This thesis proposes encoding information onto random distribution parameters so that the same exact signal may be used for both radar and communications, simultaneously. Orthogonal Frequency Division Multiplexing (OFDM) sub-carrier weights are used to transmit samples of a random distribution. Weibull and Rayleigh distributions are explored and friendly reception in simulation for Rayleigh achieved a bit error rate (BER) of $10^{-3}$. Assuming a worst case scenario where an eavesdropper is missing only one piece of information about the signal, we show that using at least 32 sub-carriers they are unable to achieve a BER less than 30% for both Weibull and Rayleigh distributions. Finally, experimental results show similar trends in penalization using a Weibull distribution.
A Thesis Proposal

Submitted to the

Faculty of Miami University

in partial fulfillment of

the requirements for the degree of

Master of Science in Computational Science and Engineering

Department of Electrical & Computer Engineering

by

Daniel Kellett

Miami University

Oxford, Ohio

2016

Advisor

____________________________

Dr. Dmitriy Garmatyuk

Reader

____________________________

Dr. Chi-Hao Cheng

Reader

____________________________

Dr. Paul Urayama
# Contents

## Contents

List of Tables iv

List of Figures v

Acknowledgments vi

1 Introduction 1

2 Background 4

2.1 Wireless Communication Essentials 4

2.1.1 From Maxwell to the Electromagnetic Wave Equation 4

2.1.2 The Basic Communication System 6

2.1.3 Orthogonal Frequency Division Multiplexing 7

2.1.4 Ultra-Wideband Signals 10

2.2 Properties of Statistical Distributions 11

2.2.1 Weibull Distribution 12

2.2.2 Rayleigh Distribution 13

2.3 Radar Functionality and Modes of Operation 14

2.3.1 The Radar Equation 14

2.3.2 Range Resolution and Maximum Unambiguous Range 15

2.3.3 The Doppler Effect 18

2.3.4 Noise Radar 19

2.4 Fusion of Radar and Communication Systems 20

2.4.1 Example Scenarios 21

2.4.2 Current State of the Art 21

2.4.3 Random Sequence Encoding 23

3 Random Sequence Encoding 24

3.1 Simulation Method 24

3.1.1 Encoding Data onto a Weibull Distribution 25

3.1.2 Encoding Data onto a Rayleigh Distribution 27

3.2 Communication Reliability 28

3.2.1 Weibull Distribution BER vs SNR 28
List of Tables

4.1 System Specifications ......................................................... 37
# List of Figures

2.1 Block Diagram of a Basic Communication System ........................................ 6
2.2 OFDM Signal Sub-Carrier Spacing .............................................................. 8
2.3 Block Diagram of an OFDM Communication System ....................................... 8
2.4 UWB vs Narrowband Signal ......................................................................... 10
2.5 Weibull PDF for $\lambda = 1$ and different $k$ .................................................... 13
2.6 Rayleigh PDF for different scale parameters, $b$ .............................................. 14
2.7 Example of Range Resolution and Target Ambiguity ....................................... 16
2.8 Signal Return from Multiple Targets with Different Spacing over Discrete Time Steps .......................................................................................... 16
2.9 Maximum Unambiguous Range for a $PRF = 500$ kHz .................................... 17
2.10 Visualization of Doppler Shift due to a Moving Source .................................... 18
3.1 Example signals for Weibull distribution using $2^x$ data mapping and 32 sub-carriers .................................................................................. 26
3.2 Example signals for Rayleigh distribution using linear data mapping and 32 sub-carriers .................................................................................. 28
3.3 BER vs SNR using Sample Variance of a Weibull Distribution ......................... 29
3.4 BER vs SNR using Sample Mean of a Weibull Distribution ............................... 29
3.5 BER vs SNR using Sample Variance of a Rayleigh Distribution ....................... 31
3.6 BER vs SNR using Sample Mean of Rayleigh Distribution .............................. 31
3.7 Interceptor Penalization using Sample Variance of a Weibull Distribution with Shape Parameter, $k$, and $2^x$ Data Mapping (Top) and Linear Data Mapping (Bottom) ......................................................... 32
3.8 Interceptor Penalization using Sample Mean of a Weibull Distribution with Shape Parameter, $k$, and $2^x$ Data Mapping (Top) and Linear Data Mapping (Bottom) ......................................................... 33
3.9 Interceptor Penalization using Sample Variance of Rayleigh Distribution .......... 34
3.10 Interceptor Penalization using Sample Mean of Rayleigh Distribution .......... 34
4.1 Block Diagram of Miami University’s SDRCS AFE ........................................ 36
4.2 BER vs SNR using a Weibull Distribution for RSE ......................................... 39
4.3 Interceptor Penalization using Sample Mean (Top) and Sample Variance (Bottom) of a Weibull Distribution with Shape Parameter, $k$, $2^x$ Data Mapping. 40
Acknowledgments

Many thanks to Dr. Dmitriy Garmatyuk for his support, patience, and advise throughout my Master’s work, without which this work would never have been done. Thank you, Dr Chi-Hao Cheng and Dr. Paul Urayama for making the time to serve on my committee, as well as for the thoughtful and insightful criticism of my work during the proposal stage. I would also like to thank Dr. Donald R. Ucci and Dr. Qihou Zhou for their encouragement throughout my time at Miami University. I am particularly grateful to all of the faculty in the Department of Electrical and Computational Engineering for their guidance and support regarding my decision to pursue both my master’s Degree, and now my Ph.D. Lastly, I would like to thank my friends and peers in the ECE department who helped me maintain my sanity throughout my work.
Chapter 1

Introduction

Compared to the prevalence and reliance upon electromagnetic communications and sensing in today’s society, the origins of such technology seem humble instead of the ground-breaking achievements they were. Wireless telegraphs developed in the late 1800s were larger and less capable than the most basic functions of the phones we now carry in our pockets with us everywhere. The radar systems developed in the early 1930’s and used to simply detect incoming bombers as an early warning system during WWII are now capable of aiding in weather predication, studying the upper atmosphere, guiding aircraft, creating topographical maps, and allowing for imaging of low-visibility environments [1]. These sophisticated systems have become a unanimous part of everyday life. As such, we are constantly striving to reduce the size and cost while improving the functionality and spectral efficiency of electromagnetic signalling devices.

One of the earliest improvements in reducing the weight and space required by these systems was driven by NASA’s space shuttle orbiter. NASA engineers, needing to minimize weight and maximize space efficiency on the space shuttle, realized that due to the primary difference between a communications signal and a radar signal being in the signal processing aspect rather than the physical aspect of the radiowaves themselves and so proposed combining the two systems for the Ku-band Space Shuttle Orbiter [2]. While they were ultimately only able to share the transmitter, receiver, and antenna of the subsystem due to the reliance on analog signal processing of the time, this still marked the beginning of combining radar and communication systems.

As radar and communication systems increased in complexity, the cost of hardware for individual analog components for each system continued to rise. However, many of these components were redundant, and with advancements made in digital computing, E-Systems (now Raytheon) developed one of the first software-defined radios (SDR) [3]. Such a system allowed for a single computer to take over the job of multiple analog components and made it possible for one hardware system to adapt to a multitude of technology standards requiring use of different frequency bands [4]. With this technology, the transmitter, receiver, and antenna shared by the Ku-band Space Shuttle Orbiter subsystem could become the only hardware required aside from the computer used to define the signal processing.
Indeed, the complex systems we have developed are in great need of components with multifunctional capabilities if trends in research are any indicator. In 1996, the US Navy launched the Advanced Multifunction RF System (AMRFS) Program to combat the issue of ever-increasing top-side ship antennas, while continuing to not only maintain, but increase the effective functionality and bandwidth of the RF system apertures [5]. Outside of the military, the quasi-orthogonality of opposite-sloped linear frequency modulated (LFM) signals was exploited to achieve simultaneous communication and radar operation through a shared antenna aperture by UC-Santa Barbara [6]. In autonomous land vehicles, low-cost, reliable equipment for navigation is necessary to be able to reach a broader, commercial market [7].

With radar and communication systems’ front-end architecture becoming increasingly similar through the help of SDR and demand for multifunctional systems, the possibility of system fusion starts becoming a realistic option [8]. This is particularly desirable from the view point of increasing spectral efficiency as well. Though some may argue that the issue of spectral crowding is currently an organizational, rather than technical, problem [9], in our increasingly digitized world it may only be a matter of time before organizational changes will no longer suffice as a solution. System fusion is particularly desirable for, and has been driven by, unmanned aerial vehicles whose fixed payload from the required sensors to enable unmanned flight becomes very costly.

Fusion of radar and communication systems can be achieved in several ways, each naturally coming with their advantages and disadvantages. A straightforward approach was seen with the Ku-band Space Shuttle Orbiter subsystem, where the system could operate either as a radar system or a communications system. By dividing the access time each system had to the medium, both systems could continue to operate independently from each other. However, this also limits the time each function of the subsystem may be active. An example of this Time Division Medium Access (TDMA) approach can be seen in [10] where they used an LFM chirp for radar functionality and binary phase-shift keying (BPSK) for communications. They were able to detect multiple targets up to 100 meters away and transmit data at a speed of 25 Mbps. However, they had to make a trade-off between spending more time on communications and possibly missing a detection, or spending more time on radar and reducing communication speeds.

A more complete fusion of the two systems overcomes issues surrounding time that can be allotted, but comes with its own problems to solve. In particular, radar and communication signals typically differ in waveform design and parameter selections. For example, radar waveform design typically places heavy emphasis on optimizing autocorrelation properties, and radar signals also often use a larger bandwidth than typically used for communications. In order to meet these radar requirements, a spread-spectrum signal such as orthogonal frequency division multiplexing (OFDM) is usually necessary. Furthermore, the same Doppler shift that provides valuable information to a radar system is often detrimental to a communications system. Thus, a balance must be found in a method to allow for toleration of Doppler shift when processing information and in extracting velocity information from radar processing. Strum and Weisbeck cover these considerations, and more, in great depth in [8]. They concluded that an OFDM signalling strategy offered good performance for both radar
and single-user communications, while a direct sequence spread spectrum (DSSS) signalling strategy allowed for better multi-user communications.

Of further interest in radar-communication system fusion is the possession of low probability of intercept/detection (LPI/LPD) characteristics. While an autonomous civilian vehicle may be able to get away with the easily identifiable LFM chirp for radar, autonomous vehicles in more hostile environments will ideally be able to sense their environment with radar while going unnoticed themselves. Furthermore, should the signal be intercepted by an eavesdropper, it is desirable to prevent them from being able to identify what sort of signal it is to protect the data transmitted. To achieve this, inherent qualities of signalling techniques such as ultra-wideband (UWB) transmissions are often combined with noise-like characteristics.

In this thesis, a new approach to radar and communications fusion is proposed. The above methods use either time-division or embed communications within a radar signal, ultimately making inefficient use of time, spectrum, and/or power. This thesis proposes using the exact same waveform for both communications and radar sensing, and focuses on the communications aspect for asynchronous, ad-hoc communications in a hostile environment. We encode data onto a random sequence by exploiting properties of random distributions and transmit the random sequence on an UWB, OFDM signal. The effectiveness of this random sequence encoding is assessed in terms of communication effectiveness (SNR vs BER) and eavesdropper penalization.

The remainder of this thesis will proceed as follows: In chapter 2, we will review the communications principles surrounding OFDM and UWB systems. We will also cover the statistical properties required to understand this thesis and the distributions used for random sequence encoding. This will be followed by a review of radar theory and an overview research on noise radar to provide an argument for why we believe this method to be an effective means of fusing radar and communication systems. We will then proceed to cover, in depth, the proposed random sequence encoding method used here, the metrics used to assess its effectiveness, and provide the simulation results used to establish a proof-of-concept. In addition, we will compare the use of the two parameter Weibull distribution with the single parameter Rayleigh distribution. We then cover the system used for experimentation, present the experimental results obtained using a Weibull distribution, and discuss their significance. Finally, we present our conclusions and outline the future work we see is necessary and that could further benefit this research.
Chapter 2

Background

2.1 Wireless Communication Essentials

2.1.1 From Maxwell to the Electromagnetic Wave Equation [11]

The possibility of wireless communication was first postulated by James C. Maxwell in 1864, shortly after his 1861 paper which integrated Gauss’s Law for electrostatics and magnetostatics, Ampere’s Law, and Faraday’s Law. This theory was later studied and verified by Heinrich Hertz in the 1880’s [12].

The basic principle of wireless communications relies on the interaction of Faraday’s (2.1) and Ampere’s (2.2) Laws, which can roughly be interpreted as ”a time-varying magnetic field induces an electric field” and ”a time-varying electric field induces a magnetic field,” respectively.

\[
\nabla \times E = -\frac{\partial B}{\partial t} \quad (2.1)
\]

\[
\nabla \times B = \mu_0 \epsilon_0 \frac{\partial E}{\partial t} \quad (2.2)
\]

\[
\nabla \times E(x,t)\hat{j} = \left| \begin{array}{ccc}
\hat{i} & \hat{j} & \hat{k} \\
\frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \\
0 & 0 & E(x,t) \\
\end{array} \right|
\]  

\[
\nabla \times E(x,t)\hat{j} = \frac{\partial E}{\partial x} \hat{k} \quad (2.3)
\]

It is from this relation of these equations that we are able to define the speed of light, as well as create useful equations for describing electromagnetic waves. If we assume an electric field is varying with time along the y-axis of a 3-D plot, then we can find the solution to (2.1) as follows

Similarly, if we assume a magnetic field is varying with time along the x-axis, we can find a
solution to (2.2)

$$\nabla \times \mathbf{B}(x,t) \hat{k} = \begin{vmatrix} \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \\ \hat{i} & \hat{j} & \hat{k} \\ 0 & 0 & B(x,t) \end{vmatrix}$$

$$= - \frac{\partial B}{\partial x} \hat{j}$$

Equating the magnitudes of (2.3) and (2.4) to the right hand sides of (2.1) and (2.2), respectively, we get

$$\frac{\partial E}{\partial x} = - \frac{\partial B}{\partial t} \quad (2.5)$$

$$\frac{\partial B}{\partial x} = - \mu_0 \epsilon_0 \frac{\partial E}{\partial t} \quad (2.6)$$

From this we can see that a spatial changes in $E$ or $B$ produce variations in $B$ or $E$, respectively, with time. We can solve the partial derivatives of (2.5) and (2.6) with respect to $x$ to get (2.7) and (2.8)

$$\frac{\partial^2 E}{\partial x^2} = - \frac{\partial}{\partial x} \frac{\partial B}{\partial t}$$

$$= - \frac{\partial}{\partial t} \left( - \mu_0 \epsilon_0 \frac{\partial E}{\partial t} \right)$$

$$= \mu_0 \epsilon_0 \frac{\partial^2 E}{\partial t^2} \quad (2.7)$$

$$\frac{\partial^2 B}{\partial x^2} = - \frac{\partial}{\partial x} \left( - \mu_0 \epsilon_0 \frac{\partial E}{\partial t} \right)$$

$$= - \frac{\partial}{\partial t} \left( - \frac{\partial B}{\partial t} \right)$$

$$= \mu_0 \epsilon_0 \frac{\partial^2 B}{\partial t^2} \quad (2.8)$$

Simple solutions to these differential equations are

$$E(x,t) = E_{max} \cos (kx - \omega t) \quad (2.9)$$

$$B(x,t) = B_{max} \cos (kx - \omega t) \quad (2.10)$$

These can then be tweaked to match the familiar form of the general equation for a sinusoidal electromagnetic wave shown in (2.11) with the addition of the attenuation factor $e^{\alpha z}$ and phase offset, $\phi$.

$$A(z,t) = A_0 e^{\alpha z} \cos(\omega t - \beta z + \phi) \quad (2.11)$$

Equation (2.11) gives the amplitude of the wave $A$, at position, $z$, and time, $t$, with attenuation constant, $\alpha$ and phase constant (sometimes referred to as wave number), $\beta$. From
this, we can identify 3 main attributes of the wave that can be used for communicating information which depend on their initial values at the point of origin, $z = 0$:

- The original amplitude, $A_0$.
- The frequency $f$, embedded in the angular frequency $\omega = 2\pi f$.
- The phase offset, $\phi$.

The fact that we can express electromagnetic waves as a sinusoid is significant in that we can now analyze a wide variety of complex signals using the Fourier series, which expands the periodic function $f(x)$ into an infinite sum of weighted sines and cosines.

### 2.1.2 The Basic Communication System

Traditionally, we have transmitted information on electromagnetic waves by varying the amplitude (amplitude modulation), frequency (frequency modulation), or the phase (phase modulation). By mapping a number of binary symbols to pre-determined values of these wave attributes, we can transmit a stream of binary information from a transmitting antenna, to a receiving antenna. Fig. 2.1 depicts a block diagram of this basic communication system, obtained from [13].

![Figure 2.1: Block Diagram of a Basic Communication System](image)

Signals may be broadly classified into continuous-time analog signals or discrete-time digital signals. In Fig. 2.1, the input message is the exact information we wish to convey, typically in human-understandable form (e.g., human speech). Oftentimes, this message is some continuous-time, analog signal. In order to convert this message from human-understandable form to a form suitable to for transmission, an input transducer is required. Continuing with the human voice example, when talking on the phone, the microphone will convert the sound waves from into variations of one (or possibly more) parameters of an electromagnetic signal. By converting the sound waves to, say, voltage variations over time, the input transducer generates a message signal that can then be transmitted.

Depending on the device, the transmitter and and transducer may be one and the same. However, it is often desirable, even when not necessary, to systematically vary an attribute of the EM wave as a function of the message signal. Furthermore, the EM signal needs a base carrier frequency around which its attribute(s) are varied. This modulation of the signal allows for ease of signal radiation, reduction of noise and interference, channel assignment for
transmission, multiplexing multiple signals onto the same channel, and overcome equipment
limitations. Beyond modulation of the message signal, the transmitter couples the modulated
signal to the channel and will likely also include filtering and amplification of the message
signal.

The transmitted signal will then pass through the channel, or space between the transmitter
and receiver. Radio signals, such as those emitted by a cellphone, propagate through the
atmosphere, or free space. Some devices may have a wire connecting the transmitter to the
receiver such as computer peripherals like a monitor. In both cases, the transmitted signal
will experience various forms of degradation from the channel. Thermal noise, interference
from other signals in the channel, switching noises in networks, and more all degrade the
signal that is received.

Once a signal is received, it is necessary to reverse the modulation done by the transmitter,
or demodulate the signal. The receiver may perform other functions as well, such as scaling
or delaying the received signal. Once we finally have a demodulated output signal, we must
convert the signal back into the desired form. (e.g. electric waves back into acoustic waves
emitted by a speaker.)

### 2.1.3 Orthogonal Frequency Division Multiplexing

In communications, it is advantageous to transmit on multiple carriers in the same broadband
medium. However, using multiple carrier frequencies within the same spectrum results in
intersymbol interference (ISI) due to carrier harmonics. OFDM simultaneously mitigates
ISI and increases the datarate by minimizing the frequency spacing of sub-carriers while
maintaining their orthogonality as shown in Fig. 2.2. Signals are considered orthogonal
when they have [13]:

1. good autocorrelation at zero time delay
2. zero or low autocorrelation at non-zero time delays
3. zero or low cross-correlation at any time delay with other waveforms the system may
   encounter

Item 3 must be designed for on a case-by-case basis, while items 1 and 2 do not. By lining
up the peak of each sub-carrier to correspond to the troughs of all other sub-carriers used,
OFDM can ensure orthogonality of its sub-carriers, thus allowing for tightly packed sub-
carriers. (Fig. 2.2)

For this reason, OFDM is a method of transmitting signals with excellent spectral effi-
ciency and can approach near-theoretical channel capacity when appropriate information is
known[14]. Though first explored in the 1960s, OFDM first began to be used in practice in
the 90s with availability of fast digital signal processors and software-defined systems [15].
Since then, it has been implemented in a variety of systems and networks, including the
IEEE 802.11a local area network (LAN) standard [16] and IEEE 802.16a metropolitan area
network (MAN) standard [17].
A block diagram of the OFDM system used in this thesis is shown in Fig. 2.3. The message signal $a(t)$ is mixed with the carrier frequency and transmitted across the channel. On the reception, the in-phase and quadrature components are demodulated and then combined.

The following equations step through the transmit-receive process of OFDM signals and begins by defining the vector $\mathbf{S}$ in the frequency domain by selecting the desired coefficients of the $N$ sub-carriers used. These will create the $(2N+1)$-length vector in (2.12), where $0$ is the DC-point.

$$\mathbf{S} = [S(N), S(N-1), ..., S(1), 0, S(1), ..., S(N)]$$

The resultant discrete time signal in the time-domain is then found by performing the inverse discrete time Fourier transform using the inverse fast Fourier transform (IFFT) shown in
In (2.13), \( S(k) \) is the real amplitude of the \( k^{th} \) sub-carrier. Once we have the discrete time signal, \( s[n] \), we can obtain the baseband analog signal by sampling with a digital-to-analog converter (DAC) shown in (2.14):

\[
s_{BB}(t) = \sum_{n=1}^{2N+1} s[n] \cdot \prod_{\tau=1/f_s}^\infty \left( t - \frac{n-1}{f_s} \right)
\]

(2.14)

where \( \prod_{\tau=1/f_s}^\infty \left( t - \frac{n-1}{f_s} \right) \) is the square pulse function with a duration of \( 1/f_s \) that approximates the DAC’s sampling window[18]. The baseband signal, \( s_{BB} \), is up-converted by mixing with the carrier frequency \( f_c \) and then transmitted. The received signal will experience various scaling (range attenuation, use of amplifiers, etc) and additive (noise, clutter, etc) effects as it passes through the channel. Reception of the of the signal via I/Q detection can be modelled for ad-hoc, asynchronous communications using (2.15) [19]

\[
\begin{align*}
    s_{RX,I}(t) &= \frac{1}{2} \sum_{m=1}^{M} a_m s_{TX} \cos \left( \frac{2\pi f_c \Delta \phi_m - \Delta \phi}{\Theta_m} \right) + w_I(t) \\
    s_{RX,Q}(t) &= \frac{1}{2} \sum_{m=1}^{M} a_m s_{TX} \sin \left( \frac{2\pi f_c \Delta \phi_m - \Delta \phi}{\Theta_m} \right) + w_Q(t)
\end{align*}
\]

(2.15)

where \( a_m \) is the percent amplitude of the of the originally transmitted signal and \( w_{I,Q} \) are representations of the total additive noise and other disturbances. Both \( a_m \) and \( w_{I,Q} \) are functions of frequency, allowing for the time indexing to be dropped in preference of their respective \( k^{th} \) frequency after analog-to-digital conversion (ADC) the transformation of the signal into the Frequency Domain. Therefore, after performing the FFT on (2.15) we obtain

\[
\begin{align*}
    \hat{S}_{RX,I}(k) &= S(k) \sum_{m=1}^{M} a_m(k) e^{-j2\pi \frac{k f_s}{2N+1} t_{m}} \cos (\Theta_m) + w_{k,I} \\
    \hat{S}_{RX,Q}(k) &= S(k) \sum_{m=1}^{M} a_m(k) e^{-j2\pi \frac{k f_s}{2N+1} t_{m}} \sin (\Theta_m) + w_{k,Q}
\end{align*}
\]

(2.16)

which we can then combine the I/Q components using (2.17) to obtain the respective \( k^{th} \)
magnitude and phase.

\[
\hat{S}_{IQ}(k) = \sqrt{\hat{S}_I(k)^2 + \hat{S}_Q(k)^2}
\]

\[
\angle \hat{S}_{IQ}(k) = \arctan \left( \frac{\hat{S}_Q}{\hat{S}_I} \right)
\]

Thus we obtain the final combined I/Q frequency domain vector of the received signal:

\[
\hat{S}_{IQ} = [\hat{S}_{IQ}(N), ..., \hat{S}_{IQ}(1), 0, \hat{S}_{IQ}(1), ..., \hat{S}_{IQ}(N)]
\]

(2.18)

2.1.4 Ultra-Wideband Signals

Like OFDM, ultra-wideband (UWB) signals are not a relatively new concept, rather, advances in semi-conductor technology during the 1990s allowing for fast switching speeds and microprocessors have made it practical for commercial use and therefore, an area of research interest. It is defined as a signal occupying a bandwidth that is either greater than 500 MHz or over 25% of its center frequency. Unlicensed use of UWB signalling is allowed in the 3.1-10.6 GHz range under the condition of strict power emission limitations (on the order of 0.5 mW) so as to avoid interfering with the multitude of other devices in the systems bandwidth [20]. However, this power limitation, coupled with the large bandwidth to transmit on, make UWB signals excellent for use with OFDM and where a low probability of detection (LPD) is desired.

UWB signals have already been studied for communications for a number of reasons:

- Large bandwidth allows for high datarates [22].
• Short pulse length good for robust multipath diversity [23].
• Availability of small form-factor equipment with low power consumption [24].

Furthermore, UWB signals are inherently resistant to jamming attempts [25] and possess high penetration capabilities due to energy being distributed over many frequencies [26]. Combining this with a small time resolution allows for high-precision ranging, making it very powerful for radar systems as well [27].

### 2.2 Properties of Statistical Distributions

In this thesis, it is proposed that we transmit info embedded into a random distribution. The specifics of how this is implemented is covered in Chapter 3, here, we briefly review the key properties of random processes that enable us to embed information within them.

All random processes have three common measures: mean, median, and variance. Of particular interest to this thesis are the properties of mean and variance. Given \( n \) samples, \( \{X_1, X_2, ..., X_n\} \) from a distribution, we can compute the arithmetic sample mean \( E[X] \) using (2.19).

\[
E[X] = \frac{\sum_{i=1}^{n} X_i}{n} \tag{2.19}
\]

If the particular distribution \( \{X\} \) was drawn from is known, along with relevant parameters, it is possible to analytically compute the mean of a sufficiently large number of samples. However, as discussed previously, in wireless communications transmitted signals are often mixed with noise and other signals in the same spectrum, and lose energy as they propagate through space. This results in distortion of the single values. Thankfully, mixing with noise can be represented as an addition of samples from another random distribution, and attenuation losses can be represented as a simple scalar change to the samples of a transmitted distribution. Using (2.20) we can extract the desired mean \( E[X] \) after propagation through a noisy channel.

\[
E[cX + Y] = cE[X] + E[Y] \tag{2.20}
\]

Assuming we can know the approximate value of scalar \( c \) and have sufficient samples of \( E[cX + Y] \) and \( E[Y] \) to obtain an accurate estimate of both, we can rearrange (2.20) to get

\[
E[X] = \frac{E[cX + Y] - E[Y]}{c} \tag{2.21}
\]

Variance has a similar set of properties, allowing for it to be utilized in a similar manner. Given a set of \( n \) samples, \( \{X_1, X_2, ..., X_n\} \) from a distribution, we can compute the sample variance \( \sigma^2[X] \) using (2.22).

\[
\sigma^2[X] = \frac{\sum_{i=1}^{n} (X_i - E[X])^2}{n - 1} \tag{2.22}
\]
To extract the desired variance of multiple mixed and scaled distributions we can use (2.23).

\[
\sigma^2[cX + Y] = c^2\sigma^2[X] + \sigma^2[Y] + \text{cov}(X, Y)
\]

(2.23)

where \(\text{cov}(X, Y)\) is the covariance between the samples of \(X\) and the samples of \(Y\). Therefore, if we can know the approximate value of the scalar \(c\), have a sufficient number of samples to estimate \(\sigma^2[cX + Y]\) and \(\sigma^2[Y]\), and can assume \(\text{cov}(X, Y)\) is negligible (i.e. \(X\) and \(Y\) are pulled from unrelated distributions), we can extract the desired variance using (2.24).

\[
\sigma^2[X] = \frac{\sigma^2[cX + Y] - \sigma^2[Y]}{c^2}
\]

(2.24)

(Note that \(\text{cov}(X, Y)\) was left out of (2.24) because it is assumed negligible.)

### 2.2.1 Weibull Distribution

The principle distribution this thesis focuses on is the Weibull distribution. The Weibull distribution is a generalized form of the exponential distribution, which is commonly used to model the time until some event (e.g. arrival rates in networking queues) [28]. Though the Rayleigh distribution is also explored, the Weibull distribution is of primary interest for two reasons: First, it is characterized by two parameters, scale \(\lambda\) and shape \(k\). Second, it is reasonably well-known, and thus can be used to demonstrate the ability of random sequence encoding with common probability functions.

The Weibull distribution’s probability function is and is shown in (2.25).

\[
f(x; \lambda, k) = \begin{cases} 
\frac{k}{\lambda} \left(\frac{x}{\lambda}\right)^{k-1} e^{(-x/\lambda)^k} & x \geq 0 \\
0 & x < 0
\end{cases}
\]

(2.25)

(2.25) is the probability density function of the Weibull distribution, characterized by the scale parameter, \(\lambda\), and the shape parameter, \(k\), as can be seen in Figure 2.5.

For given values of \(\lambda\) and \(k\), we can compute the expected mean and variance of samples drawn from the distribution using (2.26) and (2.27), where \(\Gamma\) is the Gamma Function.

\[
E[X] = \lambda \Gamma\left(1 + \frac{1}{k}\right)
\]

(2.26)

\[
\sigma^2[X] = \lambda^2 \left[\Gamma\left(1 + \frac{2}{k}\right) - \left(\Gamma\left(1 + \frac{1}{k}\right)\right)^2\right]
\]

(2.27)

This two-parameter characterization is useful to us in that it adds another layer of security in keeping communications hidden from potential interceptors by allowing for one parameter to be shared among friendly platforms beforehand. We will see that this additional knowledge improves the accuracy with which signals can be received while simultaneously increasing the penalization of intercepted signals.
2.2.2 Rayleigh Distribution

The second distribution of interest in this thesis is the Rayleigh Distribution. It was chosen because it is characterized by a single parameter, the scale, $b$, and therefore can be used to contrast the two-parameter Weibull distribution. The Rayleigh distribution is also a special case of the Weibull distribution, where (2.28) with scale, $b$, is equivalent to (2.25) with scale $\lambda = \sqrt{2}b$ and $k = 2$. Rayleigh distributions are commonly used to model dense scatters reaching a receiver via multiple paths [28] and as a noise model for radar simulations.

$$f(x; b) = \frac{x}{b^2} e^{\frac{x^2}{2b^2}}$$  \hspace{1cm} (2.28)

(2.28) is the PDF for a Rayleigh distribution, where $x \geq 0$. We can analytically determine the mean and variance of a Rayleigh distribution with scale, $b$, using (2.29) and (2.30) respectively.

$$E[X] = b\sqrt{\frac{\pi}{2}}$$  \hspace{1cm} (2.29)

$$\sigma^2[X] = \frac{4 - \pi}{2}b^2$$  \hspace{1cm} (2.30)
2.3 Radar Functionality and Modes of Operation

2.3.1 The Radar Equation

While the main focus of this thesis is on a communications scheme, it is intended for use as a dual communications-radar signal. Therefore, we will briefly review some fundamental radar principles along with our underlying reasoning for why we believe random sequence encoding can be used for the fusion of radar and communications signals. Standing for RAdio Detection And Ranging, all radar utilizes the reflection of radio waves and the very familiar distance equation Equation 2.31.

\[ d = |\vec{v}|t \]  \hspace{1cm} (2.31)

Where \( d \) is the distance, \( |\vec{v}| \) is the speed, and \( t \) is the time. Since \( |\vec{v}| \) is simply the speed of light, we can replace it with \( c \), and \( d = 2R \) since the radio waves must travel \( R \) to the target and then \( R \) back to the receiver. Thus, if we can measure the time delay between transmission and reception we can easily determine the range, \( R \), of the target.

\[ R = \frac{ct}{2} \]  \hspace{1cm} (2.32)

Thus, we can see the fundamental difference between a radar and communications system is that in a radar system, the transmitting platform is also the receiving platform. From here, multiple antennas, multiple pulses, and a variety of signal processing techniques can be employed to allow for obtaining more complex information about the target scene than simply detection and range of targets. However, this exceeds the scope of this thesis.
We can arrive at the more classic radar equation by considering the power distributed over the target, the amount of that power ultimately reflected by the target, and the amount of reflected power distributed back over the receiving antenna. Assuming an isotropic, spherical radar pulse, the power will be distributed along the area of a sphere with radius \( R \). Using a directed radar pulse instead of an undirected one will simply scale this power distribution by a gain factor, \( G \). Therefore, we can determine the power density at the target, \( S_t \) to be

\[
S_t = \frac{P_s G}{4\pi R^2}
\]  

(2.33)

where \( P_s \) is the power of the signal sent. The target will have an effective radar cross section, \( \sigma \) (usually determined by other radar data) that will determine how much of signal at the target is reflected. This reflected power, \( P_r \) is once again going to be distributed over a the surface area of a sphere of radius \( R \), and just like the target has an effective radar cross section scaling the power reflected, the antenna will have an effective aperture area, \( A_w \) that scales the received power. This effective aperture area depends on the antenna’s gain and geometric area, and the wavelength used.

\[
A_w = \frac{G\lambda^2}{4\pi K_a}
\]  

(2.34)

Where \( K_a \) is an efficiency factor. Therefore, the power of the signal received can be determined by

\[
P_e = \frac{P_s G}{4\pi R^2} \frac{\sigma}{4\pi R^2} A_w
\]

\[
= \frac{P_s G \sigma A_w}{(4\pi)^2 R^4}
\]

\[
= \frac{P_s G^2 \lambda^2}{(4\pi)^3 R^4}
\]

(2.35)

This received power can then be solved for \( R \) to get the Radar Equation (2.36) [29].

\[
R = \sqrt[4]{\frac{P_s G^2 \lambda^2 \sigma}{P_e (4\pi)^3}}
\]  

(2.36)

### 2.3.2 Range Resolution and Maximum Unambiguous Range

Range resolution and maximum unambiguous range are two aspects of radar that must be considered when designing a radar system and both refer to two forms of ambiguity that can arise.
Range Resolution and Pulse Compression

Range resolution is how small of a range you are able to resolve. Basically, any targets within the range resolution of each other will be interpreted by the radar system as a single target. Fig. 2.7 depicts an example waveform being reflected by two targets in two different scenarios and Fig. 2.8 shows the signal return due to both targets at discrete time steps. The first where the distance between targets is more than \( c \tau^2 \), and the second where the distance is less than \( c \tau^2 \). As can be seen in the example waveform return, the closely spaced targets return a waveform with a single peak just as a single target would.

\[
S_r = \frac{c \tau}{2}
\]  

(2.37)

Figure 2.7: Example of Range Resolution and Target Ambiguity [21]

Figure 2.8: Signal Return from Multiple Targets with Different Spacing over Discrete Time Steps

From Fig. 2.8 we can see that any targets with spacing less than \( \frac{\tau}{2} \) will yield a single return. Therefore we can define the theoretical range resolution as
However, an intrapulse modulation technique known as pulse compression, where the transmitted pulse is modulated in phase or frequency, can allow for better range resolution than seen in (2.37). This modulated pulse is then correlated with the transmitted pulse (typically through matched filtering) and can be used to resolve targets with overlapping returns. Pulse compression gets its name because it is a technique that transmits shorter pulses (temporal compression) more frequently (using fast switching speeds of modern semiconductor technology) to increase the average transmit power. Since each part of the pulse requires a unique frequency, we can then express the range resolution as a function of transmit bandwidth, $BW_{TX}$, instead.

$$S_r = \frac{c}{2BW_{TX}}$$  \hspace{1cm} (2.38)

**Maximum Unambiguous Range**

While range resolution concerns ambiguity of target detection in an area, maximum unambiguous range concerns returns that could refer to multiple ranges. Consider a circular path. Following it for $\pi$ rad and $3\pi$ rad will both end $\pi$ rad away from the starting point. Similarly, a radar system will transmit a pulse once every period, known as the pulse repetition frequency ($PRF$). Referring to (2.31) and substituting $t$ for $1/PRF$, we obtain the maximum unambiguous range of the target. This is illustrated in Fig. 2.9.

$$R_{\text{max}} = \frac{c}{2PRF}$$  \hspace{1cm} (2.39)

However, (2.39) is incomplete, as the time it takes to fully receive a signal must also be considered. Therefore, (2.39) becomes

$$R_{\text{max}} = \frac{c}{2PRF} \left( \frac{1}{PRF} - \tau \right)$$  \hspace{1cm} (2.40)

This maximum unambiguous range can be circumvented by staggering the times the pulse is repeated. By doing so, a collection of points from the ambiguous return can be analyzed to determine its unambiguous location.
2.3.3 The Doppler Effect

What is perhaps the most iconic part of radar is the Doppler effect. The Doppler effect is the shift in perceived frequency due to non-tangential movement of the target with respect to the source. It is an effect seen in all waves and can be conceptualized by visualizing the waves emitted by a source. If the source is stationary, the waves will be equally spaced. However if the source is perceived to be moving, then the wave peaks will appear closer together, or occur more frequently, in the direction of movement. Likewise, they will appear further apart, or occur less frequently, in the opposite direction of movement. This is depicted in Fig. 2.10

![Visualization of Doppler Shift due to a Moving Source](image)

The Doppler frequency, $f_D$, can therefore be used with the transmitted signal frequency, $f_{TX}$ to determine the velocity of the target along the radius of the transmitted/reflected signal, $v_r$. In order to determine the relation between these two properties, we can first consider the phase shift in the case of a stationary transmitter and target. We know that the signal will travel $2R$ m and that every $1/f_{TX}$ s the phase of the signal, $\phi$, will have cycled a full $2\pi$ rad. Therefore, we can determine the received phase from the ratio of the distance traveled $(2R)$ over the wavelength, $\lambda$, of the wave multiplied by $2\pi$.

$$\phi = \frac{(2R)(2\pi)}{\lambda} \tag{2.41}$$

Since velocity is a function of displacement over time, we can define the radial velocity $\vec{v}_r = \frac{dR}{dt}$ and substitute it in for $R$ in (2.41). Because movement towards the source would result in a shift towards higher frequencies, we can then express the change in phase with
respect to time as
\[
\frac{d\phi}{dt} = -\frac{4\vec{v}_r \pi}{\lambda} \tag{2.42}
\]
We can know obtain \( f_D \) by dividing both sides of (2.42) by \( 2\pi \).
\[
f_D = \frac{1}{2\pi} \frac{d\phi}{dt} = \frac{1}{2\pi} -\frac{4\vec{v}_r \pi}{\lambda} = -\frac{2\vec{v}_r}{\lambda} = -\frac{2\vec{v}_r}{c} f_{TX} \tag{2.43}
\]
With this, we can now not only tell if a target is moving, but also obtain it’s exact speed. However, there is a design dilemma that must be addressed here. In (2.39) we saw that the maximum unambiguous range increased as the \( PRF \) decreased. However, \( f_D \) is also subject to ambiguity from overlapping phases. In particular, to be certain of the received \( f_D \) it must be less than \( PRF \). That is
\[
PRF > |f_D| = \frac{2v_r f_{TX}}{c} \tag{2.44}
\]
Rearranged, we can find the velocity from the maximum unambiguous Doppler shift is
\[
v_r = \frac{cPRF}{2f_{TX}} \tag{2.45}
\]
Which is yet further halved if the direction of movement is unknown. While these are functions of both \( f_{TX} \) and \( PRF \), if we assuming the pulse-width of (2.40), \( \tau << PRF \), then we can see that ability of a radar to unambiguously measure range and velocity is dependent on only \( f_{TX} \) by multiplying (2.39) and (2.45). Assuming direction of movement is unknown, we get
\[
R_{max} v_r < \frac{c^2}{8f_{TX}} \tag{2.46}
\]

### 2.3.4 Noise Radar

By always knowing the signal transmitted, we can design any signal to use with a radar system. While many radar systems have used sophisticated, deterministic signals, there is nothing that prevents the use of using noise itself as the radar signal. So long as we have knowledge of the transmitted signal we can determine the time it took to receive the backscatter from a target [31]. One inherent problem that must be considered with this is long processing time necessary for integrating and smoothing data [32]. The authors of [32] successfully addressed this by formulating correlation function estimates to use noise radar for real-time tracking.
For band-limited, Gaussian noise radar, [33] has shown the autocorrelation function can be expressed as

\[ p(\tau) = A_0 \text{sinc}(B\tau)e^{j2\pi f_0 \tau} \]  

(2.47)

where \( \tau \) is a time delay, \( A_0 \) is a constant, \( B \) is the bandwidth, and \( f_0 \) is the center frequency. This Gaussian UWB noise radar was shown to be capable of radar imaging technique known as synthetic aperture radar (SAR) and using SAR to create images of target scenes behind walls. A real-time, through-wall, SAR imaging and micro-Doppler detection of humans is presented in [34].

Noise radar is a frequency modulated radar, which we saw in (2.38) is dependent on transmit bandwidth for range resolution. Thus, noise radar applications requiring high range resolution conventionally require fast analog-to-digital conversion (ADC) as per the Nyquist sampling theorem, stating the sampling rate should be at least twice the highest frequency contained within the signal.

\[ f_s > 2f_{c\text{-max}} \]  

(2.48)

However fast ADCs typically have a low bit-depth, which results in poor dynamic range. Therefore a method for using slower ADCs to enable the use of noise radar in radar applications needing high dynamic range is presented in [35].

Noise radar has been demonstrated to be useful in a large variety of radar’s numerous applications, ranging from atmospheric imaging to ground penetrating radar [36]. In [37], the use of noise radar for space-time adaptive processing is examined and in [38] is used for automobile detection through foliage.

Furthermore, noise radar even has two significant advantages over sophisticated, deterministic signals [39]:

1. Low pulse-to-pulse cross-correlation allows for many systems to operate simultaneously within the same frequency spectrum.

2. Inherently LPI due to many signal options limiting use of efficient interception schemes and LPD in that once detected they are hard to identify.

The low pulse-to-pulse cross-correlation makes it ideal for use in automotive collision avoidance, where many systems need to operate simultaneously which produces a noisy environment that requires high, anti-interference capabilities for reliable use [40]

2.4 Fusion of Radar and Communication Systems

Though there are a multitude of optimizations that can be for radar sensing and various design considerations to account for depending on the desired modes of operation, radar ultimately relies on simply having knowledge of the transmitted signal before processing the received signal. Indeed, there is an entire subject related to “passive radar” sensing, in which the system does not contain an RF transmitter of its own, relying instead on reflections from commercial broadcasts in order to operate at low power [41]. As such, it is no wonder
that a great deal of research has gone into the fusion of radar and communication systems. Furthermore the fusion of radar and communications would also allow for multiple platforms in the same area to share sensing information with each other while continuing to probe their environment. Based on work such seen in [42], having just 3 platforms in the same area could potentially allow for 3D position estimation.

As with all engineering problems, there are trade-offs that must be considered within the context of the target product operation. Therefore, we must first define some potential scenarios that are expected to be encountered before we can assess possible solutions. While we have already mentioned, in passing, some specific examples of the applicability of radar and communication system fusion. Here, we will define two broad scenarios for application and give some more specific examples of where these scenarios may be encountered. We will then give an overview of research on radar-communication system fusion and assess their advantages and disadvantages for the given scenarios before presenting the solution proposed by this thesis.

### 2.4.1 Example Scenarios

In communications, the transmitter and receiver may or may not have line-of-sight (LOS and NLOS, respectively). In less adverse situations, low power, high data rates, and synchronized communications with direct LOS may be desirable. In such situations, fusion of radar and communications can allow for reuse of transmit power and bandwidth allowing for lower power and higher spectral efficiency. Furthermore, this would use a single analog frontened (AFE), allowing for small form-factor requirements to be met. UAVs would therefore be able to reduce their payloads, commercial products such as autonomous cars could reduce costly equipment, allowing for them to be more affordable, and unmanned vehicles for the purpose of disaster relief could conserve power where chances to recharge may be limited.

On the other end of the spectrum, LOS may not be possible, communications may have to be transmitted ad-hoc, not allowing for synchronization and making high data rates difficult to achieve, and a high transmit power may be necessary. Backscatter communications in areas affected by electronic counter measures (ECM) creating a highly adverse environment where the presence of friendly platforms is not guaranteed. In this case a message would need to be broadcast along with the radar waveform ascertain the presence of friendly platforms in the area.

Beyond these considerations, the potential solutions explored below will also be considered in the case of the above scenarios having need for LPI/LPD capabilities.

### 2.4.2 Current State of the Art

A method using TDMA presented in [10], where the system operated in either radar or communications mode at a given time, was discussed in chapter 1. This particular approach
did not present characteristics considered advantageous for LPI/LPD systems, though a time-
division approach like this one could include such capabilities by changing the waveforms
used or incorporating an UWB capable transceiver if desired. The potential for such changes
were seen in a future paper by the authors, in which they were able to reduce the system size
while improving the radar sensing capabilities simply by changing the transceiver topology
[43].

TDMA approaches like this one have an advantage in generally being lower cost, easy to im-
plement, and having minimized mutual interference. However, in order to function properly,
time synchronization of the system is required and they cannot operate in both modes simulta-
nously. Thus, such time-sharing approaches are out of the question for any scenarios that
necessitate the system be capable of radar sensing and communications at all times.

Beyond TDMA, a method for system fusion using polarization differentiation is proposed
in [44] and several methods for joint waveforms have been proposed over the past decade
that do allow for simultaneous sensing and communication at all times via a shared AFE.
These methods can be broadly divided into single-carrier and multi-carrier systems, each
naturally coming with their own advantages and disadvantages as discussed in [45]. In
particular, the joint waveform approach has the advantage of being efficient in both time
and hardware.

In the single-carrier approach, pulsed, DSSS is typically used for the radar side, while variety
of communication schemes have been employed such as binary phase-shift keying and pulse
amplitude modulation. While the single-carrier approach offers resistance to interference
and jamming, LPI characteristics, and secure communications, it suffers from low spectral
efficiency and inefficient and complicated radar signal processing with poor Doppler tolerance
[45].

A variety of multi-carrier joint waveform approaches have been proposed in recent years.
[46] proposes use of an OFDM signal as a joint waveform and shows how the OFDM signal
could be exploited to circumvent the Doppler dilemma mentioned in subsection 2.3.3. In
[47], an OFDM approach is also explored, this time with UWB capabilities which help aid
in achieving LPI/LPD characteristics. Again, using OFDM signals, [48] used digital audio
broadcasting (DAB) and digital video broadcasting–terrestrial (DVB-T) for communications,
making use of passive radar techniques discussed in [49]. In [50] an OFDM joint waveform
for vehicular applications such as adaptive cruise control and lane change assistance is dis-
cussed.

Most of the works above have not shown much concern for LPI/LPD characteristics and
jamming and interference resistance; opting instead for radar and communication schemes
that are easier to implement. Since the change from these implementations to schemes
possessing LPI/LPD characteristics jamming and interference resistance should be feasible
with changes to system topology and waveform design, we can look at some methods that
have been proposed specifically for LPI/LPD characteristics and jamming resistance.

Perhaps most desirable when designing a LPI/LPD signal is including some form of random-
ness. This is easy to understand, since blending into surrounding noise is an intuitive way
of avoiding signal detection altogether, and a signal that is randomly changing is difficult to
identify [51]. Similarly, randomized signals offer resistance to jamming and interference due to difficulty to consistently have a high cross-correlation with a signal changing in an unpredictable manner. This is demonstrated in [52], where the authors use a phase-perturbed LFM chirp is used as a counter-measure to jamming. [44] used a randomly generated signal that was both horizontally and vertically polarized. The horizontally polarized signal was left as is, while the vertically polarized signal was mixed with a BPSK-controlled LO to embed data onto the signal. In [53], an UWB noise-OFDM radar has its spectrum adaptive nullification of carriers so that a narrowband communications signal can be embedded within. And in [54] samples of a random distribution were added to the communication signal amplitudes and removed by having knowledge of the mean of the distribution used.

2.4.3 Random Sequence Encoding

In all of the above methods, we fall short of true fusion of radar and communications systems. TDMA simply shares one system for two separate purposes. This makes inefficient use of time and allows for potentially missing information due to the ”wrong” system being active at the time of arrival. Meanwhile there is a very common theme in the joint waveform approach, and that is in the embedding of information within the radar signal. This results in less efficient spectrum usage and power consumption. Therefore, in the following chapter we propose a method for reusing the same exact signal for both radar and communication purposes by making use of noise radar concepts and encoding data onto the parameters of the random distribution used for sampling rather than a parameter of the transmitted EM wave itself.

We looked at the extensive use and applicability of radar using noise-waveforms seen in subsection 2.3.4. Therefore we believe that by separating the communications information from the parameters of the EM wave itself and demonstrating the feasibility of using a random waveform for communications, we will be able to extend this for use in a flexible, radar communications system with true fusion of the signals.
Chapter 3

Random Sequence Encoding

Traditional signal encoding is fairly straight-forward. The simplest case being amplitude modulation, where a signal voltage of $+A$ represents a binary ”1” while a voltage of either 0 or $-A$ represents a binary ”0”. In phase modulation and frequency modulation, the basic premise remains largely unchanged: A given phase or frequency is known to be a binary ”1”, while a different phase or frequency is known to be a binary ”0”. In each of the above modulation schemes, one can increase the number of symbols discernible in a signal from $2^1$ to $2^n$ with various advantages and disadvantages (namely, increased data-rate at the cost of increased BER). However, in all cases it is clear to anyone observing the signal that something more meaningful than noise is being transmitted. Encryption can be used to obscure the exact meaning of the signal from unintended recipients, but hiding the fact that the signal exists and is transmitting info is much more difficult. Sub-carriers also typically need to be determined between TX and RX in advance, or you risk missing transmissions or require multiple transmissions of the same data. You might try to include a method of randomizing sub-carrier values by adding a samples from a random distribution to each of your sub-carrier weights, however, a sophisticated program might be able to determine that there is a signal there after observing multiple signals. By using properties of the random processes discussed in Chapter 2, we can produce a signal that appears entirely random to all conventional communication methods.

3.1 Simulation Method

For the simulation portion of this thesis, Weibull and Rayleigh distributions were used. Data for the Weibull distribution was encoded onto the scale parameter, $\lambda$, with a constant shape parameter, $k$. The scale parameter was chosen as the carrier for the data due to the shape parameter being inside of the gamma function of both (2.26) and (2.27), which does not have an inverse. The Rayleigh distribution has only one parameter, $b$, and relies on other information being unknown to interceptors.

To assess the viability of using data encoded onto the parameters of these random distribu-
tions as a communication scheme, two metrics were used:

1. Reliability of communication measured as BER vs SNR.
2. Penalty to platforms intercepting the signal.

The first metric is used to assess whether it is possible to use this communication scheme, while the second metric is used to assess whether it can safely be used as a radar signal without worry of openly broadcasting potentially sensitive data.

### 3.1.1 Encoding Data onto a Weibull Distribution

When simulating random sequence encoding onto a Weibull distribution, we assumed a "worst-case" scenario just short of the interceptor knows everything, detailed as follows:

- They are aware we are transmitting data on the scale parameter of Weibull distribution.
- They are aware there is some upper limit to the scale parameter that can be transmitted (done in part for simplification of the simulation) and know that limit.
- They know how we have how many symbols are being used and how they are being mapped.
- They know how many re-transmissions we are using.
- They know our bandwidth and which sub-carriers we are transmitting on.

Or, more succinctly, the only information they do not possess about our communications signal is the shape parameter being used.

The simulation itself was executed as follows:

A shape parameter ranging from 0.3 to 0.9 was selected to be used for the entire simulation. This range was picked because for $k \geq 1$ the component of (2.26) due to $k$ is approximately 1, making the mean equivalent to the scale parameter, and therefore losing the advantage of having the secret shape parameter. On the other hand, for $k < 0.3$ the shape component of (2.26) becomes very large and creates a need for larger samples than were used for an accurate mean to be obtained.

Four symbols (‘00’, ‘01’, ‘10’, and ‘11’) are defined. These symbols are mapped using both a $2^x$ mapping, where the first symbol is defined as having a scale between 0 and $2^1$, the second $2^1$ to $2^2$, etc, and a linear mapping where each symbol has a total range of 4.

The bitstream to be sent is randomly generated and divided evenly into $n$ symbols. If the bitstream cannot be evenly divided into $n$ symbols, then it is front-padded with zeros until it is divisible by the number of bits in each symbol. Over all trials, a total of one million bits were transmitted for each simulation.

The scale parameter used for each symbol is picked from a normal distribution with a mean equal to the midpoint of that symbol’s mapping. Fig. 3.1 shows example signals in the
frequency domain for each symbol using 32(16?) sub-carriers and a both symbol mappings.

Next, $M$ samples are taken from a Weibull distribution using the previously picked scale parameter, and the constant shape parameter, where $M$ is equal to the number of re-transmissions multiplied by the number of sub-carriers used. To minimize error before transmission, the sample mean (or variance) is checked to ensure it corresponds to the appropriate symbol. If it does not, the distribution will be re-sampled with the same scale parameter up to 1000 times in an attempt to correct this.

The sampled values are scaled by a factor $c$ and then mixed with additive white Gaussian noise, representing the received signal. It is assumed for the simulation that $c$ is perfectly known by the receiver, but the channel noise has a uniformly random error of up to ±10% for each each sub-carrier. This information is then used to begin the demodulating process. Five total cases were considered in demodulating:

1. Perfect reception of the generated signal.
2. Neither $c$ nor the mean or variance of the AWGN are known.
3. Both $c$ nor the mean or variance of the AWGN are known.
4. An eavesdropper is using a Maximum Likelihood Estimator to obtain $\lambda$
5. An eavesdropper comparing the sample mean to the symbol mappings to interpret the signal.

The results of this simulation are seen in 3.2.1 and 3.3.1 below.

### 3.1.2 Encoding Data onto a Rayleigh Distribution

Simulation of random sequence encoding using a Rayleigh distribution follows the assumptions and process used for random sequence encoding using a Weibull distribution outlined in section 3.1.1. However, as one may note from comparing (2.28) to (2.25), the Rayleigh distribution is entirely characterized by one parameter, its scale parameter, $b$. Therefore, a small change must be made to the assumptions for eavesdropper knowledge laid out in 3.1.1, as without a secondary parameter, an eavesdropper would have full knowledge about the communication scheme. The change made for the Rayleigh distribution then, was to assume the eavesdropper does not know the precise symbol mapping used, only that there is some upper limit to the scale parameter used.
3.2 Communication Reliability

3.2.1 Weibull Distribution BER vs SNR

For the Weibull distribution, we can see from Fig. 3.3 that it is most reliable using $2^x$ data mapping when using sample variance to demodulate the scale parameter. The linear mapping could potentially be improved by expanding the range of possible scale parameters for each symbol, but this may also be beneficial for eavesdroppers. For both $2^x$ data mapping and linear data mapping the interceptor had a BER between 40% to 50%, rendering their attempts at eavesdropping useless. Using sample mean to demodulate the scale parameter we see in Fig. 3.4 that the $2^x$ data mapping outperforms the linear data mapping in terms of both friendly BER, achieving approximately 2% compared to the 12% using linear data mapping, and in terms of eavesdropper penalization.

Comparing Fig. 3.3 with Fig. 3.4 we see that while the eavesdropper performance remains
about constant regardless of whether sample mean or sample variance is used for demodulation, the actual communications performance improves significantly for the friendly receivers. It is also worth noting that reception using the sample mean for demodulation yields very consistent BER across different SNR as shown in Fig. 3.4.

In both Fig. 3.3 and Fig. 3.4, a shape parameter of $k = 0.5$ is used as it’s performance was found to be best. The simulation used three re-transmissions and 32 sub-carriers, and the
receiver varied from 1 m to 10 m from the transmitter.

One might note the peculiar behavior of the eavesdropper BER improving slightly at lower SNR in Fig. 3.3b, and both Fig. 3.4a and Fig. 3.4b. Noting that this dip begins around the same SNR the friendly reception begins to rise, we believe this to be due to the, somewhat naive, assumption made by the eavesdropper that $k = 1$. This assumption makes it so the sample mean is the scale parameter. However, $k$ is actually 0.5 in these simulations, which corresponds to $E[X] = 2\lambda$ when analytically determining the Weibull mean, and $\sigma^2[X] = 20\lambda$ when analytically determining the Weibull variance. It should also be noted that when symbol errors occurred in friendly reception, they appeared to shift towards the next symbol down (e.g. 01 to 00) more often than the next symbol up. This corresponds to a decreasing sample mean or variance. Thus, our present explanation for the perceived improvement in eavesdropper BER is that while the noise is sufficient to lower the sample mean and cause an error in friendly reception, the eavesdropping platform—which is dividing the sample mean/variance by 1 instead of 2 or 20, respectively—enjoys an improvement in BER from this reduced sample mean/variance. However, the eavesdroppers BER remained above 30% in all cases, which is effectively an unintelligible signal. Therefore, further analysis of this phenomenon was left to future work.

### 3.2.2 Rayleigh Distribution BER vs SNR

In the Rayleigh simulations for BER vs SNR, we were able to obtain a very good $10^{-6}$ BER using $2^2$ symbol mapping. However, eavesdroppers were also able to achieve a BER on the order of $10^{-2}$, limiting its potential to be used in a system requiring low probability of interception (LPI). While the linear symbol mapping performed a bit worse in terms of friendly BER, it still achieved a BER on the order of $10^{-3}$, while the eavesdropper BER was increased to an order of $10^{-1}$. Comparing Fig. 3.5 and Fig. 3.6, we see that just as in the case of the Weibull distribution, reception using the sample mean to demodulate the signal yields a very consistent BER while variance requires a strong SNR. In Fig. 3.6a, the friendly reception achieved 0% BER down to about -15 dB.
3.3 Interceptor Penalization

3.3.1 Weibull Distribution

Fig. 3.7 and Fig. 3.8 show the BER penalization of eavesdroppers for different combinations of shape parameter and symbol mapping as a the number of sub-carriers and re-transmissions.
used increase. The eavesdroppers utilized two methods of trying intercept and interpret the signal: a maximum likelihood estimate (shown in red) based on the raw, received samples and a guess at the shape parameter (shown in pink) based off of the known maximum scale parameter. The friendly reception was also compared to the error due to generation of the signal at the transmitter, however, the error produced at the transmitter upon generation of the signal can be reduced by adding a check that even for smaller sample sizes, the scale parameter found from the sample matches the symbol to be sent.

For Weibull, we again found that a $2^x$ data mapping performs better than the linear data map used, achieving a difference in BER of almost 40% compared to the approximately 35% difference achieved by the linear data mapping. In both cases, the friendly platform has a significant advantage over an eavesdropper. One disadvantage to use of the Weibull distribution is the limited range that can reasonably be used for the shape parameter. Noting the gamma function term in (2.27) and particularly in (2.26), we can see that as the shape parameter approaches 0, the gamma function term begins to rapidly grow and overwhelm
the scale parameter. Similarly, when the shape parameter begins to exceed 1, the gamma term approaches 0 in the case of (2.27) and 1 in the case of (2.26). Therefore, due to the limited range of viable shape parameters that could be used, an eavesdropper may be able to experimentally guess the shape parameter being used if given enough time and samples.

![Figure 3.8](image.png)

Figure 3.8: Interceptor Penalization using Sample Mean of a Weibull Distribution with Shape Parameter, $k$, and $2^x$ Data Mapping (Top) and Linear Data Mapping (Bottom)

### 3.3.2 Rayleigh Distribution

For the Rayleigh distribution, the overall penalization to eavesdroppers is not as using a Weibull distribution. However, while it may be easier to guess what the symbol mapping may look like than estimate the Weibull scale parameter without a known shape parameter or only an estimated shape parameter, we see that with linear symbol mapping we can achieve a difference in BER of about 15% to 20%. While using the sample mean to demodulate the
signal yields a smaller difference in BER than using sample variance, it also produces more reliable communications.

![Figure 3.9: Interceptor Penalization using Sample Variance of Rayleigh Distribution](image)

(a) $2^x$ Data Mapping  
(b) Linear Data Mapping

Overall, the Weibull distribution tended to perform better in terms of eavesdropper penalization, but worse in regards to overall communication effectiveness. The Rayleigh distribution, on the other hand, was able to achieve very low BER, staying under $10^{-3}$ down to -10 dB for both data maps when using the sample mean for demodulation, but did not penalize...
eavesdroppers as harshly as using the Weibull distribution did. In both cases, we were able to achieve communication BER on the order of $10^{-2}$ and maintain an eavesdropper penalization of 10% or greater with appropriate choice of parameters, symbol mapping, and decoding via sample mean.
Chapter 4

Experimental Validation

4.1 Hardware Implementation

The analog front-end (AFE) of the SDRCS system used to implement this research is shown in Fig. 4.1. Matlab was used to define the signal that was sent to a PXDAC4800 DAC capable of transmitting on a 600 MHz BW, though its user manual recommends using a BW of only 590 MHz. Furthermore, it was experimentally determined that sub-carriers above 525 MHz suffered from poor reception, which we believed was due to switching noise. A 25 dBm local oscillator (LO) was used to upconvert the baseband signal, $s_{BB}(t)$, to the intermediate frequency (IF) centered at 7.5 GHz. The received signal was amplified by two, 24 dB ultra-low noise amplifiers (ULNA) and a Tektronix MSO4104 oscilloscope was used as an ADC to receive the I and Q signals.

Figure 4.1: Block Diagram of Miami University’s SDRCS AFE
### Table 4.1: System Specifications

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseband signal bandwidth</td>
<td>600</td>
<td>MHz</td>
</tr>
<tr>
<td>Transmitted signal bandwidth</td>
<td>6.9...8.1</td>
<td>GHz</td>
</tr>
<tr>
<td>Transmitter DAC Sampling</td>
<td>1.2</td>
<td>Gs/s</td>
</tr>
<tr>
<td>Receiver ADC Sampling</td>
<td>5.0</td>
<td>Gs/s</td>
</tr>
<tr>
<td>Pulse Width</td>
<td>Variable</td>
<td>–</td>
</tr>
<tr>
<td>Max available CW transmit power (approximately)</td>
<td>316*</td>
<td>mW</td>
</tr>
<tr>
<td>Experimental radar range</td>
<td>1...5*</td>
<td>meters</td>
</tr>
<tr>
<td>Experimental range resolution</td>
<td>0.30</td>
<td>meter</td>
</tr>
</tbody>
</table>

Table 4.1 summarizes the general specifications and capabilities of the radar system used. Because of the differing DAC and ADC sampling speeds, received signals were downsampled in the time domain to match the transmit sampling speed before further signal processing. The radar range listed is from an older hardware setup that used a less powerful local oscillator. Using the new LO, a max indoor range of 10 m is expected to be possible. Finally, up to 28 sub-carriers were used for signalling. While more is presumably possible, we saw greater inter-carrier interference than expected and decided to stop at 28.

### 4.2 Channel State Information

While idealized simulations can assume perfect knowledge of the channel, any practical implementation will need to consider how best to accommodate for how the channel affects the propagating signal. This information can be divided into two groups: instantaneous and statistical channel state information. Statistical channel state information is typically more valuable in transmission optimization, and, since this thesis focuses on a simple proof-of-concept for random sequence encoding, instantaneous channel state information is of primary concern. This information can be considered as an impulse response, or transfer function of the channel.

#### 4.2.1 Estimating Channel Transfer Function

Any signal output, \( Y \), from a channel \( H \), given an input, \( X \), can be modeled as in Equation 4.1

\[
Y =HX + N
\]

where \( N \) is the channel noise. If we know what the transmitted signal was and assume \( N \) is already estimated or negligible, then we can easily obtain an estimate of the channel transfer function, \( \hat{H} \), by solving (4.2), where the division shown is an element-wise division.

\[
\hat{H} = \frac{Y - \hat{N}}{X}
\]
There are multiple ways this estimation can be obtained, but in this thesis we used a simple block-type channel estimation [55]. This was implemented by assigning a weight of 10 to all sub-carriers every fifth transmission. Details considering more practical systems where systems would need to synchronize in some form to know when a pilot tone was sent were considered beyond the scope of this thesis and receiving systems were assumed to simply know when a pilot tone was transmitted.

4.2.2 Estimating Channel Noise

Channel noise was estimated before each pilot tone was sent by sampling the channel \( n \) times and averaging the noise observed at each frequency. This noise estimate was then also used for determining the noise threshold for received signals.

4.3 Wireless Line-of-Sight Tests

Due to the nature of random sequence encoding relying on a randomly generated signal from a Weibull distribution, and this potentially leading to an inherently disproportionate number bit errors arising from a particular symbol, 100 symbols of each symbol type were transmitted each experiment to ensure parameters were set in a way that yielded similar BER for all symbols. Thus, each line-of-sight (LoS) experiment consisted of a total of 400 symbols, or 800 bits transmitted in total.

In addition, prior to transmission, each signal was checked to ensure the \( n \) samples generated did in fact yield the proper symbol. If not, the \( n \) samples were regenerated. If after re-sampling the distribution 50 times the \( n \) samples still did not correlate to the correct symbol, a warning was issued and the signal was transmitted so that the experiment could continue. (Note: this differs from in simulation where no such pre-check was made before transmission.)

While the DAC used is capable of transmitting a bandwidth of 590 MHz, it was found that reception of sub-carriers at frequencies above 525 MHz was unreliable for the purposes of this research and were therefore dropped. Thus, the following experiments divided the 600 MHz BW into 8, 16, and 32 sub-carriers, but only used 7, 14, and 28 sub-carriers respectively. Additionally, the DAC allowed an input signal magnitude of \( 2^{15} \), and truncated higher valued signals to the peak output voltage, so parameter considerations were made to ensure a high probability of sending untruncated signals and prevent loss of information at the transmitter.

Due to time constraints, experimental results focused on using the Weibull distribution. All assumptions made about friendly vs eavesdropper knowledge in chapter 3 are the same in the following experiments.
4.3.1 SNR vs BER

For the SNR vs BER experiments, 14 sub-carriers with 3 re-transmissions were used. For a high SNR (> 10 dB), typically achieved a BER of 1-2% for demodulation using sample mean and 10-12% for demodulation using sample variance. These values are also comparable to those seen in simulation, where 32 sub-carriers were used instead of 14. For low SNR (< 5 dB), the BER increased to between 5.8-33% when using the sample mean for demodulation and 26-36% when using the sample variance for demodulation.

On average, intercepting platforms suffer a 16% penalty to BER when demodulating using the sample mean. However, for SNR under 5 dB, this penalty decreases by about 4% on average. While overall BER increases when demodulating using the sample variance, it appears intercepting platforms suffer a greater penalty on average. The average difference here is 24%, and when considering only BER for SNR under 5 dB, 15%.

![Figure 4.2: BER vs SNR using a Weibull Distribution for RSE](image)

The rise in BER occurred at a much earlier SNR than in simulation. This can possibly be attributed to a combination of three points: The simulation used 32 sub-carriers rather than 14, and the error rate is expected to improve with more samples. The simulation also assumed an ideal estimate of the scaling parameter, $c$, from (2.26) and (2.27), while the experiment was using a non-ideal estimate of the values in $H$. Lastly, the simulation was able to use the same symbol mapping for all SNR values, while the experiment required compression of the symbol map as discussed below.

When examining Figure 4.2, there are 3 different "tests" listed. This was due to physical constraints of cable length limiting the distance the RX antenna could be placed from the TX antenna. In test 1, the scale parameter could range from [0,2); [2,8); [8,16); [16,32) (corresponding to symbols '00', '01', '10', and '11' respectively). However, in order to achieve
SNRs below 10 dB, \([0,2); [2,6); [6,12); [12,20)\) for test 2, and \([0,1); [1,4); [4,9); [9,15)\) for test 3. While the lower maximum scale value resulted in overall lower sub-carrier weights, it also compressed the room for error in reception. Since this error is expected to increase as SNR decreases, it is likely that careful selection of these parameters can allow for better performance at smaller SNR.

### 4.3.2 Interceptor Penalization

A closer look at interceptor penalization with varying shape, number of sub-carriers used, and number of re-transmissions is shown in Figure 4.3. All tests were performed over a range of 2 meters.

![Graphs showing interceptor penalization](image)

(a) \(k = 0.3\)  
(b) \(k = 0.5\)  
(c) \(k = 0.9\)

![Graphs showing interceptor penalization](image)

(d) \(k = 0.3\)  
(e) \(k = 0.5\)  
(f) \(k = 0.9\)

Figure 4.3: Interceptor Penalization using Sample Mean (Top) and Sample Variance (Bottom) of a Weibull Distribution with Shape Parameter, \(k\), 2\(^x\) Data Mapping.

Here, the simulation results showing a good advantage when using sample mean for demodulation are supported further, with the exception again being for \(k = 0.9\). In these experi-
ments, we saw a penalization of about 21% for 1 re-transmission, 36% for 3 re-transmissions, and 31% for 5-re-transmissions in Figure 4.3a compared to the MLE method and even higher for an demodulation without knowledge of the shape parameter $k$. For Figure 4.3b, we saw a penalization of about 13% for 1 re-transmission, 23% for 3 re-transmissions, and 18% for 5-re-transmissions. For Figure 4.3c, we only saw a penalization of 5% for 1 re-transmission, 3% for 3 re-transmissions, and 9% for 5 re-transmissions compared to an MLE method of interception. However, there was no penalty inflicted for not knowing the shape parameter, as it’s contribution to the sample mean is approximately 1 for $k = 0.9$.

However, when using the variance we only see a slight penalty in Figure 4.3e for 3 and 5 re-transmissions (12% and 8% respectively). Though even in simulation, the variance-based demodulation required a larger sample size to perform well.

### 4.4 Experimental and Simulation Comparison

Smaller data sizes due to the time constraints imposed by long experimental run times, as well as other non-ideal conditions that are easy to ignore in simulations yielded less smooth results from experimental data. Despite this, however, the general trend of the experimental results, especially that of eavesdropper penalization when using sample mean for demodulation, were similar. Smaller $k$ values resulted in a larger penalization in both simulation and experiment, and aside from a few exceptions, the BER was fairly consistent overall.

The significantly worse performance from use of the sample variance for demodulation could perhaps be attributed to a co-variance with added noise and other interference being less negligible than it was assumed to be. However, since both sample mean and sample variance can be used to estimate the parameter data is encoded on, so long as one demodulation method works we can utilize random sequence encoding for communications.
Chapter 5

Conclusions and Future Work

5.1 Conclusions

This thesis proposed a novel approach to achieving an LPI/LPD communications scheme for an OFDM, UWB radar-communications fused system through simulation and experiment.

The mathematical justification for using parameters of a random distribution was shown along with the background for how this can be accomplished using an OFDM system. Use of random distribution expected value properties were seen to in general perform better than use of variance properties. In simulation it was shown how the 2-parameter Weibull distribution had some advantages over the 1-parameter Rayleigh distribution and that both distributions were able to maintain a distinct penalty of at least 30% and 15% in a "worst-case" scenario, respectively. The Weibull distribution was further explored in wireless, line-of-sight experiments and shown to be maintain a penalty of about 10% or more for not having knowledge of the shape parameter used.

This thesis has demonstrated the potential viability of RSE as an encoding scheme for LPI/LPD communications fused with a noise-radar system. By doing so, we can reduce spectral crowding while increasing spectral efficiency.

5.2 Future Work

While this thesis has focused on the proof-of-concept for RSE, future work must still show it can be effectively fused to a radar system. In addition, methods for optimization and improving upon the BER, particularly at low SNR and in adverse conditions will be essential to practical implementation in an ad-hoc, asynchronous system.
5.2.1 Radar Fusion/SAR Imaging

Demonstration of communication symbols being used for radar while still effectively transmitting information to other platforms in the area could show the potential of multiple platforms being used for 3D imaging of an area.

5.2.2 Improving BER vs SNR

In this thesis, use of sample mean and sample variance were only considered separately. However, using a combination of both sample mean and sample variance could offer a better BER than either one alone. Characterizing the strengths and weaknesses of both approaches could lend insight into how this could be implemented.

5.2.3 Characteristics of Optimal Distributions

Further exploration and examination of random process characteristics that are ideal for RSE (e.g. multiple parameters being necessary for characterization) could help inspire the creation of Cumulative Distribution Functions tailored for use in RSE capable systems. While this thesis focused on two, well-known distributions, most random distributions commonly seen and used today are due to their frequent occurrence in natural and/or industrial processes. Therefore, it is quite possible that none of these random distributions are optimal for RSE.

5.2.4 Differentiating Radar Backscatter from Friendly Communications

It is possible that a platform utilizing RSE will receive backscatter from its own radar mixed with another RSE-capable platform’s communications signal. Therefore, a method of differentiating the two signals from each other will be necessary.
Appendix

A. Weibull Simulation Code

```matlab
function RSE_Sim_wbl(bitstream, varargin)

%% Simulation for communication via Weibull distribution via scale parameter
% Daniel Kellett – Miami University ECE Dept.
% 9/5/2016
% Updated 4/12/2017
%% Optional Arguments

maxargin = 11; % maximum number of optional input arguments
numvarargs = length(varargin);
% check number of optional inputs
if numvarargs > maxargin
    error('userFunction:sim_wbl_v4:TooManyInputs', ...
             'requires at most %i inputs', maxargin);
end
% Set default optional input arguments
% in this case upper_limit, division size, curve it follows
% R N r k symDiv curve mu sigma SNR useVar
optargs = {3, 32, 2, 0.5, 1, 'pow2', 0, 5, 'mean', true, true};
for (i = 1:numvarargs) % overwrite defaults with args
    if isempty(varargin{i})
        optargs{i}=varargin{i};
    end
end
% Place optional input arguments into memorable variable names for use in
% user function.
if isempty(bitstream)
    bitstream = randi([0,1], [1,100]);
end
```
[R, N, r, shape, symDiv, boundsType, mu, sigma, i, SNR, useVar] = optargs{::};

% close all; clc; clear;
% add path to user–functions
addpath(’D:\Documents\Research\Code\helperfiles’);

% useful functions
% match a symbol vector with a specified symbol type, summing the total number of correct bits.
symMatch = @(symVec, symbol) sum( sum( bsxfun (@eq, symVec, symbol), 2 ) );
% allow grabbing paranthesis of function output
% (See loren’s art of matlab blog on functional programming with anon func)
paren = @(x, varargin) x(varargin{:});
ceilfix = @(x) ceil(abs(x)).*sign(x); % round away from 0
% Determine bounds for the specified scale parameter range. ==
symSize = 2; % Symbol size to transmit
symDiv = 1;
boundsType = 'pow2';
scale = getScaleBounds(symSize, symDiv, boundsType);
% disp(scale)
% Define control variables ____________________________________
useVar = true;
T = 1000; % Number of trials to average over.
c = 1/(r^2); % Range attenuation
dist = 'wbl';
% Define binary file path to save data. ============
% save file path code omitted from appendix

% Shape component for scale calculation =============
if (useVar)
    shapeComponent = gamma(1+(2/shape)) - ( gamma(1+(1/shape)) ) ^2;
else
    shapeComponent = gamma(1+1/shape);
end
% Enemy estimation data______________________________
if (useVar)
    maxScale = scale(4,2)^2 * shapeComponent;
else
    maxScale = scale(4,2) * shapeComponent;
end
%% Reshape the bitstream so that each row corresponds to 1 symbol.
%% Note that the symbol size is given as the reshape rows, and the
%% output
%% is transposed to keep bits in the correct order.
%% order bitstream into symbols
orderedbitstream = reshape(bitstream, [symSize, size(bitstream,2)/symSize])';
%% include all trials
dupBitstream = [orderedbitstream; randi([0,1],[T*length(bitstream)/2,2])];
%% Get and show total number of bits that are going to be
%% transmitted.
bitsTransmitted = size(dupBitstream,1)*size(dupBitstream,2);
fprintf('Transmitting %i bits\n',bitsTransmitted);

fprintf('Using %i re−transmissions.\n',R);
fprintf('Using %i sub−carriers.\n',N);

%% loop through and process all of one type of symbol at a time
%% Generate scale parameter used as a normal distribution centered
%% around
%% the middle of the expected range for that symbol.

txCorrect = 0;
rxCorrect = 0;
rawCorrect = 0;
mleRawCorrect = 0;
sampleCorrect = 0;

lamdaDeviation = [0.01,0.01,0.01,0.01]; % lambda deviation for
each symbol
lambdaShift = [0,0,0,0]; % offset from center
mySyms=[0,0; 0,1; 1,0; 1,1];
countBad=zeros(1,4);
for i = 1:4
    symmat = mySyms(i,:);
id=find(ismember(dupBitstream,symmat,'rows'));
Tsym = size(id,1);
lambda = normrnd(scale(i,3)+lambdaShift(i), lamdaDeviation(i),
[1,Tsym]);

% generate the signal to be transmitted and AWGN
badTX=true; tryCnt=0;

tx=zeros(Tsym,L);

for t=1:Tsym

    while (badTX)
        thisTX = wblrnd(lambda(t),shape,[1,L]);
        if (useVar)
            valTX = sqrt(var(thisTX,[])./shapeComponent);
            symTX = rse_demod(valTX,scale);
        else
            valTX = mean(thisTX,2)./shapeComponent;
            symTX = rse_demod(valTX,scale);
        end
        correct = symMatch(symTX,symmat);
        tryCnt=tryCnt+1;
        if (correct == symSize)
            badTX = false;
        elseif (tryCnt >1000)
            badTX =false;
            countBad(i)=countBad(i)+1;
            warning('Could not generate correct symbol');
        end
    end
    badTX=true;
    tx(t,:)=thisTX;
end

AWGN = normrnd(mu, sigma, [Tsym, L]);
raw_rx = c.*tx;

% Received signal with noise
noisy_rx = AWGN + raw_rx + 0.01*ceilfix(20.*rand(size(AWGN)) -10).*AWGN;

if (SNR)
    rx_SNRDdB = 10.*log10(( (rms(c.*tx,2))./(rms(AWGN,2)) )
    .^2);
    if ~isempty(rx_SNRDdB)
        SNR_SymStats(:,i) = [mean(rx_SNRDdB);median(rx_SNRDdB)
            ;...       max(rx_SNRDdB); min(rx_SNRDdB)];
    end
end

if (useVar)
    % Generated Tx Variance (tx corruption)
smpl_var_txGen = var(tx,0,2);
% Precise sample variance of AWGN Noise

var_noise = var(AWGN,0,2);
% Sample variance of uncorrected signal

smpl_var_noisy = var(noisy_rx,0,2);
% Sample variance of corrected signal

smpl_rx = (smpl_var_noisy - var_noise) / c^2;

% Error during tx generation to decode
estScaleTx = sqrt(smpl_var_txGen ./ shapeComponent);

% noisy to decode
estScaleRaw = sqrt(smpl_var_noisy ./ shapeComponent);
% friendly to decode
estScaleRx = sqrt(smpl_rx ./ shapeComponent);

else

% Generated Tx Variance (tx corruption)
smpl_avg_tx = mean(tx,2);
% Precise sample mean of AWGN Noise
avg_noise = mean(AWGN,2);
% Sample variance of uncorrected signal
smpl_avg_raw = mean(noisy_rx,2);
% Sample variance of corrected signal

smpl_rx = (smpl_avg_raw - avg_noise) / c;
% corrupt tx to decode
estScaleTx = (smpl_avg_tx ./ shapeComponent);
% noisy to decode
estScaleRaw = (smpl_avg_raw ./ shapeComponent);
% friendly to decode
estScaleRx = (smpl_rx ./ shapeComponent);

end

mleEstRaw=zeros(Tsym,1);
parfor t=1:Tsym
    mleEstRaw(t,:)=paren(wblfit(abs(noisy_rx(t,:))'),1);
end

% Estimate for Tx
% Noisy Estimated Rx
txSym = rse_demod(estScaleTx, scale);
txCorrect = txCorrect+symMatch(txSym,symmat);
% Corrected Rx Symbols
% Corrected Rx
rxSym = rse_demod(estScaleRx, scale);
rxCorrect = rxCorrect+symMatch(rxSym,symmat);
%% Noisy Rx symbols

% Noisy Rx
rxSymRaw = rse_demod(estScaleRaw, scale);
rawCorrect = rawCorrect + symMatch(rxSymRaw, symmat);

%% MLE Rx symbols
mleSymRaw = rse_demod(mleEstRaw, scale);
mleRawCorrect = mleRawCorrect + symMatch(mleSymRaw, symmat);

%% Sample Only RX
% Corrected Estimated Rx (good intercept)
rxSample = rse_demod(smpl_rx, scale);
sampleCorrect = sampleCorrect + symMatch(rxSample, symmat);
end

txBER = 1 - txCorrect / bitsTransmitted;
rxBER = 1 - rxCorrect / bitsTransmitted;
rawBER = 1 - rawCorrect / bitsTransmitted;
mleRawBER = 1 - mleRawCorrect / bitsTransmitted;
sampleBER = 1 - sampleCorrect / bitsTransmitted;

fprintf('Bad symbol count: 00: %i, 01: %i, 10: %i, 11: %i\n', countBad);
fprintf('TX BER: %f\nRX BER: %f\nMLE BER: %f\nSMP BER: %f\n', ...
    [txBER, rxBER, mleRawBER, sampleBER] * 100);

%% Write to file
%% data writing for storage and plotting ommitted
B. Rayleigh Simulation Code

```matlab
function RSE_Sim_rayl(bitstream, varargin)

%% Simulation for communication via Weibull distribution via scale parameter
% Daniel Kellett – Miami University ECE Dept.
% 9/5/2016
% Updated 4/12/2017
%% Optional Arguments:
maxargin = 9; % maximum number of optional input arguments
numvarargs = length(varargin);
% check number of optional inputs
if numvarargs > maxargin
    error('userFunction: sim_wbl_v4: TooManyInputs', ...
          'requires at most %i inputs', maxargin);
end
% Set default optional input arguments
% in this case upper_limit, division size, curve it follows
% R  N  r  symDiv  curve  mu  sigma  SNR  useVar
optargs = {3, 32, 2, 4, 'linear', 0, 5, false, false};
for i = 1: numvarargs
    if ~isempty(varargin{i})
        optargs(i) = varargin(i); % overwrite defaults
    end
end
% Place optional input arguments into memorable variable names for use in
% user function.
if isempty(bitstream)
    bitstream = randi([0, 1], [1, 100]);
end
[R, N, r, symDiv, boundsType, mu, sigma, SNR, useVar] = optargs
{:};
% close all; clc; clear;
addpath( 'D:\Documents\Research\Code\helperfiles' );
dist = 'rayl';
% useful functions
% match a symbol vector with a specified symbol type, summing the total
% number of correct bits.
symMatch = @(symVec, symbol) sum( sum( bsxfun(@eq, symVec, symbol), 2 ) );
```

50
% allow grabbing parenthesis of function output
% (See loren’s art of matlab blog on functional programming with anon func)
paren = @(x, varargin) x(varargin{:});
ceilfix = @(x) ceil(abs(x)).*sign(x); % round away from 0
%% Determine bounds for the specified scale parameter range.
symSize = 2; % Symbol size to transmit
b = getScaleBounds(symSize, symDiv, boundsType);

%% Define control variables
SNR = true;
useVar = true;
T = 1000; % Number of trials to average over.
c = 1/(r^2); % Range attenuation
noisy = true; % flag for adding AWGN

%% Define binary file path to save data.
% save file path code omitted from appendix

%% Shape component for scale calculation
if (useVar)
    varTerm = (4−π)/2;
else
    meanTerm = sqrt(pi/2);
end

%% Pad front of bitstream with ’0’ if odd number of bits is being sent.
% TODO: generalize for symbols of size x bits?
if (symSize == 2)
    if ((mod(size(bitstream,2),2))
        bitstream = [0,bitstream];
    end
end

%% Reshape the bitstream so that each row corresponds to 1 symbol.
% Note that the symbol size is given as the reshape rows, and the output
% is transposed to keep bits in the correct order.
% order bitstream into vector of symbols
orderedbitstream = reshape(bitstream, [symSize, size(bitstream,2)/symSize])’;
% include all trials
dupBitstream = [orderedbitstream; randi([0,1],[T*length(bitstream)]]);
Get and show total number of bits that are going to be transmitted.

bitsTransmitted = size(dupBitstream,1) * size(dupBitstream,2);
fprintf('Transmitting %i bits\n', bitsTransmitted);

fprintf('Using %i re-transmissions.\n', R);

fprintf('Using %i sub-carriers.\n', N);

L = N * R; % Total length of transmitted signal.

SNR_SymStats = zeros(4,4);

% parameter deviation for each symbol
paramDeviation = [0.01, 0.01, 0.01, 0.01];

% parameter shift from middle of range
paramShift = [0, 0, 0, 0];

% for total number symbols correct

txCorrect = 0;
rxCorrect = 0;
rawCorrect = 0;
mleRawCorrect = 0;
sampleCorrect = 0;

mySyms = [0, 0; 0, 1; 1, 0; 1, 1];
countBad = zeros(1, 4);

% loop through and process all of one type of symbol at a time.
% Generate scale parameter used as a normal distribution centered around
% the middle of the expected range for that symbol.

for i = 1:4
    symmat = mySyms(i, :);
id = find(ismember(dupBitstream, symmat, 'rows'));
    Tsym = size(id, 1);
    rayl_param = normrnd(b(i, 3) + paramShift(i), paramDeviation(i), [1, Tsym]);
    %% generate the signal to be transmitted and AWGN
    badTX = true; tryCnt = 0;
    tx = zeros(Tsym, L);
    for t = 1:Tsym
        while (badTX)
            thisTX = raylrnd(rayl_param(t), [1, L]);
            if (useVar)
                valTX = sqrt(var(thisTX, [], 2) / varTerm);
            else
                valTX = mean(thisTX, 2) / meanTerm;
            end
            symTX = rse_demod(valTX, b);
            end
        end
        badTX = false;
    end
end

52
symTX = rse_demod(valTX, b);
end

correct = symMatch(symTX, symmat);
tryCnt=tryCnt+1;
if (correct == symSize)
    badTX = false;
elseif (tryCnt>1000)
    badTX =false;
    countBad(i)=countBad(i)+1;
    %warning('Could not generate correct symbol');
end

badTX=true;
tx(t,:)=thisTX;
end

if (noisy)
    AWGN = normrnd(mu, sigma, [Tsym, L]);
else
    AWGN = zeros(Tsym,L);
end

% Received signal with noise
noisy_rx = AWGN + c.*tx + 0.01*ceilfix(20.*rand(size(AWGN))
-10).*AWGN;

%%% get SNR. (See wbl code for commented out extra)
rx_SNRdB = 10.*log10(( (rms(c.*tx,2))./ (rms(AWGN,2)) ).^2);
SNR_SymStats(1:4,i) = [mean(rx_SNRdB);median(rx_SNRdB);...
max(rx_SNRdB);min(rx_SNRdB)];

%%% Initial processing to get estimated scale
if (useVar)
    % Precise sample variance of AWGN Noise
    var_noise = var(AWGN,0,2);
    % Generated Tx Variance (tx corruption)
    smpl_var_tx = var(tx,0,2);
    % Sample variance of uncorrected signal
    smpl_var_raw = var(noisy_rx,0,2);
    % Sample variance of corrected signal
    smpl_rx = (smpl_var_raw - var_noise) / c^2;

    % Solve for spread parameter using known term.
    % corrupt tx to decode
    estScaleTx = sqrt(smpl_var_tx ./ varTerm);
    % noisy to decode
```matlab
estScaleRaw = sqrt(smpl_var_raw ./ varTerm);

% friendly to decode

estScaleRx = sqrt(smpl_rx ./ varTerm);

mleEstRaw = zeros(Tsym,1);
parfor t=1:Tsym
    mleEstRaw(t,:) = paren(raylfit(noisy_rx(t,:)),1);
end

else

% Precise sample variance of AWGN Noise
avg_noise = mean(AWGN,2);
% Generated Tx mean (tx corruption)
smpl_avg_tx = mean(tx,2);
% Sample mean of uncorrected signal
smpl_avg_raw = mean(noisy_rx,2);
% Sample mean of corrected signal
smpl_rx = (smpl_avg_raw - avg_noise) / c;

% Solve for spread parameter using known term
% corrupt tx to decode
estScaleTx = (smpl_avg_tx ./ meanTerm);
% noisy to decode
estScaleRaw = (smpl_avg_raw ./ meanTerm);
% friendly to decode
estScaleRx = (smpl_rx ./ meanTerm);

mleEstRaw = zeros(Tsym,1);
parfor t=1:Tsym
    mleEstRaw(t,:) = paren(raylfit(noisy_rx(t,:)),1);
end
end

% Demodulate and sum correct symbols.
%==Estimate for Tx==
txSym = rse_demod(estScaleTx, b); % Noisy Estimated Rx
txCorrect = txCorrect + symMatch(txSym,symmat); % number correct

%==Noisy Rx symbols==
rxSymRaw = rse_demod(estScaleRaw, b); % Noisy Rx
rawCorrect = rawCorrect + symMatch(rxSymRaw,symmat);

%==Corrected Rx Symbols==
rxSym = rse_demod(estScaleRx, b); % Corrected Rx
rxCorrect = rxCorrect + symMatch(rxSym,symmat);
%==Estimate based off variance (Having knowledge of distribution)==
```
rxSample = rse_demod(smpl_rx,b); % Corrected Estimated Rx
sampleCorrect = sampleCorrect+symMatch(rxSample,symmat);
%==Using MLE (Having knowledge of distribution)==%
mleSymRaw = rse_demod(mlestRaw,b);
mleRawCorrect = mleRawCorrect + symMatch(mleSymRaw,symmat);

end

% BER = 1 − ratio of correct bits to number of bits transmitted
txBER = 1−txCorrect/bitsTransmitted;
rxBER = 1−rxCorrect/bitsTransmitted;
rawBER = 1−rawCorrect/bitsTransmitted;
mleRawBER = 1−mleRawCorrect/bitsTransmitted;
sampleBER = 1−sampleCorrect/bitsTransmitted;

fprintf( ’Bad symbol count: 00: %i, 01: %i, 10: %i, 11: %i
’,
countBad);
fprintf( ’TX BER: %f%%
RX BER: %f%%
MLE BER: %f%%
SMP BER: %f%%
’,
[txBER,rxBER,mleRawBER,sampleBER]*100);

if SNR
    if (any(txBER < 0) || any(rxBER < 0) || any(rawBER < 0) || ...
        any(mleRawBER < 0) || any(sampleBER < 0))
        error( ’NEGATIVE BER’);
    end
end

% data writing for storage and plotting ommitted
C. Weibull Experiment Code

%% -------------------------------- Cleanup previous environments --------------------------------
%%
format compact;  % removes spaces between command window readouts
clc;
%Clear the variables
clear all; close all;

%% ------------------------------ User Setup ------------------------------
%%
% for tracking how long the script takes
startTime=datenum(now);
% Path to useful user–made scripts
addpath ('C:\Users\kelletdw\Documents\MATLAB\kelletdw_research\helperfiles\');
addpath ('C:\Users\kelletdw\Documents\MATLAB\kelletdw_research\helperfiles\HardwareInterfacing\');
addpath ('C:\Users\kelletdw\Documents\MATLAB\kelletdw_research\helperfiles\PostProcessing\');
addpath ('C:\Users\kelletdw\Documents\MATLAB\kelletdw_research\helperfiles\RSE\');

% data save filepath building
% omitted from appendix

% User defined constants
F_DAC = 1.2e9;  % DAC sampling frequency (1.2 GHz)
fs = 5e9;  % Oscilloscope sampling frequency (5 GHz)

MHz = 1e6;
ns = 1e−9;

% user functions
symMatch = @(data, symbol) sum( sum( bsxfun(@eq, data, symbol), 2 ) ) ;
getSNRdB = @(RX, noise) 10.*log10(( (rms(RX))./ (rms(noise))) ).^2)

56
% The following is largely auto-generated code that I have sectioned off into easier to follow user-defined functions.

% ESSENTIAL FOR DAC CONNECTION
lib = addDACPath;
[pHandle, outHandle, outConnect, ppHandle, notfound, warnings] = connectDAC(lib);
offset = 0; % not sure what this does?

% ESSENTIAL FOR DAC CONNECTION
% Ensure DAC is properly configured for what you want to do. This can also be done via the PXDAC4800 Playback Application under the "Hardware Settings" tab at the bottom.
configureDAC(lib, pHandle, 0) % library, handle, voltage, channel

% Connect to Scope
disp('Connecting to scope...');
[interfaceObj, deviceObj] = interfaceScope('C001419');

% Create Signal Specifications

% THIS IS THE START OF USER SPECIFIC CODE

disp('Beginning User Signal Crafting...');

% User-created OFDM Signal
% The number of sub-carriers will be more or less evenly spaced over 600 MHz
% e.g. 8 sub-carriers will begin at ~600/8 MHz, then ~2*600/8 MHz etc.
Nsubbands = 16;
Nactual = Nsubbands*(1 - 1/8); % Nsubbands means all ON
\( N_p = N_{\text{subbands}} \times 2 + 1; \)

channel = 'ch1'; % use ch1, ch2, or both

R = 1; % number of re-transmissions 1, 3, 5
symSize = 2; % number of bits in a symbol
\%numBits = 100; % number of bits to be transmitted per trial
boundsType = 'pow2'; % how symbols are mapped
dist = 'wbl';
methodA = true;

mySyms=[0,0; 0,1; 1,0; 1,1];
totalSyms=100;
symbolsMean = zeros(1,totalSyms*4);
recalibrated=false;
runTime=1,%.75; % amount of time to pause after transmission begins
receiveTime=0,%.75; % amount of time to pause after reception
calibrateTime=0.25,%1; % amount of time to pause for before recalibrating
avg=@(x,y) (x+y)/2;

% originally: 0−2, 2−8, 8−16, 16−32
% also used: 0−2, 2−6, 6−12, 12−20
% also used: 0−1, 1−4, 4−9, 9−15
s00=[0,2]; s01=[s00(2),8]; s10=[s01(2),16]; s11=[s10(2),32];
scale = [s00(1), s00(2), avg(s00(1),s00(2));
        s01(1), s01(2), avg(s01(1),s01(2));
        s10(1), s10(2), avg(s10(1),s10(2));
        s11(1), s11(2), avg(s11(1),s11(2))];

SW\_AMP = 05.0; % time domain amplification factor before DAC transmit
ptf = 15/SW\_AMP; % keep pilot tone consistent with varying SW\_AMP
pilotTone = 10*ptf; % amplitude of signal sent for channel estimation
numRegens = 50; % number of times to attempt to correct a bad signal
trials = 1; % final number of trials this batch
batch = 1; % batch number for larger trials.
ndat = 10000; % Data points from scope.
shape = 0.3; % shape of weibull distribution (0.3, 0.5, 0.9)
shapeCompMean = gamma(1+1/shape); % for using sample mean
shapeCompVar = gamma(1+(2/shape)) − ( gamma(1+(1/shape)) )^2;
% todo: sample variance

\[
\text{lambdaDeviaton} = [0, 0, 0, 0]; \quad \% \text{lambda deviation for each symbol}
\]

\[
\text{lambdaShift} = [0, 0.42, 2.5, 3.5]; \quad \% \text{was } 0, 0.42, 2.5, 7.5 (alt: 0, 0.1, 0.5, 1.0)
\]

\[
\text{expectedFreq} = (600 \times \text{MHz}/ \text{Nsubbands}) \times (1: \text{Nactual})';
\]

\[
\text{Nfft} = 4096; \quad \% \text{FFT bins, based on ndat} \times \text{F_DAC}/\text{fs}
\]

\[
\text{freq_ax} = \text{build_freq}(\text{F_DAC}, \text{Nfft}, 'fs/2', \text{false});
\]

\[
\text{pos_ax} = \text{freq_ax}(1: \text{Nfft}/2+1);
\]

\[
\text{df} = \text{F_DAC}/\text{Nfft};
\]

\[
\text{Eid} = \text{expectedFreq}(1: \text{Nactual})'/\text{df}+1; \quad \% \text{expected id values of subcarriers}
\]

\[
\text{initNoise} = 1; \quad \% \text{number of times to sample noise for thresholding}
\]

\[
\text{recalibrate} = 4; \quad \% \text{how many transmissions before recalibrating channel}
\]

\[
\% \text{More filepath stuff}
\]

\[
\% \text{omitted from appendix}
\]

\[
\% \text{Begin Trials}
\]

\[
\% \text{for } \text{t} = \text{trials} \times (\text{batch} - 1) + 1: \text{trials} \times \text{batch} \quad \% \text{this will pick up trial number}
\]

\[
\% \text{generate bitstream}
\]

\[
\text{mySymbol} = 4; \quad \% 1-00; 2-01; 3-10; 4-11 (from misc code - change later)
\]

\[
\text{bitstream} = \text{reshape}(\text{repmat}(\text{mySyms},[1, \text{totalSyms}])', 1, \text{totalSyms} \times 2^\text{symSize}+1);\]

\[
\text{S} = \text{length} (\text{bitstream})/2;
\]

\[
\text{allNoise} = \text{zeros}(\text{Nactual}, \text{R} \times \text{totalSyms}, 2^\text{symSize});
\]

\[
\text{allRX} = \text{allNoise};
\]

\[
\text{allH} = \text{allNoise};
\]

\[
\text{allPosF} = \text{allNoise};
\]

\[
\text{allThresh} = \text{zeros}(\text{S} \times \text{R}/(2^\text{symSize}), 2^\text{symSize});
\]
% divide into symbols
% initialize noise threshold
noiseThresh = getNoiseThresh(interfaceObj, deviceObj, channel, ...
    fs, F_DAC, Nfft, Eid, 1, initNoise);
[i, q, ~] = receiveOFDM(interfaceObj, deviceObj, channel);
RX = processRaw(i, q, fs, F_DAC, Nfft);
n = RX(Eid);
fprintf(’Noise Threshold is: %f
’, noiseThresh);
% run initial channel estimation
[signal, posF] = genPilotSig(Nsubbands, Nactual, Np, SW_Amp, pilotTone);
subCarIndex = (1:Nactual); % (1:16) means all on.
% Transmit OFDM Signal
disp(’transmitting OFDM signal’)
% runTime = 1; % in seconds (ORIGINAL: 5) this will transmit the signal longer
transmitOFDM(signal, pHandle, offset, lib, runTime);
% Receive OFDM signal using Oscilloscope
disp(’receiving OFDM signal’)
[i, q, ~] = receiveOFDM(interfaceObj, deviceObj, channel);
% End DAC RAM Playback
disp(’End RAM playback’);
endDACPlayback(lib, pHandle);
RX = processRaw(i, q, fs, F_DAC, Nfft);
H = RX(Eid) / posF(subCarIndex); % channel estimate
nTrans = 0;

% begin transmitting signal
s = 1;
index = 0;
lastSym = 1;
snr = [0; 0; 0; 0];
while (s <= S*2)
    mySymbol = matchSymbol(bitstream(s:s+1));
    r = 1;
    thisSampleMean = 0;
    while (r <= R)
        subCarIndex = (1:Nactual); % (1:16) means all on.
        if (nTrans >= recalibrate) % update noise and channel
estimates

```matlab
fprintf(’\nRecalibrating channel...\n’);
pause(calibrateTime);

%noiseThresh = getNoiseThresh(interfaceObj, deviceObj, channel, ...)
%fs, F_DAC, Nfft, Eid, initNoise);

fprintf(’Sampling channel noise...\n’);

[i, q, ~] = receiveOFDM(interfaceObj, deviceObj, channel);
RX = processRaw(i, q, fs, F_DAC, Nfft);
noise = RX(Eid);

fprintf(’Estimating transfer function...\n’);

[signal, posF] = genPilotSig(Nsubbands, N_actual, Np, ...
SW_AM, pilotTone);
recalibrated = true;
else
    % gen TX
    fprintf(’\nCrafting retransmission %i of %i\n for symbol %i of %i symbols.\n’, ...
    r, R, ceil(s/2), S);
    thisSym = bitstream(s:s+1);

    if (isequal(thisSym,[1,1]))
        lambda = scale(4,3)+lambdaShift(4);
    elseif (isequal(thisSym,[1,0]))
        lambda = scale(3,3)+lambdaShift(3);
    elseif (isequal(thisSym,[0,1]))
        lambda = scale(2,3)+lambdaShift(2);
    elseif (isequal(thisSym,[0,0]))
        lambda = scale(1,3)+lambdaShift(1);
    else
        error(’Unknown symbol’);
end
badSignal = true;
tryCnt = 1;
while (badSignal)
    posF(subCarIndex) = wblrnd(lambda, shape, [1, ...
length(subCarIndex)]);
    [signal, xT] = craftSignal(posF, Np, SW_AM, 1);
    badSignal = checkSignal(posF(subCarIndex), ...
    signal, shapeCompMean, scale(mySymbol,:), false);
    if (tryCnt > numRegens)
        warning(’Truncating DAC signal values to
```

61
% Transmit OFDM Signal
fprintf('transmitting OFDM signal\n')
transmitOFDM(signal, pHandle, offset, lib, runTime);

% Receive OFDM signal using Oscilloscope
fprintf('receiving OFDM signal\n')
[i, q, tx] = receiveOFDM(interfaceObj, deviceObj, channel);
pause(receiveTime);
% End DAC RAM Playback
fprintf('End RAM playback\n\n');
endDACPlayback(lib, pHandle);

RX = processRaw(i, q, fs, F_DAC, Nfft);

if (recalibrated)
% update channel estimate
H = RX(Eid) ./ posF(subCarIndex) ;
nTrans = 0;
recalibrated = false;
%break;
else
% grab relevant sub-carriers
RX = RX(Eid);
% calculate new snr
\[ \text{snr}(\text{lastSym}) = \text{avg}(\text{snr}(\text{lastSym}), \text{getSNRdB}(\text{RX}, \text{noise})) ; \]

% add to storage
\[ \text{if (cei}(s/2/\text{totalSyms}) > \text{lastSym}) \]
\[ \text{index}=0; \]
\[ \]  
\[ \text{lastSym}=\text{cei}(s/2/\text{totalSyms}); \]
\[ \text{allNoise}(:, \text{index}+r, \text{lastSym}) = \text{noise}; \]
\[ \text{allRX}(:, \text{index}+r, \text{lastSym}) = \text{RX}; \]
\[ \text{allH}(:, \text{index}+r, \text{lastSym}) = \text{H}; \]
\[ \text{allPosF}(:, \text{index}+r, \text{lastSym}) = \text{posF}(1:\text{Nactual}); \]
\[ \text{allThresh}(\text{index}+r, \text{lastSym}) = \text{mean}(\text{noise}) + 1.5*\text{std}(\text{noise}); \]

% calculations for live results
\[ \text{if (methodA)} \]
\[ \text{onID} = \text{RX} > \text{noiseThresh}; \]
\[ \% \text{adjust for channel} \]
\[ \text{RX(onID) = RX(onID) / H(onID)} ; \]
\[ \% \text{running total of means} \]
\[ \text{thisSampleMean} = \text{thisSampleMean} + \text{mean}(\text{RX}); \]
\[ \]  
\[ \text{else} \]
\[ \text{meanRX} = (\text{mean}(\text{RX}) - \text{mean}(\text{noise}))/\text{mean}(\text{H}); \]
\[ \% \text{running total of means} \]
\[ \text{thisSampleMean} = \text{thisSampleMean} + \text{meanRX}; \]
\[ \]  
\[ \text{end} \]
\[ \text{nTrans} = \text{nTrans} + 1; \]
\[ \text{r} = \text{r} + 1; \]
\[ \]  
%  
\[ \text{end} \% \text{END OF RETRANSMISSION LOOP} \]

\[ \text{if (recalibrated)} \]
\[ \text{recalibrated} = \text{false}; \]
\[ \]  
\[ \text{else} \]
\[ \% \text{average means of each transmission} \]
\[ \text{sampleMean} = \text{thisSampleMean}/\text{R}; \]
\[ \text{symbolsMean(cei}(s/2)) = \text{sampleMean}; \]
\[ \text{s} = \text{s} + \text{ceil}(s/2); \]
\[ \text{index} = \text{index} + \text{R}; \]
\[ \]  
\[ \text{end} \]

\[ \text{end} \% \text{END OF SYMBOL LOOP} \]

\[ \text{[meanBER, symMeanBER]} = \text{getBER(symbolsMean, totalSyms, shapeCompMean} \]
results = sprintf('For range %i
SNR = %f dB
Mean BER = %.3f
',... range, mean(snr), meanBER*100);
symbolBER = sprintf('Symbol BERs:
mean: %.3f %.3f %.3f %.3f %.3f
',symMeanBER*100);
symbolSNR = sprintf('Symbol SNR:
mean: %.4f %.4f %.4f %.4f %.4f
',snr);
disp(results); disp(symbolBER); disp(symbolSNR);
stopTime = datenum(now);
startstop=sprintf('Start: %s
Stop : %s
',startTime, stopTime);
disp(startstop);

save(fhdr,'R','S','Nsubbands','Nactual','Eid','expectedFreq','Nfft'... 'boundsType','scale','shape','pilotTone','lambdaShift'... 'lambdaDeviation');
save(fdata,'allNoise','allRX','allPosF','allH','allThresh',... 'meanBER','symMeanBER','snr');

end % END OF TRIAL LOOP

% Final Clean up
% Close all
fclose('all');

% Delete Scope Objects:
delete([interfaceObj, deviceObj]);

% Disconnect DAC
disp('Disconnect');
res = calllib(lib,'DisconnectFromDeviceXD48', pHandle);
if res<0
  x = sprintf('Error %d disconnect device', res);
disp(x);
  unloadlibrary(lib);
return;
end
% Close the library
unloadlibrary(lib);
disp('Done')
D. User Support Functions

**rse_demod.m**

```matlab
function rxSymbols = rse_demod(modulated_rx, scale)

% Rx symbols
% Use logical indexing to locate all bits of the appropriate type.
ll = ((modulated_rx >= scale(4,1)))
rxSymbols(ll,1) = 1; rxSymbols(ll,2) = 1;

lO = (modulated_rx >= scale(3,1) ...
    & modulated_rx < scale(3,2));
rxSymbols(lO,1) = 1; rxSymbols(lO,2) = 0;

Ol = (modulated_rx >= scale(2,1) ...
    & modulated_rx < scale(2,2));
rxSymbols(Ol,1) = 0; rxSymbols(Ol,2) = 1;

OO = (modulated_rx < scale(1,2));
rxSymbols(OO,1) = 0; rxSymbols(OO,2) = 0;
```

**craftSignal.m**

```matlab
function [signal xT] = craftSignal(posF, Np, SW,AMP, use_np)

% As a requirement of the card, the TX signal in time domain must be a
% 32768 long vector
signalVectorLength = 32768;

% Craft FD signal and convert to Time domain
% flip posF to get the negative coefficients.
negF = fliplr(posF); %get negative coefficients

% Create signal in Frequency Domain and Time Domain
% DC pos–ax neg–ax
xF = [0 posF negF]; %correct vector format for MATLAB ifft

if use_np
    xT = fftshift(ifft(xF)*Np/2); %time domain representation
    % *Np/2 is to scale signal amplitude to 1
else
```

65
$x_T = \text{fftshift} \left( \text{ifft}(x_F) \right)$; \textit{%time domain representation}

end

%create smallest zero−padded vector divisible by $2^{15}$
$x_T = [x_T \; \text{zeros} \left( 1, \left(2^{\text{ceil} \left( \log_2(Np) \right)}−Np \right) \right)]$;

% Amplify signal
$x_T = x_T .* \text{SWAMP};$

signal = []; %
for $i = 1:\left(2^{15}/\text{length}(x_T)\right)$
    signal = [signal \; xT];
end

fprintf('Signal length = \%i\nZeros added = \%i\n\n',\text{length}(signal),\left(2^{\text{ceil} \left( \log_2(Np) \right)}−Np\right))

if length(signal) \neq \text{signalVectorLength}
    error('ERROR: Did not meet DAC vector size requirements. Check vector size.');
end

\textbf{genPilotSignal.m}

$$\begin{align*}
\text{function} & \quad [\text{signal}, \; \text{posF}] = \text{genPilotSig}(\text{Nsubbands}, \; \text{Nactual}, \; \text{Np}, \; \\
& \quad \quad \quad \quad \text{SWAMP}, \; \text{pilotTone}) \\
\text{posF} & = \text{zeros}(1,\text{Nsubbands}); \% \text{initialize posF} \\
\text{subCarIndex} & = (1:\text{Nactual}); \% \text{choose ON sub−carriers}. \\
\text{posF}(\text{subCarIndex}) & = \text{pilotTone}; \\
\text{[signal, \; xT]} & = \text{craftSignal}(\text{posF},\text{Np},\text{SWAMP},1)
\end{align*}$$

\textbf{checkSignal.m}

$$\begin{align*}
\text{function} & \quad \text{badSignal} = \text{checkSignal}(\text{posF}, \; \text{signal}, \; \text{shapeComponent}, \\
& \quad \quad \quad \quad \text{scale}, \; \text{useVar}) \\
\text{if} & \quad (\text{nargin} < 5) \\
& \quad \text{useVar} = \text{false}; \\
\text{end} \\
\text{if} & \quad (\text{max(abs(signal)}) > 2^{15}) \\
& \quad \text{badSignal} = \text{true}; \\
& \quad \text{return}; \\
\text{end} \\
\text{if} & \quad (\text{useVar}) \\
& \quad \text{thisScale} = \sqrt{\text{var(posF)/shapeComponent}}; \\
& \quad \text{if} \quad (\text{thisScale < scale(1)}) || (\text{thisScale} \geq\text{scale(2)}) \\n\end{align*}$$
badSignal = true;

return;

else

thisScale = mean(posF)/shapeComponent;

if ( (thisScale < scale(1)) || (thisScale >= scale(2)) )

badSignal = true;

return;

end

end

badSignal=false;

return;
Bibliography


