Implementation and Analysis of Spectral Subtraction and Signal Separation in Deterministic Wide-Band Anti-Jamming Scenarios

by

Travis F. Collins

A Thesis
Submitted to the Faculty
of the
WORCESTER POLYTECHNIC INSTITUTE
in partial fulfillment of the requirements for the
Degree of Master of Science
in
Electrical and Computer Engineering
by

April 2013

APPROVED:

Professor Alexander Wyglinski, Major Advisor

Professor Christopher Anderson

Professor Andrew Klein

Abstract

With the increasing volume of wireless traffic that military operations require, the likelihood of transmissions interfering with each other is steadily growing to the point that new techniques need to be employed. Furthermore, to combat remotely operated improvised explosive devices, many ground convoys transmit high-power broadband jamming signals, which block both hostile as well as friendly communications. These wide-band jamming fields pose a serious technical challenge to existing anti-jamming solutions that are currently employed by the Navy and Marine Corps. This thesis examines the feasibility of removing such deterministic jammers from the spectral environment, enabling friendly communications. Anti-jamming solutions in self-jamming environments are rarely considered in the literature, principally due to the non-traditional nature of such jamming techniques. As a result, a combination of approaches are examined which include: Antenna Subset Selection, Spectral Subtraction, and Source Separation. These are combined to reduce environmental interference for reliable transmissions. Specific operational conditions are considered and evaluated, primarily to define the limitations and utility of such a system. A final prototype was constructed using a collection of USRP software defined radios, providing solid conclusions of the overall system performance.

Acknowledgements

I would like to acknowledge my advisor Professor Alexander Wyglinski for his guidance and support. Along with the previous individuals involved with the BLISS project, especially Dr. Srikanth Pagadarai for all his hard work. I also want to thank Professor Andrew Klein and Dr. Christopher Anderson for serving on my committee. Additionally I want to acknowledge the Office of Naval Research and the MathWorks Inc. for their financial support, making this research possible.

Finally, I want to thank my parents Susan and Stephen for all their love and support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Since the advent of modern digital communications in the 20th century, there has been an explosion in the demand for wireless spectrum. As a result, spectrum is becoming an increasingly scare resource. This demand is a direct result of the availability and relatively inexpensive cost of such wireless devices. Therefore, in such environments as military operations, disaster relief scenarios, and natural defense situations, the probability of interfering transmissions [4], intended and unintended, has steadily grown to a point where techniques are needed in-order to combat such occurrences. More directly, in such situations when interfering signals are partially or completely understood measures need to be devised in-order to overcome such difficulties.

In military theatres, it is extremely common to observe friendly operated high-power broadband jamming signals [5]. Such devices exists as part of group convoys in several branches of the military and in many other forms across a variety of deployments. Unfortunately, such devices block both friendly and hostile communications, and current anti-jamming techniques have not provided a viable solution to this problem. Therefore, new approaches should be considered, utilizing more flexible radio technologies.

Understanding how to overcome such challenges is a complex task; with vastly different

transmission environments and differing operating devices and operating standards. A new system that could combat such pitfalls should rely on all friendly information, or be able to construct solutions of its own from a set of tools given to the radio. Such tools should be flexible and easily modified, changed, or improved. This ability to easily change or adapt is a key feature as the technical requirements can change from day to day, or between branches of the military itself. As such a solution should have the following attributes:

- **Flexible:** Easily adaptable to many situations and interference types, while still relying on the same hardware.
- **Efficient:** Relatively low hardware cost and low computational complexity
- **Robust:** Designed to provide guaranteed performance gains even under severe channel conditions

1.2 State of the Art

Current implementations of anti-jamming technology lies on the straddling point of hardware and software in the communications world. This is true because hardware provides the speed and performance needed for digital data transmission, while software provides higher level intelligence and flexibility in such layers as the media access control layer and the network layer of the Open Systems Interconnect (OSI) model [6]. An outline of the model can be seen in Figure 1.2. For anti-jamming applications, smarter radios allow for enhanced mobility against the jammer. Therefore, a largely focused software implementation, allowing for highly intelligent radio decisions, must be considered when investigating anti-jamming techniques.

Current anti-jamming techniques include channel hopping, spatial retreat, jammed area mapping, node escape, retreat restoration, frame masking, and many more [7]. All of these techniques use mechanisms of evasion or deception. These can be quite effective when attacked by generally narrowband, non-dynamic/non-learning jammers. In the case of wide and ultra-wide band jammers, these techniques are not as effective. This wide-band en-

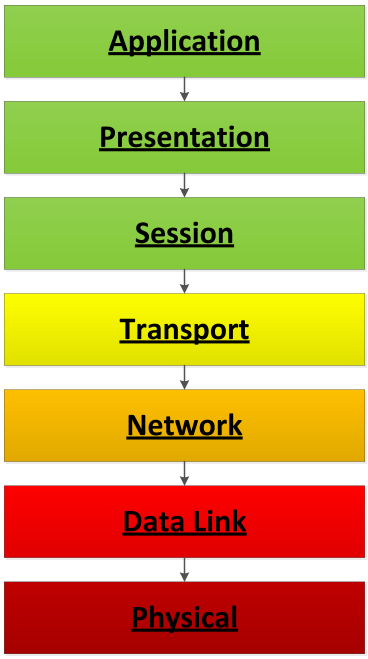


Figure 1.1: The Open Systems Interconnection model is made up of 7 layers, but three primary sections. This thesis primarily focuses on the lowest level of this model, the physical layer.

vironment is the primary situation of interest, and it generally considered a technically challenged scenario. These anti-jam techniques are design for specific situations and jammers.

Let us first examine these anti-jamming techniques which are broken down into three primary categories: Proactive countermeasures, Reactive countermeasures, and Mobile agent-base countermeasures [7]. Reactive countermeasures relies on a varying array of detection mechanisms first to determine if that node is being jammed. These detection methods must be coupled with a countermeasure or the scheme is inoperable. Examples of these detection methods include a transmitter-based approach and a receiver-based detection[8]. In a transmitter-based approach, such as ad-hoc networks, a decision algorithm is used based on four metrics: Packet Delivery Ratio (PDR), Received Signal Strength Indicator (RSSI), Physical rate, and Noise levels [9]. In the receiver-based detection additional information must be injected into frames to help the receiver determine the number of frames lost.

Since frames can be easily lost in wireless transmissions, the receiver is handicapped when determining the number of retransmissions that have occurred. In the transmitter the PDR is deterministically determined by the data-link layer, sequence numbers must be added to frames for the receiver to accurately calculate the PDR [9]. Several other detection methods exist including using a detected detector, cooperative detection among nodes in a wireless network, and more sophisticated methods of RF fingerprinting [9].

Once the jammer has been detected the reactive countermeasures come into play. Many evasion techniques exists to combat narrowband jammers such as: channel hoping, spatial retreat, retreat restoration, hybrid attacks, and many cognitive radio approaches [10]. Many of these techniques utilize the network itself to adapt to the jammer, which is an appropriate assumption because without a network wireless communications are irrelevant. Channel hopping is simple and can be considered straightforward to implement. If a channel is beginning jammed, a communication system can simply "hop" to another channel. This is easily defeated in two cases: The first case involves the jammer following you or the jammer is simply wide-band capable. In the second case, one can employ spatial retreat, which is a mechanism to physically evade the areas being jammed. Based on the detection algorithm, all nodes in a network try to estimate the jammed region and flee physically in the direction of safer place. Based on their estimation about the jammed region, nodes will utilize shortest path algorithms to determine location of retreat [11]. Retreat restoration is focused around how to rebuild a network once the jammer has left. Retreat restoration can be done by coordinated or uncoordinated communication, and the transmissions are based on a preplanned hop patterns among nodes [12].

There also exists systems that are design to resist jamming pro-actively. These hybrid systems [13] utilize preventatives measure to resist jamming such as frequency hopping spread spectrum (FHSS). Spread-spectrum signals are highly resistant to narrowband jamming, unless the jammer has knowledge of the spreading key. In military applications, the spreading key is generally created using a cryptographic function [14]. More hybrid solutions include synchronous and asynchronous spectral multiplexing where intermediary

nodes are used to communicate at multiple channels. When a node changes its channel because of jamming a neighbor will heal that connection by communicating with the node on its new channel and rest of the network on the old channel [15].

The largest problem with these techniques is they all have are designed to combat narrowband jammers, and even friendly jammers. If high powered wideband jammers enter the equation, all of these solutions begin to fall apart. Note these techniques primarily exploit the dimensionality of their environment by simply avoiding the jammer, and all techniques require intelligent flexible hardware solutions. To implement such solutions requires sophisticated hardware implementations, that can be quite rigid for rapidly changing communication environments and adversaries. To compensate solutions that push more of the radio operations from their original rigid hardware implementations into the more flexible software domain, provide a more cost effective and elegant solution. These software focused radios, also know as software defined radios (SDR), have provided a solid platform for very adaptive anti-jamming technologies under the name cognitive radios [16]. These radios have the ability to easily learn and adapt to their environment, which is the primary requirement of anti-jamming devices.

As mentioned above, it is quite common for the military to self-jam its own channels as a result of co-channel interference. Unfortunately, this can hinder their own use unintentionally. These disrupted users are known as "disadvantage users". They are commonly small mobile hand-held devices and cannot simply overcome the jammer computationally or in raw power. Therefore, more manageable and elegant solutions must be considered for such disadvantaged users. Beside self-jammming, adversarial jammers must also be considered. Fortunately, certain characteristics can be statistically exploited if these jammer abide by certain properties. Since adversarial jammers tend to inject random data or energy to block communication, if these transmissions can be shown to repeat they can be exploited. In the case of self-jamming, the signal characteristic can be know *a priori*. Therefore they also can exploited or removed, negating the effects of such devices. Such scheme must consider the energy or symbols of the jammmer that are orthogonal and/or non-orthogonal to the symbols of the communication itself.

The goal of this project is to exploit a self-jammed and statistically deterministic adversarially jammed channel, through the utilization of cognitive radio, implemented on a software defined radio platform. Software defined radios, defined as the intersection between hardware radios and computer software [17], provide a platform flexible enough to support highly intelligence operations such that anti-jamming requires. A proposed adaptive signal processing software solution for mitigating the effects of both intentional and unintentional jamming (including wideband jamming) via the combination of antenna subset selection, spectral subtraction, and blind source separation (BSS) techniques in order to extract specific transmissions from a mixture of intercepted wireless signals is shown in Figure 1.2. The goal of our proposed solution, called BLInd Spectrum Separation (BLISS), is to enable reliable, high throughput, and robust end-to-end wireless communications.

This work is a continuation of the work done through a collaboration of Worcester Polytechnic Institute and the United States Naval Academy. Primarily literature surveys and early simulations were completed or attempted before the transition of the project to the work done by this thesis. Credit is given to the following authors and there coinciding section or block as follows:

- **Blind Source Separation: Dr. Srikanth Pagadarai and Ryan Dobbins**
- **Spectral Subtraction: Robert Over**
- **Antenna Subset Selection Robert Capizzio, Benjamin Hilburn and Dr. Christopher Anderson**

This document examines the provided work done by these individuals in detail, except for the topics in Antenna Subset Selection due to time constraints.

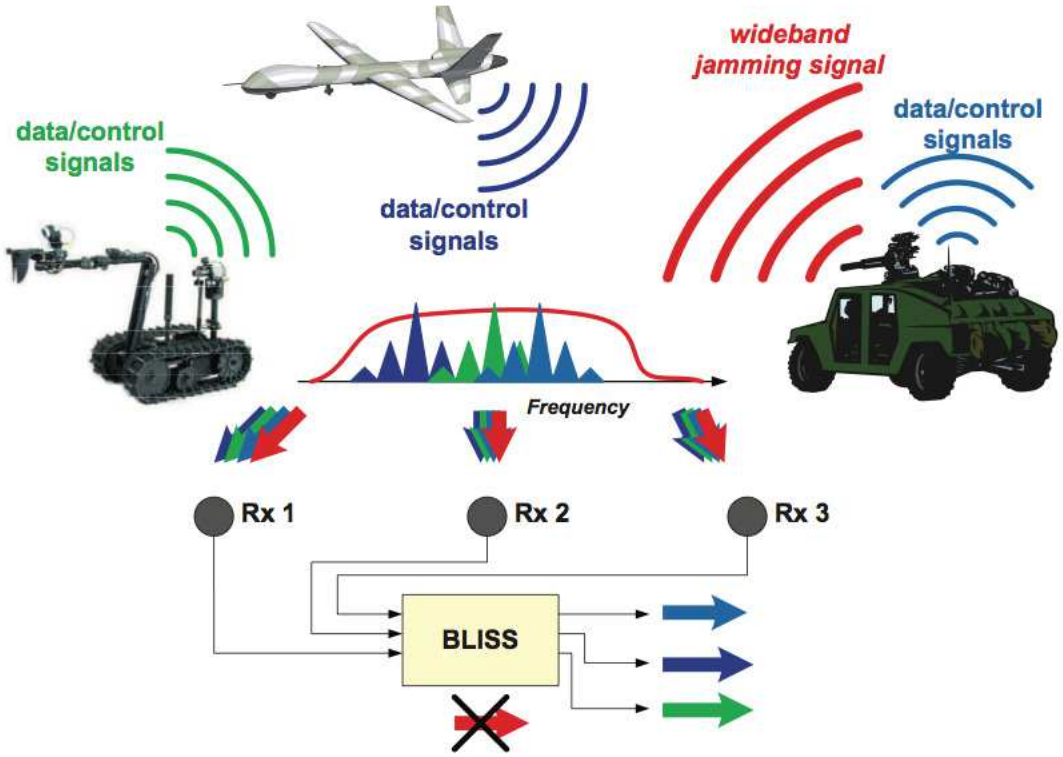


Figure 1.2: Original system proposed, with the goal of separating desired signals from a spectrally mixed cluster. This "BLISS" system would allow many systems to easily coexist.

1.3 Thesis Contributions

This thesis will contribute the following to the wireless communications and signal processing research communities:

- A theoretical and simulated technique for non-orthogonal signal removal of undesired known signals from the desired operating band.
- A theoretical and simulated technique for residual signal and noise removal, primarily manifested as frequency selective fading.
- A practical implementation using over the air communications of an anti-jamming system utilizing software defined radios. This implementation will tackle wide-band non-orthogonal and orthogonal jamming, and provide evidence of performance under

specified channel and jamming signal conditions.

1.4 Thesis Organization

This thesis will be organized into the following chapters: Chapter 2 provides the necessary background to understand basic communication system design, anti-jamming techniques, and signal processing. Chapter 3 puts forward theoretical simulations and a design of a physical anti-jamming system. Chapter 4 presents the results of the physical implementation and analysis of its findings. Chapter 5 concludes the thesis, summarizing the accomplishments and outlines possible future work.

Chapter 2

Required Background Knowledge

This chapter provides the background information needed to understand the chapters that follow. It examines the basic outline of a communication system and how non-idealities are compensated for, with the addition of multiple input multiple output (MIMO) systems and a unique filtering technique called spectral subtraction. Secondly, this chapter investigates common jammer scenarios and anti-jamming solutions. Finally, it outlines the necessary hardware and software tools used in the implementation chapter.

2.1 Jamming

In 1899 Guglielmo Marconi successfully transmitted radio messages across the English Channel, and nine months later Alexander Bell was discussing how this could be jammed during wartime [18]. Bell stated that such a wireless system can be easily disrupted with simple electromagnetic distrubances: "It's as easy as cutting the wires" [18]. In the early days of wireless communication, such systems were very fragile but today they have become substantially more resilient. In its simpliest form, radio jamming is the transmission of electromagnetic signals that interfere with communications by decreasing the signal to noise ratio (SNR) between the transmitter and receiver. This jamming can be either deliberate or unintentional, and can either entirely disable the communication link or limit its capacity. , a common example of unintentional jamming is microwave ovens which operate at a wavelength of 122 millimetres which translates to 2.45GHz from this equation: $\lambda = \frac{v}{f}$,

with *lambda* representing the wavelength, v the velocity and f the frequency. This band directly interferes with channels defined under the IEEE 802.11 standard, also known as Wi-Fi [19]. Deliberate jamming on the otherhand, is generally more saphisticated and takes many different forms.

Intentional communications jamming is usually aimed at radio signals in a combat setting, where consequences are insigificant or out of the relm of the law. In the most rudimentary designs, a jammer will simply tune their own frequency to that of their opponent and with a similar moduation scheme (and significant power) disrupt the enemies transmissions. The most common types of this form of signal jamming are: random pulses, stepped tones, warbler, tones, rotary, pulses, sparks, recorded sounds, gulls, sweep-through, and random noise [14]. These methods obviously (or subtly) disrupt transmissions by inserting electronmagnetic energy into the transmission space of the receiver. In more technical terms, the jammer is producing randomly chosen data that is non-orthogonal to the data which the friendly transmitter is producing. Since this jammer's data is pseudo-random when his transmissions are added to the 'oponent's' signal, the result appears to be random as well. Therefore, the signal is unrecoverable. As mentioned above, the jammer must produce signals that are non-orthogonal to the enemy of his jamming will have no effect. An example below shows random noise at a significant noise level is added to a previously destinquishable signal.

2.2 Anti-Jamming

Anti-jamming has been considerably outlined in the Introduction chapter, therefore this section will examine more advanced narrowband and wideband techniques that involve filtering rather than avoidance. All of these approaches have various monitary costs, constraints, and power limitations. First, narrowband mitigation techniques will be considered. These include adaptive filtering, time-frequency domain filtering, adaptive antennas and subspace processing. By combining several of the listed techniques, wideband jammers can also be ad-

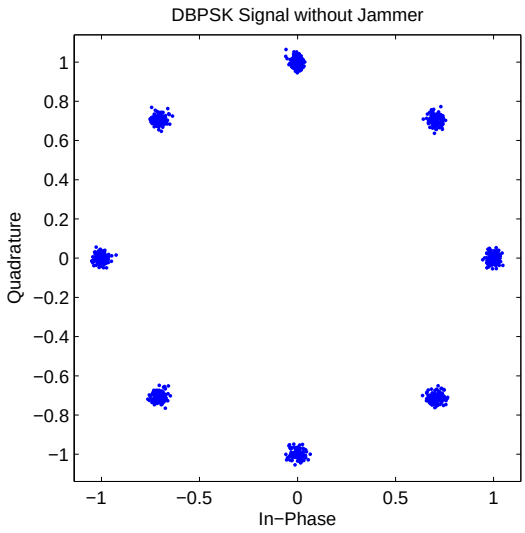


Figure 2.1: DBPSK signal uncorrupted by jammer. Clean clustering around the constellation points provides accurate symbol recovery.

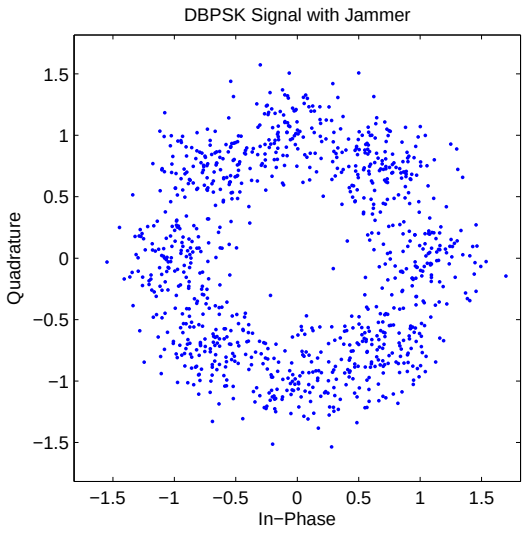


Figure 2.2: DBPSK signal corrupted by jammer. The overlapping clustering around the constellation points provides difficult symbol recovery.

dress, under certain conditions. Table 2.2 compares these techniques with various attributes.

Table 2.1: Comparision of Anti-Jamming Techniques

Technique	Cost	Size	Flexibility	Complexity
Adaptive Filtering	low	small		
Time-Frequency Domain Filtering				
STFT	low	small	Environment Specific	low
Filter Banks	low	small	Environment Specific	low
Wavelet Transform	low	small	Environment Specific/Resolution Required	low
Subspace Processing	low	small		
Adaptive Antennas				
Null Steering	high	large		high
Beam Forming	high	large		high

Adaptive filtering is a well defined solution in jammer mitigation, but it is to date the most limited. Most notably, the jammer must be a relatively narrowband and the period of the jammer must be relatively short. An example of an adaptive filtering technique is a suppression filter. Suppression filters assume statistically the signal is Gaussian, which results in the optimal filter being linear. This filter essentially solves the Wiener equation for an

optimal filter, but generally a Least Mean Square (LMS) implementation is used instead of just inversing the channel estimate [20]. The technique of inverting the channel estimate or correlation matrix is traditionally called a zero forcing equalizer and is extremely unstable in the presence of small noise.

Next, time-frequency domain filtering attempts to represent the transform the received signal in such a way that it is possible to easily distinguish the jammer from the data signal. A Short-Time Fourier Transform (STFT) can be used to accomplish this goal. A STFT operates by sliding a window across a signal and taking the fast fourier transform (FFT) of that window. Reference [21] uses the STFT to break a signal into its frequency components. From this information, with a narrowband jammer only a small number of frequency domain bins contain nearly all of the interferers. Therefore these bins can be simply nulled and an inverse FFT can be applied to the signal to regain its time domain version. This is very effective with the use of a spread spectrum signal with a narrowband jammer.

Filter banks is a second methodology that can be used to reduce spectral leakage in the frequency domain, which is the primary drawback with the STFT approach. One advantage of the filter banks approach is they do not inject interference when the jammer is not present, which is a common problem when the jammer turns on and off frequently. Filter banks provide jammer suppression after their spectral decomposition stage, since at this point sub-band encoding can be accomplished this spectral modification simply nullifys the jammer [22]. A similar decomposition is the wavelet transform. Unlike the STFT, the wavelet transform is much more flexible because the STFT has a fixed resolution for a given FFT size unlike the wavelet transform [23]. Subspace processing which is a form of wavelet transform, is applied in this way. The jammer subspace can be made orthogonal to the wanted signal subspace, nullifying the jammer's effects [24].

Besides these signal processing methods, physical techniques can also be use to do spatial filtering. These techiques make use of several antennas, and as an assumption the number of interferers must be equal to or less than the number of antennas. The first approach is

called Null Steering. Null Steering constantly computes the weights in order to minimize the received energy level. In effect, this technique attempts to steer the antenna away from the jammer. The second approach is called Beamforming. Beamforming tries to adjust the antenna in order to maximize the SNR. In effect, the antenna beam is steered in the direction of the desired signal. It is of course, possible for the jammer's signal to be in the same direction as the signal source. Therefore the postcorrelation technique is used in order to obtain the SNR. However, prior knowledge of the signal direction and the host location is required [25]. It is also important to note that larger the number of elements in the array itself, the closer the jammer can physically be located to the desired transmitter.

Historically, all of these approaches historically were applied to spread spectrum communication systems because narrowband jammers fundamentally are considerably easier to deal with in this setting. They are more straightforward because the jammer effects only a fraction of the transmitter's transmission space; therefore, when wideband jammers exist many of these schemes fall apart. Other avenues or scenarios must be considered in such situations to overcome this limitation. Before a solution is chosen, additional signal processing and communication theory must be understood. These topics will be examined in the following sections.

2.3 Communication Systems

Modern wireless digital communication systems are based on a rich tradition of analog experimentation and theory. These signal technologies surround us constantly, such as cellphones, car radios, GPS, and more. All these of these devices communicate over wireless links and are built upon the same building block of transmission and reception theory. Many perspectives can be taken, but a more generic observation should be taken at the system level. Depending on the level of sophistication these blocks can expand greatly, but still solve the same issue caused by the wireless transmission of digital access across their environment. Such non-idealities such as frequency offsets, doppler effect, signal echoes, phase

shifts, and others must be compensated for to successful receive uncorrupted information.

Let us examine the transmitter first since it is less complicated than the receiver. The transmitter's primary goal is to send data in a resilient form, or structure, to create a more managable signal for the receiver. This is accomplished in several steps, and the function, or purpose, of the overall system determines the sophistication of the design. Figure 2.3 outlines the major building blocks of the transmitter; consisting of the coder, pulse-shape filter, and frequency translator. A filter is added after the coding block in some implementations to provide such effect as pre-distortion.

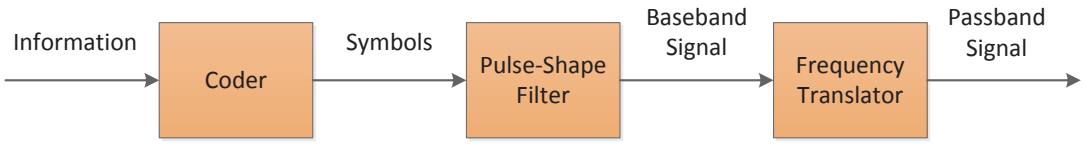


Figure 2.3: Basic transmitter outline, converting information or data to a easily recoverable signal by the receiver. The transmitter consists of three primary blocks: the encoder, pulse-shape filter, and frequency translator/modulator.

The transmitter's sole purpose is the send data that is convenient for the receiver to understand, and allow others to use the transmission medium as well. The coding phase of the transmitter can have many features and purposes, but simply it will encode data into a symbol with a form of redundancy or scheme that will help the receiver reconstructed the information more easily. Next the pulse-shape filter is used to help separate data and help maximize the SNR at the receiver. This filtering can be done with an assortment of filter shapes, but the most popular is the raised square-root cosine filter. After pulse-shaping, the signal is translated into frequency information and unconverted to a high RF with a carrier signal. The translation is done with a modulation scheme such a binary phase-shift keying (BPSK) or pulse amplitude modulation (PAM). The is up-converted by mixing the signal with a sinusoid, seen by Equation (2.1).

$$[!ht]cos(x)cos(y) = \frac{(cos(x+y) + cos(x-y))}{2} \tag{2.1}$$

This done because low-frequency signals such as speech, music, or digital data can be much more efficiently transmitted at higher frequencies [26].

Now let us discuss the receiver. At the system level, a modern digital receiver can be broken down into a small set of distinct categories or operations: carrier synchronization, timing synchronization, equalization, and frame synchronization, as outlined in Figure 2.3. These sections work together in series to provide smooth transmission of data, and many techniques exist within theses categories to accomplish its goal. In most communication systems, after the radio frequency (RF) front-end, the first operation done on the received signal is frequency compensation and down conversion. This compensation needs to accomplished because non-idealities and differences exist between the transmitter's and receiver's oscillator. Therefore this is continually compensated for and corrected. Carrier recovery can be accomplished using several methods that include but are not limited to: squared difference loops, phase-locked loops, costas loops, and decision-directed phase tracking [26].

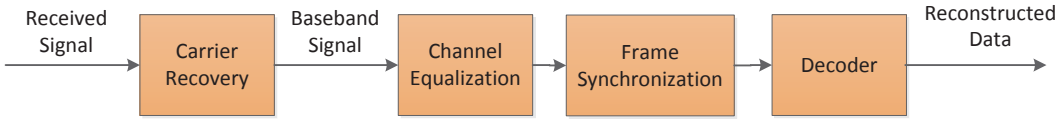


Figure 2.4: Basic wireless receiver outline, with four primary blocks. All designed to remove corruption caused by the wireless environment and translate transmitted signals back to desired data.

After carrier recovery, the signal is pulse-shaped with the same filter shape used at the transmitter. This technique will help to maximize the SNR of the signal. Then the signal must be corrected again for timing. The purpose of timing recovery is to choose the instants at which to sample the incoming signal. This is generally done through a interpolation mechanism of the transmitted signal. Since at the transmitter the signal is

upsampled to symbols, a single data point or bit is represented by several received data points. Therefore these points can be interpolated together for a more accurate estimate of the original data. Timing recovery also can be done with one of several methods including: output power maximization, Mueller-Muller method, or decision-directed. Generally, they utilize their own interpolation algorithm, such as sinc-interpolation [26].

After this point the receiver designs can vary greatly, as the design in this thesis will present, because this is where most of the digital signal processing (DSP) will take place. This section, call Equalization, is responsible to correcting any effect the channel has on the signal. This includes multi-path, noise, and other distortions that cause inter-symbol interference (ISI). Equalizer implementations are designed to compensate for types of disturbances that occur using certain systems. The equalizer stage is most often coupled with the frame synchronization stage such that the equalizer itself can adapt to changing conditions. This is known as soft decision making. Equalizer techniques include but are not limited to: LMS , decision-directed, dispersion-minimizing [26], Viterbi [27], blind, and turbo equalizers [28].

2.3.1 Equalization

Equalizers can be considered the most complicated design of an entire communication system since they combat a series of distortions. The primary result of these distortions is called inter-symbol interference (ISI). ISI simply means that symbols interact with one another in the channel space and cannot be considered independent from one another. Since this interference is generally considered a frequency selective disruption or dispersion a filter needs to be employed to reverse such effects. This filter must be adaptable because the channel distortion cannot be know prior to transmission.

As listed in the previous section, many equalizers exists and operate under specific conditions. Here several linear equalizers will be discussed in detail including maximum-likelihood sequence detection, adaptively trained equalizer, and decision-directed linear equalization.

The goal of all of these equalizers is to find a finite impulse response (FIR) filter that when convolved with the received signal produces the original transmitted data $\hat{\boldsymbol{X}} = \mathbf{Y} * \mathbf{G}$. Figure 2.3.1 outlines a typical FIR structure for which the equalizer will create the appropriate coefficients $b_0, b_1, ..., b_n$. These equalizers also examine the condition of an additive white Gaussian noise (AWGN) channel, and uncorrelated or independent interferers.

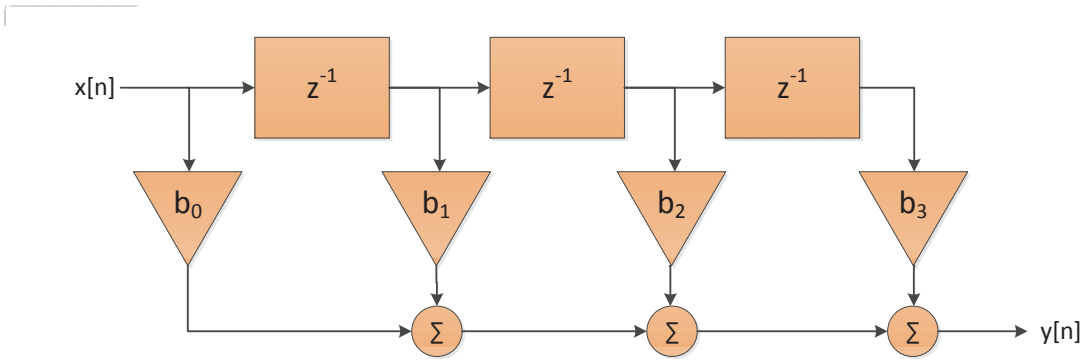


Figure 2.5: FIR filter structure with four taps. The represent b the tap coefficients, and the z represent the tap delays. The delays allow the filter access to previous signal samples for future use.

The Zero Forcing Equalizer (ZFE) uses peak distortion criteria to determine equalizer coefficients. The ZFE produces zero ISI at its output. If $H_c(f)$ is assumed to be the effects of the channel, the ideal equalizer would be $H_{eq}(f) = \frac{1}{H_c(f)}$. This can also be consider the inverse of the channel. The filter coefficients are modeled as weighted pulses convolved with the channel, which can be expressed as Equation (2.2). Here b represents the weighted filter taps, p_r represents the input signal and p_{eq} represents the output of the filter.

$$p_{eq}(t) = \sum_{k=-M}^{M} b_k p_r(t - kT) \tag{2.2}$$

Unfortunately, the ZFE has a large disadvantage; it cannot compensate for small amounts of noise. Technically, the ZFE will amplify all noise of the received signal, and if any elements of the channel matrix are considerably small, then the equalizer becomes unstable. Therefore this is generally considered a more theoretical or elementary equalizer formula-

tion. To overcome this problem the zero ISI condition must be relaxed allowing for noise which if small can easily be overcome by such operations as quantization or decision making. The Linear Minimum Mean Squared Error Filter (LMMSE) takes this relaxation into account [29].

The LMMSE assumes that the symbols are uncorrelated with one another and uncorrelated from the noise in the channel. This approach tries to minimize the mean square error, a common measure of estimator performance. The estimator is defined as $\hat{x}_{MMSE}(y) = Ex|y$, where we are given the received signal y and must guess or estimate x, which was transmitted originally. If x and y are jointly Gaussian, then the LMMSE will be linear. This function or equalizer design minimizes the mean square error. To simplify further an extension to random vectors can be examined. An estimate can be made for the original vector x represented by $\hat{x}$, resulting in the linear equation $\hat{x} = ay + b$. a and b represent the coefficients to be selected for the estimator. The LMMSE will minimize the mean square error shown in Equation 2.3:

$$MSE = \|x - \hat{x}\|^2 \tag{2.3}$$

Besides these linear equalizers outlined, an adaptive approach can also be considered. The Least Mean Squares (LMS) or Gradient Descent algorithm utilizes a traditional technique for minimizing the error in a signal. This method is historically known as the "Method of Steepest Decent" or a very closely related algorithm called "Newton's Method". By calculating the error of each received symbol, this can be fed back into the system for future symbols. This error will shape the equalizer's filter coefficents to match the inverse of the channel. The equations 2.4, 2.5, and 2.6 outline the LMS algorithm.

$$y[n] = w[n]^H u[n] \tag{2.4}$$

$$e[n] = d_n - y[n] \tag{2.5}$$

$$w[n+1] = w[n] + \mu u[n] e^*[n] \tag{2.6}$$

In these equations: w represents the adaptive filter coefficients, u the input signal, and d the known signal. μ acts as the algorithm's step-size determining how quickly it will converge. It must also be considered that the larger the step-size the higher the probability it may become unstable. As long as the channel's effects are slow changing, this equalizer can easily maintain up to date estimates while corrupting as little of the data as possible.

All of the methods proposed so far require *known* data (d_n in Equation (2.5) to correct against. This data is called training data and generally comes in the form of a preamble in a frame. The preamble is added to the beginning of each frame so the equalizer can learn from the effects on that specific data. The preamble is the same for all frames and is always used so the equalizer will always be learning. However, what happens when data is unknown in the frame, such as the payload portion of the frame. This is where blind equalization is employed.

Several blind equalizers exist but an extension of the LMS equalizer for blind situations will be examined here called the decision-directed equalizers [26]. For a blind equalizer to operate an error generation mechanism must be evaluated, but since the data symbols are unknown, a decision device must be used in place. This decision device is a quantization method and the error is generated from this quantization. This error generation is extrapolated from expression: $e = \frac{1}{2}(sgn(y[k] - y[k])^2$ Where the sgn function returns 1 for positive numbers and -1 for negative numbers. This expression is quite similar to the original LMS implementation except instead of a known symbol the data is quantized using the sign function. This type of quantization using the sign function is only applicable with binary modulation schemes such as BPSK. This equalizer method is usually combined with a training equalizer method in practice, since if a nearly closed eye is observed, when using an eye diagram, this equalizer cannot open it by itself. An example of such an eye diagram is shown in Figure 2.7, and clean/open eye diagram can be seen in Figure 2.6.

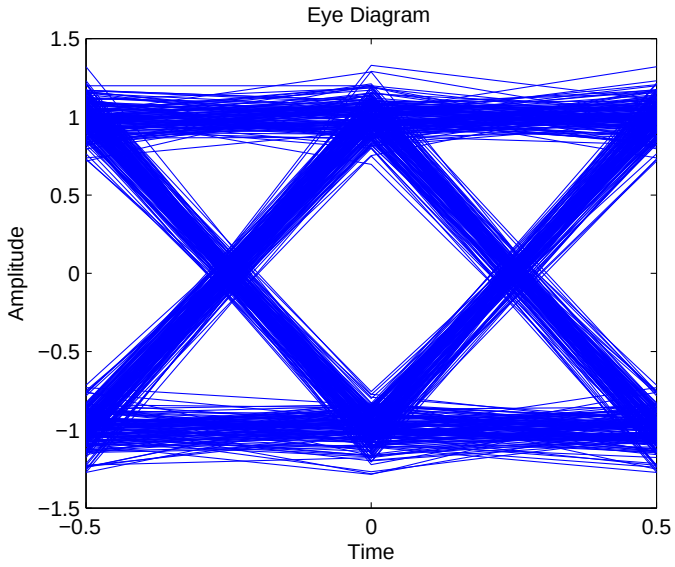


Figure 2.6: Adequate timing recovery produces open eye, which clearly defines the received symbols in time.

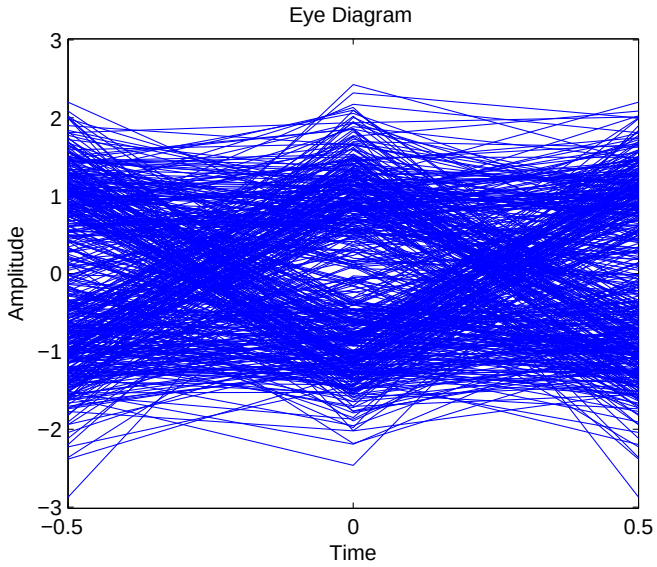


Figure 2.7: Poor timing recovery produces a closed eye, identifying that the signal cannot be recovered unless corrective measures applied.

In this section, we have examined several equalizer techniques while outlining their advantages and disadvantages. Most techniques require some training mechanisms to operate under heavy channel distortion, and blind techniques such as decision-directed equalization will fail under these conditions. Unfortunately, such training data can take considerable resources, and lower overall data throughput. In practice as much as 20% of frame information is training. Therefore other techniques must be considered to help overcome this obstacle.

2.3.2 Superimposed Training Equalization

As mentioned in the previous section, many implementations exist for equalizer designs, but this thesis will examine the effectiveness of superimposed training symbols in frequency selective channels. In traditional equalizers, channel estimation is achieved through the use of training data or pilot symbols. These symbols are both known to the transmitter and receiver, providing the basis for an estimate. In these equalizers all training symbols are placed at the start of a frame, with [30] showing that under high SNR training-based schemes are capable of capturing most of the channel capacity, while under low SNR they are highly suboptimal. Superimposed equalizers try to overcome this problem along with others to provide more optimal estimates. Superimposed equalizers physically add training symbols to the data stream instead of concatenating symbols, saving valuable bandwidth [30]. To accommodate such pilots, energy must be shared among the data and hidden pilots [3]. Reference [31] shows that for a transmitter of fixed power, with an additive pilot sequence, the decrease in data signal power is equal to:

$$K_{loss} = \frac{E[\|s(k)\|^2}{E[\|s(k)\|^2] + E[\|u(k)\|^2]}$$

equivalent to $10 log K_{loss} dB$ in signal to noise ratio (SNR). With $s(k)$ representing the source signal, and $u(k)$ representing the received signal. Other disadvantages include an increased signal envelope fluctuation that can be undesirable in nonlinear transmit power amplifiers [32].

At the receiver, channel estimation can be done using several techniques in both the frequency and time domain. Reference [32] examines a time domain approach for synchronized averaging of the received signal. It is important to note that this synchronization is not related to transmitter and receiver synchronization. References [32] and [33] both assume that the signal $\mathbf{x}(\mathbf{n})$ and noise $\mathbf{v}(\mathbf{n})$ have zero mean and $E[m_x(n)] = \mathbf{d}(\mathbf{n}) = \mathbf{p}(\mathbf{n}) * \mathbf{h}(\mathbf{n})$. Therefore, since $\mathbf{p}(\mathbf{n})$ is the known superimposed periodic pilot sequence, $\mathbf{h}(\mathbf{n})$ can be determined. Note that $\mathbf{h}(\mathbf{n})$ is generally considered frequency selective, and such channels can be quite difficult to deal with especially with multi-path. Multi-path interference is a distortion caused when copies of the original signal arrive at the receiver delayed on top of the originally received non-delayed signal. This delayed signals essentially take other paths to the receiver, and this interference's manifestation is commonly called *ghosting* in such applications as television broadcasts [34].

Superimposed equalizers are able to better compensate for large multi-path channels because they can spread their training symbols throughout the signal itself. This spreading not only provides a spreading in time but also in other dimensions such a frequency. Therefore, if the training symbols are chosen correctly and placed correctly, they can then be spread across the frequency spectrum efficiently and capture its selectivity. Before the pilots can be examined, the channel must be defined. The channel is of block length N, and the channel is also time invariant across single blocks, but variable across blocks. The memory of this channel is of maximum length $L-1$, and the impulse response of the channel is defined as $\mathbf{h} = [h_0, ..., h_{L-1}]^T$. Since there are N blocks in the channel, the channel matrix H is modeled as an NxN circulant matrix, with the received signal as expressed as:

$$\mathbf{x} = \mathbf{Hs} + \mathbf{v} \tag{2.7}$$

Here, $\mathbf{v}$ is assumed to be zero mean white noise. The vector $\mathbf{s}$ is a combination of known training symbols and unknown data. The optimal placement for such training is where the channel undergoes non-ergodic fading considered here [35]. Reference [30] continues on to say that optimally, assuming symbols are placed in clusters of length $\alpha \geq 2L + 1$, this scheme is quasi-periodic. The variable α represents the cluster size in this scenario. It is

also important to note that this placement makes sure that the training is always orthogonal.

Another effect that must be considered is how these training symbols interfere with the data itself, and is the training symbols dependent on the data or even the modulation scheme. Reference [3] examines this aspect and proposes solutions that provides a data independence condition. As explained previously, since the training data is periodic it can be placed in equispaced frequency bins, while data is spread across all frequency bins. Therefore, the pilot must be designed to distort the data vector of the discrete Fourier transform to zero. In the superimposed training data case, this is done by using the cyclic mean of the data. Therefore, all that needs to be done is the removal of the cyclic mean $\mathbf{e} = \mathbf{J}\mathbf{w}$. Note that $\mathbf{J}$ is the Kronecker product of an identity matrix and the fractionally spaced locations of the pilot tones. Therefore, at the pilot frequency only the training symbols are visible for the channel estimation. Formally here is the transmitted, or precoded, result including pilots and data: $\mathbf{s} = (I - \mathbf{J})\mathbf{w} + \mathbf{c}$.

In summary, research on superimposed training focuses primarily on the training symbol generation for a certain type of communication systems design from single transmission to multiple-input multiple-output (MIMO). Unfortunately, little to no physical implementations exists for such systems. This is true because of the synchronization issue that exists when using superimposed training symbols. Since they are directly placed with transmission data it can be difficult to determine their locations in a sequence blindly, which is done in real world systems. This problem must be considered when physical implementations are proposed.

2.4 Spectral Subtraction

Now that methods of reconstructing information distorted by the channel itself has been discussed, we can now focus on spectral removal of known signals without demodulation. Such a technique is needed to improve the effectiveness of equalization operations done

downstream, while limiting corruption to the desired signals themselves. In this thesis a new application for a relatively standard technique was examined, called Spectral Subtraction (SS). The SS technique was first published in 1979 by Steven Boll [36]. SS is formally used to reduce ambient noise in audible sources, improving the overall quality and intelligibility of digitized speech. It is a dominant speech processing algorithm and many extensions including [37], [38], and [39]. Due to the large amount of literature and investigation into the SS process it was assumed to be a solid option for removing unwanted signals in the spectrum.

SS primarily was designed for audio signal processing, small bandwidth signals roughly from 20Hz to 20,000Hz. Many forms of SS exist, but the approach examined here is Magnitude Spectral Subtraction (MSS). It works by first generating an estimate of the noise in the signal itself, which is usually attained at the first first few seconds of the signal itself. This noise is then subtracted, as the name suggests, from the rest of the signal. Mathematically, let us explain this further. The received signal is assumed to be a combination of two signals, the transmitted and the noise itself $y(t) = x(t) + n(t)$. Next the power spectral densities (PSD) are calculated for these components:

$$E\{|Y(e^{jw}|^2\} = E\{|X(e^{jw}|^2\} + E\{|N(e^{jw}|^2\} + 2E\{|X(e^{jw}|^2\}\{|N(e^{jw}|^2\}$$

$$E\{|Y(e^{jw}|^2\} = E\{|X(e^{jw}|\} + E\{|N(e^{jw}|\}$$

Here $Y(e^{jw})$, $X(e^{jw})$, and $N(e^{jw})$ represent the frequency domain transform of the given signal, also x and n are uncorrelated. Since at points when the desired signal is not present in the spectrum, a silent period, the measurement for N is taken and then subtracted from the entire received signal $E\{|X(e^{jw}|\} = E\{|Y(e^{jw}|\} - E\{|N(e^{jw}|\}$. The is noise is assumed to be quite stationary during the signal period. Therefore, the original estimate $\hat{N}$ can be quite accurate.

2.4.1 Residual Noise

As a result of the changes over time in the noise spectrum (whether power or magnitude) around its expected value, there is always some difference between the actual noise and its mean value. Hence, some of the noise remains in the spectrum in the case that the value of noise is greater than its mean and some of the speech spectrum also is removed in the case that the estimate of noise to be greater than the actual value of noise. The latter produces negative values in the spectrum. These negative values are prevented or set to a floor (sometimes zero) using different techniques. The overall effect puts noise in the output signal known as residual. The narrow band relatively long-lived portion of residual noise is sometimes referred to as musical noise [40].

A close examination of musical noise, shows that peaks and valleys exist in the short term power spectrum of white noise. These frequency locations for one frame are random and they vary randomly in frequency and amplitude from frame to frame. When a smoothed estimate of the noise spectrum is subtracted from the actual noise spectrum, all spectral peaks are shifted down while the valleys are set to zero. Therefore, after this subtraction sharp peaks remain in the noise spectrum and pre-existing ones can be sharpened. The wide peaks are generally estimated as time varying broadband noise. The narrower peaks, which are relatively large spectral distances because of the deep valleys that define them, are perceived as time varying tones which are generally referred to as musical noise [41].

Therefore, [36] continues by introducing a "smoothing" technique before the signal is convert back into the frequency domain. Two additional parameters are introduced: The parameter α the over-subtraction coefficient, and β the noise floor lower bound. α is used to provide a more aggressive subtraction to the signal, attacking high peaks which are generally a result of high noise and an inaccurate initial estimate. The second parameter β is used to fill in the valleys of the signal. Since if an over-subtraction takes too much signal it can cause valleys in the spectrum below or above the zero threshold. This value is used to simply quantize values within its +/- limits. As a result these operations together produce a smoother signal removing much of the residual noise from just a plain subtraction. Reference [36] provides several results examining the benefits of such a technique.

2.5 Software Defined Radio

Now that the signal processing techniques have been discussed, a platform for implementation is needed. The alley chosen for this thesis is to utilize a new hardware frontier called Software-Defined Radio, which will be discussed in this section.

For the past two decades there has been a paradigm shift is the definition of a radio device. The conversation has to do with the question of where hardware ends and where software begins. The term Software Defined Radio, coined by Dr. J. Mitola III, defined as a set of digital signal processing (DSP) primitives, a meta-level system for combining the primitives into communication system functions (transmitter, channel model, receiver, etc.), and a set of target processors on which the software radio is hosted for real-time communications [42]. Dr. Mitola understood how software provided the flexibility that hardware never could, and as time made it more mailable SDR would become dominant.

SDRs can be flexible enough to avoid the "limited spectrum" assumptions of designers of previous kinds of radios, in one or more ways including: Ultrawideband transceivers, cognitive radio, dynamic mesh networks, software-defined antenna arrays among others [43]. One of the first SDR implementations was a project called "SpeakEasy". The original purpose of SpeakEasy was to use programmable processing to emulate more than ten existing military radios, operating in frequency bands between 2 MHz and 2 GHz [44]. Therefore with this single radio, the operator could talk to ten radios operating under ten different standards. As simple enough idea, but unfortunately the implementation left much to be desired. For example, physically the device encapsulated the entire back of a common pickup truck [44]. This might be great for a ground station that does not move, but for a mobile unit this was highly impractical. Secondly, in 1992 field programmable gate arrays (FPGA) required significant time, comparatively to re-flash or change their operational parameters. Again, this also limited SpeakEasy's flexibility.

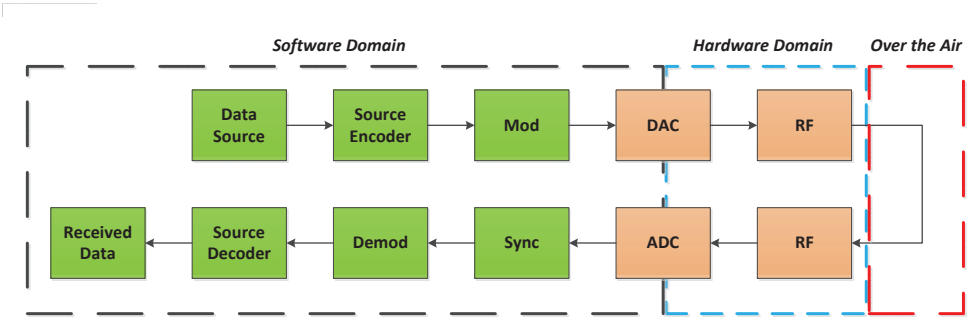


Figure 2.8: Software defined radio pushes all the adaptive elements and data manipulation operation into software. The goal of SDR is to provide or define all of the radio operation in software.

Today, the main target implementations are within cellular base stations and military applications such as the JTRS project. The JTRS or Joint Tactical Radio System, was a program of the US military to produce radios that provide flexible and inter-operable communications [45]. Examples of radio terminals that require support include hand-held, vehicular, airborne and dismounted radios, as well as base-stations[45]. Again, this project still has limited results and many setbacks have occurred. Commercially, from a wide spread penetration standpoint, SDR is still many years away due to the size and cost of current devices. The two barriers to this are speed and size. To provide enough data throughput, modern SDRs need to quite large physically, which is a serious drawback in many applications. Aside from these limitations, SDRs provide excellent flexibility especially in a laboratory and proof of concept environment. Rapid prototyping is an obvious place where such radios shine, allowing massive changes without hardware modification. To support this flexibility several software packages have been constructed around the SDR concept, allowing for aggressive prototyping. The two examples discussed here were selected because of operability with the selected hardware, which will be discussed future in chapter 3.

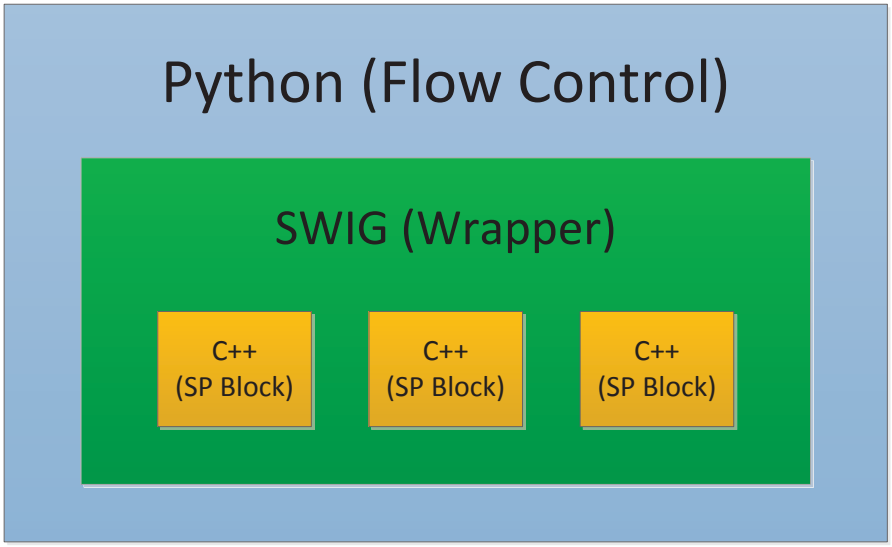


Figure 2.9: GNU Radio code structure based on signal processing C++ blocks and controlling, through SWIG, Python layers.

2.5.1 GNU Radio

The first software package to be discussed by this thesis is GNU Radio. GNU Radio provides the reconfigurable signal processing blocks that are necessary for software defined radios. GNU Radio is an open source project allowing for SDR developers to develop unique signal processing blocks and SDR systems. GNU Radio was started in 2001, originally forked from the SpectrumWare project developed at the Massachusetts Institute of Technology [46]. Since 2001, the code base has undergone massive changes, containing almost no code from the original SpectrumWare project. Physically the code consist of three languages Python, C++, and SWIG. Python provides the overarching control of the system or program, while C++ provides the actual signal processing blocks and mathematics. SWIG is a wrapper for C++ which allows Python to dynamically wrap around C++ and control or compile with it. A diagram below better illustrates this architecture. It is also important to mention that there as significant paradigm shifts in the community, pushing more and more code to Python rather than C++, due to its easier programming syntax and structure.

GNU Radio provides a very structured framework of flow design. Data processing segments are extremely self contained to minimize error propagation during system debugging. Since the software is open-source full access to all code is provide, giving low-level access to all operation within GNU Radio. Much of the actions have been abstracted to limited the knowledge of the lower layers, but if specific actions are required for an application. Then serious depth or knowledge is needed about the overall project's structure, which is quite overloading.

2.5.2 MATLAB

MATLAB is an extremely well known engineering, mathematical, biological, and financial software suite. MATLAB provide massive data leverage and advanced communication system models and algorithm for significant data processing. Since 2007, they have also provided hardware compliance with specific SDR platforms through their Simulink platform, and more recently within MATLAB itself [47]. This thesis primarily utilizes the signal processing and communication system aspects of MATLAB, since MATLAB cannot fully utilize all aspects of the chosen hardware. It is important to note under alternate constraints, MATLAB can provide adequate performance directly interfacing with hardware, especially when accessing its targeting features seen here [48]. Figure 2.10 shows an example of a common MATLAB SDR model through Simulink.

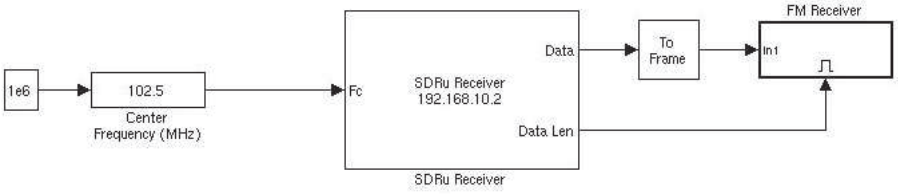


Figure 2.10: Sample SDRU MATLAB model created in Simulink. This model has the ability to demodulate and play FM radio stations when using a USRP device.

MATLAB provides allow connection to a USRP device through MATLAB directly of

with Simulink. The device simply acts as a real-time data source complementing their signal processing and communications toolboxes very well. Rather than full implementations like GNUR Radio, MATLAB focuses on signal analysis rather than real-time performance. The primary limitation is the single threaded nature of MATLAB and Simulink, which is slowly being improved upon for the SDR application.

2.5.3 Reference Comparison

It is important to compare GNU Radio and MATLAB, from a user's perspective they perform quite differently. Firstly GNU Radio, is extremely fast, will the ability of sustaining the maximum throughput of the selected hardware. GNU Radio is also multi-threaded, and while maintaining high throughput and complete background tasks on multi-core machines quite easily. This performance has a cost, comparatively GNU Radio has an extremely learning curve and debugging can be challenging. However if you need the performance, GNU Radio is your option, providing significantly more advanced hardware support in SDR implementations. If data analysis is more heavily desired MATLAB is the obvious choice. MATLAB provides easy and advanced data visualization functionality, and built in tools for analysis. Since MATLAB does not compile itself normally, it can be much easier to debug and solve problems. MATLAB's syntax provide similar data manipulation, especially in communication system primitives. Therefore, it can be a rather simple choice, speed or ease of use.

2.6 Summary

This chapter outlined and examined the topics of jamming and anti-jamming techniques, and provided a foundation in communication system theory and advanced equalizer design. Secondly it setup an understanding of Software-Defined Radio, the power of such an architecture, and examples of implementations and existing software for future designs. Next, this thesis will consider a new anti-jamming technique and design an implementation of

such a system. After the implementation is investigated, the result of specific experiments on such an implementation will be analyzed.

Chapter 3

Implementation

3.1 Overview

This chapter outlines the proposed implementation of a receiver design, for wide-band jammer scenarios and low-mobility situations. An adaptive signal processing software solution for mitigating the effects of both intentional and unintentional jamming (including wide-band jamming) through a combination of three techniques. These include: antenna subset selection, spectral subtraction, and signal separation, which work in conjunction with one another to extract specific transmissions from a mixture of intercepted wireless signals. The goal of the proposed solution, originally called BLInd Spectrum Separation (BLISS), is to enable reliable, high throughput, and robust end-to-end wireless communications, especially high capacity multimedia (voice, data, imagery) transmissions. This was later rename to AS^6. In particular, the focus of the proposed work is the so-called "disadvantaged user". These users are generally considered limited in transmission and processing power such as small-deck combatants, submarines, unmanned air vehicles (UAVs), dispersed ground units in urban and radio frequency (RF) challenged environments [49]. The previous research is also discussed for each section and implementation consideration are examined from this work.

The AS^6 solution integrates three well-known adaptive signal processing algorithms

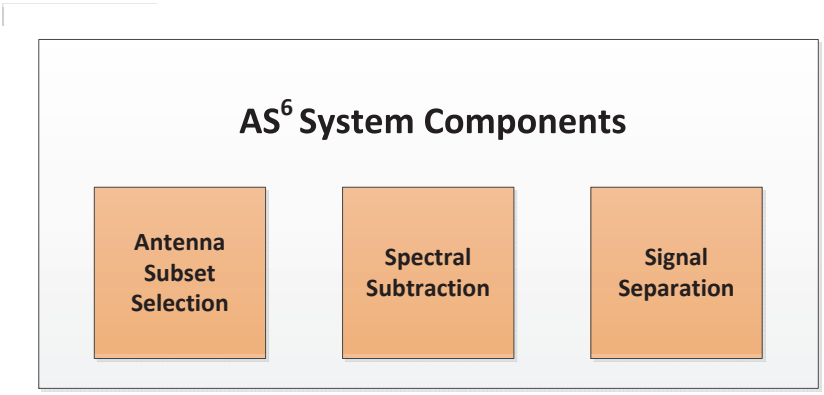


Figure 3.1: The AS^6 system is made up of three primary blocks: antenna subset selection, spectral subtraction, and signal separation. These combine to provide reliable signal reception in low-mobility spectral environments

found in the open literature: antenna subset selection [50], spectral subtraction [36], and signal separation [51]. Each of these algorithms is employed within the AS^6 framework in order to enable the process of extracting individual transmissions intercepted from several mixtures of wireless signals. Although signal separation can readily extract transmissions under ideal conditions, the AS^6 system is aimed at harsh spectral environments consisting of many users and in some cases jamming devices. Therefore AS^6 will not provide adequate signal separation for robust throughput. Hence, the other two algorithms, spectral subtractions and antenna subset selection, will aid in this effort.

In previous sections it has been understood that current anti-jamming techniques cannot compensate in deterministic wide-band jamming scenarios. These scenarios must be throughly understood before a practical solution can be provided. For this thesis, the worst case scenario will be considered for the jamming device. Since wide-band jamming devices are difficult to build, a band limited interferer was chosen instead to implement while placing restrictions on the communicating transceivers themselves. For simplification a narrow-band jammer will be considered as an adversary, and the communicating devices cannot frequency hop thus remaining on the same frequency as the jammer. The jammer has an identical modulation scheme as the friendly radios and the constellation is in phase.

Finally, the jammer is assumed at a similar distance and transmit power as the friendly communicating devices. Under these conditions, the jammer is completely non-orthogonal and historically impossible to remove.

This chapter is broken down into several sections which include a system level overview, the hardware and software chosen, signal removal evaluation, the superimposed equalizer design, and the antenna subset selection work. Each of the systems that makeup AS^6 have different purposes and goals allowing them to tackle different problems that occur. It is important to note that these systems are at differing stages of development due to the limited time and initial development put into these blocks.

3.2 System

To provide a more straight-forward explanation of the AS^6 system, it is appropriate to provided a system level overview. The system's original purpose was to remove the effects of narrow and wide-band jamming. It accomplishes this goal through a series of processing blocks and a selection block. These blocks include: the antenna subset selection (AntSS) block, spectral subtraction block, and finally the signal separation block. The figure below shows the interconnections between these blocks and certain modification were made from the original design of the system due to practical constraints. These changes will be brought fourth as the blocks themselves are discussed in detail. Since collaborating research group is responsible to the AntSS block, it will not be throughly discussed by this thesis, but its fundamental purpose will be examined.

The first step in the AS^6 system is to pass through the AntSS block. Physically this block is equipped with many antenna in groups of 4. As the block title portrays a subset of these antennas will be selected and they will be passed on to the next block. In particular, a $2^M - to - 2^N$ down selection from an array of receive antennas to a set of BLISS receiver inputs is performed. Each individual AntSS board provides 4-to-2 antenna down

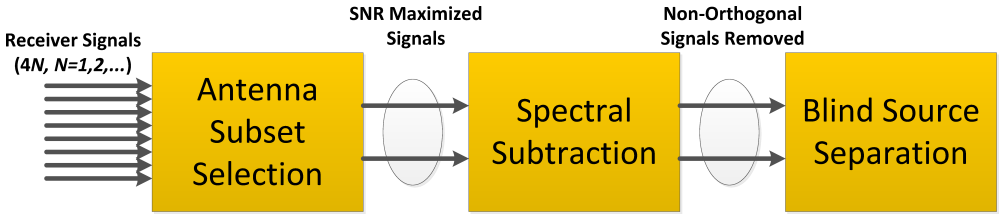


Figure 3.2: Overview of the original BLISS system as present in the proposal document [1], outlines three basic blocks: antenna subset selection, spectral subtraction, and blind source separation.

selection through a set of RF switches. The goal of AntSS is to provide spatial separation through an array of antennas maximizing the SNR of the wanted signal. It is important to note that the antenna spacing must be adequate to provide enough separation or independence, depending on the operating frequencies or wavelength of the signals themselves. Once the appropriate antennas are selected two signals are to the spectral subtraction block.

The Spectral Subtraction block is next, which is used to removal known unwanted signal from the spectrum so the source separation block and work properly. The original design of the spectral subtraction block is to use an existing audio technique of removing noise or signals in the frequency domain through a subtraction and smoothing technique. This technique was discussed previously in the background section, therefore its historical literature will not be examined further. To enable the removal of unwanted signals, the Spectral Subtraction block maintains a database of known power spectral densities (PSD) of common modulation schemes. A recognition system would be implemented to automatic identification of the interfering signal and the block would simply subtract it out, through its already known estimate from its database. Next, the subtracted signal is passed to the Signal Separation System, where the signal is unmixed.

The Signal Separation block separates signals when only their mixtures are observed. The operation is called blind, since the signal sources and mixing procedure are unknown to the receiver. Under several conditions, this constraint cannot be completely upheld. This is true because the solutions needed to solve such an event become generally intractable.

An initial approach in this project was to use a technique called AMUSE (Algorithm for Multiple Unknown Signals Extraction) [51]. AMUSE works by first collecting an estimate of the covariance matrix of the received signal $R_y = E[yy^\top]$. Then computing the singular value decomposition of that covariance matrix, and estimating the number of sources, noise variance σ and singular values $\psi_1, \psi_2, ..., \psi_N,$. Next a data transform is done $z = Cy$, where $C = diag(1/\psi(1), 1/\psi(2), 1/\psi(3), ..., 1/\psi(N),)$. Then an offset τ is selected and $R_z = E[z(t)z(t-\tau)^\top]$ is estimated. From this estimate an eigenvalue decomposition is done on $\frac{R_z(\tau)+R_z(\tau)^\top}{2}$ and the singular values are collected in Σ. Finally an estimate of the signal sources can be computed $\hat{s} = \Sigma Cy$.

It is important to note that for simplicity the mixing matrix for the original proposed solution involving AMUSE is generally constructed as a linear time invariant (LTI) system. There is some activity occurring with nonlinear mixing, but that was considered outside of the scope of this problem.

3.3 Hardware and Software Platforms

Before any implementation was considered a platform needed to be chosen for the end result. This selection provided the work flow-path for the implementation, eliminating many options. As discussed in previous chapters, the end result wants to leverage the power of Software-Defined radios (SDR). The hardware platform chosen was the USRP2 designed and built by Ettus Research [52]. These radios are readily available in the Wireless Innovation Laboratory and since the number of radios required for the design was still unknown, it was an obvious choice. There are several software packages that support the USRP2 hardware and several will be examined in this chapter.

The USRP2's or Universal Software Radio Peripherals (Version 2) is a comparatively inexpensive hardware platform for software radio, and is commonly used by research labs, universities, and hobbyists [53]. The USRP2 connects directly to a host computer through a Gigabit Ethernet link, which relays baseband sample that have been receiver or to be

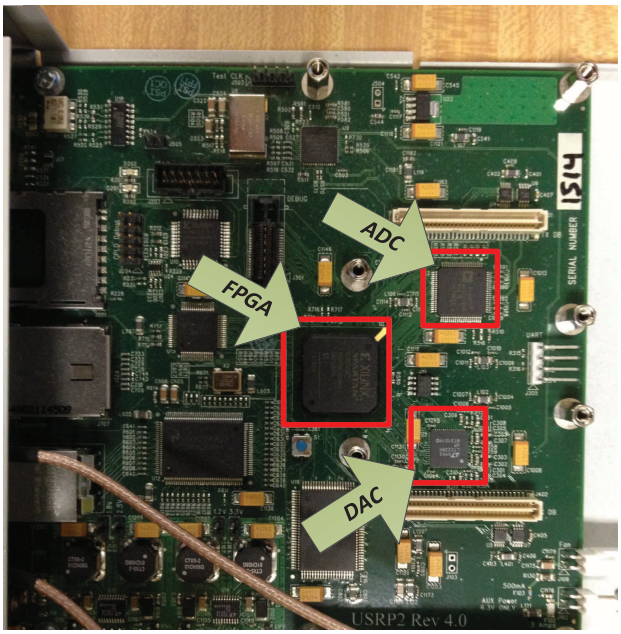


Figure 3.3: Full USRP2 hardware with daughtercard attached. Outline are the three primary IC's: the FPGA, ADC, and DAC.

translated. The motherboard provides the following subsystems: clock generation and synchronization, FPGA, ADCs, DACs, host processor interface, and power regulation. Several of these component are seen in the image in Figure 3.3. These are the basic components that are required for baseband processing of signals. A modular front-end, called a daughtercard, is used for analog operations such as up and down conversion, filtering, and other signal conditioning. By replacing this RF daughtercard many different frequency ranges can be examined.

The information flow is important to understand within the physical radio. This SDR block diagram shown below, outlines the common tasks done by the: daughtercard, FPGA, DAC/ADC, and host computer. Since the FPGA is programmable the operations can change if desired, but the three dominating software packages that utilize the USRP2 follow this structure. Beginning on the far left of the diagram and continuing to the right, at the daughtercard are RF emissions are received and transmitted. The daughtercard also contain mixers that translate the signal to an intermediate frequency. Next come the

dual 100 MS/s 14-bit ADCs, dual 400 MS/s 16-bit DACs, two digital down-converters with programmable decimation rates, and two digital up-converters with programmable interpolation rates [52]. These are located on the main-board of the USRP2 itself. The FPGA is a Xilinx Spartan 3 XC3S2000, whose current FPGA software is 59% free in general logic but only 3% free in memory. Note that the FPGA also does not have any DSP resources. The limited memory left in the USRP2 FPGA severely limited any additional development. As a result, newer models, such as the N210, have an upgraded FPGA [54].

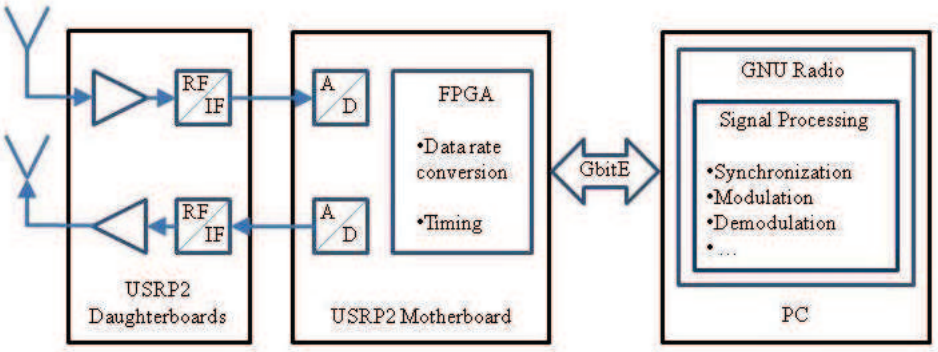


Figure 3.4: USRP system block diagram outlining the two operational sections of the USRP itself and their tasks, as well as the PC connected to radio [2]. All adaptive functions and upper layer control is done in the PC, while the radio usually only provides access to the RF environment.

The data itself contains several pieces of metadata in a frame. RX metadata structure for describing sent IF data includes time specification, fragmentation flags, burst flags, and error codes. The receive routines convert IF data headers into metadata [55]. Such metadata can be used to indicate the position and FPGA timestamp associated with the sample that corresponds to the start of the underlying frame. By default, existing blocks will transparently propagate any attributes contained on their input streams to their output streams. Blocks that use the attributes can query their input streams to locate all (key, value, offset) tuples in the region of the stream that they are currently working on in their "work" method. Likewise, blocks can copy, add or delete attributes on their output streams [56]. This knowledge is extremely useful when doing multiple receive antenna arrays when alignment is necessary, or in any situation where fine timing information is required.

With the USRP2 hardware, several software options are available including: GNU Radio, MATLAB, LabVIEW, and several custom solutions. MATLAB and GNU Radio have already been discussed, therefore the selection between them shall be discussed. Since this system is a MIMO implementation, signal alignment is a requirement. At the moment MATLAB does not support sample alignment in a multiple USRP2 system. Nevertheless, the sample alignment is possible through either external means such as through an external clock or through the option chosen here, the MIMO cable. The MIMO cable, a picture of it can be seen in Figure 3.5, is a standard 16-pole flatcable to connect tvrx, basic-rx or dbsrx boards. Of this 16pin flatcable only two pins are used (io15 and ground) [57]. An image also of the combined dual radio source block can be seen in Figure 3.6 from GNU Radio. With this requirement GNU Radio must be used for direct access with the USRP2. As well, full implementation of the systems blocks were first attempted with GNU Radio. Fortunately, if necessary, data can be passed to MATLAB for signal processing from GNU Radio through the use of the file blocks and a script located in Appendices B-E.

3.4 Spectral Subtraction

Now that a formal system level approach has been presented and hardware setup chosen, a more detailed understanding of the blocks themselves can be examined. Spectral Subtraction will be discussed first then the Signal Separation block. The goal of the Spectral Subtraction block is to removal signals to allow the Signal Separation block to work properly, or to properly recover the desired signal. As discussed previously signals would first need to be identified and then removed based on information supplied in a precompiled database of known signals. The technique to remove such signals is called Spectral Subtraction, which primarily takes place in the frequency domain. This approach only relies on known PSD's of the interfering signal. All previous research into this topic area was done by Robert Over, using the original definition, providing MATLAB and C++ simulations. Unfortunately, the C++ simulations, which were designed to be push to the real-time pro-

Figure 3.5: USRP2 MIMO cable connecting two USRP2 devices. This cable is used to couple local oscillators together to phase allign output samples.

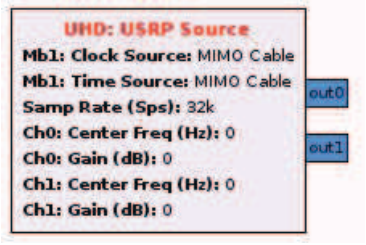


Figure 3.6: GNU-Radio MIMO enabled source block, used to interface with two connected USRP2's.

totype, could not be utilized or adapted because of their incompatible library functions with GNU-Radio. This research was used as a foundation to provide a viable and implementable solution for Spectral Subtraction. Initially this technique seemed quite sound, but further investigation proved otherwise.

Initial simulations were created to examine this spectral estimation technique at RF frequencies rather than the standard audio frequencies for which Spectral Subtraction is

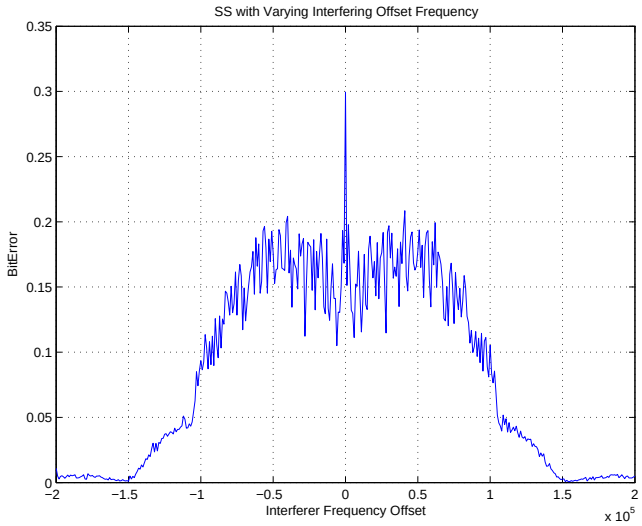


Figure 3.7: Original Spectral Subtraction technique results showing massive error, especially when the interferer is directly ontop of the desired signal. Only when the interferer moves out of band does the error decrease to an acceptable value.

formally used. Only two signals were used in these simulations, both utilized the same modulation scheme and pulse-shaping filters. The signals were chosen to be non-orthogonal, since orthogonal signals are inherently non-destructive. The frequency of the interfering signal was varied, along with the over-subtraction parameter α and quantization floor β. Through experimentation α worked best at a value greater than 10, and β worked best between 0.05 and 0.2. The graph in Figure 3.7 shows the bit error rate (BER) as the interferer is shifted across the wanted signal in frequency.

As you can see this spectral subtraction technique operates extremely poor when the signal are overlapping at all. The reason system performs well at large frequency shifts is due to the bandpass filter which is used before the signal is quantized. The reason the result is poor is because the estimate is largely incorrect. Since the subtraction only utilizes the PSD's of the signals, half of the information is completely ignored. This results in an inaccurate estimate. The problem with traditional Spectral Subtraction is that its results are subjectively evaluated or done, which isn't accurate enough in a digital communication system. Other evaluations uses such metrics as SNR, which can be quite deceptive espe-

cially in digital communications. Such examples are covered in [58], [59].

Since the initial simulations for traditional Spectral Subtraction proved inadequate, other options needed to be explored. First, the problem needed to be an analyzed further for better understanding, then the appropriate solution could be formulated. Since the interfering signal and the wanted signal are non-orthogonal to one another they will share dimensional space, in this case the signals are in phase with one another. Therefore, both planes real and imaginary must be considered. Non-orthogonal signal removal is a common task in communication system, which is done primarily by equalizers. Therefore, an equalizer approach was considered next.

3.4.1 Equalizer Approach

The equalizer approach used in this Spectral Subtraction approach is a Least Means Square (LMS) equalizer, utilizing training data used in the front portion of each transmitted frame [26]. This a common equalizer used in practice, allowing for future translation into a realized implementation. The LMS equalizer was also chosen for it robustness to noise, which is a weakness of such equalizers as the zero-forcing equalizer, and requires no matrix inversion such as the Least Square (LS) equalizer. For proof of concept the entire data-stream was used as training data, which provides the best results of any given channel for an adaptive equalizer, since the maximum knowledge is gained about the channel for each frame received. The results below show the BER as the signals pass over one another in frequency, similar to the previous evaluation using traditional Spectral Subtraction.

As you can see the Figure 3.8, the equalizer approach doesn't provide any improvement beyond the traditional Spectral Subtraction approach. Figure 3.8 shows massive errors when the frequency of the jammer is within kilohertz of the communicating channel. The reason the error decreases on the sides of the transmission, is due to the lack of any overlap in the frequency domain of the interferer and the desired signal. The problem with using traditional adaptive equalizers is that they can only be used with a comparative slowly fad-

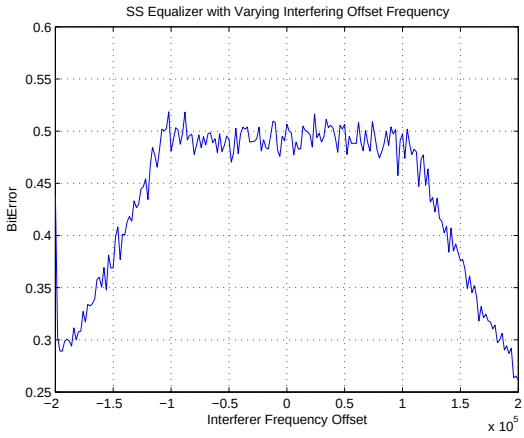


Figure 3.8: Spectral Subtraction equalizer test shifting jammer across a large frequency range, causing overlaps or interference with the desired signal.

ing channel. Since knowledge learned from the training data can be applied at the earliest to the next frame, if the interference changes enough it can render the equalizer useless. This rapidly changing spectrum or energy within the spectrum is unfortunately a common characteristic of jammers. Even though this approach failed it provided an important observation and incite into the requirements and scenarios in which jammers can be overcome.

The important conclusion drawn from the previous experiment is that the when signals are orthogonal the receiver needs to be able to predict what data or energy is being transmitted at a given time. Therefore, the jammer problem must be constrained future. As a result two jammer conditions will be defined. The first condition is that the jammer's modulated data or energy is completely known to the receiver and the second is that the data sequence repeats with period being small. The larger the period the more resources the receiver will need to devote to its determination and evaluation. The sequence being completely known to the jammer is a reasonable assumption; primarily if the jammer is friendly, as discussed previously in this thesis, then that knowledge can be readily available.

Now that the jammer scenarios have been defined further they can be evaluated. The first will be when the data sequence of the jammer is completely known to the receiver. The

approach here will be to synchronize with the interfering signal, so the interferer will simply be subtracted off. To synchronize the signals a mathematical tool called correlation will be used. Correlation is a common tool used in synchronization in communication systems when looking for known symbols in a stream of data. Equation (3.1) for cross-correlation simply passes signals over one another, the resulting sequence creates peaks where the data is most correlated.

$$R_{XY}(t_1, t_2) = E[X_{t_1} Y_{t_2}][60] \tag{3.1}$$

An example of two sequences being cross-correlated with one another can be seen in Figure 3.9, where the larger first signal contains the smaller second signal. The peaks, in red, are the locations the desired signal is most correlated, and the index of these large correlations is the location of the desired signal with the larger mixed signal. Therefore from this data the location of the start of the interferer's data can easily be located and removed. A simulation was created with this design in mind, with a unique result. Since the signals are frequency shifted over one another, when there frequencies match, it produces the best result, but as soon as they are offset, errors start to occur. This can be compensated for using a complex exponential multiplied by either the received or cataloged waveform. This will induce a frequency shift canceling out the shifting signal. Therefore the error will primarily be a function of the noise itself.

This simulation was also repeated but with prior knowledge of the interferer's carrier frequency was assumed known. As expressed previously the subtraction estimate was frequency shift to match the interferer, and then finally subtracted off. Figure 3.11 shows the overall desired result, with uniform error across all offset frequencies.

Further analysis was done to provide a solid understand of the new requirements of the system and its limitations. The first is the precise understanding of the interfering signal itself. Since it is assumed that the signal modulation and data stream are known, attention can be shifted to the other issues that can occur, which are the non-idealities of the system. The first is the power constraint or effects. When subtracting the interferer, accurate power

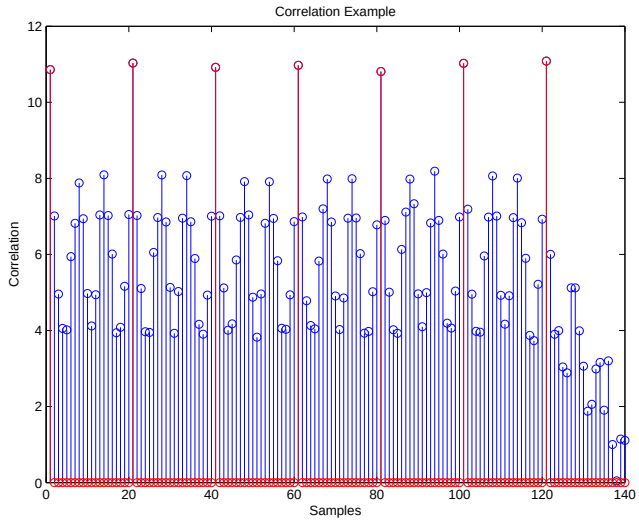


Figure 3.9: Autocorrelation of random signal and signal subsection. The red peaks show the locations of the subsection embedded in the random signal.

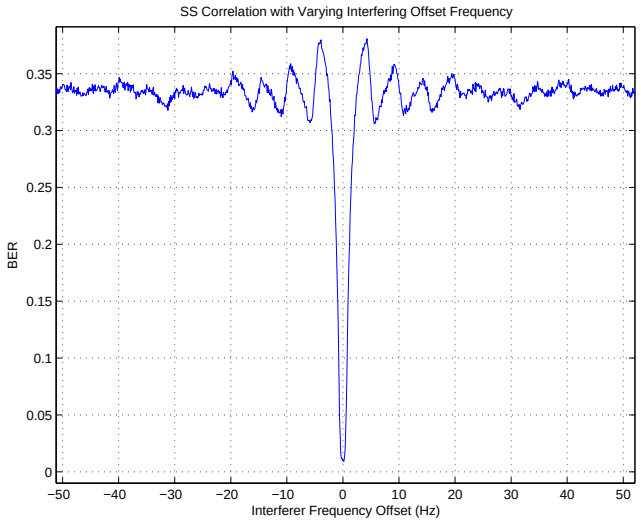


Figure 3.10: Spectral Subtraction correlation method, with fixed subtraction estimate frequency and shift actual interferer center frequency.

measurements must taken so the estimate is scaled correctly for the environment. By accurately estimating the power of the interferer, minimal energy will be subtracted from the the desired signal, which is also in the band. An examination of the incorrect estimation

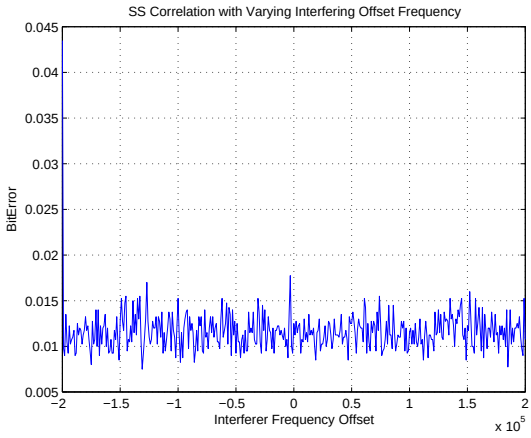


Figure 3.11: Spectral Subtraction correlation method, with matched subtraction estimate frequency to shifting actual interferer center frequency.

of the interfering signal's power was examined in Figure 3.12.

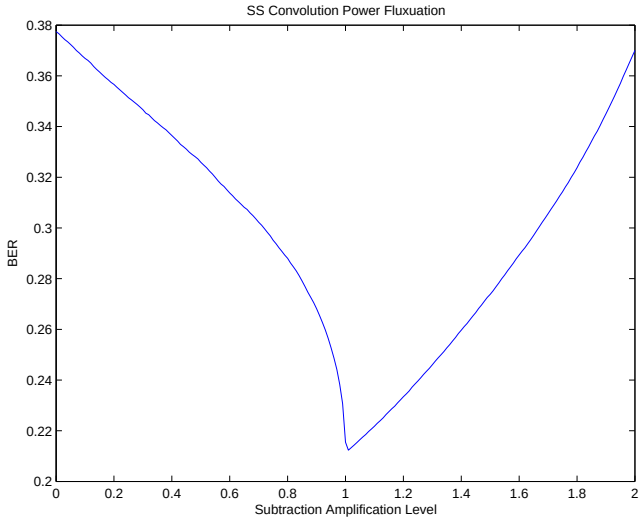


Figure 3.12: Spectral Subtraction correlation test with varying power level scaling interfering signal, while examining the BER of the decoded desired signal. This shows the sensitive to power difference of the two signals.

Besides power, frequency and timing offsets were examined. These are especially concerning because the signal cannot be demodulated when in operation for feedback information. This is because it must be fed into the signal separation block for equalization.

Therefore a timing recovery mechanism cannot down-sample the signal either. The timing recovery and carrier recovery mechanism therefore needed to be unobtrusive as possible to the signal. As a result frequency and timing shifts were examined to understand the sensitivity of the subtraction, and how accurate the recovery mechanism needed to be.

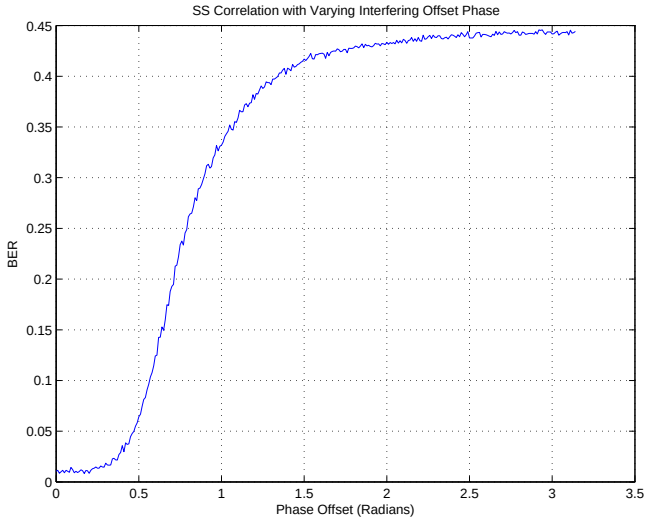


Figure 3.13: Spectral Subtraction with phase offsets of interfering signal, showing large error in the decoded desired signal.

As you can see from Figure 3.13, the subtraction mechanism is very sensitive to phase timing offsets. If a full bit-shift occurs then the entire system falls apart. This is a rather large problem and must be considered when attempting such operation on non-simulated data. The frequency sensitivity can be seen from a previous figure, Figure 3.10. This problem was foreseen because of the time variance or sensitivity of the signals themselves. Therefore, adaptive compensation methods must be utilize in the final implementation to fix these corruptions.

3.4.2 Hardware Consideration

Now that a viable subtraction technique has been determined, the final implementation for the Spectral Subtraction block can be realized. As discussed in the Hardware and Soft-

ware Platform section of this thesis, GNU Radio was the first to be examined because of its real-time attributes. This was quite an involved process requiring many weeks of trial and error. The first implementation was entirely written in C++, which is the recommended language for signal processing blocks in GNU Radio.

Since C++ within GNU differs from many modern programming styles a code implementation or route was taken to ensure accuracy and speed up development. Therefore all coding was done with C++ itself, using no GNU Radio built-in libraries [61]. To allow for matrix operations the Aramdillo C++ Library [62] was imported and provided needed vector operations such as correlation and faster mathematical functions instead of having to rewrite common search operations. This library would also be needed for the Signal Separation block, therefore coding with Armadillo would provide the knowledge for future implementations needed in that block. From the standard C++ implementation results were compared with MATLAB, and the code was ported into GNU Radio.

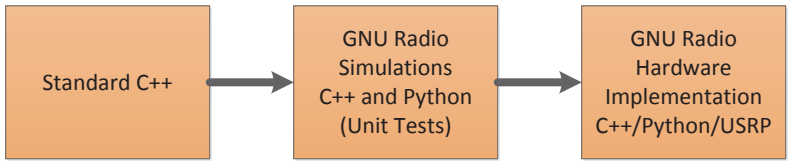


Figure 3.14: GNU-Radio code implementation workflow first beginning all work in standard C++, then transitioning to C++ with only GNU-Radio supplied libaries, and finally to full implementation flow-graph with USRP.

GNU Radio C++ are basically written by first creating test cases and writing your code until they are solved. This is a common practice among the programming community and provides a definitive endpoint to the code itself. The code was written and compiled successfully but unfortunately the python wrapper called SWIG [63], which GNU Radio uses to interact with the C++ block through python, was unable to export the library. This is an undocumented problem within GNU Radio and was only identified through discussions directly with the GNU Radio core development team. Therefore another approach had to

be considered.

The next option was to use python itself for signal processing. This is approach was primarily developed by Josh Blum, one of the core developer of GNU Radio. It isn't recommended due to speed issue, but it is quite easier to implement and debug for those familiar with python. As a result the previously C++ code was ported to python standards libraries and then to GNU Radio. The NumPy libraries were used within python. NumPy is the fundamental package for scientific computing in Python [64]. It like the armadillo library provides matrix operations such as correlation. Again under the Python standard libraries with NumPy the results were verified with MATLAB. Then the code was port to GNU Radio.

Again more problems occurred, which inhibited progress. The signal processing block was written as a subprocess using the queuing system built into GNU Radio. Queuing provides barriers between the connected blocks. Therefore they can run freely, limiting bottlenecks. The system built would operate correctly for several hundred samples but would eventually segmentation fault. Several attempts to fix this error with even architectural changes to the code. Finally the lead developer of GNU Radio was consulted, Tom Rondeau [65], but he was also unable to determine a solution. The assumed problem was a type casting occurring within the queue itself, that would eventually accumulate and cause a segmentation fault.

With these setbacks, it became necessary to look beyond GNU Radio and utilize MATLAB for signal processing. Therefore the decision to load captured signals from GNU Radio and process them in MATLAB. Although not real-time, but this configuration would process the data appropriately. The simple GNU Radio needs front-end is important because it allows tight synchronization between multiple receive antenna, which is a require of the original design of the system. The GNU Radio model can be seen in Figure 3.15, the modulator block has been customized to remove the quantization, producing floating point results.

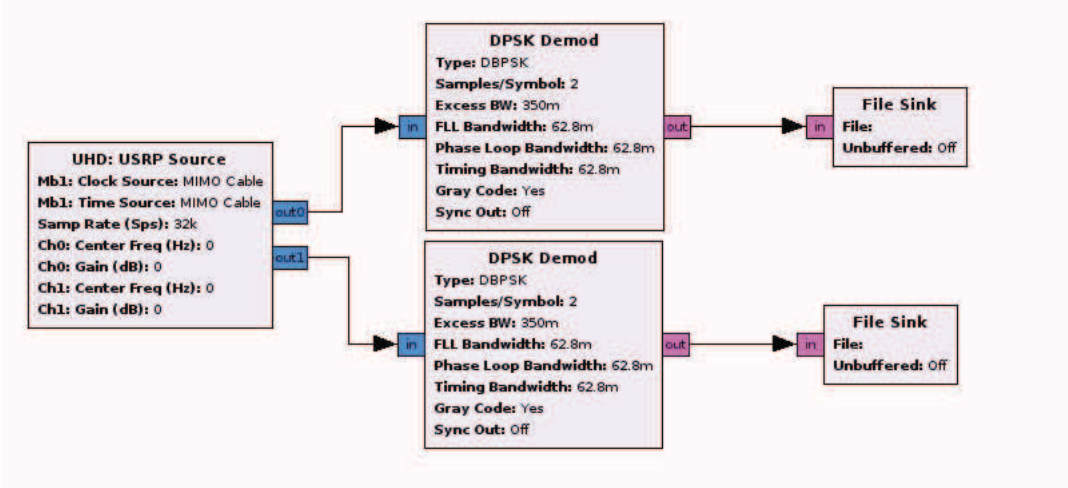


Figure 3.15: GNU-Radio receiver portion of final prototype, utilizing custom demodulation blocks removing specific algorithms and quantizers. The MIMO USRP source block is also shown, which has yet to be implemented in MATLAB/Simulink.

3.4.3 Non-deterministic Scenarios

For completeness it is important to discuss the scenarios when the interferer's modulated data is unknown but repetitive with a small period. The approach to estimating short sequences is a rather obvious one, an autoregressive algorithm is used to predict samples. The simulation here, which was just used for proof of concept, uses a linear predictive filter. The filter determines coefficients of a forward linear predictor by minimizing the prediction error in the least squares sense[66]. It finds the coefficients of a pth-order linear predictor (FIR filter) that predicts the current value of the real-valued time series $\mathbf{x}$ based on past samples.

$$\hat{x}[n] = -a(2)x(n-1) - a(3)x(n-2) - ... - a(p+1)x(n-p)$$

For the linear predictive filter to operate effectively the number of filter taps must be equal to or greater than the period of the repeated sequence. If the number of taps is smaller it cannot capture the randomness the in the interferer's data.

3.4.4 Over the Air Implementation Considerations

When moving towards a real implementation of the Spectral Subtraction block, the non-idealities introduced by the environment needed to be considered. These include frequency and phase shifts, as well as timing offsets. Certain considerations needed to be made as well, since instantaneous changes occur when signals overlap. Therefore a more advanced control scheme needed to be constructed around the common signal compensation or correction. The basic idea used here is a receiver within a receiver, one for each signal received. This will be discussed in detail.

The system assumes that the jammer is always present within the environment therefore it was concluded that the jamming signal should be synchronized with first, be removed and then all that remains should be the wanted signal. There is were the receiver within a receiver design comes in, since first the interferer will be synchronized to, utilizing phase and frequency recovery and then timing recovery. Unfortunately such an implementation isn't as straight forward as expected. Since when both signal are present in the spectrum it is impossible for these algorithm to operate correctly. This is due the fact that they cannot separate one signal from the other. For example, phase information cannot be accurately calculated in the presence of two signals. Therefore modifications need to made, which is where a controlling mechanism comes into play.

When multiple signals are in the environment the compensation algorithm learn incorrectly; as a result, a decision was made to pause these algorithms when both signals were present and continue when the signal interferer was only present. This operation relies on two assumptions, the first is that both signals are present for short periods of time which can be controlled. The second is that the calculated offsets of cause by the environment don't rapidly vary during the periods of time for which the two signals are visible. This assumption is quite reasonable especially with relatively non-mobile transceivers, which was assumed in the original documentation.

To accomplish this algorithm holding mechanism, energy detection was chosen to be uses for its simplicity. Below you can see an image of the jammer signal by itself and the combined signals. A large increase in energy or step can be seen, which can easily numerically detected. Energy is calculated using the following equation: $E_s = |x(t)|^2 dt$. This was implemented with a moving average filter with a small window to reduce spurious changes due to signals gaps or outliers. In practice the average peak energy level of the interferer is first calculated, then when the power in the spectrum increases to a level roughly 1.5x of the original level, the compensation algorithms are halted. A simple technique, which is commonly uses but in the inverse fashion. An example of the combined and signals can be seen in Figure 3.16.

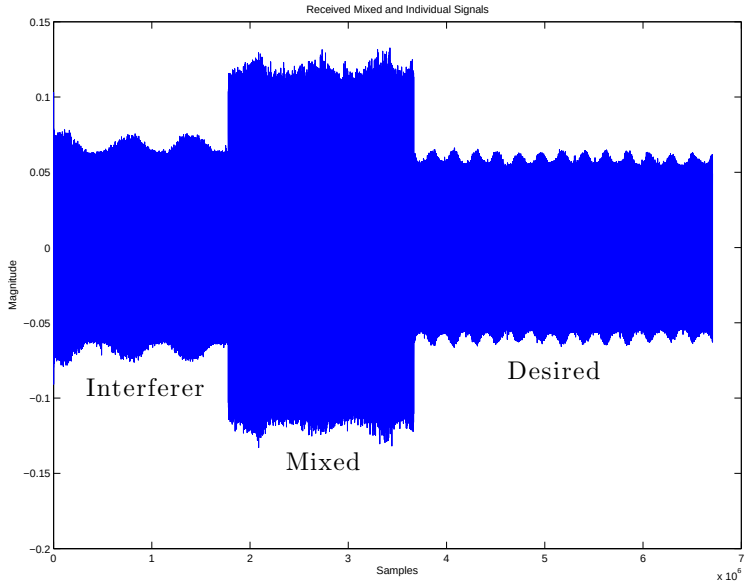


Figure 3.16: Sample periods of time when the interferer is only transmitting, both interferer and desired signal is transmitting, and the desired signal alone. Energy detection can easily determine periods of signal mixing and non-mixing.

Now that the triggering mechanism for halting the algorithms has been determined, further detail must be given to the front or subtraction receiver part of this block. To help replicate the effects of the channel on the subtraction estimate, first a channel estimation

is done using the interferer's actual data symbols. Therefore the received signal is goes through carrier recovery using a signal PLL. Then using the Mueller Muller method, the symbols are timing recovered and demodulated. This data is then fed through a LMS adaptive filter and a channel estimate is produced. Using this channel estimate, a pre-generated known interferer waveform is filtered after passing through an Automatic Gain Control block. This normalizes the previously known signal with respect to the received one. Next the signals are aligned using the correlation method examined previously, and the first found frame in the received signal is copied from that signal. This section is then duplicated to the size of the full received stream at a given time, then finally the two are subtracted from one another. Since the received signal cannot be demodulated or quantized before it reaches the Signal Separation block, all subtractions are done on the baseband received signal. The final results of the Spectral Subtraction block are examined in the next chapter.

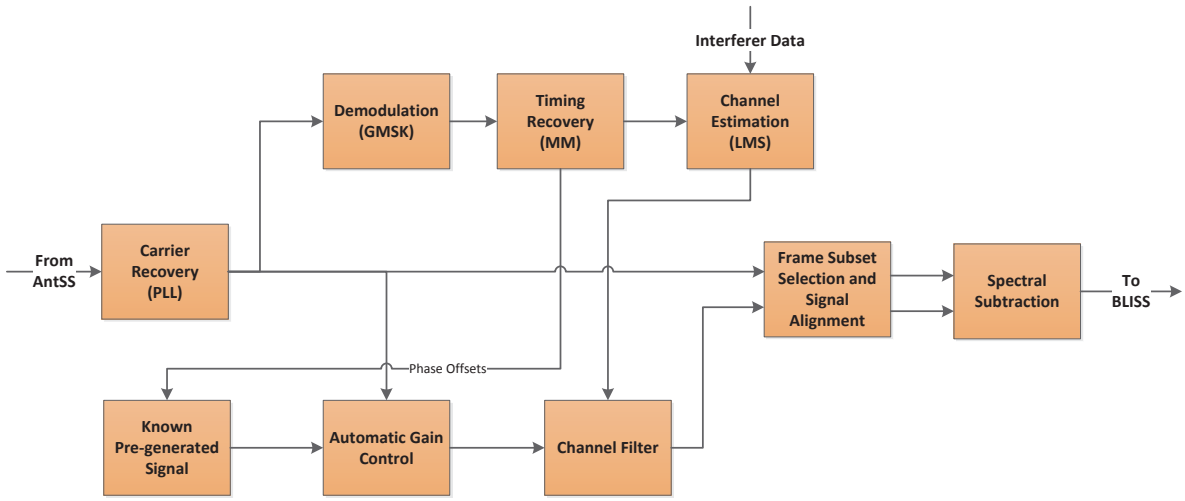


Figure 3.17: Spectral Subtraction block diagram with three datapaths. The signals the AntSS used to estimate the offsets of the interferer and the channel conditions, which are applied to an existing signal in the radio's library. Then the corrected library source signal is correlated with the received signal, to find an appropriate signal section to be used to subtraction from the rest of the signal. Finally the selection signal subtraction is replicated and subtracted from the entire received waveform.

3.5 Signal Separation

The next block to discuss is the signal separation block. The development of this block is very staggered and due to time requirements shortcuts needed to be made. The original desired result was to use a blind source separation technique outlined here [51], which is able to separate multiple signals from one another under specific constraints.

Instead of utilizing a AMUSE, it was decided a new avenue was to be taken, with a more theoretical channel estimation problem in mind. The primary investigator here was Dr. Srikanth Pagadarai. He focused on deriving a MIMO co-channel correlation model, by utilizing Superimposed equalization to acquire channel information. His goal was to provide a method of channel estimation for a MIMO channel mixing problem by utilizing shared channel information. Primarily focusing on training symbols themselves. It it important that this research be discussed here, since it received the most attention of all three of the AS^6 blocks.

The first objective for channel estimation is to examine the channel mixing model which assumes a single-input multiple-antenna broadcast channel. A J-channel FIR system excited by K transmit antennas is considered. A quasi time-invariant multi-path channel is assumed which remains constant during the transmission of a set of consecutive symbols, which are called slots. These slots are assumed independent from one another. The channel estimation is performed over each slot. Each symbol inside a slot is assumed to be the result of a known redundant precoder acting on an input transmit symbol vector drawn from an M-PSK constellation. Therefore the receiver receives signals not only from the intended transmitter but also from Q other interferers. The interferers are assumed to employ the same redundant precoder as the desired signal [67].

With this model, reference [49] shows that no assumptions are necessary regarding the number of the transmit antennas of each of the interferers and the channel orders as long as they are smaller than the block size. The block size in this case is equal to the combination of the individual channel lengths and the number of transmit antenna used by the desired

transmitter. But this evaluation rely on three assumptions:

1. The data sequence x_d is an i.i.d. sequence such that $x_d \sim \mathcal{CN}(0, \sigma_d^2)$.

2. The distribution over the MIMO channel vector is $p(y;\theta) \sim \mathcal{CN}(\mu_y, R_w)$, and the interference vector is distributed normal with covariance R_w.

3. The transmitted symbols, the channel vector, and the interference vector are jointly independent.

With these assumptions the mixing process can be undone but the channel estimation needs to be calculated first. Since this is a MIMO channel, frequency selective fading will need to be captured to providing appropriate channel knowledge. To accomplish this, the original research decided to utilize superimposed equalization, whose operation was heavily discussed in the background section of this thesis. In summary, [3] uses a superimposed symbol transmission scheme to estimate frequency-selective channels. Several points of the DFT of the data are set to known values. This operation can be easily implemented in the time domain when these DFT points are equi-spaced. The channel is estimated using the DFT of the received signal at these selected DFT points. The detection itself is done using an iterative method across these points. Unlike traditional equalizers, the proposed method does not require bandwidth for training. It instead trades spectral power for those symbols themselves, spreading its energy over the entire bandwidth capturing the entire spectrum space. Reference [3] also proves that by placing the training symbols in quasi-periodic position they will not interferer with the data itself. It is important to note that this research discusses no synchronization mechanism, and now with the training data spread throughout the signal is very complex. This addition complexity was factored into the design decisions of the implementation, which is discussed later in this section.

With our channel estimation method chosen, a simulation was created to prove the effectiveness of such a scheme. The result of this implementation were directly compared with the results of the paper to prove correctness. The results of this simulation are seen in Figure 3.18. It examines a random frequency selective channel, with very high suppression

with a across a number of SNR values. As expected there is a linear relationship between SNR and MSE.

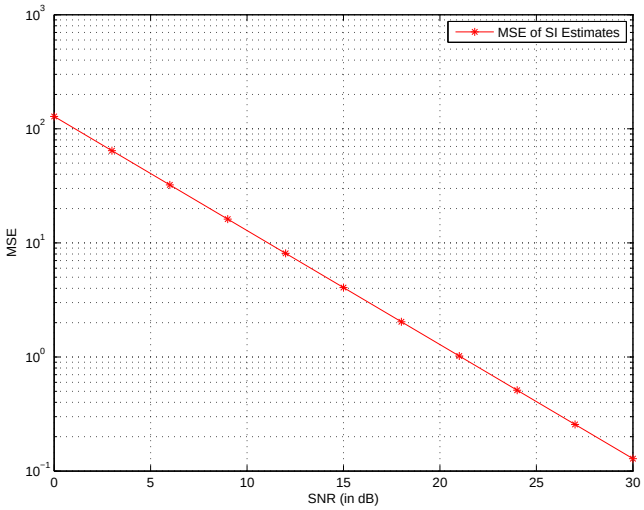


Figure 3.18: Superimposed Equalizer MSE across several SNR values. The more pronounce (higher SNR) the signal the better our estimation becomes. This result was identical to reference [3], for the SI case.

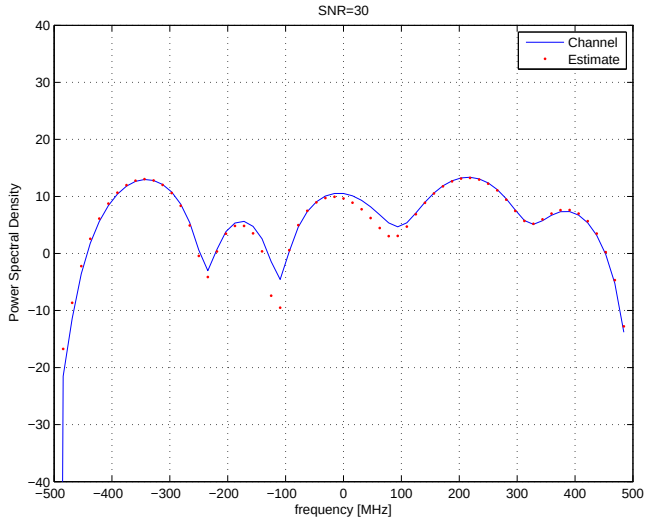


Figure 3.19: Superimposed Equalizer channel estimate in red, superimposed on actual the channel in blue. With SNR ¿ 25, estimates are considered accurate.

From Figure 3.19 you can see in a frequency selective fading channel is very well estimated by the superimposed equalizer. This was exactly the result that was desired, but symbol recovery after equalization still needed to be examined. This unfortunately is the drawback of this formulation. The first is the condition if the mixing matrix H is rank deficient, the channel become inestimable, since there become an infinite amount of solutions to the actual unmixing process. The second is the equalizer, which is numerically intensive, and the alternative has limiting conditions [3]. This conclusion, which is not discussed in [3], is when the equalized signal is very suppressed due to the incomplete channel estimate. The quantization feedback method overcome the previously received frame and dominated all future frames, essentially neglecting any information they have. Again, this only in very suppressive channels. With these results the superimposed equalizer started to become less desirable due to its complexity, but evaluations continued due to the amount of time invested into the topic already.

These simulations provide the necessary foundation to push towards the final implementation of the signal separation block. However, due to time constraints certain decisions needed to be made about this block and the feasibility of its operation. With that in mind the primary goal for the signal separation block was to provide signal separation or maximization from several received signals. Since the previous research provided substantial implementation considerations and issue, a new direction needed to be considered. Therefore instead of the proposed AMUSE [51] technique, another MIMO cross-channel technique called Maximal Ratio Combining (MRC) was chosen as an alternative.

Maximal Ratio Combining is a method of diversity combining in which the signals first weighted, based on their SNR value, and then added together. These weights or gains are made proportional to the RMS signal level and are inversely proportional to the mean square noise level in that channel, and the same proportionality constant is used for all channels [68]. Therefore the channel with the best SNR provides the greatest impact on the resulting sequence. Figure 3.20 outlines MRC with N input signals. First the symbols are weighted then simply summed based on this weighting.

Assuming the received signal is an array of samples received the individual antennas $\boldsymbol{x}(t) = \boldsymbol{h}(t)u(t) + \boldsymbol{n}(t)$ and the individual channels $\boldsymbol{h} = [h_0, h_1, ..., h_{N-1}]^T$, and the additive noise $\boldsymbol{n} = [n_0, n_1, ..., n_{N-1}^T$. The equalized symbol $\hat{x} = x + (h^H n)/(h^H h)$ [69].

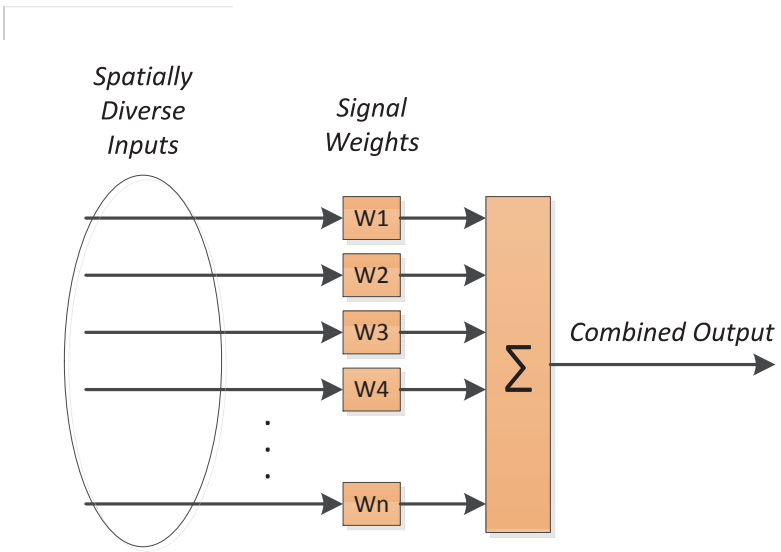


Figure 3.20: Block diagram of Maximal Ratio Combining, which combines spatially separated signals weighted by according to their SNR values.

A simple evaluation of MRC was done to prove its capabilities, which is based on the simulations here [70].

As you can see as you increase the number of antennas in a frequency selective channel the better the result. Therefore MRC was introduce into the framework of the signal separation block, and the new model for this block can be seen in Figure 3.22. This block first utilizes adaptive equalizers on the channels individually then uses MRC to combine there results maximizing the SNR of the desired signal. MRC was combined with the channel estimate approach using adaptive equalizers due to there known feasibility. From the knowledge learned in the previous with the implementation involving GNU Radio, the signal separation block was created entirely in MATLAB. The results of this operation will be discussed in the final chapter of the thesis.

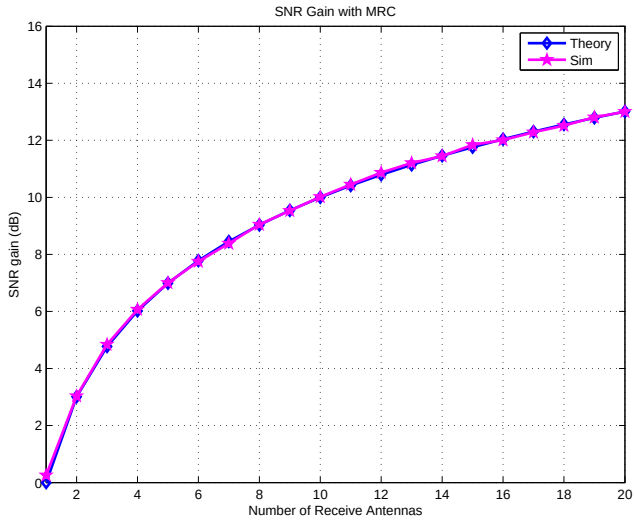


Figure 3.21: Maximal Ratio Combining gain across varying number of antennas. The more antennas provided at the input the more spatial diversity, resulting in a higher combined SNR.

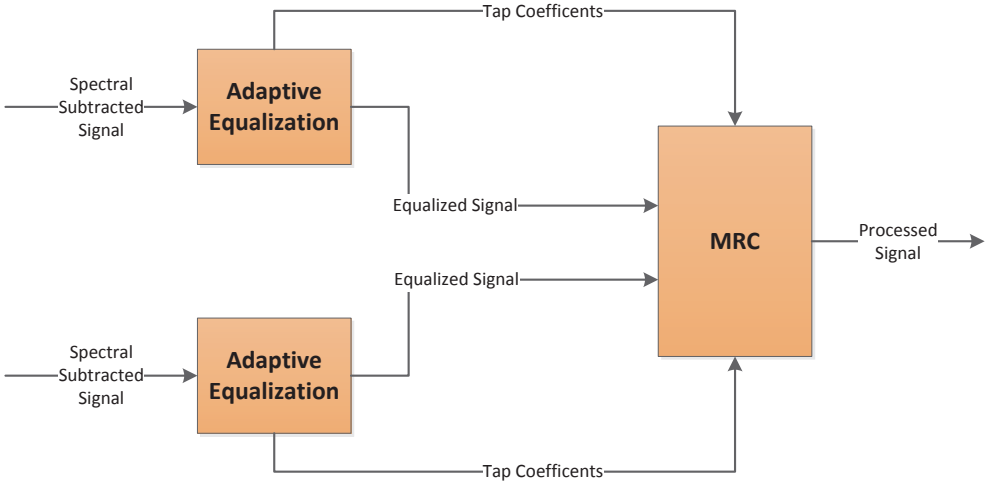


Figure 3.22: Maximal Ratio Combining block design to replace original Super Imposed Equalizer design. AntSS currently is only designed to supply two signal, but could be modified for future implementations.

3.6 Antenna Subset Selection

The Antenna Subset Selection (AntSS) block was partially implemented by a research collaborating team but was never fully completed. It is important to understand the purpose of this block for future research, and how it should interact with the other blocks of the system. The AntSS system itself consists of a series of AntSS boards that provides $2^M - to - 2^N$ down-selection from an array of receive antennas to a set of BLISS receiver inputs. Each individual AntSS board provides 4-to-2 antenna down-selection via a set of RF switches. A basic block diagram of an individual AntSS board is given in Figure 3.23. Each AntSS board has four receive paths. Each path consists of a bandpass filter and low-noise amplifier. The output of each receive path is connected to a switch matrix composed of a series of Single-Pole-Double-Throw (SPDT) RF Switches. The switch matrix is configured so that all possible permutations of the 4-to-2 down-selection are possible. The switch matrix is controlled by software running on a simple PIC processor. The PIC interfaces with the BLISS hardware platform via an RS-232 link that is connected from the BLISS receiver to all AntSS boards.

During operation, training information which was used also by the signal separation block is also used by AntSS. It uses this data to provide SNR values at each Antenna of the desired signal. Once all antennas have been evaluated the system then selects a subset of the best antennas, which is defined by the highest SNR levels. The signals from these antennas are then fed into the remain BLISS blocks. AntSS can have 2^M antennas, but will always deliver two signals. The antennas need to be space at least a single wavelength apart or they will not be considered independent of one another, reducing the effectiveness of AntSS. The physical boards have been built, which can be seen in figure 3.24 and their frequency response on the individual channels can be seen in Appendix A.

Unfortunately none of the control mechanisms in software have been created, therefore AntSS couldn't be fully tested and effectiveness determined. This will be a reasonable avenue for future work.

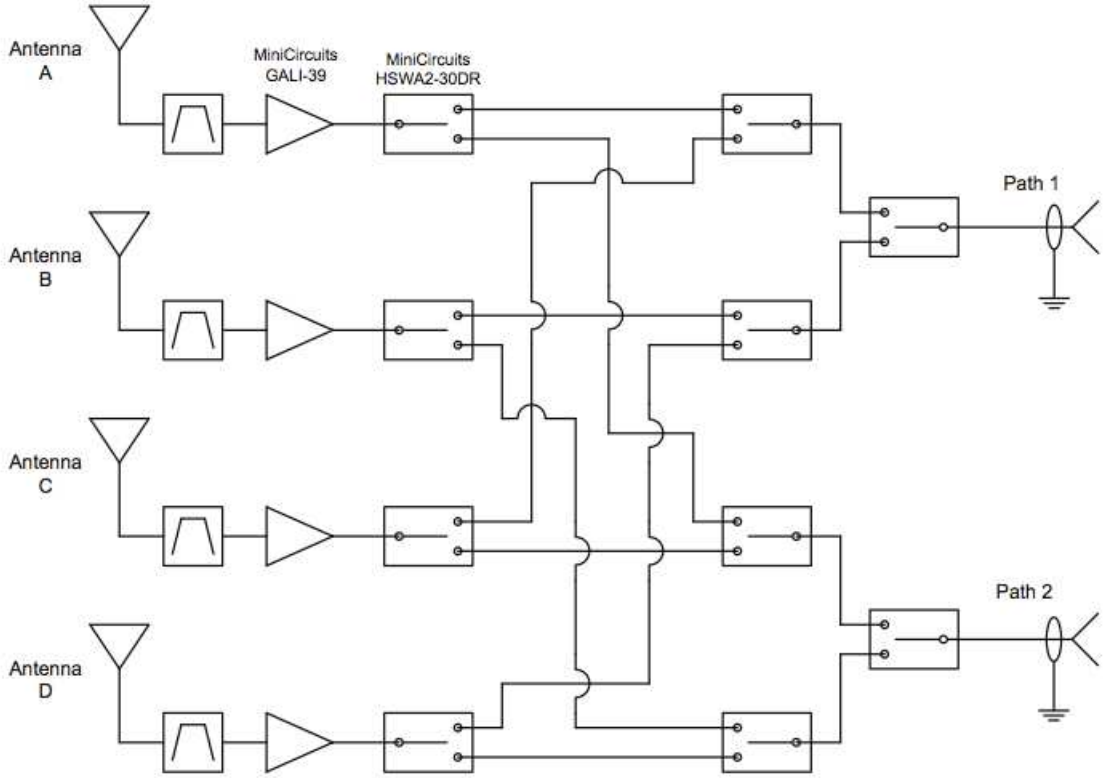


Figure 3.23: Antenna Subset Selection block outline of four receive antennae, of which two are selected through a series of switches.

3.7 Summary

This chapter discussed the implementation fall-backs and successes of the BLISS system. The entire AS^6 system was outlined and scrutinized for feasibility and operational performance. Spectral Subtraction and Signal Separation were transitioned from original research goals to more manageable problems, with simplified solutions. Overall it can be said that optimality and technical complexity were sacrificed in the end for realizability. Many directions needed to be changed, especially in the Signal Separation block, and constraints needed to be tightened on the Spectral Subtraction block. Although AntSS couldn't be realized a solid foundation exists for future work.

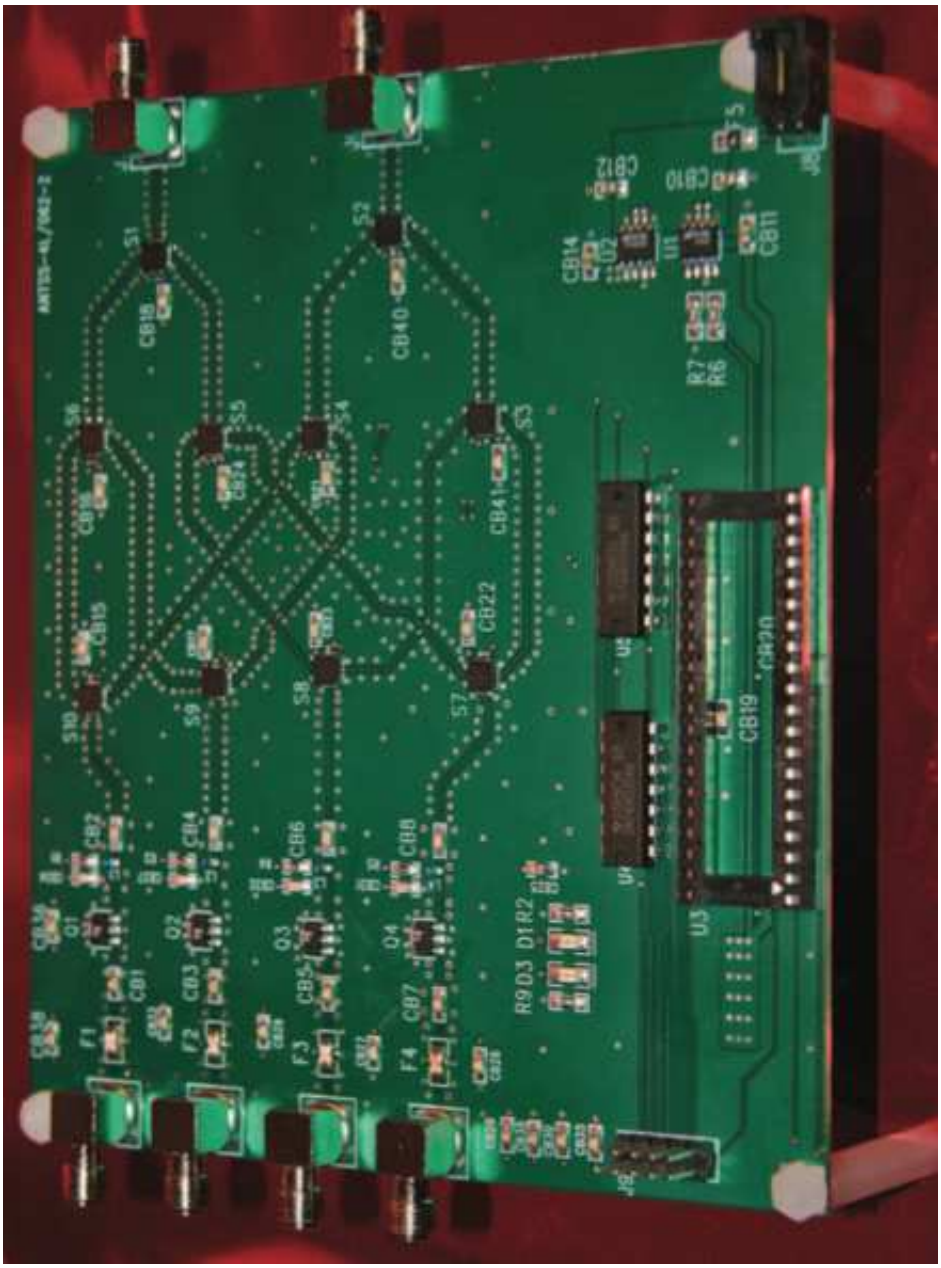


Figure 3.24: Single Antenna Subset Selection physical board, able to select two among its four receive antennas. This selection logic is controlled through a programmable integrated circuit on the board, connecting to a master controller. The master board itself passes messages through a RS232 connection to the PC.

Chapter 4

Experimental Results

4.1 Overview

This chapter outlines the outcomes of the individual AS^6 system blocks and evaluates their performance and realizability. Since a large transition was made from original research and goals, many limitations were realized in the final prototype design presented in this thesis. Feasibility was desired during the implementation stage of this work, causing this transition, but physical implementations were produced and analyzed. Since certain channel conditions couldn't be replicated in reality baseline evaluations were produced for proof of concept, and rigorous hardening is a viable path for future work.

4.2 Spectral Subtraction

The goal of this section is to present the results of the Spectral Subtraction signal processing block. This block, as mentioned in previous sections, tries to remove non-orthogonal known signals from the spectrum that are located on the same frequency of the desired signal. Since the receiver is not allowed to move to other bands, this mimics the effects of a wide-band jamming system, causing all effective bandwidth or throughput to diminish. From the theoretical simulations it is possible to remove the interfering signal but only under severe limitations or constraints. The primary issue is that the signal is extremely time

varying and alignment is quite difficult. To be precise, this block tries to align a source signal with the received signal, and then subtract that signal from all future received segments. This alignment is difficult because the received signal is corrupted by noise and the channel which it passes through. That corruption is time varying as well, and difficult to model over many frames to provided sufficient signal removal. In order to operate, this system assumes the interferer's system is completely known, including the data that is being modulated into the spectrum. Therefore it is the goal of the Spectral Subtraction block is to regenerate the corruption caused by the channel, apply that to the source signal, and then subtract. The experimental evaluation is outlined next and the analysis of its performance.

4.2.1 Experiment

This experiment consists of four USRP2 radio transceivers. One radio acting as a transmitter, one the interferer, and two receive radio connected through a MIMO-USRP cable. The cable causes direct synchronization between the receive radios. This is required to perform maximal ratio combining to correctly align signals constructively. During testing, the interferer first begins transmitting data, then the desired transmitter begins to transmit. At all times the receiver is actively receiving all signals in the spectrum. A sample of the combined received signals can be seen in Figure 3.16. The lower energy sections represent the individual signals and the high energy level sections show the mixed signals.

All transmissions utilize GMSK, due to its resilience to changes in the radio's power amplifier. All radios operate at 100kbits/sec, well under the maximum rate of the radios, minimizing the load on the machines themselves and the amount of data generated. The machines connected to the radios are all have Core Series Intel processors, installed with Ubuntu Linux 10.10 and 12.04. All machines were also running MATLAB 2011B, and GNU-Radio 3.6.2git-145-g7c8347ca built from the git repository. All signal recordings/reception was done in GNU-Radio and all signal processing at baseband was done in MATLAB. Figure 4.1 is a picture of the experimental setup with all four radios

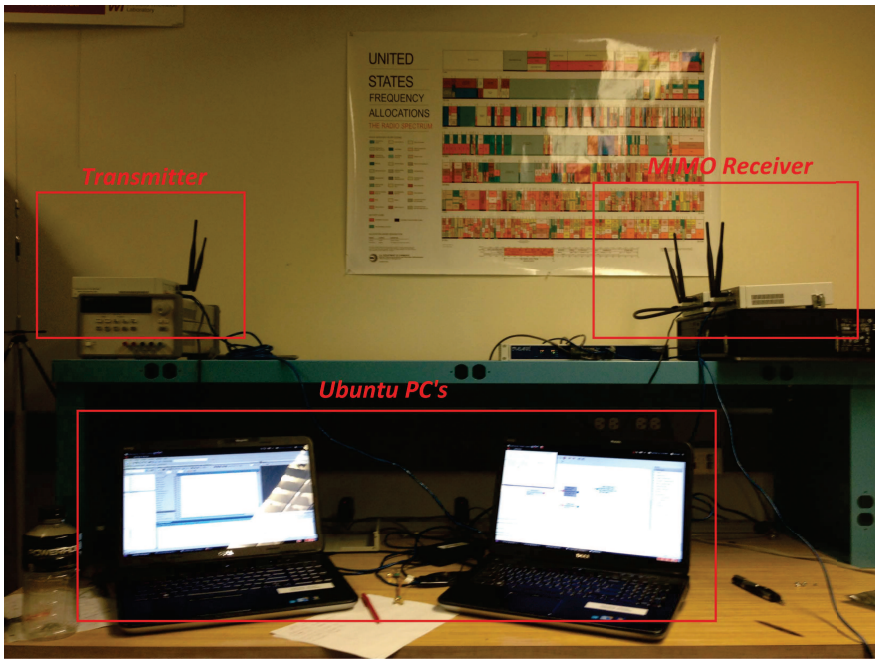


Figure 4.1: Spectral Subtraction Hardware Setup

4.2.2 Analysis

The system was able to correctly identify power changes in the signal and provide the most accurate estimates before the signals begin to mix. This mixing of signals cannot last for long periods of time because the non-idealities of the hardware and environment begin to corrupt the estimate of the interfering signal. This corruption removed any hope of desired signal recovery. The corruption can be seen clearly in Figure 4.2, as the frames further from the estimate become more and more corrupted. This period of time for which the estimate provides enough accuracy for spectral removal depends on the changes in phase frequency and channel effects.

Many of the non-idealities associated with Spectral Subtraction were discussed in the Implementation chapter. Though most of effort was put towards compensating for these effects, they still produced a large amount of error in the final design. These sources of corruption manifested as several different errors in the output of the block. The first was timing offsets. Since the interferer is extremely time-varying, if a single bit is missed, the en-

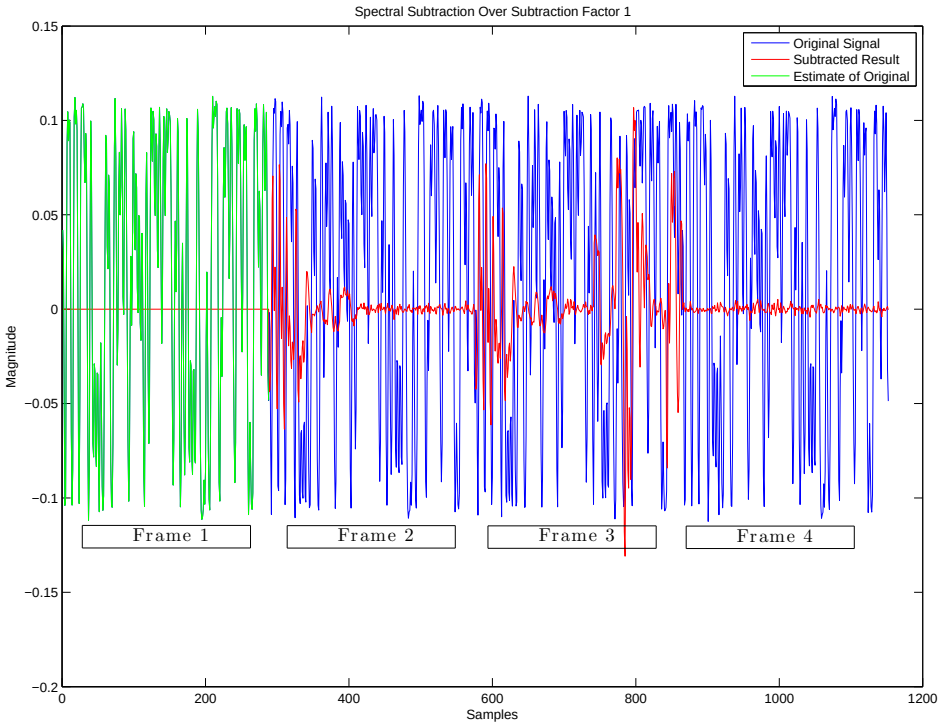


Figure 4.2: After Spectral Subtraction the error, or the remaining not removed interferer, appears as an oscillation at the remaining frame positions. Green represents the subtraction frame estimate, blue the originally received frames, and red the frames after subtraction.

tire signal downstream becomes corrupted. If this timing is out of phase π radians, then an identical corruption occurs. As the phase offset ϕ is removed, this exponentially decreases the error. As for carrier frequency and phase offsets, this error manifests as an oscillating error in the output signal. Figure 4.2 shows severals frames and the resulting energy left behind after subtraction. This plot was generate by a simplified test, just the interferer transmitted data and attempts were made to nullify these transmissions. These oscillations point towards problems in phase or frequency estimations.

These non-ideality seemed to be the most difficult problem to compensated for, and become worse when the signals are mixed. This is because they cannot be directly measured. A oscillating error in the spectrum, shown in Figure 4.2, mirror the effects shown in the

simpler nullifying case. As express previously, Spectral Subtraction is very susceptible to data corruption. It is analogous to hitting a moving target, since the operation is extremely time varying and time dependent.

To help compensate for these non-idealities phase synchronization and carrier synchronization was attempted with the received signal. As outlined in the Implementation chapter, Spectral Subtraction is designed as a receiver in a receiver system. The front received locks onto the interferer, applies necessary phase shifts, and subtracts a synthetically generated signal from the previously known data which has also been modulated and pulse-shaped. Unfortunately this the received data also contains many other effects cause by the channel, including multi-path and large amounts of noise. To resolve this the front receiver must estimate the channel and apply the same amount of noise to the signal to remove it sufficiently. With H_I and H_D representing the interferer's channel and desired signal's channel respectively. The transmitted symbols of the desire signal $\mathbf{x_D}$ and the interferer $\mathbf{x_I}$. The received signal can be written as: $\mathbf{y} = \mathbf{H_I^H x_I} + \mathbf{H_D^H x_D} + \mathbf{w}$. Only $\mathbf{x_I}$ is known in this realization, and both $\mathbf{H_I}$ and $\mathbf{w}$ are time dependent. As mentioned previously SS is very susceptible to any carrier, power, and timing effects.

It is important to show Spectral Subtraction operating correctly and when errors occur in the estimation. Figure 4.3 show a desired result from Spectral subtraction when all timing is aligned, while Figure 4.4 shows the error propagation through the frames. Both these results looked 20 frames ahead of the section used for subtraction. Finally Figure 4.5, shows the error associated with increasing the frames looking ahead. As expected, the further you move away from your original subtraction frame, the worse your results become.

4.2.3 Summary

Spectral Subtraction in a extremely well studied area in signal processing, but no existing literature exists for its application in a digital communication system. It has been shown here that it can be quite difficult for it to be applied, even under strict constraints.

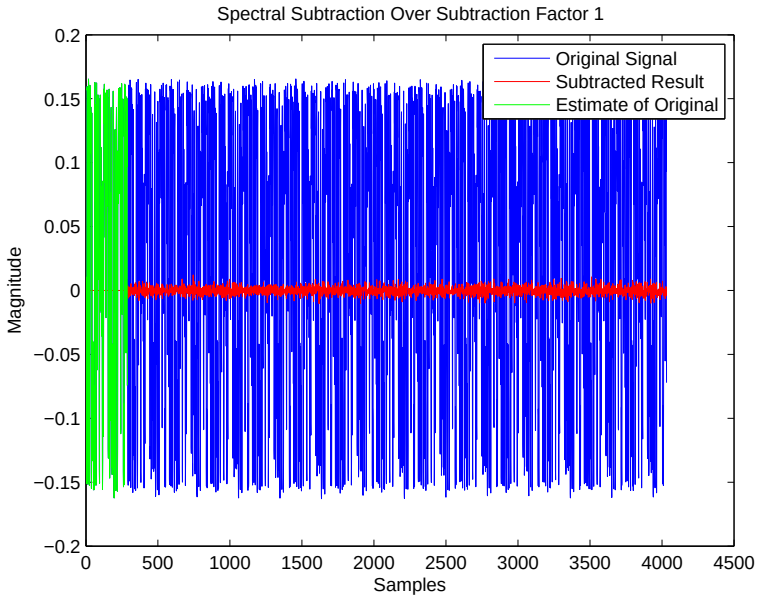


Figure 4.3: Desired result after Spectral Subtraction with 20 forward frames, producing limited residual signal. Phase changes minimally across these frames.

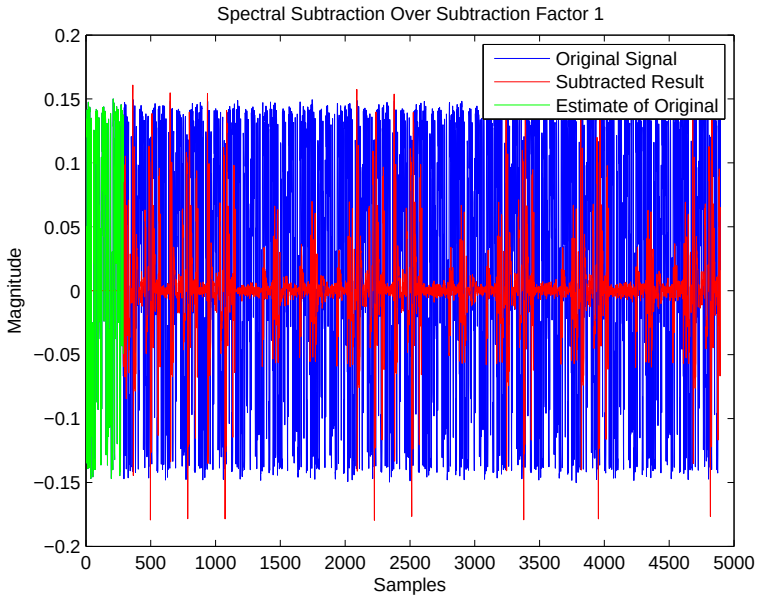


Figure 4.4: Error after Spectral Subtraction with incorrect phase estimates with 20 forward frames. The error oscillates among the remaining frames.

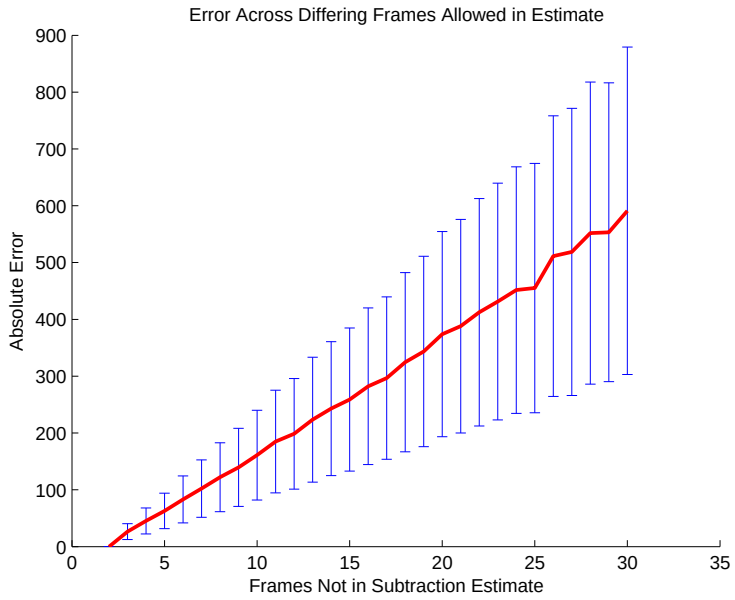


Figure 4.5: Error associated with increasing duration between restimation of subtractions frames. The far the subtraction pushed into future frames, the more inaccurate that estimate becomes. The error was calculated by summing all of the residual noise left after subtraction.

Under the conditions of this thesis, the assumptions are quite reasonable, but due to the large amount of error in the results, more may need to be considered. These may include accuracy requirements for physical equipment, primarily to reduce carrier frequency drift. Burst scenarios may also be considered to reduce bit error rate. Overall, for a completely non-existent field of study, these results point the possibility of operational success. Future work will we required, especially during the implementation phase of designs.

4.3 Signal Separation

The Signal Separation block changed the most from the original research design. Much of the design needed to be reconsidered because of time constraints and lack of robust research conclusions. The design distilled down to a combination of Maximal Ratio Combining and adaptive equalization. Two well know concepts in communication system design, and rather straight forward to implement. Maximal Ratio Combining provides the benefits of maximizing the spectrum spatially, while also adaptively equalizing these separated data streams to help remove corruption left over by spectral subtraction of the channel itself. The more dimensionality that can be exploited the more performance can be extracted.

4.3.1 Experiment

Signal Separation utilized an identical setup as the Spectral Subtraction testing, which is quite obvious since Signal Separation is downstream from Spectral subtraction in the data path. The interferer was also removed from this testing, and the noise that pre-existed in the spectrum used to reflect the noise remaining from Spectral Subtraction. This decision was made to separate problems or issues with the performance of the Spectral Subtraction block, providing direct analysis and evaluation of the Signal Separation block. It takes in two data stream, which have been timing synchronize through the use of the MIMO USRP cable. These streams are equalized by an LMS Adaptive equalizer of length 14. These taps provided the Maximal Ratio Combining weight analysis, providing information on which of

the channel was less corrupted. This is done by combining the filter tap values and taking the ratio of the two equalizers, and these weights determine how much of each signal is added to the final output signal.

Again all transmissions utilize GMSK, due to its resilience to changes in the radio's power amplifier. All radios operate at 100kbits/sec, well under the maximum rate of the radios, minimizing the load on the machines themselves and the amount of data generated. The machines connected to the radios are all Core Series Intel processing, installed with Ubuntu Linux 10.10 and 12.04. All machines were also running MATLAB 2011B, and GNU-Radio 3.6.2git-145-g7c8347ca built from the git repository. All signal recordings/reception was done in GNU-Radio and all signal processing at baseband was done in MATLAB. Therefore the entire Signal Separation block resides in MATLAB.

Several separate transmissions were made and baseline bit error rates were calculated. This was done by quantizing the final output and comparing the results. The Mueller Muller method again was used for timing recovery due to its robustness. It doesn't account for bit slips, but due to the rather short periods of transmission these can be ignored. Since this block is designed to help in rather noisy and localized corruption situations, it can be very hard to test because there is no control over the spectral environment to set such conditions. Rather this design shows a proof of concept with SDR technology. The results for the baseband tests are shown in Figure 4.6.

4.3.2 Analysis

From the results seen from the baseline testing, reception seems to be quite reasonable given the inferior hardware. To provide a comparison to the simulated results, the transmitter power was lowered to synthetically reduce the SNR of the signal. The results of these tests can be seen in Figure 4.6.

Since only antennas are provided by the AntSS block, MRC will suffer. Higher performance will be provided by more input signals, but that would increase the complexity of

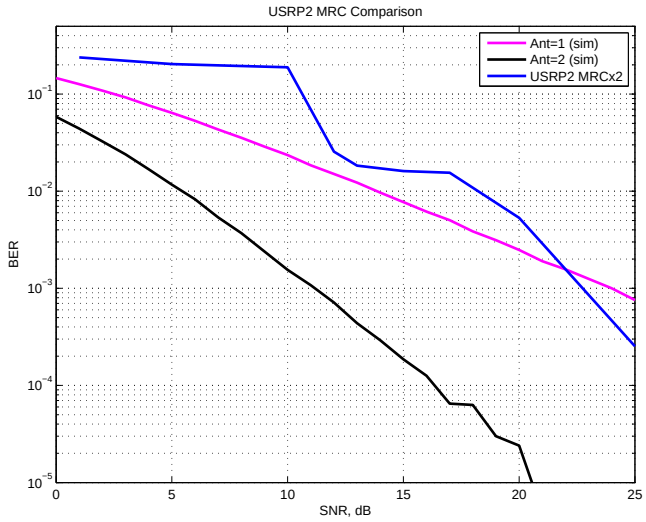


Figure 4.6: Signal Separation MRC performance comparison against simulations. Each data point represents the average BER of 67,000 frames received at varying SNR levels.

the system significantly. More input signals could be a future alley for BLISS system for future research. For the current system architecture the results are acceptable given the limitations and variability in the hardware itself.

4.4 Summary

These results show that it is extremely difficult to avoid a wide-band jammer even when fine details are known about the interferer itself. Spectral Subtraction has large downfalls in its implemented realization due to its fragile nature. Small non-idealities can cause large errors, especially in timing, causing a large misalignment in the subtracted signal. Equipment with higher tolerances would be a way of minimizing these effects. Signal Separation is focused in solid theoretical foundation and provides additional SNR above single antenna reception, allowing an extrapolation in the dimensionality of the environment and the signal itself. It was fully implemented with reasonable BER given the non-ideal hardware. Additional antennas should be considered for further gains from Maximal Ratio Combining.

Chapter 5

Conclusions

This chapter summarizes the work performed as part of this project and then suggests related research that can be performed in the future. The research achievements includes a description of the system developed, the capabilities of the system and the results it produced. The future work section includes a list of improvements that can be made to the system and some ideas that can start future research projects.

5.1 Research Outcomes

As part of this research effect, the following was achieved:

- The tasked original proposed research was examined and evaluated. Synthesizing several years of development and work from many individuals.
- A viable solution was theoretically developed for the removal of non-orthogonal wide-band jamming sources. Conservative constraints were applied to the construction, providing enough flexibility for hardware implementations. The constraints only limited the interfering signal, determining that it must repeat in a relatively short period. This made it more easily implementable and simplify the receiver design. This constraint can be removed, but more resources will be required during subtraction, especially in memory resources.

- A hardware implementation was produced for Spectral Subtraction utilizing both GNU Radio and MATLAB.
- The proposed Spectral Subtraction block provided significant signal removal of actual over the air signals, but timing issues still remain.
- In-depth analysis was provided into the sources of error with the Spectral Subtraction block.
- An alternative solution was provided for the Signal Separation block originally presented. This solution was based on a well know method, Maximal Ratio Combining, for combining signals in a constructive way by utilizing their dimensionality. This decision was made in combination with the project advisors.
- Theoretical simulations were generated in MATLAB providing a basis for performance for Maximal Ratio Combining.
- A hardware implementation was produced for the Signal Separation block utilizing both GNU Radio and MATLAB.
- A comparison was provided between the theoretical simulations and the hardware implementation. It is unfair to directly compare the implementation and theoretical results, since the theoretical results don't account for the non-idealities associated with over the air transmissions but the comparison was provided.

5.2 Future Work

Future research activities that are related to this work are discussed in this section.

- Removal of the short repeated signal constraint must be explored to not hinder the effects or purpose of the jammer itself. Long sequences or a method for sequence generation such as LFSR (Linear Feedback Shift Registers) could be implemented in such a way. The receiver must be able to very accurately predict what sample will come next in the stream itself in-order to provide accurate signal removal.

- Original theoretical derivation in [67] for Signal Separation must be readdressed and superimposed equalizer design explored. The end resulting work focused on the effectiveness of specifically designed training data using a affine precoder. A complex timing recover will need to be constructed to utilize such a system due to the scattered nature of the training symbols superimposed on the data itself.

- A larger number of signals feed into the Spectral Subtraction and Signal Separation blocks should be explored to improve the functionality of Maximal Ratio Combining.

- GNU Radio controlling blocks for Antenna Subset Selection block should be implemented to provide the necessary dimensionally separated signals into the downstream blocks.

- System integration needs to be done between all three blocks to provide a complete system. Performance metrics should be evaluated on this system once constructed to determine its effectiveness under wide-band jamming conditions.

- Current implementations need to be optimized and code re-factored allowing for better code portability and performance gains.

- Hardware considerations need to be addressed, determining requirements for actual deployment of such a system in the field. Hardware constraints will be the largest limiting factor, especially on the RF front end of the design.

Bibliography

[1] D. A. M. Wyglinski and D. C. R. Anderson, "BLISS: A Blind Spectrum Separation Approach for Jamming-Resistant Communications," Office of Naval Research, Technical Proposal ONR 09-016, 2009.

[2] "Resource Allocation Studies Using Software Defined Radio (SDR)," March 2013. [Online]. Available: http://confluence.qu.edu.qa/display/NPRPRESEARCH/USRP2+Testbed

[3] M. Ghogho, D. McLernon, E. Alameda-Hernandez, and A. Swami, "Channel Estimation and Symbol Detection for Block Transmission Using Data-Dependent Superimposed Training," *Signal Processing Letters, IEEE*, vol. 12, no. 3, pp. 226 – 229, march 2005.

[4] K. Pretz, "Overcoming Spectrum Scarcity," August 2012. [Online]. Available: http://theinstitute.ieee.org/technology-focus/technology-topic/overcoming-spectrum-scarcity

[5] M. R. Frater and M. Ryan, *Electronic Warfare for the Digitized Battlefield*. Artech Print on Demand, 2001.

[6] H. Zimmermann, "OSI Reference Model–The ISO Model of Architecture for Open Systems Interconnection," *Communications, IEEE Transactions on*, vol. 28, no. 4, pp. 425 – 432, apr 1980.

[7] A. Mpitziopoulos, D. Gavalas, C. Konstantopoulos, and G. Pantziou, "A Survey on

Jamming Attacks and Countermeasures in WSNs," *IEEE Communications Surveys and Tutorials*, vol. 11, no. 4, pp. 42–56, 2009.

[8] W. Xu, W. Trappe, Y. Zhang, and T. Wood, "The feasibility of launching and detecting jamming attacks in wireless networks," in *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, ser. MobiHoc '05. New York, NY, USA: ACM, 2005, pp. 46–57. [Online]. Available: http://doi.acm.org/10.1145/1062689.1062697

[9] K.-C. Chen and R. Prasad, *Cognitive Radio Networks*. John Wiley and Sons, 2009.

[10] F. Ahsan, A. Zahir, S. Mohsin, and K. Hussain, "Survey on Survival Approaches in Wireless Network Against Jamming Attack," *Journal of Theoretical and Applied Information Technology*, vol. 30, no. 1, pp. 55–67, August 2011.

[11] M. Acharya and D. Thuente, "Intelligent Jamming Attacks, Counterattacks and (Counter)2 Attacks in 802.11b Wireless Networks," in *Proceedings of the OPNETWORK Conference*, 2005.

[12] J. Shi, T. Salonidis, and E. W. Knightly, "Starvation mitigation through multi-channel coordination in csma multi-hop wireless networks," in *in CSMA Multi-hop Wireless Networks," in Proceedings of the International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2006, pp. 214–225.

[13] K. W. Reese and A. Salem, "A survey on jamming avoidance in ad-hoc sensory networks," *J. Comput. Sci. Coll.*, vol. 24, no. 3, pp. 93–98, Jan. 2009. [Online]. Available: http://dl.acm.org/citation.cfm?id=1409873.1409891

[14] C. H. Sterling, *Military Communications: From Ancient Times to the 21st Century*. ABC-CLIO, 2007.

[15] K. Mahadevan, S. Hong, and J. Dullum, "Anti-Jamming: A Study," December 2005.

[16] H. Arslan, *Cognitive Radio, Software Defined Radio, and Adaptive Wireless Systems*, ser. Signals and Communication Technology. Springer London, Limited, 2007. [Online]. Available: http://books.google.com/books?id=yMgGGe-z7mYC

[17] J. Mitola, "The Software Radio Architecture," *IEEE Communications Magazine*, pp. 26–38, 1995.

[18] F. Ellersick and D. Schilling, *Special Issue on Progress in Military Communications*, ser. IEEE journal on selected areas in communications : a publication of the IEEE Communications Society. IEEE, 1985. [Online]. Available: http://books.google.com/books?id=NDq8HAAACAAJ

[19] I. C. Society, "Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," IEEE, Tech. Rep., 2012.

[20] A. Haimovich and A. Vadhri, "Rejection of Narrow-band Interferences in PN Spread Spectrum Systems Using an Eigenanalysis Approach," in *Military Communications Conference, 1994. MILCOM '94. Conference Record, 1994 IEEE*, oct 1994, pp. 1002 –1006 vol.3.

[21] R. DiPietro, "An FFT Based Technique for Suppressing Narrow-band Interference in PN Spread Spectrum Communications Systems," in *Acoustics, Speech, and Signal Processing, 1989. ICASSP-89., 1989 International Conference on*, may 1989, pp. 1360 –1363 vol.2.

[22] W. Jones and K. Jones, "Narrowband Interference Suppression using Filter-Bank Analysis/Synthesis Techniques," in *Military Communications Conference, 1992. MILCOM '92, Conference Record. Communications - Fusing Command, Control and Intelligence., IEEE*, oct 1992, pp. 898 –902 vol.3.

[23] Y. Xu, J. Weaver, J. Healy, D.M., and J. Lu, "Wavelet transform domain filters: a spatially selective noise filtration technique," *Image Processing, IEEE Transactions on*, vol. 3, no. 6, pp. 747–758, 1994.

[24] L. Zhao, M. Amin, and A. Lindsey, "Subspace Projection Techniques for Anti-FM Jamming GPS Receivers," in *Statistical Signal and Array Processing, 2000. Proceedings of the Tenth IEEE Workshop on*, 2000, pp. 529 –533.

[25] A. Kandangath, "Jamming mitigation techniques for spread spectrum communication systems," University of Arizona, Tech. Rep., 2003.

[26] J. C. Richard Johnson, W. A. Sethares, and A. G. Klein, *Software Receiver Design: Build Your Own Digital Communications System in Five Easy Steps.* Cambridge University Press, 2011.

[27] M. Lagunas, A. Perez-Neia, and J. Vidal, "Joint beamforming and viterbi equalizer in wireless communications," in *Signals, Systems amp; Computers, 1997. Conference Record of the Thirty-First Asilomar Conference on*, vol. 1, 1997, pp. 915–919 vol.1.

[28] R. Koetter, A. Singer, and M. Tüchler, "Turbo equalization," *Signal Processing Magazine, IEEE*, vol. 21, no. 1, pp. 67–80, 2004.

[29] N. Morinaga, R. Kohno, and S. Sampei, *Wireless Communication Technologies: New Multimedia Systems*, ser. The Springer International Series in Engineering and Computer Science. Springer, 2000. [Online]. Available: http://books.google.com/books?id=rXV7jsjw8C8C

[30] S. Adireddy, L. Tong, and H. Viswanathan, "Optimal Placement of Training for Frequency-Selective Block-Fading Channels," *Information Theory, IEEE Transactions on*, vol. 48, no. 8, pp. 2338 – 2353, aug 2002.

[31] B. Farhang-Boroujeny, "Experimental study of semi-blind channel identification/equalization through pilot signals," in *Signal Processing, 1996., 3rd International Conference on*, vol. 1, oct 1996, pp. 618 –621 vol.1.

[32] G. Zhou, M. Viberg, and T. McKelvey, "A First-Order Statistical Method for Channel Estimation," *Signal Processing Letters, IEEE*, vol. 10, no. 3, pp. 57 –60, march 2003.

[33] A. Orozco-Lugo, M. Lara, and D. McLernon, "Channel Estimation Using Implicit Training," *Signal Processing, IEEE Transactions on*, vol. 52, no. 1, pp. 240 – 254, jan. 2004.

[34] "SOLVING TV RECEPTION PROBLEMS," 2002. [Online]. Available: http://www1.electusdistribution.com.au/imagesuploaded/tvrecepe.pdf

[35] S. Adireddy and L. Tong, "Optimal Placement of Known Symbols for Nonergodic Broadcast Channels," *IEEE Trans. Info. Theory*, vol. 2192, 2002.

[36] S. F. Boll, "A spectral subtraction algorithm for suppression of acoustic noise in speech," *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '79*, vol. 4, pp. 200–203, 1979.

[37] W. Kim, S. Kang, and H. Ko, "Spectral subtraction based on phonetic dependency and masking effects," *Vision, Image and Signal Processing, IEE Proceedings* -, vol. 147, no. 5, pp. 423–427, Oct.

[38] N. Upadhyay and A. Karmakar, "A perceptually motivated multi-band spectral subtraction algorithm for enhancement of degraded speech," in *Computer and Communication Technology (ICCCT), 2012 Third International Conference on*, Nov., pp. 340–345.

[39] W. Guang-Yan, Z. Xiao-Qun, and W. Xia, "Musical noise reduction based on spectral subtraction combined with wiener filtering for speech communication," in *Wireless Mobile and Computing (CCWMC 2009), IET International Communication Conference on*, Dec., pp. 726–729.

[40] T. Inoue, H. Saruwatari, Y. Takahashi, K. Shikano, and K. Kondo, "Theoretical analysis of musical noise in generalized spectral subtraction based on higher order statistics," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 19, no. 6, pp. 1770–1779, 2011.

[41] M. Berouti, R. Schwartz, and J. Makhoul, "Enhancement of speech corrupted by acoustic noise," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '79.*, vol. 4, apr 1979, pp. 208 – 211.

[42] I. Mitola, J., "Software Radios: Survey, Critical Evaluation and Future Directions," *Aerospace and Electronic Systems Magazine, IEEE*, vol. 8, no. 4, pp. 25 –36, april 1993.

[43] R. Staple and K. Werbach, "The End of Spectrum Scarcity," march 2004. [Online]. Available: http://spectrum.ieee.org/telecom/wireless/the-end-of-spectrum-scarcity

[44] R. Lackey and D. Upmal, "Speakeasy: The Military Software Radio," *Communications Magazine, IEEE*, vol. 33, no. 5, pp. 56 –61, may 1995.

[45] E. Koski and C. Linn, "The JTRS program: software-defined radios as a software product line," in *Software Product Line Conference, 2006 10th International*, 0-0 2006, pp. 10 pp. –191.

[46] V. Bose, M. Ismert, M. Welborn, and J. Guttag, "Virtual radios," *Special Issue on Software Radios*, 1999.

[47] M. Inc., 2013. [Online]. Available: http://www.mathworks.com/discovery/sdr/usrp.html

[48] ——, January 2013. [Online]. Available: http://www.mathworks.com/products/embedded-coder/index.html?s_cid=0909_webg_9b_ccslink_trans_268513

[49] A. Wyglinski, C. Anderson, and S. Pagadarai, "Semi-blind estimation of correlated mimo channels using optimal training design in multiuser environments," Office of Naval Research, Tech. Rep., 2011.

[50] E. Telatar, "Capacity of Multi-antenna Gaussian Channels," *European Transactions on Telecommunications*, vol. 10, no. 6, pp. 585–595, 1999. [Online]. Available: http://dx.doi.org/10.1002/ett.4460100604

[51] R. Li and F. Wang, "Adaptive semiblind signal separation approach using temporal structure of sources," in *Anti-counterfeiting, Security, Identification, 2007 IEEE International Workshop on*, April, pp. 315–318.

[52] E. Research, "USRP2 The Next Generation of Software Radio Systems," 1043 North Shoreline Blvd, September 2010. [Online]. Available: http://www.ece.umn.edu/users/ravi0022/class/ee4505/ettus_ds_usrp2_v5.pdf

[53] Q. Norton, "GNU Radio Opens an Unseen World." [Online]. Available: http://www.wired.com/science/discoveries/news/2006/06/70933?currentPage=all

[54] E. Research, "USRP N200/210 Networked Series," 1043 North Shoreline Blvd, September 2012. [Online]. Available: https://www.ettus.com/content/files/07495_Ettus_N200-210_DS_Flyer_HR_1.pdf

[55] *UHD Documentation*, Ettus Research. [Online]. Available: http://files.ettus.com/uhd_docs/doxygen/html/structuhd_1_1rx__metadata__t.html

[56] D. Symeonidis, "The SDR Blog: A blog about Software Defined Radio," March 2013. [Online]. Available: http://sdrblog.wordpress.com/2009/03/16/timestamp-data-from-the-usrp/

[57] "MultiUsrp or Mimo use of the USRP," March 2013. [Online]. Available: http://gnuradio.org/redmine/projects/gnuradio/wiki/MultiUsrp

[58] R. Udrea and S. Ciochina, "Speech enhancement using spectral over-subtraction and residual noise reduction," in *Signals, Circuits and Systems, 2003. SCS 2003. International Symposium on*, vol. 1, 0-0, pp. 165–168 vol.1.

[59] B. BabaAli, H. Sameti, and M. Safayani, "Spectral subtraction in model distance maximizing framework for robust speech recognition," in *Signal Processing, 2008. ICSP 2008. 9th International Conference on*, Oct., pp. 627–630.

[60] J. A. Gubner, *Probability and Random Processes for Electrical Engineers*. Cambridge University Press, 2010.

[61] T. G. R. Project, "GNU Radio 3.6.4 C++ API Documentation." [Online]. Available: http://gnuradio.org/doc/doxygen/index.html

[62] C. Sanderson, "Armadillo: An open source c++ linear algebra library for fast prototyping and computationally intensive experiments," NICTA, Tech. Rep., 2010.

[63] D. Beazley, "SWIG (Simplified Wrapper and Interface Generator)," 1996–. [Online]. Available: http://www.swig.org/

[64] E. Jones, T. Oliphant, P. Peterson *et al.*, "SciPy: Open source scientific tools for Python," 2001–. [Online]. Available: http://www.scipy.org/

[65] E. Blossom, "[Discuss-gnuradio] s/Eric/Tom/g," TheMailArchive, September 2010. [Online]. Available: http://www.mail-archive.com/discuss-gnuradio@gnu.org/msg26468.html

[66] S. Haykin, *Adaptive Filter Theory*, 4th ed. Prentice Hall, 2002.

[67] S. Pagadarai, "Wireless communications and spectrum characterization in impaired channel environments," Ph.D. dissertation, Worcester Polytechnic Institute, 2012.

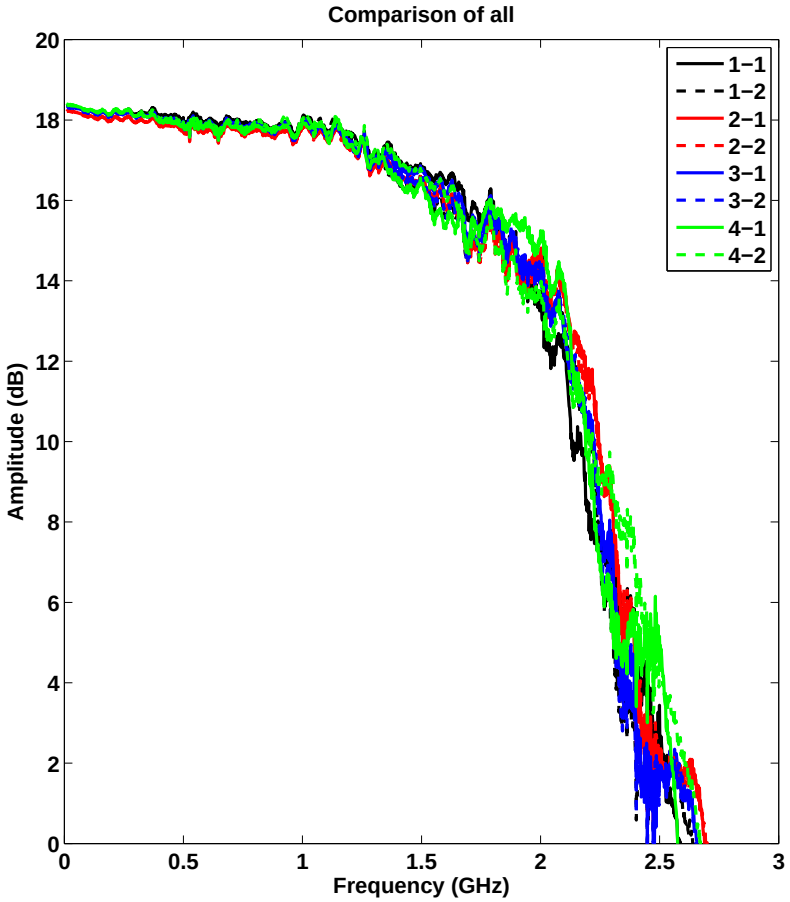
[68] I. for Telecommunication Sciences, "Federal standard 1037c." [Online]. Available: http://www.its.bldrdoc.gov/fs-1037/fs-1037c.htm

[69] R. Adve, "Receive diversity." [Online]. Available: http://www.comm.utoronto.ca/~rsadve/Notes/DiversityReceive.pdf

[70] K. S. M. Pillai, "Maximal Ratio Combining Effective SNR," apr 2009. [Online]. Available: ttp://images.dsplog.com/db-install/wp-content/uploads/2008/09/script_maximal_ratio_combining_effective_snr.m

Appendix A

AntSS Board Frequency Response



Appendix B

SS.m

```
clc;
!sudo sysctl -w net.core.rmem_max=50000000
!sudo sysctl -w net.core.wmem_max=1048576

!LD_LIBRARY_PATH="" && /home/sdruser/COLLINS/SS/
    top_block_SS_R.py

%% Spectral Subtraction
addpath('/home/sdruser/GNURadio/gnuradio/gnuradio-core/src/utils
    ');
received=read_complex_binary('received.txt');
received_GMSK=read_float_binary('received_GMSK.txt');

known=read_complex_binary('known_modulated.txt').*0.1;

%remove startup transient
cut=20001;
received=received(cut:end);
```

```
recSig=received;

%% Timing Recovery
L=2;
g=0.07;
hSync = comm.MuellerMullerTimingSynchronizer('SamplesPerSymbol',
    L, ...
     'ErrorUpdateGain', g);

% Estimate the delay from the received signal
[sig, phase] = step(hSync, recSig);

% apply phase
recSig=recSig.*exp(1i*phase(end)*pi/180);



%% find data
r=double(sign(received_GMSK));



s_saved=[ -1     1     1     1     1    -1     1    -1];
s_saved=[s_saved s_saved s_saved]';
prea=s_saved;
s=s_saved.';

% Find Preambles
indexs=[];
for i=1:length(r)-length(s_saved)-2000
    x=sum(r(i:i+length(s_saved)-1)==s_saved);
```

```
    %if x>10
    %    disp(x);
    %end
    if sum(r(i:i+length(s_saved)-1)==s_saved)==length(s_saved)
        indexs=[indexs i];
    end
end


% Retry is no signal found
if isempty(indexs)
    disp('Looped');
    SS_Final;
    break
end

%% calculated channel coefficients
loc=indexs(1);
w=zeros(10,1);
%Real message
dd=read_char_binary('/home/sdruser/COLLINS/SS/input.txt');
true_message=de2bi(dd',8,'left-msb');
mbits=reshape(true_message',size(true_message,2)*size(
    true_message,1),1);
mu=0.001;
e_s=[];
for i=length(w)+1:length(mbits)-length(w)
    rr=mbits(i:-1:i-length(w)+1);
```

```
    disp([w'*rr received_GMSK(loc+i-1) mbits(i)]);

    e=w'*rr-received_GMSK(loc+i-1);
    e_s=[e_s e];

    w=w-mu*rr*conj(e);

end

%% Section of Signal
large_SR=recSig;
frames=20;
sample=18*8*2*frames;
section=filter(w,1,known);
error_saved=[];
for i=1:sample:length(large_SR)-sample

recSig=large_SR(i:i+sample-1);

%% Find position of Frame in full signal
section=filter(w,1,known);
xc = xcorr(recSig,section);
middle=ceil(length(xc)/2);
xc=xc(middle:end);
%stem(real(xc));
[~,index]=max(real(xc));
section=filter(w,1,known);

if index+length(section)>sample
```

```
    continue
end
% Spectral Subtract
subtraction=1;
result=[];
section=recSig(index:index+length(section)-1);

for i=index:length(section):length(recSig)-length(section)

    result=[result;recSig(i:i+length(section)-1)-section.*
        subtraction];

end

% Try Catch if no signal found
if isempty(result)
    continue
end

%% Plots
figure;
plot(real(recSig(index:index+length(result)-1)));
hold on;
plot(real(result),'r');
plot(real(section*subtraction),'g');
xlabel('Samples')
ylabel('Magnitude')
title(['Spectral Subtraction Over Subtraction Factor ',num2str(
    subtraction)]);
```

```
legend('Original Signal','Subtracted Result','Estimate of
    Original');
hold off;
%refreshdata
%drawnow

disp('Paused');
pause(.1);
error_saved=[error_saved sum(abs(result))];

end

%% Plots
figure;
plot(error_saved(1:300));
title('Error Across Transmission Frames')
xlabel('Frames');
ylabel('Error');
```

Appendix C

SStb.py

```
#!/usr/bin/env python
##################################################
# Gnuradio Python Flow Graph
# Title: Top Block Ss R
# Generated: Thu Mar 21 17:41:25 2013
##################################################

from gnuradio import digital
from gnuradio import eng_notation
from gnuradio import gr
from gnuradio import uhd
from gnuradio.eng_option import eng_option
from gnuradio.gr import firdes
from optparse import OptionParser

class top_block_SS_R(gr.top_block):

        def __init__(self):
```

```
gr.top_block.__init__(self, "Top Block Ss R")

##################################################

# Variables
##################################################

self.samp_rate = samp_rate = 100000

##################################################

# Blocks
##################################################

self.uhd_usrp_source_0 = uhd.usrp_source(
        device_addr="",
        stream_args=uhd.stream_args(
                cpu_format="fc32",
                channels=range(1),
        ),
)
self.uhd_usrp_source_0.set_samp_rate(samp_rate)
self.uhd_usrp_source_0.set_center_freq(2.4e9, 0)
self.uhd_usrp_source_0.set_gain(25, 0)
self.gr_pll_carriertracking_cc_0 = gr.
    pll_carriertracking_cc(1.5*3.1459/200, 4, -4)
self.gr_head_0 = gr.head(gr.sizeof_gr_complex*1,
    1000000)
```

```
self.gr_file_sink_0_0 = gr.file_sink(gr.
    sizeof_float*1, "/home/sdruser/COLLINS/SS/
    received_GMSK.txt")
self.gr_file_sink_0_0.set_unbuffered(False)
self.gr_file_sink_0 = gr.file_sink(gr.
    sizeof_gr_complex*1, "/home/sdruser/COLLINS/SS
    /received.txt")
self.gr_file_sink_0.set_unbuffered(False)
self.digital_gmsk_demod_0 = digital.gmsk_demod(
        samples_per_symbol=2,
        gain_mu=0.175,
        mu=0.5,
        omega_relative_limit=0.005,
        freq_error=0.0,
        verbose=False,
        log=False,
)

##################################################

# Connections
##################################################

self.connect((self.gr_pll_carriertracking_cc_0,
    0), (self.gr_file_sink_0, 0))
self.connect((self.uhd_usrp_source_0, 0), (self.
    gr_head_0, 0))
self.connect((self.gr_head_0, 0), (self.
    gr_pll_carriertracking_cc_0, 0))
```

```
                self.connect((self.digital_gmsk_demod_0, 0), (
                    self.gr_file_sink_0_0, 0))
                self.connect((self.gr_pll_carriertracking_cc_0,
                    0), (self.digital_gmsk_demod_0, 0))

        def get_samp_rate(self):
                return self.samp_rate

        def set_samp_rate(self, samp_rate):
                self.samp_rate = samp_rate
                self.uhd_usrp_source_0.set_samp_rate(self.
                    samp_rate)

if __name__ == '__main__':
        parser = OptionParser(option_class=eng_option, usage="%
            prog: [options]")
        (options, args) = parser.parse_args()
        tb = top_block_SS_R()
        tb.run()
```

Appendix D

SigSepMRC.m

```
%Cal GRC
!sudo sysctl -w net.core.rmem_max=50000000
!sudo sysctl -w net.core.wmem_max=1048576
close all;
errors=0;
samples=0;
%% Plotting stuff
n=10;
f=zeros(n,1);
f2=zeros(n,1);

[hf,w]=freqz(f,1);
hold on;
h=semilogy(w,abs(hf),'r');
h2=semilogy(w,abs(hf),'b');
hold off;
ylim([0 2]);
title('Channel response');
```

```
grid on;

freqzz=[];

%Real message
%dd=read_char_binary('/home/sdruser/COLLINS/BLISS/Data/input.txt
    ');

dd=read_char_binary('/home/sdruser/COLLINS/Pre/Input.txt');
true_message=de2bi(dd',8,'left-msb');
mbits=reshape(true_message',size(true_message,2)*size(
    true_message,1),1);

final_BER=[];
RUNS=30;
for j=RUNS:-2:1
%% Run USRP Receiver
command=['LD_LIBRARY_PATH="" && /home/sdruser/COLLINS/BLISS/
    GNURadio/USRP_Receiver/receiver.py ',num2str(j)];
system(command);
%clc
%% Read File
%received
r=read_float_binary('/home/sdruser/COLLINS/Pre/Channel1_f.txt');
r2=read_float_binary('/home/sdruser/COLLINS/Pre/Channel2_f.txt');
r_saved=r;
r_saved2=r2;
r=double(sign(r));
r2=double(sign(r2));
```

```
%known symbols
%s_saved=[1 1 -1 1 1 1 1 -1 1 1 1 -1 1 1 -1 1]';
s_saved=[ -1     1     1     1     1    -1     1    -1];
s_saved=[s_saved s_saved s_saved]';
prea=s_saved;
s=s_saved.';

%% Find Preambles
indexs=[];
for i=1:length(r)-length(s_saved)-2000
    if sum(r(i:i+length(s_saved)-1)==s_saved)==length(s_saved)
        indexs=[indexs i];
    end
end
indexs2=[];
for i=1:length(r2)-length(s_saved)-2000
    if sum(r2(i:i+length(s_saved)-1)==s_saved)==length(s_saved)
        indexs2=[indexs2 i];
    end
end
est=length(r)/(18*8);

disp(['Indexs: ',num2str(length(indexs)),' | Estimated: ',num2str
    (est)]);

%% Look through all preambles
for k=1:min([length(indexs),length(indexs2)])
    if ((length(r_saved)-indexs(k)+1000)<0)
```

```
        disp('Break');
        break
    end
start=indexs(k);
start2=indexs2(k);


% Two frames
% message = 15 message + 3 preambles bytes
frames=2;
r=r_saved(start:start+(15+3)*8*frames).';
r2=r_saved2(start2:start2+(15+3)*8*frames).';


%% Equalize
preamble_len=24;
%n=10; %f=zeros(n,1);            % initialize equalizer at 0
%f2=zeros(n,1);
mu=.01; delta=0;                % stepsize and delay delta
for i=n+1:preamble_len                      % iterate
  %F1
  rr=r(i:-1:i-n+1).';           % vector of received signal
  e=s(i-delta)-rr'*f;           % calculate error
  f=f+mu*e*rr;                  % update equalizer coefficients
  %F2
  rr2=r2(i:-1:i-n+1).';           % vector of received signal
  e=s(i-delta)-rr2'*f2;           % calculate error
  f2=f2+mu*e*rr2;                  % update equalizer coefficients
```

```
end

%% MRC get weights
fs=sum(f);
f2s=sum(f2);
ftotal=fs+f2s;
w1=1-fs/ftotal;
w2=1-f2s/ftotal;

mu=.01;                          % stepsize
for i=preamble_len+1:length(r)          % iterate
  %F1
  rr=r(i:-1:i-n+1)';           % vector of received signal
  e=sign(f'*rr)-f'*rr;         % calculate error
  f=f+mu*e*rr;                 % update equalizer coefficients
  %F2
  rr2=r2(i:-1:i-n+1)';           % vector of received signal
  e=sign(f2'*rr2)-f2'*rr2;       % calculate error
  f2=f2+mu*e*rr2;                % update equalizer coefficients

end

%Filter
result=filter(f,1,r);
result2=filter(f2,1,r2);


%% MRC
result=sum([result.*w1;result2.*w2]);
```

```
%% Quantize to bits
final=int8(result>0);

%% Find errors
m=length(mbits);
mbits=int8(mbits);
for sh=0:n                          % if equalizer is working, one
  err(sh+1)=0.5*sum(abs(final(sh+1:m)'-mbits(1:m-sh)));
end                                 % of these delays has zero error
[~,mm]=min(err);

final=final(mm:end-mm+1);

%Add Messages together bitwise
last=floor(length(final)/length(mbits));
final=final(1:last*length(mbits));
finals=reshape(final,length(mbits),last)';
final=sum(finals,1)>=last/2;



%Decode Messages
last=round(length(final)/8);
final=final(1:last*8);% remove exending bits
f_r=reshape(final,8,length(final)/8)';
f_c=bi2de(f_r,'left-msb');
message=char(f_c)';
```

```
%% Calculate Errors
%disp('--------');
%disp(message);
samples=samples+15*8;
errors=(errors+sum(final'~=mbits));

end
% Wait between loops
disp('paused');
disp(j);
final_BER=[final_BER; errors/samples];

end

%% Plot
disp(['Final BER: ',num2str(mean(final_BER))]);
```

Appendix E

SigSep.py

```
#!/usr/bin/env python
##################################################
# Gnuradio Python Flow Graph
# Title: Top Block
# Generated: Fri Nov 16 20:33:07 2012
##################################################

from gnuradio import eng_notation
from gnuradio import gr
from gnuradio import uhd
from gnuradio import window
from gnuradio.eng_option import eng_option
from gnuradio.gr import firdes
from gnuradio.wxgui import fftsink2
from grc_gnuradio import wxgui as grc_wxgui
from optparse import OptionParser
import wx
```

```
class top_block(grc_wxgui.top_block_gui):

        def __init__(self):
                grc_wxgui.top_block_gui.__init__(self, title="Top
                    Block")
                _icon_path = "/usr/share/icons/hicolor/32x32/apps
                    /gnuradio-grc.png"
                self.SetIcon(wx.Icon(_icon_path, wx.
                    BITMAP_TYPE_ANY))

                ##################################################

                # Variables
                ##################################################

                self.samp_rate = samp_rate = 100000

                ##################################################

                # Blocks
                ##################################################

                self.wxgui_fftsink2_0 = fftsink2.fft_sink_c(
                        self.GetWin(),
                        baseband_freq=0,
                        y_per_div=10,
                        y_divs=10,
                        ref_level=0,
                        ref_scale=2.0,
```

```
            sample_rate=samp_rate*10,
            fft_size=1024,
            fft_rate=15,
            average=False,
            avg_alpha=None,
            title="FFT Plot",
            peak_hold=False,
    )
    self.Add(self.wxgui_fftsink2_0.win)
    self.uhd_usrp_source_0 = uhd.usrp_source(
            device_addr="addr0=192.168.10.2,addr1
                =192.168.10.3",
            stream_args=uhd.stream_args(
                    cpu_format="fc32",
                    channels=range(2),
            ),
    )
    self.uhd_usrp_source_0.set_clock_source("mimo",
        1)
    self.uhd_usrp_source_0.set_time_source("mimo", 1)
    self.uhd_usrp_source_0.set_samp_rate(samp_rate)
    self.uhd_usrp_source_0.set_center_freq(4.9e9
        +20000, 0)
    self.uhd_usrp_source_0.set_gain(15, 0)
    self.uhd_usrp_source_0.set_center_freq(4.9e9
        +20000, 1)
    self.uhd_usrp_source_0.set_gain(15, 1)
    self.gr_file_sink_0_0 = gr.file_sink(gr.
        sizeof_gr_complex*1, "/home/sdruser/COLLINS/
```

```
            Pre/UChannel2_S.txt")
        self.gr_file_sink_0_0.set_unbuffered(False)
        self.gr_file_sink_0 = gr.file_sink(gr.
            sizeof_gr_complex*1, "/home/sdruser/COLLINS/
            Pre/UChannel1_S.txt")
        self.gr_file_sink_0.set_unbuffered(False)

        ##################################################

        # Connections
        ##################################################

        self.connect((self.uhd_usrp_source_0, 0), (self.
            gr_file_sink_0, 0))
        self.connect((self.uhd_usrp_source_0, 1), (self.
            gr_file_sink_0_0, 0))
        self.connect((self.uhd_usrp_source_0, 1), (self.
            wxgui_fftsink2_0, 0))

def get_samp_rate(self):
        return self.samp_rate

def set_samp_rate(self, samp_rate):
        self.samp_rate = samp_rate
        self.uhd_usrp_source_0.set_samp_rate(self.
            samp_rate)
        self.wxgui_fftsink2_0.set_sample_rate(self.
            samp_rate*10)
```

```
if __name__ == '__main__':
        parser = OptionParser(option_class=eng_option, usage="%
           prog: [options]")
        (options, args) = parser.parse_args()
        if gr.enable_realtime_scheduling() != gr.RT_OK:
                print "Error: failed to enable realtime
                   scheduling."
        tb = top_block()
        tb.Run(True)
```