eucl_dist_4qam(ind,1:1:len) = abs(symb_4qam(ind).*ones(1,len)
                                    - rx_wavefm_4qam);
    eucl_dist_qpsk(ind,1:1:len) = abs(symb_qpsk(ind).*ones(1,len)
                                    - rx_wavefm_qpsk);
end;

% Select shortest Euclidean distances
[mdist_4pam,min_ind_4pam] = min(eucl_dist_4pam);
[mdist_4qam,min_ind_4qam] = min(eucl_dist_4qam);
[mdist_qpsk,min_ind_qpsk] = min(eucl_dist_qpsk);

% Decode into estimated binary streams
bin_str_est_4pam = dec2bin(min_ind_4pam-ones(1,len)).';
bin_str_est_4qam = dec2bin(min_ind_4qam-ones(1,len)).';
bin_str_est_qpsk = dec2bin(min_ind_qpsk-ones(1,len)).';

% Calculate bit error rate
ber_4pam = sum([abs((bin_str_est_4pam(1,:)-'0') - bin_str2) ...
    abs((bin_str_est_4pam(2,:)-'0') - bin_str1)])/(2*len);
ber_4qam = sum([abs((bin_str_est_4qam(1,:)-'0') - bin_str2) ...
    abs((bin_str_est_4qam(2,:)-'0') - bin_str1)])/(2*len);
ber_qpsk = sum([abs((bin_str_est_qpsk(1,:)-'0') - bin_str2) ...
    abs((bin_str_est_qpsk(2,:)-'0') - bin_str1)])/(2*len);
```

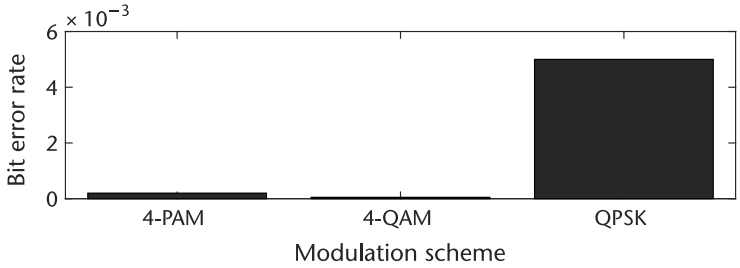

Figure 4.15 Impact of noise on modulation scheme performance.

Hands-On MATLAB Example: We have previosly observed the performance of three simple communication systems using different modulation schemes operating in a noisy environment. Although the results displayed in Figure 4.15 were insightful, we often want to observe how a communication system performs across a broad range of conditions, especially as the intensity of noise varies. In this MATLAB script, we examine the performance of a simple binary communication system across a range of different channel environments possessing varying degrees of noise.

Code 4.7 Create Waterfall Curves for the Bit Error Rate of a Communication System via Monte Carlo: **`chapter4.m`**

```
% Define parameters
len = 1000; % Length of individual data transmission
N_snr = 9; % Number of SNR values to evaluation
N_tx = 100; % Number of transmissions per SNR
nvar = [(10.^((1:1:N_snr)/10)).^(-1)]; % Noise variance values

ber_data = zeros(N_snr,N_tx);
for ind = 1:1:N_snr, % Different SNR values
  for ind1 = 1:1:N_tx, % Different transmissions for same SNR value

    % Generate BPSK waveform (we will keep this the same for each
    % SNR value for now)
    tx_sig = 2*round(rand(1,len))-1;

    % Create additive noise
    noise_sig = sqrt(nvar(ind))*randn(1,len);

    % Create received (noisy) signal
    rx_sig = tx_sig + noise_sig;

    % Decode received signal back to binary
    decode_bin_str = zeros(1,len);
    decode_bin_str(find(rx_sig >= 0)) = 1;

    % Determine and store bit error rate
    ber_data(ind,ind1) = sum(abs(decode_bin_str - (tx_sig+1)/2))/len;
  end;
end;

% Calculate mean bit error rate and its standard deviation
mean_ber = mean(ber_data,2).';
std_ber = std(ber_data,'',2).';
```

The end result of this analysis, which explores transmission reliability when operating across an AWGN channel, is something referred to as a waterfall curve, as shown in Figure 4.16. Waterfall curves are extensively used by researchers and designers in the digital communications community as a way of characterizing the error robustness of a communication system operating in a noisy environment. The reason why we call these plots waterfall curves is due to the shape they make whenever we generate them using either theoretical analyses or via experimentation (computer simulation or hardware testing). The x-axis describes the SNR of the operating environment and is a gauge of how much noise is present in the channel. The y-axis describes the probability of bit error as a ratio of corrupted bits versus total number of bits transmitted. In Figure 4.16, we not only show the mean BER curve but also the standard deviation above and below the mean in order to establish the degree of confidence we have with respect to the technique we used to generate these curves. Since we are using Monte Carlo techniques for generating the transmission and then corrupting the bits with additive noise, we need to make sure that we conduct this experiment long enough such that the performance results we obtain are reliable (e.g., running an experiment and only obtaining one bit error

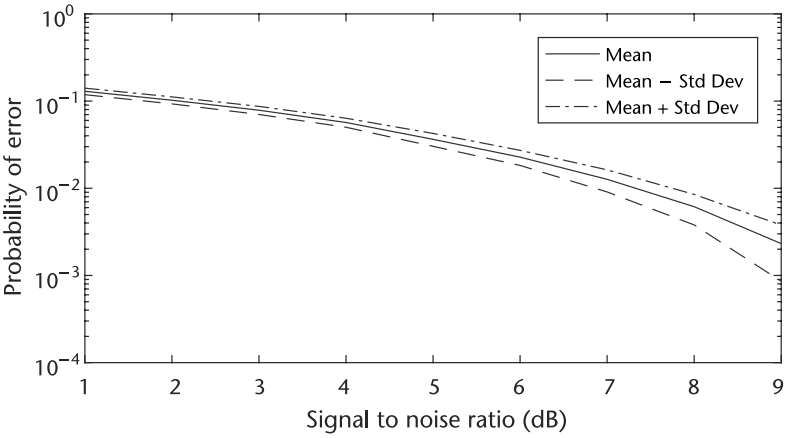

Figure 4.16 Example of waterfall curves for the bit error rate of a communication system employing binary phase shift keying via Monte Carlo simulation techniques.

is not statistically adequate with respect to an accurate assessment of the BER for that communication system). Thus, having the standard deviation curves close to the mean BER curve shows that an adequate number of bits have been used in the experiment, and that a sufficient number of errors have been obtained. Note that for different SNR values, the amount of errors obtained might be different for the same total number of bits transmitted. Consequently, we often have to transmit more bits at higher SNR values in order to obtain an adequate number of bit errors.

4.4 Signal Space Concept

Until this point we have studied digital communication systems from a signal waveform perspective. Leveraging this perspective, we have developed mathematical tools for analyzing the power efficiency and BER of different modulation schemes. However, there are several instances where the use of a signal waveform framework can be tedious or somewhat cumbersome. In this section, we will introduce another perspective on how to characterize and analyze modulation scheme using a different mathematics representation: signal vectors.

Suppose we define $\phi_j(t)$ as an orthonormal set of functions over the time interval $[0, T]$ such that

$$\int_0^T \phi_i(t)\phi_j(t)dt = \begin{cases} 1 & i = j \\ 0 & \text{otherwise} \end{cases}$$

Given that $s_i(t)$ is the ith signal waveform, we would like to represent this waveform as a sum of several orthonormal functions; that is,

$$s_i(t) = \sum_{k=1}^{N} s_{ik}\phi_k(t), \tag{4.53}$$

which can be equivalently represented by the vector

$$\mathbf{s}_i = (s_{i1},\ s_{i2},\ s_{i3}, \ldots s_{iN}), \tag{4.54}$$

where the elements of the vector $\mathbf{s}_i$ define the amplitude scaling of each orthonormal function used to represent the waveform. An illustration of a signal waveform represented by three orthonormal functions is shown in Figure 4.17. Consequently, given this relationship between the signal waveform and the orthonormal functions, where the former can be represented as the weighted sum of the latter, we can readily describe the signal waveform as a simple vector, which we will see next possesses the advantage of enabling us to employ relatively straightforward mathematical operations based on linear algebra.

In order to find the vector elements, s_{il}, we need to solve the expression

$$\int_0^T s_i(t)\phi_l(t)dt = \sum_{k=1}^{N} s_{ik} \int_0^T \phi_k(t)\phi_l(t)dt = s_{il}, \tag{4.55}$$

which is essentially a *dot product* or *projection* of the signal waveform $s_i(t)$ on the orthonormal function $\phi_l(t)$. At the same time, if we perform the vector dot product between the signal waveforms $s_i(t)$ and $s_j(t)$, we get a correlation operation that is equal to

$$\int_0^T s_i(t)s_j(t)dt = \mathbf{s}_i \cdot \mathbf{s}_j = \rho_{ij}, \tag{4.56}$$

while the energy of a signal $s_i(t)$ is equal to

$$E_{s_i} = \int_0^T s_i^2(t)dt = \mathbf{s}_i \cdot \mathbf{s}_i = ||\mathbf{s}_i||^2. \tag{4.57}$$

All of these mathematical operations will be employed when determining the power efficiency of a modulation scheme or deriving the optimal decision rule for a receiver.

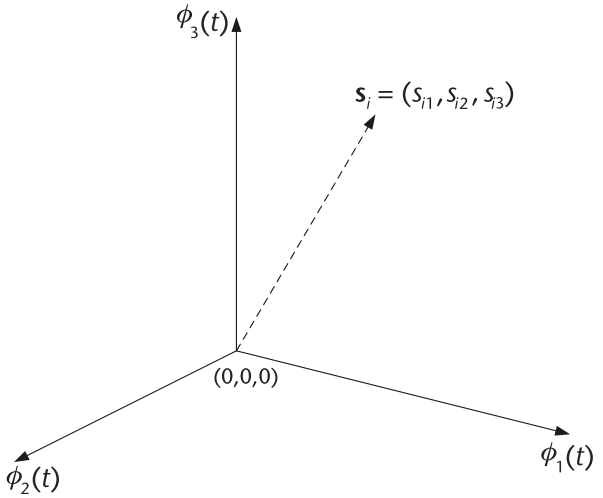

Figure 4.17 Sample vector representation of $s_i(t)$ in three-dimensional space using basis functions $\phi_1(t)$, $\phi_2(t)$, and $\phi_3(t)$.

Suppose we would like to compute the power efficiency for a modulation scheme using a signal vector approach rather than a signal waveform approach. The first step would be to calculate the minimum Euclidean distance, which can be solved using the following:

$$\begin{aligned} d_{\min}^2 &= \int_0^T \Delta s_{ij}^2(t)dt = \int_0^T (s_i(t) - s_j(t))^2 dt \\ &= ||\mathbf{s}_i - \mathbf{s}_j||^2 = (\mathbf{s}_i - \mathbf{s}_j) \cdot (\mathbf{s}_i - \mathbf{s}_j) \\ &= E_{s_i} + E_{s_j} - 2\rho_{ij} \end{aligned}$$

where the correlation term between signal waveforms $s_i(t)$ and $s_j(t)$ is given by

$$\rho_{ij} = \int_0^T s_i(t)s_j(t)dt = \mathbf{s}_i \cdot \mathbf{s}_j. \tag{4.58}$$

In order to solve for the power efficiency, we choose a set of orthonormal basis functions $\phi_i(t)$, $i = 1, 2, \ldots, k$, where k is the dimension of the signal vector space. Given this set of functions, we can now represent the vector $\mathbf{s}_i$, $i = 1, 2, \ldots, M$ where $\mathbf{s}_i = (s_{i1}, s_{i2}, \ldots s_{ik})$ and

$$s_{ij} = \int_0^T s_i(t)\phi_j(t)dt. \tag{4.59}$$

Consequently, using the vector representations for the signals and the orthonormal functions, we can calculate the minimum Euclidean distance:

$$d_{\min}^2 = \min_{i \neq j} ||\mathbf{s}_i - \mathbf{s}_j||^2, \tag{4.60}$$

the average symbol and bit energy values:

$$\begin{aligned} \bar{E}_s &= \frac{1}{M}\sum_{i=1}^{M} ||\mathbf{s}_i||^2 \\ \bar{E}_b &= \bar{E}_s / \log_2(M), \end{aligned} \tag{4.61}$$

and the power efficiency:

$$\varepsilon_p = d_{\min}^2 / \bar{E}_b. \tag{4.62}$$

4.5 Gram-Schmidt Orthogonalization

In mathematics, particularly linear algebra and numerical analysis, the Gram-Schmidt orthogonalization process is a method for creating an orthonormal set of functions in an inner product space such as the Euclidean space $\mathbb{R}^n$. The Gram-Schmidt orthogonalization process takes a finite set of signal waveforms $\{s_1(t), \ldots, s_M(t)\}$ and generates from it an orthogonal set of functions

$\{\phi_1(t), \ldots, \phi_i(t)\}$ that spans the space $\mathbb{R}^n$. Note that an orthonormal function possesses the following property:

$$\int_0^T \phi_i(t)\phi_j(t)dt = \begin{cases} 1 & i = j \\ 0 & \text{otherwise} \end{cases}.$$

Furthermore, it is possible to represent a signal waveform $s_i(t)$ as the weighted sum of these orthonormal basis functions; that is,

$$s_i(t) = \sum_{k=1}^{N} s_{ik}\phi_k(t). \tag{4.63}$$

However, what we need now is an approach to generate the set of orthonormal basis functions $\{\phi_j(t)\}$.

To derive a set of orthogonal basis functions $\{\phi_1(t), \ldots, \phi_i(t)\}$ from a set of signal waveforms denoted by $\{s_1(t), \ldots, s_M(t)\}$, let us first start with $s_1(t)$ and normalize it:

$$\phi_1(t) = \frac{s_1(t)}{\sqrt{E_{s_1}}}$$

where E_{s_1} is the energy of the signal $s_1(t)$. This normalized version of the signal waveform $s_1(t)$ will serve as our first orthonormal basis function from which we will construct the rest of our orthonormal basis function set. In other words, we are effectively bootstrapping a set of orthonormal basis functions based on the existing signal waveforms of the modulation scheme to be used. Note that we can represent $s_1(t)$ as

$$s_1(t) = \sqrt{E_{s_1}}\phi_1(t) = s_{11}\phi_1(t)$$

where the coefficient $s_{11} = \sqrt{E_{s_1}}$ and the orthonormal function $\phi_1(t)$ satisfy the unit energy constraint as required.

Next, we would like to create the second orthonormal function, $\phi_2(t)$. In order to accomplish this task, we use the signal waveform $s_2(t)$ as a starting point. However, $s_2(t)$ may contain elements of $\phi_1(t)$, and thus we need to remove this component before normalizing it in order to transform this waveform into $\phi_2(t)$. To achieve this, we first determine how much of $\phi_1(t)$ is contained within $s_2(t)$ by taking the dot product between these two functions and determining how much $s_2(t)$ projects onto $\phi_1(t)$; that is,

$$s_{21} = \int_0^T s_2(t)\phi_1(t)dt.$$

To help in getting the basis function $\phi_2(t)$, we define the intermediate function:

$$g_2(t) = s_2(t) - s_{21}\phi_1(t),$$

which is orthogonal to $\phi_1(t)$ over the interval $0 \leq t \leq T$ by virtual of the fact that we have removed the $\phi_1(t)$ component from $s_2(t)$. Finally, normalizing $g_2(t)$ yields

the basis function $\phi_2(t)$:

$$\phi_2(t) = \frac{g_2(t)}{\sqrt{\int_0^T g_2^2(t)dt}} \tag{4.64}$$

which can be expanded to

$$\phi_2(t) = \frac{s_2(t) - s_{21}\phi_1(t)}{\sqrt{E_{s_2} - s_{21}^2}} \tag{4.65}$$

where E_{s_2} is the energy of the signal $s_2(t)$. A quick sanity check clearly shows that the orthonormal basis function $\phi_2(t)$ satisfies the constraint

$$\int_0^T \phi_2^2(t)dt = 1 \quad \text{and} \quad \int_0^T \phi_1(t)\phi_2(t)dt = 0$$

In general, we can define the following functions that can be employed in an iterative procedure for generating a set of orthonormal basis functions:

$$\begin{aligned} g_i(t) &= s_i(t) - \sum_{j=1}^{i-1} s_{ij}\phi_j(t) \\ s_{ij} &= \int_0^T s_i(t)\phi_j(t)dt, \; j = 1, 2, \ldots, i-1 \\ \phi_i(t) &= \frac{g_i(t)}{\sqrt{\int_0^T g_i^2(t)dt}}, \; i = 1, 2, \ldots, N. \end{aligned} \tag{4.66}$$

We will now work out an example that deals with the Gram-Schmidt orthogonalization process.

An Example: Suppose we want to perform the Gram-Schmidt orthogonalization procedure of the signals shown in Figure 4.18 in the order $s_3(t)$, $s_1(t)$, $s_4(t)$, $s_2(t)$ and obtain a set of orthonormal functions $\{\phi_m(t)\}$. Note that the order in which the signal waveforms are employed to generate the orthonormal basis functions is very important, since each ordering of signal waveforms can yield a potentially different set of orthonormal basis functions.

Starting with $s_3(t)$, we get

$$\phi_1(t) = \frac{s_3(t)}{\sqrt{E_{s_3}}} = \frac{s_3(t)}{\sqrt{3}}. \tag{4.67}$$

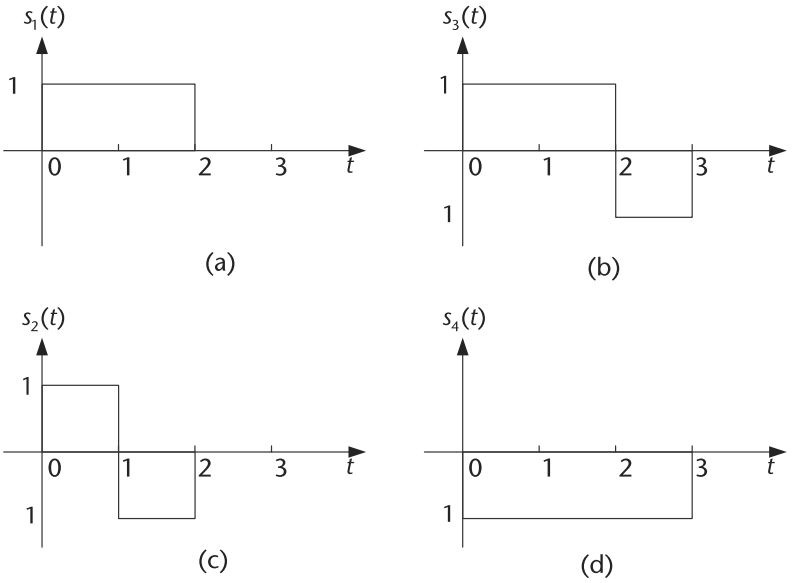

Figure 4.18 Example signal waveforms.

Then, leveraging the result for $\phi_1(t)$, we then derive the orthonormal basis function $\phi_2(t)$ using $s_1(t)$:

$$g_2(t) = s_1(t) - s_{12}\phi_1(t) = s_1(t) - \frac{2}{3}s_3(t) = \begin{cases} 1/3, & 0 \le t < 2 \\ 2/3, & 2 \le t < 3 \\ 0, & t \ge 3 \end{cases}$$
$$\therefore \phi_2(t) = \frac{g_2(t)}{\sqrt{\int_0^T g_2^2(t)dt}} = \begin{cases} 1/\sqrt{6}, & 0 \le t < 2 \\ 2/\sqrt{6}, & 2 \le t < 3 \\ 0, & t \ge 3 \end{cases}. \tag{4.68}$$

We subsequently repeat this operation for $s_4(t)$:

$$g_3(t) = s_4(t) - \sum_{j=1}^{2} s_{4j}\phi_j(t) = 0$$
$$\therefore \phi_3(t) = 0, \tag{4.69}$$

but we notice the resulting $\phi_3(t)$ is equal to zero. This implies that the signal waveform $s_4(t)$ can be entirely characterized by only $\phi_1(t)$ and $\phi_2(t)$. Finally, for $s_2(t)$, we get the following:

$$g_4(t) = s_2(t) - \sum_{j=1}^{3} s_{2j}\phi_j(t) = 0$$
$$\therefore \phi_4(t) = \frac{g_4(t)}{\sqrt{\int_0^T g_4^2(t)dt}} = \frac{s_2(t)}{\sqrt{2}}. \tag{4.70}$$

Consequently, with the orthonormal basis functions $\{\phi_1(t), \phi_2(t), \phi_4(t)\}$ defined, we can now express the four signal waveforms as

- $\mathbf{s}_1 = (2/\sqrt{3}, \sqrt{6}/3, 0)$,
- $\mathbf{s}_2 = (0, 0, \sqrt{2})$,
- $\mathbf{s}_3 = (\sqrt{3}, 0, 0)$,
- $\mathbf{s}_4 = (-1/\sqrt{3}, -4/\sqrt{6}, 0)$.

Now let us implement these waveforms via the orthonormal basis functions using the following MATLAB script. In this script, we assume that there are `N_samp` sampling instants per unit time. Consequently, since each waveform is of a duration of 3 seconds, each waveform representation in this MATLAB model is `3*N_samp` long.

Code 4.8 Gram-Schmidt Orthogonalization and Vectorization: **`chapter4.m`**

```
% Define parameters
N_samp = 1000; % Number of samples per time unit

% Create orthonormal basis functions
phi1 = [( 1/sqrt(3))*ones(1,N_samp) ...
    ( 1/sqrt(3))*ones(1,N_samp) ...
    (-1/sqrt(3))*ones(1,N_samp)];
phi2 = [( 1/sqrt(6))*ones(1,N_samp) ...
    ( 1/sqrt(6))*ones(1,N_samp) ...
    ( 2/sqrt(6))*ones(1,N_samp)];
phi3 = [0*ones(1,N_samp) 0*ones(1,N_samp) 0*ones(1,N_samp)];
phi4 = [( 1/sqrt(2))*ones(1,N_samp) ...
    (-1/sqrt(2))*ones(1,N_samp) ...
      0*ones(1,N_samp)];

% Based on these orthonormal basis functions, create the four symbol
% waveforms
sig_s1 = (2/sqrt(3))*phi1 + (sqrt(6)/3)*phi2 + 0*phi3 + 0*phi4;
sig_s2 = 0*phi1 + 0*phi2 + 0*phi3 + sqrt(2)*phi4;
sig_s3 = (sqrt(3))*phi1 + 0*phi2 + 0*phi3 + 0*phi4;
sig_s4 = (-1/sqrt(3))*phi1 + (-4/sqrt(6))*phi2 + 0*phi3 + 0*phi4;
```

Using these orthonormal basis functions, and the results of the Gram-Schmidt orthogonalization process, we are able to produce the same waveforms shown in Figure 4.18 using this MATLAB script, as shown in Figure 4.19.

4.6 Optimal Detection

Detection theory, or signal detection theory, is used in order to discern between signal and noise [2]. Using this theory, we can explain how changing the decision threshold will affect the ability to discern between two or more scenarios, often exposing how adapted the system is to the task, purpose, or goal at which it is aimed.

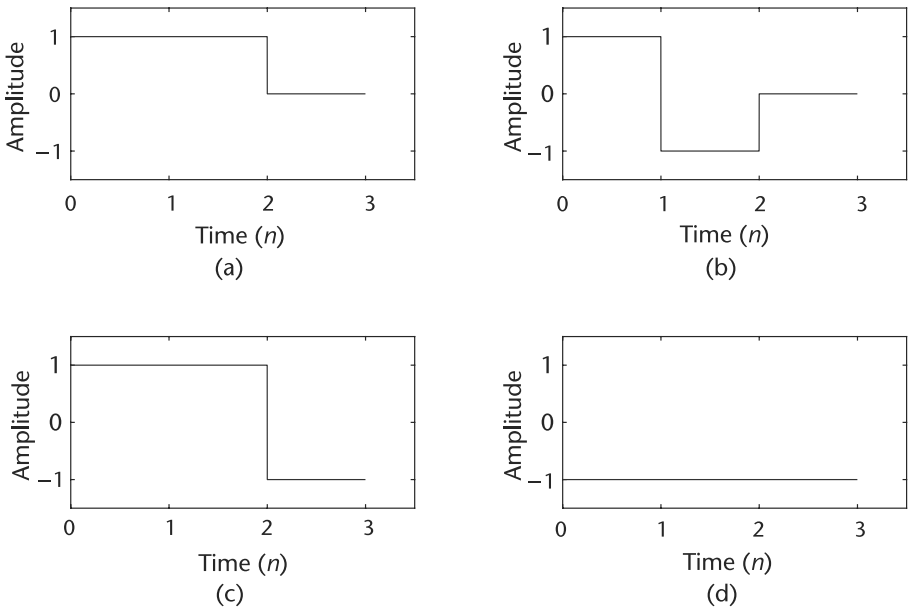

Figure 4.19 Creation of the waveforms (a) $s_1(n)$, (b) $s_2(n)$, (c) $s_3(n)$, and (d) $s_4(n)$ from a collection of orthonormal basis functions.

4.6.1 Signal Vector Framework

Let us assume a simple digital transceiver model as shown in Figure 4.20. As mentioned previously, the receiver only observes the corrupted version of $s_i(t)$ by the noise signal $n(t)$; namely, $r(t)$. The noise signal $n(t)$ usually represents the culmination of all noise sources into a single variable. Therefore, our detection problem in this situation can be summarized as follows: Given $r(t)$ for $0 \leq t \leq T$, determine which $s_i(t)$, $i = 1, 2, \ldots, M$, is present in the intercepted signal $r(t)$.`s1(n)`.

Suppose we decompose the waveforms $s_i(t)$, $n(t)$, and $r(t)$ into a collection of weights applied to a set of orthonormal basis functions; namely,

$$s_i(t) = \sum_{k=1}^{N} s_{ik}\phi_k(t), \quad r(t) = \sum_{k=1}^{N} r_k\phi_k(t), \quad n(t) = \sum_{k=1}^{N} n_k\phi_k(t).$$

Given that all of these signal waveforreliablems use the same orthonormal basis functions, we can rewrite the waveform model expression $r(t) = s_i(t) + n(t)$ into

$$\sum_{k=1}^{N} r_k\phi_k(t) = \sum_{k=1}^{N} s_{ik}\phi_k(t) + \sum_{k=1}^{N} n_k\phi_k(t)$$

$$\mathbf{r} = \mathbf{s}_i + \mathbf{n}.$$

Since $\mathbf{r}$ consists of a combination of the deterministic waveform $\mathbf{s}_i$ and probabilistic signal $\mathbf{n}$, our attention now turns to mathematically characterizing $\mathbf{n}$. Since the noise signal $n(t)$ is assumed to be a Gaussian random variable, we need to determine how the characteristics of this random variable translates into a signal

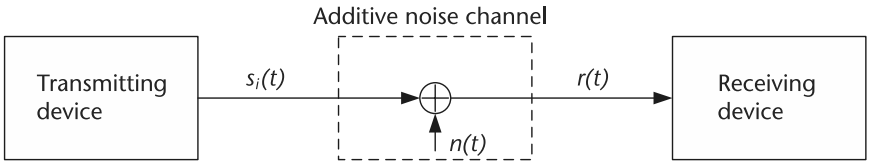

Figure 4.20 Simple digital transceiver model.

vector representation. We know that the noise vector element n_k is equal to

$$n_k = \int_0^T n(t)\phi_k(t)dt, \tag{4.71}$$

which is the projection of the noise signal waveform on the orthonormal basis function $\phi_k(t)$. Since the noise signal $n(t)$ is a Gaussian random variable and the integration process is a linear operation, this means that n_k is a Gaussian random variable as well. Thus, the noise signal vector $\mathbf{n}$ is a Gaussian vector. Let us now proceed with determining the statistical characteristics of $\mathbf{n}$ in order to employ this knowledge in signal waveform detection.

First, we would like to calculate the mean of these vector elements. Thus, by applying the definition for the expectation, this yields

$$\begin{aligned} E\{n_k\} &= E\left\{\int_0^T n(t)\phi_k(t)dt\right\} \\ &= \int_0^T E\{n(t)\}\phi_k(t)dt \\ &= 0 \end{aligned} \tag{4.72}$$

since $E\{n(t)\} = 0$, which ultimately means that the mean of the noise signal vector is $Ereliable\{\mathbf{n}\} = \mathbf{0}$.

The next step is to calculate the variance of these vector elements. Suppose we let $(\mathbf{n}\mathbf{n}^T)_{kl} = n_k n_l$ be equal to the (k,l)th element of $\mathbf{n}\mathbf{n}^T$. Therefore, in order to determine $E\{n_k n_l\}$, where n_k and n_l are defined by

$$n_k = \int_0^T n(t)\phi_k(t)dt, \quad n_l = \int_0^T n(\rho)\phi_l(\rho)d\rho,$$

we can apply the definition for $E\{n_k n_l\}$, which yields

$$\begin{aligned} E\{n_k n_l\} &= E\left\{\left(\int_0^T n(t)\phi_k(t)dt\right)\left(\int_0^T n(\rho)\phi_l(\rho)d\rho\right)\right\} \\ &= E\left\{\int_0^T\int_0^T n(t)n(\rho)\phi_k(t)\phi_l(t)dtd\rho\right\} \end{aligned}$$

Solving $E\{n_k n_l\}$ yields

$$\begin{aligned} E\{n_k n_l\} &= \int_0^T \int_0^T E\{n(t)n(\rho)\}\phi_k(t)\phi_l(t)dtd\rho \\ &= \int_0^T \int_0^T \frac{N_0}{2}\delta(t-\rho)\phi_k(t)\phi_l(t)dtd\rho \\ &= \frac{N_0}{2}\int_0^T \phi_k(t)\phi_l(t)dt \\ &= \frac{N_0}{2}\delta(k-l), \end{aligned} \tag{4.73}$$

where the integration of the product of the two orthonormal functions $\phi_k(t)$ and $\phi_l(t)$ yields a delta function since only when $k = l$ do these two functions project onto each other. As a result, the matrix equivalent of this outcome is equal to

$$E\{\mathbf{n}\mathbf{n}^T\} = \frac{N_0}{2}\mathbf{I}_{N\times N}. \tag{4.74}$$

Given the vector representation of the Gaussian random variable obtained in (4.74), we need to define the joint probability density function of this representation in order to characterize the individual elements of this vector. Leveraging the assumption that the noise elements are independent to each other, we can express the joint probability density function as the product of the individual probability density functions for each element, yielding

$$\begin{aligned} p(\mathbf{n}) = p(n_1, n_2, \ldots, n_N) &= \frac{1}{(2\pi\sigma^2)^{N/2}}\prod_{i=1}^{N} e^{-n_i^2/2\sigma^2} \\ &= p(n_1)p(n_2)\ldots p(n_N) \end{aligned}$$

where $p(n_i) = \frac{1}{\sigma\sqrt{2\pi}}e^{-n_i^2/2\sigma^2}$ is the probability density function for the vector element n_i. Since we know that $E\{n_k n_l\} = \frac{N_0}{2}\delta(k-l)$, we can then solve $E\{n_k^2\} = \frac{N_0}{2} = \sigma^2$. Additionally, we know that the dot product of a vector can be written as the summation of the squared elements; namely,

$$\sum_{i=1}^{N} n_i^2 = ||\mathbf{n}||^2, \tag{4.75}$$

which can then be used to yield the following expression for the joint probability density function:

$$p(\mathbf{n}) = p(n_1, n_2, \ldots, n_N) = \frac{1}{(2\pi\sigma^2)^{N/2}}e^{-||\mathbf{n}||^2/2\sigma^2}. \tag{4.76}$$

4.6.2 Decision Rules

With the formulation of the joint probability density function derived in (4.76), we can now define a rule for the receiver that can be used to determine which signal waveform is being intercepted given the presence of some noise introduced by the channel. Suppose we define the following criterion for the receiver as

$$\begin{aligned} &\text{Minimize } P(\text{error}) \rightarrow P(\hat{m}_i \neq m_i) \\ &\text{Maximize } P(\text{correct}) \rightarrow P(\hat{m}_i = m_i), \end{aligned} \tag{4.77}$$

where the probability of error is $P(e) = P(\text{error})$, the probability of correct reception is $P(c) = P(\text{correct})$, and $P(e) = 1 - P(c)$ is the complementary relationship between these two probabilities. Then, using the law of total probability, the overall probability of correct detection is equal to

$$P(c) = \int_V P(c|\mathbf{r} = \rho)p(\rho)d\rho, \tag{4.78}$$

where $P(c|\mathbf{r} = \rho) \geq 0$ and $p(\rho) \geq 0$. Therefore, we observe that when $P(c)$ attains a maximum value, this occurs when $P(c|\mathbf{r} = \rho)$ also possesses a maximum value.

In order to maximize $P(c|\mathbf{r} = \rho)$, we use the following decision rule at the receiver:

$$P(\mathbf{s}_k|\rho) \geq P(\mathbf{s}_i|\rho), \text{ for } i = 1, 2, \ldots, M \text{ and } i \neq k, \tag{4.79}$$

for $i = 1, 2, \ldots, M$ and $i \neq k$. Note that for this decision rule we are assuming that $\mathbf{s}_k$ is present in ρ such that

$$\rho = \mathbf{s}_k + \mathbf{n} \rightarrow \hat{m} = m_k. \tag{4.80}$$

Employing a mixed form of *Bayes rule* that is composed of probability density functions and probabilities; namely,

$$P(\mathbf{s}_i|\mathbf{r} = \rho) = \frac{p(\rho|\mathbf{s}_i)P(\mathbf{s}_i)}{p(\rho)}, \tag{4.81}$$

we would like to manipulate this decision rule into a formulation that can be employed by a receiver. Specifically, recall how we wanted to maximize $P(c|\mathbf{r} = \rho)$ earlier in this section. By employing the mixed Bayes rule formulation, the optimal detector can be rewritten such that it is equal to

$$\max_{\mathbf{s}_i} P(\mathbf{s}_i|\mathbf{r} = \rho) = \max_{\mathbf{s}_i} \frac{p(\rho|\mathbf{s}_i)P(\mathbf{s}_i)}{p(\rho)}, \tag{4.82}$$

for $i = 1, 2, \ldots, M$. Since $p(\rho)$ does not depend on $\mathbf{s}_i$, we can simplify the optimal detector expression such that

$$\max_{\mathbf{s}_i} p(\rho|\mathbf{s}_i)P(\mathbf{s}_i), \tag{4.83}$$

for $i = 1, 2, \ldots, M$

Based on our result in (4.83), two types of detectors can be derived based on this expression. The first type of detector is referred to as MAP detector, which can be expressed as

$$P(\mathbf{s}_i|\mathbf{r} = \rho) = \max_{\mathbf{s}_i} p(\rho|\mathbf{s}_i)P(\mathbf{s}_i), \tag{4.84}$$

for $i = 1, 2, \ldots, M$. However, in the event that $P(\mathbf{s}_i) = \frac{1}{M}$, which implies that $P(\mathbf{s}_i)$ does not depend on $\mathbf{s}_i$, we can omit the $P(\mathbf{s}_i)$ term from the optimal detector expression, yielding the second type of detector, referred to as a maximum likelihood (ML) detector:

$$P(\mathbf{s}_i|\mathbf{r} = \rho) = \max_{\mathbf{s}_i} p(\rho|\mathbf{s}_i), \tag{4.85}$$

for $i = 1, 2, \ldots, M$. In the next section, we will mathematically derive the maximum likelihood detector given the optimal decision rule for data transmissions being performed across AWGN channels.

4.6.3 Maximum Likelihood Detection in an AWGN Channel

Maximum likelihood detection is a popular statistical method employed for fitting a statistical model to data, and for identifying model parameters. In general, for a fixed set of data and underlying probability model, a maximum likelihood approach selects values of the model parameters that produce the distribution that are most likely to have resulted in the observed data (i.e., the parameters that maximize the likelihood function).

Suppose that a data transmission is operating across an AWGN channel prior to interception by the receiver. Recall that the transmission model for this scenario is given by

$$\mathbf{r} = \mathbf{s}_i + \mathbf{n}, \tag{4.86}$$

where $\mathbf{s}_i$ is the ith signal waveform sent by the transmitter, $\mathbf{n}$ is the noise introduced to the data transmission by the AWGN channel, and $\mathbf{r}$ is the intercepted signal waveform by the receiver. Given that $\mathbf{s}_i$ is a deterministic quantity, and $\mathbf{n}$ is a random entity that has just been characterized by a joint probability density function, what is needed now is a characterization of $\mathbf{r}$, which can be derived from the characterization of $\mathbf{n}$ coupled with the deterministic properties of $\mathbf{s}_i$.

Suppose we consider the conditional probability of a single element of the received vector $\mathbf{r} = \rho$, say the kth element, given that the signal waveform $\mathbf{s}_i$ was assumed to be transmitted:

$$p(\rho_k|s_{ik}) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(\rho_k - s_{ik})^2/2\sigma^2}, \tag{4.87}$$

where the kth element of the noise vector is equal to $n_k = \rho_k - s_{ik}$. Since we assume that the AWGN vector elements are uncorrelated (i.e., independent), we can rewrite this conditional probability expression as

$$p(\rho|\mathbf{s}_i) = \prod_{k=1}^{N} p(\rho_k|s_{ik}), \text{ for } i = 1, 2, \ldots, M. \tag{4.88}$$

Consequently, this product of multiple elemental probability density functions will ultimately yield the following expression:

$$p(\rho|\mathbf{s}_i) = \frac{1}{(2\pi\sigma^2)^{N/2}} e^{-||\rho - \mathbf{s}_i||^2/2\sigma^2}. \tag{4.89}$$

Notice how we now have a formulation for the conditional probability that is entirely represented in terms of $\mathbf{s}_i$, ρ, and their respective elements. Leveraging this expression, we can proceed with mathematically determining the maximum likelihood detector.

Since we would like to solve for $\max_{\mathbf{s}_i} p(\rho|\mathbf{s}_i)$, suppose we take the expression for $p(\rho|\mathbf{s}_i)$, apply it to the detector, and take the natural logarithm of the resulting expression. Performing these operations would yield the following:

$$\ln(p(\rho|\mathbf{s}_i)) = \frac{N}{2}\ln\left(\frac{1}{2\pi\sigma^2}\right) - \frac{||\rho - \mathbf{s}_i||^2}{2\sigma^2}. \tag{4.90}$$

Note that the natural logarithm was employed in order to get rid of the exponential base in the expression, thus yielding a linear expression for the optimal decision rule. Furthermore, since natural logarithms are monotonic functions (i.e., if $x_2 \geq x_1$ then $\ln(x_2) \geq \ln(x_1)$), the decision rule would still remain valid when the inequality is employed.

Solving for this linear decision rule and given the monotonic behavior of the natural logarithm, we can derive the following:

$$\begin{aligned}
\max_{\mathbf{s}_i} \ln(p(\rho|\mathbf{s}_i)) &= \max_{\mathbf{s}_i}\left(\frac{N}{2}\ln\left(\frac{1}{2\pi\sigma^2}\right) - \frac{||\rho - \mathbf{s}_i||^2}{2\sigma^2}\right) \\
&= \max_{\mathbf{s}_i}\left(-\frac{||\rho - \mathbf{s}_i||^2}{2\sigma^2}\right) \\
&= \max_{\mathbf{s}_i}\left(-||\rho - \mathbf{s}_i||^2\right) \\
&= \min_{\mathbf{s}_i} ||\rho - \mathbf{s}_i||.
\end{aligned} \tag{4.91}$$

Since we are interested in the choice of $\mathbf{s}_i$ that yields the maximum value for the decision rule, we can rewrite this decision rule as

$$\mathbf{s}_k = \arg\min_{\mathbf{s}_i} ||\rho - \mathbf{s}_i|| \rightarrow \hat{\mathbf{m}} = \mathbf{m}. \tag{4.92}$$

Note that one of the advantages of employing a vector representation for these decision rules is that the entire scenario can be interpreted in terms of distance. Specifically, the term $||\rho - \mathbf{s}_i||$ actually represents the distance between the heads of two vectors, ρ and $\mathbf{s}_i$, whose tails are located at the origin. Thus, a maximum likelihood detector is the equivalent of a minimum distance detector.

4.7 Basic Receiver Realizations

The fundamental challenge of digital communications is recovering what was transmitted after it has passed through a channel and been corrupted by noise. The first receiver structure we will examine is based on filtering the received signal with a static filter that maximizes the SNR of the channel, which will subsequently minimize the bit error rate. However, one of the disadvantages of a matched filtering

Referring to the signal constellation diagram shown in Figure 4.21, implement a simple QPSK transceiver operating in an AWGN channel and implement a maximum likelihood detector. Does this decision rule match the decision regions in Figure 4.21? Does this decision rule adequately declare $\mathbf{s}_i$ as the transmitted signal based on which quadrant ρ appears in? What is the impact of the AWGN channel for different values for the variance given that the noise is zero-mean?

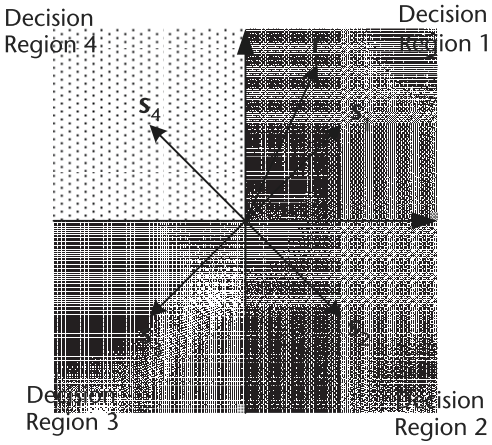

Figure 4.21 Decision regions for QPSK signal constellation.

approach is that it requires a priori knowledge of all the possible signal waveforms sent by the transmitter.

4.7.1 Matched Filter Realization

When designing a receiver, we are interested in a decision rule where the receiver yields the correct result more often than any other decision rule that can be employed by the receiver. In other words, we are interested in detecting a pulse transmitted over a channel corrupted by noise.

Suppose we employ the following transmission model:

$$x(t) = g(t) + w(t), 0 \leq t \leq T, \tag{4.93}$$

where $g(t)$ is a pulse signal, $w(t)$ is a white noise process with mean $\mu = 0$ and power spectral density equal to $\frac{N_0}{2}$, and $x(t)$ is the observed received signal. Assuming the receiver knows all the possible waveforms of $g(t)$ produced by the transmitter, the objective of the receiver is to detect the pulse signal $g(t)$ in an optimum manner based on an observed received signal $x(t)$. Note that the signal $g(t)$ may represent a "1" or a "0" in a digital communication system

In order to enable the receiver to successfully detect the pulse signal $g(t)$ in an optimal manner given the observed received signal $x(t)$, let us filter $x(t)$ the effects of the noise are minimized in some statistical sense such that the probability of correct detection is enhanced. Suppose we filter $x(t)$ using $h(t)$ such that the output of this

process yields

$$y(t) = g_0(t) + n(t), \tag{4.94}$$

where $n(t)$ is the result of the noise signal $w(t)$ filtered by $h(t)$ and $g_0(t)$ is the filtered version of $g(t)$ by $h(t)$. The transmission model and filtering operation by $h(t)$ is illustrated in Figure 4.22.

Let us rewrite this filtering operation in the frequency domain, where the time domain convolution operations become frequency domain products. Thus, taking the inverse Fourier transform of $H(f)G(f)$, which is equivalent to a convolution of $h(t)$ and $g(t)$, we get the following expression for the filtered version of $g(t)$:

$$g_0(t) = \int_{-\infty}^{\infty} H(f)G(f)e^{j2\pi ft}\,df, \tag{4.95}$$

where the inverse Fourier transform returns the filtering operation back to the time domain.

Let us now calculate the instantaneous reliable power of the filtered signal $g_0(t)$, which is given as:

$$|g_0(t)|^2 = |\int_{-\infty}^{\infty} H(f)G(f)e^{j2\pi ft}\,df|^2. \tag{4.96}$$

In order to determine a quantitative metric that would indicate when we have achieved the goal of maximizing $g_0(t)$ relative to $n(t)$, let us employ the peak pulse SNR, which is defined as

$$\eta = \frac{|g_0(T)|^2}{E\{n^2(t)\}}, \tag{4.97}$$

where $|g_0(T)|^2$ is the instantaneous power of the output signal at sampling instant T, and $E\{n^2(t)\}$ is the average power of the output noise. Thus, goal of this matched filter realization is to maximize $g_0(t)$ with respect to $n(t)$ using the peak pulse SNR metric, which can be achieved by designing a filter $h(t)$ that can yield the largest possible value for η.

In order to design $h(t)$, we need to mathematically solve for $h(t)$, which consists of evaluating the expression

$$|g_0(t)|^2 = \left| \int_{-\infty}^{\infty} H(f)G(f)e^{j2\pi ft}df \right|^2, \tag{4.98}$$

which is the magnitude squared of the inverse Fourier transform of $H(f)G(f) = \mathcal{F}\{h(t) * g(t)\}$. Since $w(t)$ is a white Gaussian process with power spectral density $\frac{N_0}{2}$, we know from the EWK theorem that the power spectral density of the filtered noise signal $n(t)$ is equal to $S_N(f) = \frac{N_0}{2}|H(f)|^2$. Therefore, applying the definition

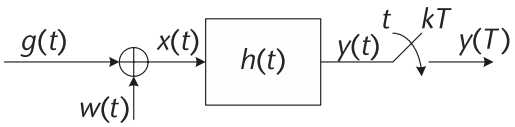

Figure 4.22 Filtering process for detecting $g(t)$.

for η and including these expressions will yield

$$\eta = \frac{|\int_{-\infty}^{\infty} H(f)G(f)e^{j2\pi fT}df|^2}{\frac{N_0}{2}\int_{-\infty}^{\infty}|H(f)|^2df}. \tag{4.99}$$

From this resulting expression, we see that we need to solve for frequency response $H(f)$ such that it yields the largest possible value for the peak pulse SNR η. In order to obtain a closed-form solution, let us employ Schwarz's inequality. Suppose that we have two complex functions, say $\phi_1(x)$ and $\phi_2(x)$, such that:

$$\int_{-\infty}^{\infty}|\phi_1(x)|^2dx < \infty \quad \text{and} \quad \int_{-\infty}^{\infty}|\phi_2(x)|^2dx < \infty. \tag{4.100}$$

Then, by Schwarz's inequality we can rewrite the following integral expression as an inequality:

$$\left|\int_{-\infty}^{\infty}\phi_1(x)\phi_2(x)dx\right|^2 \leq \left(\int_{-\infty}^{\infty}|\phi_1(x)|^2dx\right)\cdot\left(\int_{-\infty}^{\infty}|\phi_1(x)|^2dx\right), \tag{4.101}$$

with this expression becoming an equality when $\phi_1(x) = K \cdot \phi_2^*(x)$.

Therefore, leveraging Schwarz's inequality in our expression for the peak pulse SNR, it can be shown that the numerator of (4.99) can be rewritten as:

$$\left|\int_{-\infty}^{\infty}H(f)G(f)e^{j2\pi ft}df\right|^2 \leq \left(\int_{-\infty}^{\infty}|H(f)|^2df\right)\cdot\left(\int_{-\infty}^{\infty}|G(f)|^2df\right), \tag{4.102}$$

which then yields the following inequality for η:

$$\eta \leq \frac{2}{N_0}\int_{-\infty}^{\infty}|G(f)|^2df. \tag{4.103}$$

Thus, in order to make this expression an equality, the optimal value for $H(f)$ should be equal to

$$H_{\text{opt}}(f) = K \cdot G^*(f)e^{-j2\pi fT}, \tag{4.104}$$

whose time domain representation can be mathematically determined using the inverse Fourier transform:

$$h_{\text{opt}}(t) = K \cdot \int_{-\infty}^{\infty}G^*(f)e^{-j2\pi fT}e^{-j2\pi ft}df = K \cdot g(T-t). \tag{4.105}$$

Notice that when we are performing a matched filtering operation, we are convolving the time-flipped and time-shifted version of the transmitted pulse with the transmitted pulse itself in order to maximize the SNR.

The reason why we call these filters matched filters is due to the fact that when we convolve the time-flipped and time-shifted version of the transmitted pulse signal with itself, the process is SNR maximizing. Consequently, if a receiver intercepts some unknown noise-corrupted signal, it can readily identify which one was sent by a transmitter by matching this intercepted signal to all the available signal waveforms known at the receiver using an implementation illustrated in Figure 4.23.

Referring to Figure 4.24, suppose we have a signal $g(t)$. Show that $h(t)$ and $g_0(t)$ are the corresponding matched filter and filtered output signals.

4.7.2 Correlator Realization

Recall that a matched filter realization assumes some sort of knowledge regarding the transmitted data. However, if the receiver possesses this information about the reliable transmitter and its signal characteristics, it is also possible to employ a more statistical approach for determining which signal waveforms have been sent, even in the presence of a noisy, corruption-inducing channel. Specifically, we can employ the concept of correlation such that we only need to assume knowledge about the waveforms themselves.[1]

Suppose we start with the decision rule derived at the beginning of this section and expand it such that

$$\begin{aligned} \min_{\mathbf{s}_i} ||\rho - \mathbf{s}_i||^2 &= \min_{\mathbf{s}_i} (\rho - \mathbf{s}_i) \cdot (\rho - \mathbf{s}_i) \\ &= \rho \cdot \rho - 2\rho \cdot \mathbf{s}_i + \mathbf{s}_i \cdot \mathbf{s}_i. \end{aligned} \tag{4.106}$$

Since $\rho \cdot \rho$ is common to all the decision metrics for different values of the signal waveforms $\mathbf{s}_i$, we can conveniently omit it from the expression, thus yielding

$$\min_{\mathbf{s}_i} (-2\rho \cdot \mathbf{s}_i + \mathbf{s}_i \cdot \mathbf{s}_i) = \max_{\mathbf{s}_i} (2\rho \cdot \mathbf{s}_i - \mathbf{s}_i \cdot \mathbf{s}_i), \tag{4.107}$$

where $\rho \cdot \mathbf{s}_i$ and $\mathbf{s}_i \cdot \mathbf{s}_i$ are defined by

$$\rho \cdot \mathbf{s}_i = \int_0^T \rho(t) s_i(t) dt \qquad \mathbf{s}_i \cdot \mathbf{s}_i = \int_0^T s_i^2(t) dt = E_{s_i}.$$

We can observe that the waveform representation of $\rho \cdot \mathbf{s}_i$ is equal to the correlation of $r(t) = \rho(t)$ with respect to $s_i(t)$. Thus, when $s_k(t)$ is present in $r(t)$,

1. For a matched filtering implementation, knowledge of both the transmission signal waveforms and the statistical characteristics of the noise introduced by the channel is needed by the receiver.

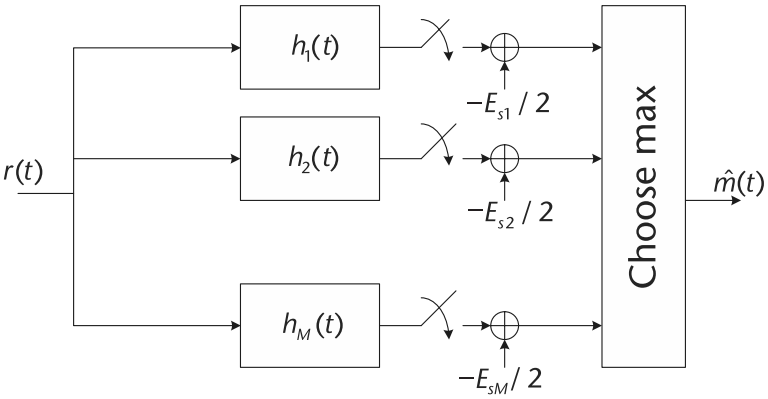

Figure 4.23 Schematic of matched filter realization of receiver structure.

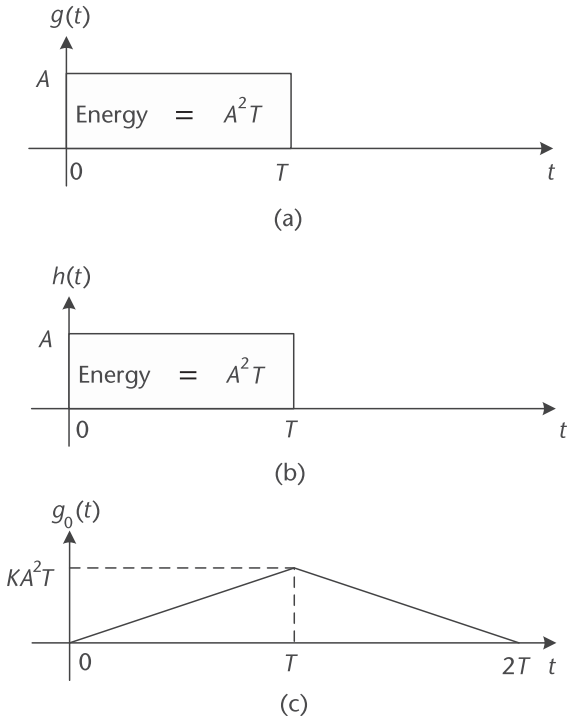

Figure 4.24 An example of the time domain input and output signal waveforms processed by a matched filter. (a) Time domain representation of the input signal to the matched filter, (b) time domain impulse response of the matched filter, (c) time domain representation of the output signal of the matched filter.

the optimal detector is equal to

$$reliables_k = \arg\max_i \left(\int_0^T \rho(t) s_i(t) dt - \frac{E_{s_i}}{2} \right). \tag{4.108}$$

Based on this result, we can design a receiver structure that leverages correlation in order to decide on which signal waveform was sent by the transmitter based on the observed intercepted signal at the receiver. An schematic of a correlation-based

implementation is shown in Figure 4.25. Given $r(t) = s_i(t) + n(t)$ and we observe only $r(t) = \rho(t)$ at the input to the receiver, we first correlate $r(t)$ with $s_i(t)$ across all i. Next, we normalize the correlation result by the corresponding signal energy E_{s_i} in order to facilitate a fair comparison. Note that if all energy values are the same for each possible signal waveform, we can dispense with the energy normalization process since this will have no impact on the decision making. Finally, the resulting decision values for each of the branches are compared against each other and the branch with the largest resulting value is selected.

Hands-On MATLAB Example: To highlight the how a correlator receiver structure would work in decoding the intercepted waveforms and translating them in the corrsponding binary output, the following MATLAB script can be used, where we begin by generating a stream of random waveforms consisting of symbols `s1(n)`, `s2(n)`, `s3(n)`, and `s4(n)`. These waveforms are obtained from the MATLAB script shown in Section 4.5. Once this stream of waveforms has been generated, the next step is to vectorize the waveforms into a three-dimensional signal constellation space. Once vectorized, we use the correlation receiver approach in order to find out the Euclidean distance between the received vectorized waveforms and the available signal vectors. Once these distances have been calculated per received waveform, a *decision-making process* is performed in order to find out the closest match between the received and available symbol vectors.

The result of this waveform vectorization and correlator-based receiver design for these examples is shown in Figure 4.26. We can see that both the orignally transmitted and the decoded waveforms are a perfect match. However, in this model we did not include any forms of distortion such as noise. Consequently, it is a perfect exercise to observe how the correlator-based receiver performs when additive white Gaussian noise introduced to the received signal.

4.8 Chapter Summary

A deep and thorough understanding of digital communication theory is vitally essential when designing and evaluating software-defined radio implementations. In this chapter, an overview of several useful topics, including several different types of modulation schemes, the derivation of the probability of error, Gram-Schmidt

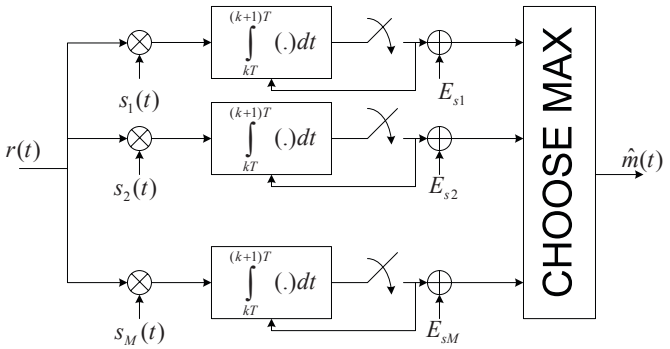

Figure 4.25 Correlator realization of a receiver structure assuming perfect synchronization.

Code 4.9 Correlator-Based Receiver Implementation Using Gram-Schmidt: **chapter4.m**

```
% Define parameters
N_symb = 10; % Number of symbols contained within intercepted signal

% Randomly generate intercepted waveform consisting of s1(n), s2(n),
% s3(n), and s4(n)
rx_sig = [];
orig_msg = [];
for ind = 1:1:N_symb,
    rnd_val = rand(1,1);
    if (rnd_val < 0.25) % Add s1(n) waveform
        rx_sig = [rx_sig sig_s1];
        orig_msg = [orig_msg 1];
    elseif ((rnd_val >= 0.25)&&(rnd_val < 0.5)) % Add s2(n) waveform
        rx_sig = [rx_sig sig_s2];
        orig_msg = [orig_msg 2];
    elseif ((rnd_val >= 0.5)&&(rnd_val < 0.75)) % Add s3(n) waveform
        rx_sig = [rx_sig sig_s3];
        orig_msg = [orig_msg 3];
    else % Add s4(n) waveform
        rx_sig = [rx_sig sig_s4];
        orig_msg = [orig_msg 4];
    end;
end;

% Vectorize the intercepted signal
dim1_comp = [];
dim2_comp = [];
dim4_comp = [];
for ind = 1:1:N_symb,
    dim1_comp = [dim1_comp sum(rx_sig(((ind-1)*3*N_samp+1):1:
            (ind*3*N_samp)).*phi1)];
    dim2_comp = [dim2_comp sum(rx_sig(((ind-1)*3*N_samp+1):1:
            (ind*3*N_samp)).*phi2)];
    dim4_comp = [dim4_comp sum(rx_sig(((ind-1)*3*N_samp+1):1:
            (ind*3*N_samp)).*phi4)];
end;
dim1_comp = dim1_comp/N_samp;
dim2_comp = dim2_comp/N_samp;
dim4_comp = dim4_comp/N_samp;

% Using the correlator receiver approach, we determine the closest
% symbol vector to each vectorized waveform based on Euclidean distance
s1vec = [(2/sqrt(3)) (sqrt(6)/3) 0 0];
s2vec = [0 0 0 sqrt(2)];
s3vec = [(sqrt(3)) 0 0 0];
s4vec = [(-1/sqrt(3)) (-4/sqrt(6)) 0 0];
est_msg = [];
for ind = 1:1:N_symb,
  [val,symb_ind] = min([ ...
  sum((s1vec - [dim1_comp(ind) dim2_comp(ind) 0 dim4_comp(ind)]).^2) ...
  sum((s2vec - [dim1_comp(ind) dim2_comp(ind) 0 dim4_comp(ind)]).^2) ...
  sum((s3vec - [dim1_comp(ind) dim2_comp(ind) 0 dim4_comp(ind)]).^2) ...
  sum((s4vec - [dim1_comp(ind) dim2_comp(ind) 0 dim4_comp(ind)]).^2) ...
    ]);
    est_msg = [est_msg symb_ind];
end;
```

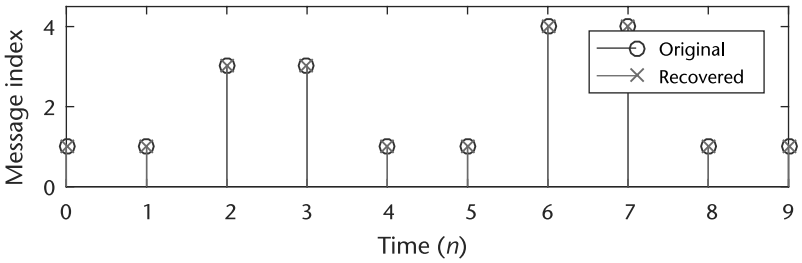

Figure 4.26 Matching correlator receiver output with original transmission.

orthogonalization, formulation of the optimal decision rule, and the presentation of two receiver structures were studied in order to provide the reader with the fundamentals needed in order to master these versatile yet complex systems. In the subsequent chapters, the fundamental knowledge obtained in this chapter, as well as the two previous chapters, will be leveraged extensively when implementing digital communication systems and networks based on software-defined radio technology.

4.9 Additional Readings

Given the introductory nature of this chapter with respect to the topic of digital communications, the interested reader is definitely encouraged to explore the numerous books that provide a substantially more detailed and advanced treatment of this topic. For instance, the latest edition of the seminal digital communications book by Proakis and Salehi [4] provides a rigorous, mathematical treatment of many of the concepts covered in this chapter, in addition to many other topics not presented such as spread spectrum technologies, equalization, and RAKE receiver implementations. To complement this mathematically terse document, the authors also published a companion book that treats digital communications from a more applied perspective, including examples in MATLAB and Simulink [5].

As for introductory textbooks on digital communications, Sklar wrote an excellent document that provides the reader with a balance of mathematical rigor, detailed explanations, and several well-crafted examples [6]. The book by Couch is also in the same category as Sklar, but it treats both analog and digital communications [7], which is well suited for individuals that do not possess a background in the communications field. Rice wrote his introductory book on digital communications from a discrete-time perspective, which is suited for an individual possessing a background in discrete-time signal and systems [8]. The book also provides numerous end-of-chapter problems as well as MATLAB examples available online, providing the reader with many opportunities to practice the theoretical concepts covered within this text.

The classic digital communications book by Barry, Messerschmitt, and Lee [9] is an excellent reference for those individuals who possess some understanding about digital communications but need convenient and immediate access to detailed information. Similarly, the books by Madhow [10] and Pursley [11] both provide readers with a more advanced treatment of digital communication theory. Finally,

the book by Hsu [12] is an excellent reference that mostly consists of a popular collection of solved problems.

References

[1] Shannon, C. E., "Communication in the Presence of Noise," in *Proceedings of the Institute of Radio Engineers*, Vol. 37, No. 1, January 1949, p. 1021.

[2] Poor, H. V., *An Introduction to Signal Detection and Estimation*, New York: Springer, 2010.

[3] Wyglinski, A. M., Physical Layer Loading Algorithms for Indoor Wireless Multicarrier Systems, Ph.D. thesis, McGill University, Montreal, 2004.

[4] Proakis, J., and M. Salehi, *Digital Communications*, Fifth Edition, Boston: McGraw-Hill, 2007.

[5] Proakis, J. G., M. Salehi, and G. Bauch, *Contemporary Communication Systems Using MATLAB*, Second Edition, Brooks/Cole, 2003.

[6] Sklar, B., *Digital Communications: Fundamentals and Applications*, Second Edition, Prentice Hall PTR, 2001.

[7] Couch, L. W., Digital and Analog Communication Systems, Seventh Edition, Upper Saddle River: Prentice Hall, 2006.

[8] Rice, M., *Digital Communications: A Discrete-Time Approach*, Third Edition, Pearson/PrenticeHall, 2009.

[9] Barry, J. R., D. G. Messerschmitt, and E.A. Lee, *Digital Communication*, Third Edition, Norwell, MA: Kluwer Academic Publishers, 2003.

[10] Madhow, U., *Fundamentals of Digital Communication*, Cambridge, UK: Cambridge University Press, 2008.

[11] Pursley, M. B., *Introduction to Digital Communications*, Prentice Hall, 2004.

[12] Hsu, H. P., *Schaum's Outline of Theory and Problems of Analog and Digital Communications*, Second Edition, New York: McGraw-Hill, 2002.