2-9-2010

RF channel characterization for cognitive radio using support vector machines

Thomas Atwood

Thomas Dean Atwood
Candidate

Electrical and Computer Engineering
Department

This dissertation is approved, and it is acceptable in quality and form for publication:

Approved by the Dissertation Committee:

Dr. Christos Christodoulou [illegible], Chairperson

Dr. Manel Martinez-Ramón [illegible]

Dr. Mark Gilmore [illegible]

Dr. Sudharman Jayaweera [illegible]

Dr. Pedro Embid [illegible]

Dr. Armin Doerry [illegible]

RF CHANNEL CHARACTERIZATION FOR COGNITIVE RADIO USING SUPPORT VECTOR MACHINES

BY

THOMAS DEAN ATWOOD

B.S., Electrical Engineering, Southern Illinois Univeristy, 1984
M.S., Electrical Engineering, New Mexico State Univeristy, 1993

DISSERTATION

Submitted in Partial Fulfillment of the
Requirements for the Degree of

Doctor of Philosophy

Engineering

The University of New Mexico
Albuquerque, New Mexico

December, 2009

DEDICATION

To Sue, my wife and best friend. I could not have done this without your love and support. I love you dearly.

To My Lord Jesus Christ. Thank You for the talent and the abilities to get here.

To my uncles, Richard Ulrich and Kurt Ulrich. Your lifelong examples have been a great inspiration.

To my mother, Emilie Atwood. Your quiet strength and dignity is beyond description.

ACKNOWLEDGEMENTS

To Dr. Christos Christodoulou and Dr. Manel Martinez-Ramón. Your leadership and guidance through this effort is most appreciated.

To my committee, Dr. Sudharman Jayaweera, Dr. Mark Gilmore, Dr. Pedro Embid, and Dr. Armin Doerry. A finer committee could not be found.

To my children, Jessica and Daniel. Thanks for understanding when Dad wasn't there.

Thanks to Sandia National Laboratories for their support through the University Part-Time Program.

Thanks to the lunch bunch for the humor.

RF CHANNEL CHARACTERIZATION FOR COGNITIVE RADIO USING SUPPORT VECTOR MACHINES

BY

THOMAS DEAN ATWOOD

ABSTRACT OF DISSERTATION

Submitted in Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy

Engineering

The University of New Mexico
Albuquerque, New Mexico

December, 2009

RF CHANNEL CHARACTERIZATION FOR COGNITIVE RADIO USING SUPPORT VECTOR MACHINES

by

Thomas Dean Atwood

B.S., Electrical Engineering, Southern Illinois University, 1984

M.S., Electrical Engineering, New Mexico State University, 1993

Ph.D., Engineering, University of New Mexico, 2009

ABSTRACT

Cognitive Radio promises to revolutionize the ways in which a user interfaces with a communications device. In addition to connecting a user with the rest of the world, a Cognitive Radio will know how the user wants to connect to the rest of the world as well as how to best take advantage of unused spectrum, commonly called “white space”. Through the concept of Dynamic Spectrum Acccess a Cognitive Radio will be able to take advantage of the white space in the spectrum by first identifying where the white space is located and designing a transmit plan for a particular white space.

In general a Cognitive Radio melds the capabilities of a Software Defined Radio and a Cognition Engine. The Cognition Engine is responsible for learning how the user interfaces with the device and how to use the available radio resources while the SDR is the interface to the RF world. At the heart of a Cognition Engine are Machine Learning Algorithms that decide how best to use the available radio resources and can learn how the user interfaces to the CR.

To decide how best to use the available radio resources, we can group Machine Learning Algorithms into three general categories which are, in order of computational cost: 1.) Linear Least Squares Type Algorithms, e.g. Discrete Fourier Transform (DFT) and their kernel versions, 2.) Linear Support Vector Machines (SVMs) and their kernel versions, and 3.) Neural Networks and/or Genetic Algorithms. Before deciding on what to transmit a Cognitive Radio must decide where the white space is located. This research is focused on the task of identifying where the white space resides in the spectrum, herein called RF Channel Characterization. Since previous research into the use of Machine Learning Algorithms for this task has focused on Neural Networks and Genetic Algorithms, this research will focus on the use of Machine Learning Algorithms that follow the Support Vector optimization criterion for this task. These Machine Learning Algorithms are commonly called Support Vector Machines.

Results obtained using Support Vector Machines for this task are compared with results obtained from using Least Squares Algorithms, most notably, implementations of the Fast Fourier Transform. After a thorough theoretical investigation of the ability of Support Vector Machines to perform the RF Channel Characterization task, we present results of using Support Vector Machines for this task on experimental data collected at the University of New Mexico.

TABLE OF CONTENTS

LIST OF FIGURES xi
LIST OF TABLES xv
1 Introduction 1
2 Cognitive Radios and Support Vector Machines 7
  2.1 Cognitive Radio 7
  2.2 Why SVMs instead of the Discrete Fourier Transform (Fast Fourier Transform)? 9
  2.3 Support Vector Machines 13
  2.4 Fundamentals of SVMs for Classification 14
    2.4.1 Linear, Separable - Primal Formulation 14
    2.4.2 Linear, Separable - Dual Formulation and Lagrangian Solution 19
    2.4.3 Linear, Non-Separable – Primal Formulation 20
    2.4.4 Linear, Non-Separable – Dual Formulation and Lagrangian Solution 21
    2.4.5 Kernel Trick and Mercer Condition 22
    2.4.6 Non-Linear, Separable 23
  2.5 SVMs for Regression 25
  2.6 Parametric Spectrum Estimation 29
  2.7 Non-Parametric Spectrum Estimation 40
3 Modeling and Simulation 43
  3.1 Preliminaries 43
  3.2 Spectrum Estimations Algorithms 47
  3.3 Single Bluetooth Transmitter 50
    3.3.1 Introduction 50
    3.3.2 Results 51
  3.4 Single WiFi Transmitter 60
    3.4.1 Introduction 60
    3.4.2 Results 61
  3.5 Bluetooth Piconet 73
    3.5.1 Introduction 73
    3.5.2 Results 74
  3.6 Bluetooth and WiFi 83
    3.6.1 Introduction 83
    3.6.2 Results 81
  3.7 Performance with Amplitude Variations 87
4 Experimental Setup 96
  4.1 Description 96
    4.1.1 RF Background Characterization 104
5 Experimental Results 106
  5.1 Discussion and Presentation 106
6 Conclusions 119
  6.1 Summary, Discussions and New Contributions 119
  6.2 Recommendations for Future Work 122

7 References 124
8 Appendices 133
A Basic MATLAB Code for “Non-Parametric” 133
B Basic MATLAB Code for “Parametric” 136
C Additional Information on Adding Noise to Signals 138
D Basic MATLAB Code for WiFi Estimator 144
E Further Parametric Estimates for a Single Bluetooth Transmitter 148
F 2, 5.5 and 11 Mbps WiFi Simulink Models 156
F.1: 2 Mbps WiFi Simulink Model 156
F.2: 5.5 Mbps WiFi Simulink Model 157
F.3: 11 Mbps WiFi Simulink Model 158
G Further Parametric Estimates for a Single WiFi Transmitter 159
H Piconet Models 162
H.1: MATLAB/Simulink Piconet Model 162
H.2: Detail of an Individual Transmitter 163
I Further Estimates for the Bluetooth and WiFi Combinations 164
J Hardware Data Sheets 166
J.1 MiniCircuits LNA– ZX60-3011 166
J.2 Hewlett Packard Power Splitter 11667B 167
J.3 Mark Microwave Mixer M10208LA 168

LIST OF FIGURES

Figure 1.1: Cognitive Radio Block Diagram 3
Figure 2.1: Spectrum occupied by two 802.11 transmitters and one Bluetooth Piconet (during one 625µS period) 11
Figure 2.2: Optimal Separation Hyperplane (Solid Line) and two margin Hyperplanes (dashed line) in a binary classification example; Support Vectors are bolded 17
Figure 2.3: Non-linearly separable problem in 2-D space becomes linearly separable in 3-D space after a non-linear transformation “φ” is applied 23
Figure 2.4: Graphical representation of a “loss” function 26
Figure 2.5: Composite Cost Function 28
Figure 3.1: Average White Gaussian Noise (AWGN) 46
Figure 3.2: Impulsive Noise 46
Figure 3.3: Block Noise 47
Figure 3.4: Proposed Analysis Flowchart 48
Figure 3.5: Single Bluetooth Transmitter Simulink Model 51
Figure 3.6: Single Bluetooth Transmitter – AWGN Only 53
Figure 3.7: Spectral Estimate, AWGN + Impulse, Sigma = 1 56
Figure 3.8: Spectral Estimate, AWGN + Block, AWGN = 1 56
Figure 3.9: Spectral Estimate, AWGN + Impulse + Block, AWGN = 1 57
Figure 3.10: Parametric Estimator – Channel 42 – AWGN, Sigma = 1 59
Figure 3.11: 1 Mbps WiFi Transmitter Simulink Model 61
Figure 3.12: Single WiFi Transmitter – SVM vs FFT – 1 Mbps, Sigma = 1 62
Figure 3.13: Spectral Estimate, AWGN + Impulse, Sigma = 1 65
Figure 3.14: Spectral Estimate, AWGN + Block, Sigma = 1 65
Figure 3.15: Spectral Estimate, AWGN + Block + Impulse, Sigma = 1 66
Figure 3.16: Spectral Estimate – 2 Mbps WiFi Signal 68
Figure 3.17: Spectral Estimate – 5.5 Mbps WiFi Signal 68
Figure 3.18: Spectral Estimate – 11 Mbps WiFi Signal 69
Figure 3.19: Spectral Estimate, Double WiFi, Adjacent Channels, Sigma = 1 70
Figure 3.20: Spectral Estimate, Double WiFi, Spread Channels, A+B+I, Sigma = 5 71
Figure 3.21: Parametric Estimator, Single WiFi, AWGN, Sigma = 1 72
Figure 3.22: Parametric Estimator, Single WiFi, A+B+I, Sigma = 5 72
Figure 3.23: Parametric Estimator, Double WiFi, Sigma =1 73
Figure 3.24: Piconet Network, AWGN Only, Sigma = 1 75
Figure 3.25: Piconet Network, AWGN Only, Sigma = 5 75

Figure 3.26: Piconet Network, AWGN + Impulsive, Sigma = 1 76
Figure 3.27: Piconet Network, AWGN + Impulsive, Sigma = 5 76
Figure 3.28: Piconet Network, AWGN + Impulsive + Block, Sigma = 1 77
Figure 3.29: Piconet Network, AWGN + Impulsive + Block, Sigma = 5 77
Figure 3.30: Two Bluetooth Transmitters, AWGN, Sigma = 1 78
Figure 3.31: Five Bluetooth Transmitters, AWGN, Sigma = 1 79
Figure 3.32: Five Bluetooth Transmitters, AWGN, Sigma = 1,
Bad Estimate 80
Figure 3.33: Six Bluetooth Transmitters, AWGN, Sigma = 1 80
Figure 3.34: Seven Bluetooth Transmitters, AWGN, Sigma = 1 81
Figure 3.35: Seven Bluetooth Transmitters, AWGN, Sigma = 1, Bad Estimate.. 82
Figure 3.36: Eight Bluetooth Transmitters, AWGN, Sigma = 1, Bad Estimate.... 82
Figure 3.37: WiFi and Bluetooth, AWGN, Sigma = 1 84
Figure 3.38: WiFi and Bluetooth, A+B+I, Sigma = 5 84
Figure 3.39: Parametric, WiFi and Two Bluetooth, AWGN, Sigma = 1 85
Figure 3.40: Parametric, WiFi and Five Bluetooth, AWGN, Sigma = 1 86
Figure 3.41: Parametric, WiFi and Six Bluetooth, AWGN, Sigma = 1................. 86
Figure 3.42: Amplitude Difference Estimate,
Channel 7 @ 7 dB below Master 89
Figure 3.43: Amplitude Difference Estimate,
Channel 7 @ 8 dB below Master 89
Figure 3.44: Amplitude Difference Estimate,
Channel 7 @ 9 dB below Master 90
Figure 3.45: Amplitude Difference Estimate,
Channel 7 @ 12 dB below Master 90
Figure 3.46: Amplitude Difference Estimate,
Channel 7 @ 15 dB below Master 91
Figure 3.47: Amplitude Difference Estimate,
Channel 7 @ 18 dB below Master 91
Figure 3.48: Amplitude Difference Estimate,
All Channels @ 12 dB below Master 92
Figure 3.49: Amplitude Difference Estimate,
Channel 7 @ 7 dB below Master, AWGN 94
Figure 3.50: Amplitude Difference Estimate,
Channel 7 @ 7 dB below Master, AWGN + Impulsive 92
Figure 3.51: Amplitude Difference Estimate,
Channel 7 @ 7 dB below Master, AWGN + Impulsive + Block 94
Figure 4.1: Notional Schematic Diagram of Test Setup 98

Figure 4.2: View inside Chamber Showing Transmit Antenna 99
Figure 4.3: View inside Chamber Showing Receive Antenna 100
Figure 4.4: Chamber Equipment Setup #1 101
Figure 4.5: Chamber Equipment Setup #2 102
Figure 4.6: Equipment Setup – ECE Quad 103
Figure 4.7: RF Background – Site 1 105
Figure 4.8: RF Background – Site 2 105
Figure 5.1: Filter – Magnitude Response 107
Figure 5.2: Lab Test, Single Bluetooth at 2.4 GHz 107
Figure 5.3: Spectrum Analyzer Plot for Figure 5.2 108
Figure 5.4: Site 1, Piconet 109
Figure 5.5: Site 1, WiFi 109
Figure 5.6: Spectrum Analyzer Plot for Figure 5.5 110
Figure 5.7: ECE1, 6 October 2006 @ 2 PM 111
Figure 5.8: Spectrum Analyzer Plot for Figure 5.7 111
Figure 5.9: ECE2, 6 October 2009 @2:20 PM 112
Figure 5.10: Spectrum Analyzer Plot for Figure 5.9 112
Figure 5.11: ECE6, 7 October 2009 @ 1:30 PM 113
Figure 5.12: Spectrum Analyzer Plot for Figure 5.11 113
Figure 5.13: Final, 7 October 2009 @ 2:00 PM 114
Figure 5.14: Spectrum Analyzer Plot for Figure 5.13 114
Figure 5.15: Spectrum Analyzer Block Diagram 115
Figure 5.16: Spectrum Analyzer Sweep at Site 2 after 2 Averages 117
Figure 5.17: Spectrum Analyzer Sweep at Site 2 after 14 Averages 118
Figure C.1: Time History, AWGN, SIGMA = 1 139
Figure C.2: Mean Square Spectrum, AWGN, SIGMA = 1 139
Figure C.3: Time History, Impulsive, SIGMA = 1 140
Figure C.4: Mean Square Spectrum, Impulsive, Sigma = 1 140
Figure C.5: Time History, Block, Sigma = 1 141
Figure C.6: Mean Square Spectrum, Block, Sigma =1 141
Figure C. 7: Time History, AWGN+Impulsive+Block, Sigma = 1 142
Figure C.8: Mean Square Spectrum, AWGN+Impulsive+Block, Sigma = 1 142
Figure C.9: Time History, AWGN+Impulsive+Block, Sigma = 5 143
Figure C.10: Mean Square Spectrum, AWGN+Impulsive+Block, Sigma = 5 143
Figure E.1: Parametric Estimator, Channel 70, AWGN, Sigma =1 148
Figure E.2: Parametric Estimator, Channel 42, A+1, Sigma = 1 148
Figure E.3: Parametric Estimator, Channel 70, A+I, Sigma = 1 149
Figure E.4: Parametric Estimator, Channel 42, A+B, Sigma = 1 149

Figure E.5: Parametric Estimator, Channel 70, A+B, Sigma = 1 150
Figure E.6: Parametric Estimator, Channel 42, A+B+I, Sigma = 1 150
Figure E.7: Parametric Estimator, Channel 70, A+B+I, Sigma = 1 151
Figure E.8: Parametric Estimator, Channel 42, AWGN, Sigma = 5 151
Figure E.9: Parametric Estimator, Channel 42, A+I, Sigma = 5........................ 152
Figure E.10: Parametric Estimator, Channel 42, A+B, Sigma = 5 152
Figure E.11: Parametric Estimator, Channel 47, A+B+I, Sigma = 5 153
Figure E.12: Parametric Estimator, Channel 70, AWGN, Sigma = 5 153
Figure E.13: Parametric Estimator, Channel 70, A+I, Sigma = 5...................... 154
Figure E.14: Parametric Estimator, Channel 70, A+B, Sigma = 5 154
Figure E.15: Parametric Estimator, Channel 70, A+B+I, Sigma = 5 155
Figure G.1: Parametric Estimator, Single WiFi, AWGN, Sigma = 5 159
Figure G.2: Parametric Estimator, Single WiFi, A+I, Sigma = 1........................ 159
Figure G.3: Parametric Estimator, Single WiFi, A+I, Sigma = 5........................ 160
Figure G.4: Parametric Estimator, Single WiFi, A+B, Sigma = 1 160
Figure G.5: Parametric Estimator, Single WiFi, A+B, Sigma = 5 161
Figure G.6: Parametric Estimator, Single WiFi, A+B+I, Sigma = 1 161
Figure I.1: WiFi and Bluetooth, AWGN, Sigma = 5 164
Figure I.2: WiFi and Bluetooth, A+I, Sigma = 1... 164
Figure I.3, WiFi and Bluetooth, A+I, Sigma = 5... 165
Figure I.4, WiFi and Bluetooth, A+B+I, Sigma = 1 165

LIST OF TABLES

Table 2.1: Advantages/Disadvantages of SVR algorithms vs the DFT (FFT) 9

Table 2.2: Summary of SVM-like Algorithm 39

Table 3.1: Wireless Characteristics in the 2.4 GHz ISM band 44

Table 3.2: SVM vs FFT Amplitude Ratios, AWGN ONLY 54

Table 3.3: SVM vs FFT Amplitude Ratios, AWGN + Impulse, AWGN + Block and AWGN + Impulse + Block 55

Table 3.4: Mean Frequency Error +/- Standard Deviation 57

Table 3.5: SVM vs FFT Amplitude Ratios, AWGN only 63

Table 3.6: SVM vs FFT Amplitude Ratios, AWGN + Impulse, AWGN + Block and AWGN + Impulse + Block 64

Table 3.7: WiFi Channel Numbers and Equivalent Frequencies 69

Table 3.8: Allowable Transmit Power (EIRP) for WiFi and Bluetooth 87

Table 4.1: Airlink Tests Equipment List 97

1 INTRODUCTION

Demand for wireless services continues to explode [1]. Consider just the proliferation of Bluetooth-enabled devices, everything from Nintendo’s Wii Gaming Platform communicating to a remote gaming accessory to cameras downloading pictures and video to a user’s home computer. Since most frequency spectrum is used by licensed services, most of the growth for these new wireless services is in the Instrument, Scientific and Medical (ISM) bands because most uses in these bands are unlicensed [2]. Unlicensed doesn’t necessarily mean unregulated, but as long as a communications device meets certain minimal requirements, the device can operate in one of the ISM Bands [3].

As an example, consider the 2.4 GHz ISM Band (2.4 – 2.485 GHz), where we have Bluetooth (IEEE 802.15.1), Wireless Local Area Networks (WLAN – IEEE 802.11 b/g/n), and Wireless Personal Area Networks (WPAN – IEEE 802.15.4, “Low-Rate WPAN and its derivative called ZIGBEE” and IEEE 802.15.3, “High-Rate WPAN”) [23] – [26]. Considering the myriad of services that use the 2.4 GHz ISM Band, one can easily conclude that there would be little if any available spectrum in this particular ISM Band.

However, occupancy studies conducted in the US and abroad has shown that the total percent of time that the 2.4 GHz band is unoccupied can range as high as 82.6 % in a metropolitan area such as Chicago, Illinois, USA and London, England [4] - [7]. Additionally, even when a particular piece of spectrum is occupied, we can see from [4] – [7] that other parts of the spectrum are unoccupied. Assuming a communication device that possesses the intelligence to understand the dynamic nature of spectrum usage in a particular band, it is apparent from [4] - [7] that there are numerous opportunities for such a device to take advantage of unused spectrum. As early as 1999 such a device, called a "Cognitive Radio", was identified as being feasible [9].

In 2002, the United States Federal Communications Commission (USFCC) recognized the need for such a device and issued ET Docket No. 02-135 that addressed the issue of "white space" in spectrum. This report identified "Cognitive Radio" as one method whereby the white space could be utilized [8, pp 67 and 71].

As mentioned earlier the term "Cognitive Radio (CR)" was first coined by Mitola, et al. in [9], [10] to introduce a concept that combined Software Defined Radios (SDRs) with Computational Intelligence. The purpose was to create a device that could learn about the radio environment, allowing the device to characterize the RF channel and to take advantage of the white spaces in a particular spectrum band through a technique referred to as Dynamic Spectrum Access (DSA). DSA also allows a CR to operate in multiple frequency bands, subject to antenna availability. A well designed CR operating in the 2.4 GHz ISM Band should be able to act as a Bluetooth device, and, when the user requires the change, act as a Wi-Fi device without having to go through any hardware changes. All the while, the CR must change device modes without interfering with other users of the band, in accordance with [14].

The computational intelligence allows the CR to learn how the user interacts with the radio. As an example, imagine a user who has a CR that can operate as described above, either as a Bluetooth device or a Wi-Fi device. However, through past experience the device has learned that the user doesn't want the CR to access a Wi-Fi network, say an unsecured Wi-Fi network.

Consider a generic block diagram of a CR as depicted in Figure 1.1 [Ref. 12, adapted from Figure 7.1, page 222]

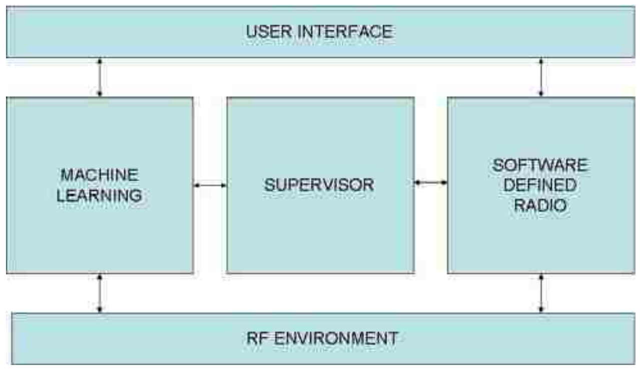


Figure 1.1: Cognitive Radio Block Diagram

The Cognition Engine (CE), which is part of the "Machine Learning" (ML) block in the above diagram, has the responsibility to properly characterize the RF environment and learn how the user wants their CR to interface to the outside world. In this context we can see that the term "properly characterize" means more than just acting as a spectrum analyzer. We can see that the CE must also be able to distinguish between the various types of services, and their attendant signaling schemes, that may be present in the channel. When we consider how rapidly things change in the RF environment of the 2.4 GHz band, 1600 times per second in the case of Bluetooth, we can see that techniques in use by the ML block have to be very fast and have multiple capabilities. In addition, these techniques, to enable pocket-sized CR devices, must be compact in the context of hardware resources necessary to implement the techniques. Many researchers have investigated, and written about, how a network of CRs can be used to share spectrum usage information (typically referred to as “Cooperative Sensing), but few, if any, have investigated how an individual device can perform the spectrum estimation task before feeding this information to a network [70] – [73] . This dissertation investigates a technique based on a class of machine learning algorithms called “Support Vector Machines” (SVMs) [40], [42] that can be used by the CE to properly characterize, on a real time basis, the RF environment.

This dissertation contains 6 Chapters. In Chapter 2 we discuss in more detail the concept of cognitive radio (CR), define what is meant by a CR, introduce Support Vector Machines (SVMs), discuss the role of Support Vector Machines (SVMs) in CRs and discuss the focus of this research in the 2.4 GHz ISM band with particular attention paid to the noise environment that is present in the band due to the presence of different wireless services and other sources. Also in Chapter 2 we present a mathematical treatment of SVMs starting with the simplest type of classification, i.e. binary classification with linear, separable algorithms, to illustrate the concept of “Support Vectors”. We then progress to linear, non-separable algorithms and then to non-linear, separable algorithms. For the non-linear algorithms we discuss kernel construction and the needed Mercer Condition for kernels that we then use to introduce a kernel for our non-linear algorithm that includes information of the communications channel as well as information about the modulations that may be present. We wrap up Chapter 2 by discussing how a modification in the FFT spectrum estimate technique, i.e. the loss function, can lead to our linear spectrum estimation technique.

In Chapter 3 we present results using SVRs as the spectral estimation tool with various combinations of the two primary services of services present in the 2.4 GHz band, i.e. Bluetooth (IEEE 802.15.1) and Wireless Local Area Networks (WLAN – IEEE 802.11 b/g/n). Data for these investigations are generated in the MATLAB/SIMULINK environment and the data from the SIMULINK models are imported directly into the MATLAB workspace for processing.

In Chapter 4 we discuss the experimental setups that were used to test these techniques in environments that present an increasingly harsher, i.e. more realistic, noise environment.

In Chapter 5 we present and discuss the results of the experimental campaign. The experimental campaign was divided into 2 different environments, ranging from propagation experiments in a Faraday cage (Room L206, Electrical and Computer Engineering (ECE) Building on campus) giving the most controllable noise environment to propagation experiments outside the ECE building in the presence of multiple WiFi signals and various Bluetooth devices.

Finally, in Chapter 6 we present our summaries, conclusions, and recommendations for future research.

2 COGNITIVE RADIOS AND SUPPORT VECTOR MACHINES

2.1 Cognitive Radio

As mentioned earlier the term “cognitive radio” was first coined by Mitola, et al. in [9], [10] to introduce a concept that combined Software Defined Radios (SDRs) with Computational Intelligence. According to the IEEE’s 1900.1 standard, “Cognitive Radio is a type of radio in which communication systems are aware of their environment and internal state and can make decisions about their radio operating behavior based on that information and predetermined objectives [11].” In general a CR melds the capabilities of a software-defined radio (SDR) with a cognition engine (CE).

There are many definitions for what an SDR is, but the International Telecommunications Union (ITU) definition may be the most succinct: “A radio in which RF operating parameters including but not limited to, frequency range, modulation type, or output power can be set or altered by software, or the technique by which this is achieved [13].”

SDR has its roots in a program funded by Air Force Rome Labs (AFRL) in 1987 that developed a programmable modem to replace the “Integrated Communications, Navigation, and Identification Architecture” (ICNIA). This led to the development of Speakeasy I and II and eventually the stand up of the Joint Tactical Radio System (JTRS) Joint Program Executive Office (JPEO) in 2005 [15, section 1.3, pp. 4 - 8], [15]. Therefore, given that SDR and SDR technology has been under development for 20+ years we can reasonably say that SDR technology is mature. On the other hand, development of the ML block, especially development of the algorithms that enable an ML block is a relatively new research area [16], [17], [18, Chapter 1].

As stated earlier, the ML block is responsible for a variety of tasks, but we can represent these tasks in the following three general categories:

1. Perception of the radio environment,
2. Learning from the environment and adapting the performance of the CR to the statistical variations in the incoming RF stimuli, and
3. Learning how the user interacts with the radio.

With regards to information on the radio environment, the CR, to have maximum flexibility for the user, must be able to operate in diverse radio environments with no a-priori knowledge of the radio environment. As an example, consider a business traveler who travels from Washington, DC to Tokyo, Japan who doesn't want to have to carry around multiple communication devices. That traveler wants to get off the plane at their destination and have the device that they carry be instantly useful, regardless of the communications environment.

As mentioned earlier the entire area of research for the CE, especially as applied to the task of spectrum estimation, is relatively new. To date, the majority of the research in algorithms for the CE has centered on the use of Genetic Algorithms (GA) or Neural Networks (NN) as the heart of the CE [19] - [22]. Therefore, this PhD research chose to focus on the use of Machine

Learning Algorithms that follow the Support Vector optimization criterion for the task of spectral estimation. These Machine Learning Algorithms are commonly called Support Vector Machines. This approach is based on Support Vector Machines acting in a regression mode to extract parameters, i.e. occupied bandwidth, modulation types, etc, that define, based on operating standards, what type of radio service is in use in a particular chunk of spectrum. Because of the demand for services in the 2.4 GHz ISM band and as a practical limit to these initial exploration we are limiting our research to the 2.4 GHz Instrumentation, Scientific and Medical (ISM) band.

2.2 Why SVMs Instead Of The Discrete Fourier Transform (Fast Fourier Transform)?

We use SVMs because of their abilities in the presence of non-Gaussian noise, the ability of the algorithm to use a sparse input vector and other advantages detailed in Table 2.1.

Table 2.1:

Advantages/Disadvantages of SVM algorithms vs the DFT (FFT)

	Advantages	Disadvantages
FFT	Computational requirements are low.	Uses the Least Squares error which is suboptimal for non-Gaussian noise.
	A-priori knowledge of signal models is not required.	

	Many implementations of FFT algorithms for spectral estimation are available and can be used right away.	
SVM	Acting on the margins, i.e. the Support Vectors, yields a sparse solution.	Computational time is long.
	Capacity control that prevents overfitting to outliers.	More computational resources required.
	Training time depends only on the dimensionality of the input space, not the feature space.	A-priori knowledge of signal model required to construct proper kernels, if needed
	Dual formulations yield a global solution with no local minima.	Robust to non-Gaussian Noise
	Use of kernels as a similarity measure allows the tailoring of the algorithm to the specific problem at hand.	Parameter selection, e.g. # of points, is required to adequately use these SVM techniques.

The long-term goal of this research is to develop techniques that can enable a cognitive radio that is as portable as any cell phone currently in production. Therefore, to be useful in such a communications device the spectral sensing described above has to be performed using small, broadband and non-directive antennas under the assumption of heavy noise and interference [27] - [31].

Further, the spectral environment that is present at any time in the 2.4 GHz ISM band presents an essentially non-Gaussian noise environment to any algorithm that is trying to estimate the spectrum. As an example of the type of RF environment present in the 2.4 GHz ISM band, consider Figure 2.1 that shows a notional representation of the spectrum occupied by two 802.11 transmitters and

1 Bluetooth Piconet. This spectrum only stays constant for 625 μs before the Bluetooth transmitters change the channel in which they are transmitting. Since the Bluetooth transmitters utilize a Frequency- Hopped Spread Spectrum channel access scheme, we can consider the total ensemble of spectrum users in this band to be a random process. In addition to the noise sources that are present from the other services interfering with each other, there is the Electromagnetic Interference (EMI) present from microwave ovens [61] – [63]. The combination of these noise sources can appear as "blocks" of noise, i.e. on for periods of time or 'impulsive" noise, i.e. on for very short periods of time.

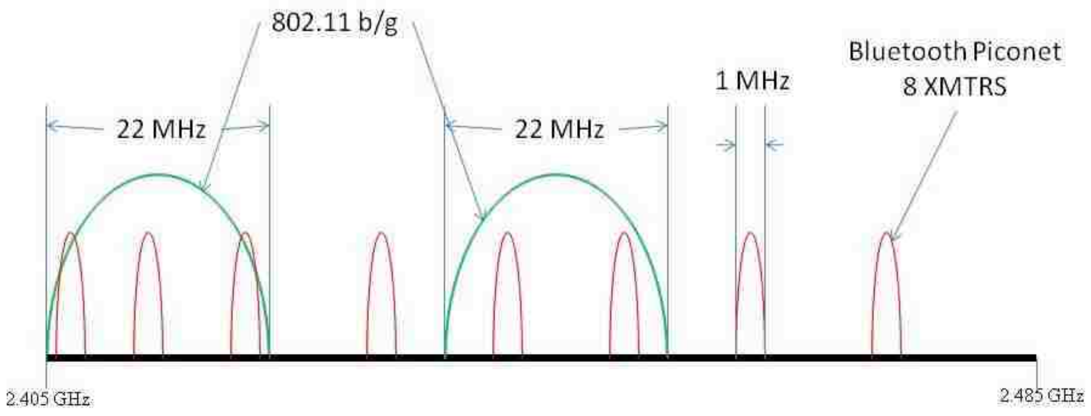


Figure 2.1: Spectrum occupied by two 802.11 transmitters and one Bluetooth Piconet (during one 625μS period)

Given these requirements we can see, based on Table 2.1, that SVMs offer distinct advantages that outweigh the FFT as the spectral estimating technique for Cognitive Radio.

Traditional spectral estimation, as implemented through the Fast Fourier Transform (FFT), is characterized by tradeoffs in windowing, time domain averaging and frequency domain averaging of sampled data obtained from random processes in order to balance the need to reduce sidelobes and to ensure adequate spectral resolution [50] – [52].

Another way to think about the use of SVMs instead of the FFT is to consider some of the mathematical constraints that are placed on the use of the FFT as a spectral estimation tool.

According to the Weiner-Khinchin theorem we can relate the autocorrelation, $R_{xx}(\tau)$, via the Fourier Transform, to the Power Spectral Density (PSD), $P(f)$, by:

$$P(f) = \int_{-\infty}^{\infty} R_{xx}(\tau)\exp(-j2\pi f\tau)d\tau \tag{1}$$

For our purposes we may not have the autocorrelation function available, but if we make the assumption that the random process is ergodic in the mean and standard deviation we can rewrite the PSD as follows:

$$P(f) = \lim_{T\to\infty} E\left[\frac{1}{2T}\left|\int_{-\infty}^{\infty} x(t)\exp(-j2\pi ft)dt\right|^2\right] \tag{2}$$

Unfortunately, as can be seen from the spectrum presented in Figure 2.1, we cannot make the assumption that the process is ergodic, i.e. constant for a long period of time. This fact is important because according to [53] the performance of spectral estimations may be characterized by an inequality that relates the stability-time- bandwidth product by:

$$\Delta S \Delta T \Delta f > 1 \tag{3}$$

where ΔT is the time interval over which we have taken our time samples, Δf is the resolution in Hertz and ΔS is a measure of the stability of the frequency estimate. If we desire a small ΔS and our ΔT is small (remember that the time during which the spectrum in the band is constant is short) we must have a large Δf . We will see later that this is exactly what happens when using the FFT to estimate the transmitters in the band especially when we desire to get Δf in the kilohertz range.

2.3 Support Vector Machines

For the spectral estimation task we present an approach based on Support Vector Machines in a regression mode (SVR) [32] - [34]. These methods implicitly use a cost function (also called the loss function) which is linear, thus being part of the so-called robust regression methods. While SVR methods are sub-optimal under Gaussian noise, they are very robust under non-Gaussian noise conditions [35], [74], [75].

First introduced by Vladimir Vapnik and his co-workers in the early 90's, Support Vector Machines (SVM) are a very specific class of machine learning algorithms, characterized by the absence of local minima, the sparseness of the solution and the capacity control obtained by acting on the margin, or on other dimension independent quantities such as the number of support vectors [40], [42]. Being particularly useful for solving classification and regression problems, SVM based techniques have achieved superior performances in a wide variety of real world problems due to their generalization ability and robustness against noise and outliers.

As a review of the extent to which SVMs have been used in real world problems the following quick literature review of the use of SVMs in a classification mode is presented to give the reader a feel for how useful these techniques have become in a little over 10 years. SVMs have been used for isolated handwritten digit recognition, [43], [44], [45], [46], object recognition [47], speaker identification [48], and face detection in images [49]. A more detailed literature review is contained in [33].

Before discussing the issue of spectrum estimation we discuss the foundation of SVMs, touching first on the topic of SVM Classification, then a discussion of SVM Regression (SVR). The following discussion is taken from a variety of references, in particular [34], [36] - [39], [64], [65].

2.4 FUNDAMENTALS OF SVMS FOR CLASSIFICATION

2.4.1 Linear Separable - Primal Formulation

The most fundamental SVM classification problem is binary classification. In the binary case, a training set is spanned by a collection of input vectors $\mathbf{x}_i$ (i=1,…,l), with each vector being assigned a label $y_i \in \{+1, -1\}$ to indicate the corresponding class to which each belongs. Each input vector together with its label represents a specific data point lying in the input space. Each training pair is assumed to be generated i.i.d (independent and identically distributed) from an unknown probability distribution $\mathrm{P}(\mathbf{x}, y)$,

$$(\mathrm{x},y),\ldots,(\mathrm{x}_n,y_n) \in \mathrm{R}^N \times Y, Y = \{-1,+1\} \tag{4}$$

We need to find a function f such that the next set of unseen examples $(\mathbf{x}, y)$

are correctly classified. This function is defined by a set of possible mappings $\mathbf{X} \mapsto f(\mathbf{x},\alpha)$, where the functions $f(\mathbf{x},\alpha)$ themselves are labeled by the adjustable parameters α. The learning machine that is defined by these functions is assumed to be deterministic, i.e. for a given input $\mathbf{x}$, and choice of α, the output is always the same.

The best function f would minimize the expected error, or risk,

$$R(f)=\int c(f(\mathrm{x}),y)dP(\mathbf{x},\mathrm{y}) \tag{5}$$

where c is a loss function. For our simple binary classification problem, where $Y=\{-1,+1\}$, a common choice for the misclassification error is: $c(f(\mathrm{x}),y)=\frac{1}{2}|f(\mathrm{x})-y|$. The problem in evaluating $R(f)$ is the fact that we do not know the probability function P.

We can proceed by using the training data to approximate Equation 5 by a finite sum:

$$\int f(\mathrm{x})P=\frac{1}{m}\sum_{i=1}^{m} f(\mathrm{x}_i) \tag{6}$$

This leads to the empirical risk (See Definition 3.4, page 67, Reference [37]) defined as:

$$R_{emp}(f)=\frac{1}{m}\sum_{i=1}^{m} c(\mathrm{x}i, yi, f(\mathrm{x}i)) \tag{7}$$

We can see that we need to limit the complexity of the class of functions from which f is drawn [42]. We can introduce a regularization term to limit this complexity (See Reference [65], Chapter 5 for a more thorough discussion on regularization). A more specific way to control the complexity is through VC (Vapnik-Chervonenkis) Theory, with the attendant VC Dimension and the structural risk minimization (SRM) principle [40] and [42]. The VC Dimension is a property of a set of functions $\{f(\alpha)\}$ and can be defined for various classes of function f . In our case of the binary classification we have a given set of ℓ points that can be labeled in all possible 2^{ℓ} ways. If we can find a member of the set $\{f(\alpha)\}$ which correctly assigns those labels, the set of points is said to be "shattered" by that set of functions. The VC Dimension for a set of functions $\{f(\alpha)\}$ is defined as the maximum number of training points that can be "shattered" by $\{f(\alpha)\}$. See [33, especially Sections 2.1 – 2.5] for a more detailed explanation with some examples to illustrate the VC Dimension concept in more detail.

Returning to our original input samples we have the task of finding a hyperplane that separates the training samples into two classes with maximal separation margin, as demonstrated in Fig. 2.2. This hyperplane actually represents the optimal classifying function being found through a learning process that is carried out on the knowledge extracted from the training data set, such that the classifying function is able to predict the labels of novel samples with minimum error.

We choose functions of the form:

$$f(\mathrm{x}) = (\mathbf{w} \bullet \mathrm{x}) + b \tag{8}$$

where $\mathbf{w}$ is normal to the hyperplane.

Vapnik showed in [42] that the VC bounds can be bounded in terms of another quantity, i.e. the margin, as depicted in Figure 2.2.

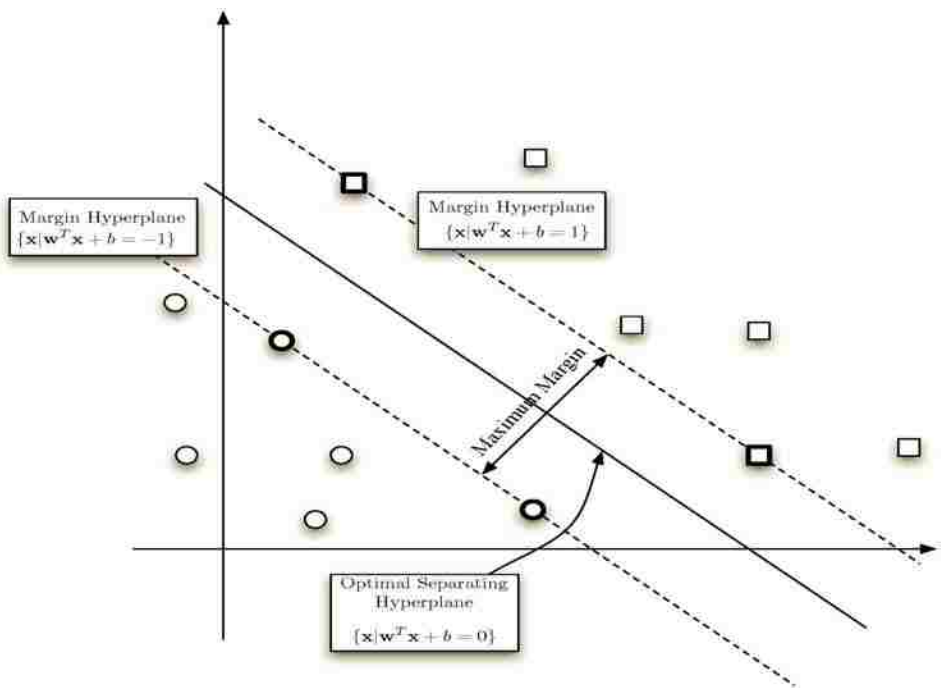

Margin Hyperplane $\{\mathbf{x}|\mathbf{w}^T\mathbf{x}+b=-1\}$
Margin Hyperplane $\{\mathbf{x}|\mathbf{w}^T\mathbf{x}+b=1\}$
Maximum Margin
Optimal Separating Hyperplane $\{\mathbf{x}|\mathbf{w}^T\mathbf{x}+b=0\}$

Fig. 2.2: Optimal Separation Hyperplane (Solid Line) and two margin Hyperplanes (dashed line) in a binary classification example; Support Vectors are bolded

In linearly separable cases, there exists one unique optimal separating hyperplane that is distinguished by the maximum margin between either class of samples. It can always be expressed in the form of:

$$\left\{\mathrm{x}\,\middle|\,\mathbf{w}^T\mathrm{x} + b = 0\right\} \text{ Subject to max margin: } \mathit{max}\left\{\mathit{min}\left\{\left\|\mathbf{x} - \mathbf{x}_i\right\|\right\}\right\} \tag{9}$$

where **w**, the weight vector, defines the inclination direction of the separation hyperplane, and *b*, the bias, indicates the Euclidian distance from origin to this hyperplane.

Consequently, the decision function to predict the label of a new sample can be written as:

$$\mathrm{f}\left(\mathrm{x}_{\mathrm{j}}\right)=\mathrm{sign}\left(\mathbf{w}^{\mathrm{T}}\mathrm{x}_{\mathrm{j}}+\mathrm{b}\right) \tag{10}$$

where $\mathbf{x}_j$ represents the new input data sample to be classified. The output of the decision function, which is a sign, predicts the label of $\mathbf{x}_j$. We can see that the binary classification process has been reduced to an optimization problem of finding **w** and *b* for the optimal hyperplane separating the classes subject to the maximum margin constraint.

It can be shown that maximizing the margin of separation is equivalent to minimizing the norm of **w**. The capacity of a classifier, which is inversely proportional to its generalization ability, will decrease with the increasing of separation margin. It is one of the best features of SVM based classifiers because being able to control their capacity by acting on the margin maintains generalization ability. Another commonly used form of this constraint optimization problem (primal formulation) is:

$$min\left\{\frac{1}{2}\left\|\mathbf{w}\right\|^{2}\right\} \tag{11}$$

subject to:

$$y_i\left(\mathbf{w}^T\mathrm{x}_i+b\right)\geq 1 \quad for \quad i=1,\cdots,l \tag{12}$$

2.4.2 Linear Separable - Dual Formulation and Lagrangian Solution

A traditional way of solving the above constrained optimization problem is to apply Lagrangian Formalism. As developed in [36, especially Section 5.3] the primal Lagrangian is:

$$L\left(\mathbf{w},b,\boldsymbol{\alpha}\right)=\frac{1}{2}\|\mathbf{w}\|^2-\sum_{i=1}^{l}\alpha_i\left[y_i\left(\mathbf{w}^T\mathrm{x}_i+b\right)-1\right] \tag{13}$$

where α_i are the nonnegative Lagrange multipliers. The corresponding dual is found by differentiating with respect to $\mathbf{w}$ and b to yield:

$$\frac{\partial L(\mathrm{w},b,\boldsymbol{\alpha})}{\partial \mathrm{w}}=\mathrm{w}-\sum_{i=1}^{\ell}y_i\alpha\mathrm{x}_i \tag{14}$$

$$\frac{\partial L(\mathbf{w},b,\boldsymbol{\alpha})}{\partial b}=\sum_{i=1}^{\ell}y_i\alpha_i \tag{15}$$

The complementary conditions from the Karush-Kuhn-Tucker conditions, see for example [67, Appendix A], for 14 and 15 state that the product of the dual variables and the constraints should be zero at the optimal solution.

$$\frac{\partial L}{\partial \mathrm{w}}=0\rightarrow \mathrm{w}=\sum_{i=1}^{\ell}y_i\alpha\mathrm{x}_i \tag{16}$$

$$\frac{\partial L}{\partial b}=0\rightarrow\sum_{i=1}^{\ell}y_i\alpha_i=0 \tag{17}$$

As a result, both **w** and *b* can be denoted as combinations of support vectors and their corresponding Lagrange multipliers:

$$\mathbf{w}=\sum_i\alpha_i\mathrm{y}_i\mathbf{x}_i \qquad b=-\frac{1}{2}\sum_i y_i\alpha_i\left[\mathbf{x}_i^T\mathbf{s}_1+\mathbf{x}_i^T\mathbf{s}_2\right] \qquad i=1,\cdots l_{SV} \tag{18}$$

Physically, they are the input data samples that lie on the margin hyperplane.

The $\mathbf{s}_1$ and $\mathbf{s}_2$ are two arbitrary support vectors from each class respectively. Therefore, the decision function can be improved as:

$$f\left(\mathbf{x}_j\right) = sign\left(\sum_{i=1}^{\#SV} \alpha_i y_i \mathbf{x}_i^T \mathbf{x}_j + b\right) \tag{19}$$

2.4.3 Linear, Non-Separable – Primal Formulation

Compared with the linearly separable case, linear, non-separable classification conveys significantly more practical information and for the research proposed here has significantly more value. In the linear, non-separable case, one can still place a linear classification boundary to classify the patterns, but since both classes are partially overlapped, there will be a number of errors. In order to minimize those errors, the SVM approach takes into account the patterns that are inside the margin or outside the margin but in the incorrect side of the separation hyperplane. The distances ξ of such patterns to the margin hyperplane are then accounted for in the minimization. Thus, the constraint optimization problem associated with nonlinearly separable classification can be described (primal formulation) as following:

$$min\left\{\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{l}\xi_i\right\} \tag{20}$$

subject to:

$$y_i\left(\mathbf{w}^T\mathbf{x}_i + b\right) \geq 1 - \xi_i \quad for \quad \xi_i \geq 0 \quad and \quad i = 1,\cdots,l \tag{21}$$

where ξ_i are the slack variables that measure the deviation of outliers from the margin hyperplane, and C is the penalty parameter which controls the compromise between maximizing the separation margin and minimizing the training errors. The slack variables are forced to be positive or zero, that is, the slack variables for correctly classified samples that are outside the margin are set to zero.

2.4.4 Linear, Non-Separable – Dual Formulation and Lagrangian Solution

After applying Lagrangian formalism to this problem, it yields:

$$L\left(\mathbf{w}, b, \boldsymbol{\alpha}, \boldsymbol{\mu}\right) = \frac{1}{2}\left\|\mathbf{w}\right\|^2 + C\sum_{i=1}^{l}\xi_i - \sum_{i=1}^{l}\alpha_i\left[y_i\left(\mathbf{w}^T\mathbf{x}_i + b\right) + \xi_i - 1\right] - \sum_{i=1}^{l}\mu_i\xi_i \qquad (22)$$

where α_i and μ_i are both nonnegative Lagrange multipliers. By forcing all the derivatives of (22) with respect to **w,** b and ξ_i to be zero, and applying the KKT complementary condition, a Dual Lagrangian function can be derived in matrix form as follows:

$$L_D = 1^T\boldsymbol{\alpha} - \frac{1}{2}\boldsymbol{\alpha}^T\mathbf{YKY}\boldsymbol{\alpha} \qquad (23)$$

where $\boldsymbol{\alpha}$ is the vector of Lagrange multipliers α_i**,** **Y** is a diagonal matrix containing the labels, and **K** is the matrix of dot products between training data. Accordingly, the optimal solution of **w** appears in the following formation:

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i \quad i = 1, \cdots, l_{SV} \qquad (24)$$

Obviously, only the support vectors out of the whole set of the training data are involved in the computation. This will greatly reduce the computational complexity during the test phase in the nonlinear version of the SVM, as we explain below.

2.4.5 Kernel Trick and Mercer Condition

SVMs belong to the class of linear algorithms that can be expressed as a linear combination of dot products between data. SVMs can be non-linearized through the Kernel Trick [40] and [42]. The basic idea is to have a nonlinear transformation $\phi(\bullet)$ of the data $x[n]$ into a higher (possibly infinite) Hilbert space for which the associated dot product is expressible as a function $K(\bullet,\bullet)$ of the input data as:

$$\left\langle \phi(\mathbf{x})_i, \phi(\mathbf{x})_j \right\rangle = K(\mathbf{x}_i, \mathbf{x}_j) \tag{25}$$

Such a function is called a Mercer Kernel. A Hilbert Space provided with a kernel is called a Reproducing Kernel Hilbert Space. Mercer's Theorem states the conditions for a kernel to be a dot product in a Hilbert Space. In particular, it states that a mapping function $\phi(\bullet): \mathbb{R}^n \mapsto \mathcal{H}$ and a function $K(\bullet,\bullet)$, as in (25), exists if and only if $K(\bullet,\bullet)$ is an integral operator in a Hilbert Space, i.e., if the kernel satisfies:

$$\int K(\mathbf{x}_i, \mathbf{x}_j) g(\mathbf{x}_i) g(\mathbf{x}_j) d\mathbf{x}_i d\mathbf{x}_j \geq 0 \tag{26}$$

for any square integrable function $g(\bullet)$.

Equation 25 is often called the "Kernel Trick" which enables the ability to work in huge dimensional feature spaces without actually having to perform explicit computations in this space. In the case of support vector machines we start from a primal formulation with a high dimensional feature space by applying transformations $\phi(\bullet)$. The problem is not solved in the space of the primal formulation but in the dual space of Lagrange multipliers after applying the kernel trick.

2.4.6 Non-Linear, Separable

Now that we have the kernel trick available a non-linear, separable problem can be transformed into a linearly separable one by mapping from the input space into a higher (possibly infinite) dimensional hypothesis Hilbert space (also called feature space), as illustrated by Fig. 2.3.

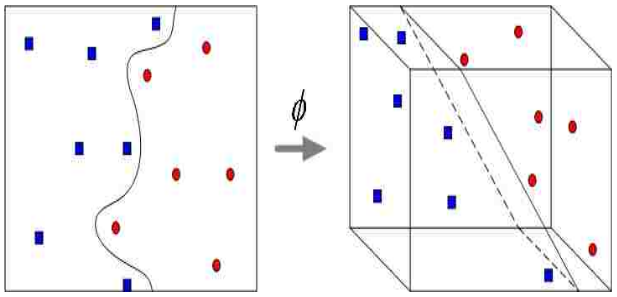


Fig. 2.3: Non-linearly separable problem in 2-D space becomes linearly separable in 3-D space after a nonlinear transformation "$\boldsymbol{\varphi}$" being applied.

A short explanation of what Figure 2.3 is trying to tell us is warranted. If we examine the left hand image we see that there is no conceivable way to place a separating hyperplane to separate the red from the blue samples. However, if we add another dimension, by the use of the transformation ϕ , we can see that the placement of the separating hyperplane is now readily apparent. If we were to view the 3D space (right hand image) from the top we would still see the same pattern that we do in the left hand image.

Using the techniques developed above in Sections 2.4.3 and 2.4.4 we can write the primal as:

$$f\left(\mathbf{x}_j\right) = sign\left(\mathbf{w}^T \varphi\left(\mathbf{x}_j\right) + b\right) \tag{27}$$

As before we use Langrangian Formalism to construct:

$$L\left(\mathbf{w}, b, \boldsymbol{\alpha}, \boldsymbol{\mu}\right) = \frac{1}{2}\left\|\mathbf{w}\right\|^2 + C\sum_{i=1}^{l} \xi_i - \sum_{i=1}^{l} \alpha_i \left[y_i\left(\mathbf{w}^T \mathbf{x}_i + b\right) + \xi_i - 1\right] - \sum_{i=1}^{l} \mu_i \xi_i \tag{28}$$

Applying the KKT conditions:

$$\begin{aligned} &\frac{\partial \mathrm{L}}{\partial \mathrm{w}} = 0 \rightarrow \mathbf{w} = \sum_{k=1}^{\ell} \alpha_k y_k \phi(x_k) \\ &\frac{\partial \mathrm{L}}{\partial \mathrm{b}} = 0 \rightarrow \sum_{k=1}^{\ell} \alpha_k y_k = 0 \\ &\frac{\partial \mathrm{L}}{\partial \xi_k} = 0 \rightarrow 0 \leq \alpha k \leq c, k = 1,..,\ell \end{aligned} \tag{29}$$

yields:

$$L(\alpha) = -\frac{1}{2}\sum_{k,l=1}^{\ell} y_k y_l K(x_k, x_l)\alpha_k \alpha_l + \sum_{k,l=1}^{\ell} \alpha_k \tag{30}$$

This yields a new decision function:

$$f\left(\mathbf{x}_j\right) = sign\left[\sum_{i=1}^{l_{SV}} y_i \alpha_i \varphi\left(\mathbf{x}_j\right)^T \varphi\left(\mathbf{x}_i\right) + b\right] \tag{31}$$

Applying the kernel trick:

$$\varphi\left(\mathbf{x}_i\right)^T \varphi\left(\mathbf{x}_j\right) = K\left(\mathbf{x}_i, \mathbf{x}_j\right) \tag{32}$$

where *K* is a continuous symmetric non-negative definite kernel. With this result, without knowing the explicit form of the non-linear transformation *φ*(.), the decision function can be re-organized into the following form:

$$f\left(\mathbf{x}_j\right) = sign\left[\sum_{i=1}^{\ell} y_i \alpha_i K\left(\mathbf{x}_i, \mathbf{x}_j\right) + b\right] \tag{33}$$

In this way, the learning process in the feature space does not involve anything related with the transformation *φ*(.). The new input data samples will be classified according to the output of decision function with maximized generalization ability.

2.5 SVMs For Regression

A more rigorous treatment of the following material is contained in [32].

To extend this discussion on SVM classification into a regression algorithm we return to Eqn 1:

$$\left\{\mathbf{x} \middle| \mathbf{w}^T \mathbf{x} + b = 0\right\} \text{ Subject to max margin: } max\left\{min\left\{\left\|\mathbf{x} - \mathbf{x}_i\right\|\right\}\right\} \tag{34}$$

In the case of SVM regression (SVR), we measure the error of approximation instead of the margin used in the classification. Vapnik developed a unique loss (error) function, called the “linear loss function with ε-insensitivity” as:

$$E(\mathbf{x}, y, f) = \left|y - f(\mathbf{x}, \mathbf{w})\right|_{\varepsilon} = \left\{\begin{matrix} 0, & \text{if } \left|y - f(\mathbf{x}, \mathbf{w})\right| \le \varepsilon, \\ \left|y - f(\mathbf{x}, \mathbf{w})\right| - \varepsilon, & \text{otherwise} \end{matrix}\right\} \tag{35}$$

Or, the loss is equal to zero if the difference between the estimate and the measured value is less than ε. Graphically, we can see this in Fig. 2.4 by representing ε as a “tube” around the actual function. For all predicted points outside the tube, the loss equals the magnitude of the difference between the estimate and the radius of the tube.

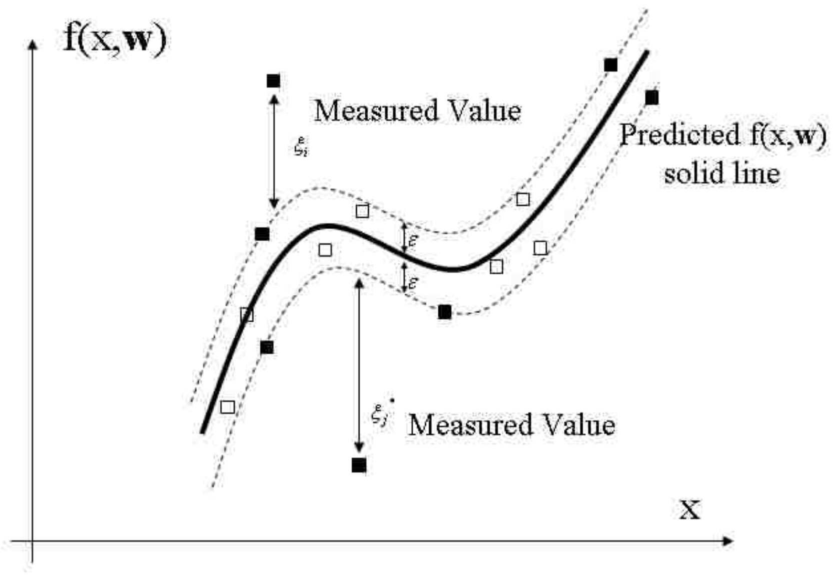


Figure 2.4: Graphical representation of a "loss" function

Formally speaking, the SVR is intended to minimize the functional

$$L = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}(\xi_i + \xi_i^*) \tag{36}$$

subject to the constraints:

$$\begin{aligned} y_i - \mathbf{w}^T \mathrm{x}_i + b &\le \varepsilon + \xi_i \\ -y_i + \mathbf{w}^T \mathrm{x}_i - b &\le \varepsilon + \xi_i^* \end{aligned} \tag{37}$$

where the first constraint equation is used for positive errors and the second one for negative errors. These constraints actually implement the epsilon insensitive cost function. The application of Lagrangian analysis to the above constrained optimization problem leads to a solution which is similar to the one for classification. In particular, the solution for the weights is

$$\mathbf{w} = \sum_{i=1}^{N} \left(\alpha_i - \alpha_i^* \right) \mathrm{x}_i \tag{38}$$

Here, the Lagrange variables are: α_i, α_i^*.

Adding this result to the Lagrangian formulation results in a dual optimization problem that can be expressed in matrix form as

$$-\left(\alpha + \alpha^*\right)^T \mathbf{K}\left(\alpha + \alpha^*\right) + \left(\alpha + \alpha^*\right)^T \mathbf{y} + \left(\alpha + \alpha^*\right)\mathbf{1}\varepsilon \tag{39}$$

where α, α*, **y** are vectors containing the Lagrange multiplier and the regressors y_i respectively, and where matrix **K** contains the dot products $\mathbf{K}_{ij} = \mathbf{x}_i^T \mathbf{x}_j$ between training data.

Equation 39 can be derived by applying a modified cost function that combines both Vapnik's ε-insensitive cost function [40], [42] and Huber cost function [77] which was introduced in [78] for SVMs applied to system identification. See Figure 2.5 for a graphical representation of this cost function.

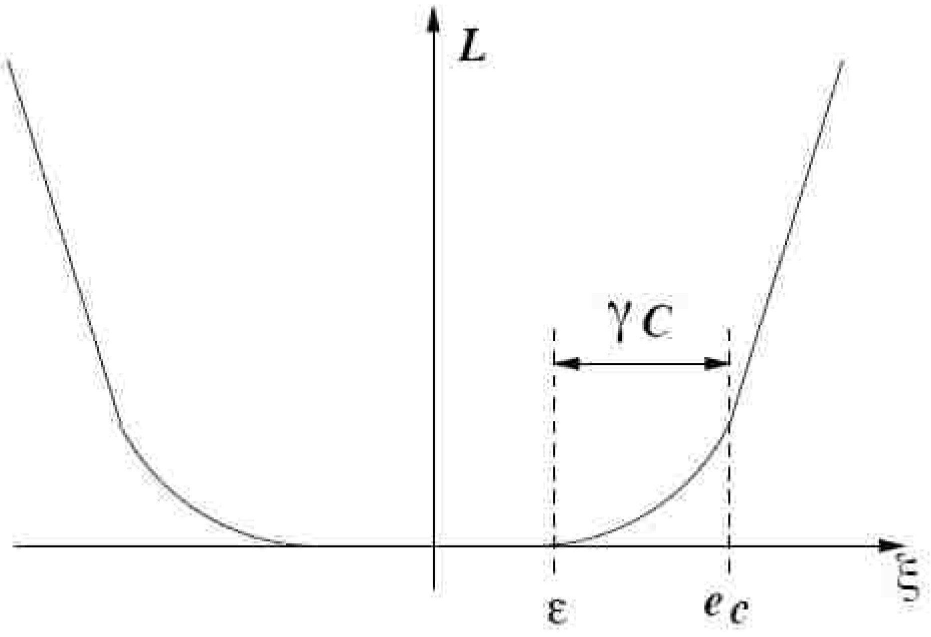


Figure 2.5: Composite Cost Fuction

Mathematically we can represent this cost function as:

$$L = \begin{cases} 0 & |\xi| < \varepsilon \\ \frac{1}{2\gamma C}\left(|\xi| - \varepsilon\right)^2 & \varepsilon \le |\xi| \le e_C \\ \left(|\xi| - \varepsilon\right) - \frac{1}{2}\gamma C & e_C \le |\xi| \end{cases} \tag{40}$$

where $e_C = \varepsilon + \gamma C$. Using this cost function leads to the Lagrange multipliers [79]:

$$\alpha_n = \begin{cases} 0 & |\xi| < \varepsilon \\ \frac{1}{\gamma}\left(\xi[n] - \varepsilon\right) & \varepsilon \le |\xi[n]| \le e_C \\ C & e_C \le |\xi[n]| \end{cases} \tag{41}$$

For $\varepsilon \neq 0$ only a subset of the Lagrange multipliers will be zero [37] and the input samples associated with the non-zero Lagrange mulitpliers are the Support Vectors. The cost function, as defined in Equation 40, provides a functional that can be regularized by the matrix $\gamma\mathbf{I}$. A closer examination of the cost function explains why the SVM algorithm is robust against non-Gaussian noise. The cost function is quadratic for data that produces errors between ε and e_C and linear for errors above e_C. Therefore we can adjust e_C to apply a quadratic cost for samples that are affected by thermal noise. The linear cost is applied to samples that are affected by non-Gaussian noise.

This optimization problem can be solved using quadratic programming [54]. Equation 39 is in quadratic form, thus having a unique solution. Nevertheless, the problem may be ill posed, so the matrix needs to be numerically regularized by a small diagonal. It can be shown that this is equivalent to a modification of the cost function that adds a small quadratic section between the epsilon insensitive section and the linear section. This property can be exploited in terms of adaptation to the noise probability density, and as we show in our results allows SVR algorithms to operate as spectral estimators in a non-Gaussian noise environment.

2.6 Parametric Spectrum Estimation

For this research we will be investigating two types of algorithms for the spectral estimation task. The first algorithm is a linear algorithm that we will call the Non-Parametric Algorithm as we assume no a-priori knowledge of a model of the signals that may be present in the channel.

We will also use a non-linear algorithm that we will call the Parametric Algorithm because we assume some knowledge of what may be present in the channel. Because it is a non-linear algorithm, in accordance with Section 2.4.6, we must develop a kernel, "*K*" in Equation 25, which helps project the measured data into a higher space to develop the estimation hyperplane. Another way to think about the role that kernels play in SVR is to think of the kernels chosen as a similarity measure [37]. When we adopt this view towards what kernels are, and since we are trying estimate spectrum usage, we need to choose kernels that are composed of models of what we are trying to estimate.

This research does not intend to delve into the intricacies of kernel theory. Instead the reader is referred to [36] - [39], and [55] for a more complete mathematical treatment of the topic.

Our assumption of what types of kernels to use has a flaw that could be devastating to our spectral estimation and that is the fact that our kernels have no information about the type of propagation channel that may be present. If the channel is flat than our spectral estimation can neglect the affects of the channel, at best we will have an attenuation factor to contend with, but in most cases where a cognitive radio might be used the assumption of a flat channel is invalid.

The predominant types of channel distortion are Rician and Rayleigh Fading as well as "fast" and "slow" fading [68]. In our case the main problem with this type of fading is the so-called multipath effect where versions of the original signal are scaled and phase shifted by the channel and is received at different times by the receiver. These effects are caused by the fact that a signal may have to travel separate paths to get to the receiver. In addition, a line-of-sight version of the signal, which will be stronger than all the other versions of the signal may (Rician Fading) or may not (Rayleigh Fading) be received as well. Therefore we must somehow include channel effects in the kernels that we use for our spectral estimation.

We have developed an SVM-like methodology for generating the kernels to be used for this research that takes into account the channel affects on the signal.

In the case of the 2.4 GHz ISM band we have knowledge of the types of signals that might be present in the channel but do not necessarily have to be present. Given any set of time samples $\mathrm{x[n]}$ we can write the autocorrelation as:

$$r[\mathrm{n}] = \sum_{j} x[\mathrm{n}]x^{*}[\mathrm{n}-j] = x[\mathrm{n}] * x^{*}[-\mathrm{n}] \tag{42}$$

If these time samples came from the channel in question, we seek to find an approximation to the spectrum of $x[\mathrm{n}]$ using an estimation model that consists of a mixture of the signals that we believe might be present in $x[\mathrm{n}]$. Working with the autocorrelation function in Equation 42 we can write an expression for the estimate of the total autocorrelation function:

$$\hat{r}[\mathrm{n}] = \sum_{k=1}^{K} a_k r_k[\mathrm{n}] \tag{43}$$

where the a_k terms represent the amplitudes, including $a_k = 0$, of each of the autocorrelation terms which correspond to a different signal that may be present in the channel.

As explained before this estimation model has a serious flaw without the inclusion of the propagation channel. For this derivation we will work with the channel impulse response $\mathrm{h[n]}$. Our new estimation model is:

$$\hat{r}[\mathrm{n}] = \mathrm{h[n]} * \sum_{k=1}^{K} a_k r_k[\mathrm{n}] \tag{44}$$

where we assume that the $\mathrm{h[n]}$ is the same for all signals and time invariant. As the channel is unknown this model has two sets of unknowns; the α_k that represent the spectrum and the channel impulse response.

Adding the channel impulse response to the estimate and then not knowing the channel impulse response may seem like an intractable problem. But if we use the same sort of formulation that allows for solutions to the SVM formulation, hence the term “SVM-like algorithm”, we can arrive at a methodology that allows us to calculate what we need, i.e. the “K” for Equation 25 above, without actually knowing the channel response.

Since SVMs are grounded in statistical learning theory [40] we return to statistical learning theory to help us find an interpretation of $\mathrm{h[n]}$ and for that we introduce Hilbert Spaces into our derivations.

We assume a non-linear function of time, ϕ_n, that provides a mapping,

$n : \mathrm{R} \rightarrow \phi[n] : \mathcal{H}$ from our input space to a higher dimension Hilbert Space. According to [36] and [37], we do not need to know what the transformation ϕ is as long as we know the results of the dot product in the Hilbert Space are. We can write this as:

$$K(n,m) = \phi^T[n] \sum \phi[m] \tag{45}$$

where $\sum$ is a positive definite matrix.

To be a kernel this function must fit the conditions of Mercer's Theorem [36], which provides a characterization of when a function $K(n,m)$ is a kernel. In particular this function must be positive definite, see [36, Proposition 3.5, page 33]. A function that fits Mercer's Theorem and is a dot product in a Hilbert Space is known as a Mercer's Kernel. $\mathcal{H}$ is then known as a Reproducing Kernel Hilbert Space [37].

We assume that we have available a set of $\phi_j[n], 1 \leq j \leq N$ for training purposes in this Hilbert Space. As an estimator of their autocorrelations we can write:

$$\hat{r}[n] = \mathbf{w}^T \phi[n] \tag{46}$$

where $\mathbf{w}$ is a parameter vector estimated using the data. We will refer to this estimator as a "primal representation" and $\mathbf{w}$ are its "primal variables".

Since the estimator parameters $\mathbf{w}$ are obtained using a linear algorithm based on data, we can state that these parameters are a linear combination of the parameters because of the representer theorem [37], [56]. Because of this we can write that the relationship between $\mathbf{w}$ and the channel impulse response is:

$$\mathbf{w} = \sum_{j=1}^{N} h[j]\phi[j] \tag{47}$$

Equation 47 leads to another representation of Equation 46, which is:

$$\hat{r}[n] = \sum_{j=1}^{N} h[j]\phi[j]^T \sum \phi[n] \tag{48}$$

Referring to the original SVM formulation presented earlier we can see that Equation 46 and 48 yield a "dual representation" for the autocorrelations. Further, the $h[j]$ are the "dual variables".

Since the nonlinear transformation "ϕ" remains unknown from Equation 48, we need to find another representation. Since the Hilbert space provides us with a kernel dot product, we have access to those dot products. The estimator can be rewritten as:

$$\hat{r}[n] = \sum_{j=1}^{N} h[j]k[j-n] \tag{49}$$

We note that Equation 49 is a convolution between signals $k[n]$ and $h[n]$.

From before, we know that the kernel function is a dot product in a Hilbert space and that the kernel function must be positive definite. As well, autocorrelations have the same positive definite characteristic as kernel functions. Therefore, $k(j-n)$ can be an autocorrelation function or a linear combination of other autocorrelation functions. This means that we can restate

Equation 43 as:

$$k(n)=\sum_{k=1}^{K} a_k r_k[n] \tag{50}$$

The Fourier Transform of Equation 50 can be written as:

$$K(j\omega)=\sum_{k=1}^{K} a_k R_k(j\omega) \tag{51}$$

Note that this expression is the joint spectrum of all possible signals present in the signal but still does not contain information about the channel. However, to continue the derivation we assume that a set of a_k have been chosen for the kernel. Then we can write the Fourier Transform for Equation 49 as:

$$R(j\omega)=H(j\omega)K(j\omega)=H(j\omega)\sum_{k=1}^{K} a_k R_k(j\omega) \tag{52}$$

We can see that $H(j\omega)$ represents the envelope of the signal $x[n]$ and a_k represent the amplitudes of each of the spectra that might be present in the signal.

Equation 50 is a form of a composite kernel called a summation kernel. The interpretation of such a kernel in terms of Hilbert spaces can be expressed if we model each kernel of the summation as a kernel in a Hilbert subspace. For the same purpose, we can write our nonlinear transformation $\phi[n]$ as a concatenation of vectors in different subspaces, or:

$$\phi[n]=\left\{\phi_1^T[n],\ldots,\phi_K{}^T[n]\right\}^T \tag{53}$$

and the dot product between two vectors $\phi[n]$ and $\phi[m]$ as:

$$\begin{aligned}\phi^T[n]\phi[m] &= \phi_1^T[n]\phi_1[m] + \ldots + \phi_K{}^T[n]\phi_K[m] \\ &= k_1(n-m) + \ldots k_K(n-m)\end{aligned} \tag{54}$$

Equation 50 contains weights for each kernel which means that their interpretation into a Hilbert Space has the concatenation of vectors weighted as well, or

$$\phi[n] = \left\{\sqrt{a_1}\phi_1^T[n], \ldots, \sqrt{a_K}\phi_K{}^T[n]\right\}^T \tag{55}$$

Now we have all of the pieces necessary to proceed but we have two sets of unknowns $h[n]$ and a_k. To solve this problem we choose to attempt a parameter optimization routine in which we have several choices. We choose to use the SVM strategy to optimize both sets of parameters, hence the choice of SVM-like algorithm earlier in this discussion. Choosing the SVM strategy allows us to not to have to make any assumptions about the parameters, such as their distribution and what the sources for possible model error might be.

Before proceeding we briefly discuss why we didn't choose an easier method, such as assuming that the parameters have a Gaussian distribution. Choosing a Gaussian distribution would allow a maximum likelihood optimization but the result would not be sparse. Further, a Gaussian distribution on the parameters implies a Gaussian Process in that channel and for the reasons stated earlier we cannot make that assumption when attempting to estimate spectrum usage in the 2.4 GHz ISM band.

To construct an SVM-like optimizer for this estimator, we mix Equation 46 with Equation 55. Recall that we have a linear combination of autocorrelations which are interpreted as a linear combination of dot products in Hilbert spaces. We can split the data into "P" Hilbert Spaces and likewise the estimator parameters can also be split into "P" subsets of parameters that lie inside these subspaces. We can therefore rewrite Equation 46 as:

$$\hat{r}[n] = \sum_{k=1}^{P} \sqrt{a_k}\, \mathrm{w}_k{}^T \phi_k[n] \tag{56}$$

The goal is to minimize an SVM-like functional with respect to the dual variables $h[n]$, for which $\mathbf{w}$ are the primal, and a_k. In Equation 13 we were able to use Lagrangian formalism to solve the optimization problem. Here we are not so lucky, but we can optimize these two functionals iteratively. We will design each functional to optimize one of the two variables and during each iteration we will keep the other variable constant.

Considering the primal formulation of a standard SVM regression machine [36], we can construct an SVM algorithm to optimize the primal variables $\mathbf{w}$. The corresponding dual is:

$$-\frac{1}{2} h^t K h + h^t r - h^t 1\varepsilon \tag{57}$$

subject to $-C \leq h[n] \leq C, C$ being one of the SVM free parameters. The other free parameter, ε, forces sparseness in the standard SVM algorithm. Since we are optimizing the primal variables $\mathbf{w}$ and they are concerned with the channel impulse response and we are making the assumption that the channel is linear, we can set ε to zero in Equation 57.

For the parameters a_k we seek a transformation that allows them to be cast as the dual variables of some primal quantity. Let us rewrite Equation 44 as:

$$\hat{r}[n] = \sum_{k=1}^{K} a_k r_k[n] h = \Theta^t a \tag{58}$$

where Θ^t is an N by k matrix where column k contains the vector $r_k h$. We desire parameters a_k be dual variables of an optimization problem. Variables a_k are a linear combination of vectors of matrix Θ. If we write:

$$a = \Theta b \tag{59}$$

we want b to be primal variables. This can be rewritten as:

$$b = \Theta\left(\Theta^t \Theta\right)^{-1} \Theta^t a = Za \tag{60}$$

If $Z = \Theta\left(\Theta^t \Theta\right)^{-1} \Theta^t$ is a new set of data, then variables a are the dual variables of an estimator that can be written as:

$$p = Z^t b \tag{61}$$

Combining Equations 59 and 60 yields:

$$p = Z^t Z a = \Theta\left(\Theta^t \Theta\right)^{-1} \left(\Theta^t \Theta\right)^{-1} \Theta^t a \tag{62}$$

and using Equation 57 we get:

$$p = \Theta\left(\Theta^t \Theta\right)^{-1} \left(\Theta^t \Theta\right)^{-1} r \tag{63}$$

Taking the same set of steps we can arrive at an SVM functional for the variables a_k as:

$$-\frac{1}{2}a^t\Theta\left(\Theta^t\Theta\right)^{-1}\left(\Theta^t\Theta\right)^{-1}\Theta^t a + a^t\Theta\left(\Theta^t\Theta\right)^{-1}\left(\Theta^t\Theta\right)^{-1}r - a^t 1\varepsilon \qquad (64)$$

Table 1 presents a summary of the algorithm just developed.

TABLE 2.2
Summary of SVM-like Algorithm

Step 1: Compute the autocorrelation of the known signal as: $r[n] = x[n] * x[-n]$

Step 2: Construct and optimize the dual functional (Equation 57 with $\varepsilon = 0$) as

$-\frac{1}{2}h^t K h + h^t r$ with respect to h

Step 3: Construct the matrix $\Theta^t = [r_1 h, ..., r_k h]$, with r_k defined in Equation 44.

Step 4: Construct and optimize the dual functional (Equation 64) as:

$$-\frac{1}{2}a^t\Theta\left(\Theta^t\Theta\right)^{-1}\left(\Theta^t\Theta\right)^{-1}\Theta^t a + a^t\Theta\left(\Theta^t\Theta\right)^{-1}\left(\Theta^t\Theta\right)^{-1}r - a^t 1\varepsilon$$

Step 5: Repeat Steps 2-4 until convergence

We realize that this formulation assumes two unknowns: 1.) The signals that may be used in the channel, as represented by a_k in the derivation above, and the channel impulse response autocorrelation, as represented by h in the derivation above. For this research we choose to limit the channel response to a flat channel and propose follow on research in Chapter 6 that takes advantage of being able to extract information about the channel with this derivation.

2.7 Non-Parametric Spectrum Estimation

Finally, by way of an introduction, we discuss how we can mathematically start with the DFT as a spectrum estimation tool and end with the Non-Parametric Algorithm

The DFT solves a parametric model of the signal with the form

$$y[n] = \sum_{i=0}^{K\text{-}1} a_i \cos i\omega_0 + b_i \sin i\omega_0 + e[n] \tag{65}$$

by minimizing the expectation of $e^2[n]$. In order to ease the analysis, let **c** be a column vector containing all parameters a_i and b_i, and $e(\omega_0)$ be a vector containing all functions $\cos i\omega_0$ and $\sin i\omega_0$ concatenated. Then the squared error $e^2[n]$ can be expressed as

$$e^2[n] = \left(y[n] - \mathbf{c}^T\mathbf{e}(\omega_0)\right)^2 = y^2[n] + \mathbf{c}^T\mathbf{e}(\omega_0)\mathbf{e}^T(\omega_0)\mathbf{c} - 2y[n]\mathbf{c}^T\mathbf{e}(\omega_0) \tag{66}$$

The error must be minimized with respect to **c** and then averaged along all data. Thus, one must compute the gradient of the squared error and set it equal to zero in order to find the optimal values for **c**.

$$\Delta_{\mathbf{c}} e^2[n] = \frac{\delta}{\delta\mathbf{c}}\left(y[n] - \mathbf{c}^T\mathbf{e}(\omega_0)\right)^2 = \mathbf{e}(\omega_0)\mathbf{e}^T(\omega_0)\mathbf{c} - y[n]\mathbf{e}(\omega_0) = 0 \tag{67}$$

which, after applying expectations, results in

$$E\left(\mathbf{e}(\omega_0)\mathbf{e}^T(\omega_0)\right)\mathbf{c} = E\left(y[n]\mathbf{e}(\omega_0)\right) \tag{68}$$

In order to get a solution, one must obtain the optimal value of ω_0, which is ω_0=2 /N. For this value, sinusoid functions are eigenvectors of the data. Then, equation above results in

$$\mathbf{c} = E\left(y[n]\mathbf{e}(\omega_0)\right) \tag{69}$$

since the matrix at the left side is the identity matrix. Substituting the expectation by a summation operator gives the DFT formulation. We can recognize that the loss function for the DFT is $e[n]$. The SVM approach uses the same model but instead of applying the MMSE criterion, we apply the $"\varepsilon"$ loss function identified in Equation 35 or depicted graphically in Figure 2.4. Simply, we identify

$$\begin{aligned} &\mathbf{w} = \mathbf{c} \\ &\mathbf{x}[n] = \left(1, \cos\omega n, \ldots, \cos(k-1)\omega n, \sin\omega n, \ldots, \sin(k-1)\omega n\right)^T \end{aligned} \tag{70}$$

Observe that here we do not necessarily use the same value of ω_0 as in the DFT. Here we consider this frequency as a free parameter.

The end result of this manipulation, i.e. substituting the SVM cost function for the for the MMSE criterion, yields a formulation that is more robust against non-Gaussian noise, at the expense of being sub-optimal in the presence of Gaussian noise. The introduction of the “C” parameter helps control the complexity of the solution set and limits the possibilities of overfitting against outliers that is seen in Neural Networks and Genetic Algorithms. As we will see in Chapter 3; the performance of the SVM regression algorithms is very robust against non-Gaussian noise, as predicted by the use of the different cost function, but not as good in Gaussian noise.

3 MODELING AND SIMULATION

3.1 Preliminaries

This research concentrated on performing this spectrum sensing in the 2.4 GHz Instrumentation, Science and Medical (ISM) band. We chose the 2.4 GHz ISM band because of the proliferation of different types of wireless services. These different types make the use of the 2.4 GHz ISM band quite interesting [23] - [26], especially when we consider that each type of service presents a noise source to the other services that might be trying to operate in this band. In fact, the interference between Bluetooth transmitters and the WiFi services got so bad that the Institute for Electrical and Electronic Engineers had to issue standard 802.15.2 to attempt to address the interference concerns between the two services [57]. Table 3.1 presents a brief synopsis of the various signaling and channelization schemes that different wireless services in this band use. Examining the table we see that there are four types of channelization schemes: 1.) Frequency Hopped Spread Spectrum (FHSS), 2.) Direct Sequence Spread Spectrum (DSSS), 3.) Offset Frequency Division Multiplexing (OFDM), and 4.) Fixed Allocations. The specifications for each of these services specify not only how the service accesses the channel but also how much power, thereby defining range, each service can transmit.

We can see that if we were to try all of the various combinations of services, with all of the variability in modulation and channelization that each service can have this research could go on for years just trying all the permutations. Instead, we chose to limit our research to the areas of Bluetooth and WiFi signals in the channel because these two services are the most popular at the current time as evidenced by what services appear most often in the various spectrum studies [4] – [7] and these two services seem to have the most problems coexisting [58], [59], and [60].

TABLE 3.1: Wireless Characteristics in the 2.4 GHz ISM band

Service	Channelization	Bandwidth	Modulation	Data Rate
WiFi 802.11 b	FHSS 2.5/second	22 MHz	2GMSK 4GMSK	1 Mbps 2 Mbps
	DSSS	22 MHz	DBPSK DQPSK 8 Chip CCK 8 Chip CCK	1 Mbps 2 Mbps 5.5 Mbps 11 Mbps
WiFi 801.11g	OFDM 2.5/second	5,10 or 20 MHz	Up to @ 64-QAM	Up to 54 Mbps
Bluetooth 802.15.1	FHSS 1600/second	1 MHz	GMSK	1 Mbps
HR WPAN 802.15.3	5 Fixed Channels 3 Main 2 Coexistent	11 MHz	QPSK@8 State TCM DQPSK@8 State TCM 16-QAM@8 State TCM 32-QAM@8 State TCM 64-QAM@8 State TCM	11 Mbps 22 Mbps 33 Mbps 44 Mbps 55 Mbps
LR WPAN	DSSS 16 channels	22 MHz	O-QPSK	250 Kbps

Since we are limiting the research we have chosen to implement the following simulations:

1.) Single Bluetooth Transmitter, emulating a Bluetooth device searching for a Piconet to join,
2.) Single Wi-Fi transmitter,
3.) Full Bluetooth Piconet comprised of 8 transmitters – 1 master in an even channel and 7 slaves in odd channels, and
4.) Combinations of Bluetooth and WiFi transmitters

We had two goals in mind for the simulations, 1.) Generate time samples that could be exported to the MATLAB workspace for processing with the SVM algorithms that we have developed for this research, and 2.) Generate these time samples in such a way that they can be directly loaded into an arbitrary waveform generator for transmission over an airlink, in various settings, for evaluation of the algorithms in simulated and real noise situations.

To facilitate our ability to create the needed simulations we chose to use the MATLAB/Simulink software suite with the following additional toolboxes/blocksets, Communications Blockset, Signal Processing Blockset, Filter Design Toolbox, and the Communications Toolbox.[1]

In the Simulink models we do not add noise to corrupt the signal. The noise models that were available through Simulink were judged to be inadequate when considering the types of noise that could corrupt a signal in the 2.4 GHz ISM band. In the MATLAB code we have code that can implement three different types of noise: 1.) Average White Gaussian Noise (AWGN), 2.) Impulsive Noise,

[1] More information can be found at www.mathworks.com

and 3.) Block Noise. Figures 3.1 (AWGN), 3.2 (IMPULSIVE), and 3.3(BLOCK) give representative examples of what these noise sources look like before being added to the signal that has been exported from Simulink.

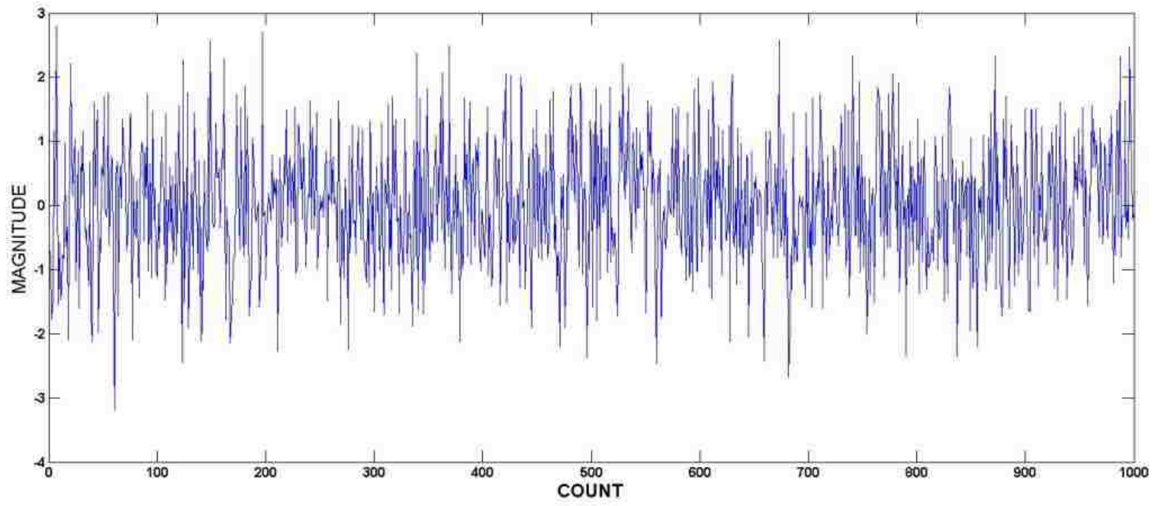


Figure 3.1: Average White Gaussian Noise (AWGN)

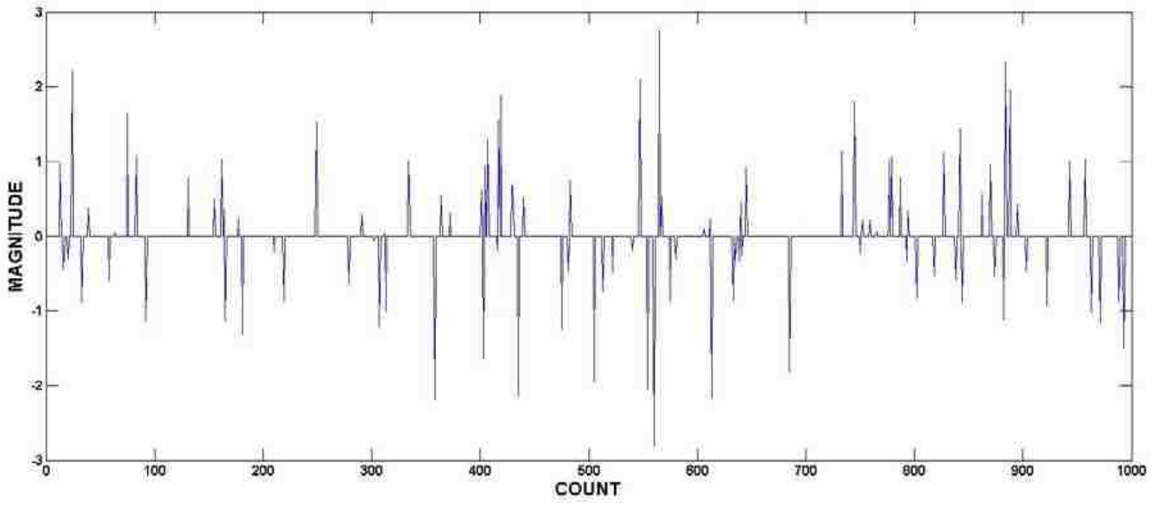


Figure 3.2: Impulsive Noise

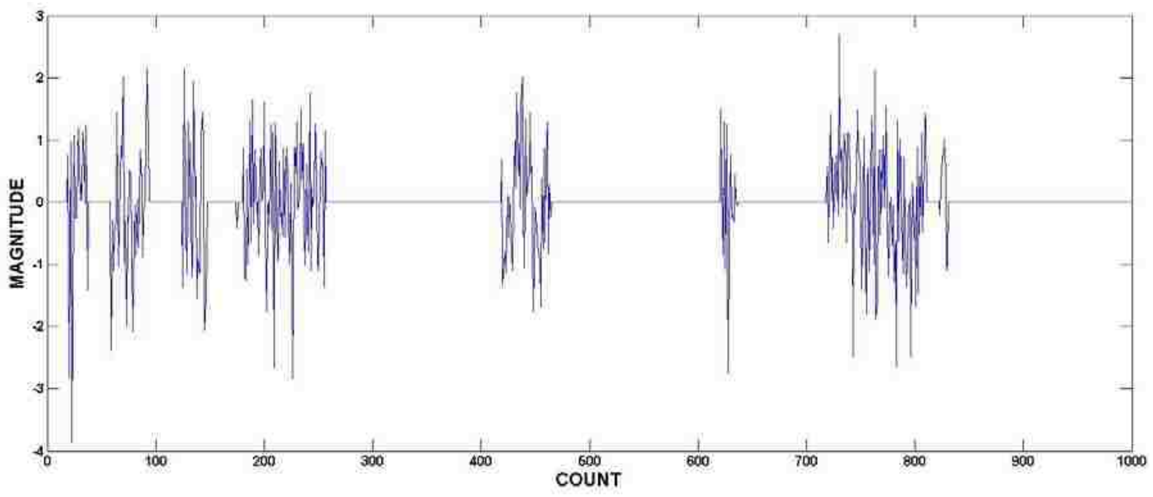


Figure 3.3: Block Noise

3.2 Spectrum Estimation Algorithms

We when started this research we developed a processing flowchart that outlined a step by step process by which a CR could decide if, and when, the radio could transmit. This flowchart was based on characteristics in the Bluetooth and Wi-Fi standards, namely channelization characteristics and modulation usage, which needed to be identified before a CR could decide if a channel was available for transmission. See Figure 3.4 for this flowchart.

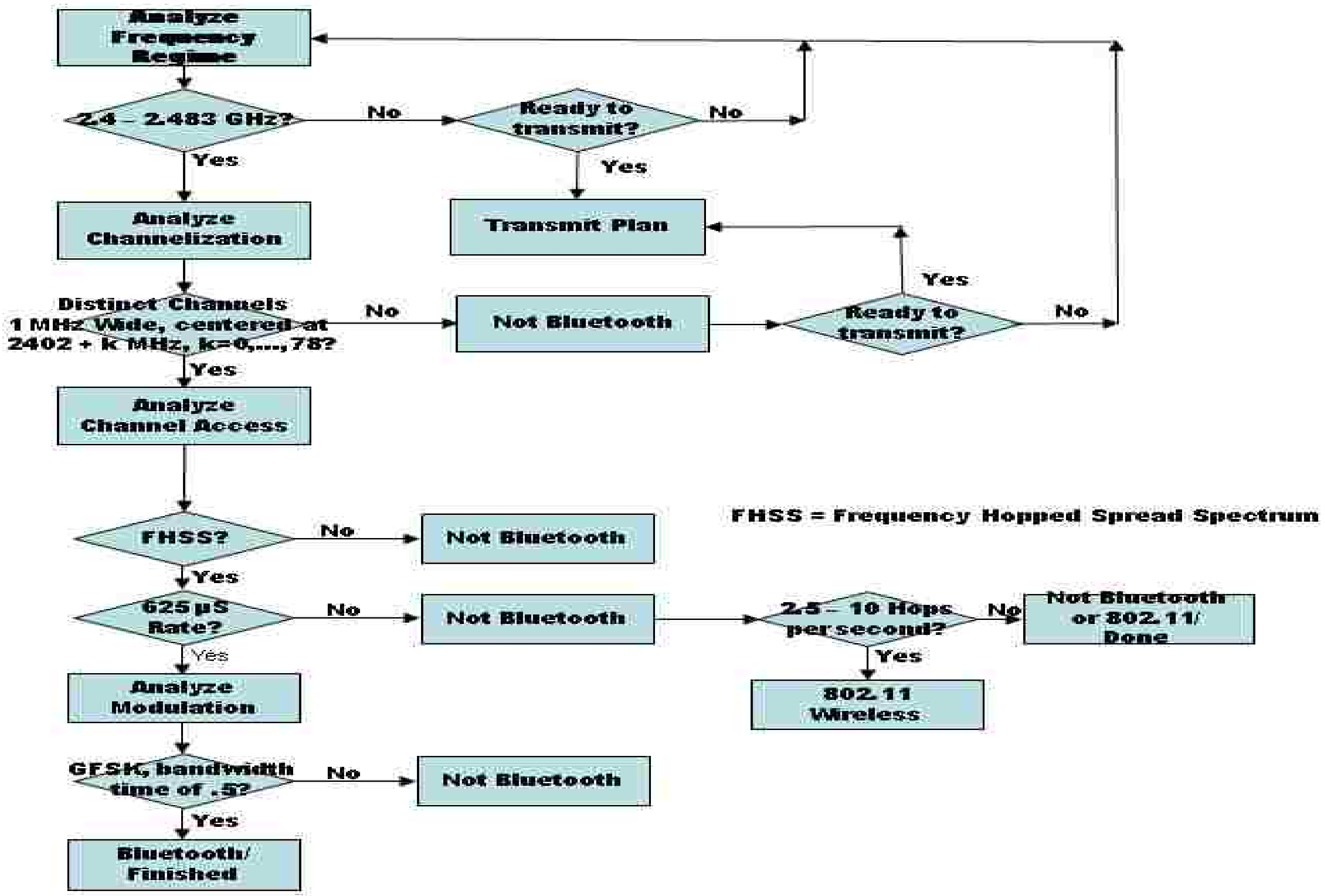


Figure 3.4: Proposed Analysis Flowchart

Therefore, we started developing the spectrum analysis algorithm based on this flowchart. We can see from the steps that the analysis first seeks to analyze channelization and then to analyze the modulation that is used. With these two pieces of information we believed that we could adequately decide what services were in use at any given time.

However, during development of the theory for Section 2.6 we came to realize that, since we need a-priori knowledge about the signals that may be present in the band, we could combine steps in the flowchart.

We had already developed and tested the initial algorithm that would allow analysis of the channelization, hereafter called the “Non-Parametric” algorithm, and will continue to use Non-Parametric algorithm in areas where we have no a-priori knowledge of the signals that may be present. Appendix A contains the basic MATLAB code for the Non-Parametric algorithm.[2]

The theory developed in Section 2.6 above has led to a different algorithm, hereafter called the “Parametric” algorithm. See Appendix B for the basic MATLAB code for the Parametric algorithm.

All of the simulations described below were analyzed with both versions to demonstrate the different abilities of each algorithm.

[2] These algorithms are incomplete. Please contact the author for further information to complete these algorithms

3.3 Single Bluetooth Transmitter

3.3.1 Introduction

For the rest of these discussions the terms “Bluetooth” or IEEE 802.15.1 refer to the same service.

A thorough explanation of the characteristics that affect channel usage by the Bluetooth service is in order before discussing our simulations and subsequent results. See [24, especially section 7 “PHY Layer”] for more information. Bluetooth is designed as a relatively low data rate (when compared with subsequent services) Wireless Personal Area Network and as such the channelization and transmit powers were adjusted accordingly.

The Bluetooth specification calls for 79 channels in the 2.4 GHz ISM band according to the following channelization formula:

$$2402 + k\,MHz, k = 0,...,78 \tag{71}$$

Transmitted symbol rate is 1 Ms/s for date rates of 1 Mbps with a 2 Gaussian Minimum Shift Keying (2GMSK) Modulation or 2 Mbps with 4GMSK. Transmitted powers range from a minimum of 0 dBm Effective Isotropic Radiated Power (EIRP) to a maximum of 20 dBm EIRP.

Figure 3.5 shows the Simulink model that was used for these simulations.[3]

[3] For full model details contact the author at:tatwood@ece.unm.edu

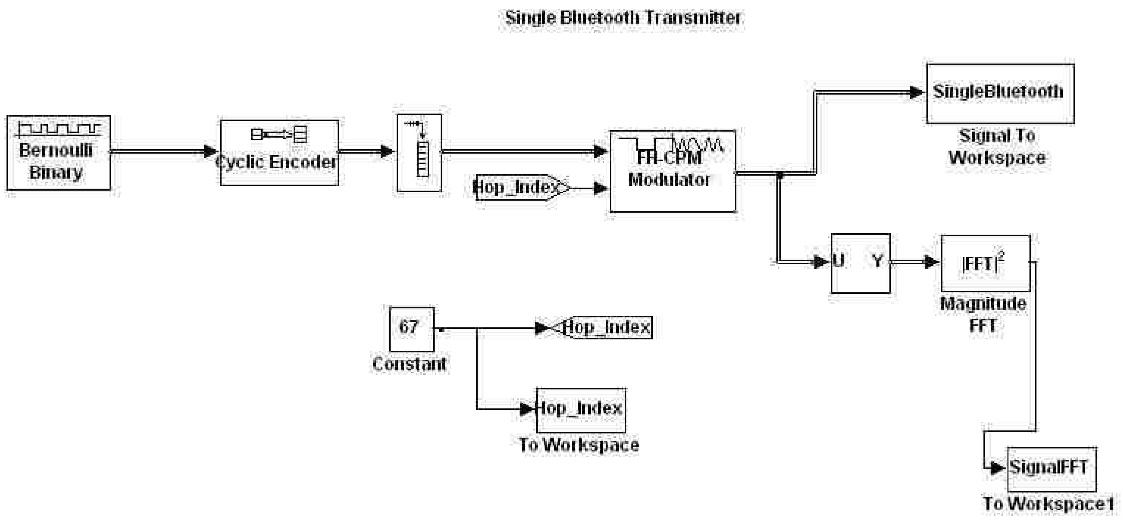


Figure 3.5: Single Bluetooth Transmitter Simulink Model

Since Bluetooth uses a Frequency Hopping Spread Spectrum approach for channelization at a rate of 1600 hops per second, and to get an aggregate data rate of 1 Mbps, the simulation must send 625 symbols every hop. Each symbol is sampled 100 times; therefore, the data that gets exported to the MATLAB workspace equals 62500 samples for each hop.

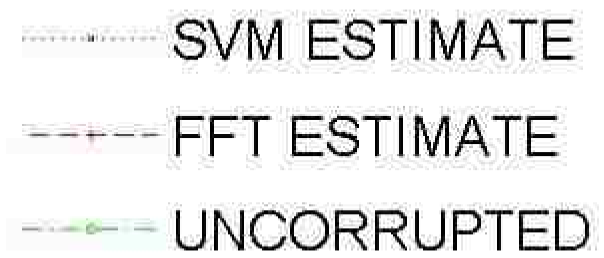
3.3.2 Results

The first type of signal that we examine is the simplest type, i.e., a single Bluetooth transmitter. We would see this type of transmission when a Bluetooth device is attempting to establish a Bluetooth piconet. When a Bluetooth device is first powered on it enters a listening mode to establish whether or not a piconet is present. If a piconet is present, with less than 7 slave devices on the piconet, then the device that has just been powered on will join that piconet. If a piconet is not present; than the device will seek to establish a piconet and will switch to acting as the master.

Without looking inside the data that is being transmitted the only way to tell if a Bluetooth device is the master or a slave is to examine in what slot the device is transmitting. The IEEE specification requires that the master device transmit in an even channel, i.e., $k = 0, 2, ...78$ in Equation 53. All slave devices are to transmit in odd channel, i.e. $k = 1, 3, ..., 77$ in Equation 59.

We start with a simulation that has no impulsive or block noise added to the signal.[4] We add AWGN with a sigma of 1 to represent a channel with no interferers present. Figure 3.6 shows the spectrum estimation for this type of channel. As stated earlier, in this type of noise environment there is no advantage to using an SVM algorithm for spectrum estimation for this type of channel. We can see that each technique, SVM or FFT, give an equally good estimate for the signal when compared to the uncorrupted signal.

In all of the following plots the following colors, line styles and line markers are used:



[4] See Appendix C for a discussion on how noise is added to the signals.

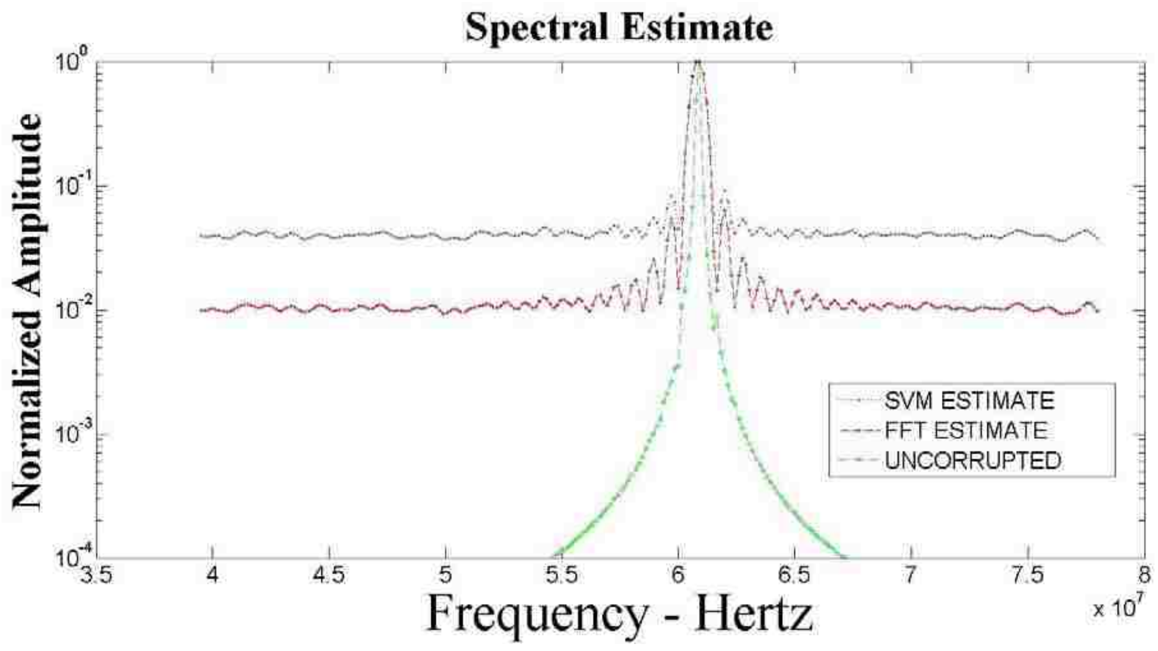


Figure 3.6: Single Bluetooth Transmitter – AWGN Only

To compare the effectiveness of the SVM estimate vs the FFT estimate in the presence of this type of noise the following tables present some statistics that were developed to quantify the performance of the SVM estimate vs the FFT estimate over a repetition of 100 experiments. Each experiment chooses a different noise representation which is a more realistic representation of what we might encounter when performing these spectral estimates on real world data. The numbers are obtained by dividing the amplitude of the highest peak by the amplitude of the next highest peak for each experiment. The mean of the 100 results is than converted to decibels. This gives a measure of the amplitude ratio between the first and second peak as a way of demonstrating the capability of the SVM algorithm versus the FFT algorithm. After each run of 100 repetitions the level of the AWGN was increased in steps of 1 until a maximum of sigma = 5 was reached.

Table 3.2: SVM Vs FFT Amplitude Ratios, AWGN Only

SIGMA	1	2	3	4	5
SVM	10.6	6.3	3.4	1.7	.8
FFT	12.4	10.6	8.8	7.0	5.5

Consistent with our expectations of the efficiency of the SVM estimate we see that at all levels of “Sigma” the SVM technique is sub-optimal when compared to the FFT technique. Of particular interest is how the amplitude differences (for the SVM estimates) tends towards 0 dB (which means there is no amplitude difference) at the higher levels of “Sigma”.

We also compared the SVM Estimate vs the FFT Estimate for the following noise combinations as well: 1.) AWGN + Impulse, 2.) AWGN + Block, and 3.) AWGN + Impulse + Block. Table 3.3 presents the results for these combinations.

TABLE 3.3 :

SVM vs FFT Amplitude Ratios,

AWGN + IMPULSE, AWGN + BLOCK and AWGN + IMPULSE + BLOCK

	SIGMA	1	2	3	4	5
NOISE TYPE						
A,I[5]	SVM	9.8	5.5	2.8	1.3	.6
	FFT	8.2	3.8	1.7	.6	.2
A,B[6]	SVM	7.9	3.8	1.7	.6	.4
	FFT	4.2	.7	.2	.03	.02
A,I,B[7]	SVM	7.1	3.2	1.3	.6	.5
	FFT	3.6	.6	.1	.04	.04

The following estimates are presented to show how the level of sigma increases the noise floor and lessens the amplitude difference between the peak and the noise floor. This illustrates graphically what has been presented in Table 3.3. We can see how, in the presence of non-Gaussian noise, the SVM technique does much better than the FFT technique.

[5] AWGN + IMPULSE

[6] AWGN + BLOCK

[7] AWGN + IMPULSE + BLOCK

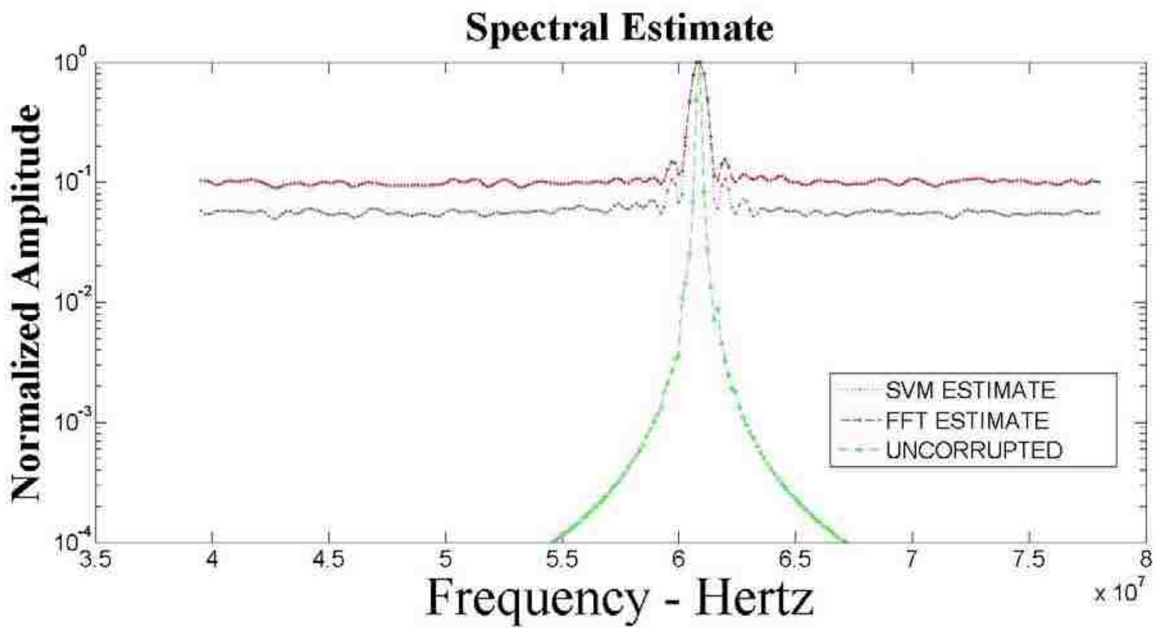


Figure 3.7: Spectral Estimate, AWGN + Impulse, Sigma = 1

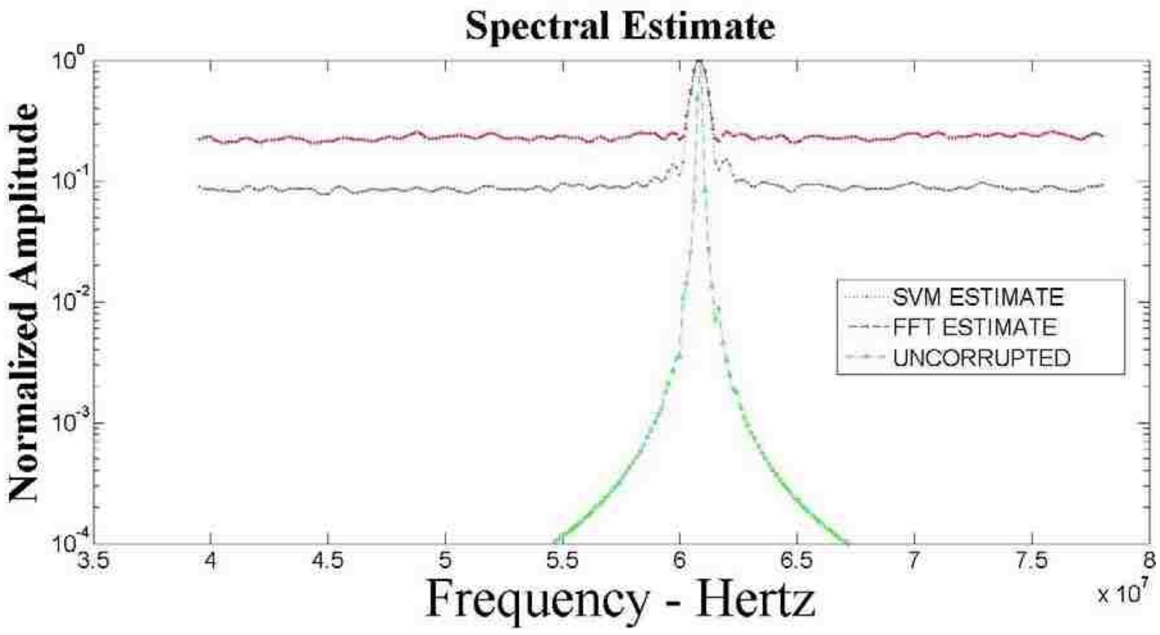


Figure 3.8: Spectral Estimate, AWGN + Block, Sigma = 1

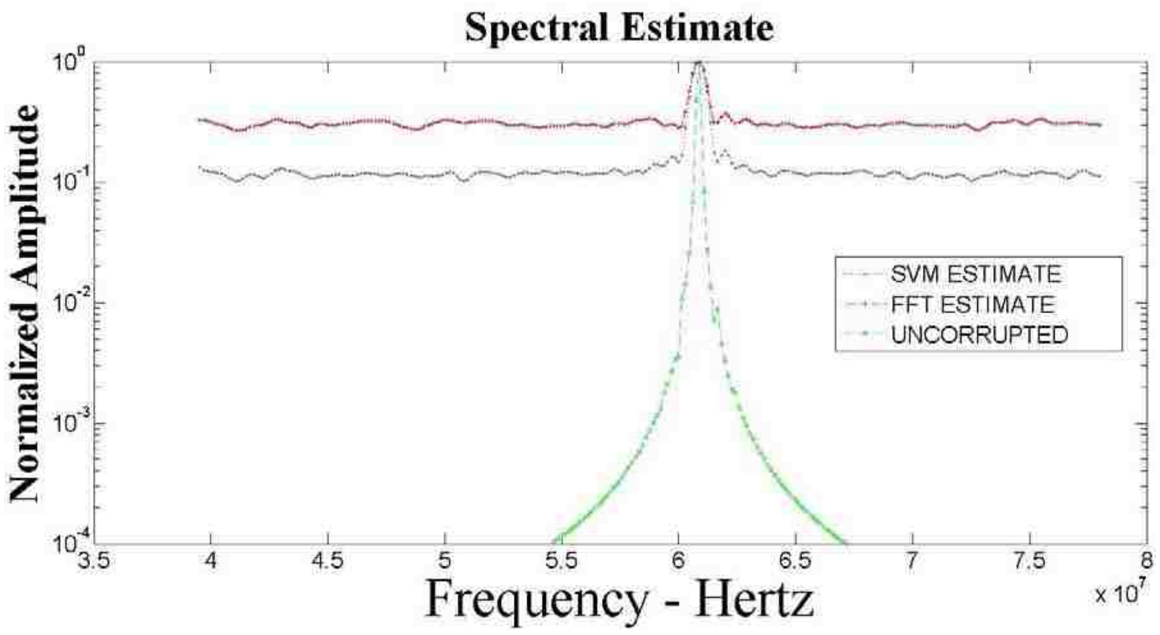


Figure 3.9: Spectral Estimate, AWGN + Impulse + Block, Sigma = 1

Finally, as presented in [66], we investigated the frequency stability of the estimation techniques as a function of the number of symbols that were available for input to the estimation techniques. Table 3.4 presents the mean frequency error +/- the standard deviation versus the number of symbols.

TABLE 3.4: MEAN FREQUENCY ERROR +/- STANDARD DEVIATION

Symbols	1	2	3	4
SVM	-0.51 ± 3.08	-0.25 ± 0.12	-0.03 ± 0.12	-0.02 ± 0.07
FFT	9.37 ± 17.57	3.01 ± 13.70	3.18 ± 12.58	1.14 ± 8.94

These results are significant when we remember that the channelization for

Bluetooth usage is divided into 1 MHz slots. If we are unable to resolve at the sub-MHz level then we are unable to positively declare that a particular channel is being used. As we see from Table 3.4 the FFT technique cannot distinguish channel usage; even after using multiple symbols to refine the estimate. On the other hand the SVM estimate is able to resolve channel usage after just one symbol. This is an important result because of the need to characterize channel usage fast enough to be able to allow the CR to transmit before the channel usage changes due to the Bluetooth FHSS setup.

The estimates we get using the Parametric algorithm are presented in the following figures. The data that we can extract from the SVM-like algorithms does not allow for the same type of statistics that we were able to gather as we did earlier in this section. Instead we present the results of estimates from Channel 42 at each of the noise combinations, i.e. AWGN, AWGN + Impulse, AWGN + Block, and AWGN + Impulse + Block, with sigma =1. Appendix G also presents further estimates with Channel 47 with sigma = 5 for each of these noise combinations. We also show, in Appendix G, the estimates from channel 75. At first glance these estimates do not seem to be as good (as measured by difference from baseline to peak) until we remember that these estimates use kernels that have information about the modulation encoded inside them. The shoulders of the signal estimate have the broadening that one would expect from a modulated signal. Compare the estimate in Figure 3.10 to the estimate in Figure 3.6. The estimate in Figure 3.6 was generated using a kernel that was composed of sines and cosines and these are relatively narrow. The estimate in Figure 3.14 was generated with a kernel that has knowledge of the types of signals that could possibly be in the channel. Since SVR estimates, especially the kernels that are used, can be likened to a similarity measure [37], and since the kernel was constructed with the modulation information as input, we would expect the estimate to reflect information about the modulation. For the purposes of this research; to be able to extract an estimate of the spectrum occupied as well as information about the modulation being used in one step is preferable to the two step process that was envisioned in Figure 3.4.

As detailed in [76] these features can be used to classify the modulation being used to characterize the type of wireless service that is being used in a particular frequency band.

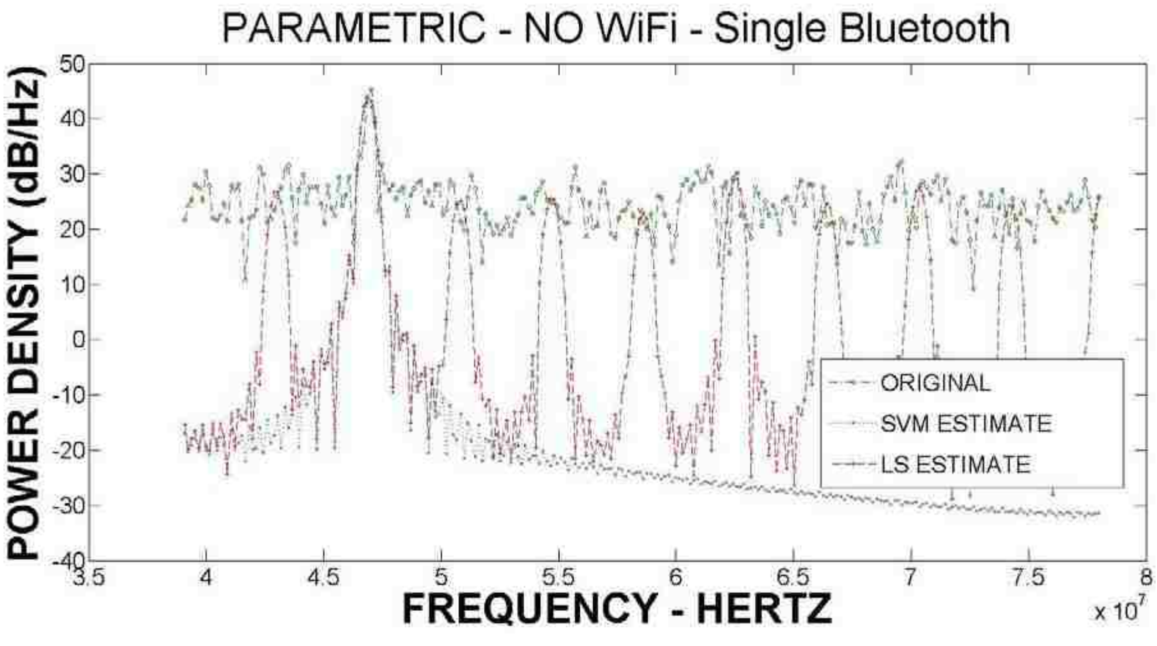


Figure 3.10: Parametric Estimator – Channel 42 – AWGN, Sigma = 1

3.4 Single WiFi Transmitter

3.4.1 Introduction

Current WiFi services available for general usage by the public in the 2.43 GHz ISM band are known as 802.11b and g. The services are described in [23]. WiFi services use a combination of Direct Sequence Spread Spectrum (DSSS) channel access and Orthogonal Frequency Division Multiplex channel access schemes to help reduce interference between 802.11 and Bluetooth (802.15), which uses a Frequency Hopped Spread Spectrum (FHSS) channel access scheme.

In the 802.11b specification there are provisions for 1, 2, 5.5 and 11 Mbps raw data rates and up to 54 Mbps in 802.11g. 802.11g uses Orthogonal Frequency Division Multiplex (OFDM) for its channel access as opposed to the DSSS of 802.11b. See Table 3.1 for a view of the various WiFi services in the 2.4 GHz ISM band.

Since the spectrum usage of an OFDM system is similar to what one would see from any other M-ary Frequency Shift Key (FSK) system [68], i.e. Bluetooth, we chose not to include 802.11g in our investigations as we felt that the investigations done in Section 3.3 showed the ability of the SVM technique for M_ary type signals. For the 802.11b services all 4 available data rates have the same channel bandwidth, e.g. 22 MHz, the difference being the modulation and the chipping codes used to spread the data over the bandwidth.

The 1 Mbps service uses Differential Binary Phase Shift Keying (DBPSK) and the 2 Mbps service uses Differential Quadrature Phase Shift Keying (DQPSK) for data modulation. The 1 and 2 Mbps services use an 11 chip Barker code to spread the output over the 22 MHz band. The 5.5 Mbps and 11Mbps services use the same modulation schemes but use an 8 chip Complementary Code Keying (CCK) to spread the energy over the 22 MHz band. We will investigate whether or not different modulation schemes affect our spectral estimates using the developed SVM techniques.

3.4.2 Results

We use a Simulink model to generate the time history that is then fed to the estimation algorithms. Figure 3.11 presents the Simulink model that we developed for these simulations. The model for the 2 Mbps transmitter is essentially the same except we replaced the DBPSK modulator by the DQPSK modulator.

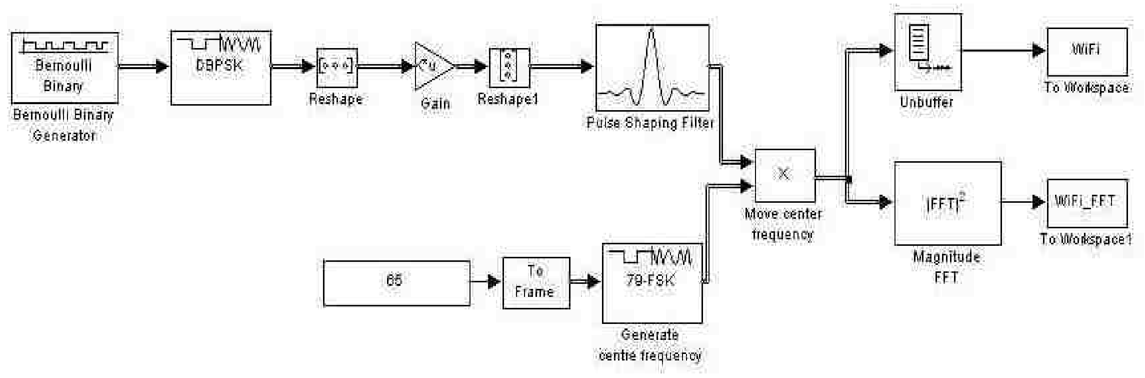


Figure 3.11: 1 Mbps WiFi Transmitter Simulink Model

Figure 3.12 shows the results of the SVM Estimate vs the FFT estimate. An FFT estimate of the uncorrupted signal from the Simulink model is included as well. The transmit signal is centered at 2.463 GHz or Channel 11 as defined by the WiFi specifications. We add AWGN with a sigma of 1 to represent a channel with no interferers.

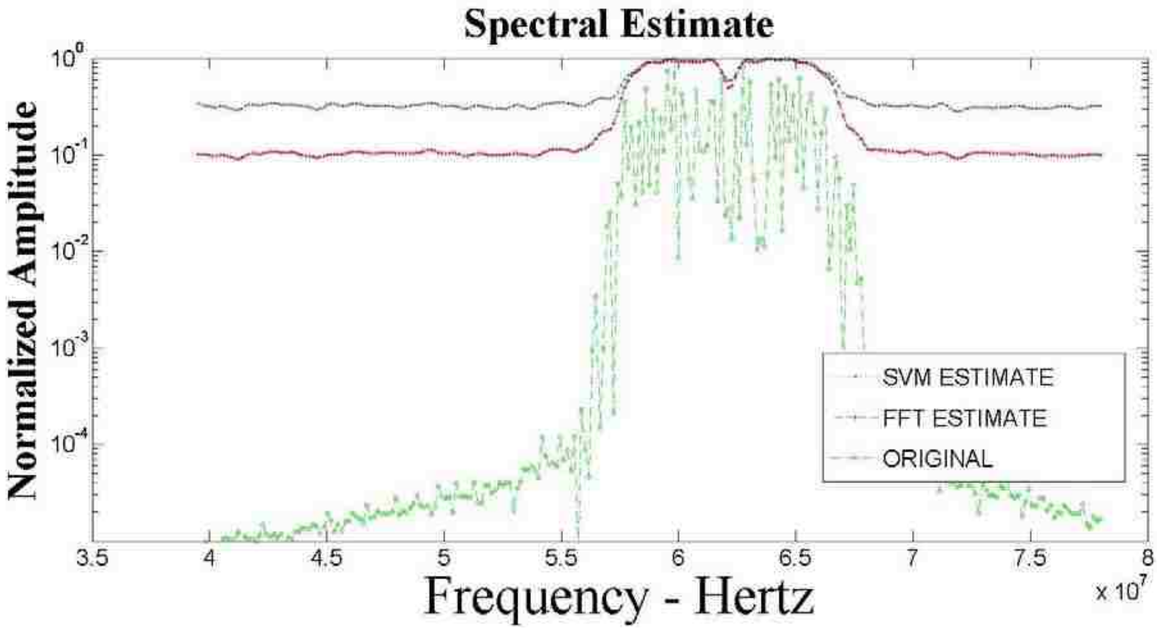


FIGURE 3.12: Single WiFi Transmitter – SVM vs FFT – 1 Mbps, Sigma = 1

As stated earlier, in this type of noise environment there is no advantage to using an SVM algorithm for spectrum estimation for this type of channel. In fact, we can see that the FFT estimate yields superior performance in this type of channel as measured by the difference between the baseline and the peak.

To compare the effectiveness of the SVM estimate vs the FFT estimate in the presence of AWGN noise we conducted 100 experiments that used a random number generator to produce an AWGN sample to be added to the uncorrupted signal obtained from Simulink. Table 3.5 presents these results. In the same general manner as we did for the single Bluetooth experiments divide the peak of the estimate by the average of several samples in the noise floor for each experiment. We take the mean of those 100 experiments and convert that number to decibels and document in Table 3.5. After each run of 100 repetitions the level of the AWGN was increased in steps of 1 until a maximum of sigma = 5 was reached.

TABLE 3.5: SVM vs FFT Amplitude Ratios, AWGN Only

SIGMA	1	2	3	4	5
SVM	6.1	3.2	2.7	2.6	2.5
FFT	11.1	6.3	4.4	3.5	3.0

We also compared the SVM estimate vs the FFT estimate for the following noise combinations as well: 1.) AWGN + Impulse, 2.) AWGN + Block, and 3.) AWGN + Impulse +Block. Table 3.6 presents the results for these combinations.

TABLE 3.6 :

SVM vs FFT Amplitude Ratios,

AWGN + IMPULSE, AWGN + BLOCK and AWGN + IMPULSE + BLOCK

	SIGMA	1	2	3	4	5
NOISE TYPE						
A,I[8]	SVM	5.2	3.1	2.7	2.5	2.5
	FFT	4.1	2.7	2.5	2.4	2.4
A,B[9]	SVM	4.4	2.9	2.7	2.6	2.5
	FFT	3.4	3.2	3.1	3.2	3.0
A,I,B[10]	SVM	3.8	2.7	2.5	2.5	2.4
	FFT	3.1	3.0	2.8	2.9	2.9

Finally, Figures 3.13, 3.14 and 3.15 show the spectral estimates for each of these noise combinations when sigma =1 to illustrate graphically what occurs to the signal estimate as the level of “sigma” is increased.

[8] AWGN + IMPULSE

[9] AWGN + BLOCK

[10] AWGN + IMPULSE + BLOCK

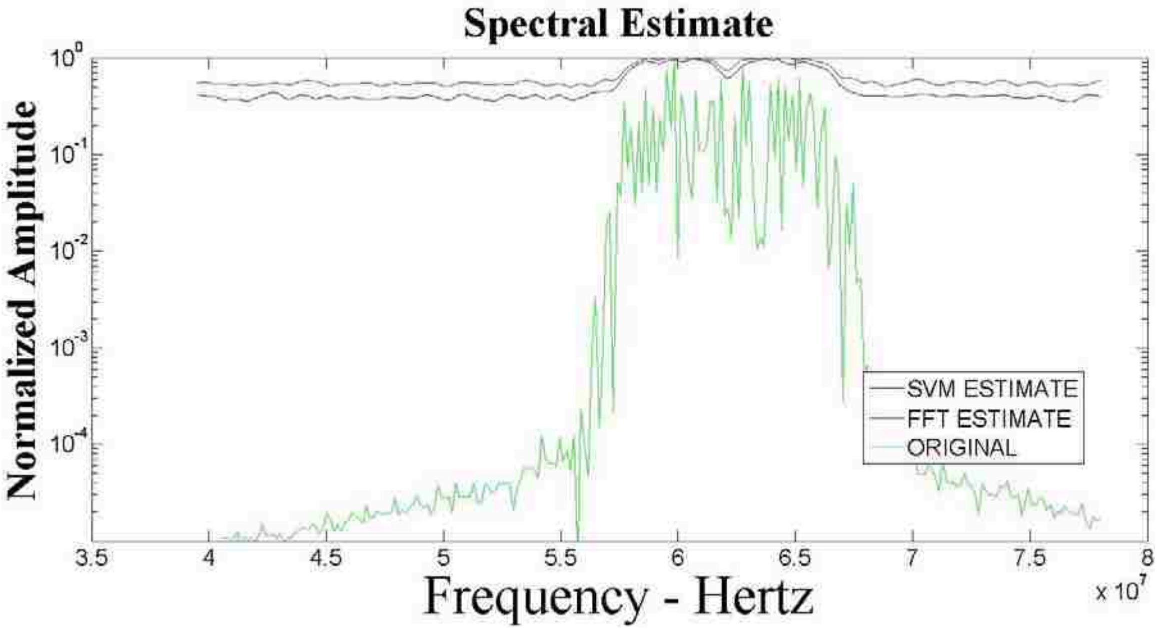


Figure 3.13: Spectral Estimate, AWGN + Impulse, Sigma = 1

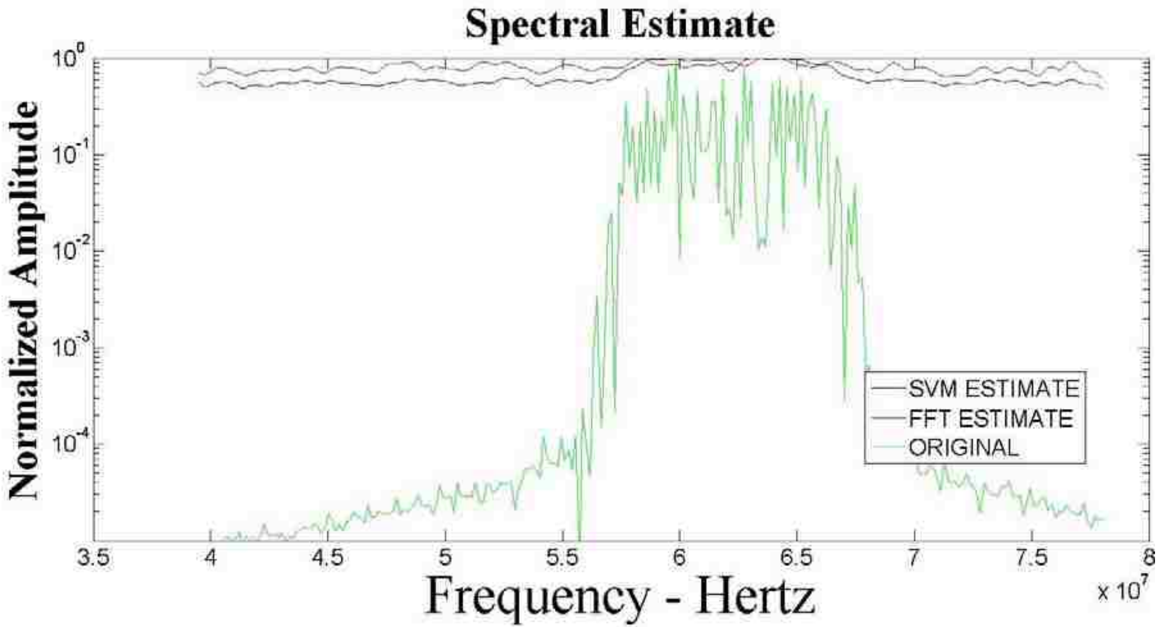


Figure 3.14: Spectral Estimate, AWGN + Block, Sigma = 1

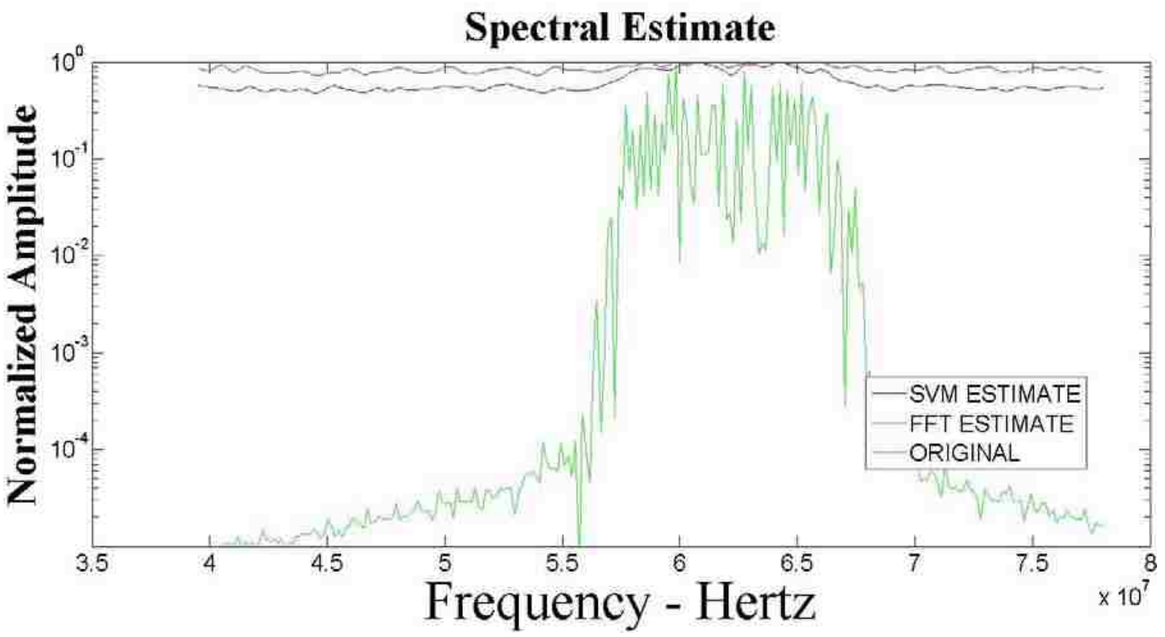


Figure 3.15: Spectral Estimate, AWGN + Block + Impulse, Sigma = 1

As stated earlier the WiFi services use different modulation schemes, and spreading codes, to be able to have faster data rates fit in the same 22 MHz band. The 1 Mbps service uses DBPSK and the 2 Mbps service uses DQPSK while using an 11 chip Barker code for the spreading code. The 5.5 Mbps and 11 Mbps services use the same modulation but use an 8 chip CCK code for the spreading codes. Since the occupied bandwidth is still the same, i.e. 22 MHz, we would not expect to see any drastic difference in the spectral estimate using either the SVM or FFT estimate. As we can see from Figures 3.23, 3.24, and 3.25 below that is not exactly correct. The CCK spreading code for the 5.5 and 11 Mbps service does not have the deep null in the middle of the occupied bandwidth that we see in the slower data rate estimates. This can be attributed to the fact that the CCK tends to spread the energy more evenly over the occupied bandwidth than the Barker codes.

Reference [23] requires that the header information that is appended to the raw data stream be modulated differently than the manner in which it is modulated for the 1 Mbps. The Simlink model for the 1 Mbps service was relatively easy to create in Simulink. The models for the higher data rate services were much harder to create and we adapted/modified a file that was available on the MATLAB website [69][11]. We created individual Simulink Models for each of the higher data rates and they are available in Appendix E. Figure 3.16, 3.17 and 3.18 show these estimates. These estimates are done with AWGN only and Sigma = 1. Again we can see that the FFT estimation works better than the SVM technique in this type of channel.

[11] If the file has disappeared from the MATLAB website please contact the author (tatwood@ece.unm.edu) for a complete file.

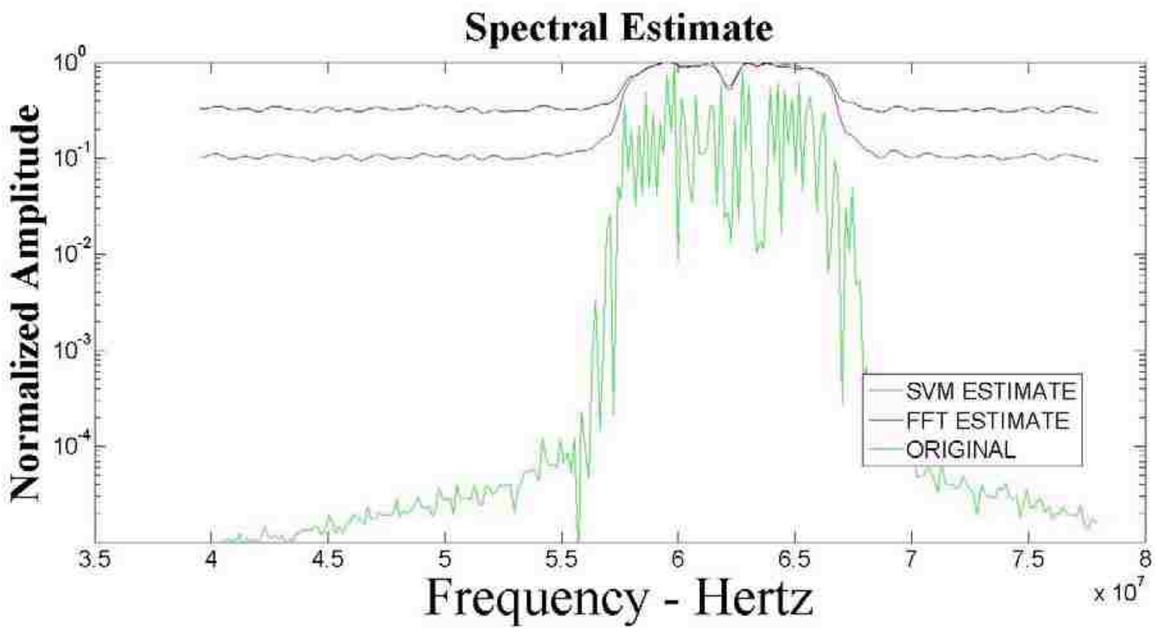


Figure 3.16: Spectral Estimate – 2 Mbps WiFi Signal

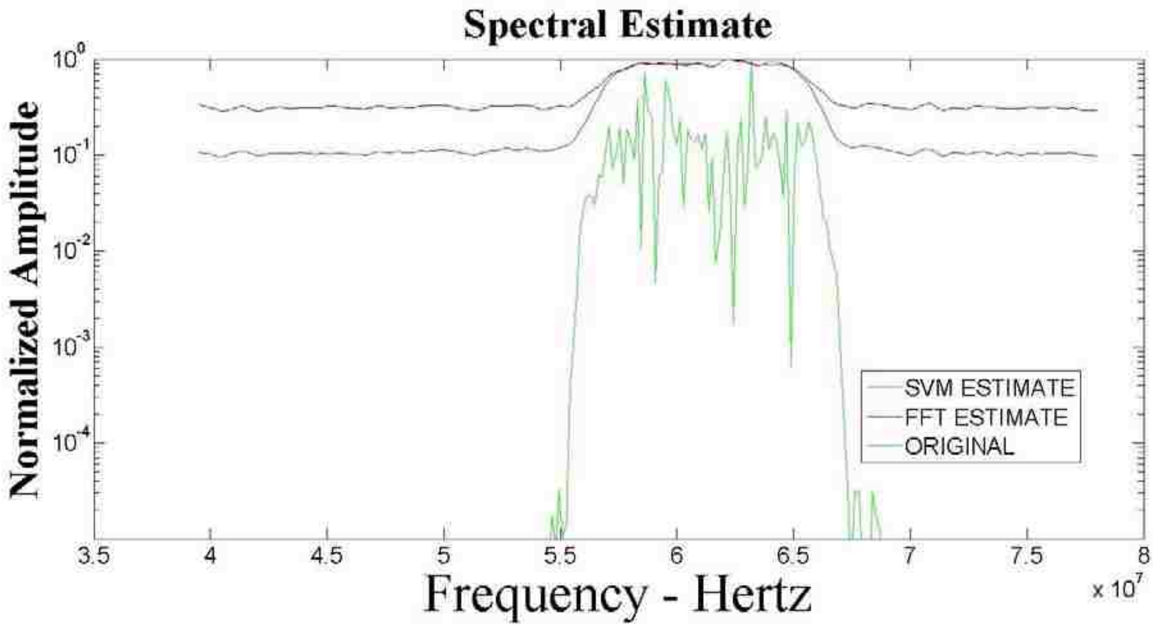


Figure 3.17: Spectral Estimate – 5.5 Mbps WiFi Signal

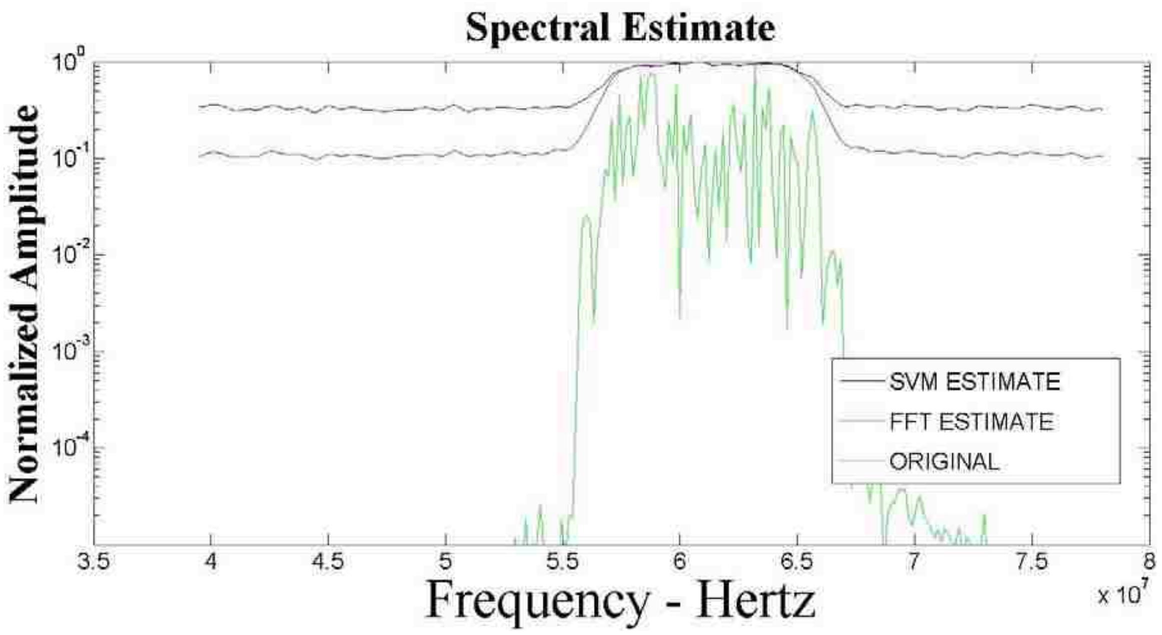


Figure 3.18: Spectral Estimate – 11 Mbps WiFi Signal

Finally, since a standard WiFi usage has to have two transmitters to complete a two-way link, we present a few estimates for two WiFi transmitters. As a means to deconflict channel usage, i.e. lessen channel interference, users are allowed to limit their usage to Channels 1, 6 and 11 by the 802.11 specifications. Table 3.7 lists the equivalent frequencies for these three channels [23].

TABLE 3.7

WiFi Channel Numbers and Equivalent Frequencies

CHANNEL	**1**	**6**	**11**
FREQUENCY (MHz)	**2412**	**2437**	**2462**

These channel allocations are separated to ensure the non-overlap of the three different channels at the 22 MHz bandwidth point with an additional guardband between the channels included. Figure 3.19 shows an estimate of a Double WiFi transmitter with AWGN only at sigma =1. This estimate has the channels in basically adjacent channels so that there is no overlap in their occupied spectrum. The data for these estimates were generated by the same Simulink models that generated the Single WiFi data, we just created a double WiFi model. Since the estimates for the different modulations and data rates for the single WiFi channel did not yield any significant changes in our earlier investigations we limited our estimates to the 1 Mbps channel.

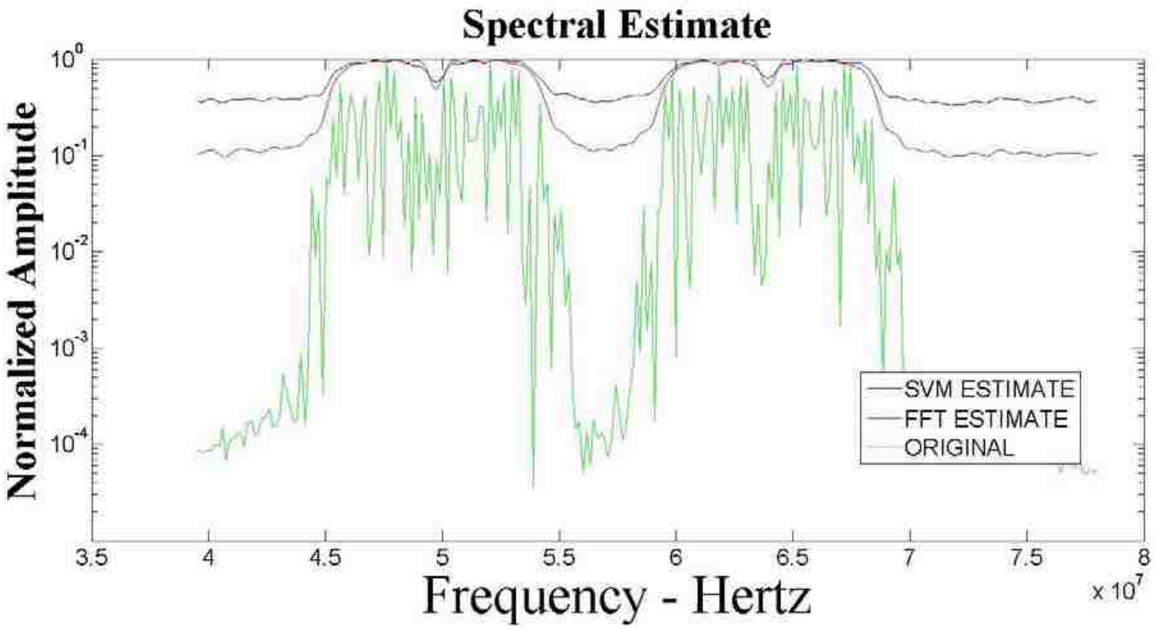


Figure 3.19: Parametric Estimator, Double WiFi, Adjacent Channels Sigma = 1

Figure 3.20 shows an estimate with AWGN + Impulsive + Block noise with sigma =1. As well the channels are spread much further apart to represent a more common usage of the channel for WiFi traffic. One thing that we see in these estimates is the fact that the FFT estimate appears to do somewhat better than the SVM estimate. We will see later that when we use the Parametric estimator we get a much better estimate including modulation information, even when the channels are close together.

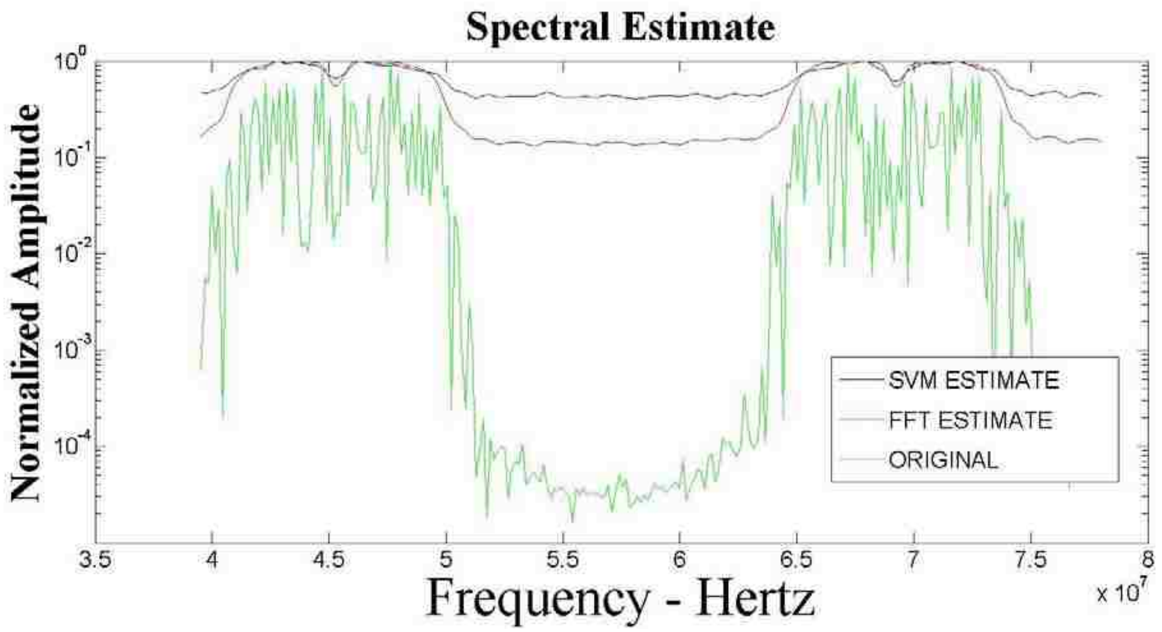


Figure 3.20: Spectral Estimate, Double WiFi, Spread Channels, A+B+I, Sigma = 5

Using the Parametric estimator for this type of signal shows the same good results for all levels of noise that might be encountered in this band. See Figures 3.21 and 3.22 as examples. Further spectral estimates are contained in Appendix G.

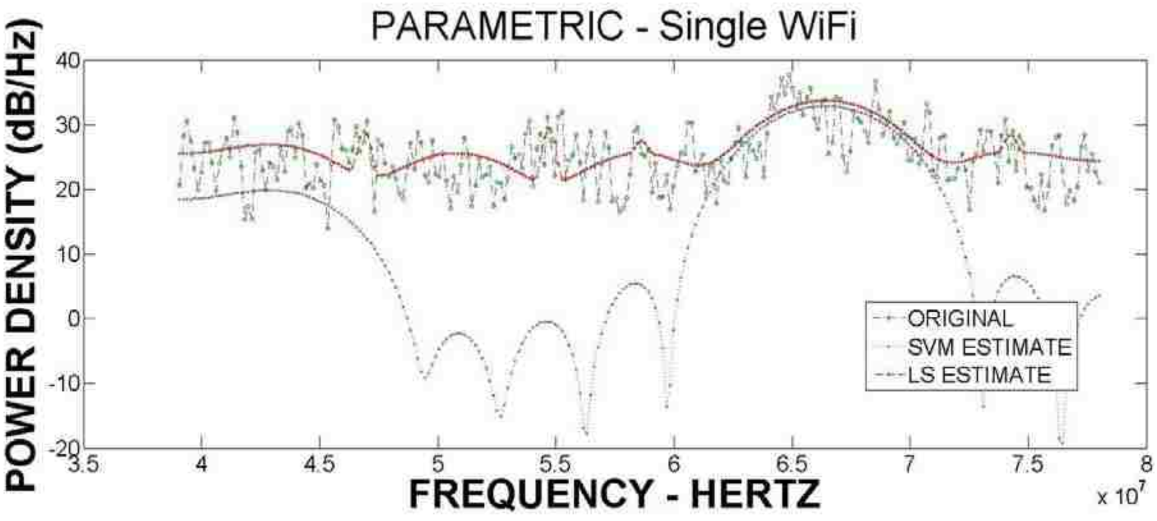


Figure 3.21: Parametric Estimator, Single WiFi, AWGN, Sigma = 1

As in the case of the Bluetooth transmitter we can see the artifacts in the estimate that are a function of the modulation.

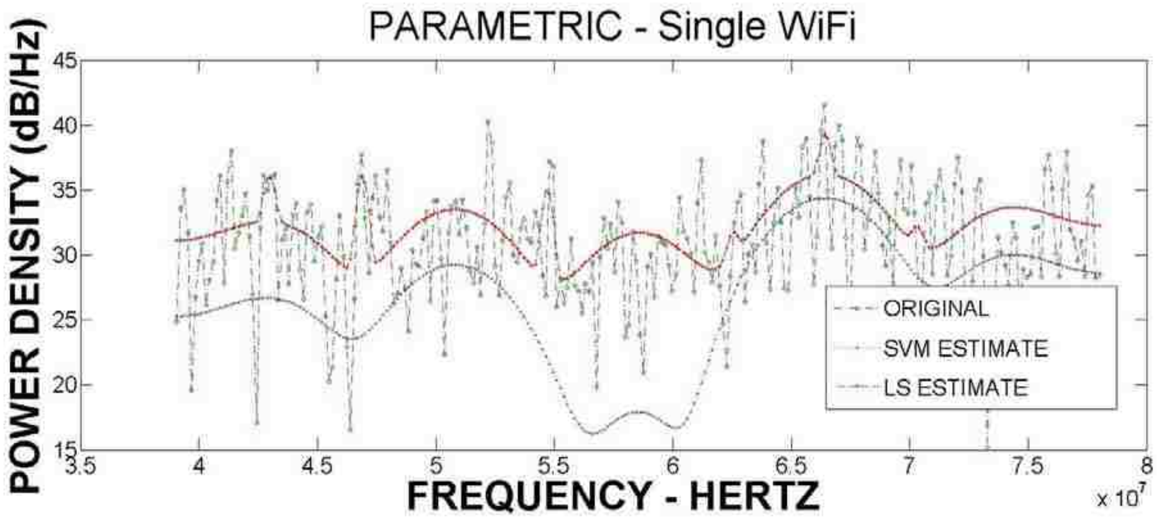


Figure 3.22: Parametric Estimator, Single WiFi, A+B+I, Sigma = 5

When the noise levels start getting really high (Sigma = 5) neither estimate does really well. Since this is a single estimate of the signal, only using 320 samples, perhaps a strategy like the Welsh periodogram might be useful in helping to build a better spectral estimate.

We can see that the Parametric estimate does a much better job of resolving two different WiFi transmitters in the same band as shown in Figure 3.23. The estimate is good enough to resolve the shoulders that are introduced by the modulation used in the channel.

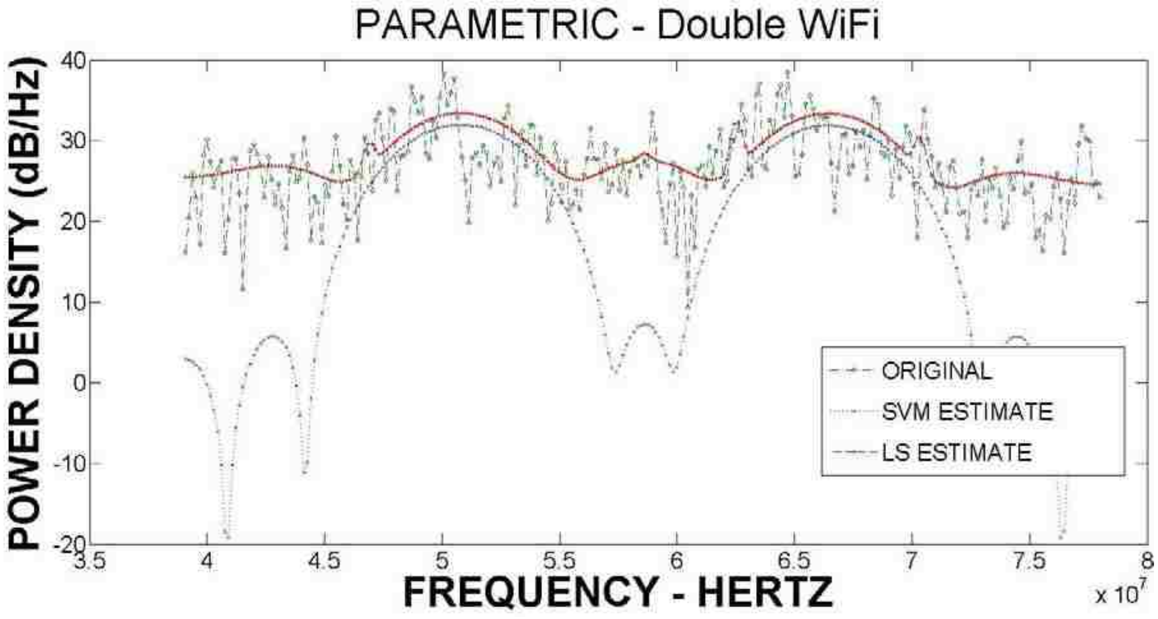


Figure 3.23: Parametric Estimator, Double WiFi, Sigma = 1

3.5 Bluetooth Piconet

3.5.1 Introduction

A full Bluetooth Piconet can have up to 8 devices, one of which is the master device, in one Piconet. A single device can be in more than one Piconet, but each Piconet can have a total of 8 devices. The master device regulates the hopping pattern so that all of the devices are hopping at the same time, each 625 µS, with a different pattern. This regulation of the hopping pattern by the master keeps the probability of more than one device, in a particular Piconet, occupying an individual frequency bin in the 2.4 GHz ISM band extremely low. Paying attention to the amount of time between Bluetooth hops defines an upper limit in which a CR has to characterize the RF environment and do its own required transmissions. If the CR is still transmitting when the channel changes, because of the Bluetooth hopping rate, the CR runs the risk of interfering with other transmitters.

To generate the required samples to feed to the estimation algorithms we use the MATLAB/Simulink model as detailed in Appendix F. This model is the summation of 8 of the transmitters that were developed for the “Single Bluetooth Transmitter” as detailed in Figure 3.5. The difference is that the “slave” transmitters operate in the odd numbered bands as defined in Equation 69 above.

3.5.2 Results

We start with a simulation that has no impulsive or block noise added to the signal. We add AWGN with a sigma of 1 to represent a channel with no interferers present. Figure 3.24 shows the spectrum estimation for this type of channel. Again we can see that the SVM technique offers no apparent advantage over the FFT technique. In fact all of the estimates for Sigma = 1 to 5 shows that there is no apparent advantage of the SVM technique over the FFT technique. Figure 3.25 shows the spectrum estimate with sigma = 5.

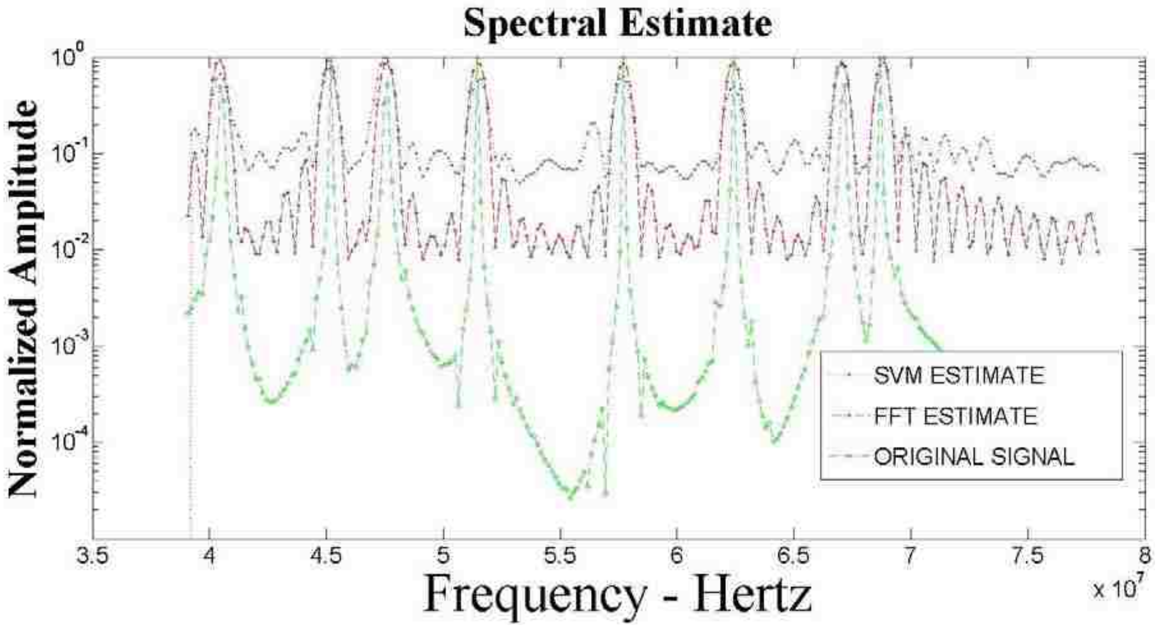


Figure 3.24: Piconet Network, AWGN Only, Sigma = 1

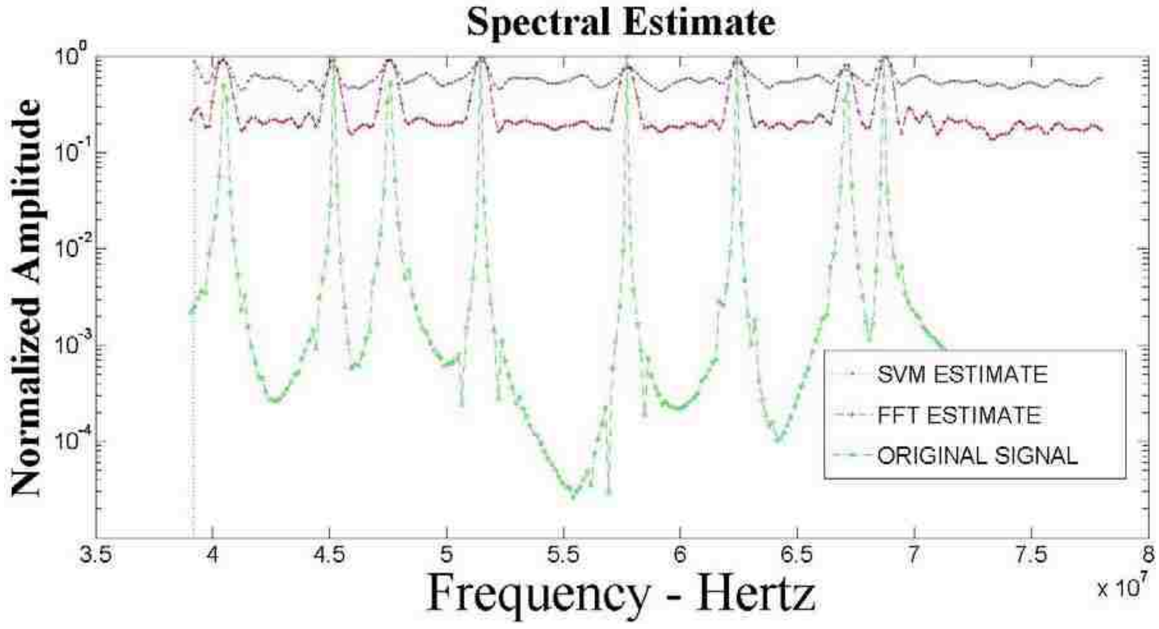


Figure 3.25: Piconet Network, AWGN Only, Sigma =5

In fact; we can see from the following figures that we do not see an advantage of the SVM technique over the FFT technique until we get to extremely heavy levels of Impulsive and Block noise being added to the base signal. We do see a minor improvement over the FFT estimate when there are heavy levels of Impulsive noise and no Block noise added as in Figure 3.27.

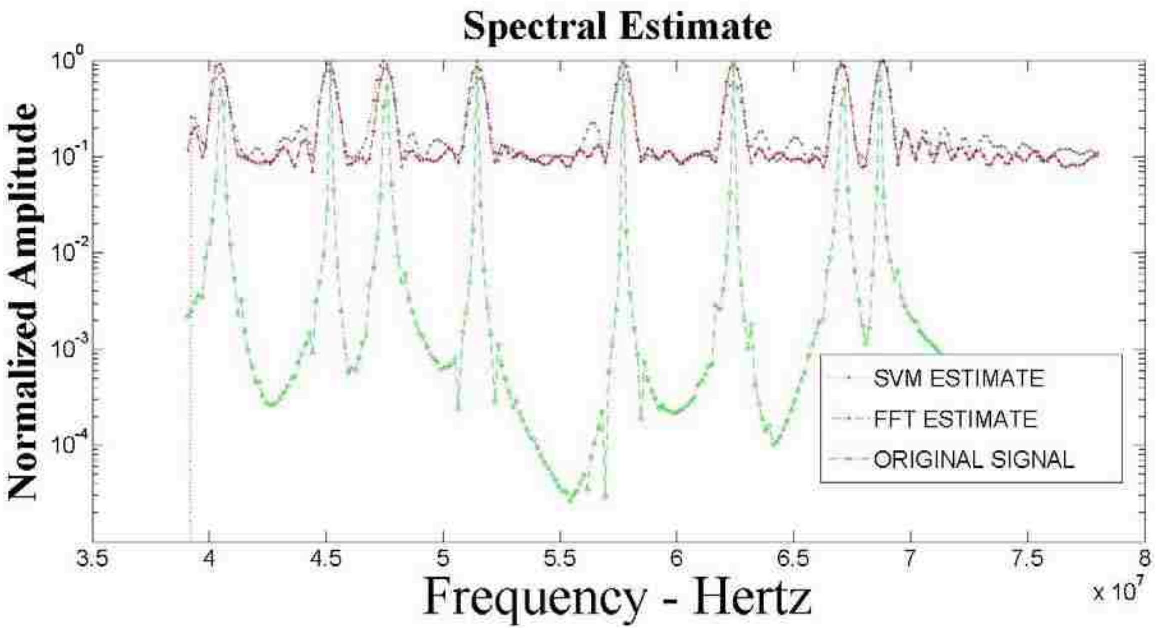


Figure 3.26: Piconet Network, AWGN + Impulsive, Sigma = 1

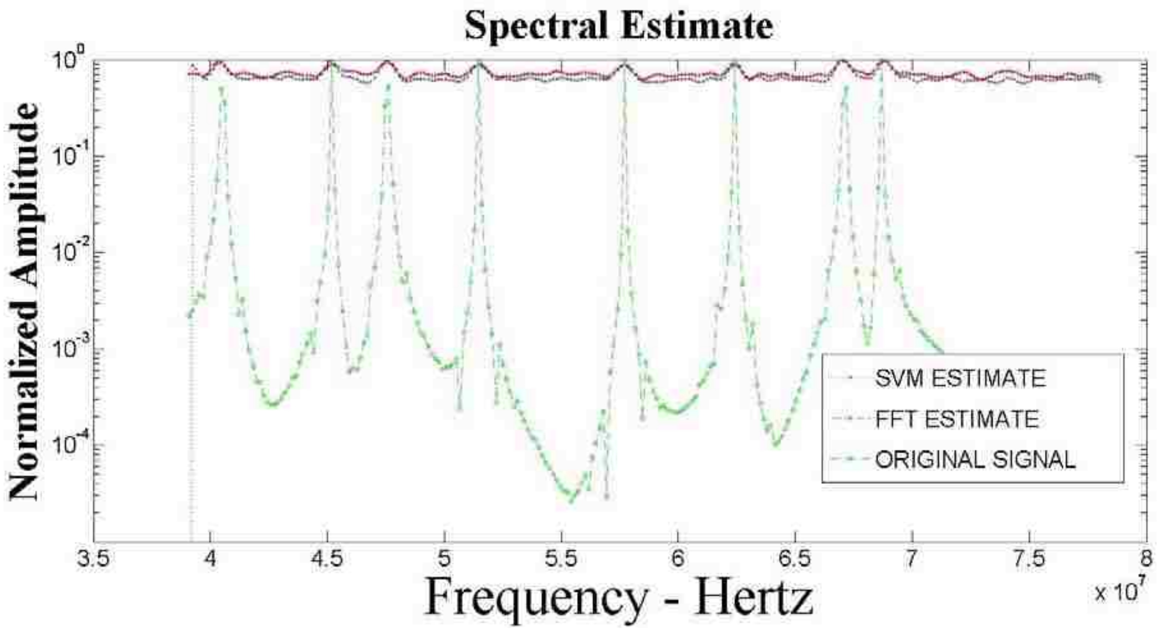


Figure 3.27: Piconet Network, AWGN + Impulsive, Sigma = 5

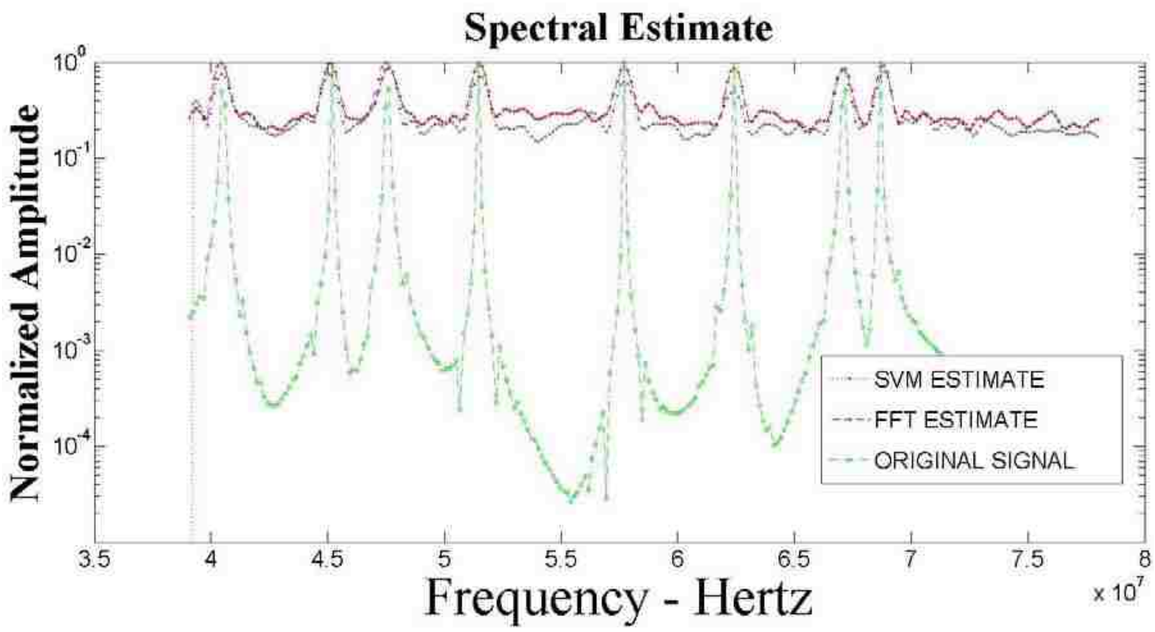


Figure 3.28: Piconet Network, A+I+B, Sigma = 1

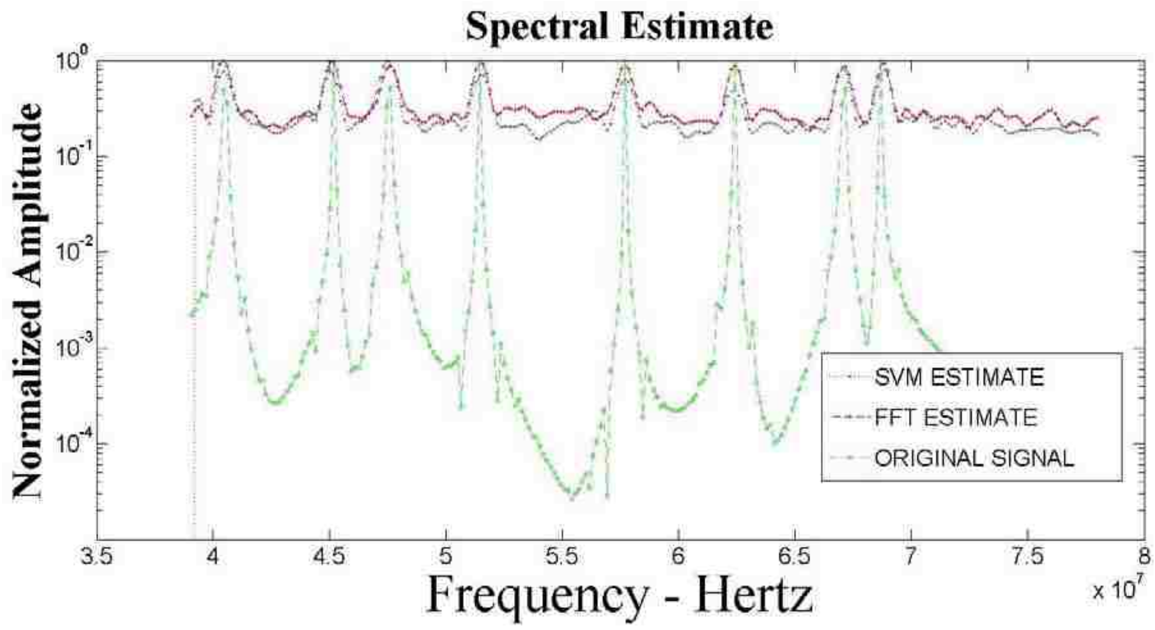


Figure 3.29: Piconet Network, A+I+B, Sigma = 5

Use of the Parametric estimator shows good results until we reach a situation of more than 6 transmitters. Since we investigated a single Bluetooth earlier in

Section 3.3 we start with two transmitters, then 5 transmitters and ending with 6 transmitters.

We present the estimates for 7 and 8 transmitters to show what happens to the estimate. These poor results are more common when the number of transmitters in the channel goes up then when the number is low as in around 1 or 2 transmitters. We are documenting these effects, and offer some thoughts as to what may be happening, in an effort to cover all possible outcomes from this research.

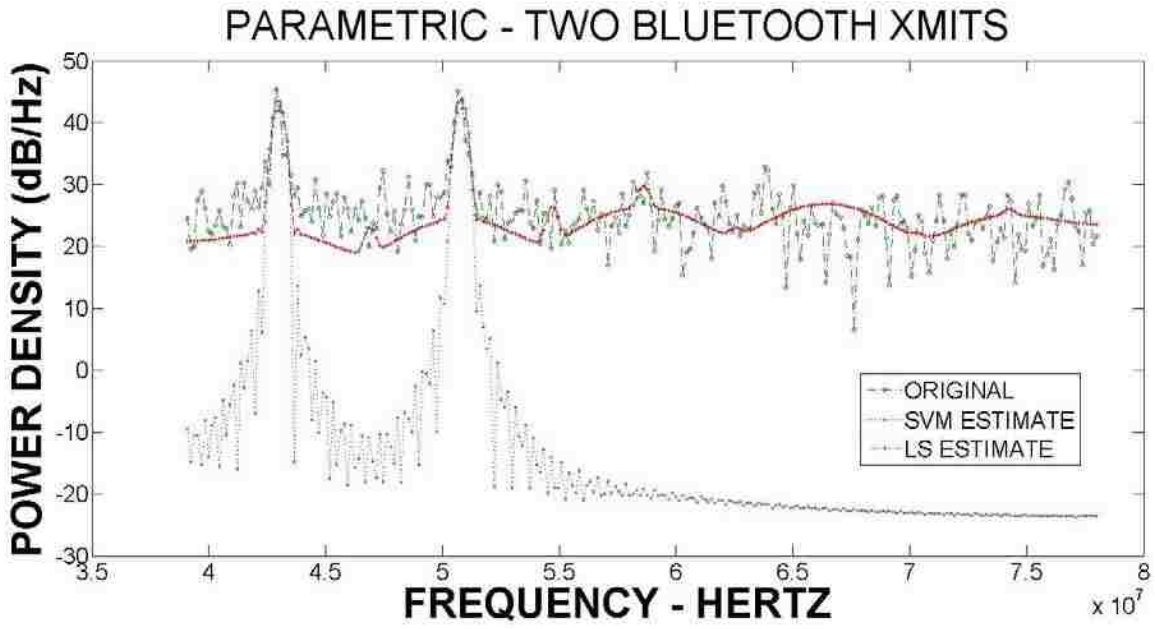


Figure 3.30: Two Bluetooth Transmitters, AWGN, Sigma = 1

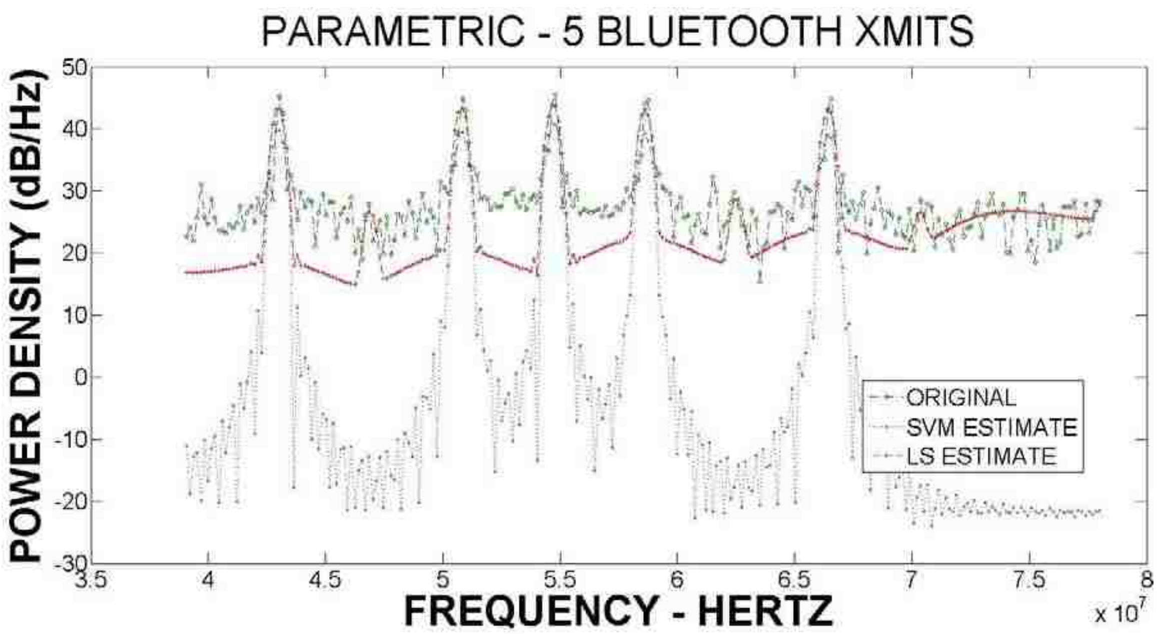


Figure 3.31: Five Bluetooth Transmitters, AWGN, Sigma = 1

In Figure 3.31 we note the presence of false estimates that the FFT estimate places in the spectrum even though the original data does not have these transmitters. This is important when a CR is trying to ascertain what the correct channel usage is for a particular frequency band.

To show that all is not sweetness and light Figure 3.32 presents an estimate where the FFT technique catches all of the transmitters while the SVM technique doesn't catch any of the transmitters. While this type of estimate is uncommon it does happen and deserves further investigation into possible mitigation steps, possibly a Welch periodogram where we average the estimate over a few different sets of symbols.

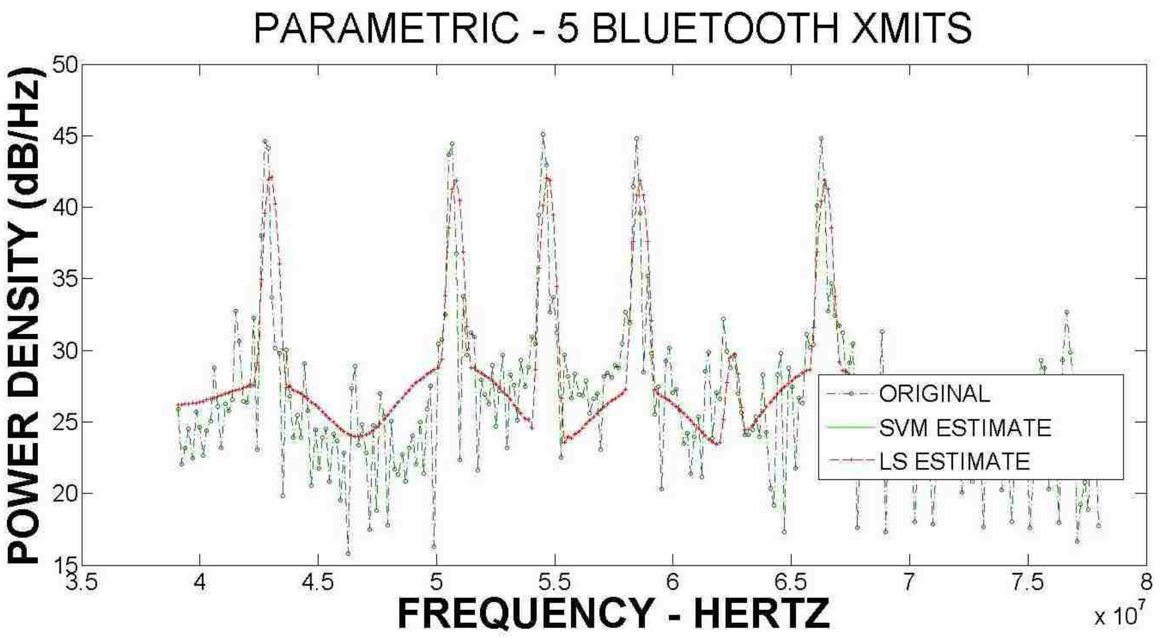


Figure 3.32: 5 Bluetooth Transmitters, AWGN, Sigma = 1, Bad Estimate

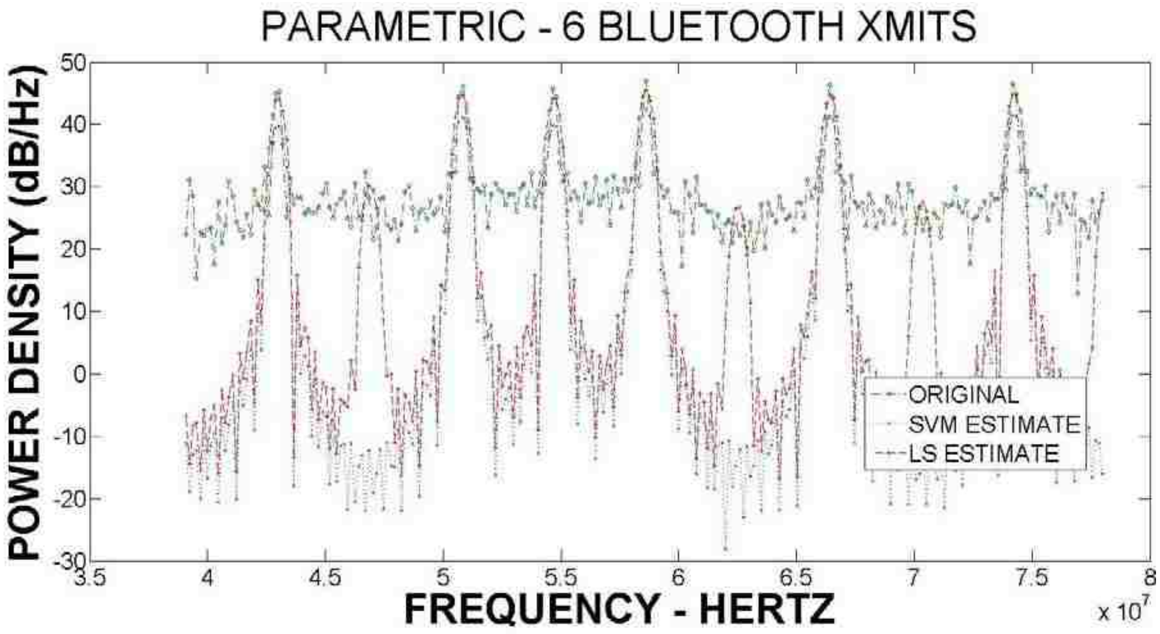


Figure 3.33: Six Bluetooth Transmitters, AWGN, Sigma = 1

In contrast to Figure 3.32; we see in Figure 3.33 a well defined estimate where we can see the modulation information as well as being able to identify where the transmitters are in frequency space. Also of note in all of these estimates for the multiple Bluetooth transmitters is the separation between the noise floor and the peak facilitating an easy distinction between noise levels and transmitter peak.

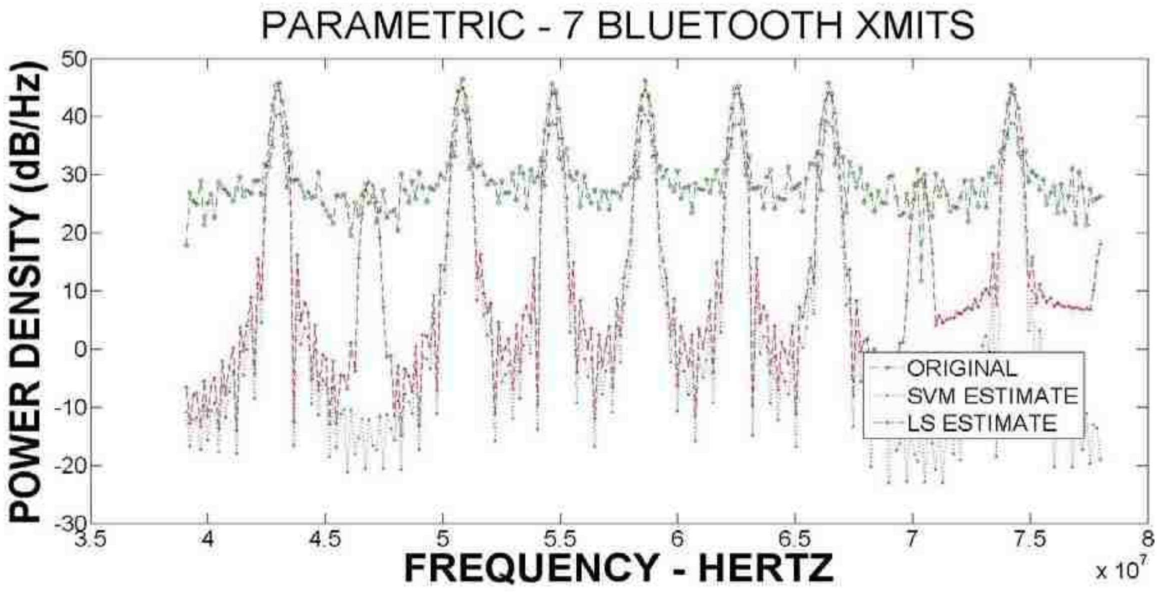


Figure 3.34: Seven Bluetooth Transmitters, AWGN, Sigma = 1

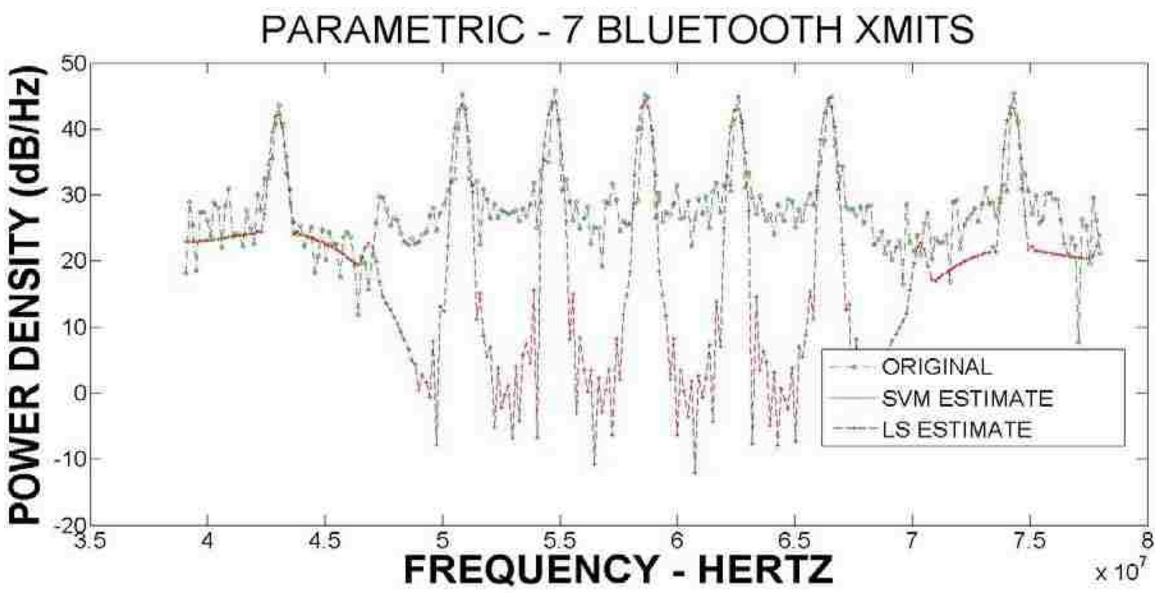


Figure 3.35: Seven Bluetooth Transmitters, AWGN, Sigma =1, Bad Estimate

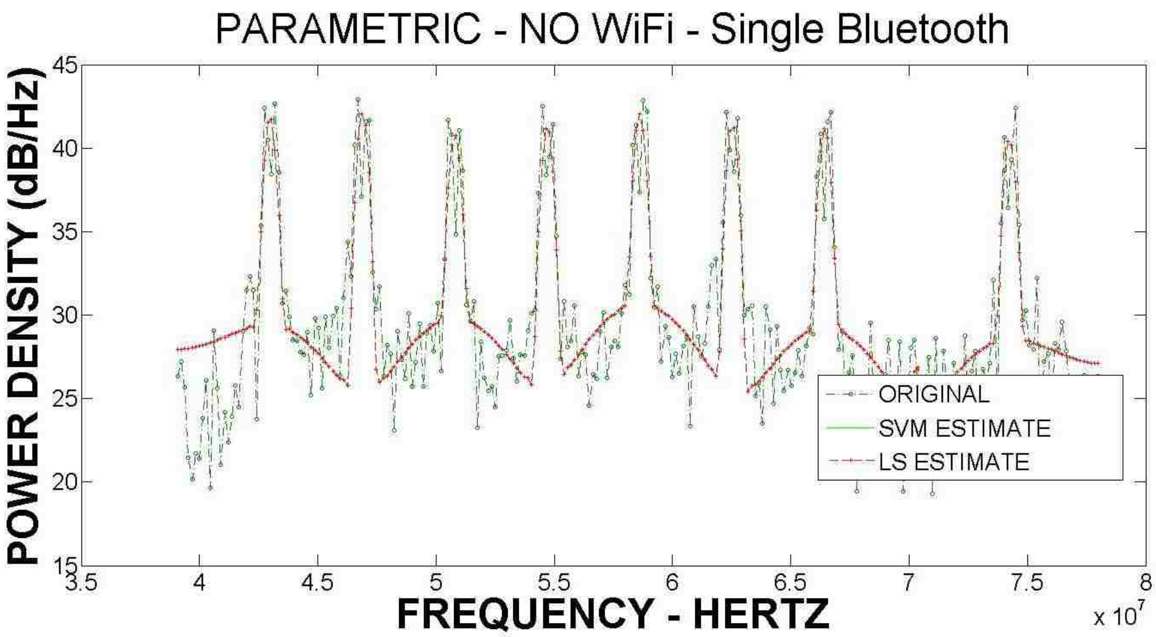


Figure 3.36: Eight Bluetooth Transmitters, AWGN = 1, Bad Estimate

Figures 3.35 and 3.36 show the worst type of estimate that the SVM techniques yield. It seems to appear when the total number of transmitters in the channel is high, around 7 or 8. The absence of the SVM estimate denotes that the algorithm returns a “null” estimate. Without extensive research the causes of this bad estimate cannot be determined.

3.6 Bluetooth And WiFi

3.6.1 Introduction

When we review any of the spectrum studies, for examples see [4] – [7], that have been published we see that the normal usage of the 2.4 GHz ISM is a mixture of Bluetooth transmitters and WiFi transmitters. Therefore to conclude Chapter 3 – Modeling and Simulation, we present several estimates of a mix of Bluetooth and WiFi transmitters in various frequency bins in the 2.4 GHz ISM band.

The data for these estimates were generated by combining the SIMULINK simulator for the Bluetooth Piconet with the simulator for the WiFi.

3.6.2 Results

Figures 3.37 and 3.38 presents estimates of an estimate with AWGN only with sigma = 1 and an estimate of AWGN + Impulsive + Block Noise with sigma = 5 respectively. See Appendix I for further estimates that explore more of the various noise combinations.

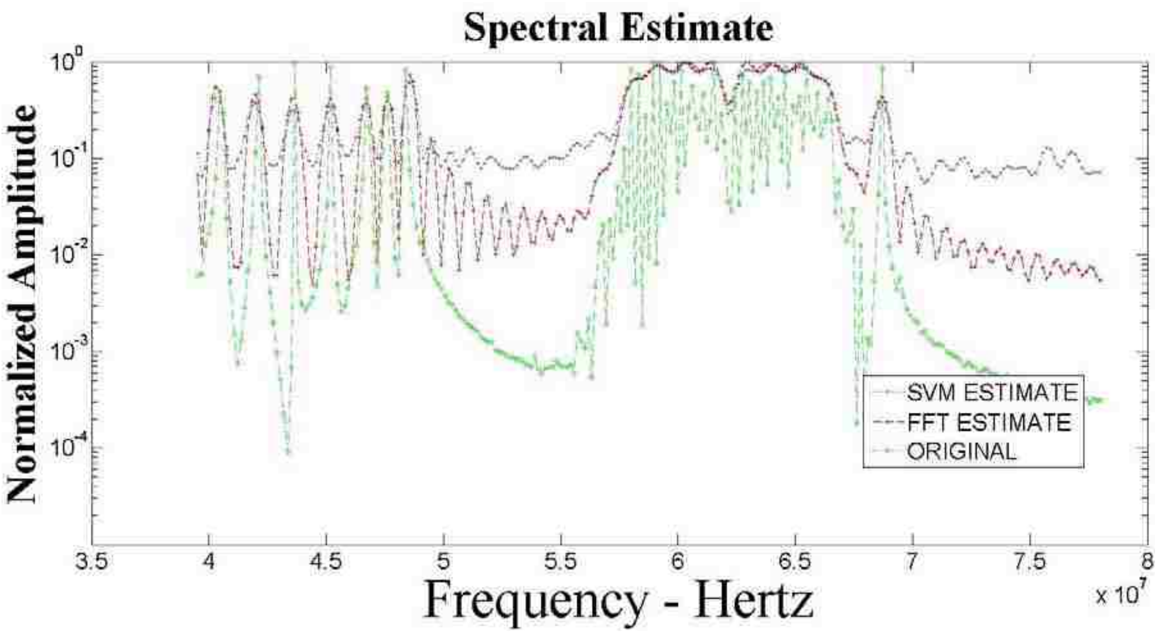


Figure 3.37: WiFi and Bluetooth, AWGN, Sigma = 1

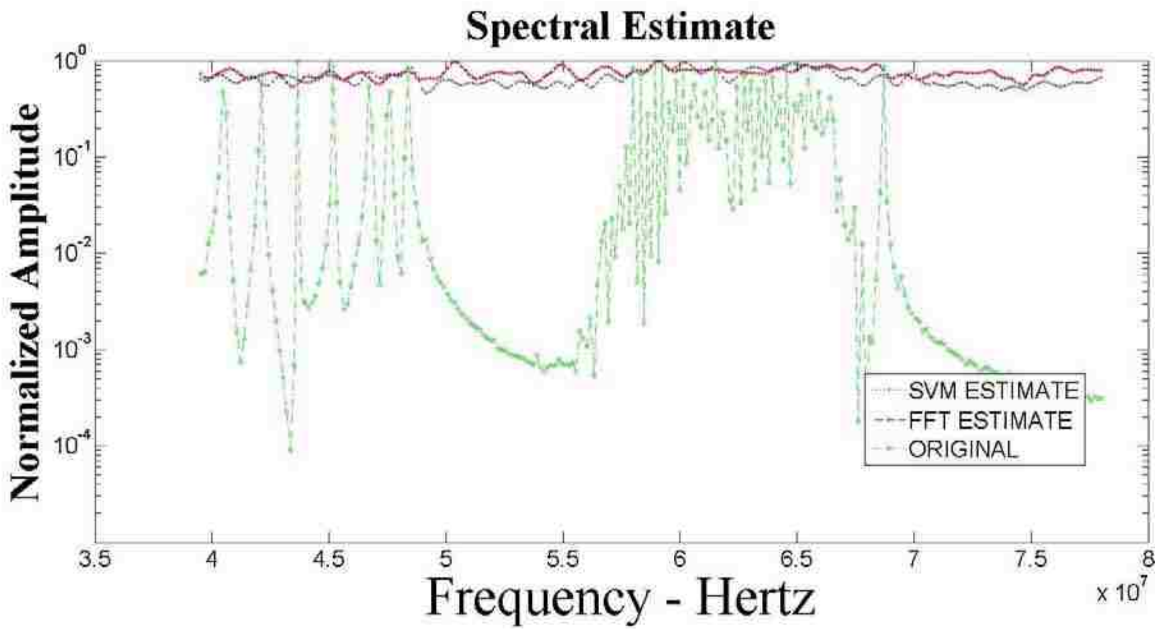


Figure 3.38: WiFi and Bluetooth, A+B+I, Sigma = 5

We can see from these estimates, as well as from earlier estimates, that when the spectrum starts to become congested the "Non-Parametric" estimate has no advantage over the FFT estimate. In fact, in a lot of the estimates the FFT technique does a better job than the SVM estimate.

In the Parametric results presented below we see a dramatic improvement in the SVM estimate over the FFT estimate. While the definitive answer as to why this occurs, good follow-on work, the answer probably lies in the use of the kernels that are used by the SVM algorithms to perform the similarity measure. Since we carefully craft the kernels, using knowledge of what we expect to be in the channel, for use in the “Parametric” estimator versus a simple kernel composed of sines and cosines for the “Non-Parametric” estimator we are seeing the types of results that we would expect. Figures 3.39, 3.40 and 3.41 show estimates from the “Parametric” algorithm for a single WiFi transmitter and 2, 5 or 6 Bluetooth transmitters.

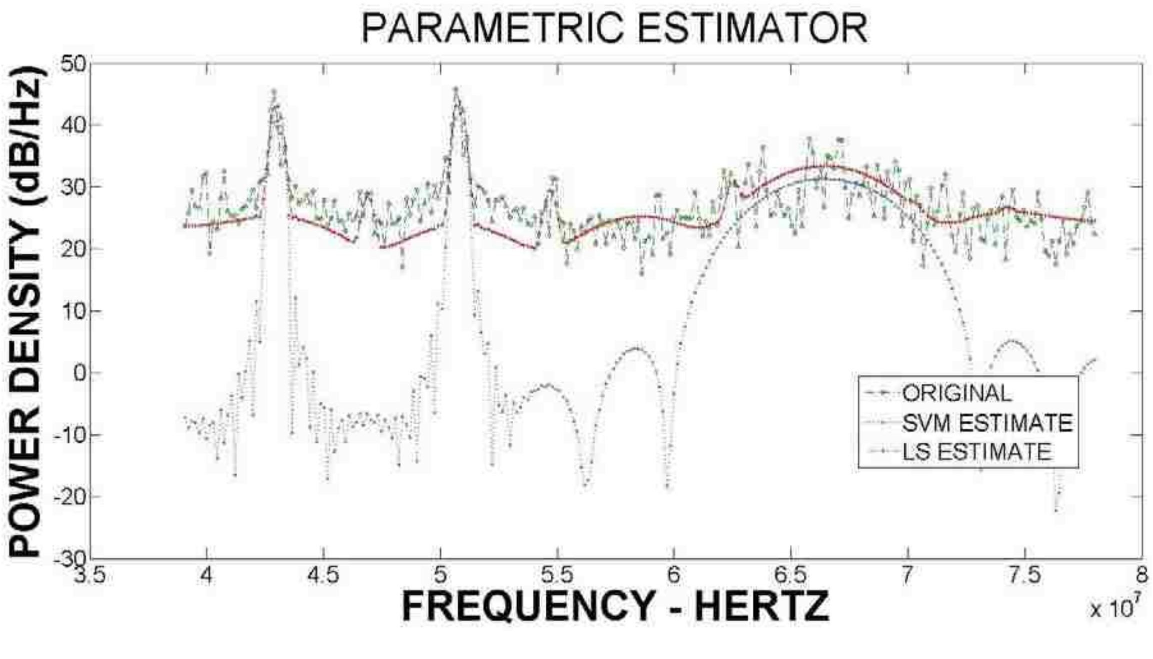


Figure 3.39: Parametric, WiFi And Two Bluetooth, AWGN, Sigma =1

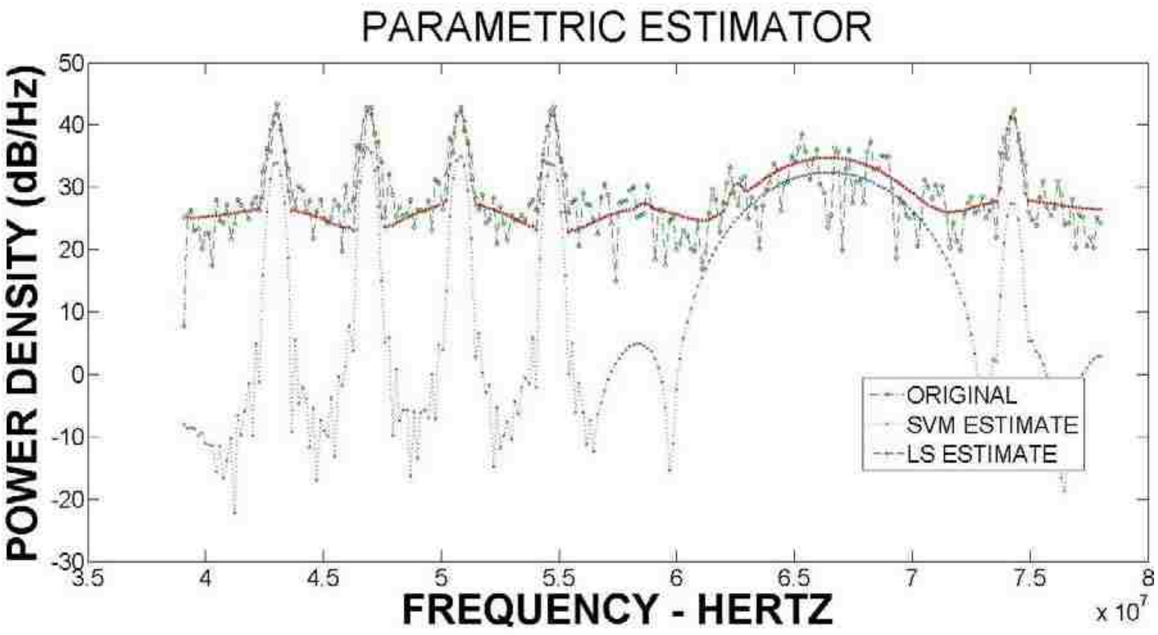


Figure 3.40: Parametric, WiFi And Five Bluetooth, AWGN, Sigma = 1

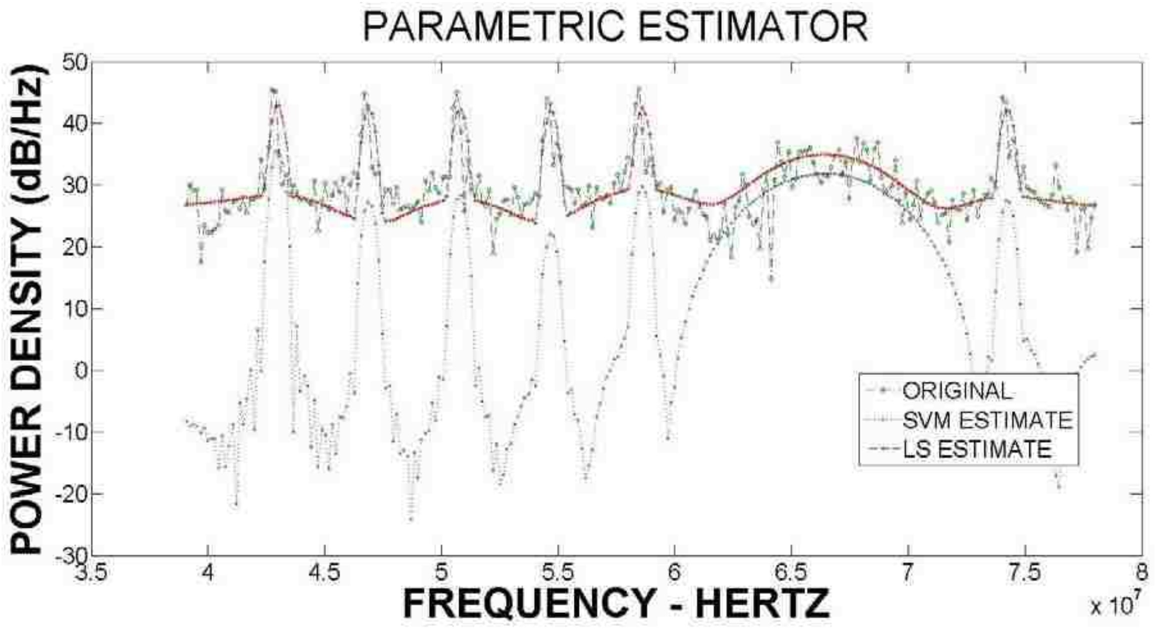


Figure 3.41: Parametric, WiFi And Six Bluetooth, AWGN, Sigma = 1

3.7 Performance with Amplitude Variations

All of the previous estimates in Sections 3.3 thru 3.6 assumed that the nominal amplitudes of the signals that were generated were equal. In reality, this is an unreasonable assumption for these two types of signals. We would expect that the mix of signals that are received at a particular receiver would have a range of amplitudes due to a variety of channel impairments, including multipath and spreading loss from the distance between transmitter and receiver. As we can see in Table 3.8 the range of allowable transmit powers is 20 dB for Bluetooth transmitters and 10 dB for WiFi transmitters. What this means: even if all of the transmitters in a channel were at the same distance from a cognitive radio, the variations in allowable transmit power will mean that the received amplitudes will be different.

Table 3.8: Allowable Transmit Power (EIRP) for WiFi and Bluetooth

	Minimum EIRP	Maximum EIRP
Bluetooth	1 mW	100 mW
WiFi	10 mW	100 mW

Obviously, the various transmitters that a cognitive radio could see at a particular time in a particular channel will be at different distances from the cognitive radio, meaning the received amplitude of each signal will be different. In addition, each channel could have different fading characteristics that affect the received amplitude as well.

Modeling the various types of channel, and the effect on received amplitudes, that are possible in real world situations is beyond the scope of this research; for the following investigations we will limit the channel impairments to a fixed channel attenuation of 0 – 20 dB for Bluetooth transmitters. We accomplish this by inserting an adjustable attenuator in the transmit path of each device and present some of the estimates, using the non-Parametric algorithm, that are obtained from these simulations.

We present estimates from a full Piconet simulation, keeping the master transmitter at 0dB attenuation and the other channels at attenuations of 1 dB less for each channel, i.e. for the first estimate channel 7 is 7dB below the master. For each successive estimate an additional 1 dB of loss is added to each channel until we reach a point where the estimate can no longer extract the signal. Additionally, all added noise is disabled to show how the estimates work without additional noise to start. we present the estimate of Figure 3.42 with the different types of noise that we have been using before, i.e. AWGN, AWGN + Impulsive, AWGN + Block, and AWGN + Block + Impulsive.

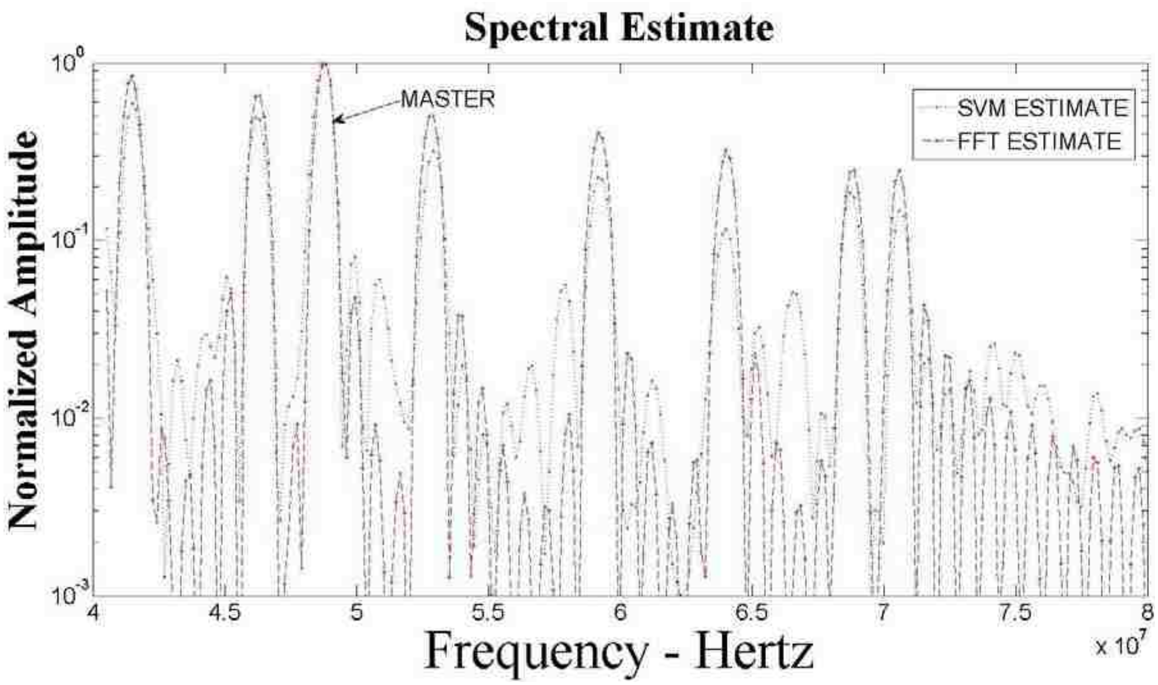


Figure 3.42: Amplitude Difference Estimate, Channel 7 @ 7 dB below Master

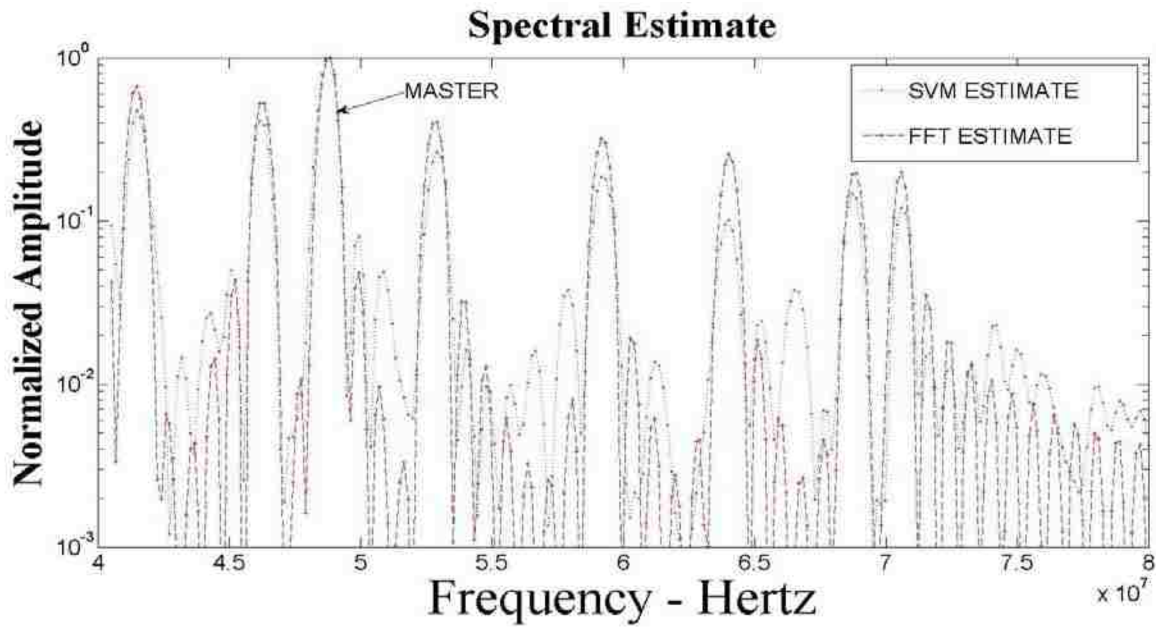


Figure 3.43: Amplitude Difference Estimate, Channel 7 @ 8 dB below Master

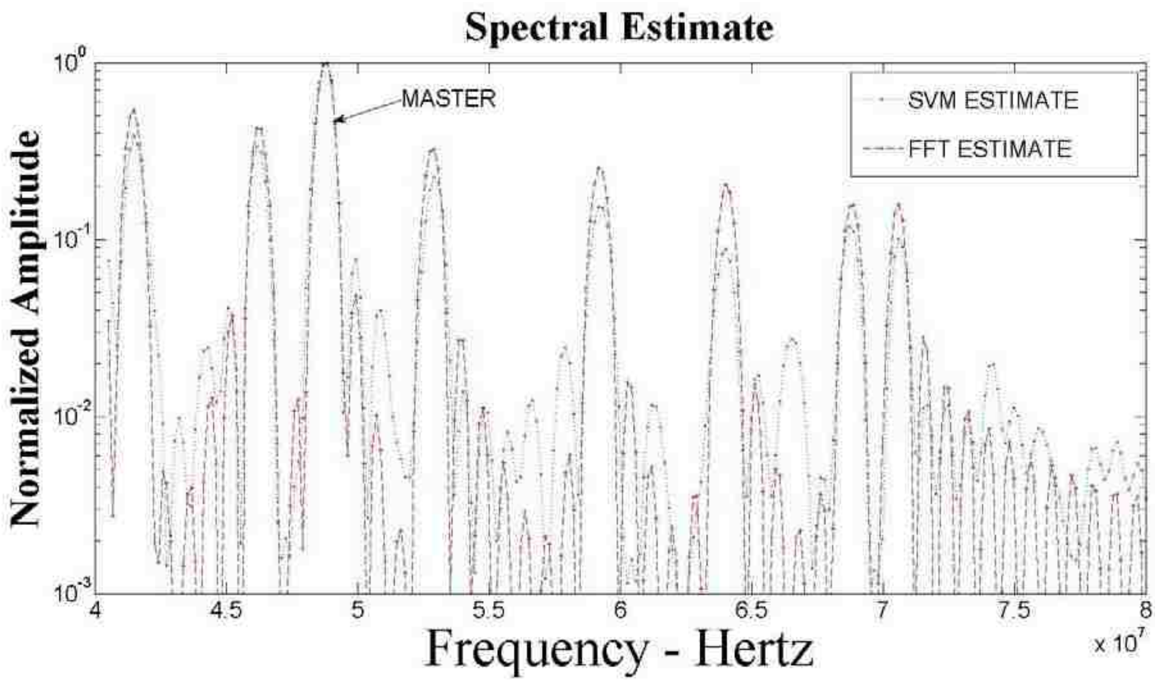


Figure 3.44: Amplitude Difference Estimate, Channel 7 @ 9 dB below Master

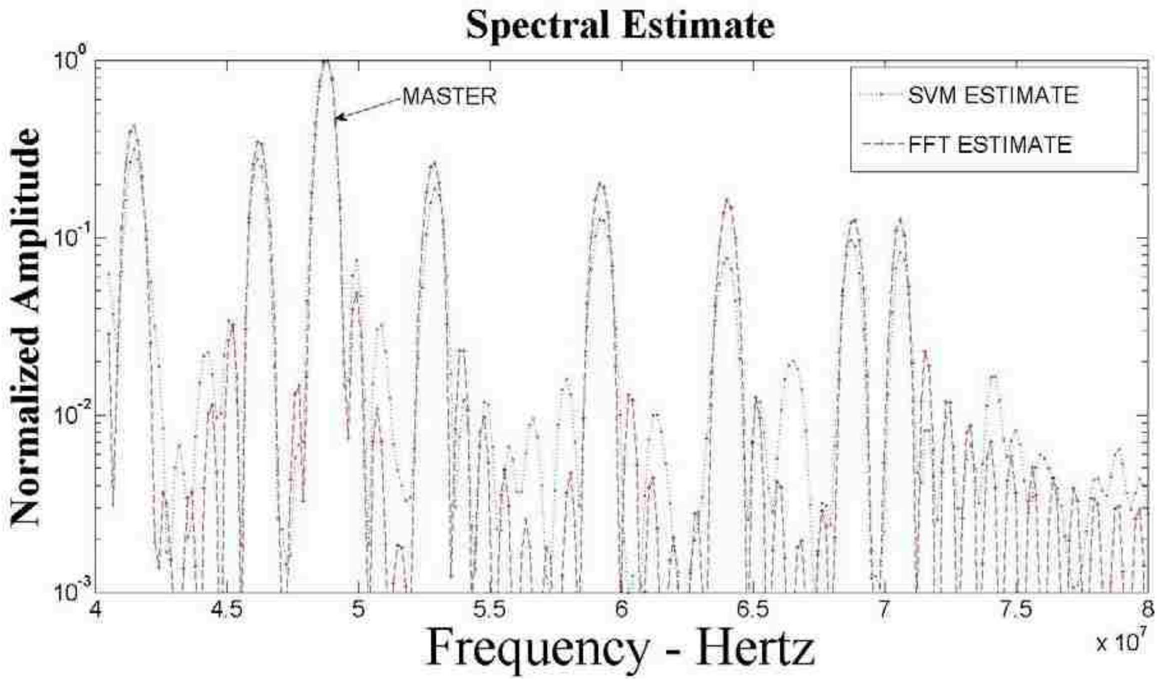


Figure 3.45: Amplitude Difference Estimate, Channel 7 @ 12 dB below Master

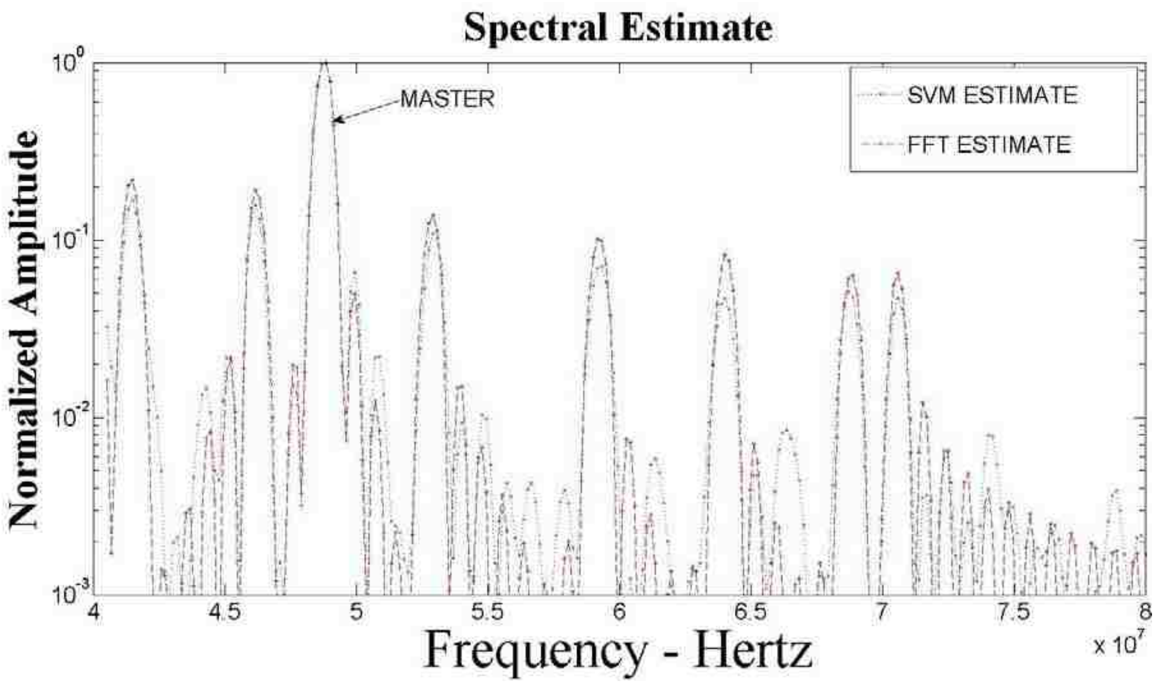


Figure 3.46: Amplitude Difference Estimate, Channel 7 @ 15 dB below Master

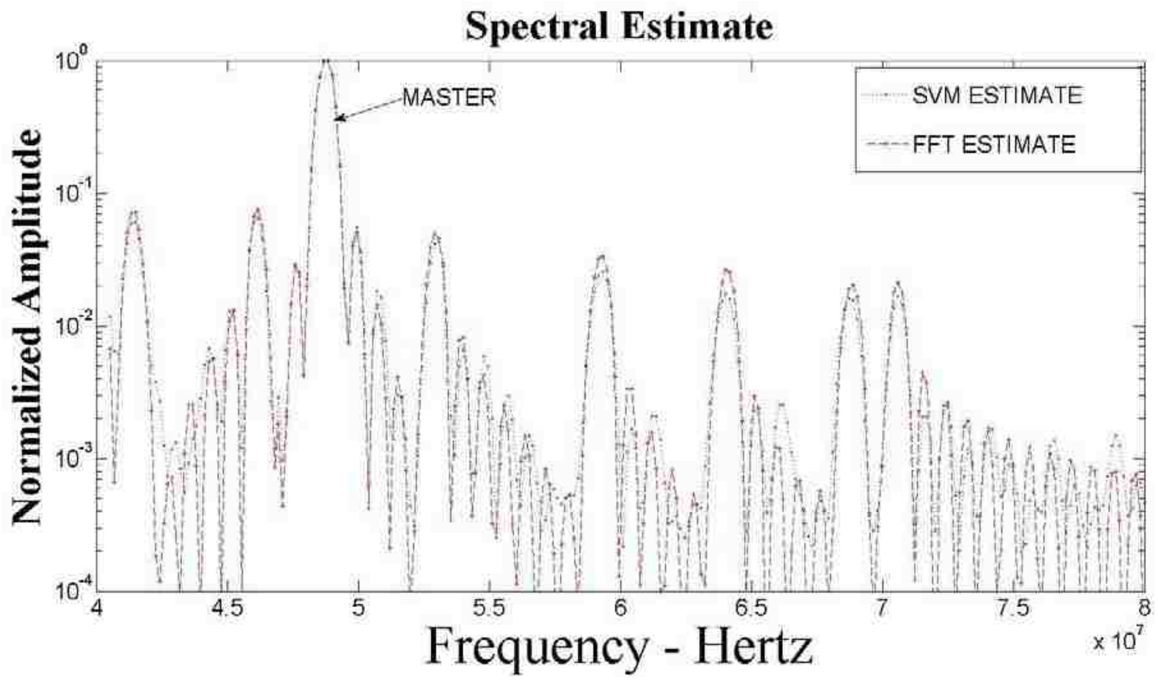


Figure 3.47: Amplitude Difference Estimate, Channel 7 @ 18 dB below Master

We see that, in the presence of no noise, both estimation techniques tend to do well down to a ~20 dB separation between the highest and the lowest transmitter. Looking at the transmitter that is to the left of the channel occupied by the master, Channel 2, shows that the estimates are starting to place the transmitter in the sidelobes generated by the master. Jumping ahead; Figure 3.48 shows an estimate when all of the transmitters have a nominal channel attenuation of 20 dB. Now; Channel 2 and Channel 3 (directly to the right of the master) appear at the same level as the sidelobes of the master raising the possibility of a mis-identification of the correct channels occupied by these two transmitters.

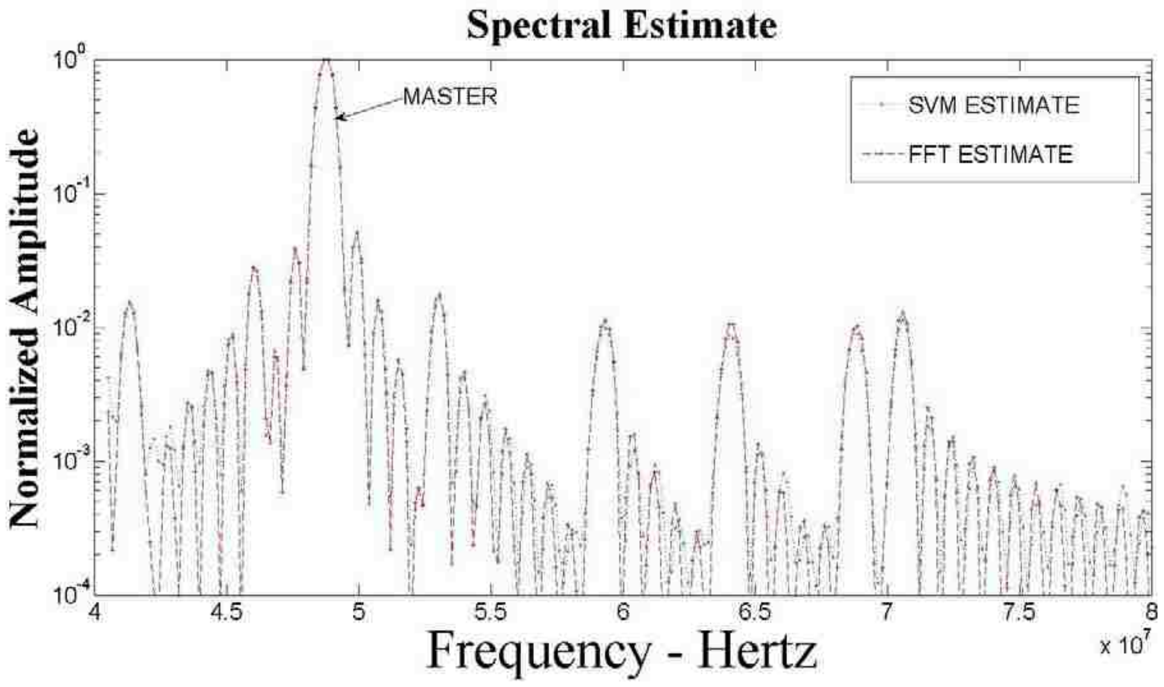


Figure 3.48: Amplitude Difference Estimate, All Channels @ 20 dB below Master

To investigate the affect of noise combined with amplitude differences we present the estimate of Figure 3.42 with the different types of noise that we have been using before, i.e. AWGN, AWGN + Impulsive, AWGN + Block, and AWGN + Block + Impulsive and sigma set to 1.

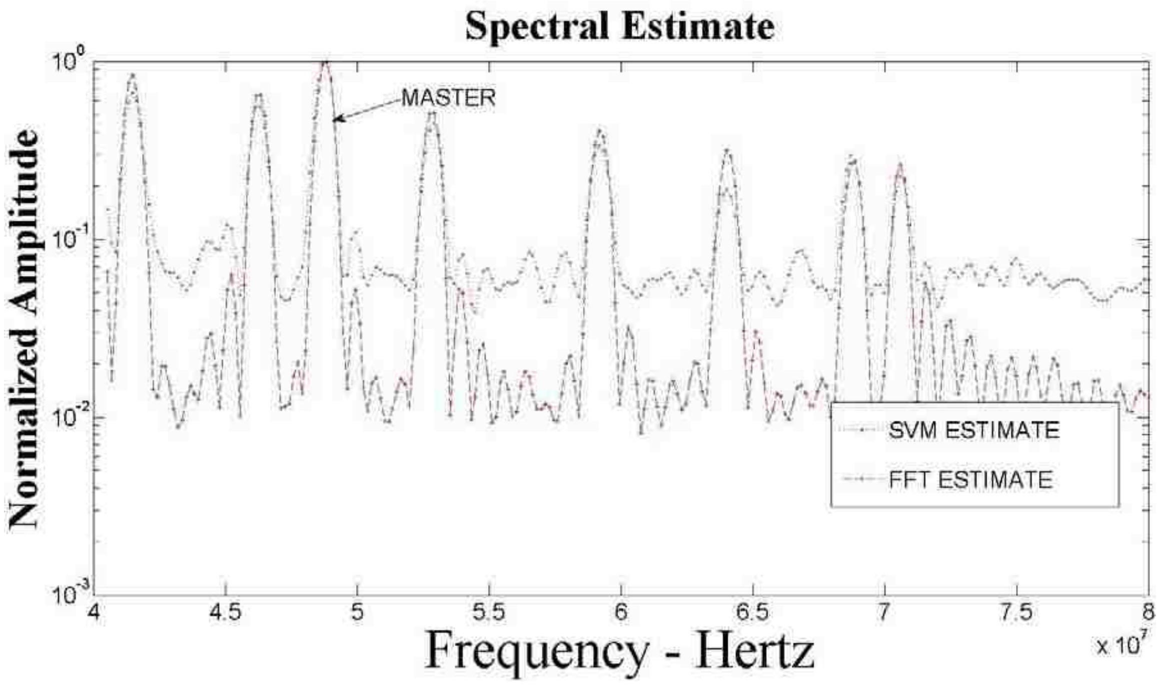


Figure 3.49: Amplitude Difference Estimate, Channel 7 @ 7 dB below Master, AWGN

As expected, in the presence of only AWGN, the SVM estimate is unable to resolve the occupied channels as well as the FFT estimate this is consistent with the findings from earlier when we held all the amplitudes equal between the transmitters. Now let's look at what happens when we introduce different types of noise.

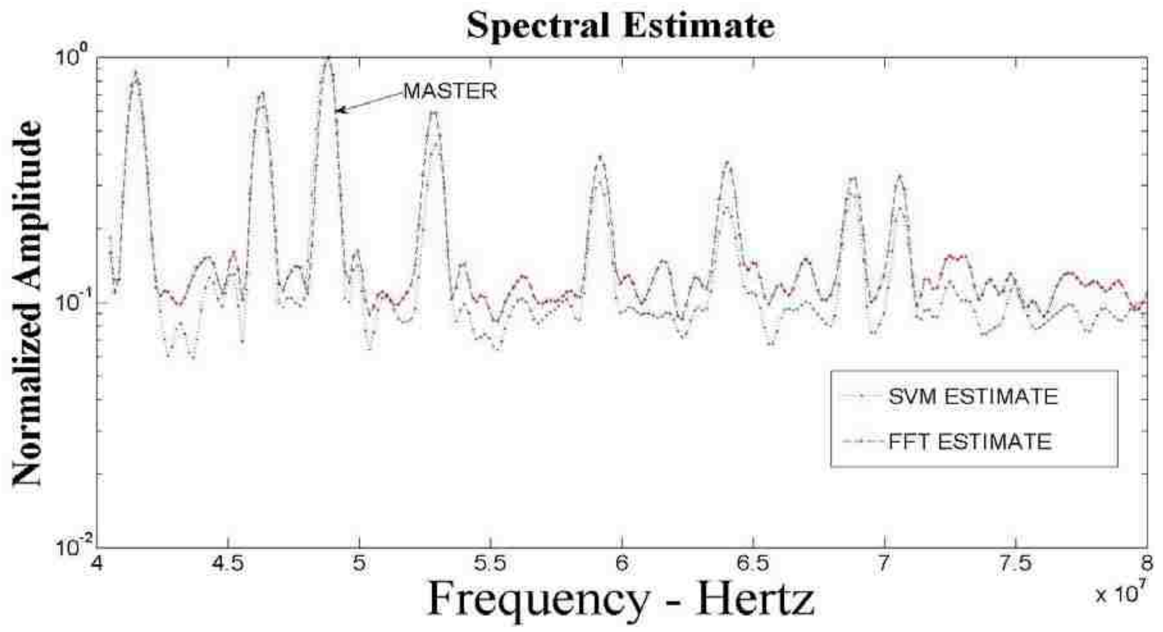


Figure 3.50: Amplitude Difference Estimate,
Channel 7 @ 7 dB below Master, AWGN + Impulsive

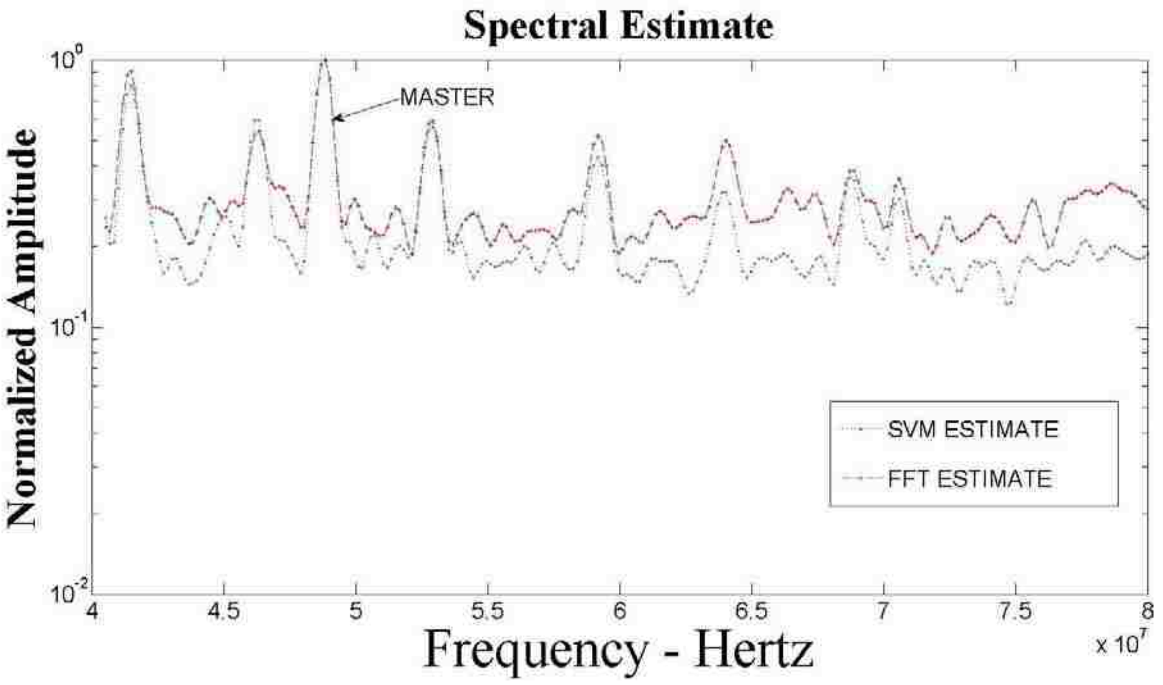


Figure 3.51, Amplitude Difference Estimate,
Channel 7 @ 7 dB below Master, AWGN + Impulsive + Block

As expected the SVM technique is better in the presence of non-Gaussian noise, see Figure 3.51. In Figure 3.51, because of a combination of the noise and the amplitude differences in the transmitter, the FFT technique places the estimate for Channel 7’s amplitude at approximately the same level as the noise background. This type of estimate could cause a Cognitive Radio to attempt to transmit in a portion of the spectrum that seems to be unoccupied but, in reality, is occupied.

4 EXPERIMENTAL SETUP

4.1 Description

Two experimental setups were constructed or modified to conduct airlink tests. The goal of the airlink tests was to propagate Bluetooth and/or WiFi signals in an increasingly more realistic propagation environment. The first site is located in a modified Faraday Cage, considered the best RF environment because of being shielded from external noise sources. The second site is an Open Air Propagation environment that has at least 3 distinct WiFi services in the immediate vicinity. For the remainder of this discussion we will name the sites as follows: 1.) Site 1 - Electrical Engineering Department Anechoic Chamber and 2.) Site 2 - Open Air Propagation Area.

Table 4.1 lists the equipment that was used at each site. Figure 4.1 shows a notional schematic diagram of how the equipment was set up. The major difference at each of the sites was the length of the cabling that was used.

Table 4.1: Airlink Tests Equipment List

Tektronix Digitizing Storage Oscilloscope – Model TDS 644A[12]
A.H Systems Dual Ridge Horn Antenna – Model SAS-571[13]
Mini Circuits Power Amplifier – Model ZX60-3011[14]
Hewlett Packard Power Splitter – Model 11667B[15]
Mark Microwave Mixer – Model M10208LA[16]
Cumings Microwave RF Absorber, Site 1– Model LF-79[17]
Agilent Spectrum Analyzer – Model N9912[18]
Agilent Waveform Generator – Model E4438C

[12] Information can be found at www.tek.com

[13] Information can be found at www.AHSystems.com

[14] See Appendix J for datasheet

[15] See Appendix J for datasheet

[16] See Appendix J for datasheet

[17] Information can be found at www.cumingmw.com

[18] Information can be found at www.aglient.com

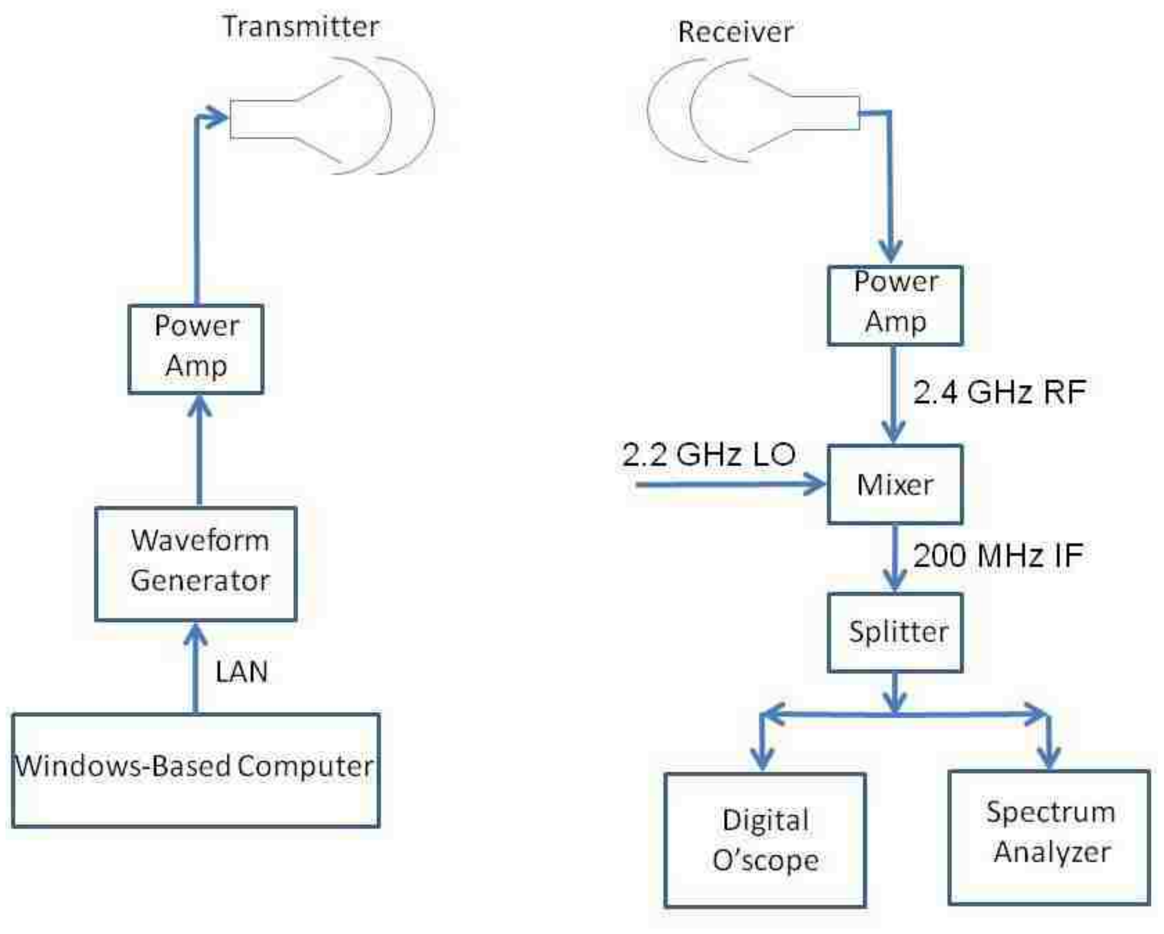


Figure 4.1: Notional Schematic Diagram of Test Setup

The first location, The Electrical Engineering Department Anechoic Chamber, is located in Room L206 in the basement of the Electrical Engineering Building on the University of New Mexico (UNM) Campus. This facility is a 20' x 20' x 20' enclosure used as a shielded screen room for High Power Pulsed Power experiments. As a shielded room, with bare metal walls, this room is not necessarily suited as a facility in which to conduct this type of propagation experiment. However, addition of RF absorbing material on the wall behind the receiving antenna and on the floor of the room gave an advertised reflection coefficient of -40 dB. Since we were only transmitting milliwatts of energy we decided that this attenuation was adequate for our needs. The compensation was the shielding effectiveness of the room, advertised at 100 dB, which would effectively exclude all outside signal sources, but we had no way to verify this number. Figures 4.2 and 4.3 show the inside of the chamber and Figures 4.4 and 4.5 show the outside, with test equipment setup, of this room.

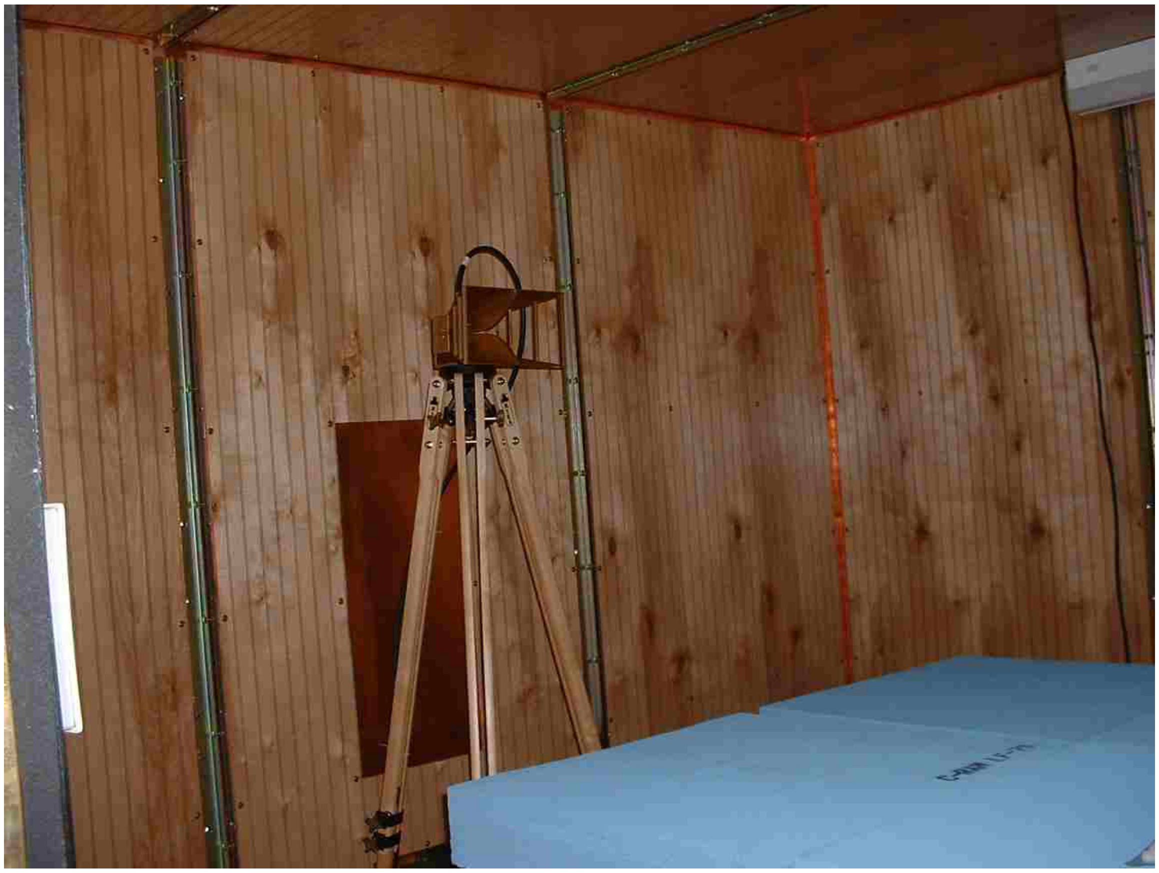


Figure 4.2: View inside Chamber Showing Transmit Antenna

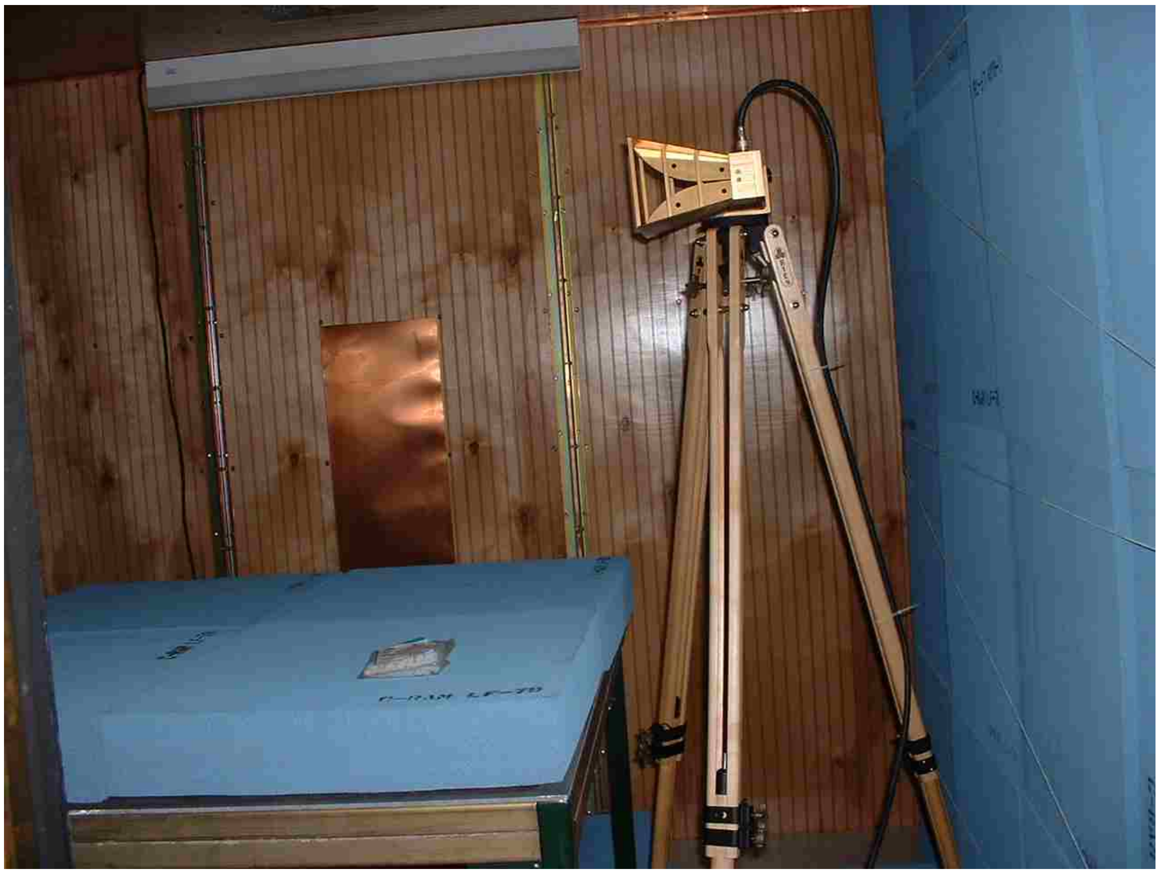


Figure 4.3: View inside Chamber Showing Receive Antenna

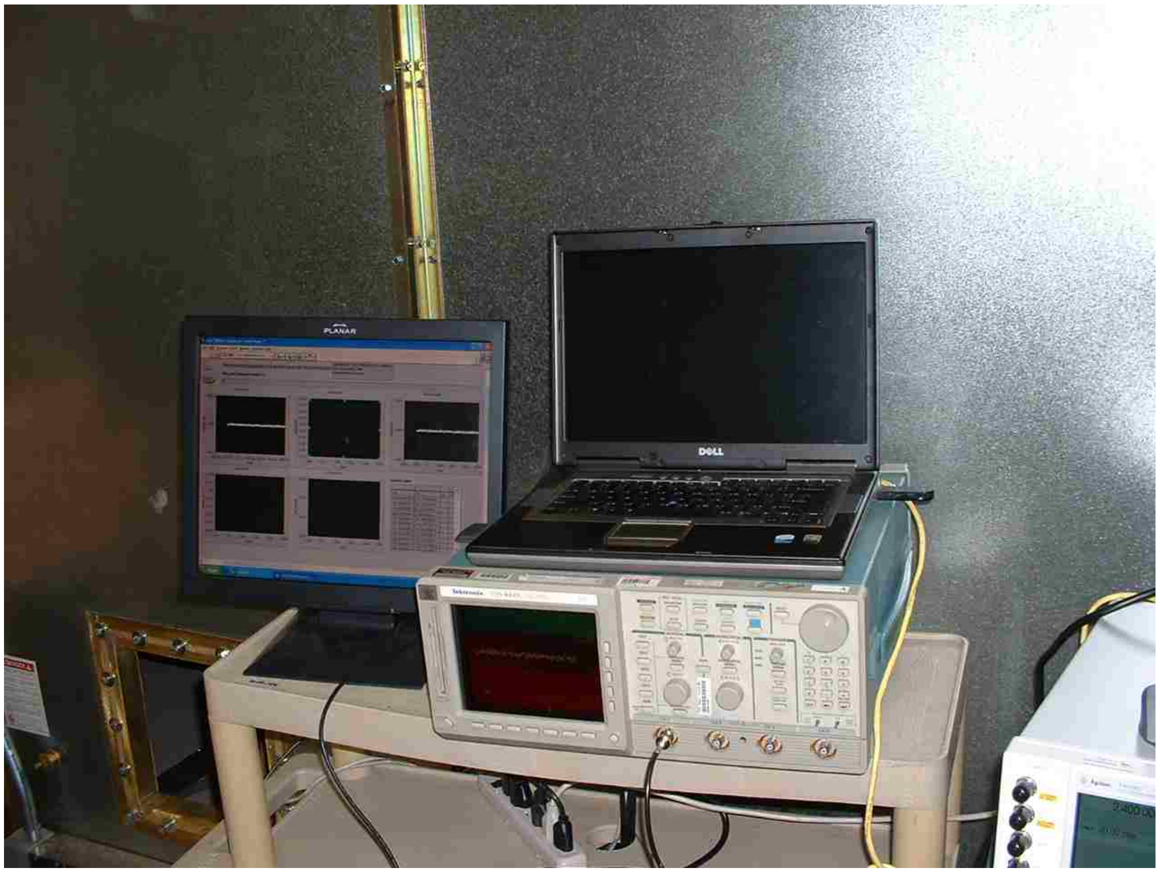


Figure 4.4: Chamber Equipment Setup #1

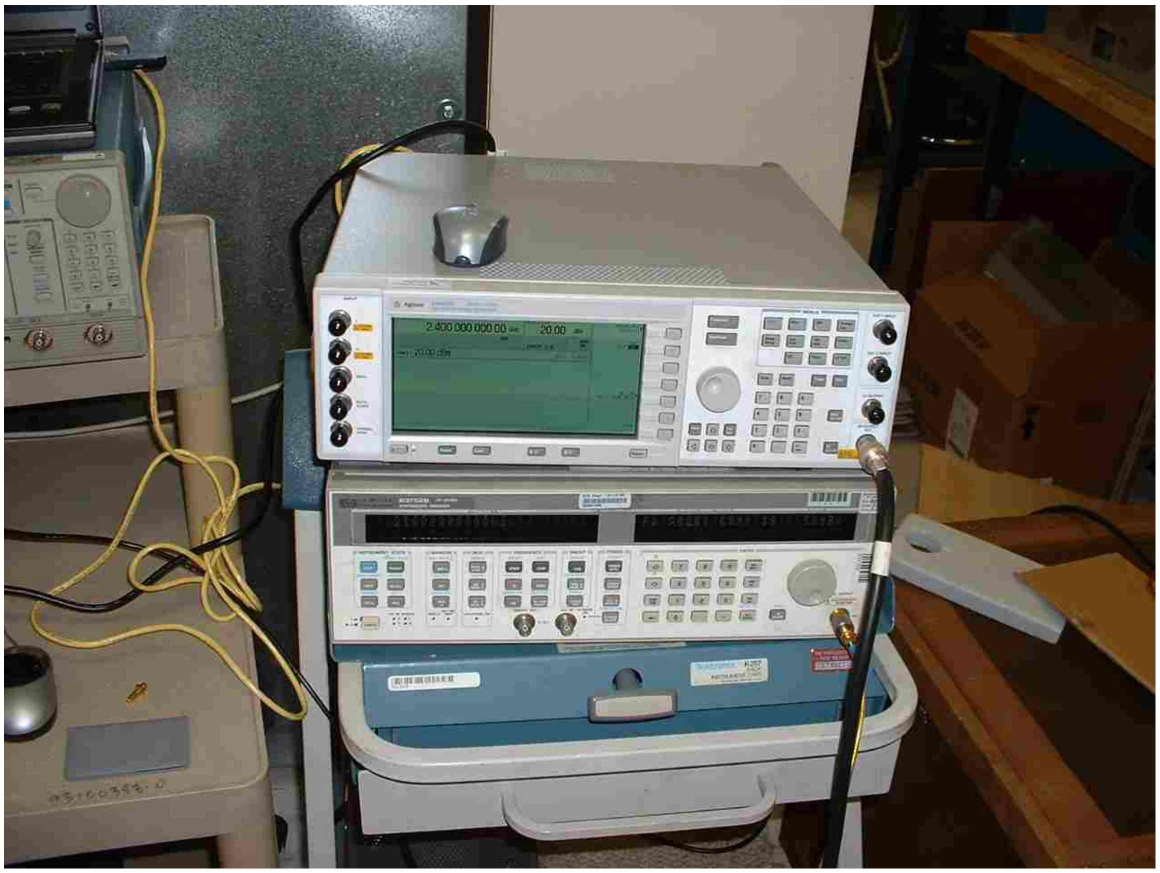


Figure 4.5: Chamber Equipment Setup #2

The second location, where the open air airlink tests were conducted, is located in the quad area between the Electrical Engineering Building and the Centennial Library on the main campus of UNM. In the area one can detect at least 3 WiFi networks, depending on how good their WiFi antenna can receive. These 3 networks, Lobo WiFi (Main Campus WiFi), Electrical and Computer Engineering (ECE) WiFi, and the School of Engineering WiFi, are located at geographically diverse points (ECE WiFi is the closest, Lobo WiFi is the farthest) from this area giving an interesting mix of signals when collecting data. Figure

4.6 shows the equipment setup for this location.

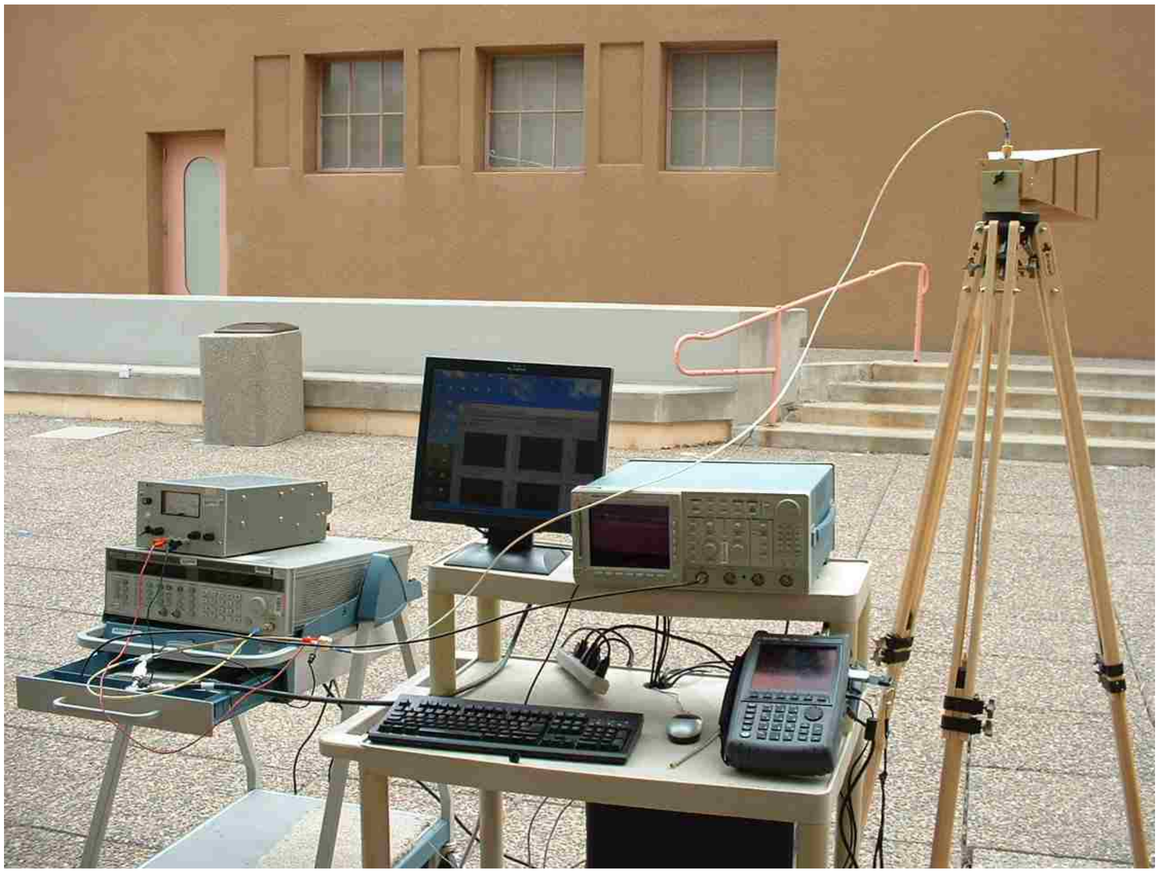


Figure 4.6: Equipment Setup – ECE Quad

For testing at Site One we used the Agilent Arbitrary Waveform Generator, commonly called an “arb”, to generate all of the signals that were transmitted. At Site 2 there were enough signals already in the environment so we used the arb to generate a carrier tone so that we had a marker for the start of the 2.4 GHz ISM band.

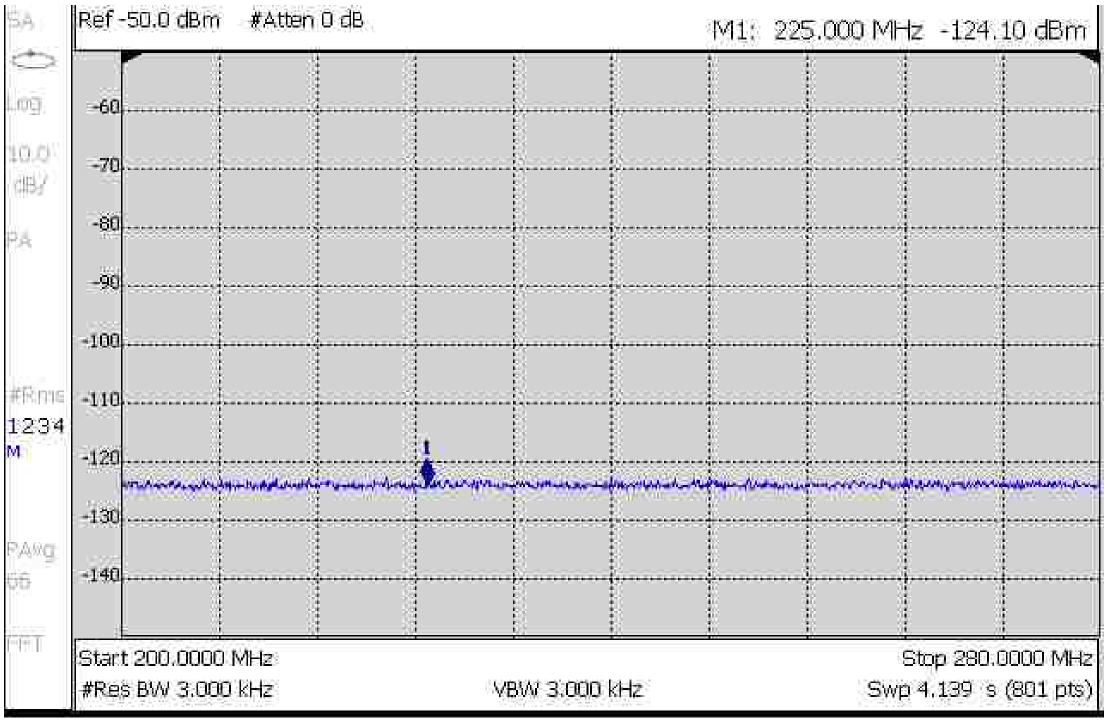
The arb has an RF bandwidth of 6 GHz and a baseband bandwidth of 100 MHz. This capability means that the output of the arb does not have to be upconverted for use at 2.45 GHz. The user defines where they may want the baseband signal to start in frequency space and the arb places the carrier at that frequency point. Unfortunately, the arb also places a tone at the carrier frequency that the user either has to nulled out in some manner or allow to appear in whatever signals that are being generated. For our purposes we allowed the carrier to remain and filtered the tone out with a bandpass filter.

The O'scope that we used has an RF bandwidth of 500 MHz at a maximum sampling rate of 2 Gs/s. The RF bandwidth required a downconversion of the received signal and we chose to place the downconverted signal in the middle of the O'scope's RF bandwidth, hence the choice of the Local Oscillator at 2.2 GHz. In other words, the bandwidth of interest, 2.4 – 2.5 GHz, is downconverted to 200 – 300 MHz for sampling by the O'scope.

4.1.1 RF Background Characterization

To get an idea of the type of RF background that is present in each of the areas a series of spectrum analyzer sweeps were conducted using an Agilent 9912 Handheld Spectrum Analyzer.[19] The figures below show a representative sweep of the RF environment in the 2.4 GHz ISM band.

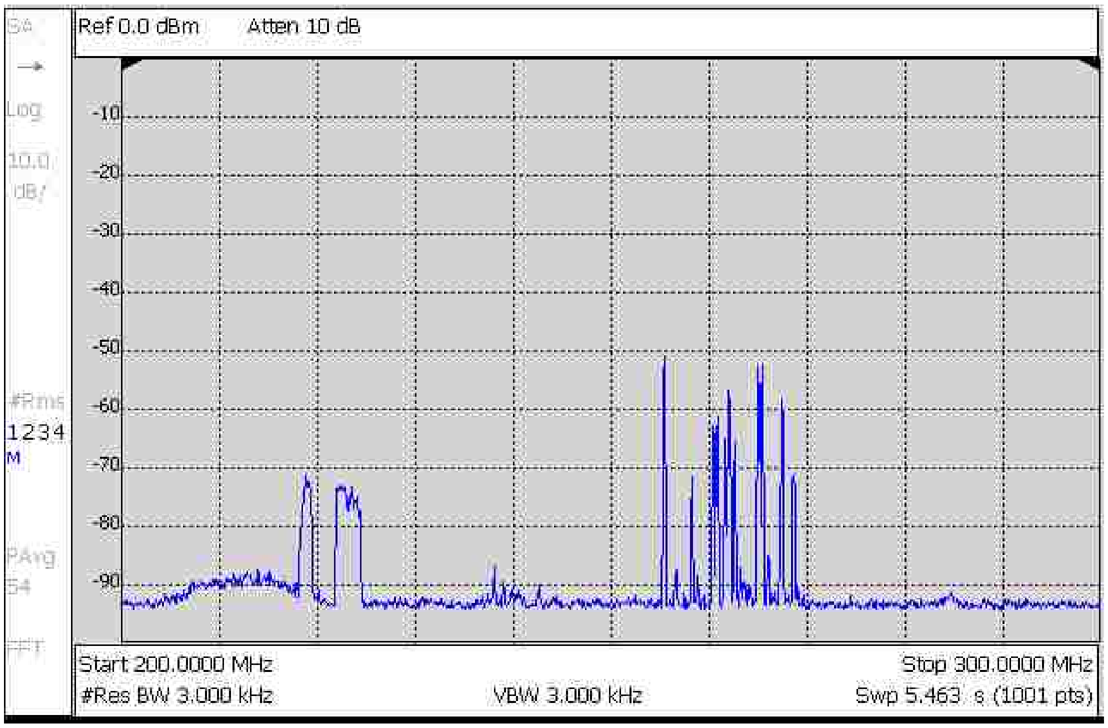
[19] See the Agilent website for more details on this instrument.



Ref -50.0 dBm #Atten 0 dB M1: 225.000 MHz -124.10 dBm

Start 200.0000 MHz Stop 280.0000 MHz
#Res BW 3.000 kHz VBW 3.000 kHz Swp 4.139 s (801 pts)

Figure 4.7: RF Background – Site 1



Ref 0.0 dBm Atten 10 dB

Start 200.0000 MHz Stop 300.0000 MHz
#Res BW 3.000 kHz VBW 3.000 kHz Swp 5.463 s (1001 pts)

Figure 4.8: RF Background – Site 2

5 EXPERIMENTAL RESULTS

5.1 Discussion and Presentation

The SVM algorithms that were developed for this research requires that time samples be available to feed to the algorithms. In order for us to be able to conduct experiments to gather real data some type of sampling system needed to be used to gather these time samples. Since we had neither the funds to buy a custom sampling system nor the personnel to design and construct a custom sampling system (a major research project itself) we chose to use what was available; a Tektronix TDS 644A Digital Storage Oscilloscope. The O'Scope has a maximum sampling rate of 2 Gs/s and is able to collect 2000 samples for a 1 μS time record.

A word about the data preprocessing that we perform on the data. As previously noted the data is downconverted to place the 2.4 GHz ISM band in the 200 – 300 MHz band of the Oscope. The raw data that is downloaded from the Oscope is filtered by a bandpass filter that also removes the carrier that the arb places in the signal being generated. Figure 5.1 shows the magnitude response of the filter.

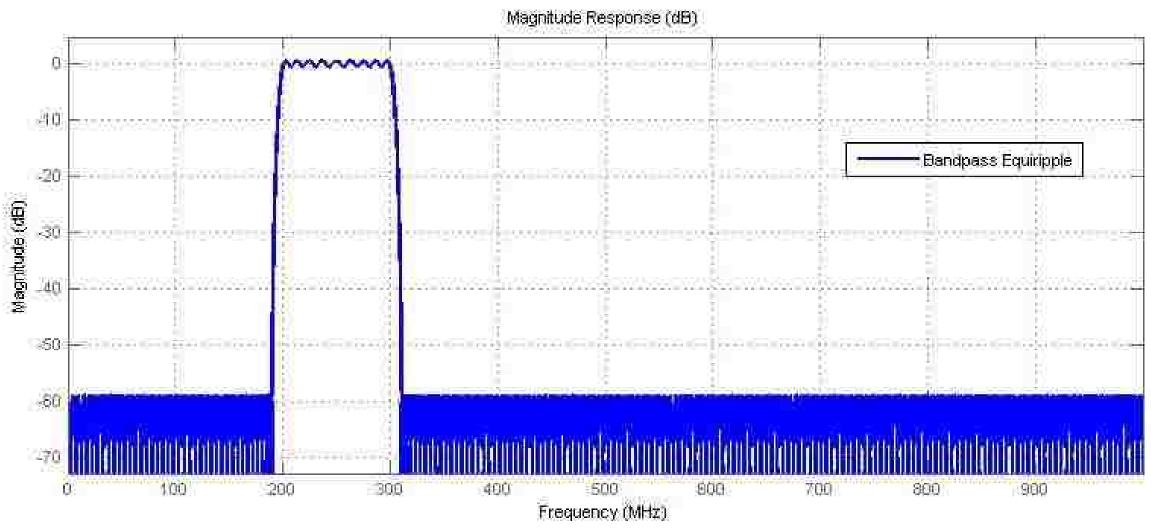


Figure 5.1: Filter – Magnitude Response

Before taking the data collection equipment to the previously defined test locations, we hooked together all the equipment as defined in Figure 4.1. Instead of airlinks we hooked everything together with cable and used the Agilent Waveform generator to generate a single Bluetooth Transmitter. Figure 5.2 shows a single Bluetooth transmitter placed at 2.42 GHz.

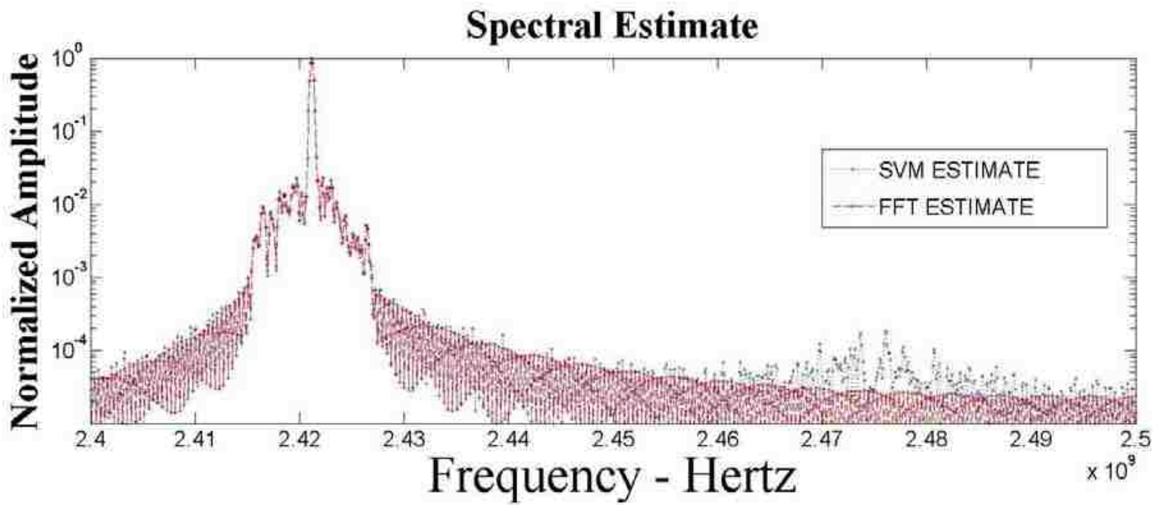


Figure 5.2: Lab Test, Single Bluetooth at 2.42 GHz

For some of these collects we had data available from a handheld spectrum analyzer. Figure 5.3 shows a spectrum analyzer plot of the Bluetooth signal in Figure 5.2.

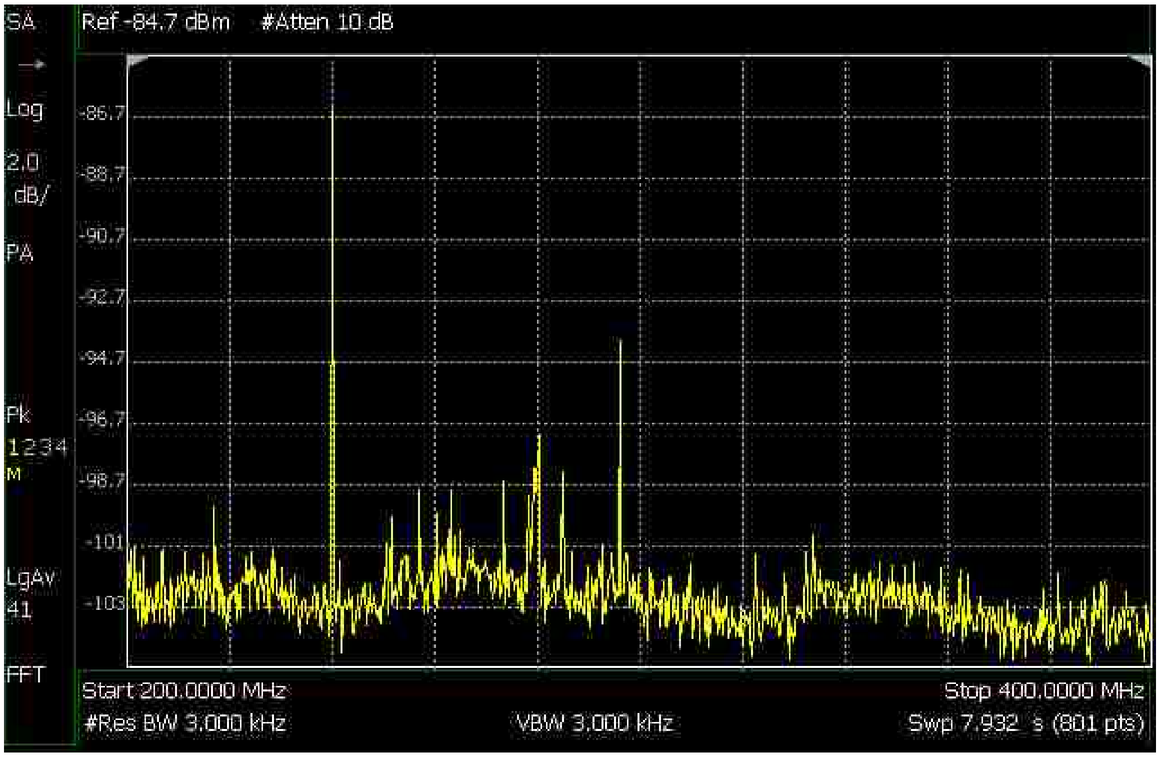


Figure 5.3: Spectrum Analyzer Plot for Figure 5.2

After the lab tests were considered to be satisfactory, i.e. we knew how to do what we needed in terms of arb programming we installed the equipment at Site 1 as shown in Figures 4.2 – 4.5. Figure 5.4 shows the results of transmitting a Piconet that is when we started noticing something happening whose explanation we will save for the end of this chapter.

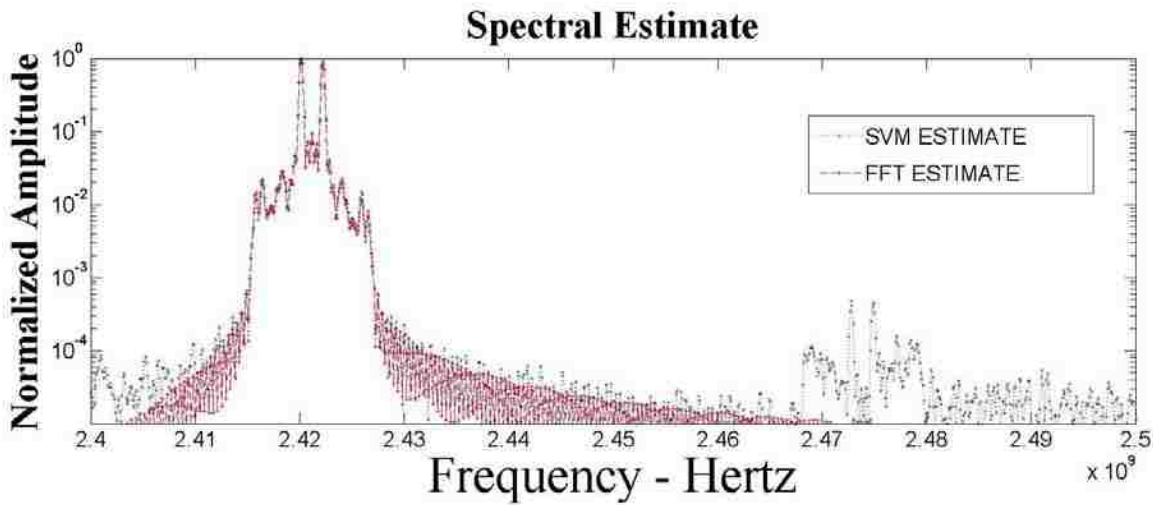


Figure 5.4: Site 1, Piconet

Then we attempted to transmit a single WiFi in the upper end of the spectrum.

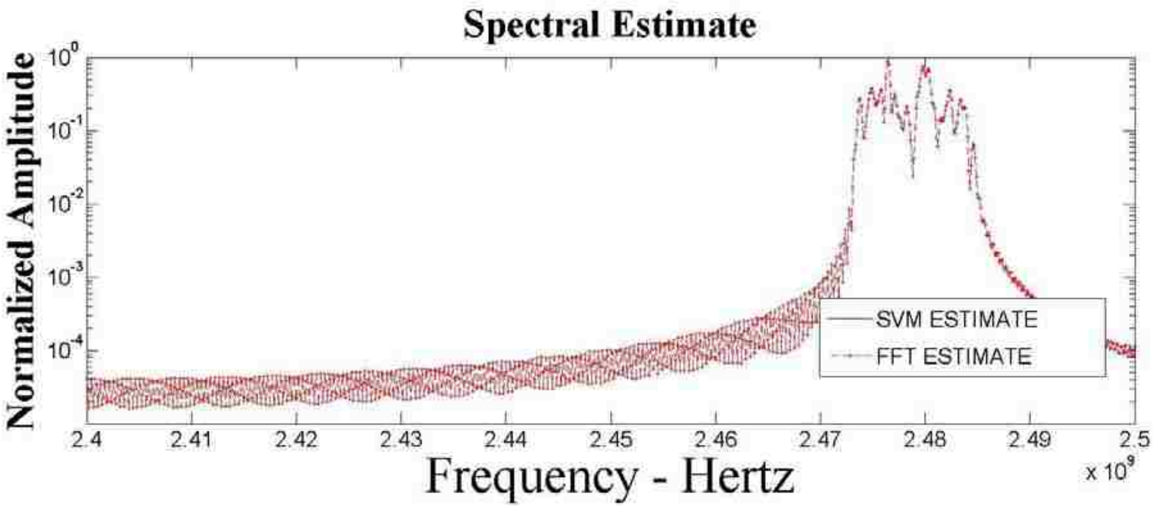


Figure 5.5: Site 1, WiFi

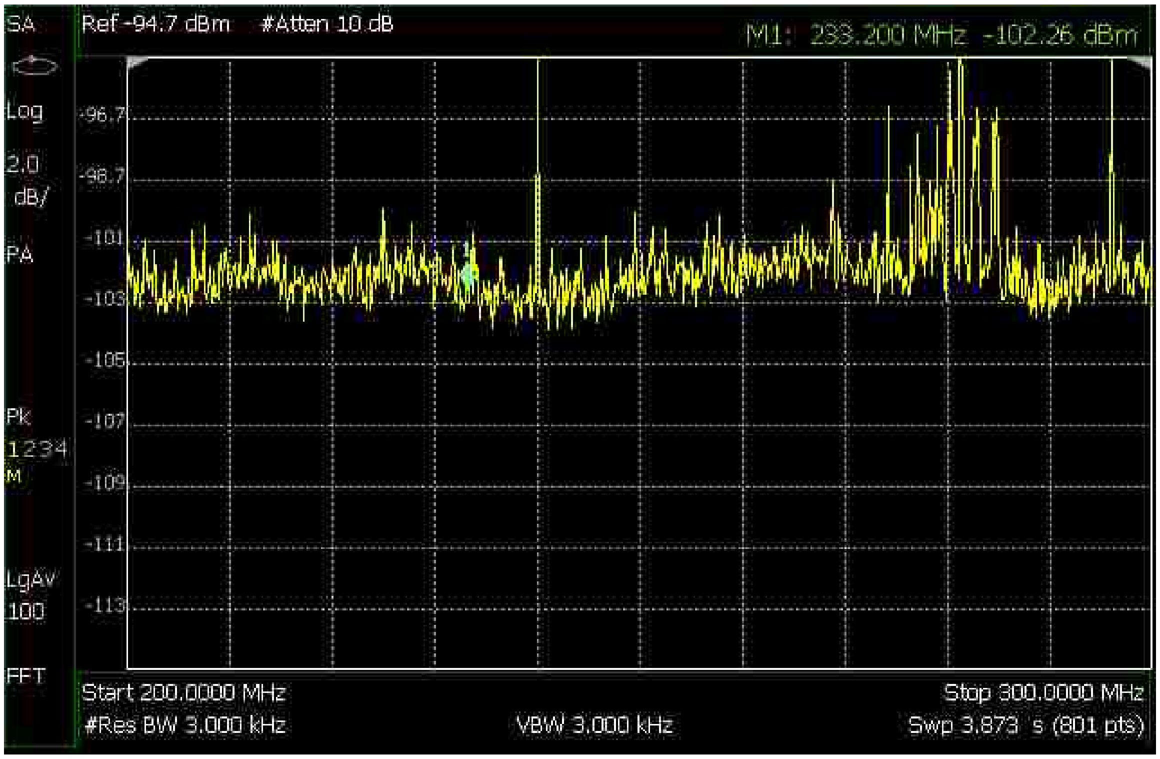


Figure 5.6: Spectrum Analyzer for Figure 5.5

After taking the equipment outside to the area of Site 2, directly to the east of the main entrance to the Electrical and Computer Engineering Building, the following was collected. These collects were taken ~20 minutes apart during the afternoon of a two school days, there are nine of them, stretching over ~2.5 hours.

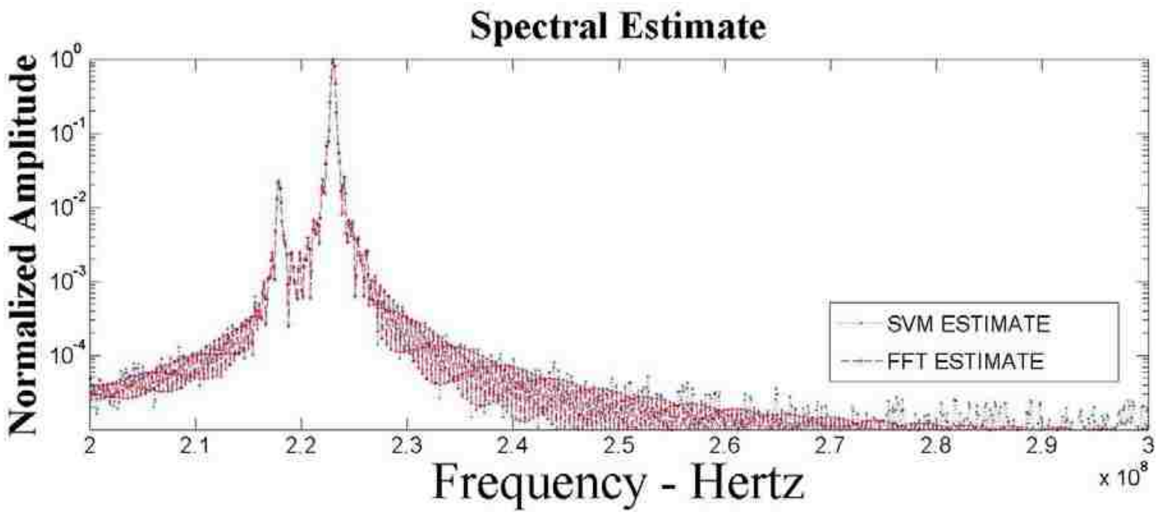
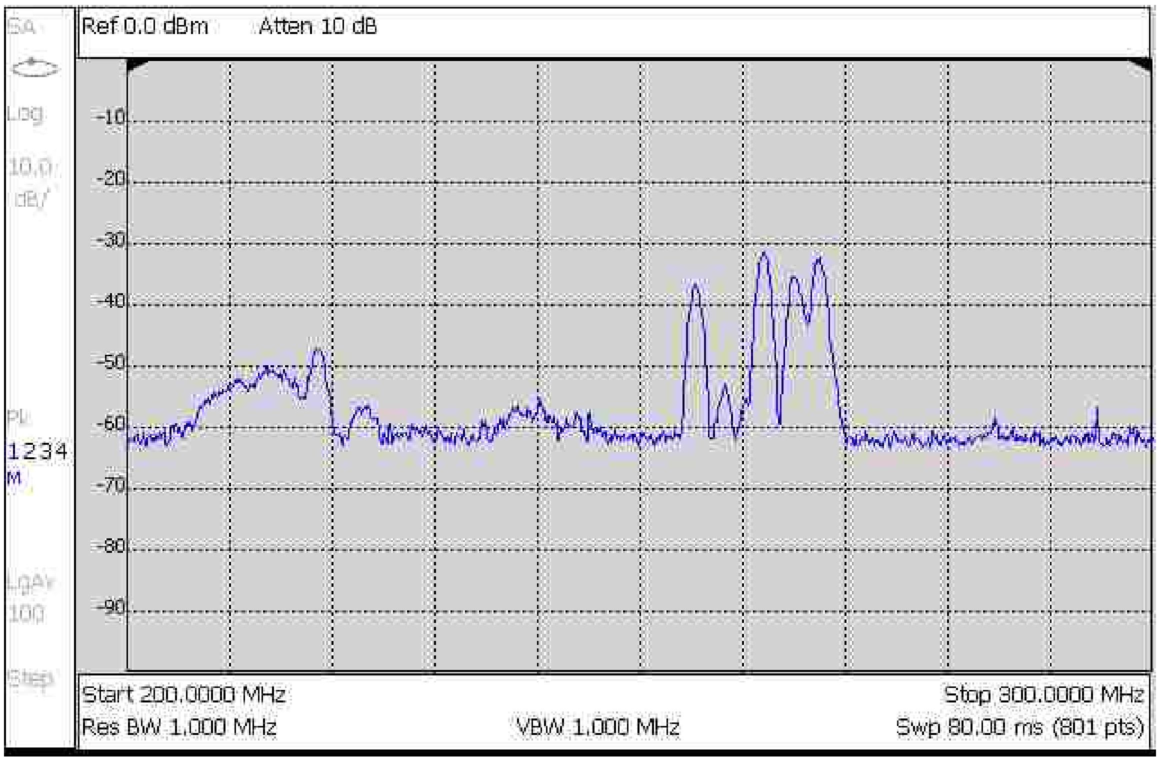


Figure 5.7: ECE1, 6 October 2006 @ 2 PM



Ref 0.0 dBm   Atten 10 dB

Start 200.0000 MHz   Stop 300.0000 MHz

Res BW 1.000 MHz   VBW 1.000 MHz   Swp 80.00 ms (801 pts)

Figure 5.8: Spectrum Analyzer Plot for Figure 5.7

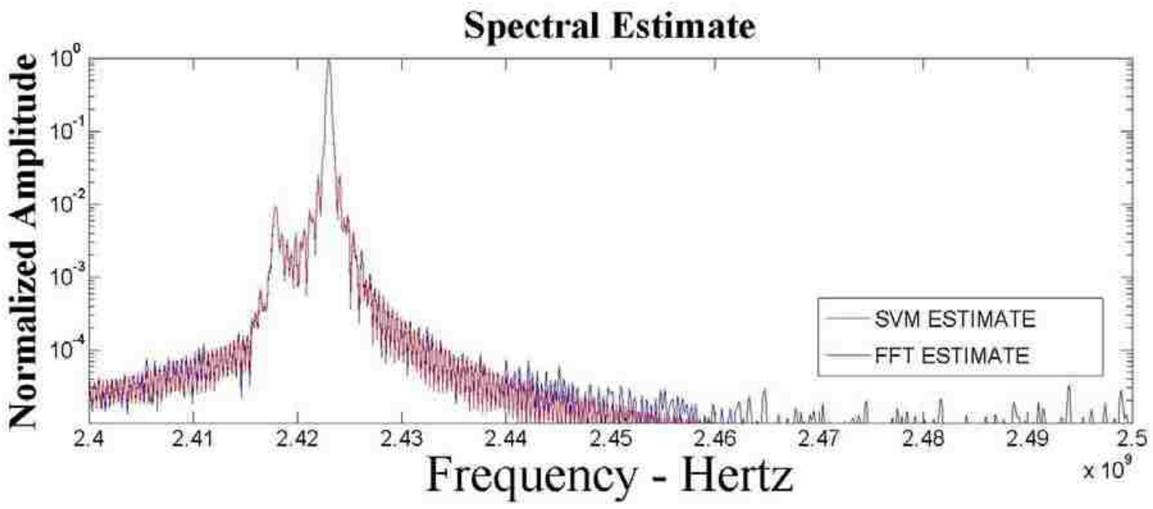


Figure 5.9: ECE2, 6 October 2006 @ 2:20 PM

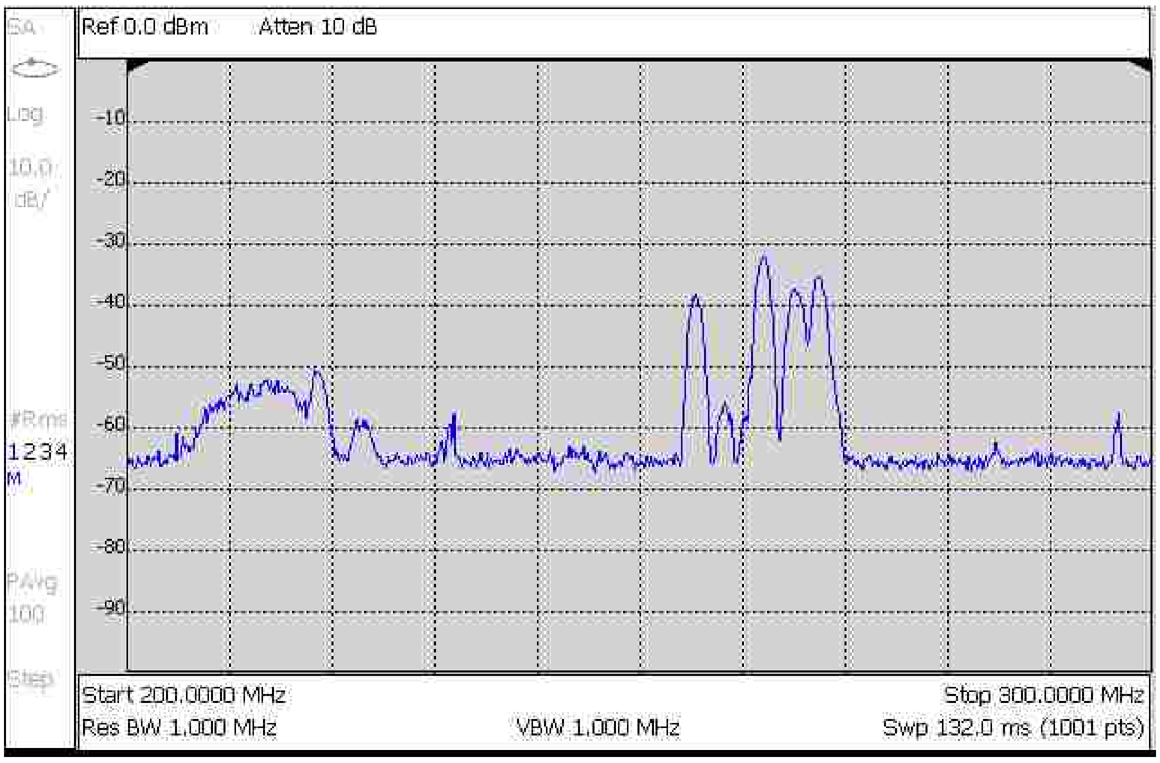


Figure 5.10: Spectrum Analyzer Plot for Figure 5.9

Skipping to the 6th collect which picked up on the afternoon of 7 October 2009 to complete the data collects.

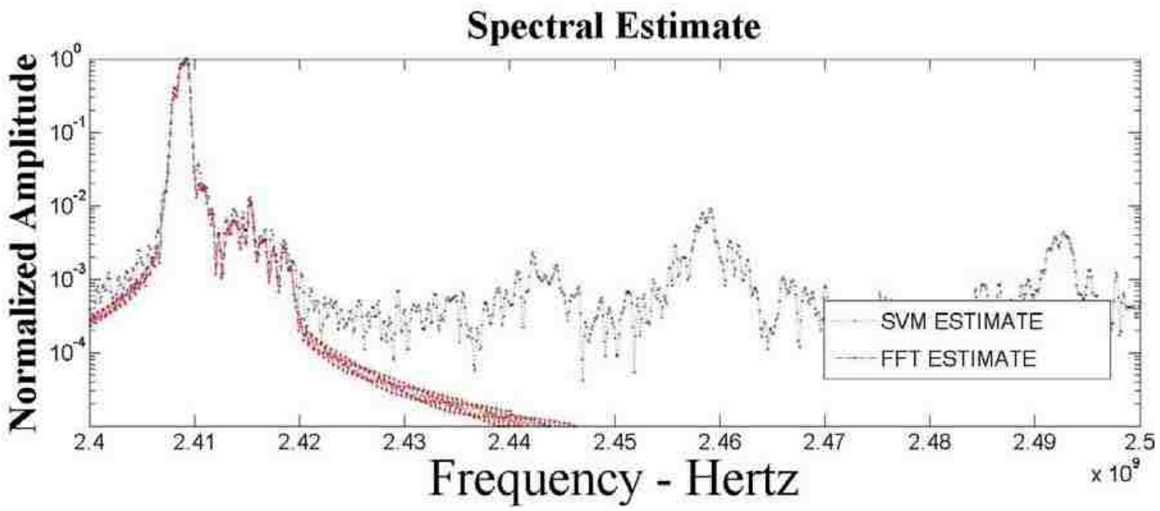
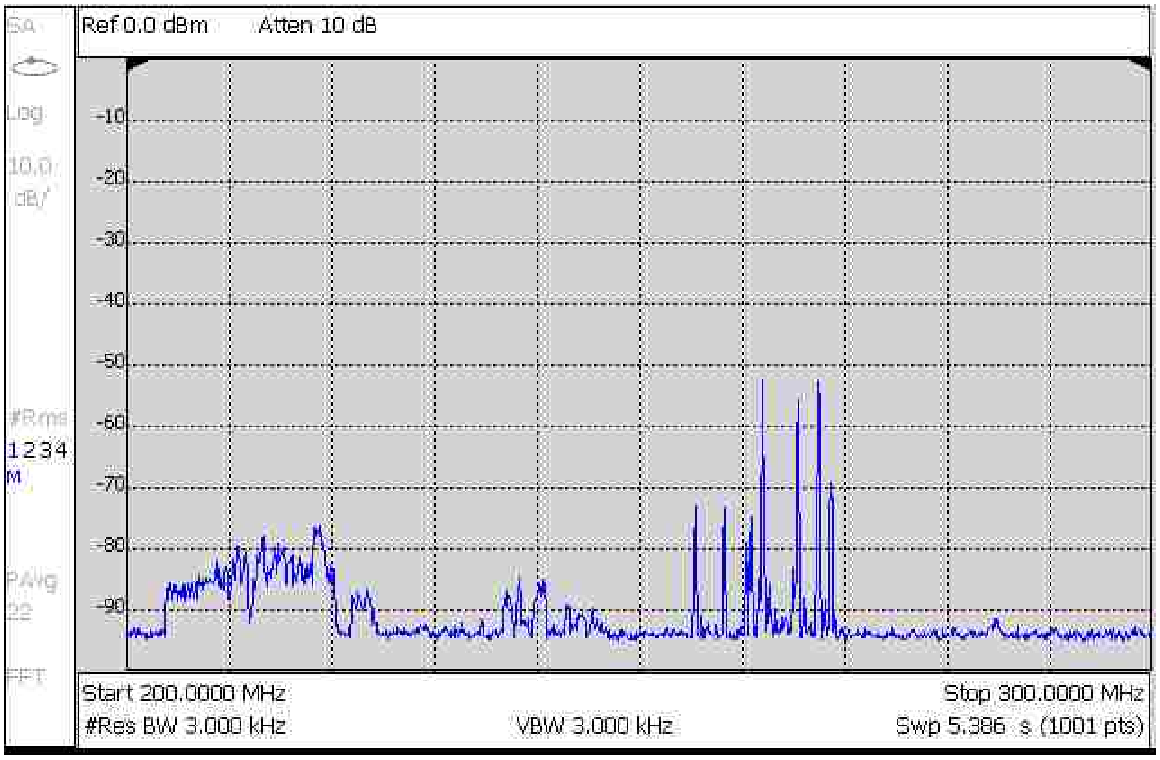


Figure 5.11: ECE6, 7 October 2009 @ 1:30 PM



Ref 0.0 dBm Atten 10 dB

Start 200.0000 MHz Stop 300.0000 MHz

#Res BW 3.000 kHz VBW 3.000 kHz Swp 5.386 s (1001 pts)

Figure 5.12: Spectrum Analyzer Plot for Figure 5.11

As a final data collect we had 4 people stand ~25' away from the receive antenna and activate their Bluetooth-enabled cell phones. Figures 5.13 and 5.14

are the results of this collect.

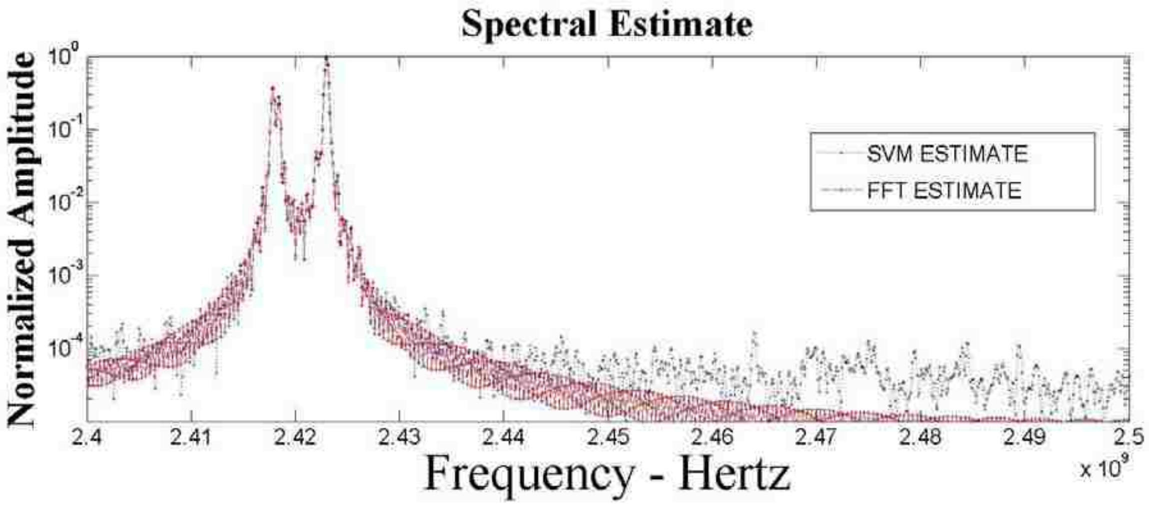


Figure 5.13: Final, 7 October 2009 @ 2:00 PM

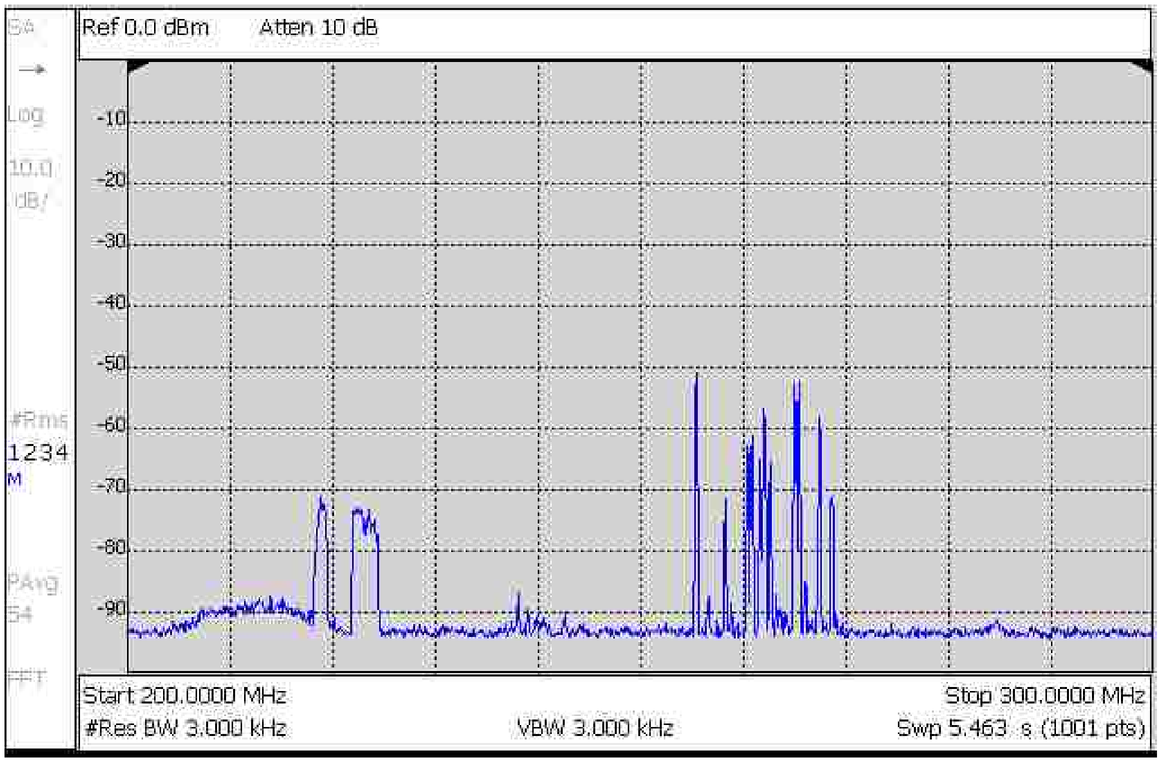


Figure 5.14: Spectrum Analyzer Plot for Figure 5.13

In all of the results for these data collects we can see that there are discrepancies between the information presented by the spectrum analyzer and our estimates of the spectrum information contained in the raw data as collected by the oscilloscope. To explain why these discrepancies exist we need to consider a block diagram of the spectrum analyzer as presented in Figure 5.15.

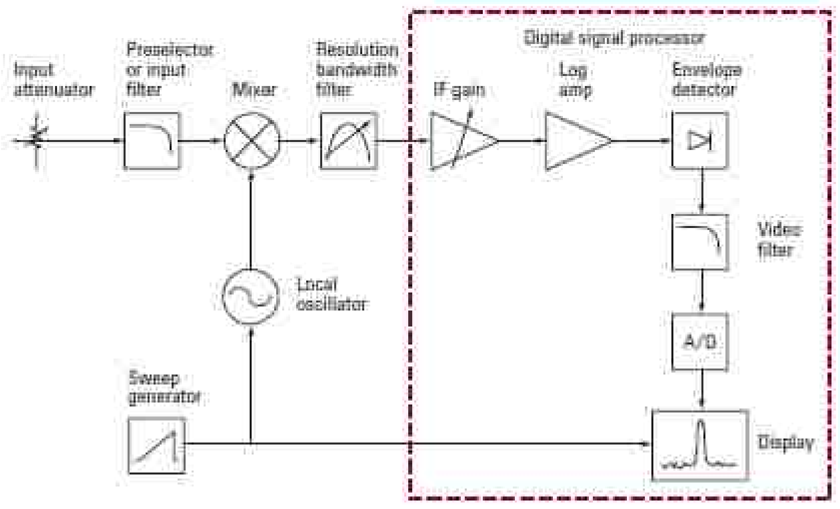


Figure 5.15: Spectrum Analyzer Block Diagram

In the figures that were presented we can see that the "Resolution Bandwidth Filter" and "Video Filter" were set to either 1 MHz or 3 KHz. For our usage the "Preselector or input filter" was set to a band of 200 – 300 MHz. When a sweep is initiated the analyzer will move the center of Resolution Filter so that the filter covers contiguous, but non-overlapping portions of the spectrum to be analyzed. That portion of the band is then fed to the "A/D" for sampling. In the case of the spectrum analyzer that we used the A/D samples at a fixed rate of 30 Ms/s and approximately 4500 samples are generated for each 3 KHz band that the analyzer sweeps through. The analyzer then performs an N-point FFT on those samples to get an estimate of the energy content in a particular resolution bandwidth.

In contrast; the oscilloscope yields a fixed number of samples, 2000, for the entire 500 MHz RF bandwidth of the input. Given the limitations on record length and sampling rate, and adding in the fact that a Bluetooth transmitter hops once every 625 µS, collection of these types of signals by an O’scope like the TDS 644A becomes a probability exercise. This statement is given credibility by the fact that the spectrum analyzer (SA) had to be set to average many signals and the resolution bandwidth of the SA had to be set as low as 3 KHz to adequately represent the spectrum present.

Consider the following two figures. Figures 5.16 and 5.17 present a spectrum analyzer sweep of the RF environment at Site Two. The two spectrum analyzer plots are different because the spectrum analyzer is set to average the spectrum every ~5 seconds. Figure 5.16 is after 2 averages (~10 seconds) and Figure 5.17 is after 14 seconds (~70 seconds)

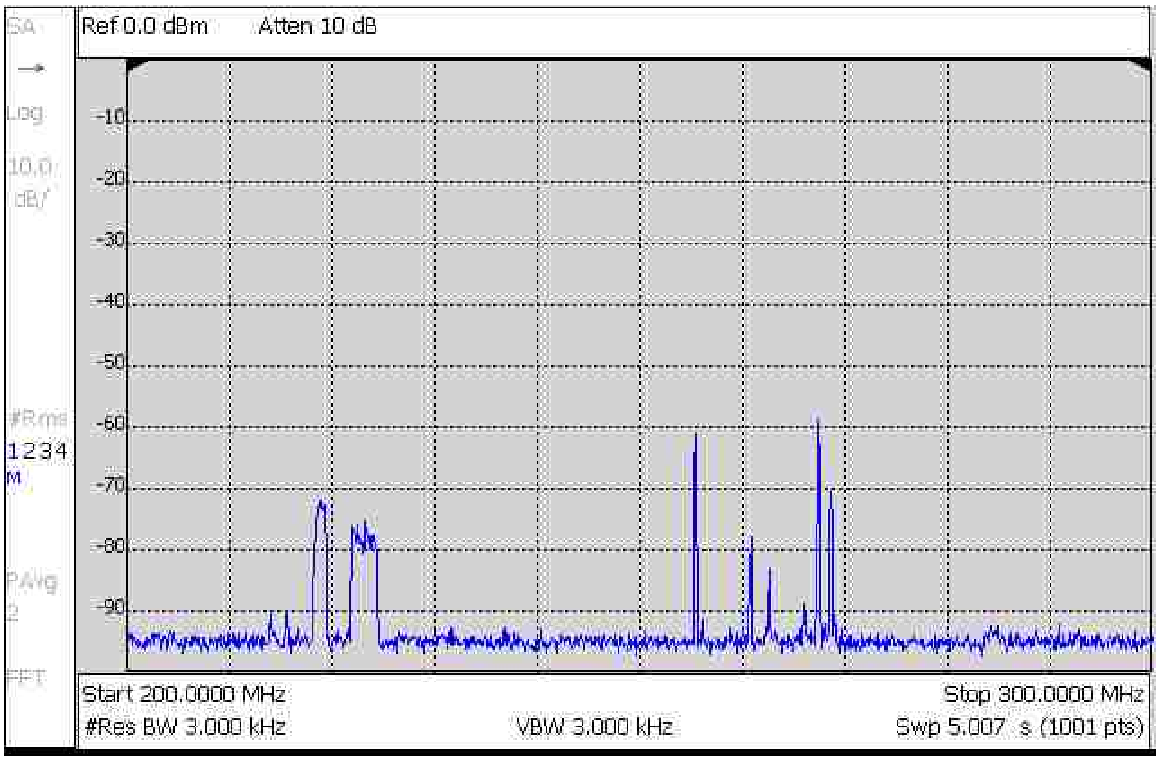
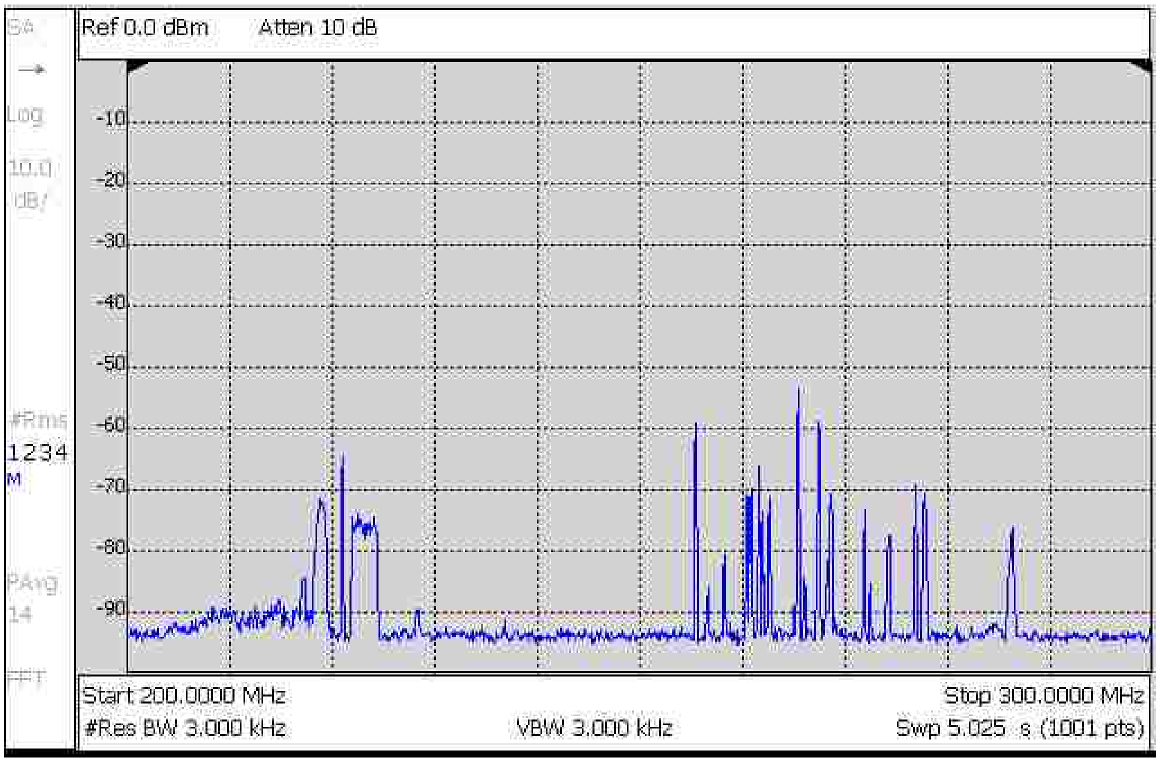


Figure 5.16: Spectrum Analyzer Sweep at Site 2 After 2 Averages



Ref 0.0 dBm Atten 10 dB

Start 200.0000 MHz Stop 300.0000 MHz
#Res BW 3.000 kHz VBW 3.000 kHz Swp 5.025 s (1001 pts)

**Figure 5.17: Spectrum Analyzer Sweep at Site 2
After 14 Averages**

The conclusion that we can draw from these results is that these results point to characteristics of the sampling system that will need to be used for a practical cogntitve radio. These characteristics; 1) Programmable filters banks to select the spectrum of interest and 2) Fast Analog to Digital Conversion that uses little power, will have to fit into a device that the average consumer will require be no bigger than current cell phones and use no more power than current cell phones.

6 CONCLUSIONS

6.1 Summary, Discussions and New Contributions

We have demonstrated, at least theoretically, that machine learning algorithms that use the Support Vector criterion for optimization can work as well if not better than techniques that implement a least squares criterion, most notably the FFT. In most situations that a Cognitive Radio can be expected to operate the algorithms developed in this research have shown to be extremely robust in the presence of non-Gaussian noise.

The two techniques developed, the Non-Parametric algorithm when no knowledge of the channel is assumed and the Parametric algorithms when a-priori knowledge of the channel is available show great promise as an extremely capable alternative to the FFT.

Additionally the development of the kernels for the Parametric algorithm, developed here for the first time, has shown that we also have a methodology to characterize channel impulse response when we have a-priori knowledge of the modulation in use in the channel. This will warrant further investigation along a different research path.

Chapter Two starts by laying out the case for using SVM techniques for the spectrum estimation task by developing a notional model for the types of noise that may be present in the channel. We discuss why we believe the noise environment is non-Gaussian because of the spread spectrum nature, i.e. signals are designed to “hop” around the ISM band in a random manner.

Then in Chapter Two we covered the mathematical underpinnings for SVMs as a classification tool. After laying the groundwork for SVMs as a classification tool we covered the so-called “Kernel Trick” and saw how we can use this trick to turn a non-linear problem into a linear problem, albeit in a higher feature space. We then had the material necessary to cover the topic of Support Vector Regression.

Once we covered Support Vector Regression we were able to turn to the topic of kernels for our non-linear algorithm, i.e. the Parametric algorithm, and developed a new class of kernels that include information on the types of signals that may be present in the channel and information about the channel impulse response. It is our belief that this research is the first research ever at attempting to develop these types of kernels.

Although our simulations assumed a flat channel we now have the groundwork by which we can pursue further research whereby we can gather information about the channel impulse response.

Finally in Chapter Two, we discussed how we can go from the DFT (FFT), and those algorithms dependence on the least squares criteria for optimization, to the Support Vector Regression algorithm and its dependence on the ε-insensitive cost function to provide robustness in the face of non-Gaussian noise.

In Chapter Three we present a thorough treatment of the SVM algorithm versus the FFT algorithm in the presence of various types of noise at different levels of noise power. We investigate many different types of signals that may be present in the 2.4 GHz ISM band. Included in our investigations are: 1.) Single Bluetooth Transmitter, 2.) Single WiFi Transmitter, 3.) Bluetooth Piconet with up to 8 transmitters in the net, and 4.) Combination Bluetooth and WiFi transmitters in the channel. For each of these different scenarios we used the non-Parametric and the Parametric Algorithm.

For the single Bluetooth transmitter the SVM algorithms are clearly superior under all noise scenarios to the FFT algorithm.

For the single WiFi transmitter we note some difficulties with the non-Parametric when comparing the peak of the estimate to the noise floor, see Figure 3.16 as an example, when the noise is benign. However, when the noise gets very heavy we see that the SVM algorithms are superior to the FFT, see Figure 3.23 as an example. We note that the Parametric algorithm is able to extract information about the modulation as in Figure 3.27.

In the case of the Bluetooth Piconet we can see that the algorithms are comparable to the FFT estimates until we get to a very heavy noise environment as in Figure 3.37. Again using the Parametric algorithm yields information about the modulation in light noise environments. However, we note that there is some anomalies present in the estimates that are unexplainable at this time, see Figures 3.40 and 3.43 as examples.

Combinations of Bluetooth and WiFi transmitters are presented using both the non-Parametric and Parametric estimators. The algorithms tend to work very well until we get to a very heavy noise environment as in Figure 3.43.

Finally, we investigated how the estimates are affected by the difference in transmitter power as allowed by the IEEE specifications.

Chapter 4 details our experimental setups, with equipment lists and a notional schematic of our test setups, and then presents some data taken to show the RF background in the environments.

Chapter 5 details the results or our experimental campaign. Suffice it to say that things did not go as planned. Limitations in the use of a Digital Storage Scope, Tektronix TDS 644A, severely restricted the usefulness of the data that we collected.

When we consider that the ultimate goal of this research is to develop a practical Cognitive Radio; one can say that the most interesting result of Chapter 5 is to reveal how tough it will be to develop a time sampling system for a Cognitive Radio that doesn’t require a lot of space, can do its required tasks in the time allotted and not draw too much power from the Cognitive Radio’s batteries.

6.2 Recommendations for Future Research

1.) The issue of a time sampling system will have to be addressed before any further experimental work for this effort can be done,

2.) A comparison of these techniques against the work done with Genetic Algorithms and Neural Networks will need to be done as these three techniques are the main methods for this type of work, 3.) Porting these algorithms to an FPGA and measurements of how fast (efficient) these techniques are is necessary to determine their viability in a practical device,

4.) Investigation of how to meld the information obtained from these spectrum estimations with current work in reconfigurable antennas is necessary to move along the path towards a practical device,

5.) Can the core SVM algorithms be optimized for use in Cognitive Radios? There many different algorithms in play for SVM research, how do we decide what is best for the future of Cognitive Radio. In addition, we will need to evaluate the other SVM algorithms for this type of work before deciding which of the available algorithms is best suited.

REFERENCES

[1] Wireless LAN Association (2001). *Wireless LANs – Poised for Untethered Growth*. Online: http://www.wlana.org/pdf/wlana_industry.pdf

[2] United States Department of Commerce - National Telecommunications and Information Administration (2003, October). *United States Frequency Allocation Chart*. Online: http://www.ntia.doc.gov/osmhome/allochrt.pdf

[3] Federal Communications Commission (2008, July 10). *47CFRPart 15 – Radio Frequency Devices*. Online: http://www.fcc.gov/oet/info/rules/part15/PART15_07-10-08.pdf

[4] M. A. Henry et al., "Chicago Spectrum Occupancy Measurements & Analysis and a Long-Term Studies Proposal," in *Proc. Of TAPAS Conf.*, Boston, MA, Aug. 2006

[5] M. Biggs, A. Henley and T. Clarkson, "Occupancy Analysis of the 2.4 GHz Band," *IEE Proc. Commun.*, vol. 151, no. 5, Oct. 2004

[6] D. A. Roberson, C. S. Hood, J. L. LoCicero and J. T. McDonald, "Spectral Occupance and Interference Studies in Support of Cognitive Radio Technology Deployment," in *1st IEEE WKSP in Networking Technologies for Software Defined Radio Networks*," Reston, VA, Sept. 2006, pp. 26-35

[7] M. A. Henry et al., "Chicago Spectrum Occupancy Measurements & Analysis and a Long-Term Studies Proposal," in *Proc. Of TAPAS Conf.*, Boston, MA, Aug. 2006

[8] United States Federal Communications Commission (2002). *Spectrum Policy Task Force Report*. Online: http://hraunfoss.fcc.gov/edocs_public/attachmatch/DOC-228542A1.pdf

[9] J. Mitola, "Cognitive Radio: An Integrated Agent Architecture for Software Defined Radio," Ph.D. dissertation, KTH, May 1999.

[10] J. Mitola and G. Q. Maguire, "Cognitive Radio: Making Software Radios more Personal," *IEEE Personal Communications*, vol. 6, no. 4, pp. 13-18, Aug. 1999

[11] *IEEE Standard Definitions and Concepts for Dynamic Spectrum Access: Terminology Relating to Emerging Wireless Networks, System Functionality, and Spectrum Management*, IEEE Standard 1900.1, 2008

[12] B. Fette, Ed, *Cognitive Radio Technology.* Burlington, MA: Elsevier, 2006.

[13] ITU-R M.2064, Software Defined Radio in Land Mobile Service.

[14] Federal Communications Commission. *47CFRPart 18 – Industrial, Scientific and Medical Equipment*. Online:http://ecfr.gpoaccess.gov/cgi/t/text/text-idx?type=simple;c=ecfr;cc=ecfr;sid=726ac021b48450f8ecee213aec8496db;idno=47;region=DIV1;q1=part%2018;rgn=div5;view=text;node=47%3A1.0.1.1.16

[15] Joint Tactical Radio System Joint Program Executive Office. *Joint Program Executive Office (JPEO) for Joint Tactical Radio System (JTRS): Organization Overview*. Online: http://jpeojtrs.mil/files/org_info/JPEO_JTRS_Org_Overview.pdf

[16] J.M Chapin and L.E. Doyle, "A Path Forwards for Cognitive Radio Research," in *2nd Int. Conf. Cognitive Radio Oriented Wireless Network and Communications*, Orlando, Fla., 2007, pp. 127-132

[17] C. Clancy et al., "Applications of Machine Learning to Cognitive Radio Networks," *IEEE Wireless Communications*, vol. 14, no. 4, pp.2-3, Aug. 2007

[18] E. Hossain and V.K. Bhargava, Eds, *Cognitive Wireless Communication Networks*. New York: Springer, 2007.

[19] N. Baldo and M. Zorzi, "Learning and adaptation in cognitive radios using neural networks," in *Proceedings of the IEEE CCNC 2008, 2008*, pp. 998-1003.

[20] Z. Zhang and X. Xie, "Intelligent cognitive radio: Research on Learning and Evaluation of CR Based on Neural Networks," in *Proceedings of 5th International Conf. on Information and Communications Technology*, Dec. 2007, pp. 33-37.

[21] Z. Zhang and X. Xie, "Application Research of Evolution in Cognitive Radio based on GA,", in Proceedings of 3rd IEEE Conf. on Industrial Electronics and Applications, June 2008, pp. 1575-1579.

[22] C. Clancy, J. Hecker, E. Stuntebeck and T. O'Shea, "Applications of Machine Learning to Cognitive Radio Networks," *IEEE Wireless Communications*, vol. 14, no. 4, pp. 47-52, Aug. 2007

[23] *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer Specifications*, IEEE STD 802.11 – 2007

[24] *Part 15.1: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Wireless Personal Area Networks (WPANs)*, IEEE STD 802.15.1 – 2005

[25] *Part 15.3: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for High-Rate Wireless Personal Area Networks (WPANs)*, IEEE STD 802.15.3 – 2003

[26] *Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)*, IEEE STD 802.15.4 – 2006

[27] A. Sikora and V. F. Groza, "Coexistence of IEEE 802.15.4 with other Systems in the 2.5 GHz-ISM-Band," in *Instrumentation and Measurement Technology Conf.*, Ottawa, Ontario, Canada, May 2005

[28] J. A. Park et al., "Experiments on Radio Interference Between Wireless LAN and Other Radio Devices on a 2.4 GHz ISM Band," presented at the Vehicular Technology Conference, Orlando, Fla., Oct. 2003

[29] M. Petrova et al., "Interference Measurements on Performance Degradation between Colocated IEEE 802.11 g/n and IEEE 802.15.4 Networks," presented at *6th Int. Conf. Networking*, Saint-Luce, Martinique, April 2007

[30] L. Sydanheimo et al., "Performance issues on the Wireless 2.4 GHz Ism Band in a multisystem environment," *IEEE Trans. Consumer Electronics*, vol. 48, no. 3, pp. 638-643, Aug. 2002

[31] National Institute of Standards and Technology. *Interference in the 2.4 GHz ISM Band: Challenges and Solution*. Online: http://w3.antd.nidt.gov/pubs/golmie.pdf

[32] A. J. Smola and B. Schölkopf, "A tutorial on Support Vector Regression," *Statistics and Computing*, vol 4. No. 3. pp. 199-222, Aug. 2004.

[33] C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery,* vol 2, no. 2, pp. 1-32, 1998.

[34] S. R. Gunn, “Support Vector Machines for Classification and Regression,’ University of Southampton Technical Report, Southampton, England, May 1998

[35] K.-R. Müller, A. J. Smola, G. Rätsch, B. Schlökopf, J. Kohlmorgen and V. Vapnik. *Predicting Time Series with Support Vector Machines.* Online: http://www.svms.org/regularization/MSRS97a.pdf

[36] J. Shawe-Taylor and N. Cristianini, *Support Vector Machines and other kernel-based learning methods*, Cambridge University Press, 2000

[37] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, MIT Press, Cambridge, Massachusetts, 2002

[38] A. J. Smola, et al, Regression Estimation with Support Vector Learning Machines, M.S. Thesis, Dept Phys, Technische Universitat, Munchen Germany, 1996

[39] L. Wang, ed, *Support Vector Machines: Theory and Applications*, Springer, 2005

[40] V. N. Vapnik, *Statistical Learning Theory*, Wiley and Sons. New York, N.Y., 1998

[41] C. Alippi, et al, “Adaptive Sampling for Energy Conservation in Wireless Sensor Networks for Snow Monitoring Applications,” presented at *2007 Int. Conf. Mobile Adhoc and Sensor Systems*, Pisa, Italy, October 2007

[42] V.N. Vapnik, *The Nature of Statistical Learning Theory*, Springer-Verlag, New York, N.Y. 1995

[43] Cortes, C. and Vapnik, V, “Support vector networks,” *Machine Learning*, vol 20, pp 273–297, 1995.

[44] Schölkopf, B., Burges, C. and Vapnik, V. , “Extracting support data for a given task,” in U. M. Fayyad and R. Uthurusamy, editors, *Proceedings, First International Conference on Knowledge Discovery & Data Mining*. AAAI Press, Menlo Park, CA, 1995.

[45] Schölkopf, B., Burges, C. and Vapnik, V, “Incorporating invariances in support vector learning machines,” in C. von der Malsburg, W. von Seelen, J. C. Vorbrüggen, and B. Sendhoff, editors, *Artificial Neural*

Networks — ICANN'96, pages 47 – 52, Berlin, 1996. Springer Lecture Notes in Computer Science, Vol. 1112.

[46] Burges, C. and Schölkopf, B., "Improving the accuracy and speed of support vector learning machines," in M. Mozer, M. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, pp. 375–381, Cambridge, MA, 1997. MIT Press.

[47] Blanz, V., Schölkopf, B., Bülthoff, H., Burges, C., Vapnik, V. and Vetter, T," "Comparison of view–based object recognition algorithms using realistic 3d models," in C. von der Malsburg,W. von Seelen, J. C. Vorbr¨uggen, and B. Sendhoff, editors, *Artificial Neural Networks— ICANN'96*, pages 251 – 256, Berlin, 1996. Springer Lecture Notes in Computer Science, Vol. 1112.

[48] Schmidt, M., "Identifying speaker with support vector networks,"in *Interface '96 Proceedings*, Sydney, 1996.

[49] Osuna, E., Freund, R. and Girosi, F. "An improved training algorithm for support vector machines," in *Proceedings of the 1997 IEEE Workshop on Neural Networks for Signal Processing, Eds. J. Principe, L. Giles, N. Morgan, E. Wilson*, pages 276 – 285, Amelia Island, FL, 1997.

[50] Kay, S. M. and Marple, S. L., Jr., "Spectrum Analysis – A Modern Perspective", *Proceedings of the IEEE*, vol. 69, pp. 1380-1419, November 1981

[51] Jenkins, G.M. and Watts, D.G., *Spectral Analysis and Its Applications*, Holden-Day, Inc. 1968

[52] Geckinii, N.C. and Yavuz, D., Discrete Fourier Transform and Its Applications to Power Spectral Estimation, Elsevier Scientific Publishing Company, Amsterdam, 1983

[53] S.L. Marple, Jr., "A Tutorial Overview of Modern Spectral Overview", presented at ICASSP-89 Conference, Glasgow,Scotland, May 1989

[54] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004 Available online, in its entirety at: http://www.stanford.edu/~boyd/cvxbook/

[55] T. Hotman, B. Schölkopf, and A. J. Smola, "Kernel Methods in Machine Learning", *The Annals of Statistics*, vol. 36, no. 3, pp. 1171-1220, 2008

[56] B. Schölkopf, R. Herbrich, A. J. Smola, and R. Williamson, "A Generalized Representer Theorem", ESPRIT Working Group in Neural and Computational Learning II, NeuroCOLT II Technical Report NC2-TR-2000-81, May 2000

[57] *Part 15.2: Coexistence of Wireless Personal Area Networks with other wireless networks operating in unlicensed frequency bands,* IEEE Std 802.15.2 – 2003

[58] O. Adrisano, et al. "BLUETOOTH AND 802.11 COEXISTENCE: ANALYTICAL PERFORMANCE EVALUATION IN FADING CHANNEL," presented at 13th Int. Sym. Personal, Indoor and Mobile Wireless Communications, Lisbon, Portugal, Sep. 2002

[59] J. Lansford, A. Stephens, R. Nevo, "Wi-Fi (802.11b) and Bluetooth: Enabling Coexistence," *IEEE Network*, vol. 15, issue 3, Sep/Oct 2001, pp. 20-27

[60] N. Golmie, R.E. VanDyck, and A. Soltanian, "Interference of Bluetooth and IEEE802.11: Simulation, Modeling and Performance Evaluation," in *Proceedings of the 4th ACM Workshop on Modeling*, 2008

[61] S. Krishnamoorthy, et al, "Background Interference Measurements at 2.45 GHz in a Hospital Environment," presented at the 1st Student Research Symposium, Blacksburg, VA., May 2002

[62] T.A. Wysocki and H.J. Zepernick, Characterization of the indoor radio propagation channel at 2.4 GHz," *Journal of Telecommunications and Information Technology*, vol. 3, issue 4, pp.84-90, 2000

[63] T. Keller and J. Modelski, "Experimental Results of Testing Interferences in 2.4 GHz ISM Band," presented at the 33rd European Microwave Conference, Munich, Germany, 2003

[64] K.R. Muller, etal, "An Introduction to Kernel-Based Learning Algorithms", IEEE Trans. On Neural Networks, vol. 12, No. 2, pp. 182-202, March 2001

[65] V. Kecman, "*Learning and Soft Computing*", Cambridge, MA, MIT Press, 2001

[66] T.D. Atwood, M. Martinez-Ramón and C.G. Christodoulou, “Robust Support Vector Machine Spectrum Estimation in Cognitive Radio,” presented at the 2009 IEEE International Symposium on Antennas and Propagation and USNC/URSI National Radio Science Meeting, Charleston, SC, June 1-5 2009

[67] Suykens. etal, *Least Squares Support Vector Machines*, Singapore, World Scientific Publishing, 2005 reprint.

[68] T.S. Rappaport, *WIRELESS Communications – Principles and Practices*, Upper Saddle River, NJ, Prentice Hall PTR, 1996

[69] http://www.mathworks.com/matlabcentral/fileexchange/2262

[70] X. Wang, A. Wong, and P-H Ho, “Stochastic Channel Prioritization for Spectrum Sensing in Cognitive Radio”, presented at the 6th IEEE Consumer Communications and Networking Conference, Las Vegas, NV, Jan 10-13 2009

[71] Z. Quan, S. Cui, and A. H. Sayed, “Optimal Linear Cooperation for Spectrum Sensing in Cognitive Radio Networks”, *IEEE JNL of Selected Topics in Signal Processing*, vol. 2 Issue 1, pp 28-40, Feb 2008

[72] Z. Damljanovic, “Cognitive Radio Access Discovery Strategies”, presented at the 6th INTL SYM on Communications Systems, Networks and Digital Signal Processing, Graz, Austria, 23-25 July 2008

[73] T. Yücek and H. Arslan, “Spectrum Characterization for Opportunistic Cognitive Radio Systems”, presented at the 2006 Military Communications Conference, Wash. D.C., 23-25 2006

[74] V. Cherkassay and Y. Ma, “Practical selection of SVM parameters and noise estimation for SVM regression”, *Neural Networks*, vol. 17, issue 1, pp113-126, May 2003

[75] J. T. Kwok, “Linear Dependency between ε and the Input Noise in ε-Support Vector Regression”, presented at the Intl Conf on Artificial Neural Networks, Vienna, Austria, August 21-25 2001

[76] M. Martinez-Ramon, T. Atwood, S. Barbin and C.G. Christodoulou, “Signal Classification with an SVM-FFT Approach for Feature Extraction in Cognitive Radio”, accepted for presentation at the 2009 INTL Microwaves and Optics Conference, Belem, Brazil, November 3-6, 2009

[77] P. J. Huber, “Robust Statistics, A review,” *Ann. Statist.*, p.43, 1972

[78] J. L. RojoAlvarez, M. Martinez-Ramon, M. dePrado Cumplido, A. Artes-Rodriquez and A. R. Figueiras-Vidal, “Support vector method for robust ARMA system indentification,” *IEEE Trans. Signal Process.*, vol. 52, no. 1, pp. 155-164, Jan. 2004.

[79] J. L. RojoAlvarez, M. Martinez-Ramon, M. dePrado Cumplido, A. Artes-Rodriquez and A. R. Figueiras-Vidal, “Support vector machines framework for linear signal processing,” *Signal Process.*, vol. 85, no. 12, pp.2316-2326, Dec. 2005.

APPENDICES

Appendix A: Basic MATLAB Code for “Non-Parametric”

```
function Sigma1_512points_Single %The main program finishes at line 25.
%The rest are aux functions

%% Data Import and PreProcessing
global y1 y2 y
y=evalin('caller','SingleBluetooth');
HopIndex=evalin('caller','Hop_Index.signals.values');
Original=evalin('caller','SignalFFT.signals.values');
Original=fftshift(Original);
sigma=2; % This is the noise standard deviation
y=y+sigma*randn(size(y)); %AWGN added
y1=(10*sigma*(randn(size(y))).*(rand(size(y))>(1-0.1))); %Impulse noise
y2=10*sigma*noise(y); %Block noise.
y=y+y1+y2;

%% Parameters of the SVM. We will use linear kernel.
pars.ker='linear'; %Linear kernel
pars.gamma=10; %This is the numerical regulation parameter
pars.C=0.1;%Capacity Parameter
pars.epsilon=0;%Parameter that defines how cloase we need the estimation
%to get to the input signal
pars.methods={'svm','fft'};
pars.NFFT=512;
W=100; %Window in which we compute the spectrum. This corresponds to one
%symbol in the Bluetooth model

%% Start the estimation
figure(1)
S=welch(y,pars,W,HopIndex,Original); %Welch periodogram that uses svm
and/or fft
%to compute and represent the spectra

%% Computation of frequencies and representation
[w0svm,amps]=find_frequencies(S.m1/max(S.m1),S.w1,20); %Find the highest
%peaks of the SVM estimation
figure(2)
stem(amps,'*')
hold on
[w0fft,amps]=find_frequencies(S.m2/max(S.m2),S.w1,20); %Find the
%highest peaks of the FFT estimation
```

```
stem(amps,'k')
title('Amplitudes of the highest peaks')
hold off
%% Here the auxiliary functions
function [S]=welch(y,pars,B,HopIndex,Original)
%Computes the welch periodogram using the methods described in
pars.methods

Y=buffer(y,B,0); %Here we split the signal in frames of B samples in
%order to average spectra

NFFT=pars.NFFT;

for i=1:size(Y,2)
   ytemp=Y(:,i);

   if sum(strcmp(pars.methods,'svm'))==1
      [s1,a1,b1]=svm_fft_spectrum(pars,ytemp,NFFT); %Compute the spectrum
      s1=fftshift(s1); %shift
      s_svm_fft(i,:)=s1; %Store in a matrix
      S.asvm(i,:)=a1'; %These are the fourier coefficients
      S.bsvm(i,:)=b1';
      m1=mean(abs(s_svm_fft).^2,1); %Compute the average spectrum
      S.m1=m1;

   end
   if sum(strcmp(pars.methods,'fft'))==1 %The same but with FFT
      s2=fftshift(abs(fft(ytemp,NFFT)).^2)';
      s_fft(i,:)=s2;
      m2=mean(s_fft,1);
      S.m2=m2;
      std2=std(s_fft);
      S.stdfft=std2;
   end

   %% Representation
   w=78e6*linspace(0,1,NFFT/2);
   w1=78e6*linspace(0,1,NFFT);
   S.w=w;
   S.w1=w1;
   if HopIndex<40
      plot(w1(1:250),m1(1:250)/max(m1),'b')
      hold on
      plot(w1(1:250),m2(1:250)/max(m2),'r')
      hold off
```

```
    else
        plot(w1(260:512),m1(260:512)/max(m1),'b')
        hold on
        plot(w1(260:512),m2(260:512)/max(m2),'r')
        hold off
    end
    hold on
    plot(w1(260:512),Original(260:512)/max(Original),'g')
    hold off
    title('Spectral Estimate','FontWeight','bold','FontSize',36,...
    'FontName','Times New Roman');
    ylabel('Normalized Amplitude','FontWeight','bold','FontSize',36,...
    'FontName','Times New Roman',...
    'EdgeColor',[1 1 1],...
    'BackgroundColor',[1 1 1]);
    xlabel('Frequency - Hertz','FontSize',48,'FontName','Times New Roman');
    legend1 = legend('SVM ESTIMATE','FFT ESTIMATE','UNCORRUPTED');
    set(legend1,'Position',[0.6716 0.2571 0.2113 0.1546]);

end

function [s_svm_fft,a,b]=svm_fft_spectrum(pars,y,NFFT)
if length(y)<NFFT %Pad with zeros
    y=[y;zeros(NFFT-length(y),1)];
end
if length(y)>NFFT
    y=y(1:NFFT);
end

[L,M]=meshgrid(0:NFFT-1,0:NFFT-1);
ec=sin(2*pi*(L.*M)/NFFT); %These are the functions that reconstruct the signal
es=cos(2*pi*(L.*M)/NFFT);
X=[ec;es];
K=X'*X/NFFT+pars.gamma*eye(NFFT); %with them, we compute the kernel
matrix
%SVM options: regression, precomputed kernel
options=['-s 3 -t 4 -c ' num2str(pars.C) ' -p ' num2str(pars.epsilon)];
model = svmtrain2(y,K, options); % Train the model. Type "svmtrain2"
%without parameters for more information
alpha=model.sv_coef; %Alphas and support vector
SV=X(:,model.SVs+1);
A=SV*alpha;
a=A(1:end/2);
```

```
b=A(end/2+1:end);
s_svm_fft=a'+1j*b';
function r=noise(y)
%This produces the block noise
I1=(rand(size(y))>(1-0.01));
a=find(I1==1);
aT=a';
b=[diff(aT) length(y)-aT(end)];
b=min([b;ones(size(b))*100]);
c=round(rand(1,length(a)).*b);
I2=zeros(size(y));I2(aT+c)=-1;
I=cumsum(I1+I2);
r=I.*randn(size(y));

function [w0,amps,inds]=find_frequencies(s,w,nfreq)
%Find the peaks
s=s(:);
sdiff=diff(s);
ind=[];
w0=zeros(size(sdiff));
for i=1:length(sdiff)-1
    if sign(sdiff(i))~=sign(sdiff(i+1))
        w0(i)=w(i+1);
        ind=[ind i+1];
    end
end


[a,b]=sort(s(ind),'descend');
amps=a(1:nfreq);inds=ind(b(1:nfreq));
w0=w(inds);
```

Appendix B: Basic MATLAB Code for “Parametric”

```
Total=evalin('caller','Sum.signals.values');
y=Total(1:62500);
sigma=1; % This is the noise standard deviation
%y=y+sigma*randn(size(y)); %AWGN added
y=y+(10*sigma*(randn(size(y))).*(rand(size(y))>(1-0.1))); %Impulse noise
%y=y+10*sigma*noise(y); %Streamy noise.
Delay = 1;
DataL = 30; %Data length
R = .5; %Roll-off
Fs = 10; %Sample rate
Fd = 1; %Data rate
PropD = 0;
%% The algorithm in table I (Sparse Primal SVM, this is the first implementation
ever of this new algorithm :)
%% Generation of the autocorrelation kernels
yo=baseband_generator(Delay,DataL,R,Fs,Fd,PropD,0);       % 10MHz pulse
y1=baseband_generator(Delay,DataL*10,R,Fs,Fd*0.1,PropD,0); % 1Mhz pulse

N=min([length(yo),length(y1)]); % Normalization in pulse length
yo=yo(1:N);
y1=y1(1:N);
yo=sqrt(y'*y)*yo/sqrt(yo'*yo);  % Normalization in pulse energy
y1=sqrt(y'*y)*y1/sqrt(y1'*y1);

wt=(0:(length(yo)-1))'*pi*0.1;  % Normalized frequency

%Modulation of the pulses to all the possible frequencies and generation of
%the pulse autocorrelation matrices (the kernels!)
R=[];
for i=1:2:9
   R(:,:,(i+1)/2)=toeplitz((xcorr(real(yo.*exp(1j*wt*i)))));
end
for i=6:15
   R(:,:,i)=toeplitz((xcorr(real(y1.*exp(1j*wt*(i-5))))));
end
r=xcorr(y);                %STEP 1. Computing the correlation

h=zeros(size(R,1),1);h(1)=1; %STEP 2 not implemented. We assume ideal
channel here

TH=[];                     %STEP 3. This will contain the transformed variables Theta
of table I.
```

```
for i=1:size(R,3)
   TH=[TH squeeze(R(:,:,i))*h];
end

% Here we generate the variables of functional in step 4.
K1=TH'*pinv(TH*TH')^1;
K=K1*TH;
p=K1*r;
K=[K+K']/2;          %Trick to avoid lack of simmetry due to numerical inaccuracies
epsilon=0.2; C=100;  %SVM parameters. Choosing epsilon s an issue. C is
trivial.
f=-p'+epsilon*ones(size(p'));
k=0;
K1=zeros(size(p'));
Ku=C*ones(size(p'));

options=['-s 3 -t 4 -c 100 -p 0.2 -j 1'];
model=svmtrain(p,K,options);
a_SVM=model.sv_coef
a_LS=pinv(K)*p;a_LS(find(a_LS<0))=0   % STEP 4b. LS optimization (not
intrinsicly sparse)

%% Representation
S=10*log10(abs(fftshift(fft(xcorr(y),512))));     %Actual spectrum
S_SVM=10*log10(abs(fftshift(fft(TH*a_SVM,512)))); %Parametric SP-SVM
estimation
S_LS=10*log10(abs(fftshift(fft(TH*a_LS,512))));   %Parametric LS estimation

figure(1)
plot(linspace(0,1,256),S(end/2+1:end),linspace(0,1,256),S_SVM(end/2+1:end))
xlabel('Normalized frequency')
ylabel('Power density (dB)')

figure(2)
plot(linspace(0,1,256),S(end/2+1:end),linspace(0,1,256),S_LS(end/2+1:end))
xlabel('Normalized frequency')
ylabel('Power density (dB)')
```

APPENDIX C: ADDITIONAL INFORMATION ON ADDING NOISE TO SIGNALS

After the different signals are generated in the Simulink environment and then imported into the MATLAB workspace, we add three different types of noise to the signals. In addition, we can vary the noise power by the variable “sigma”. The noise added to the signals is divided into three different types: 1.) Average White Gaussian Noise, 2.) Impulsive, and 3.) Block. An analysis of the results of the studies in References [4]-[7], [27]-[31], and [58]-[63] suggests that values of sigma as high as 5 are not unreasonable when we consider the relative differences in transmit powers allowed by the IEEE specifications.

C.1 Additive White Gaussian Noise (AWGN)

AWGN is defined as wideband noise, with a constant spectral density, zero mean and a Gaussian distribution of noise samples. The MATLAB command is: y=y+sigma*randn(size(y)), where “randn” is the MATLAB random number generator, “sigma” is the RMS value of the noise power and “y” is the input signal. Figure C.1 shows a time history of 1000 samples and C.2 shows the mean square spectrum when sigma is set to 1. To show the effects of sigma =10, we present Figures C.3 and C.4 for the time history and mean square spectrum, respectively.

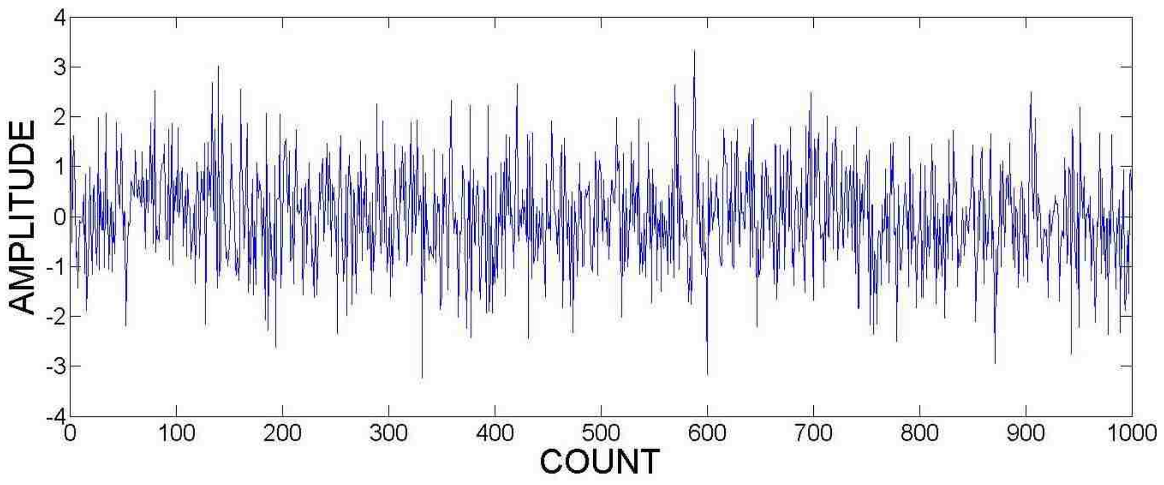


FIGURE C.1: TIME HISTORY, AWGN, SIGMA = 1

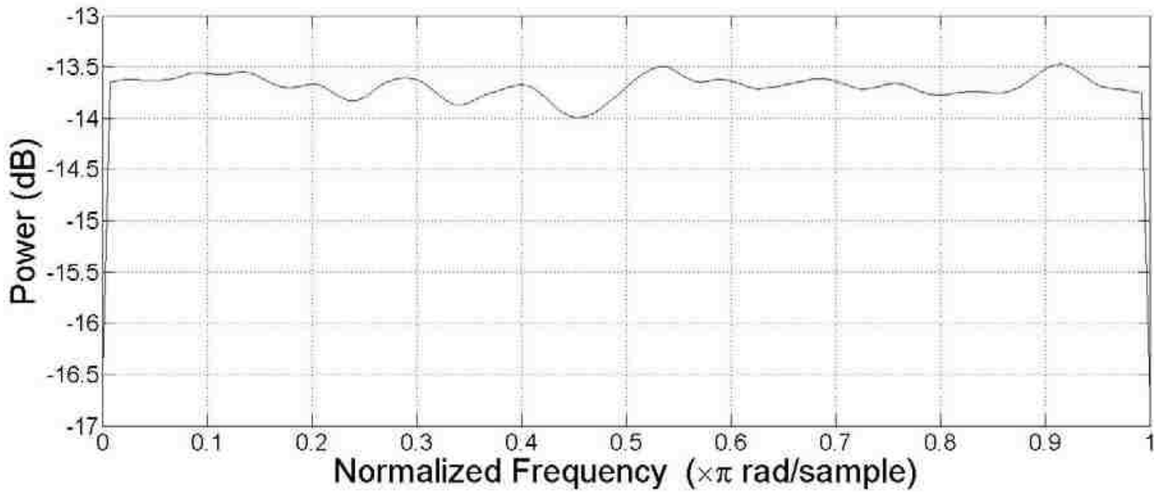


FIGURE C.2: Mean Square Spectrum, AWGN, SIGMA = 1

C.2 Impulsive Noise

Impulsive Noise is defined as random, narrowband impulses of noise. To a WiFi carrier, bandwidth = 22 MHz, the noise presented by a Bluetooth carrier, bandwidth = 1 MHz, can be characterized as impulsive noise. For our simulations we represent impulsive noise by the following code piece: (10*sigma*(randn(size(y))).*(rand(size(y))>(1-0.1))). Notice that we assign a probability of occurrence of 10% or less. Figure C.3 shows a time history of 1000 samples and Figure C.4 shows the mean square spectrum when sigma is set to 1.

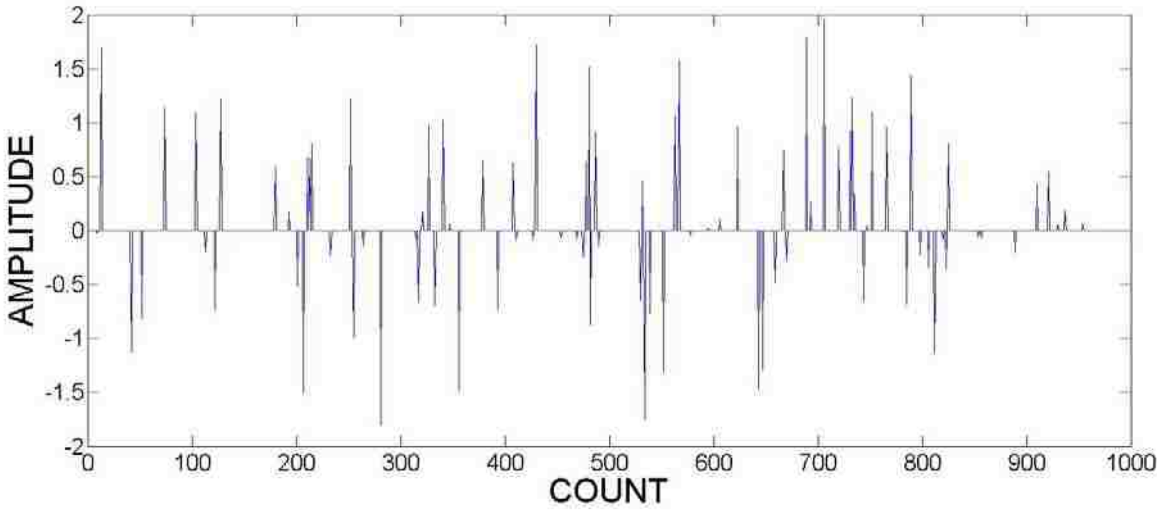


FIGURE C.3: TIME HISTORY, IMPULSIVE, SIGMA = 1

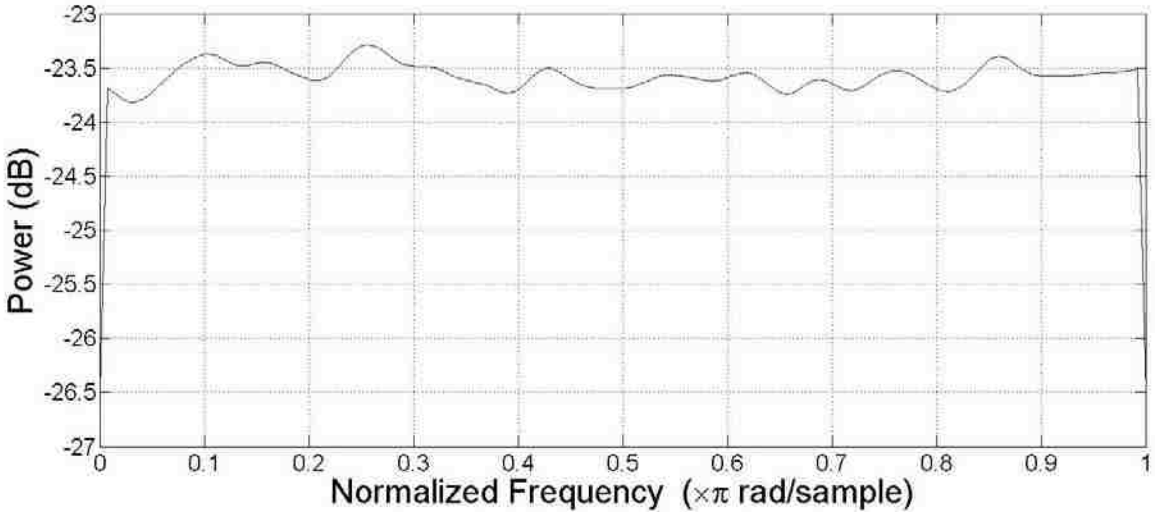


FIGURE C.4: MEAN SQUARE SPECTRUM, IMPULSIVE, SIGMA=1

C.3 Block Noise

Block Noise is defined a broadband noise but not necessarily as broadband as AWGN, e.g. consider the noise contribution that a WiFi transmitter, bandwidth 22 MHz, adds when attempting to identify Bluetooth transmitters, bandwidth 1 MHz. Figures C.6 and C.7 show the time history and mean square spectrum, respectively, for this type of noise.

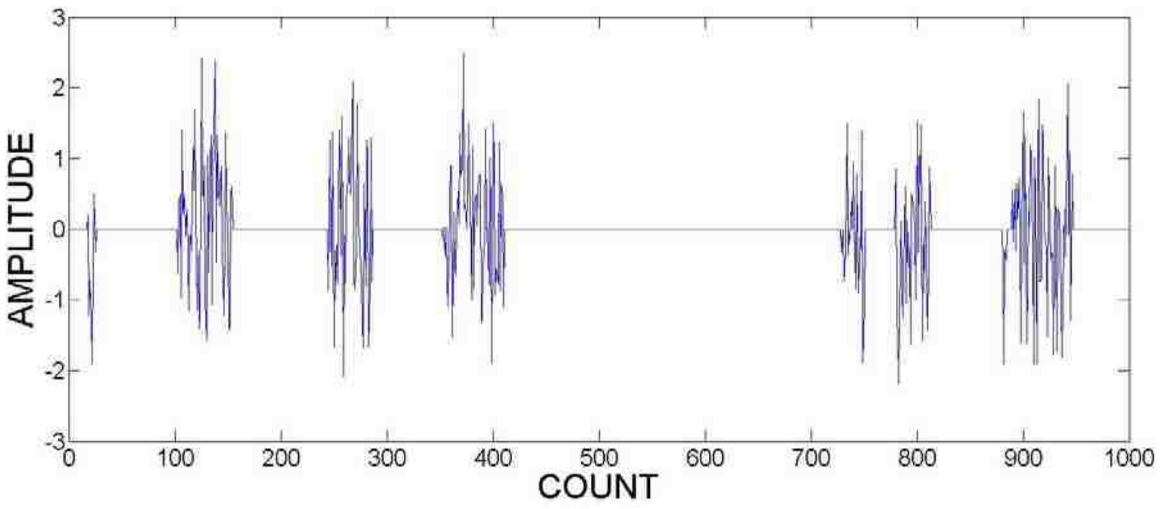


FIGURE C.5: TIME HISTORY, BLOCK, SIGMA = 1

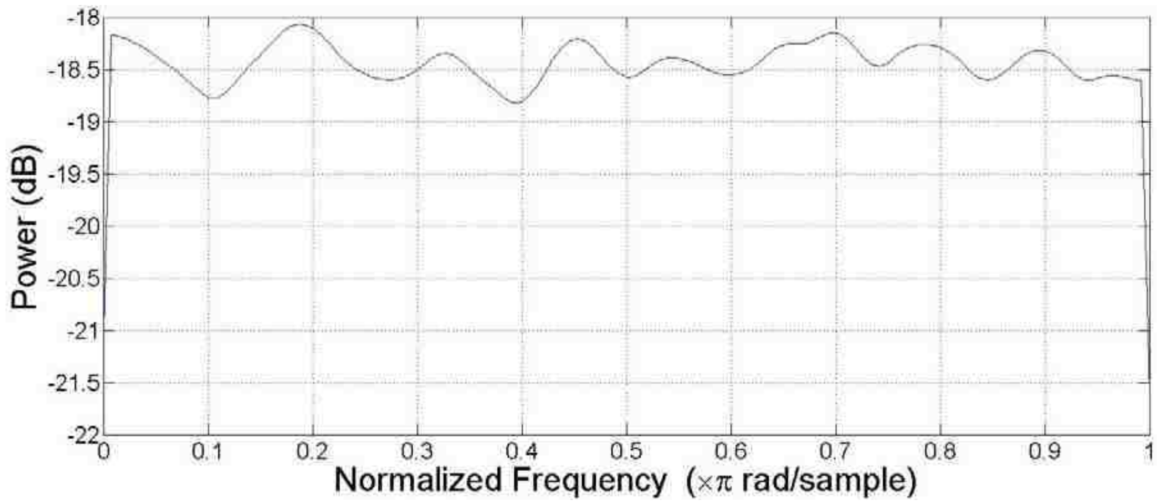


FIGURE C.6: MEAN SQUARE SPECTRUM, BLOCK, SIGMA = 1

All of the simulations that were done for this work assumed that all three of the noise sources would be present in a real channel and therefore were added to the Simulink output to simulate a real world signal. The following figures show the time history and mean square spectrum for sigma =1 and sigma = 5.

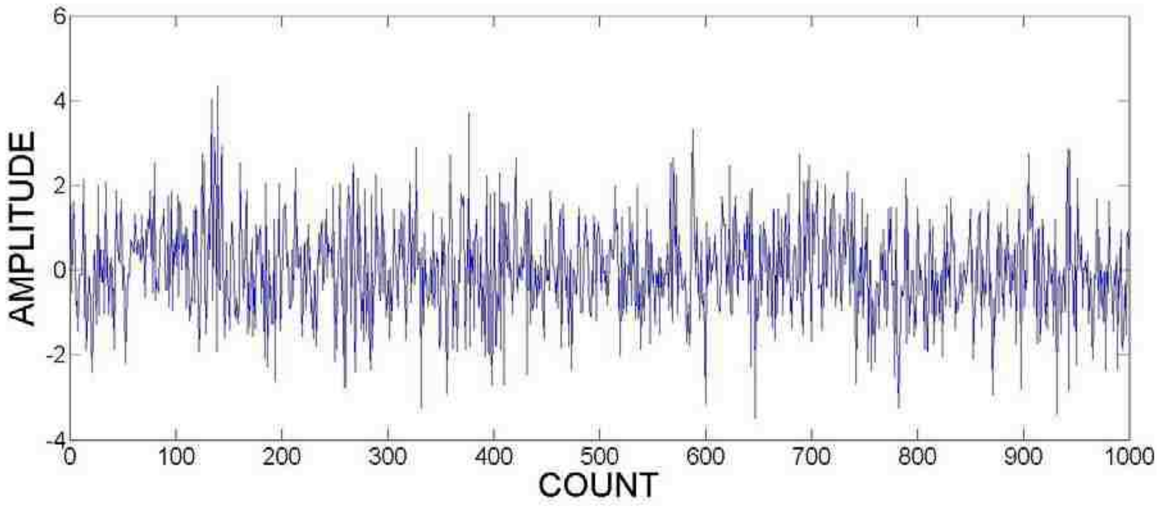


FIGURE C.7: TIME HISTORY, AWGN+IMPULSIVE+BLOCK, SIGMA = 1

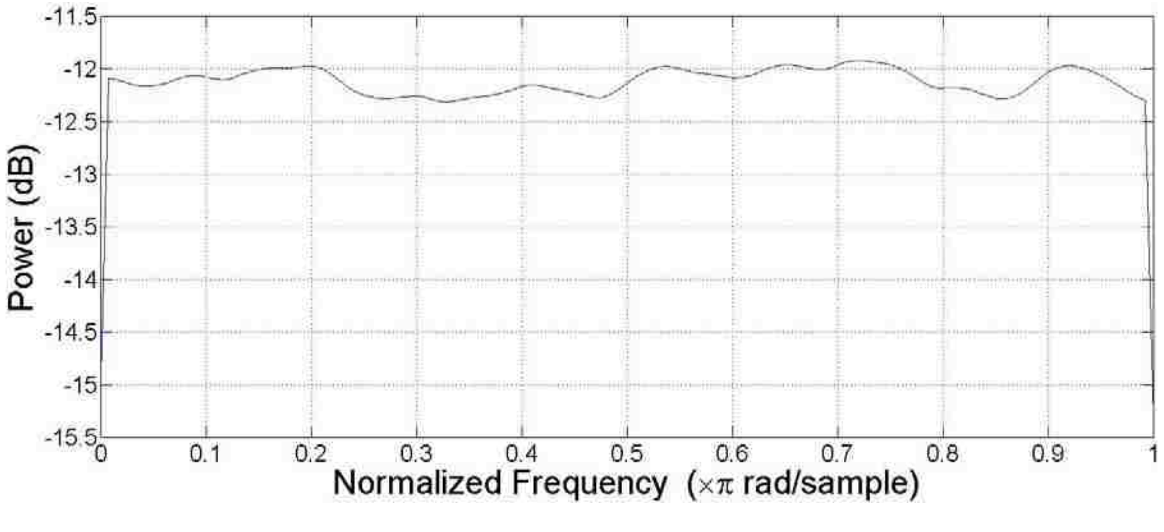


FIGURE C.8: MEAN SQUARE SPECTRUM, AWGN+IMPULSIVE+BLOCK, SIGMA = 1

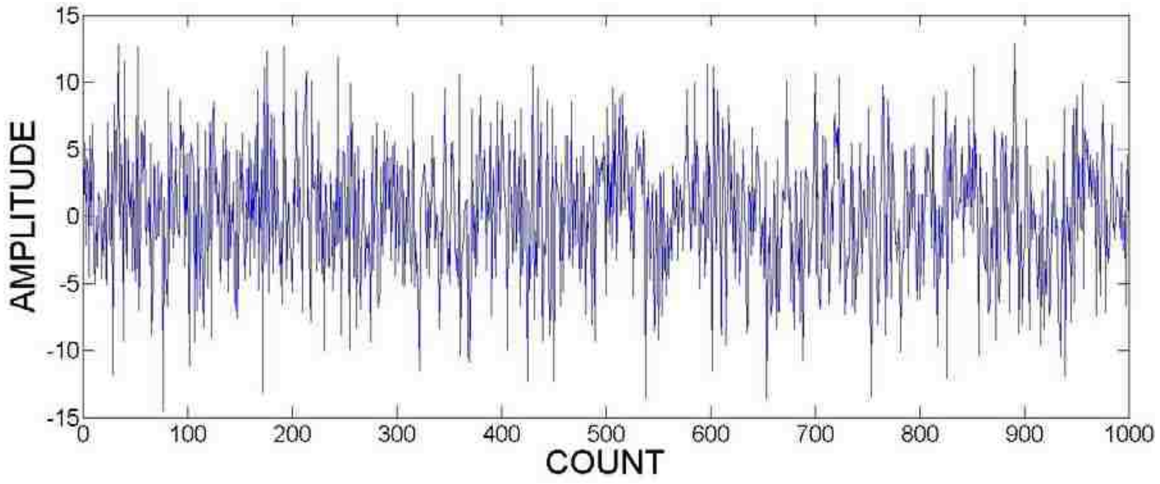


FIGURE C.9: TIME HISTORY, AWGN+IMPULSIVE+BLOCK, SIGMA = 5

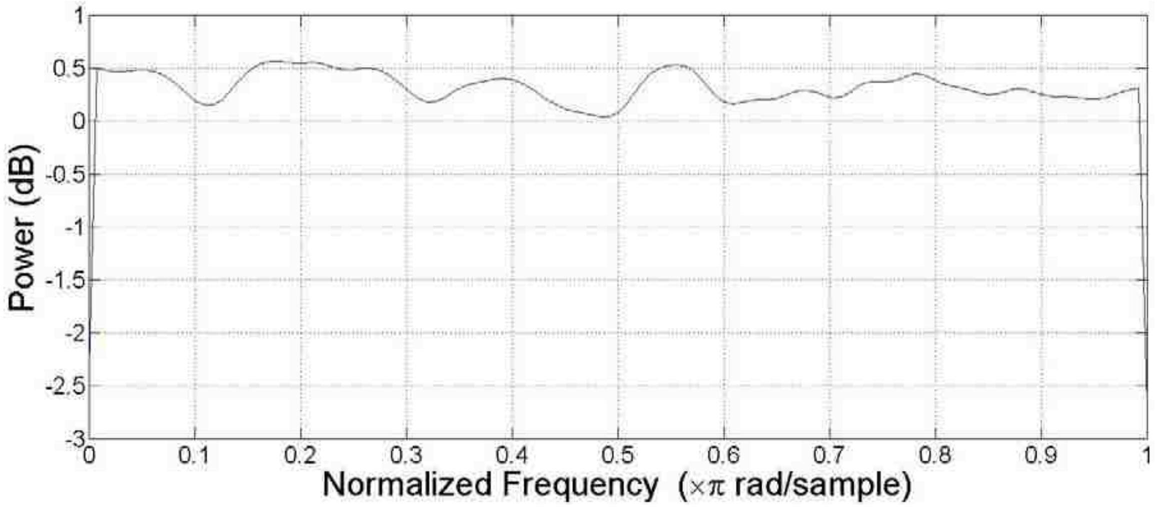


FIGURE C.10: MEAN SQUARE SPECTRUM, AWGN+IMPULSIVE+BLOCK, SIGMA = 5

APPENDIX D – BASIC MATLAB CODE FOR WiFi ESTIMATOR

```
function WiFi_Estimator %The main program finishes at line 25.
%The rest are aux functions
%% Data Import and PreProcessing
global y1 y2 y
y=evalin('caller','WiFi.signals.values');
Original=evalin('caller','WiFi_FFT.signals.values(:,1,1)');
Original=fftshift(Original);
HopIndex=45;
sigma=1; % This is the noise standard deviation
y=y+sigma*randn(size(y)); %AWGN added
y1=(10*sigma*(randn(size(y))).*(rand(size(y))>(1-0.1))); %Impulse noise
y2=10*sigma*noise(y); %Block noise.
y=y+y1+y2;

%% Parameters of the SVM. We will use linear kernel.
pars.ker='linear'; %Linear kernel
pars.gamma=10; %This is the numerical regulation parameter
pars.C=0.1;%Capacity Parameter
pars.epsilon=0;%Parameter that defines how cloase we need the estimation
%to get to the input signal
pars.methods={'svm','fft'};
pars.NFFT=512;
W=100; %Window in which we compute the spectrum.

%% Start the estimation
figure(1)
S=welch(y,pars,W,HopIndex,Original); %Welch periodogram that uses svm
and/or fft
%to compute and represent the spectra

%% Computation of frequencies and representation
[w0svm,amps]=find_frequencies(S.m1/max(S.m1),S.w1,20); %Find the highest
%peaks of the SVM estimation
figure(2)
stem(amps,'*')
hold on
[w0fft,amps]=find_frequencies(S.m2/max(S.m2),S.w1,20); %Find the
%highest peaks of the FFT estimation
stem(amps,'k')
title('Amplitudes of the highest peaks')
hold off
```

```
%% Here the auxiliary functions
function [S]=welch(y,pars,B,HopIndex,Original)
%Computes the welch periodogram using the methods described in
pars.methods

Y=buffer(y,B,0); %Here we split the signal in frames of B samples in
%order to average spectra

NFFT=pars.NFFT;

for i=1:size(Y,2)
   ytemp=Y(:,i);

   if sum(strcmp(pars.methods,'svm'))==1
      [s1,a1,b1]=svm_fft_spectrum(pars,ytemp,NFFT); %Compute the spectrum
      s1=fftshift(s1); %shift
      s_svm_fft(i,:)=s1; %Store in a matrix
      S.asvm(i,:)=a1'; %These are the fourier coefficients
      S.bsvm(i,:)=b1';
      m1=mean(abs(s_svm_fft).^2,1); %Compute the average spectrum
      S.m1=m1;

   end
   if sum(strcmp(pars.methods,'fft'))==1 %The same but with FFT
      s2=fftshift(abs(fft(ytemp,NFFT)).^2)';
      s_fft(i,:)=s2;
      m2=mean(s_fft,1);
      S.m2=m2;
      std2=std(s_fft);
      S.stdfft=std2;
   end

   %% Representation
   w=78e6*linspace(0,1,NFFT/2);
   w1=78e6*linspace(0,1,NFFT);
   S.w=w;
   S.w1=w1;
   if HopIndex<40
      plot(w1(1:250),m1(1:250)/max(m1),'b')
      hold on
      plot(w1(1:250),m2(1:250)/max(m2),'r')
      hold off
   else
      plot(w1(260:512),m1(260:512)/max(m1),'b')
```

```
        hold on
        plot(w1(260:512),m2(260:512)/max(m2),'r')
        hold off
    end
    hold on
    plot(w1(260:512),Original(260:512)/max(Original),'g')
    hold off
    title('Spectral Estimate','FontWeight','bold','FontSize',36,...
    'FontName','Times New Roman');
    ylabel('Normalized Amplitude','FontWeight','bold','FontSize',36,...
    'FontName','Times New Roman',...
    'EdgeColor',[1 1 1],...
    'BackgroundColor',[1 1 1]);
    xlabel('Frequency - Hertz','FontSize',48,'FontName','Times New Roman');
    legend1 = legend('SVM ESTIMATE','FFT ESTIMATE',’ORIGINAL’);
    set(legend1,'Position',[0.6716 0.2571 0.2113 0.1546]);

end

function [s_svm_fft,a,b]=svm_fft_spectrum(pars,y,NFFT)
if length(y)<NFFT %Pad with zeros
    y=[y;zeros(NFFT-length(y),1)];
end
if length(y)>NFFT
    y=y(1:NFFT);
end

[L,M]=meshgrid(0:NFFT-1,0:NFFT-1);
ec=sin(2*pi*(L.*M)/NFFT); %These are the functions that reconstruct the signal
es=cos(2*pi*(L.*M)/NFFT);
X=[ec;es];
K=X'*X/NFFT+pars.gamma*eye(NFFT); %with them, we compute the kernel
matrix
%SVM options: regression, precomputed kernel
options=['-s 3 -t 4 -c ' num2str(pars.C) ' -p ' num2str(pars.epsilon)];
model = svmtrain2(y,K, options); % Train the model. Type "svmtrain2"
%without parameters for more information
alpha=model.sv_coef; %Alphas and support vector
SV=X(:,model.SVs+1);
A=SV*alpha;
a=A(1:end/2);
b=A(end/2+1:end);
s_svm_fft=a'+1j*b';
```

```
function r=noise(y)
%This produces the block noise
I1=(rand(size(y))>(1-0.01));
a=find(I1==1);
aT=a';
b=[diff(aT) length(y)-aT(end)];
b=min([b;ones(size(b))*100]);
c=round(rand(1,length(a)).*b);
I2=zeros(size(y));I2(aT+c)=-1;
I=cumsum(I1+I2);
r=I.*randn(size(y));

function [w0,amps,inds]=find_frequencies(s,w,nfreq)
%Find the peaks
s=s(:);
sdiff=diff(s);
ind=[];
w0=zeros(size(sdiff));
for i=1:length(sdiff)-1
   if sign(sdiff(i))~=sign(sdiff(i+1))
      w0(i)=w(i+1);
      ind=[ind i+1];
   end
end


[a,b]=sort(s(ind),'descend');
amps=a(1:nfreq);inds=ind(b(1:nfreq));
w0=w(inds);
```

APPENDIX E
Further Parametric Estimates for a Single Bluetooth Transmitter

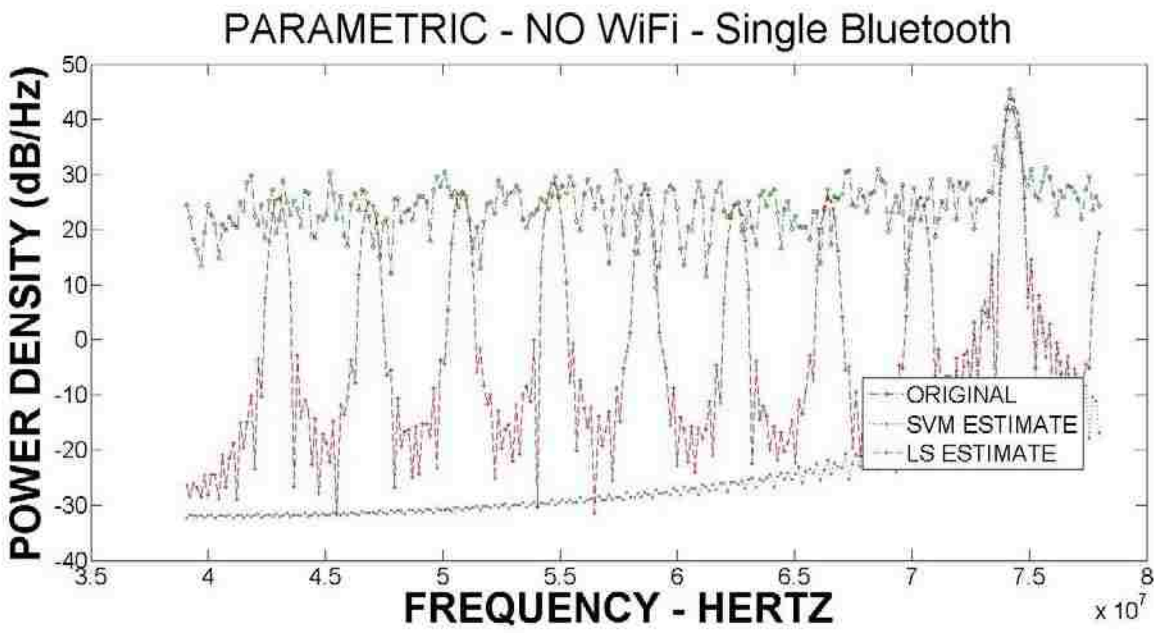


Figure E.1: Parametric Estimator, Channel 70, AWGN, Sigma = 1

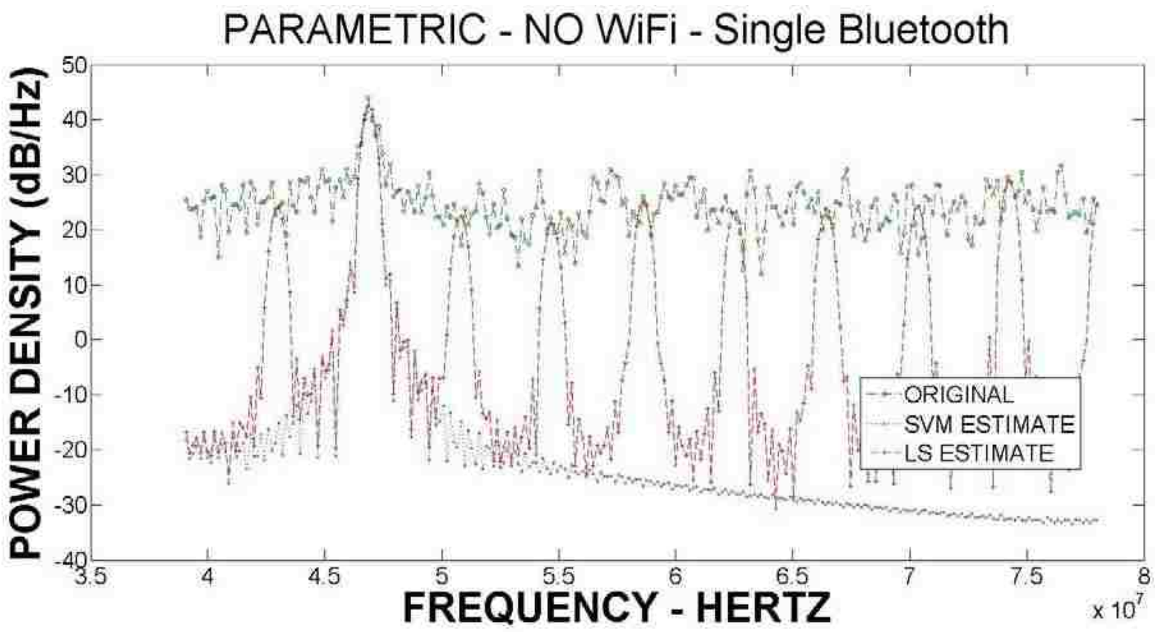


Figure E.2: Parametric Estimator, Channel 42, A+I, Sigma = 1

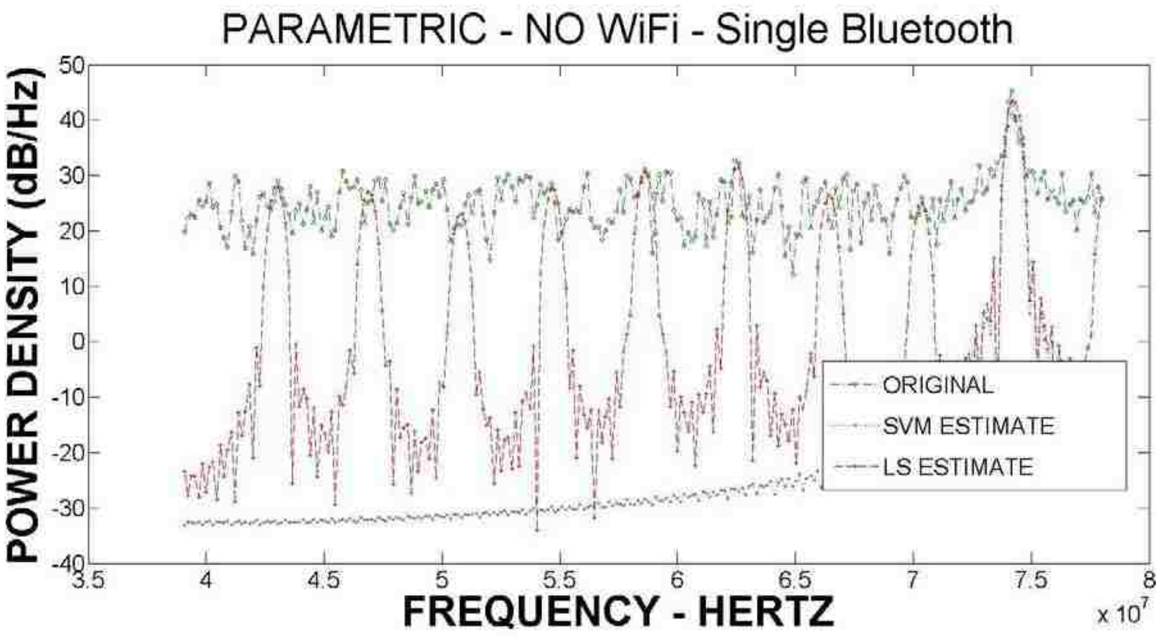


Figure E.3: Parametric Estimator, Channel 70, A+I, Sigma = 1

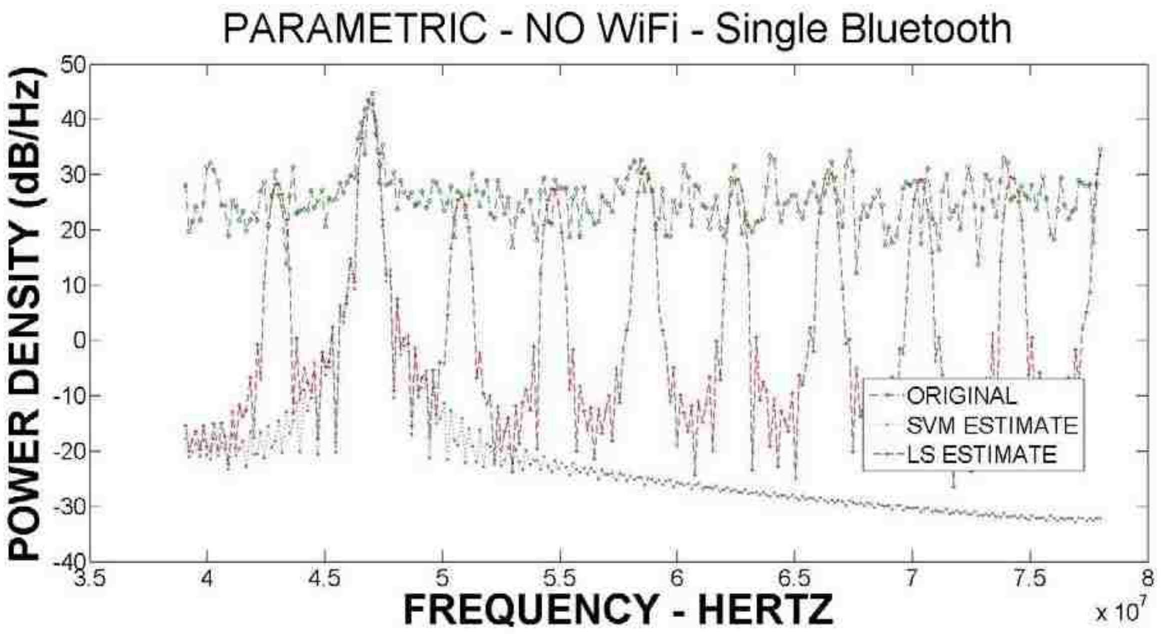


Figure E.4: Parametric Estimator, Channel 42, A+B, Sigma = 1

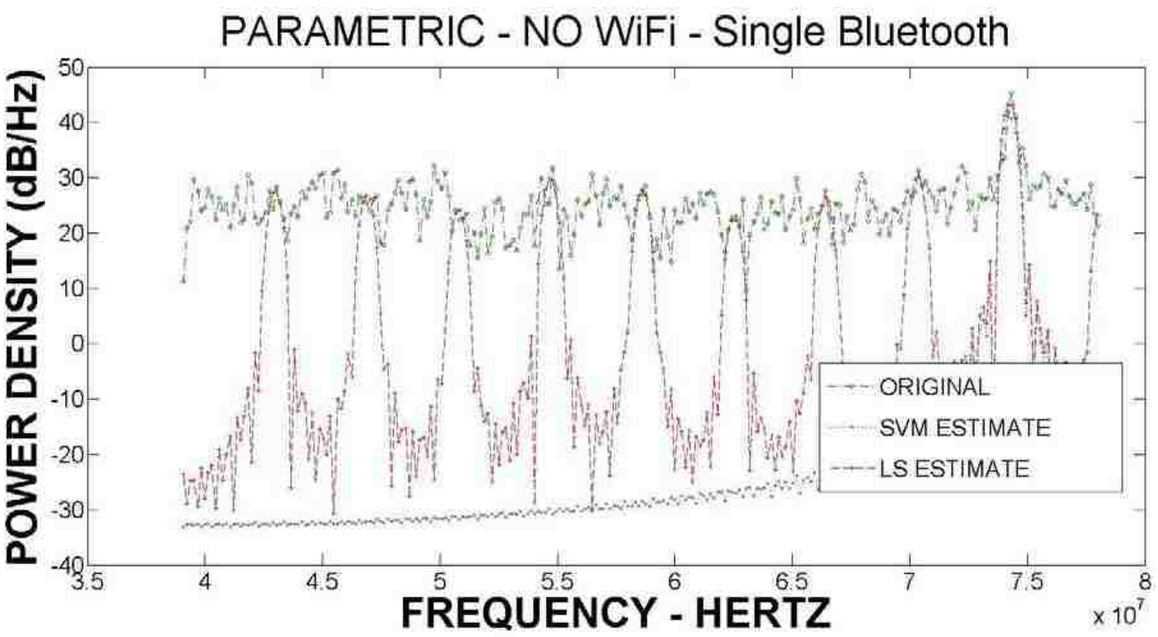


Figure E.5: Parametric Estimator, Channel 69, A+B, Sigma = 1

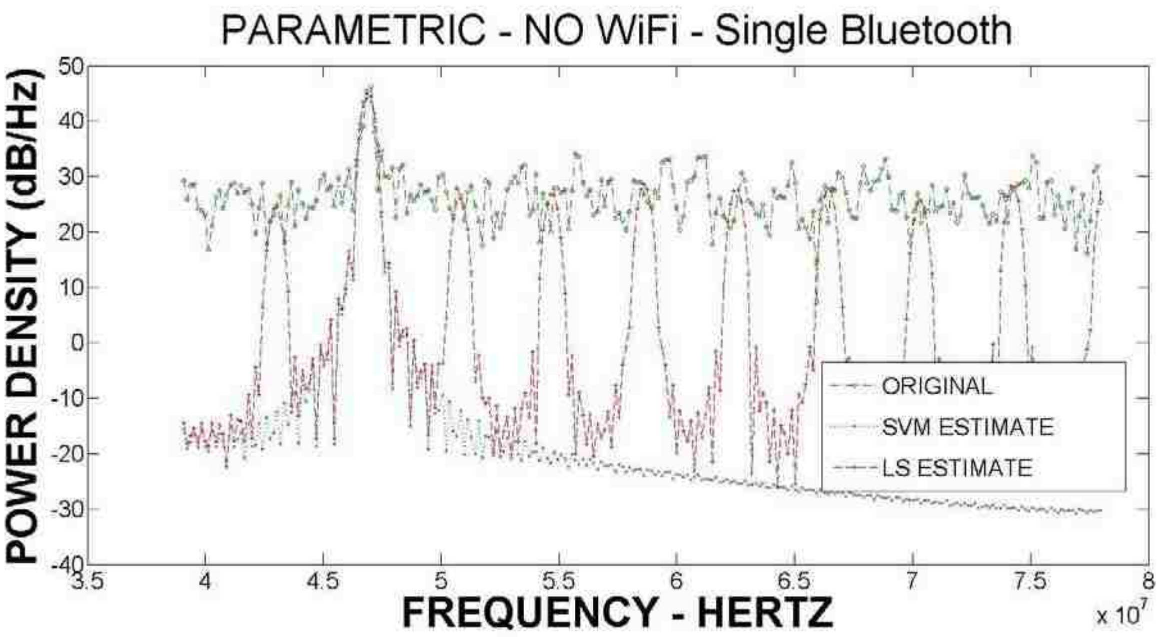


Figure E.6: Parametric Estimator, Channel 42, A+B+I, Sigma = 1

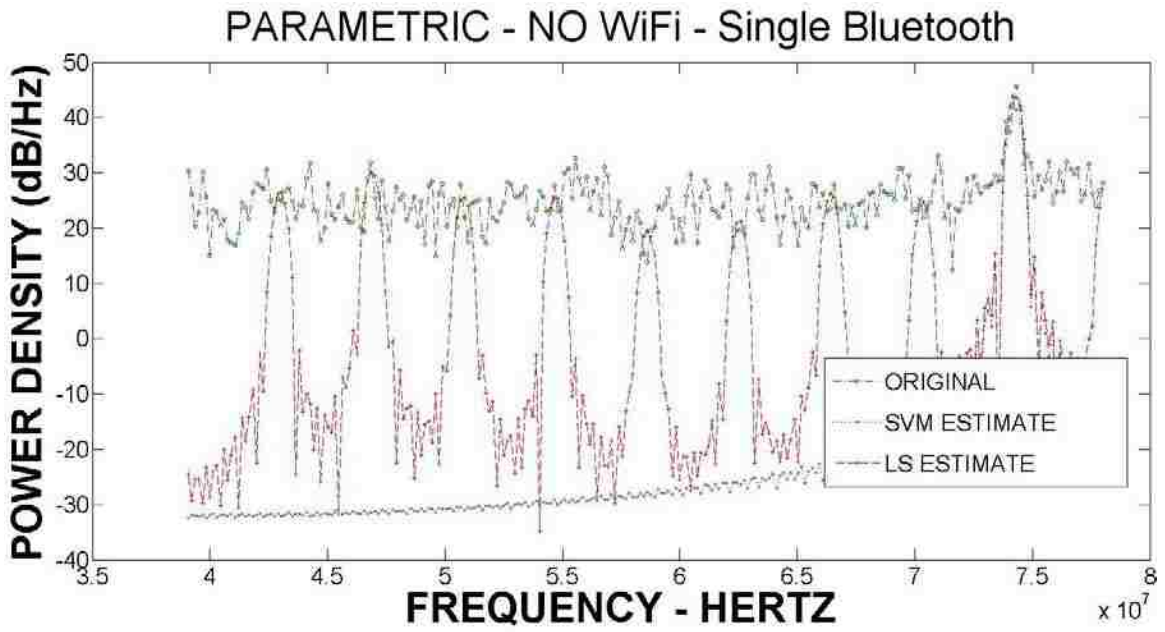


Figure E.7: Parametric Estimator, Channel 70, A+B+I, Sigma = 1

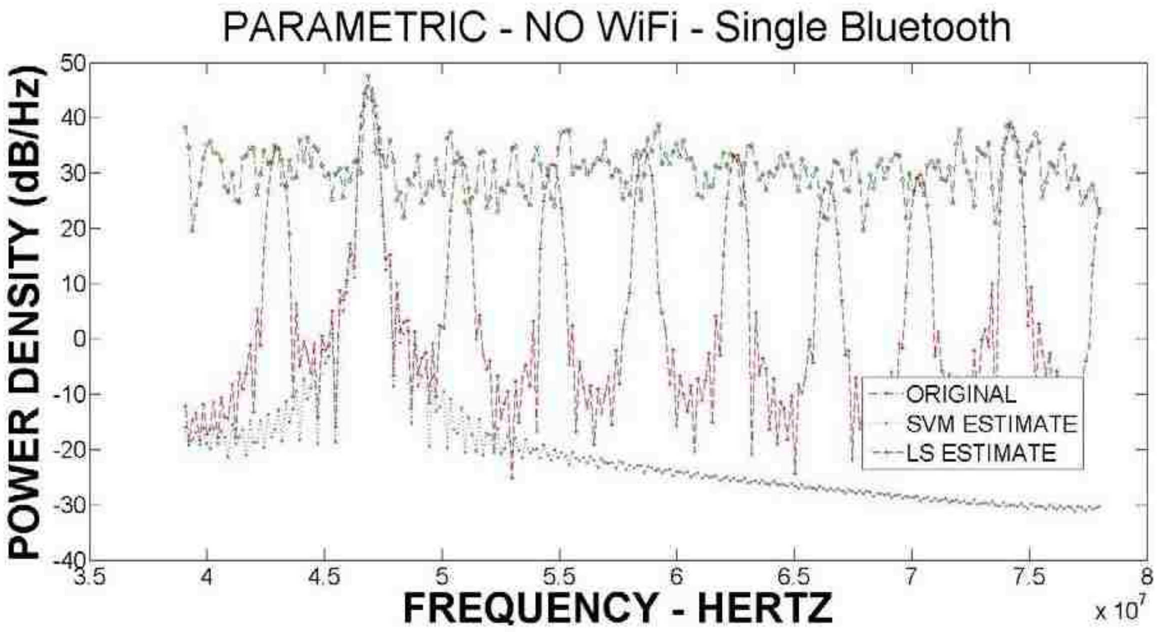


Figure E.8: Parametric Estimator, Channel 42, AWGN, Sigma = 5

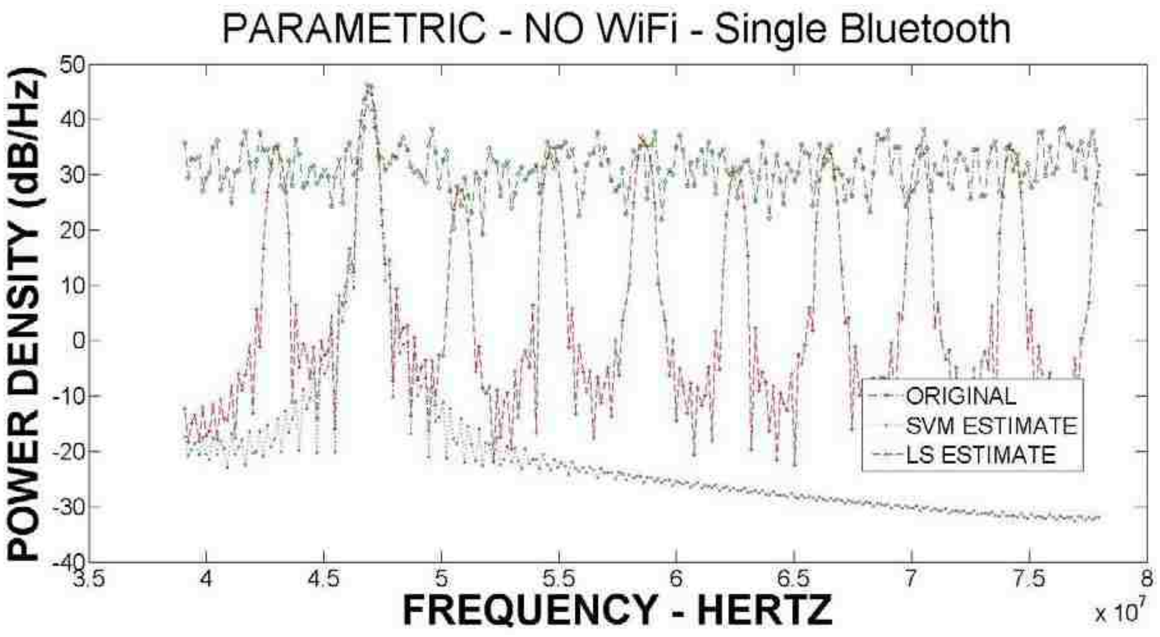


Figure E.9: Parametric Estimator, Channel 42, A+I, Sigma = 5

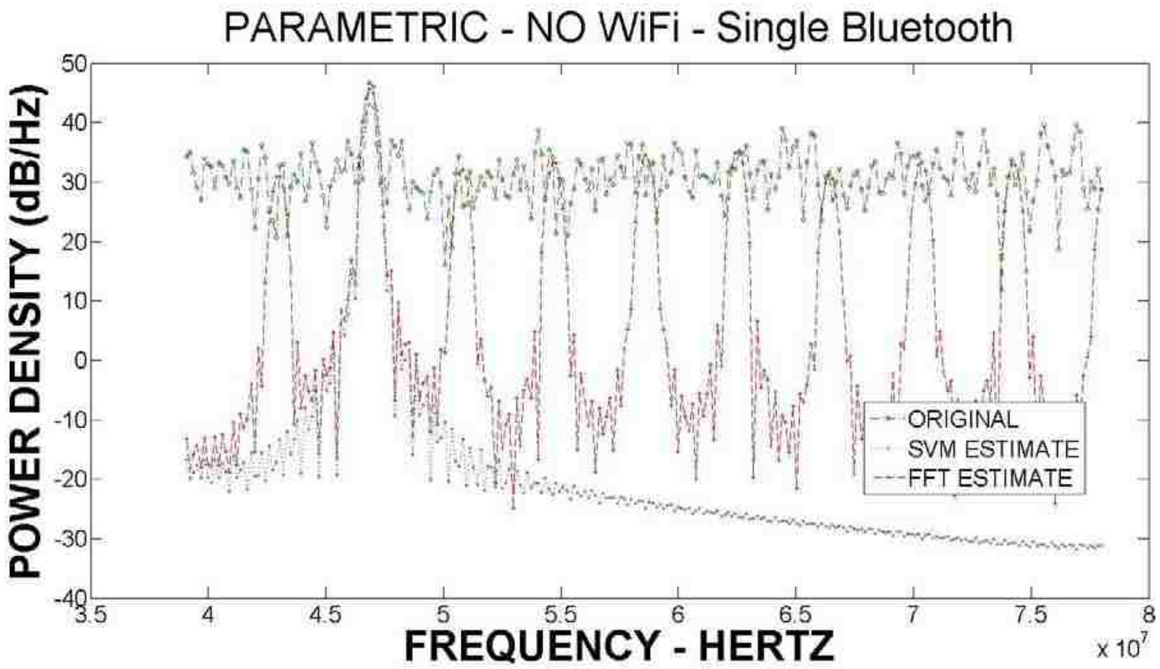


Figure E.10: Parametric Estimator, Channel 42, A+B, Sigma = 5

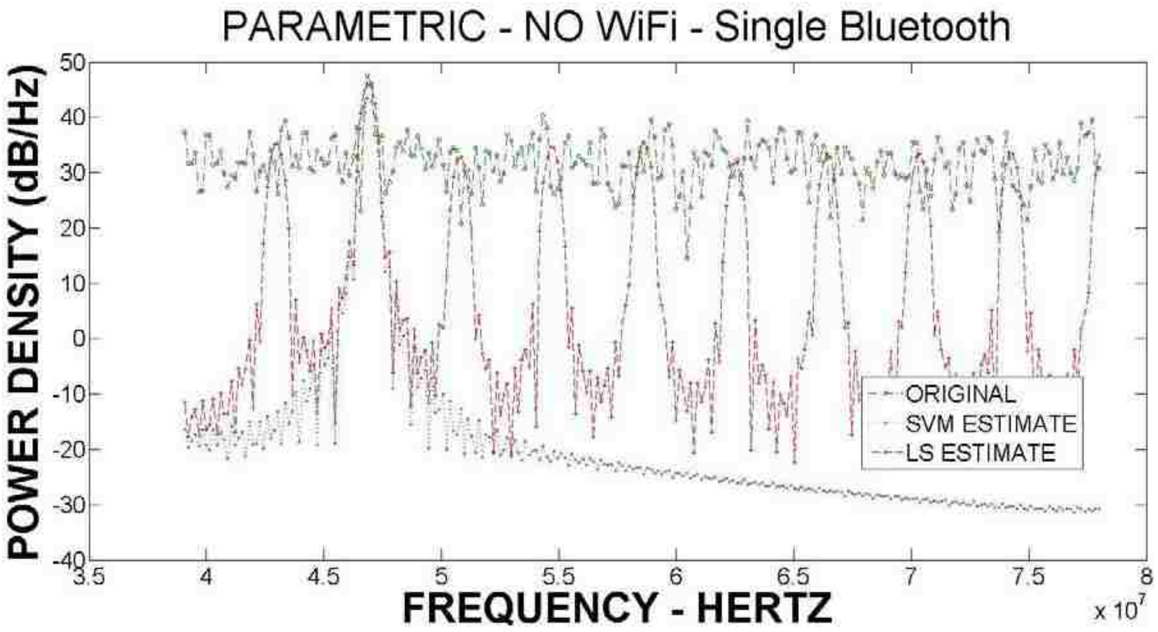


Figure E.11: Parametric Estimator, Channel 42, A+B+I, Sigma = 5

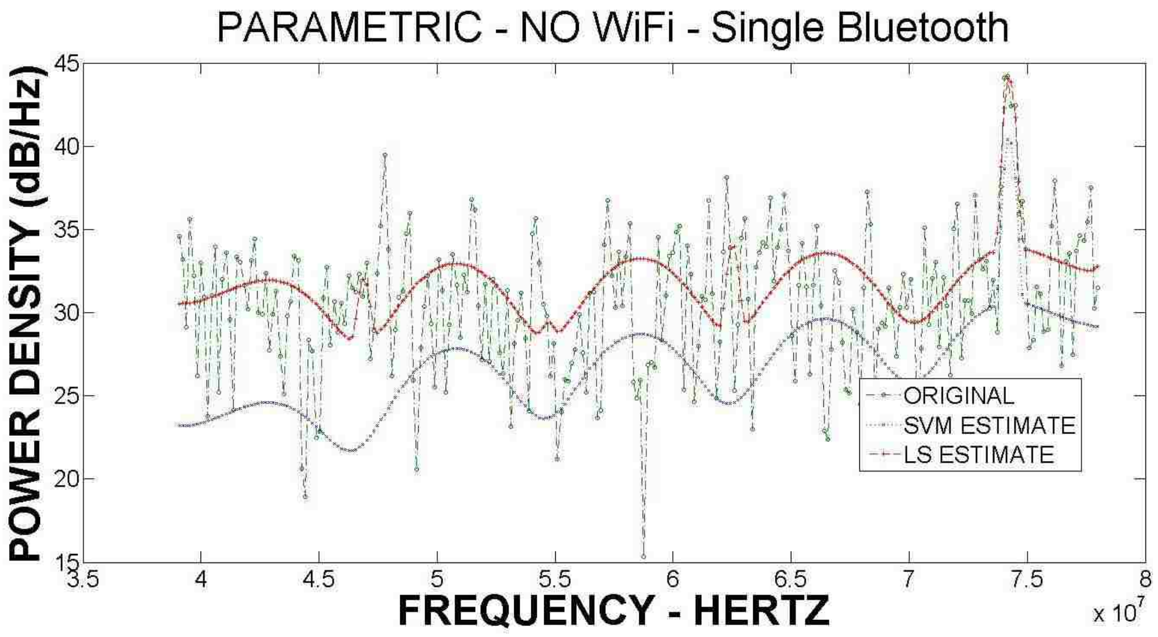


Figure E.12: Parametric Estimator, Channel 70, AWGN, Sigma = 5

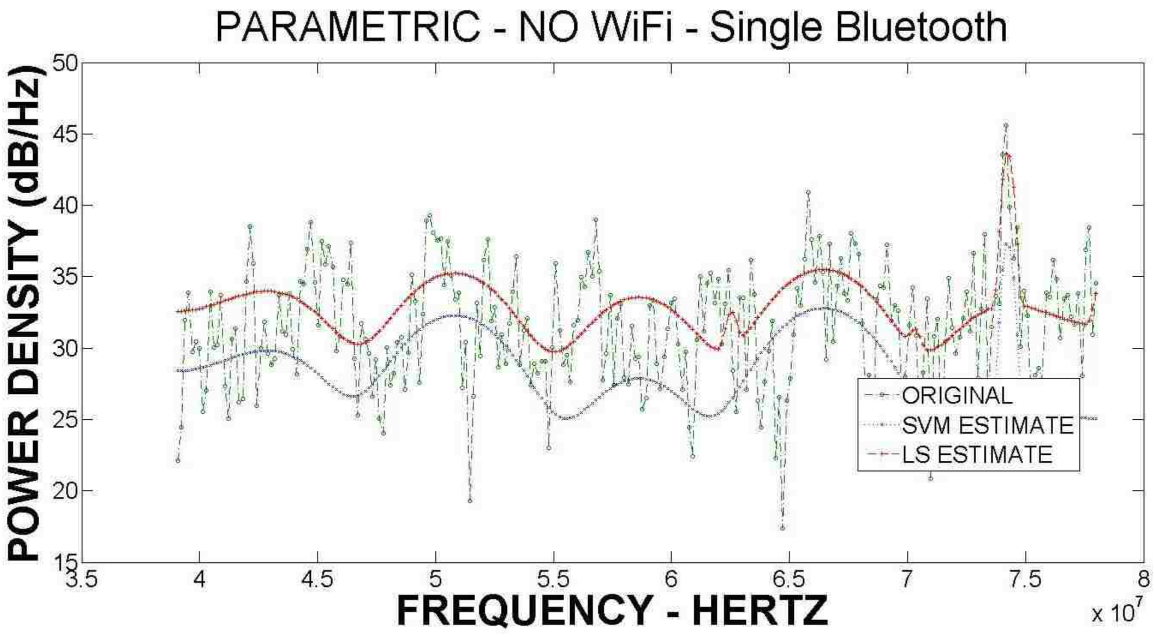


Figure E.13: Parametric Estimator, Channel 70, A+I, Sigma = 5

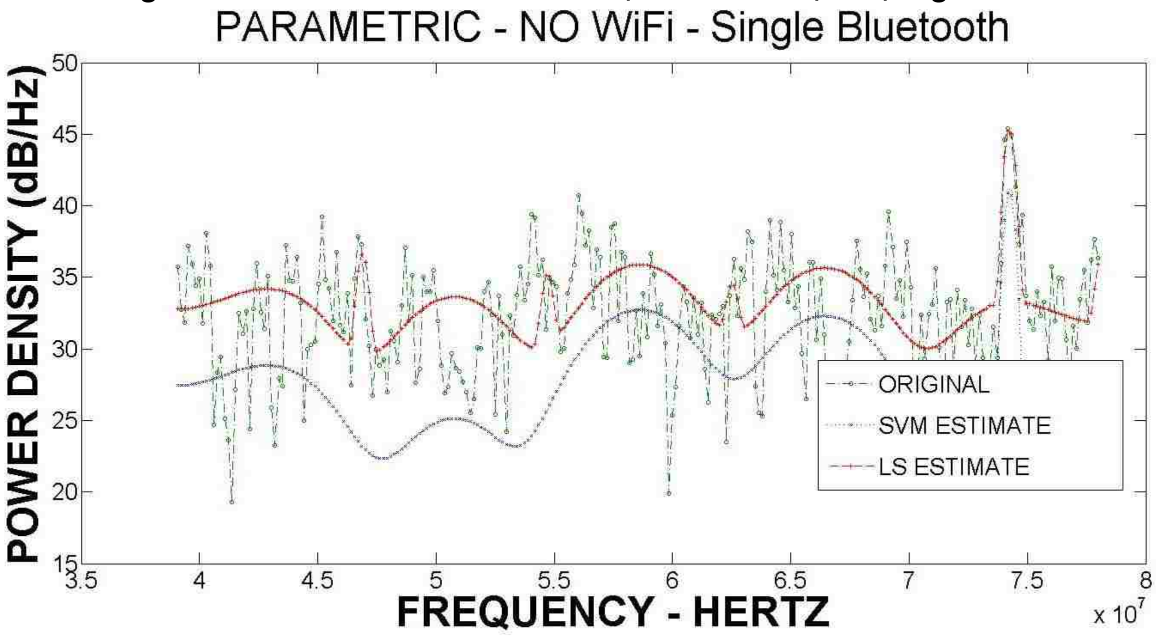


Figure E.14: Parametric Estimator, Channel 70, A+B, Sigma = 5

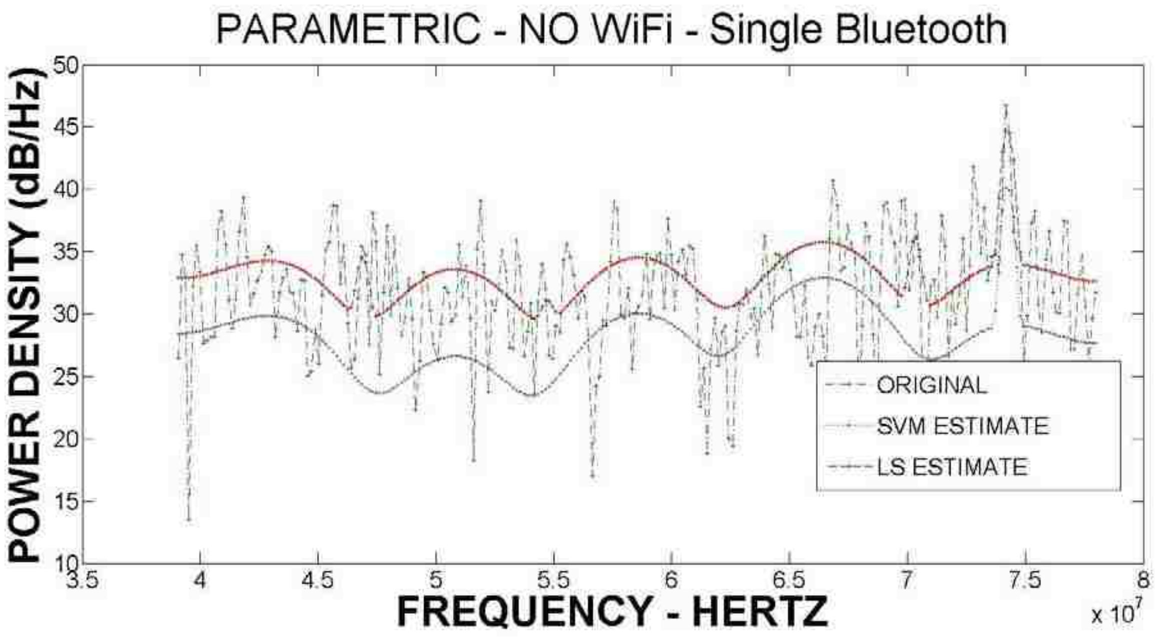
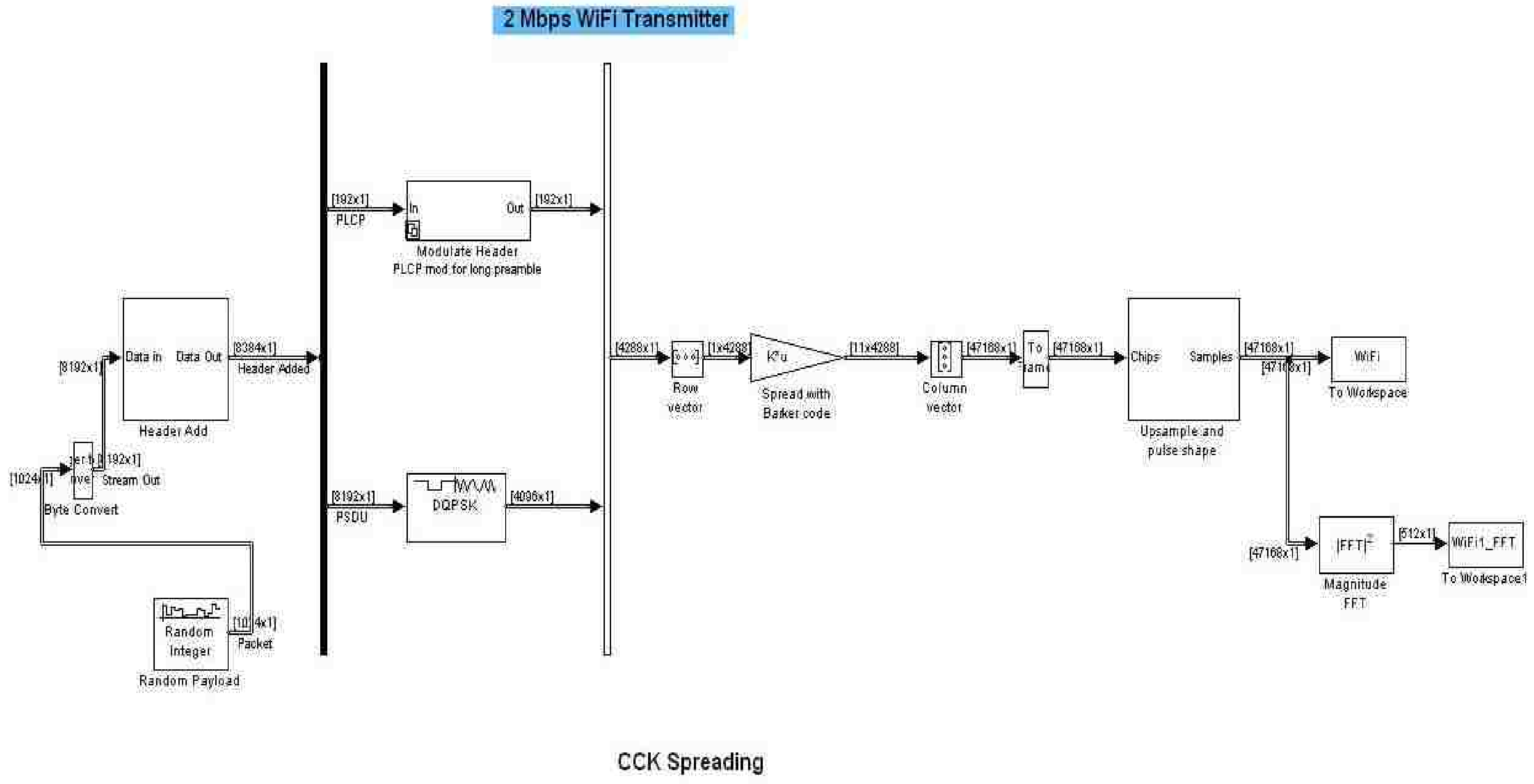


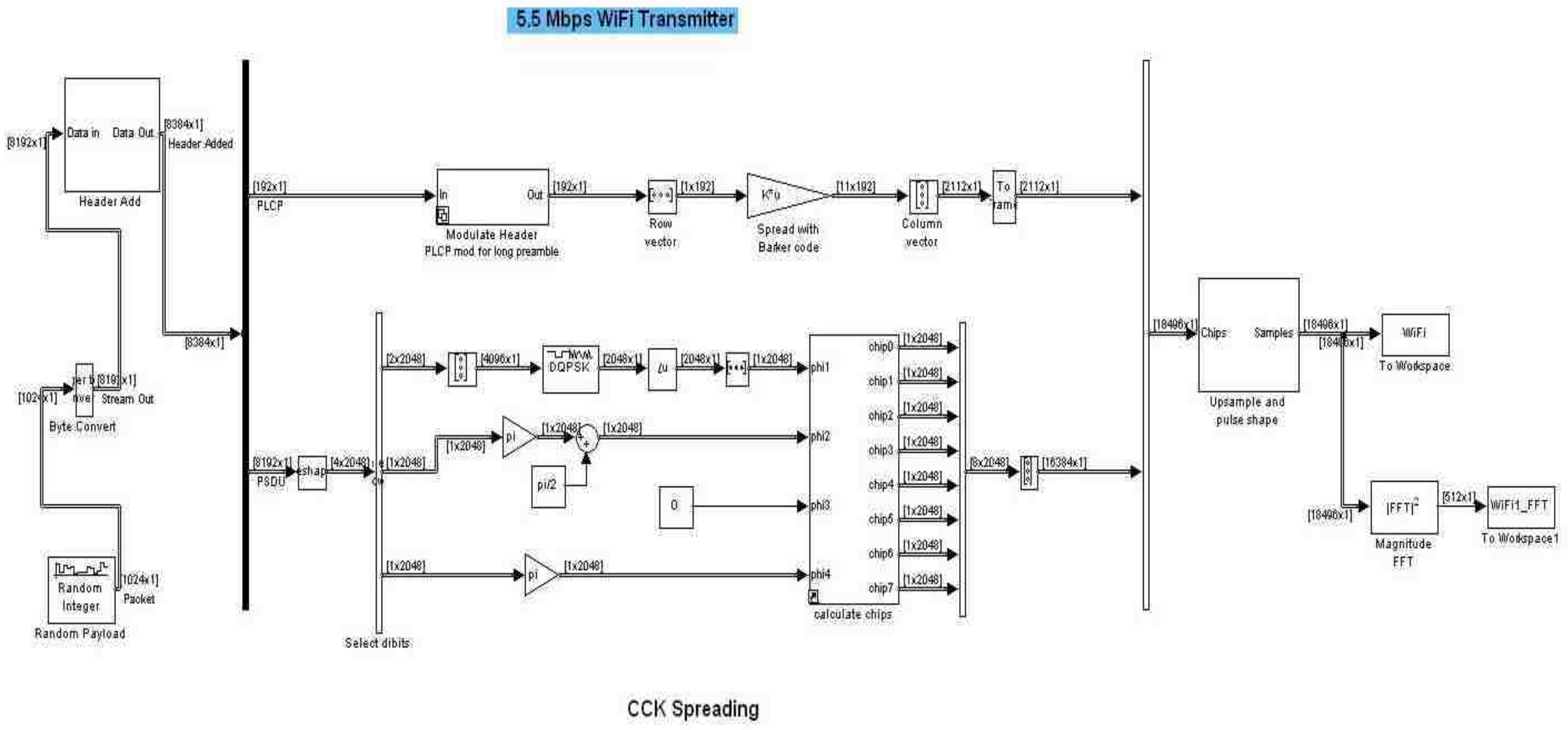
Figure E.15: Parametric Estimator, Channel 70, A+B+I, Sigma = 5

APPENDIX F

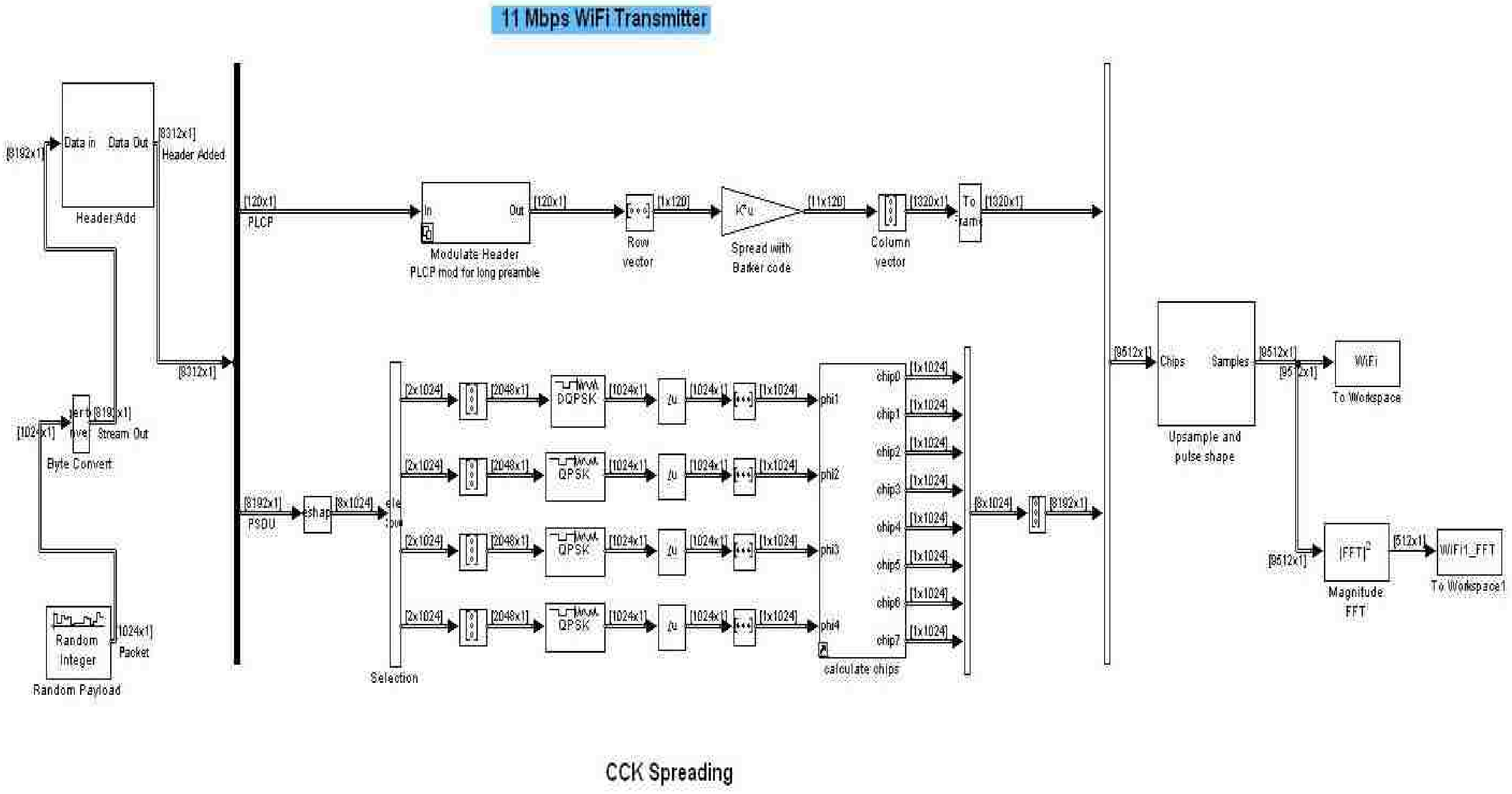
F.1 2 Mbps WiFi Simulink Model



F.2 5.5 Mbps WiFi Simulink Model



F.3 11 Mbps WiFi Simulink Model



APPENDIX G

FURTHER PARAMETRIC ESTIMATES FOR A SINGLE WiFi TRANSMITTER

This appendix presents furthers Parametric estimates for the following noise combinations. These estimates are presented to show further evidence that the parametric estimate works as expected in these types of channels.

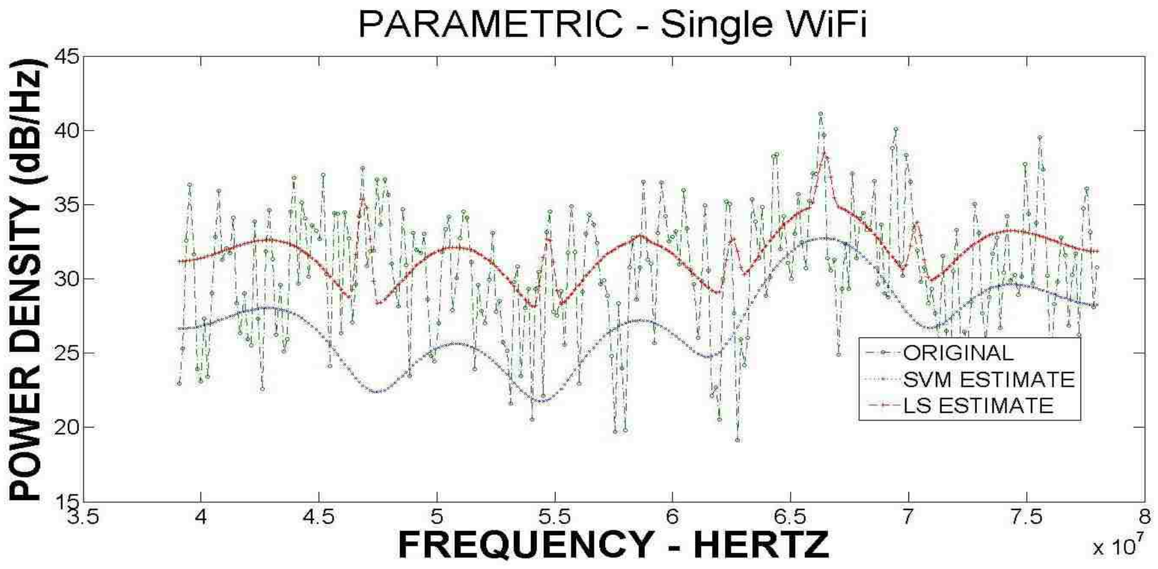


Figure G.1: Parametric Estimator, Single WiFi, AWGN, Sigma = 5

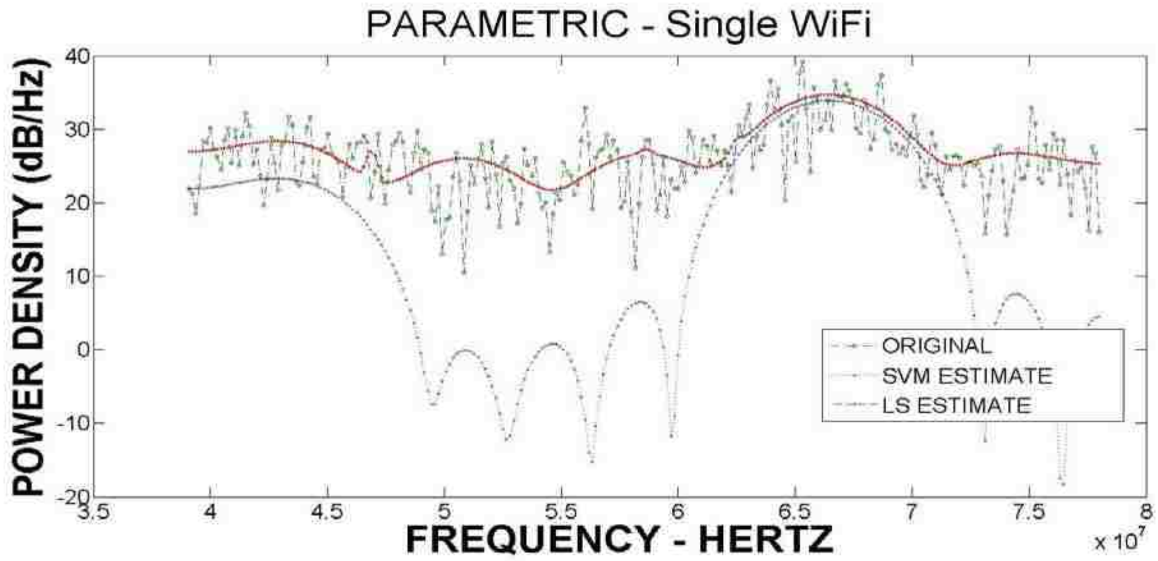


Figure G.2: Parametric Estimator, Single WiFi, A+I, Sigma = 1

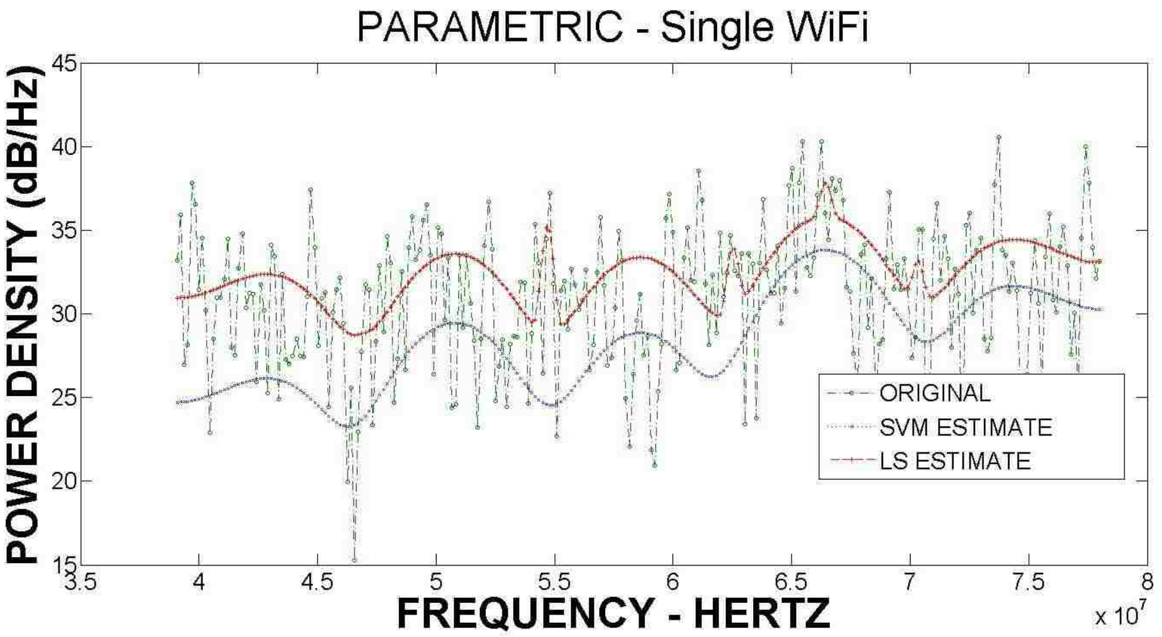


Figure G.3: Parametric Estimator, Single WiFi, A+I, Sigma = 5

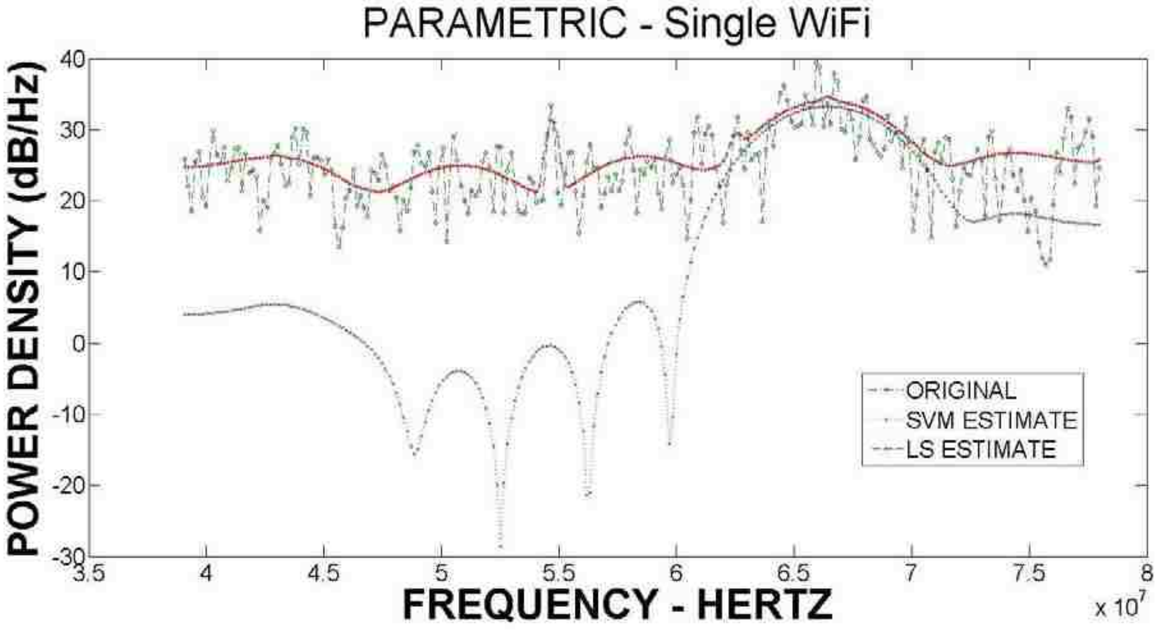


Figure G.4: Parametric Estimator, Single WiFi, A+B, Sigma = 1

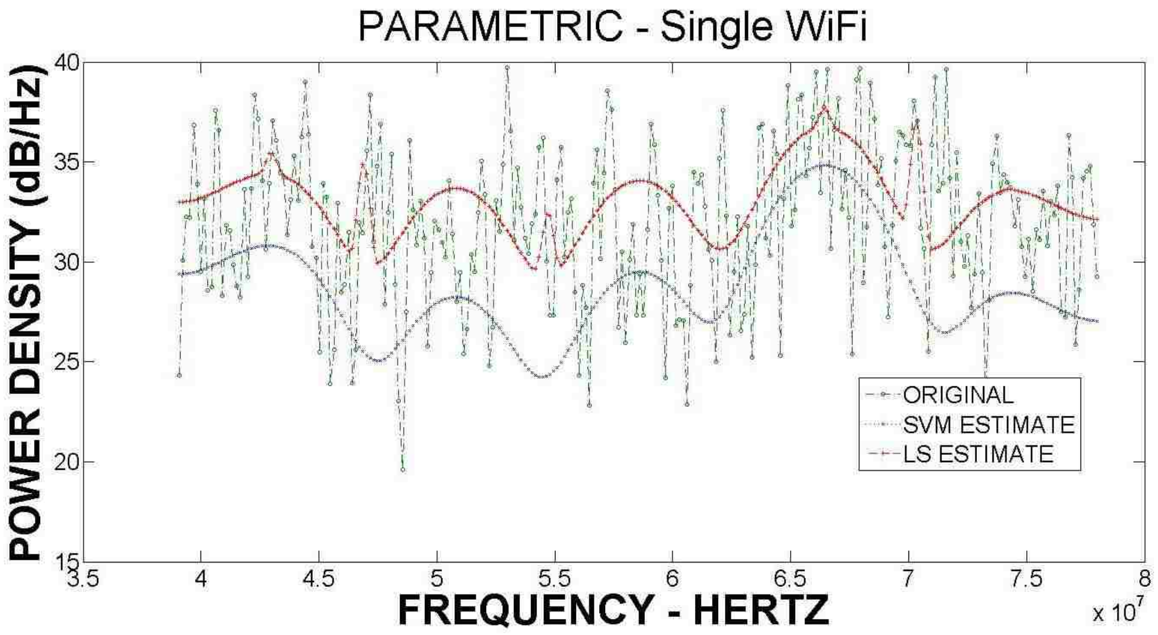


Figure G.5: Parametric Estimator, Single WiFi, A+B, Sigma = 5

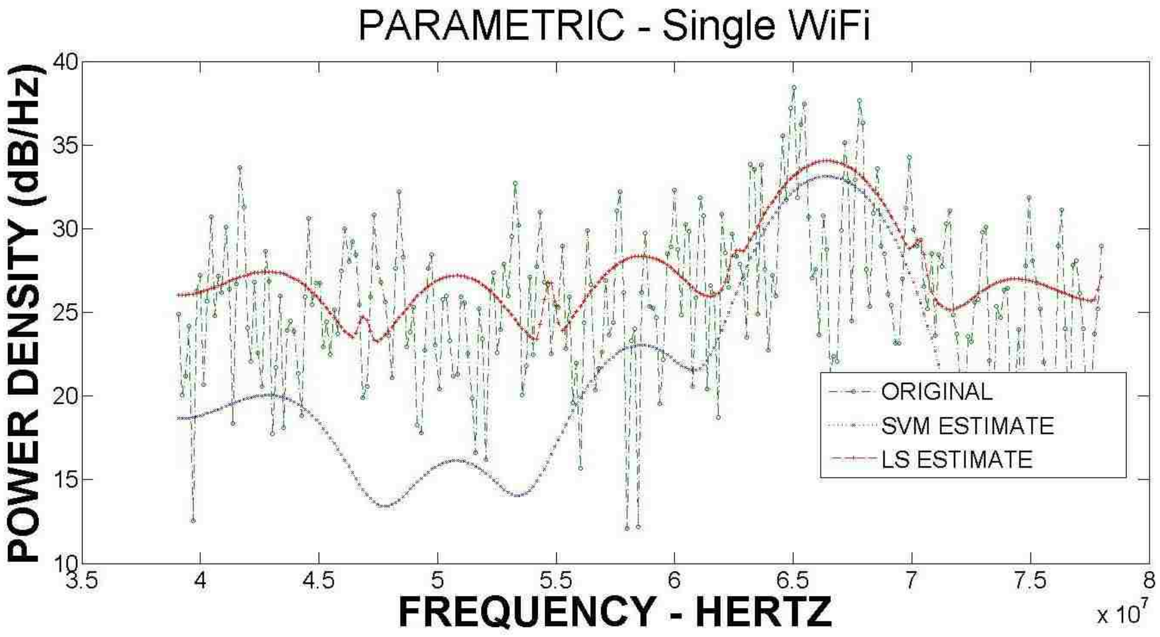
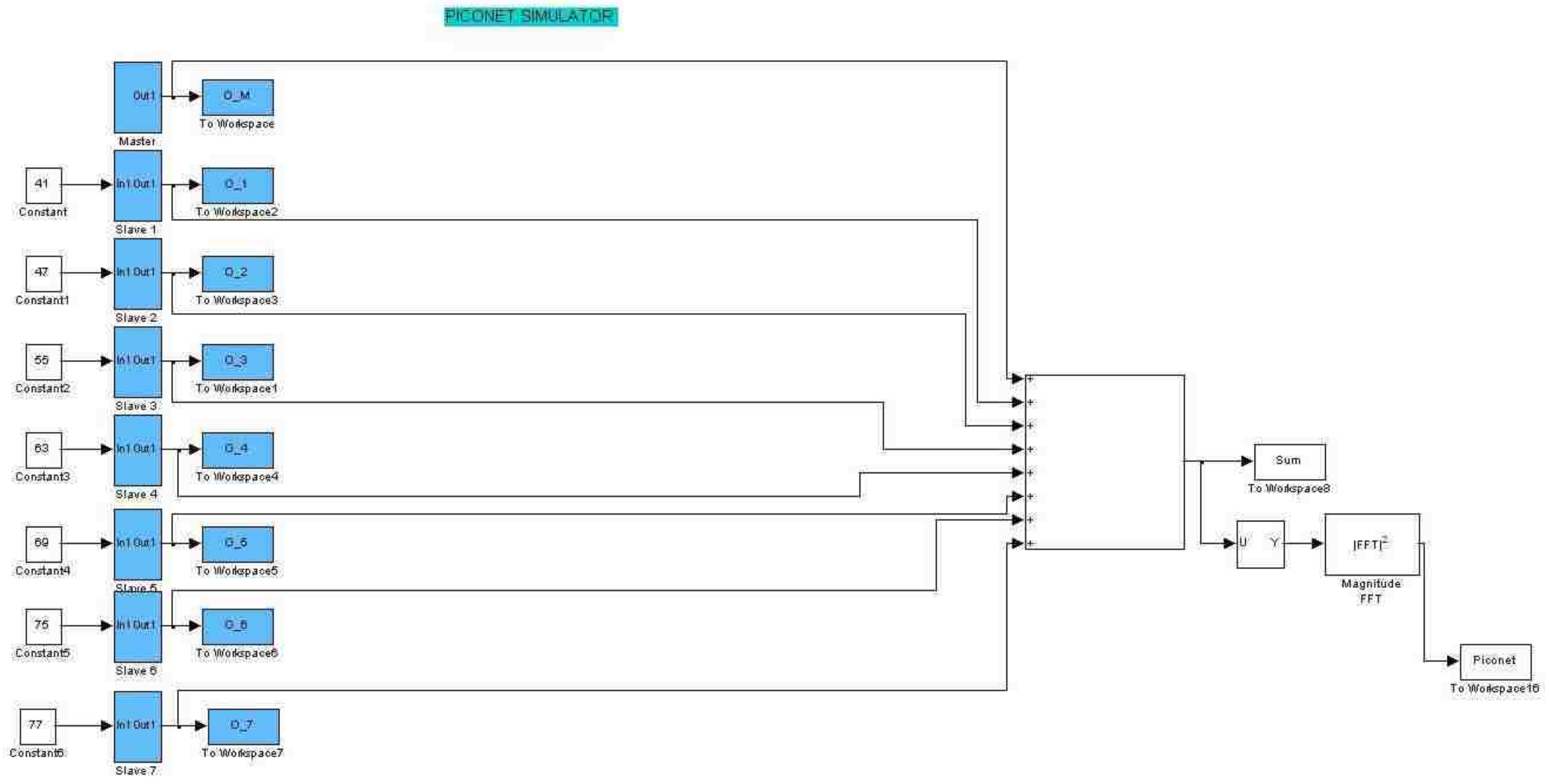


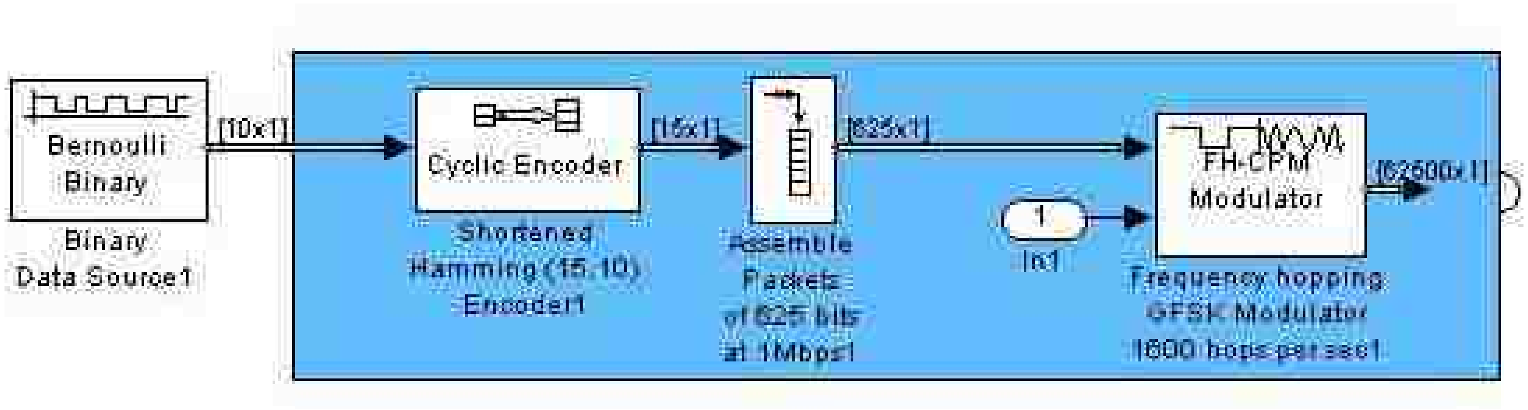
Figure G.6: Parametric Estimator, Single WiFi, A+B+I, Sigma = 1

APPENDIX H

H.1 MATLAB/Simulink Piconet Model



H.2 Detail of an individual transmitter



Bernoulli Binary
Binary Data Source1
[10x1]
Cyclic Encoder
Shortened Hamming (15,10) Encoder1
[15x1]
Assemble Packets of 625 bits at 1Mbps1
[625x1]
1
In1
FH-CPM Modulator
Frequency hopping GFSK Modulator 1600 hops per sec1
[62500x1]

APPENDIX I

Further Estimates for the Bluetooth and WiFi Combinations

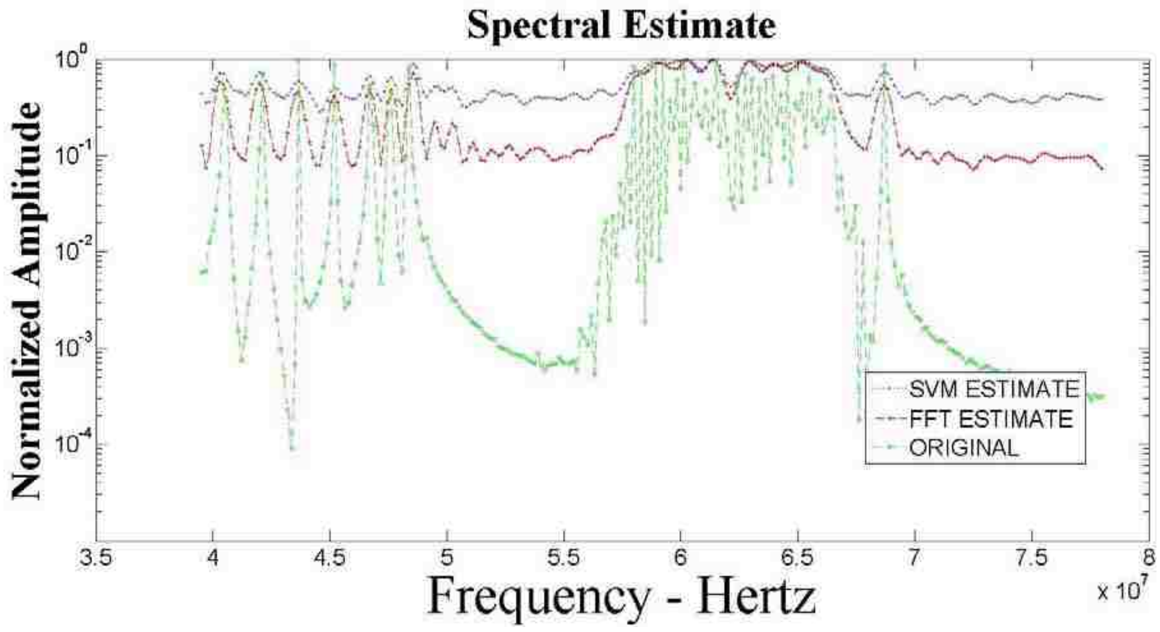


FIGURE I.1: WiFi AND BLUETOOTH, AWGN, SIGMA = 5

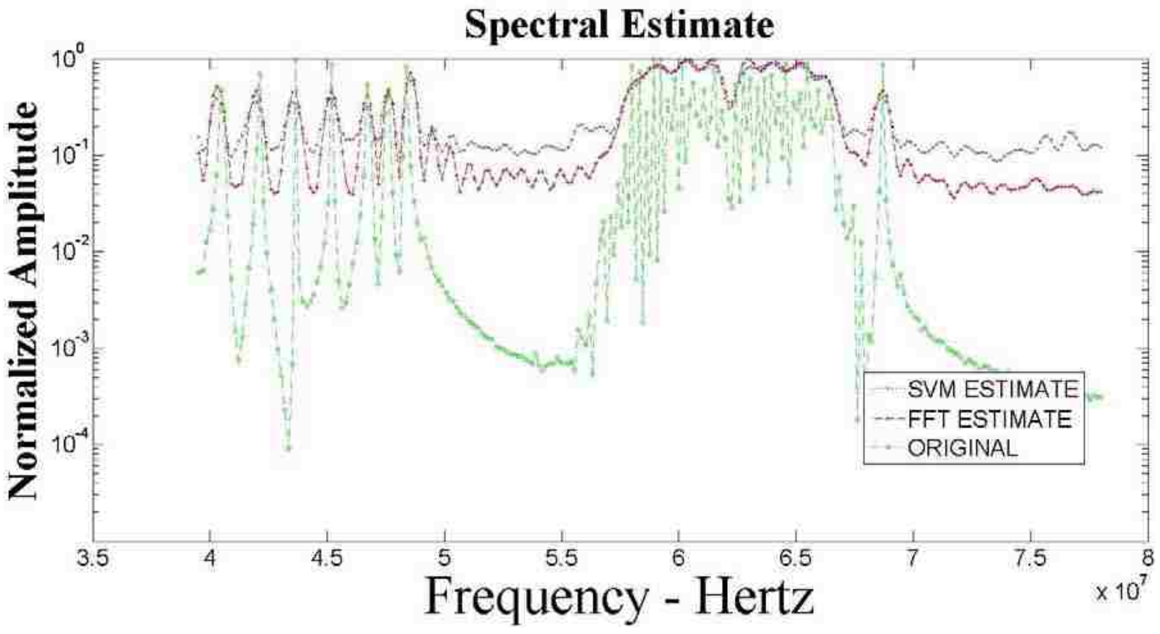


FIGURE I.2: WiFi AND BLUETOOTH, A+I, SIGMA = 1

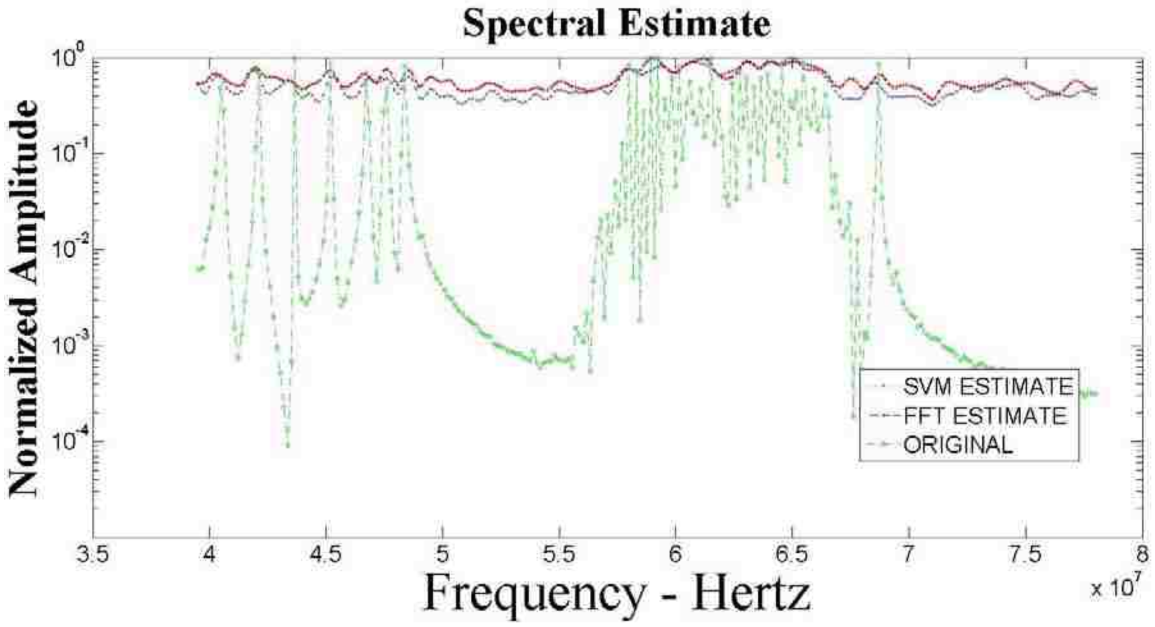


FIGURE I.3: WiFi AND BLUETOOTH, A+I, SIGMA = 5

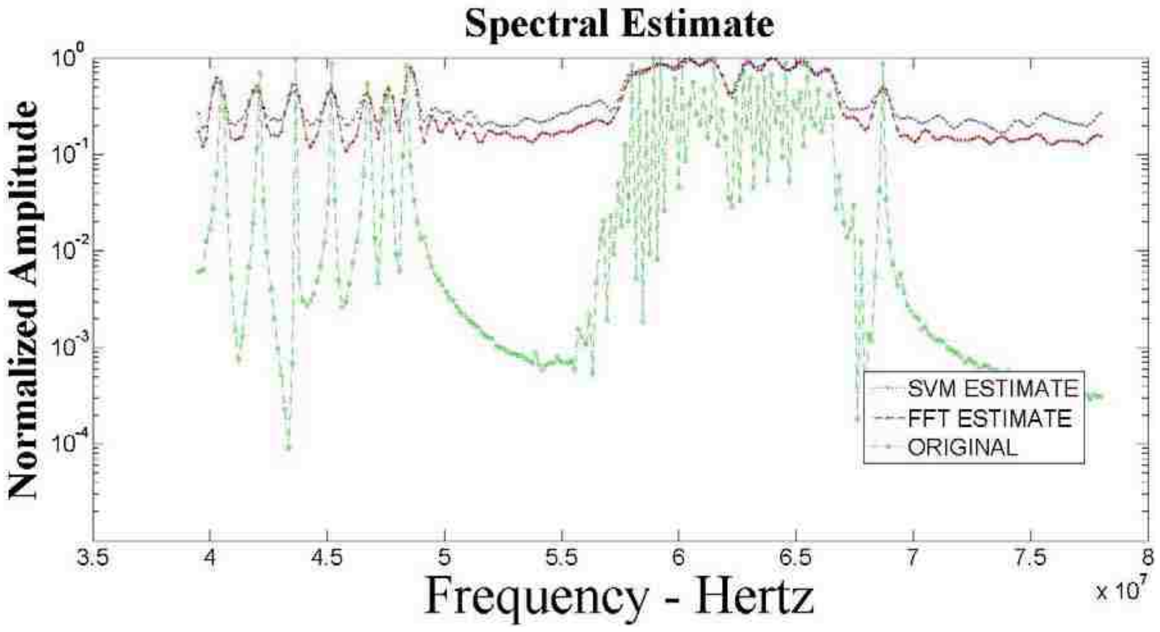
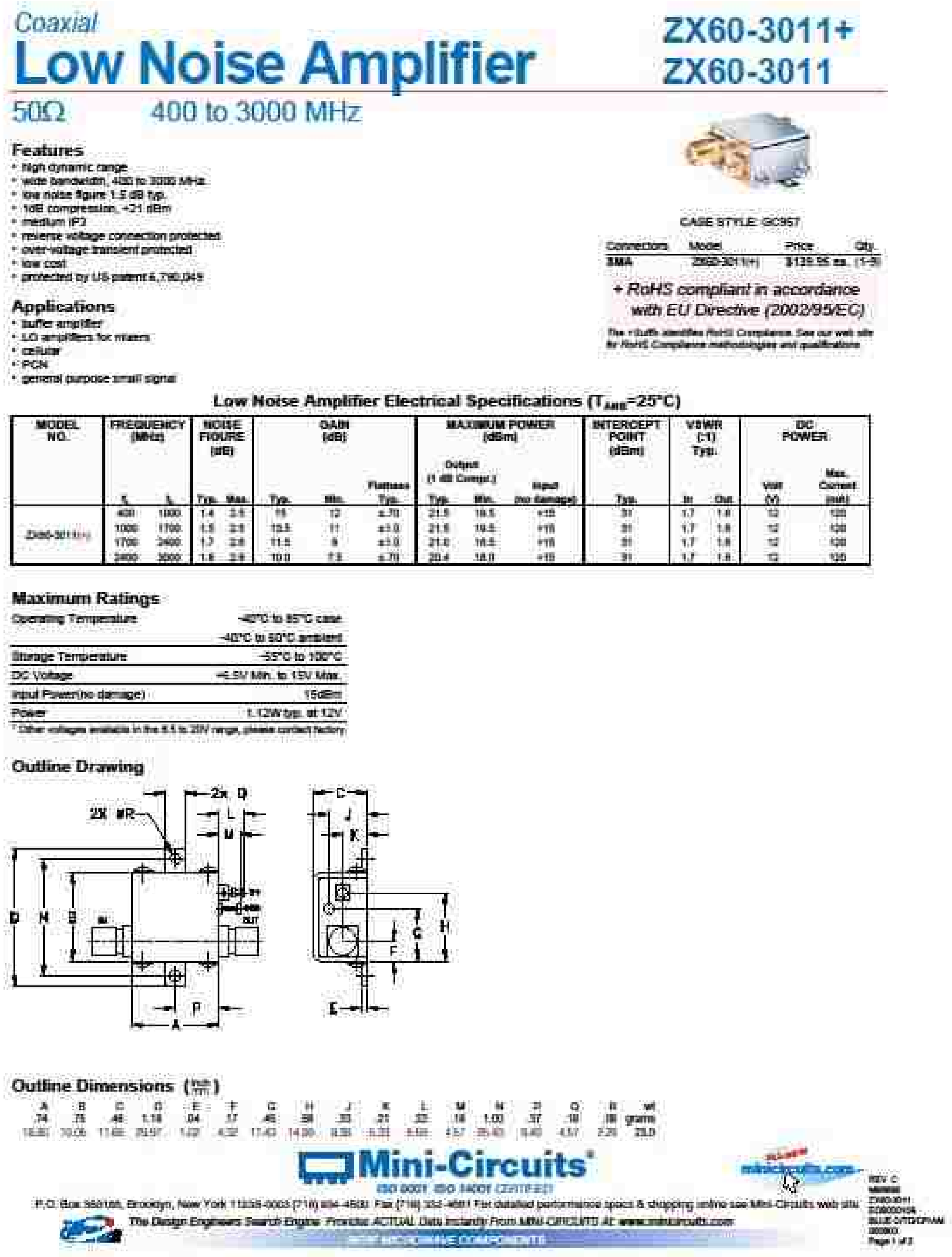


FIGURE I.4: WiFi AND BLUETOOTH, A+B+I, SIGMA = 1

APPENDIX J



Coaxial

Low Noise Amplifier

ZX60-3011+
ZX60-3011

50Ω 400 to 3000 MHz

Features

- high dynamic range
- wide bandwidth, 400 to 3000 MHz
- low noise figure 1.5 dB typ.
- 1dB compression, +21 dBm
- medium IP3
- reverse voltage connection protected
- over-voltage transient protected
- low cost
- protected by US patent 6,790,049

Applications

- buffer amplifier
- LO amplifiers for mixers
- cellular
- PCN
- general purpose small signal

CASE STYLE: GC957

Connectors	Model	Price	Qty.
SMA	ZX60-3011(+)	$139.95 ea.	(1-9)

+ RoHS compliant in accordance with EU Directive (2002/95/EC)

The +Suffix identifies RoHS Compliance. See our web site for RoHS Compliance methodologies and qualifications

Low Noise Amplifier Electrical Specifications (T_{AMB}=25°C)

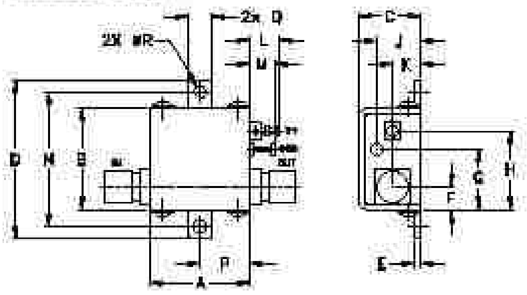
MODEL NO.	FREQUENCY (MHz)		NOISE FIGURE (dB)		GAIN (dB)			MAXIMUM POWER (dBm)			INTERCEPT POINT (dBm)	VSWR (:1) Typ.		DC POWER	
								Output (1 dB Compr.)		Input (no damage)					
	f_L	f_U	Typ.	Max.	Typ.	Min.	Flatness Typ.	Typ.	Min.		Typ.	In	Out	Volt (V)	Max. Current (mA)
ZX60-3011(+)	400	1000	1.4	2.5	15	12	±.70	21.5	19.5	+15	31	1.7	1.8	12	120
	1000	1700	1.5	2.8	13.5	11	±1.0	21.5	19.5	+15	31	1.7	1.8	12	120
	1700	2400	1.7	2.8	11.5	9	±1.0	21.0	16.5	+15	31	1.7	1.8	12	120
	2400	3000	1.8	2.8	10.0	7.5	±.70	20.4	18.0	+15	31	1.7	1.8	12	120

Maximum Ratings

Operating Temperature	-40°C to 85°C case
	-40°C to 50°C ambient
Storage Temperature	-55°C to 100°C
DC Voltage	+6.5V Min. to 15V Max.
Input Power(no damage)	15dBm
Power	1.12W typ. at 12V

* Other voltages available in the 6.5 to 20V range, please contact factory

Outline Drawing



Outline Dimensions ($\frac{inch}{mm}$)

A	B	C	D	E	F	G	H	J	K	L	M	N	P	Q	R	wt
.74	.75	.46	1.18	.04	.17	.45	.59	.33	.21	.32	.18	1.00	.37	.18	.09	grams
18.80	19.05	11.68	29.97	1.02	4.32	11.43	14.99	8.38	5.33	8.13	4.57	25.40	9.40	4.57	2.29	23.0

Mini-Circuits®

ISO 9001 ISO 14001 CERTIFIED

P.O. Box 350166, Brooklyn, New York 11235-0003 (718) 934-4500 Fax (718) 332-4661 For detailed performance specs & shopping online see Mini-Circuits web site

The Design Engineers Search Engine Provides ACTUAL Data Instantly From MINI-CIRCUITS At: www.minicircuits.com

RF/IF MICROWAVE COMPONENTS

minicircuits.com

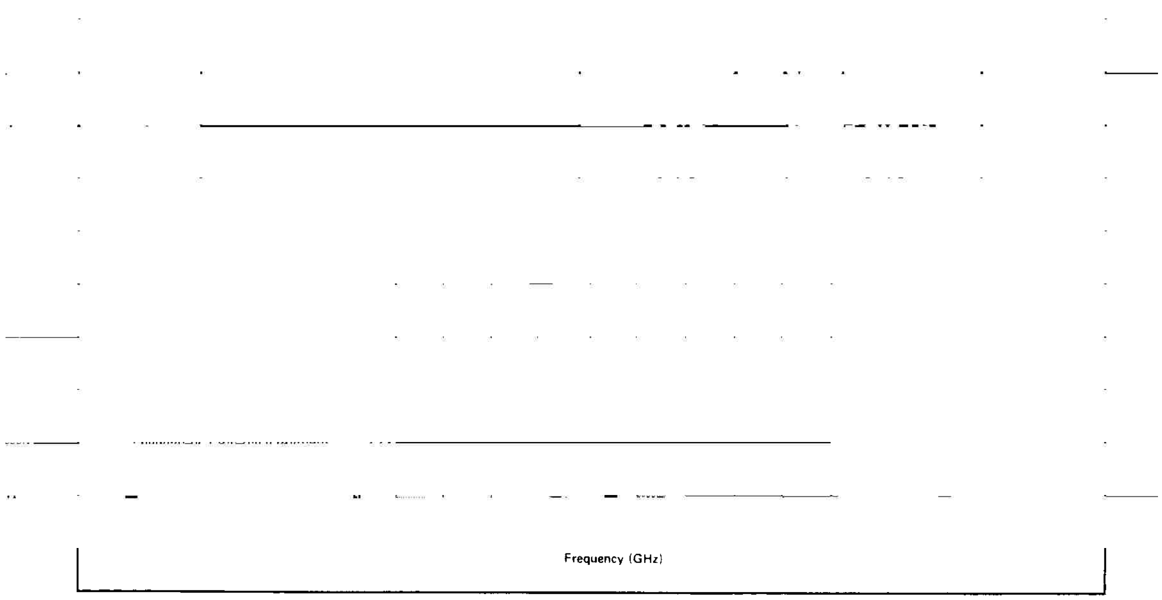
REV. C
[illegible]
ZX60-3011
[illegible]
[illegible]
[illegible]
Page 1 of 2

[illegible]

Maximum Input Power: +27 dBm (0.5 W)

[illegible]

Shipping Weight: 0.14 kg (4.94 oz.)



Marki microwave

DOUBLE-BALANCED MIXERS M1-0208

Features

- LO/RF 2.0 to 8.0 GHz
- IF DC to 2.0 GHz
- 5.5 dB Typical Conversion Loss
- 38 dB Typical LO to RF Isolation
- Multi-Octave RF and LO

Electrical Specifications - Specifications guaranteed from -55 to +100°C, measured in a 50-Ohm system.

Parameter	LO (GHz)	RF (GHz)	IF (GHz)	Min	Typ	Max	Diode Option LO drive level (dBm)
Conversion Loss (dB)	2.0-8.0	2.0-8.0	DC-1.0		6.0	7.0	
	2.0-8.0	2.0-8.0	1.0-2.0		6.5	8.5	
Isolation (dB)							
LO-RF	2.0-8.0	2.0-8.0		25	38		
LO-IF	2.0-8.0	2.0-8.0			20		
RF-IF	2.0-8.0	2.0-8.0			25		
Input 1 dB Compression (dBm)	2.0-8.0	2.0-8.0			+2		L (+7 to +10)
					+5		M (+10 to +13)
					+8		N (+13 to +16)
					+11		H (+16 to +19)
					+14		S (+19 to +22)
Input Two-Tone Third Order Intercept Point (dBm)	2.0-8.0	2.0-8.0			+12		L (+7 to +10)
					+15		M (+10 to +13)
					+18		N (+13 to +16)
					+21		H (+16 to +19)
					+24		S (+19 to +22)

Part Number Options

Please specify diode level and package style by adding to model number.	
Package Style(s)[1,2]	**Example**
A, B, C, E, EZ, F	M1-0208 L A

[1]Connectorized test fixtures available for most carrier and surface mount packages. Consult factory.
[2]For non-connectorized packages, specify I-port configuration by adding –1 or –2 suffix to model number. Default is –2 configuration when not specified.

215 Vineyard Court, Morgan Hill, CA 95037 | Ph: 408.778.4200 | Fax 408.778.4300 | info@markimicrowave.com

[illegible]