University of Southern Queensland

School of Engineering

Cognitive Radio for Microwave Spectrum Access

A dissertation submitted by

Joshua Knipe

in fulfilment of the requirements of

ENP4111 Research Project

towards the degree of

Bachelor of Engineering Honours (Electrical & Electronic) and

Bachelor of Science (Mathematics)

Submitted: November, 2024

Abstract

Within the microwave spectrum, new technologies such as the Internet of Things, 5G and Wi-Fi 6 are being implemented into an already allocated frequency spectrum. As the Australian Communications and Media Authority seeks new frequency management tools, cognitive radio techniques could reduce the alloted frequency bands or improve frequency utilisation within the microwave spectrum.

As cognitive radio research is generally focused on applying or modifying cognitive radio techniques for niche cases, there is a need for real-world applications or practical simulation data of cognitive radio techniques applied to new or upcoming technologies.

From a review of available cognitive radio techniques, a non-cooperative Maximum-Minimum Eigenvalue (MME) Detection method was chosen to gather practical signal detection data of 5G New Radio, LTE and Wi-Fi 6 signals.

Using Python, the limited practical simulation data indicated that 5G New Radio is easier to detect when compared to LTE and Wi-Fi 6. Limited testing of the correlative and additive noise suggested that low signal-to-noise environments can result in false detections or reduction in detection ability.

Further work applying cognitive radio techniques to other microwave spectrum signals is required, as well as real-world testing.

University of Southern Queensland

School of Engineering

ENP4111 *Research Project*

Limitations of Use

Certification of Dissertation

I certify that the ideas, designs and experimental work, results, analyses and conclusions set out in this dissertation are entirely my own effort, except where otherwise indicated and acknowledged.

I further certify that the work is original and has not been previously submitted for assessment in any other course or institution, except where specifically stated.

Joshua Knipe

[redacted]

Acknowledgments

I would like to thank my family and friends for their support during this project, for it has been a long year.

I would also like to thank my supervisor, John Leis, for his support and encouragement throughout the project.

Joshua Knipe

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem Specification

With the arrival of 5G and the Internet of Things, the microwave spectrum is facing increased demand for frequency space. According to the Australian Communications Media Authority, new technologies like 5G and Wi-Fi 6 are contested for frequency space and are expected to operate in the same frequency band (Australian Communications and Media Authority 2024, pp. 20-21).

To help manage the frequency allocations in Australia, the Australian Communications Media Authority have contracted Spectrum Center Inc. to develop a modern frequency spectrum management system (Australian Communications and Media Authority 2024, pg. 23). The allocation, management and control of frequency spectrum has been a serious issue since the United States Government idenitified spectrum access and utilisation problems in 2002 (Haykin 2005).

This is still an ongoing problem that the United States Government continues to recognise as 'The Spectrum Crunch' problem, where spectrum demand requires allocations for new spectrum efficient systems or the development of better multi-access communication systems (National Institute of Standards and Technology 2022).

Using cognitive radio, it is hoped that better multi-access communication systems can be developed to ease 'The Spectrum Crunch' by utilising frequency space more efficiently.

1.2 Aim and Objectives

By completing this project, it is hoped that signal detection data that allows for practical applications or real-world testing can be gathered on microwave spectrum signals.

Thus, to accomplish this aim, the following objectives must be achieved:

- Identification of signal detection techniques for microwave spectrum signals.
- Identification of frequency bands within the microwave spectrum that are currently underutilised or at high-risk of interference.
- Identification of signals operating within high-risk or underutilised microwave frequency bands.
- Creation or collection of a dataset containing signals from the microwave spectrum.
- Creation of a signal detection testing environment.
- Successful detection of a microwave frequency signal using a chosen signal detection technique.
- Verification of the impact of noise in signal detection.
- Statistical analysis of signal detection results.
- Creation and testing of a cognitive radio system in a real-world environment.

1.3 Scope and Limitations

Due to the significant scope of the microwave spectrum, 300 MHz to 300 GHz (North American Space Administration 2010), the project must be limited to testing a limited number of microwave spectrum signals or frequency bands. This is to avoid spending a significant amount of time investigating low-risk underutilised systems and/or communication systems with difficult to find documentation. For example, investigating high-end microwave frequency signals (around 300 GHz) or microwave spectrum transmission systems with little publicly available research.

Additionally, the project's aim has been reduced from implementing a cognitive radio solution (smart system capable of identifying and using an open frequency band) to the testing and collection of microwave spectrum signal data for future applications.

The project has also been limited to simulations of one signal detection method to gather data, as the testing of multiple signal detection methods and real-world testing may not be viable due to time and budgetary concerns.

1.4 Overview

This dissertation is organised as follows:

Chapter 1 contains an introduction to the frequency utilisation, allocation and interference problem, as well as the project aim, scope, objectives and limitations.

Chapter 2 contains a review of cognitive radio and signal detection research, as well as a review of the microwave spectrum signals to be tested.

Chapter 3 contains the dataset collection method, the chosen signal detection method, the creation of a signal detection testing environment, the analysis of results and the expected results.

Chapter 4 contains the various signal detection tests and their results.

Chapter 5 contains the conclusions, achieved project outcomes and suggestions for further work.

Chapter 2

Literature Review

2.1 Cognitive Radio

Given the existence of potential issues in the microwave spectrum, cognitive radio may allow for users to smartly and independently access the microwave spectrum without interfering with critical infrastructure or other users.

Building on the work of J. Mitola, Simon Haykin suggested the use of cognitive radio to intelligently detect and use spectrum holes, frequency bands within a spectrum that were not used by a primary user (user who is typically alloted to the frequency band), as a method to allow other non-primary users (secondary/opportunistic users) to access the desired spectrum (Haykin 2005).

The ideal cognitive radio system, as presented by Haykin, must be able to learn from its radio environment and adapt itself (the radio system) to changes within the radio environment to communicate reliably and efficiently within the radio system's operational frequency spectrum (Haykin 2005).

It is from this definition that the concept of the 'Cognitive Cycle' was developed, which can be seen in Figure 2.1 as taken from Shekhawat & Yadav (2021) in their review.

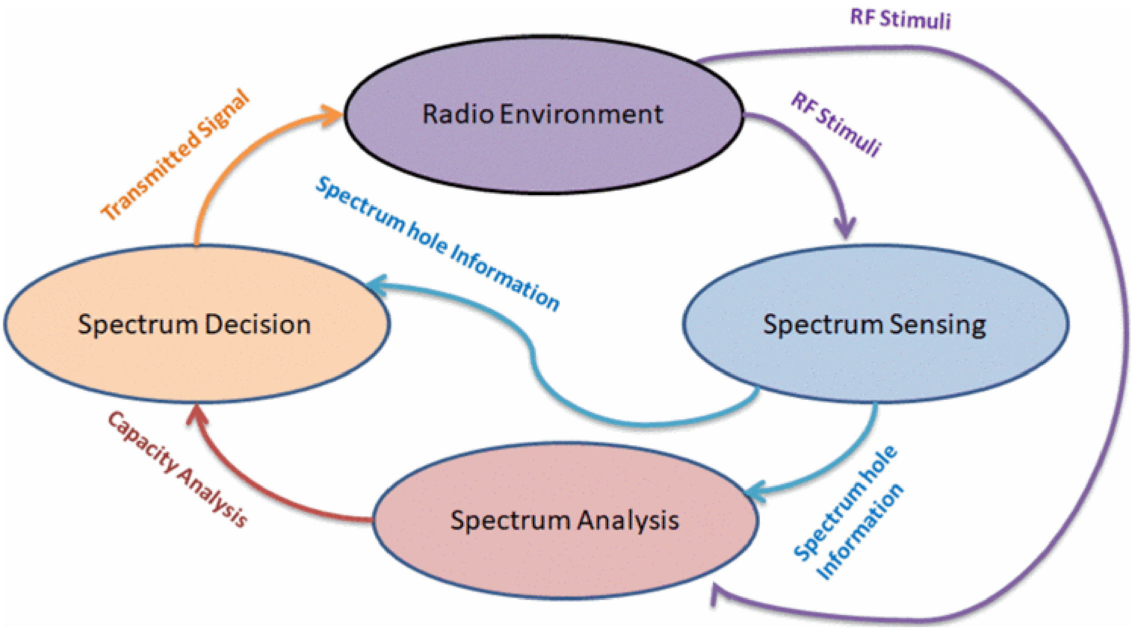

Figure 2.1: The Cognitive Cycle taken from Shekhawat & Yadav (2021)

From the works of Haykin (2005) and Shekhawat & Yadav (2021), the cognitive cycle shows how a cognitive radio system should operate. Given a radio environment, a cognitive radio system is expected to analyse and detect relevant spectrum information, which allows the cognitive radio to determine available spectrum holes (absence of primary users or unused frequency bands) and make a decision regarding the ability to transmit without interfering with primary user or others (Haykin 2005, Shekhawat & Yadav 2021).

Currently, it is expected that cognitive radio techniques can be applied to modern communications networks and applications such as 5G and Internet of Things. A recent review by Guiliano, Hilal, Alsadi, Gadsen & Yawney (2022) proposes that Cognitive Radio could be integrated into a Internet of Things (IoT) application by seperating known IoT processes into layers, forming a cognitive cycle.

For 5G, a conference paper by Kakalou, Psannis, Goudos, Yioultsis, Kantartzis & Ishibashi (2019) proposes the use of radio environment maps to record spectrum and channel information for cognitive radio learning, while the conference paper by Ramyea & Kasthuri (2018) identifies eigenvalue detection as a superior signal detection method for 5G.

In the case of Wi-Fi, a spectrum sensing study by Hanna & Sydor (2012) , identified both Wi-Fi low channel occupancy and low interference in an outdoor urban environment, using a Cognitive Radio Learning platform. It was found that a practical real-world application of cognitive radio may allow for the usage of both unused and underutilised channels within a section of Ottawa, Canada.

However, it is difficult to find real-world applications or practical simulations of cognitive radio techniques that allow for direct comparison between communication systems or signal types. Information regarding practical spectrum sensing or signal detection would be useful for future practical applications in communication channel utilisation and/or detecting hostile secondary users. The spectrum sensing study by Hanna & Sydor (2012) discussed previously, is an example of practical cognitive radio data that could be applied further.

The research identified in this project is primarily theoretical, proposals or specialised analysis of selected communication networks or signals within conference papers. Therefore, it is likely that a research gap exists, concering practical simulations results and real-world implementations of cognitive radio techniques applied to microwave spectrum signals in general.

Thus, cognitive radio techniques or applications within the microwave spectrum should be investigated. However, due to time constraints, this project is primarily focused on signal detection of a microwave spectrum-based signal, as signal detection should allow for further determination of spectrum availability.

2.2 Cooperative and Non-Cooperative Signal Detection

With Cognitive Radio identified and defined, the methods of signal detection must be identified. In general, there is two approaches to signal detection that must be considered: Cooperative and Non-Cooperative signal detection.

A review by Shekhawat & Yadav (2021) provides three models of cooperative signal detection that can be reduced to:

1. Centralised Cooperative Sensing, where information from secondary users (or signal detectors) is relayed to a Fusion Center to determine whether a secondary user can transmit safely.
2. Decentralised Cooperative Sensing, where information is shared amongst secondary users, allowing secondary users to independently determine channel availability from shared knowledge.
3. Relay Assisted Cooperative Sensing, where information is either sent directly to, or via a secondary user (acting as a relay) to the Fusion Center, allowing the Fusion Center to determine whether a secondary user can transmit safely.

In these models identified by Shekhawat & Yadav (2021), Cooperative Signal Detection is demonstrated to be the cooperation between multiple secondary users to determine the channel availabilty, which is susceptible to general sensing issues and cooperative sensor network limitations in modern or upcoming communication networks, while Non-Cooperative Signal Detection (also defined as Classical Signal Detection), is signal detection conducted by a single secondary user, which is susceptible to incorrect signal detection in communication networks.

As cooperative and non-cooperative signal detection identifies whether secondary users will cooperate to detect signals, it is important to understand that non-cooperative signal detection (or spectrum sensing) can refer to signal detection (or spectrum sensing) techniques used by secondary users. The review by Shekhawat & Yadav (2015) identifies 9 variants of non-cooperative spectrum sensing methods which are listed on the following pages.

- Current Signal Detection / Spectrum Sensing Methods

  1. Short Time Fourier Transform (Windowed Fourier Transform)
     - Fourier transform applied to segemented sections of a signal via window functions to determine frequency components of a signal.
  2. Periodogram
     - Fast Fourier Transform applied to rectangular windowed signal to determine the frequency components of a signal.
  3. Multitaper Spectrum Estimation
     - Averaged Periodogram method using varying window functions to determine the frequency components of a signal with greater accuracy.
  4. Quadrature Mirror Filter Banks
     - The transmitting frequency spectrum is split into frequency bands which is monitored by two filters per frequency band to determine the presence of a signal and its frequency components.
  5. Energy Detection
     - A signal is analysed to determine its energy and compared against a signal-to-noise threshold to determine signal presence.
  6. Matched Filter Detection
     - Through use of filters and knowledge of the transmitted signal, the transmitted signal's response to an impulse is measured against a reference to determine the presence of a signal.
  7. Cyclostationary Feature Detection
     - Using knowledge of the transmitted signal, repeated signal features (such as frequency switching sequences, etc.) are used to identify the presence of a signal.
  8. Waveform Based Detection
     - Known signals transmitted at selected times in wireless systems are used to identify the presence of a signal by comparison to reference or known signals.

- Emerging Signal Detection / Spectrum Sensing Methods

  1. Wavelet Concept

     – Using filter banks, the signal is decomposed into filtered outputs known as wavelet packets (of varying resolutions) that are used to map the signal's frequency components. Variants of this concept are examined in the study by Shekhawat & Yadav (2015).

These non-cooperative methods can be used in cooperative signal detection, as the study by Armi, Saad, Arshad & Y (2010) uses energy detection (identified previously as a non-cooperative signal detection method) to determine that a cooperative signal detection system reliant on a single secondary user detecting a signal (the OR Fusion Rule) is more likely to detect a signal, over various signal to noise ratios, compared to non-cooperative signal detection and cooperative signal detection relying on all secondary users detecting a signal (the AND Fusion Rule). Additionally, increasing the number of secondary users within a cooperative sensing system is increases the likelihood of detecting a signal (Armi et al. 2010).

Thus, cooperative signal detection may determine the presence of a signal using a decision making rule (such as the OR-Rule or AND-Rule) from the information supplied by the non-cooperative methods used at any detector (or secondary user) within the sensing network.

Despite the advantages of cooperative signal detection, the needs and capabilities of a single user in a crowded transmission environment must be considered. As such, non-cooperative signal detection techniques, such as those listed previously, must be investigated further to determine applicable signal detection methods within the microwave spectrum.

2.3 Signal Detection Methods

2.3.1 General Signal Detection Theory

The following sections (Section 2.3.2 to Section 2.3.4) will discuss energy detection, eigenvalue detection and other methods of signal detection as they apply to cognitive radio and/or spectrum sensing. In general, these methods share a common approach to signal detection.

Approaching signal detection as a hypothesis problem, signal detection methods generally create a null hypothesis and alternative hypothesis (Ivanenko & Bezruk 2016). In general, the null (H_0) and alternative (H_1) hypotheses are defined as

$$H_0 : \eta$$
$$H_1 : s + \eta$$

where η is additive noise and s is the transmitted signal (Ivanenko & Bezruk 2016). Thus, the null hypothesis (H_0) suggests that only noise is present ($\eta(n)$) while the alternative hypothesis (H_1) suggests a signal, affected by correlative and additive noise is present (Ivanenko & Bezruk 2016).

The presentation of the null (H_0) and alternative (H_1) hypotheses can vary, depending on the method and research study. However, the hypotheses generally form the basis of signal detection methods which aim to confirm or deny the null hypothesis (Ivanenko & Bezruk 2016). These methods will be discussed later within their relevant sections.

Thus, in general, signal detection methods aim to confirm or deny a null hypothesis using statistical methods.

2.3.2 Energy Detection

Energy detection, as noted earlier, analyses the power of a signal and compares the power against a signal-to-noise threshold to determine signal presence (Shekhawat & Yadav 2015).

This test statistic (average power), measured against the signal to noise threshold, is calculated using

$$T_i = \frac{1}{N}\sum_{n=1}^{N}|y_i(n)|^2 \tag{2.1}$$

where N represents the number of samples, T_i represents the energy detection test statistic at the i th secondary user (or signal detector) and y_i represents the signal combined with noise at the i th secondary user (Captain & Joshi 2018, Kalamkar & Banerjee 2013, Zeng & Liang 2009).

Cooperative signal detection scales the energy detection test statistic by using the equation

$$T = \frac{1}{M}\sum_{i=1}^{M}T_i, \tag{2.2}$$

where M represents the total number of secondary users (or detectors) and all other values are as previously defined in Equation 2.1 (Captain & Joshi 2018, Zeng & Liang 2009), allowing secondary users to be easily integrated within a cooperative cognitive network.

The average power (test statistic) is compared with a pre-selected noise power value, that requires knowledge of the noise conditions to accurately estimate (Zeng & Liang 2009, Ramyea & Kasthuri 2018). If the test statistic is greater than the selected threshold, then a signal is present, otherwise a signal is not present (Ramyea & Kasthuri 2018).

It follows then, that a major issue with energy detection is noise uncertainity, where variation in noise (due to time and position within an transmitting environment) can introduce errors into the comparison between the test statistic and selected threshold as additive or correlative noise will influence the test statistic resulting in false alarms or false positives (Yawada, Wei & Kiki 2015).

The conference paper by Kalamkar & Banerjee (2013) investigates a uniform noise uncertainity model and a noise uncertainity range model (only minimum and maximum noise known) for a generalised variant of energy detection which is shown below in Equation 2.3.

$$T = \frac{1}{N}\sum_{n=1}^{N}|y_i(n)|^p \tag{2.3}$$

It was found that the conventional energy detector model (Equation 2.1) performs better than generalised energy detector models in uniform noise uncertainity, but performs similarly for a noise uncertainity of $\pm$ 0.5 dB (Kalamkar & Banerjee 2013).

Further studies have been completed on noise uncertainity for energy detection methods. The conference paper by Captain & Joshi (2018) derives and simulates (via monte-carlo) the effect of noise uncertainity on cooperative energy detection.

As such, the energy detection method described previously (Equation 2.1) should allow for the detection of signals in the microwave spectrum, assuming an accurate noise power threshold is selected.

However, as noted earlier, an eigenvalue method (maximum eigenvalue detection) was found to be more likely to detect a 5G signal when compare to energy detection and other signal detection methods Ramyea & Kasthuri (2018). Additionally, eigenvalue detection is identified to be a blind detection method, requiring no knowledge of signal characteristics or the transmission environment (Zeng & Liang 2009, Ramyea & Kasthuri 2018).

Thus, to avoid the identified issues with energy detection, eigenvalue based detection methods will be considered.

2.3.3 Eigenvalue Detection

Eigenvector Theory

Before discussing the eigenvalue detection method, the underlying theory of eigenvectors and eigenvalues must be discussed.

To start, eigenvalues and eigenvectors are the result of the equation

$$\mathbf{AX} = \lambda \mathbf{X} \tag{2.4}$$

where λ represents a non-zero scalar value (eigenvalue), $\mathbf{X}$ represents a non-trivial (non-zero) $n \times 1$ vector solution to Equation 2.4 and $\mathbf{A}$ is a square matrix $n \times n$ in size (James, Burley, Clements, Dyke, Searl & Wright 2015).

Manipulating Equation 2.4, we arrive at the characteristic equation

$$\det \left| \lambda \mathbf{I} - \mathbf{A} \right| = 0 \tag{2.5}$$

which can be solved for a polynomial series of λ (eigenvalues) that can be solved further for the individual eigenvalues of $\mathbf{A}$ (James et al. 2015).

It is important to note that eigenvectors can be real or complex, positive or negative and repeated (James et al. 2015).

Additionally, for a real symmetrical matrix of dimensions $n \times n$, the eigenvalues will be real numbers (James et al. 2015).

Applied to eigenvalue based signal detection, eigenvalue theory is used with random matrix theory to derive a signal presence threshold to be compared with eigenvalues (Zeng & Liang 2009). These eigenvalues are obtained from a statistical matrix and are representative of the variance of the signal noise (Zeng & Liang 2009).

In the case of signal absence, these eigenvalues are equal and repeated, which is the basis of signal detection for the Maximum-Minimum Eigenvalue method (Zeng & Liang 2009). This method, and the proceeding works will be discussed in the following section (Section 2.3.3).

Thus, eigenvalues are representative of signal characteristics, taken from a statistical matrix which will be discussed in the following section (Section 2.3.3).

Eigenvalue Detection Method

In the following section, the superscript T denotes the transpose of a matrix. For example, $\mathbf{x}^T$ is the transpose of $\mathbf{x}$.

So far, the Eigenvalue detection method has been identified as a reliable method to detect 5G signals (Ramyea & Kasthuri 2018), and a blind detection method that does not require prior signal knowledge (Zeng & Liang 2009).

Within eigenvalue detection, there are two main methods to consider, Maximum Eigenvalue Detection (MED) proposed by Zeng, Liang & Koh (2008) and Maximum-Minimum Eigenvalue (MME) Detection proposed by Zeng & Liang (2009). Both of these methods rely on the formation of a sample covariance matrix, formed by auto-correlating the vector containing the sampled signal (Zeng et al. 2008, Zeng & Liang 2009).

For MED, this process is completed by the auto-correlation equation,

$$\lambda(l) = \frac{1}{N_s} \sum_{m=0}^{N_s-1} x(m)x(m-l), \quad l = 0, 1, \ldots, L-1 \tag{2.6}$$

which is used to approximate the sample covariance matrix

$$\hat{\mathbf{R}}_x(N_s) \approx \begin{bmatrix} \lambda(0) & \lambda(1) & \ldots & \lambda(L-1) \\ \lambda(1) & \lambda(0) & \ldots & \lambda(L-2) \\ \vdots & \vdots & \vdots & \vdots \\ \lambda(L-1) & \lambda(L-2) & \ldots & \lambda(0) \end{bmatrix} \tag{2.7}$$

that eigenvalues can be extracted from (Zeng et al. 2008).

In equations 2.6 and 2.7, λ is used as a variable. This is how the equations are presented in the MED conference paper by Zeng et al. (2008). Please note that going forward, λ will be used to refer to eigenvalues.

Note that L represents in both MME and MED represents smoothing factor (a positive integer selected by the user), N_s represents (in both MME and MED) the number of samples, $\hat{\mathbf{R}}_x(N_s)$ or $\mathbf{R}_x(N_s)$ represents the sample covariance matrix which is a square matrix used in statistics to determine the variance and covariance of multiple variables (Wikipedia 2024*b*, CueMath 2024) and $x(n)$ represents the sample vector of the received signal (Zeng et al. 2008, Zeng & Liang 2009).

For MME (Zeng & Liang 2009), the sample covariance matrix is formed by the equation,

$$\mathbf{R}_x(N_s) = \frac{1}{N_s}\sum_{n=L-1}^{L-2+N_s} \hat{\mathbf{x}}(n)\hat{\mathbf{x}}^T(n) \tag{2.8}$$

which is reliant on the definitions:

$$\mathbf{x}(n) = [x_1(n), x_2(n), \ldots, x_M(n)]^T \tag{2.9}$$

$$\hat{\mathbf{x}}(n) = [\mathbf{x}^T(n), \mathbf{x}^T(n-1), \ldots, \mathbf{x}^T(n-L+1)]^T \tag{2.10}$$

These definitions for $\mathbf{x}$ and $\hat{\mathbf{x}}$ are for a system with M secondary users, allowing MME detection to be scaled for a cooperative cognitive network (Zeng et al. 2008, Zeng & Liang 2009).

To link these covariance matrices to signal detection, the studies (Zeng et al. 2008) and (Zeng & Liang 2009) substitutes the null hypothesis signal or signal absence ($H_0 : \eta(n)$) in place of $x(n)$ (recieved and presently transmitted signal) in both MED and MME.

From this substitution the sample covariance matrix becomes a measure of noise variance, equating all eigenvalues (of the sample covariance matrix) to the noise variance (Zeng & Liang 2009) and therefore obtaining the ratio

$$\frac{\lambda_{max}}{\lambda_{min}} = 1 \tag{2.11}$$

as the eigenvalues are all equivalent to the variance of the noise or $\lambda_{max} = \lambda_{min} = \sigma_\eta^2$ in MME detection (Zeng & Liang 2009).

(Zeng & Liang 2009) then suggests that alternate hypothesis scenarios (signal is present) will result in

$$\frac{\lambda_{max}}{\lambda_{min}} > 1. \tag{2.12}$$

where λ_{max} is the maximum eigenvalue and λ_{min} is the minimum eigenvalue of the sample covariance matrix.

In the case of Maximum Eigenvalue Detection, the maximum eigenvalue becomes equal to the noise variance in the absence of a signal ($\lambda_{max} = \sigma_\eta^2$) or greater than the noise variance in the presence of a signal ($\lambda_{max} > \sigma_\eta^2$) (Zeng et al. 2008).

In short, using the null hypothesis and alternate hypothesis signals, the MME conference paper by (Zeng & Liang 2009) and the MED conference paper by Zeng et al. (2008), equate eigenvalues to the signal noise variance and determine eigenvalue ratios or criteria according to the method.

Both methods (Zeng et al. 2008, Zeng & Liang 2009) use Random Matrix Theory and the first order Tracy-Widom distribution (a random matrix related distribution for the maximum eigenvalue) to determine an equation for a threshold unique to each method. These thresholds must be exceeded by either the MME ratio or maximum eigenvalue (MED method) for a signal to be present (Zeng et al. 2008, Zeng & Liang 2009).

Note that within the MED conference paper by (Zeng et al. 2008), the signal presence criteria is given as

$$\lambda_{max} > \gamma\sigma_\eta^2 \tag{2.13}$$

which must be true for a signal to be present.

Note that γ is the signal presence threshold, σ_η^2 is the variance of the noise and λ_{max} is the maximum eigenvalue of the MED covariance matrix (see Equation 2.7, where λ represents an autocorrelation).

Alternatively, the MME conference paper states the signal presence criteria as

$$\frac{\lambda_{max}}{\lambda_{min}} > \gamma \tag{2.14}$$

which must be true for a signal to be present (Zeng & Liang 2009).

Thus, the Maximum-Minimum Eigenvalue (MME) detection method will be used in this project, as MED is reliant on noise variance. Further discussion on MME detection in relation to this project (testing, application, formulae) can be seen in Section REF of the Methodology.

Other detection methodologies have been dismissed for requiring prior knowledge of the signal or careful selection of thresholds to avoid false positives.

Within the eigenvalue detection field, there are many applications and modifications. A small set of these applications and modifications will be discussed in the following sections (Section 2.3.3 and Section 2.3.3).

Modifications

Within eigenvalue based sensing, there are several modifications, variants and alternatives to both MME and MED methods. Some modifications, variants or alternatives to MME and MED methods are listed below.

The MME conference paper by Zeng & Liang (2009) discussed previously also presents Energy with Minimum Eigenvalue (EME) detection as an applicable signal detection method. The EME method compares the ratio of the average (recieved) signal power and minimum eigenvalue (of the sample covariance matrix) against a threshold unique to EME (Zeng & Liang 2009). Within the conference paper, EME was found to be a viable during testing (Zeng & Liang 2009). However, due to the complexity of the threshold equation, this method was not chosen for testing.

The conference paper by Ali, Zhao, Jin & Yoo (2019) presents and simulates several difference eigenvalue detection methods and found that the Maximum-Minimum Eigenvalue Sum (MMES) and Product (MMEP) detection methods were the best performers in a multiple primary and secondary user enivronment.

A conference paper by Sharma, Chatzinotas & Ottersten (2014) revisited the derivation of the MED threshold, suggesting and testing an alternative threshold that was found to be significantly more accurate for correlated noise in -10 dB Signal-to-Noise Ratio and higher. However, the alternative threshold introduces significant complexity into the threshold calculations as two definite integrals have been incorporated (Sharma et al. 2014).

Applications

A conference paper by Ramyea & Kasthuri (2018) identifies that an eigenvalue detector is able to detect a 5G primary user more reliably compared to other spectrum sensing methods. It should be noted that the eigenvalue detection method used was Maximum Eigenvalue Detection (MED) as this conference paper has been discussed previously (Ramyea & Kasthuri 2018).

For Maximum-Minimum Eigenvalue detection, a conference paper by Althaf & Prema (2018) tested Maximum-Minimum Eigenvalue (MME), Energy with Minimum Eigenvalue (EME), Energy Detection and a Covariance Matrix detection method in a real-world environment with signal transmitting with a centre frequency of 93.5 MHz.

The results of this conference paper found MME to present the highest possible probability of detection for the lowest possible probability of false alarm (Althaf & Prema 2018). While a validation of MME, the central transmitting frequency was outside of the microwave spectrum, suggesting the paper is of limited application to the microwave frequency bands.

2.3.4 Alternative Methods

As the eigenvalue detection method is selected signal detection method, the following segments (Section 2.3.4 to Section 2.3.4) will provide a limited and brief overview of an unmentioned detection method and its applications.

Deep-Learning Applications

Deep-learning, the process of training neural networks to analyse information in stages (Google 2024), has been successfully applied to the cognitive radio as signal detection or spectrum sensing. Deep-learning is applied by training a neural network on a large dataset to learn and identify characteristics of the dataset according to the dataset's labels (Google 2024).

In the journal paper by Wei, Zheng, Zhou, Zhang, Lou, Zhao & Yang (2022), two deep-learning neural network methodolgies are proposed to detect Direct Sequence Spread Spectrum (DSSS) signals which can appear as white noise. As signal detection is formulated as a binary problem with a null and alternative hypothesis (signal is absent and signal is present respectively) deep-learning is readily applicable (Wei et al. 2022).

The two deep-learning neural networks were trained on DSSS and white noise signals, allowing the neural network (with adequate time training) to remember the difference and correctly identify signals at a greater accuracy in all cases (such as interference, multipathing, pink noise, etc.) when compared to traditional autocorrelation detection (Wei et al. 2022).

Another journal paper by Kot, Teh, Razul & Su (2022) proposes and tests a hybrid neural network design against known deep-learning and signal detection methods in a non-cooperative cognitive radio environment. Operating in uncertain noise, unknown modulation and a rayleigh fading channel (attenuation in the form of a rayleigh distribution) the proposed neural network design is able to detect signals on-par with trained models (in the case of unknown modulation) or better than current methods (Kot et al. 2022).

Statistical Tests

Statistical tests, such as the Kolmogorov-Smirnov and Pearson Chi-Square tests can be applied to spectrum sensing by evaluating the probability distribution of a signal's samples against a known distribution as Goodness-Of-Fit test (de Carvalho, Lopes & Alencar 2015*a*).

The paper by de Carvalho et al. (2015*a*) compares three known statistic tests (Kurtosis, Skewness and Jarque-Bera) by their performance in a rayleigh fading channel (attenuation in the form of a rayleigh distribution) which demonstrated the probability of detection requiring an increase in SNR by approximately 15 dB to reach parity with non-faded results. Further testing in other fading channels was announced (de Carvalho et al. 2015*a*), but the application and viability of statistical tests as a spectrum sensing techniques appears to be limited. Other spectrum sensing techniques may provide easier implementation and better results.

Further investigation has yielded similar niche cases, of research applying statistical tests to niche distributions outside the scope of this project, such as the study by Luo, Wang, Zhang & Luo (2015) applying a variation of the Kolmogorov-Smirnov test to a symmetric alpha stable noise channel.

Due to the complex and niche nature of these statistical test applications, further review will be not be conducted in favour of blind detection methods such as energy detection or eigenvalue detection.

2.4 Microwave Frequency Signals

As the microwave frequency spectrum spans 300 MHz to 300 GHz (North American Space Administration 2010), the allocated frequency bands and their usage within this spectrum are identifiable by the *Australian Radiofrequency Spectrum Plan* 2021.

As Australia is a member of the International Telecommunications Union (ITU), frequency band allocations must be made to accomodate both ITU Radio Regulations and Australian Government allocations. (Communications & Authority 2024, Australian Radiofrequency Spectrum Plan 2021).

Presented within the *Australian Radiofrequency Spectrum Plan* 2021, the microwave spectrum has been allocated from 300 MHz to 275 GHz, with general usage and frequency bands listed. For example, the frequency band 399.9 to 400.05 MHz (within the microwave spectrum) lists the band being allocated to "MOBILE-SATELLITE (Earth-to-space)" with footnotes regarding ITU and Australia usage (*Australian Radiofrequency Spectrum Plan* 2021, pg. 45).

However, due to limitations in available datasets, only 5G New Radio, LTE and two variants of Wi-Fi 6 can be discussed and analysed as these are present in the dataset by Subray (2023). The following sections (Section 2.4.1 to Section 2.4.3) cover the general usage of these signals and their general frequency bands (or operational frequencies) within Australia. However, it should be noted, that with the possible exception of 5G New Radio (Section 2.4.1), all signals should exist within the microwave frequency spectrum.

2.4.1 5G New Radio

The 5G New Radio (5G NR) is an Orthogonal Frequency Division Multiplexing (OFDM) transmission standard to allow users to access wireless communications networks (Wikipedia 2024*a*). Within Australia, 5G NR can currently operate on various frequency bands between 5 MHz and 1 GHz for various telecommunications providers (Wikipedia 2024*c*). As such, it is possible that any dataset containing 5G NR signals may contain 5G NR signals outside of the microwave spectrum.

2.4.2 LTE

Long-Term Evolution (LTE) is a wireless communications standard using Orthogonal Frequency-Division Multiple Access (OFDMA) to allow users to access wireless communication networks (Gills, Jones & Beaver 2024). Within Australia, LTE can currently operate on the frequency bands at 700 MHz, 850 MHz, 900MHz, 2.1 GHz, 2.3 GHz and 2.6 GHz (Wikipedia 2024*d*). Thus, LTE should be present within the microwave frequency spectrum during Australian operation.

2.4.3 Wi-Fi 6

Wi-Fi 6 or IEEE 802.11ax is an Orthogonal Frequency Division Multiple Access (OFDMA) wireless communication standard typically operating around 2.4 or 5 GHz (RF Wireless World 2024*b*, Intel Technologies 2024). Within the available dataset, Wi-Fi 6 MCS6 and Wi-Fi 6 MCS7 signals have been included (Subray 2023), which uses 64 Quadrature Amplitude Modulation at different bit and code rates (RF Wireless World 2024*a*). Thus, Wi-Fi 6 should be present within the microwave frequency spectrum during Australian operation.

2.4.4 Industrial, Scientific and Medical Bands

Within the *Australian Radiofrequency Spectrum Plan* 2021, there are 13 ISM Bands from ITU and Australian Government allocations, which can be seen in Table 2.1 on Page 25. According to the *Australian Radiofrequency Spectrum Plan* 2021, several of these bands require the user to operate at their own risk, subject to ITU and Australian provisions (*Australian Radiofrequency Spectrum Plan* 2021, pg. 98 and pg. 177).

As such, these bands must be investigated to determine whether significant and repeatable interference is occurring. An investigation by Wituski & Dietl (2020) found that parallel networks and conflicting transmission standards (WLAN and Blutetooth specifically) were interfering within the 2.4 GHz ISM Band. Additionally, an Italian Hospital's usage of the 2.4 GHz band was recorded, allowing for the modelling of typical interference and their distributions (Mucchi & Carpini 2014).

However, it should be noted due to the age of the Italian Hospital conference paper (published in 2014) and the situation review by Wituski & Dietl (2020) that these studies provide a limited view into potential interference sources and/or conflicting transmission schemes. The Italian Hospital study in particular will not feature potential interference from Internet of Things devices, the profileration of Bluetooth/Wi-Fi capable devices and/or new standard in communication schemes. Conversely, due to the localised nature of the interference investigation by Wituski & Dietl (2020), it is expected that several competing transmission schemes have been omitted.

Thus, it expected that the 2.4 GHz ISM band (2 400 - 2 500 MHz ISM band in Table 2.1 on Page 25) has the potential for significant interference. Thankfully, the Subray (2023) dataset contains Wi-Fi 6 which can operate at the 2.4 GHz ISM band. Therefore, signal detection techniques can be applied to a potential interfering signal type within a ISM band.

Table 2.1: Table of ISM Bands within the Interational Telecommunication Union and Australia

Frequency Band	Units	Centre Frequency
6 765 - 6 795	kHz	6 780
13 553 - 13 567	kHz	13 560
26 957 - 27 283	kHz	27 120
40.66 - 40.70	MHz	40.68
433.05 - 434.79	MHz	433.92
902 - 928	MHz	915
918 - 926	MHz	922
2 400 - 2 500	MHz	2 450
5 725 - 5875	MHz	5 800
24 - 24.25	GHz	24.125
61 - 61.5	GHz	61.25
122 - 123	GHz	122.5
244 - 246	GHz	245

2.5 Summary

In this literature review, it was identified that:

1. Cognitive Radio (Section 2.1)
    - Cognitive radio is the smart detection and utilisation of spectrum holes (available frequency bands) for transmission without interfering with primary users or critical infrastructure.
    - The majority of the featured research in the literature review is theoretical, proposals or specialised analysis.
    - There is a lack of practical simulation results and real-world testing/data.
2. Cooperative and Non-Cooperative Signal Detection (Section 2.2)
    - In cooperative signal detection, secondary users (opportunistic users not normally part of the frequency band) cooperate to detect available spectrum holes.
    - Cooperative signal detection is generally more reliable at detecting signals.
    - Multiple signal detection methods exist.
    - Non-cooperative signal detection refers to signal detection methods for a single user.
    - Cooperative signal detection can use non-cooperative signal detection methods at each secondary user.
    - To consider the needs of a single user, non-cooperative signal detection was selected.

3. Signal Detection Methods (Section 2.3)

    - Signal detection uses a null (signal absent) and alternate (signal present) hypothesis.
    - Energy detection compares average power with a chosen noise power threshold.
    - Energy detection is susceptible to noise uncertainity (changing noise power).
    - The eigenvalue method was selected due to higher chance of detecting 5G signals and its ability to detect without prior knowledge of a signal.
    - Eigenvalue and eigenvector theory was briefly covered.
    - Maximum Eigenvalue Detection (MED) and Maximum-Minimum Eigenvalue (MME) methods were discussed.
    - Eigenvalues are obtained from the sample covariance matrix in eigenvalue detection.
    - The eigenvalues correspond to noise power and therefore form the basis of eigenvalue detection.
    - The Maximum-Minimum Eigenvalue signal detection method was selected.
    - Various modifications and applications of eigenvalue signal detection were discussed.
    - Deep-Learning signal detection is more successful in identifying signals in a variety of circumstances.
    - Statistical tests can be applied to signal detection.

4. Microwave Frequency Signals (Section 2.4

    - The microwave spectrum spans 300 MHz to 275 GHz in Australia.
    - Only 5G New Radio, LTE and Wi-Fi 6 were discussed due to limitations in the chosen dataset.
    - It is possible the dataset contains 5G New Radio signals that operate outside the microwave spectrum.
    - The Industrial, Scientific and Medical (ISM) bands in Australia were identified.
    - The 2.4 GHz band may have significant interference.
    - Wi-Fi 6 operates within the 2.4 GHz band.

Chapter 3

Methodology

3.1 Eigenvalue Detection of Signals

In the paper by Zeng & Liang (2009), an algorithm to calculate the Maximum-Minimum Eigenvalue (MME) is listed with some of the require equations missing.

This algorithm for MME is:

1. Calculate the sample covariance matrix.
2. Obtain the minimum and maximum eigenvalue of the sample covariance matrix.
3. Calculate the eigenvalue ratio ($\lambda_{max}/\lambda_{min}$).
4. Calculate the threshold (γ).
5. Compare the eigenvalue ratio to the threshold. If the eigenvalue is greater than the threshold, a signal is present.

However, in it's basic state, MME is scaled for multiple secondary users or $M > 1$.

In the case of a single secondary user, the sample covariance matrix must be redetermined.

First, listing the equation for the sample covariance matrix:

$$\mathbf{R}_x(N_s) = \frac{1}{N_s} \sum_{n=L-1}^{L-2+N_s} \hat{\mathbf{x}}(n)\hat{\mathbf{x}}^T(n) \tag{3.1}$$

Note that for MME detection, N_s is the number of samples, $\mathbf{R}_x$ is the sample covariance matrix, L is the smoothing factor and superscript T represents the transpose of a matrix.

Using the definitions for $\mathbf{x}(n)$ and $\hat{\mathbf{x}}(n)$ (where $x_i(n)$ represents the ith signal sample vector) from the paper by Zeng & Liang (2009), we obtain:

$$\mathbf{x}(n) = [x_1(n), \ldots, x_M(n)]^T \tag{3.2}$$

$$\hat{\mathbf{x}}(n) = [\mathbf{x}^T(n), \mathbf{x}^T(n-1), \ldots, \mathbf{x}^T(n-L+1)]^T \tag{3.3}$$

As $x_i(n)$ represents the ith signal sample vector, and a version of MME for a single secondary user is desired, let $M = 1$. Subsituting, we obtain:

$$\mathbf{x}(n) = x(n) \tag{3.4}$$

$$\hat{\mathbf{x}}(n) = [x^T(n), x^T(n-1), \ldots, x^T(n-L+1)]^T \tag{3.4 $\rightarrow$ 3.3}$$

To calculate the sample covariance matrix, lets determine the range of $\hat{\mathbf{x}}$ that will be used.

Substituting the range $n = L-1$ to $n = L-2+N_s$ into Equation 3.4 $\rightarrow$ 3.3 we obtain:

$$\hat{\mathbf{x}}(L-1) = [x^T(L-1), x^T(L-2), \ldots, x^T(0)]^T \tag{3.5}$$

$$\hat{\mathbf{x}}(L-2+N_s) = [x^T(L-2+N_s), x^T(L-3+N_s), \ldots, x^T(N_s-1)]^T \tag{3.6}$$

As the sample covariance matrix is a square symmetrical matrix, we can substitute Equation 3.4 $\rightarrow$ 3.3 into Equation 3.1 to arrive at an equation for sample covariance matrix of a single secondary user.

Substituting Equation 3.4 $\rightarrow$ 3.3 into Equation 3.1 we should obtain

$$\mathbf{R}_x(N_s) = \frac{1}{N_s} \sum_{n=L-1}^{L-2+N_s} \begin{bmatrix} x(n) \\ x(n-1) \\ \vdots \\ x(n-L+1) \end{bmatrix} \begin{bmatrix} x(n) & x(n-1) & \ldots & x(n-L+1) \end{bmatrix} \tag{3.7}$$

which will generate a square symmetrical $\mathbf{R}_x$ matrix.

It should be noted, that the upper limit of the sum in Equation 3.7 will result in a index value of $L-2+N_s$ which exceeds N_s if $L > 2$. As N_s represents the total number of samples within the recieved signal (or number of samples total), Equation 3.7 begins to fall apart.

This is solved by treating Equation 3.7 as a auto-correlation, which was the process to determine the sample covariance matrix of the Maximum Eigenvalue Detector (Chapter 2 Section 2.3.3). This means, any value of n that exceeds N_s must result in zero.

Expressed mathematically, accounting for Python array indexing,

$$x(n) = \begin{cases} x(n) & 0 \leq n \leq N_s - 1 \\ 0 & n \geq N_s \end{cases} \tag{3.8}$$

must be observed for Equation 3.7 to result in a square symmetrical matrix that represents the sample covariance matrix.

With an equation determined for the sample covariance matrix (Equation 3.7), the algorithm and its required equations can be restated.

Copying over the required equations from Zeng & Liang (2009), the algorithm for Maximum-Minimum Eigenvalue detection becomes:

1. Calculate the sample covariance matrix using

$$\mathbf{R}_x(N_s) = \frac{1}{N_s}\sum_{n=L-1}^{L-2+N_s} \begin{bmatrix} x(n) \\ x(n-1) \\ \vdots \\ x(n-L+1) \end{bmatrix} \begin{bmatrix} x(n) & x(n-1) & \dots & x(n-L+1) \end{bmatrix} \tag{3.7}$$

   for

$$x(n) = \begin{cases} x(n) & 0 \leq n \leq N_s - 1 \\ 0 & n \geq N_s \end{cases} \tag{3.8}$$

2. Obtain the minimum and maximum eigenvalue of the sample covariance matrix $(\mathbf{R}_x(N_s))$.

3. Calculate the eigenvalue ratio $(\lambda_{max}/\lambda_{min})$.

4. Calculate the threshold (γ) using

$$\gamma = \frac{\left(\sqrt{N_s} + \sqrt{ML}\right)^2}{\left(\sqrt{N_s} - \sqrt{ML}\right)^2}\left(1 + \frac{\left(\sqrt{N_s} + \sqrt{ML}\right)^{-2/3}}{(N_s ML)^{1/6}} F_1^{-1}(1 - P_{fa})\right) \tag{3.9}$$

5. Compare the eigenvalue ratio to the threshold, if $\lambda_{max}/\lambda_{min} > \gamma$ then a signal is present. Otherwise, a signal is absent.

Note: L is the smoothing factor, n is the sample number, $x(n)$ is the recieved signal, N_s is the number of samples, $\mathbf{R}_x$ is the sample covariance matrix, M is the number of secondary users ($M = 1$), P_{fa} is the probability of false alarm and F_1^{-1} is the Tracy-Widom order 1 inverse cumulative distribution function.

Thus, the algorithm for Maximum-Minimum Eigenvalue detection is listed above.

Note: Due to large signal datasets, the MME Algorithm will be applied to N_s sized frames of the dataset with a smoothing factor of L. The probability of false alarm will be set to 10% ($P_{fa} = 10\%$) and the number of secondary users will remain $M = 1$.

Note 2: The Threshold (γ) is constant and only varies when N_s or L changes.

3.2 Collection of Data

During the project, it became apparent that a microwave spectrum dataset could not be created in a timely manner. As such, the dataset collection turned to available online datasets.

As some dataset websites were unavailable, the search expanded to sites such as Kaggle.com, a website that hosted free creative-commons licensed datasets. Doing some minor searching, the Subray (2023) dataset was found and downloaded.

The 'Real-World Wireless Communication Dataset', or Subray (2023) dataset, contained a set of signals using Wi-Fi 6, LTE and 5G New Radio. As these signals are within the microwave spectrum (assuming that the 5G New Radio data was recorded within the microwave frequency spectrum according to previous observations in Section 2.4 of the Literature Review) and the dataset was stored in an integer format, the dataset was perfect for signal detection testing.

If the 'Real-World Wireless Communication Dataset' contained Signal-to-Noise ratio data instead of a sampled waveform, then another dataset would have been chosen.

Thus, the 'Real-World Wireless Communication Dataset' was chosen for this project due to the integer format and microwave spectrum signals.

3.3 Creation of a Simulation Environment

Due to the availability of online datasets, the signal detection process will be simulated within using Python 3.9.19 within the SPYDER IDE. To ensure that the analysis was completed in a timely manner, the scientific computing Python modules NumPy and SciPy were downloaded, installed and incorporated into the constructed Python code.

The Matplotlib Python module was included to generate figures and a Tracy-Widom distribution module was included to calculate the inverse cumulative distributio function for a Tracy-Widom distribution. In the eigenvalue detection method, this is the value $F_1^{-1}(1 - P_{fa})$ which must be calculated to determine the threshold.

Using these modules and an available Tracy-Widom distribution Python module, the simulation environment was created. As the Subray (2023) dataset stores the signals in the signed NumPy 16-bit integer format, the Python code was constructed to safely extract the data from the *.data* files provided in the dataset.

With the dataset safely extracted, the Python code should be able to construct a figure for each of the signals in the dataset and save the figures as *.png* files, confirming that the dataset can be used and files can be saved successfully.

To ensure the code correctly detects signals, a signal should be manually selected to calibrate the variables L and N_s. This process should result a signal being detected in areas of high sample value (high signal power) and a detectable signal absence in areas of low or zero sample value.

Note: For large sample size datasets, the python code should break the waveform up into N_s sized frames and determine if there is a signal present within each frame.

Additionally, a figure of the signal waveform should be created with highlight areas depicting where the signal was detected.

To perform the Maximum-Minimum Eigenvalue Detection method, the Maximum-Minimum Eigenvalue (MME) detection algorithm described in Section 3.1 will be used. This algorithm should require selected values for N_s, L and P_{fa} (Number of Samples, Smoothing Algorithm and Probability of False Alarm respectively) as $M = 1$ has been selected. Calibration wil be required to determine suitable values for N_s and L, as P_{fa} has been selected previously to be 10%.

Note: For large datasets, the code must be able to break the dataset into frames of size N_s and perform the MME algorithm over the entire dataset.

After calibration the Maximum-Minimum Eigenvalue detection method should be functioning and tested against all signals of the Subray (2023) dataset as a baseline for further work with additive or correlated noise.

For signal detection within the simulation environment to be relevant to real-world scenarios, the Python code should be able to 'inject' a variety of signal noise strengths and patterns into the tested signal patterns. Thus, after loading the data it is imperative that a variety of noise profiles can be injected into the loaded datasets for testing. This should result in several variations of the same dataset for testing, allowing for the relationship between various noise profiles and the accuracy of signal detection to be determined. Note that noise profiles will vary in terms of strength and frequency distribution. I.E., Gaussian and other forms of additive noise will be considered.

By detecting signals in a variety of noisy environments, it is hoped that 'real-world' signal detection can be approximated within the simulation environment. Where possible, it is hoped that the dataset loading, noise injecting and signal detection testing methods can be automated to ensure that the STE can become an automated simulation environment to remove manual input requirements.

Should time permit, it is hoped that an automated testing environment would be able to generate new signal datasets for testing using the previously loaded signals from the online dataset. The new signal datasets should vary between dead-air (no signal transmitted) and recieved signal data with a variety of injected noise to simulate real-world signal detection.

The dead-air interval between signals should be randomised for each new dataset. Ideally, this system would randomly select signals from online datasets and randomly determine the interval between each signal to generate a new dataset for a selected number of samples. For example, the new dataset system should be able to select from known signal datasets and known noise patterns to generate a dataset of any number of sample points.

This method of additive noise injection should reveal the extent Maximum-Minimum Eigenvalue detection can detect the signal under additive or correlative noise.

If time permits further work, then the presence of multiple overlapping signals in a variety of noisy environments should also be generated using the new signal dataset generation method.

In summary, the Python code (in Appendix C) should be able to:

- Extract the dataset.
- Graph the dataset and save the figure to disk.
- Perform Maximum-Minimum Eigenvalue Detection.
- Save the Eigenvalue Ratio, Threshold and Detected Signal Presence results to disk.
- Graph the dataset, highlight the signal detected areas and save the figure to disk.
- Be calibrated from manual selections of L and N_s.
- Add noise (of selected distributions) to the dataset signals.

If possible, the Python code should be able to automatically create custom signals using the dataset signals, dead-air and additive noise.

3.4 Expected Results and Data Analysis

Note: The average, minimum and maximum eigenvalue ratios will be discussed. A general representation has been included below.

$$\lambda_{Avg.Ratio} = \text{Average}\left(\frac{\lambda_{max}}{\lambda_{min}}\right) \tag{3.10}$$

$$\lambda_{Min.Ratio} = \text{Minimum}\left(\frac{\lambda_{max}}{\lambda_{min}}\right) \tag{3.11}$$

$$\lambda_{Max.Ratio} = \text{Maximum}\left(\frac{\lambda_{max}}{\lambda_{min}}\right) \tag{3.12}$$

From the proposed Python code, we can expect three sets of results:

1. Calibration Results
2. Signal Detection Results
3. Added Noise Results

Each of the three results sets should output the following as individual *.txt* files:

1. The Eigenvalue Ratios ($\lambda_{max}/\lambda_{min}$) for all Samples/Frames of a Signal.
2. The Calculated Threshold Constant (γ).
3. Signal Presence (as a Boolean) for all Samples/Frames of a Signal.

From these results, we can determine the average, minimum and maximum eigenvalue ratio for the overall signal. This should provide a measure to compare the ease of signal detection between signal types. For example, the 5G-NR being harder to detect than Wi-Fi 6, due to a lower average eigenvalue ratio.

Note: The eigenvalue ratio, threshold and signal presence appear to be dimensionless.

For the calibration results, there are four factors to consider, which are listed below.

- L or Smoothing Factor
- M or Number of Secondary Users
- N_s or Number of Samples
- P_{fa} or Probability of False Alarm

From these factors, $M = 1$ and $P_{fa} = 10\%$ will be held constant. The variation of L and N_s should impact either the eigenvalue ratios, threshold (which varies due to N_s and L being used to calculate γ) or the boolean signal detection values.

Thus, depending how the results change from varying L and N_s, the Python code will be calibrated for optimal signal detection. Graphs will be used to visually confirm signal presence.

Otherwise, the noise results and signal detection results will mainly be compared against each other to determine the impact of noise on either signal presence or the eigenvalue ratios. It is expected that additive and correlative noise will negatively affect the signal detection code.

3.5 Summary

1. Eigenvalue Detection of Signals (Section 3.1)
   - The Maximum-Minimum Eigenvalue (MME) algorithm for M secondary users was stated, with some equations missing.
   - The MME Algorithm for $M = 1$ secondary users was derived.
   - The MME Algorithm for $M = 1$ secondary users was stated with all equations.
2. Collection of Data (Secton 3.2)
   - Could not construct a dataset in a timely matter.
   - Began searching for datasets available on the internet.
   - Found and selected a dataset stored in Integer format containing microwave spectrum signals.
3. Simulation Environment (Section 3.3)
   - In summary, the Python code (in Appendix C) should be able to:
     - Extract the dataset.
     - Graph the dataset and save the figure to disk.
     - Perform Maximum-Minimum Eigenvalue Detection.
     - Save the Eigenvalue Ratio, Threshold and Detected Signal Presence results to disk.
     - Graph the dataset, highlight the signal detected areas and save the figure to disk.
     - Be calibrated from manual selections of L and N_s.
     - Add noise (of selected distributions) to the dataset signals.
   - If possible, the Python code should be able to automatically create custom signals using the dataset signals, dead-air and additive noise.

4. Expected Results (Section 3.4)

    - Eigenvalue ratios will be compared to determine signal detectability.
    - Signal presence will be boolean.
    - All results will yield eigenvalue ratios, signal presence and the calculated threshold for the length of a signal.
    - Graphs will be used to visually confirm signal presence.
    - Noise is suspected to negatively impact signal detection.

Chapter 4

Results

4.1 Code Completeness Statement

The Python code (in Appendix C) is able to:

- Extract the dataset.
- Graph the dataset and save figures to disk.
- Perform Maximum-Minimum Eigenvalue Detection.
- Save the Eigenvalue Ratio, Threshold and Detected Signal Presence results to disk.
- Graph the dataset, highlight the signal detected areas and save the figure to disk.
- Be calibrated from manual selections of L and N_s.
- Add noise (Gaussian) to a dataset manually.

The Python code (in Appendix C) is unable to:

- Automatically create custom signals using the dataset signals, dead-air and additive noise.

4.2 Extraction of Data

Using the Python code constructed (Appendix C), the waveforms from the Subray (2023) dataset have been successfully extracted and graphed on the following pages (Page 43 to 50) as Figures 4.1 to 4.15.

Note that the dataset waveforms are 40 000 000 samples long and are in NumPy's signed 16-bit integer format. As such, the value of each sample can vary between -32 768 and +32 767.

Due to the large sample size, the Maximum-Minimum Eigenvalue Algorithm was applied to each waveform frame by frame. The definition of N_s was changed to the Number of Samples per frame.

By inspection of the dataset waveforms, Wi-Fi 6 MCS7 Signal 1 was chosen to calibrate the signal detection code, as there appeared to be several sections of inactivity (no signal present) within the waveforms graph (Figure 4.13 on Page 49). However, as a result of later calibration testing using Wi-Fi 6 MCS7 Signal 1 and custom waveforms derived from Wi-Fi 6 MCS7 Signal 1, it was discovered that these sections were found to contain some form of signal, by extrapolating results on a low-activity section of the waveform.

Thus, the waveforms of the Subray (2023) dataset has been successfully extracted and graphed on the following pages (Page 43 to 50) as Figures 4.1 to 4.15 using Python code (Appendix C). Wi-Fi 6 MCS7 Signal 1 was chosen to calibrate the MME Signal Detection code (Appendix C).

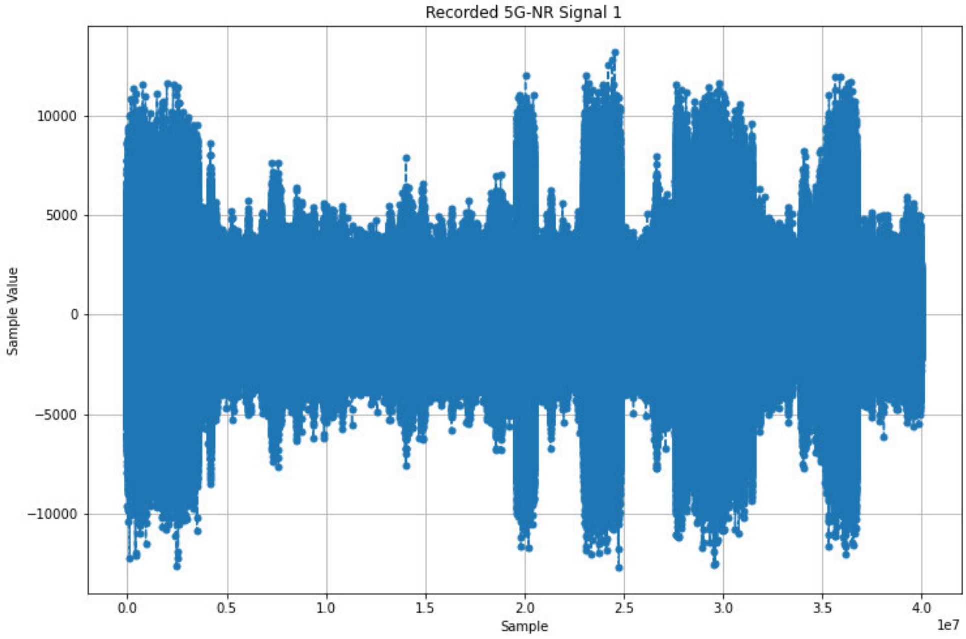

Figure 4.1: Graph of 5G New-Radio Signal 1

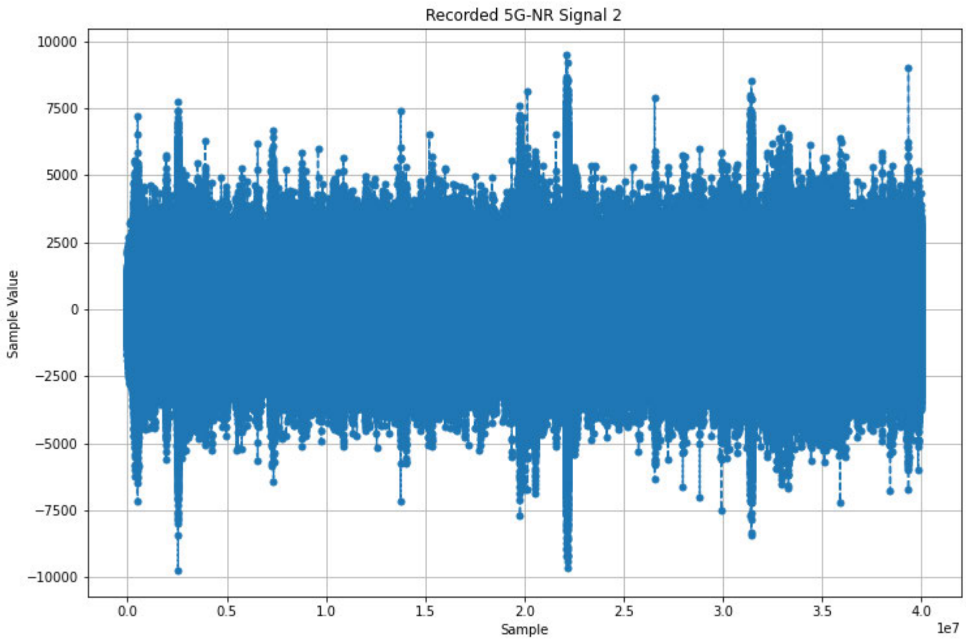

Figure 4.2: Graph of 5G New-Radio Signal 2

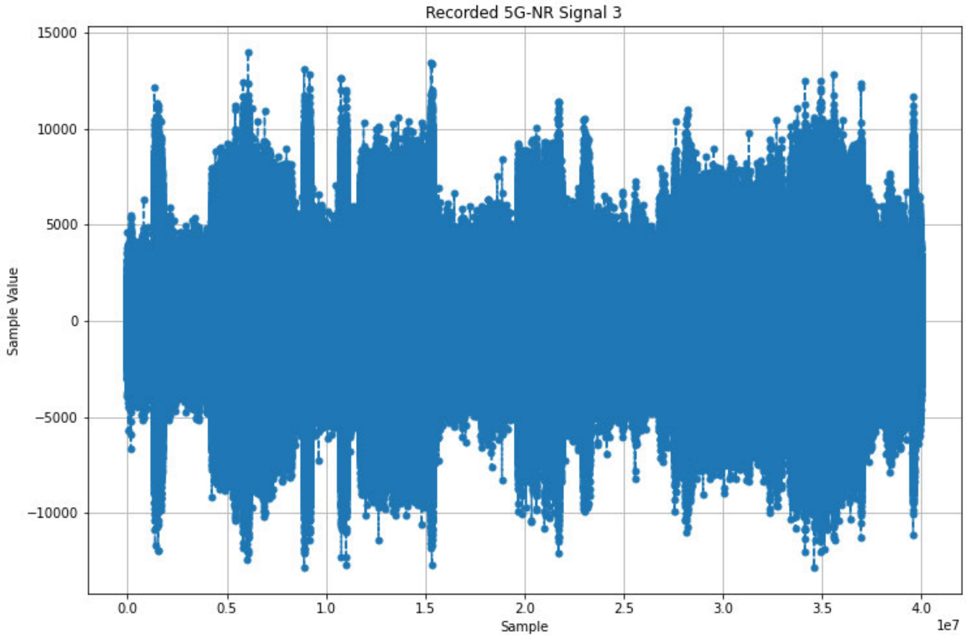

Figure 4.3: Graph of 5G New-Radio Signal 3

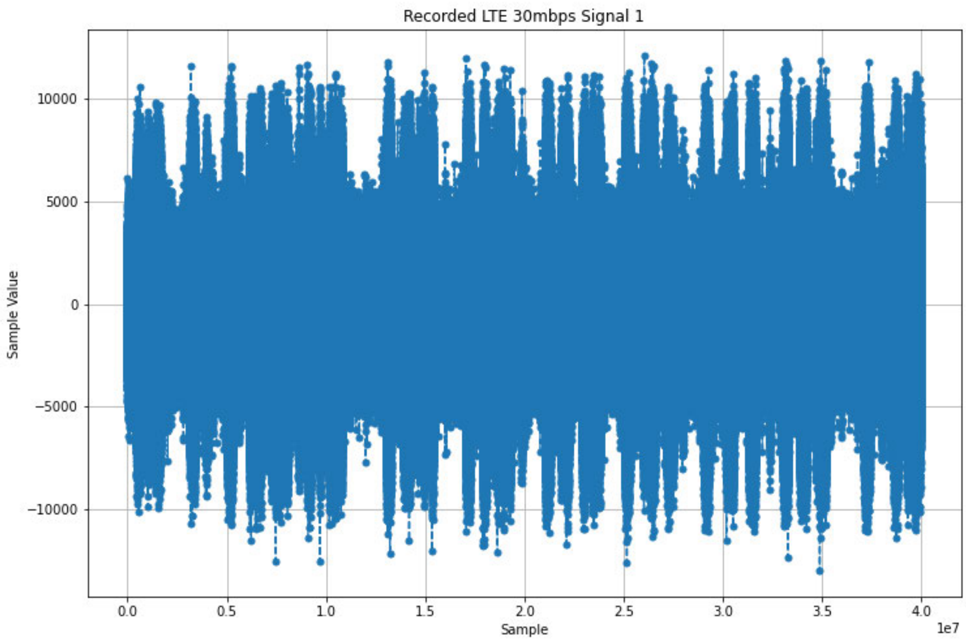

Figure 4.4: Graph of LTE 30mbps Signal 1

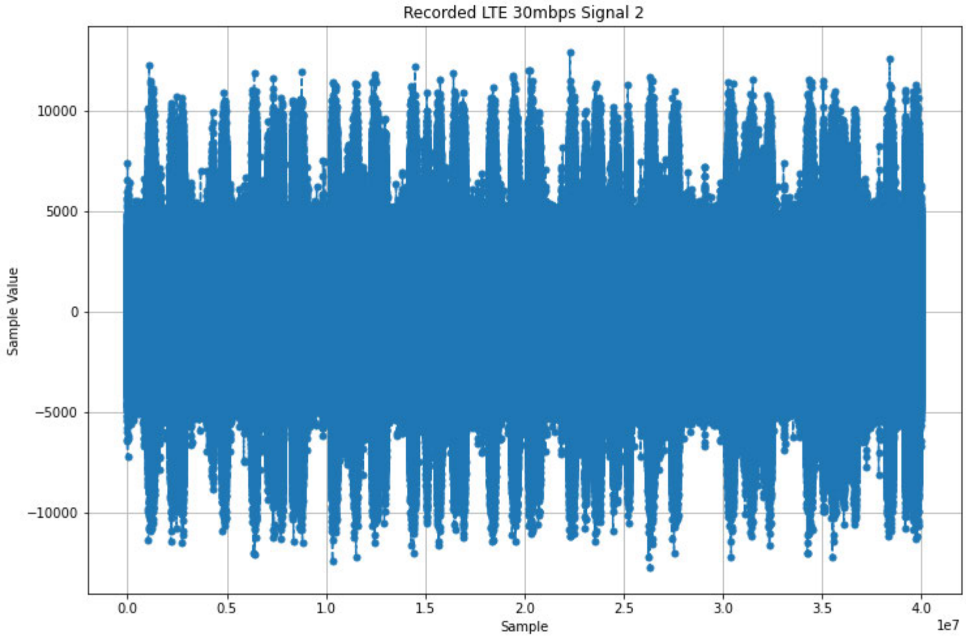

Figure 4.5: Graph of LTE 30mbps Signal 2

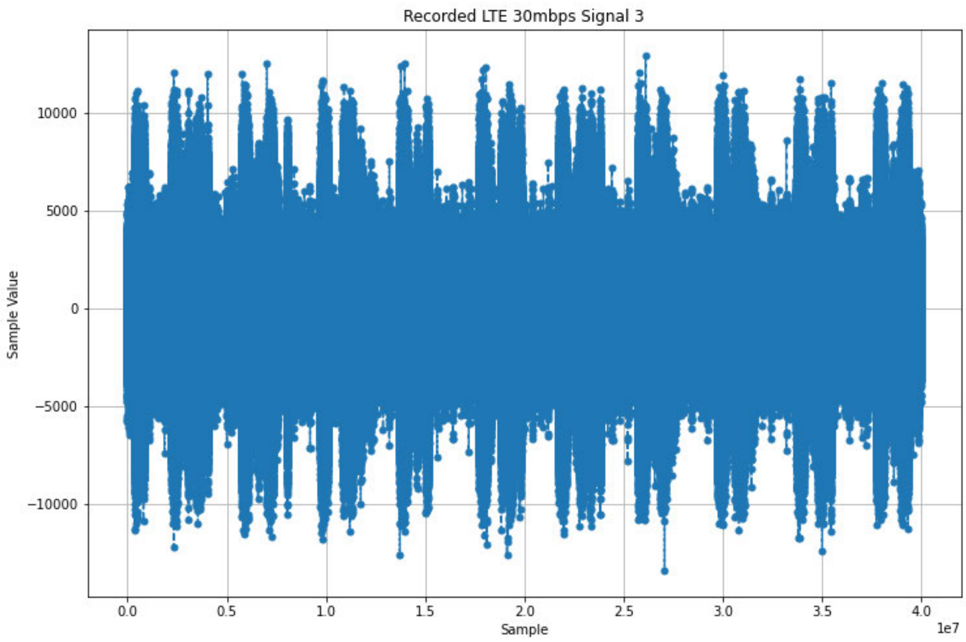

Figure 4.6: Graph of LTE 30mbps Signal 3

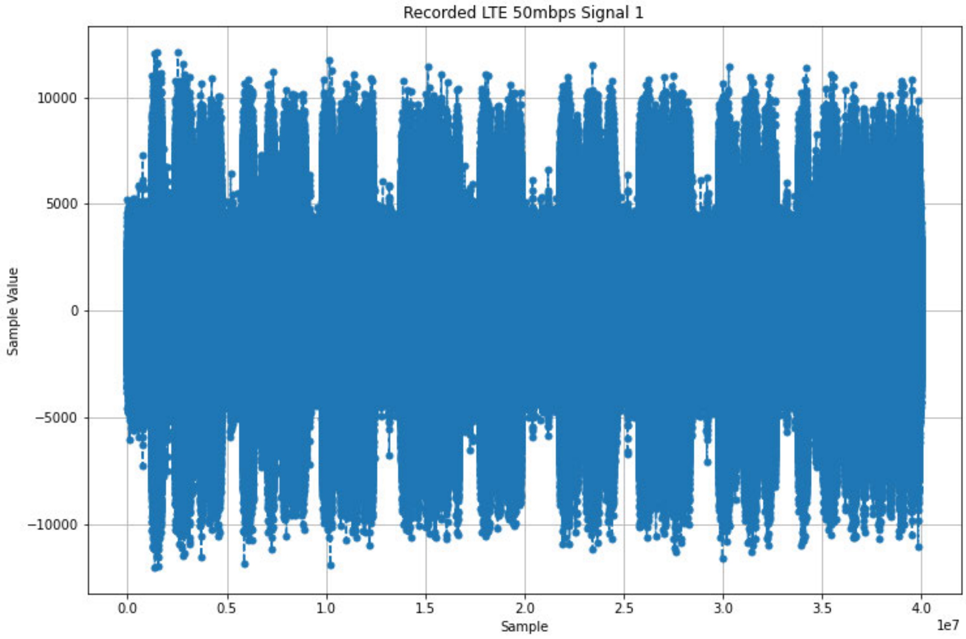

Figure 4.7: Graph of LTE 50mbps Signal 1

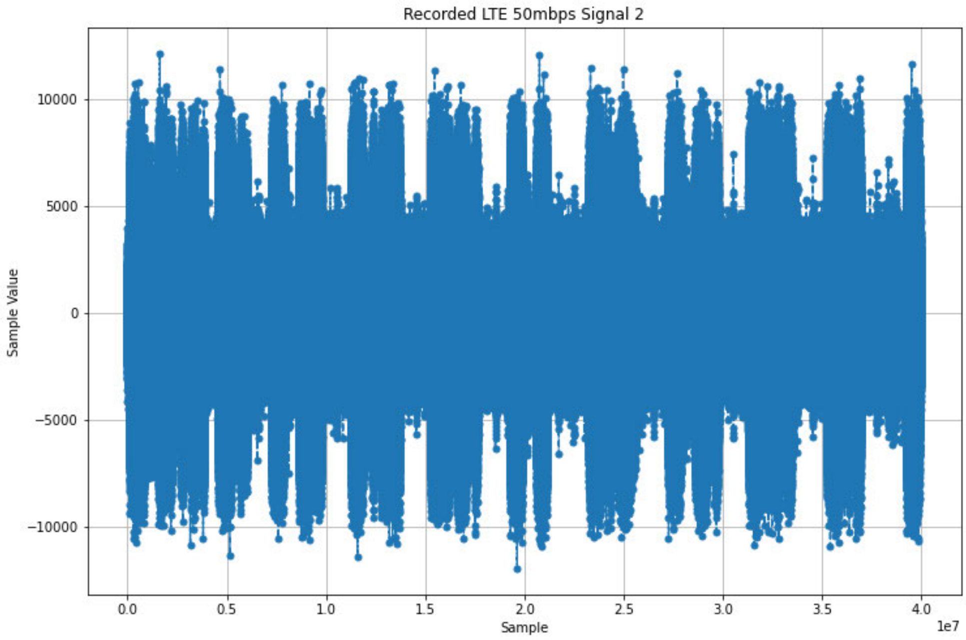

Figure 4.8: Graph of LTE 50mbps Signal 2

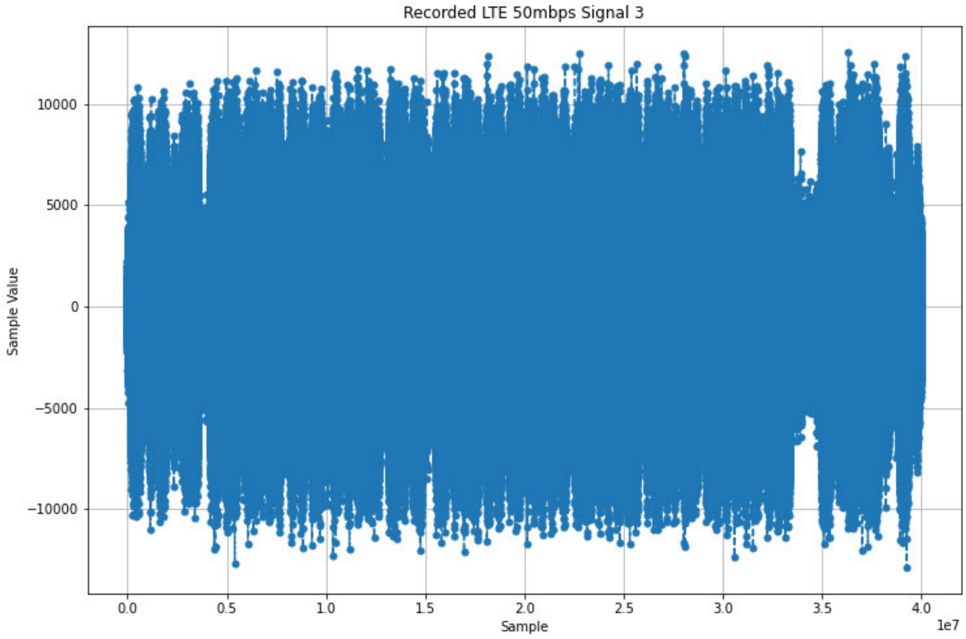

Figure 4.9: Graph of LTE 50mbps Signal 3

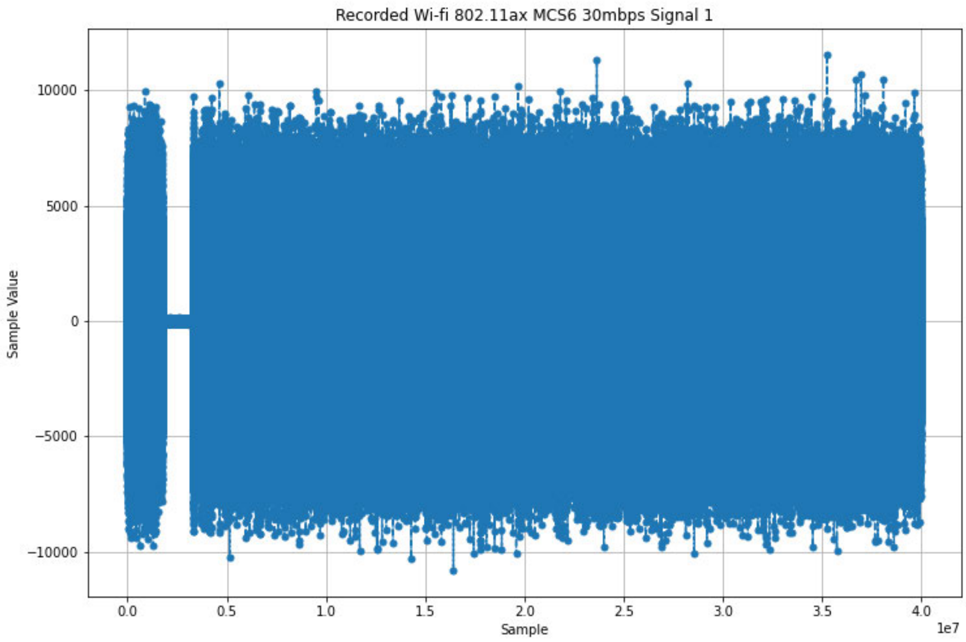

Figure 4.10: Graph of Wi-Fi 6 MCS6 30mbps Signal 1

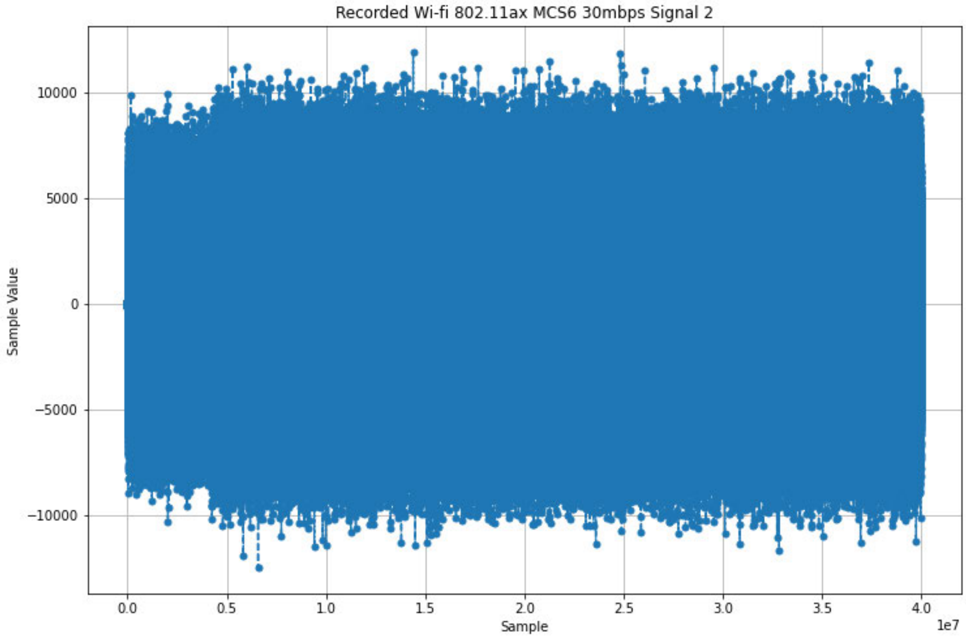

Figure 4.11: Graph of Wi-Fi 6 MCS6 30mbps Signal 2

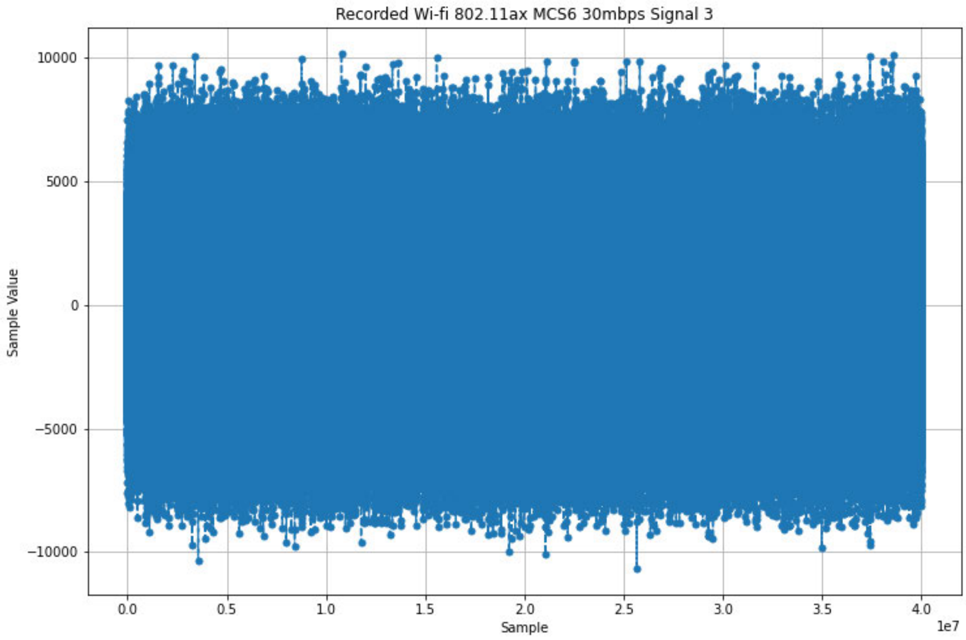

Figure 4.12: Graph of Wi-Fi 6 MCS6 30mbps Signal 3

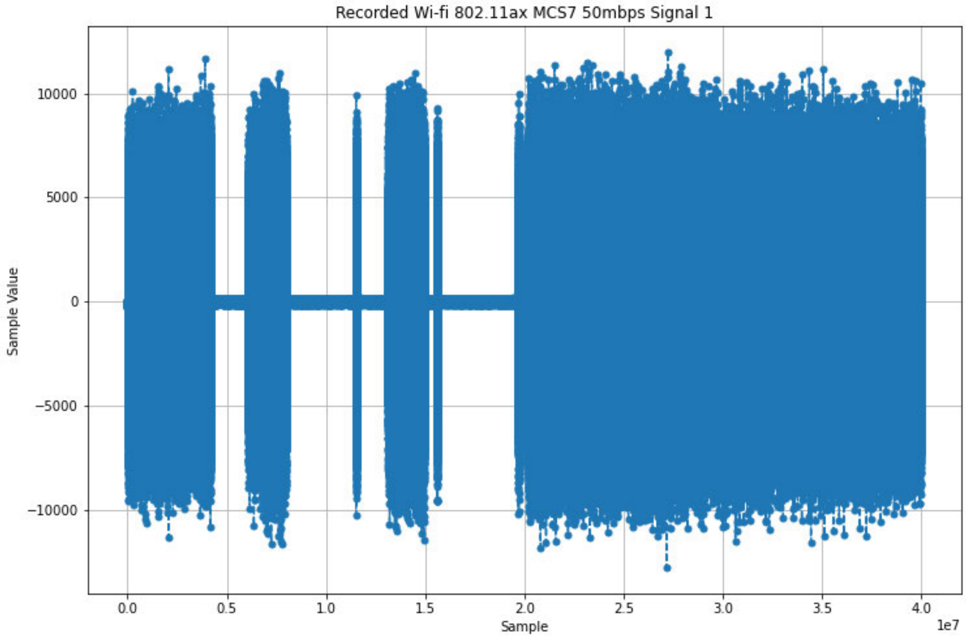

Figure 4.13: Graph of Wi-Fi 6 MCS7 50mbps Signal 1

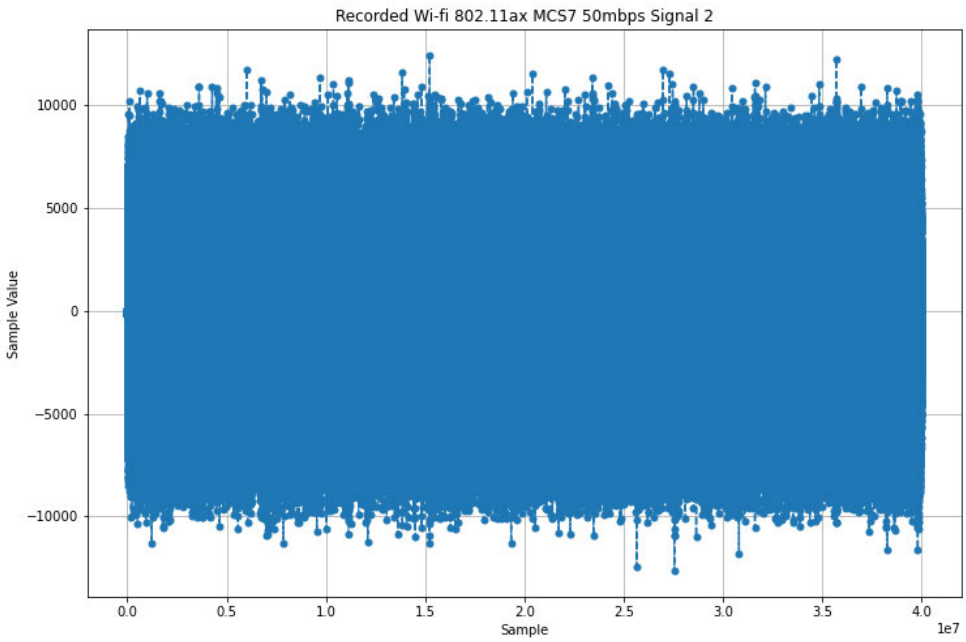

Figure 4.14: Graph of Wi-Fi 6 MCS7 50mbps Signal 2

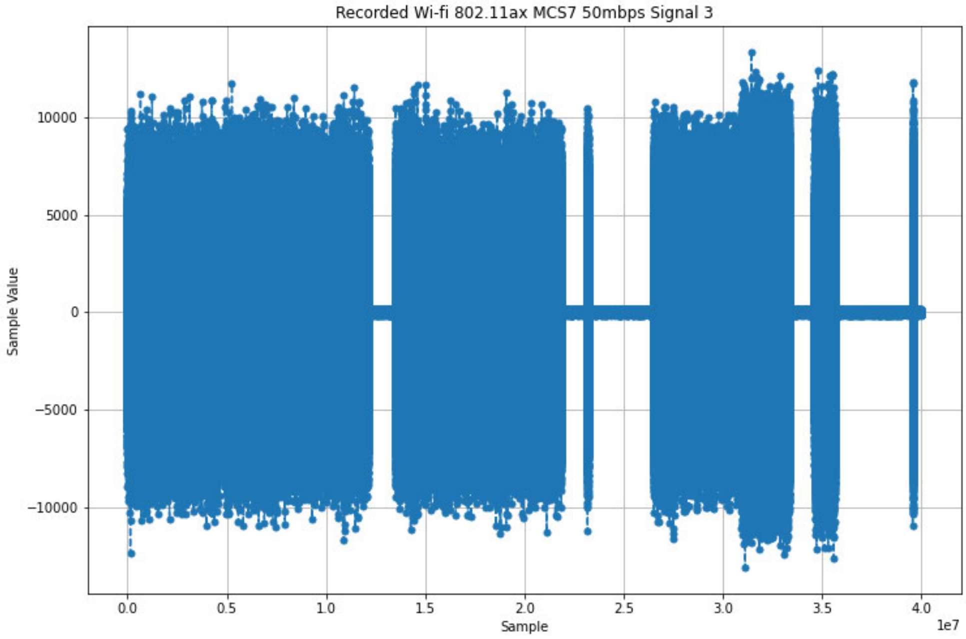

Figure 4.15: Graph of Wi-Fi 6 MCS7 50mbps Signal 3

4.3 Calibration of Signal Detection using Wi-Fi 6 MCS7 Signal 1

As Maximum-Minimum Eigenvalue (MME) signal detection is reliant on several factors, it is important to calibrate the signal detection prior to use.

In particular, MME was identified to rely on the following factors:

- M the Oversampling factor
- L the Smoothing factor
- P_{fa} the Probability of False Alarm
- N_s the Number of Samples (per frame)

Within these factors, L and N_s were chosen to be manipulated, as the oversampling factor (M) is representative of the ratio between sampling and transmission frequency (which cannot be changed or identified from the dataset) and the probability of false alarm (P_{fa}) should be held constant to simplify both signal detection and analysis.

Thus, holding $M = 1$ and $P_{fa} = 10\%$ constant and choosing $L = 5,\ 10,\ 15$ and $N_s = 1\ 000, 10\ 000, 100\ 000$, MME signal detection was performed on Wi-Fi 6 MCS7 Signal 1 for all combinations of L and N_s to calibrate the signal detection code for maximum stability in eigenvalue ratios and signal detection. This resulted in the creation of *.txt* files containing the eigenvalue ratios, detected signal presence and thresholds for all tests. The relevant minimum, maximum and average of these statistics for each test can be seen in Table 4.1 on the following page.

Table 4.1: Tabulated Statistical Results from all Wi-Fi 6 MCS7 Signal 1 Calibration Tests

L	$N_s(samples)$	Eigenvalue Ratio ($\lambda_{max}/\lambda_{min}$)			Threshold Constant (γ)	Signal Presence (Boolean, 0 or 1) Average
		Minimum	Maximum	Average		
5	1 000	0.0	66.9218	1.6261	1.3413	0.79343
5	10 000	1.1996	2.9642	1.5172	1.0973	1.0
5	100 000	1.2404	1.7885	1.4480	1.0298	1.0
10	1 000	0.0	189.8948	2.3184	1.5074	0.9669
10	10 000	1.3375	9.4778	2.0582	1.1382	1.0
10	100 000	1.5003	4.6010	1.8668	1.0418	1.0
15	1 000	0.0	237.2686	3.5515	1.64992	0.9957
15	10 000	1.6397	22.9371	3.1520	1.1709	1.0
15	100 000	1.8143	16.2344	2.4995	1.0511	1.0

Note that for Table 4.1, the Signal Presence at each sample frame has been averaged from their boolean values of 0 (False - No Signal) and 1 (True - Signal Present). Any value less than 1.0 suggests that a signal was not detected in some frames. As identified later, a signal is present for all frames of the dataset waveforms. Therefore, a Signal Presence Average of 1.0 is desirable.

Additional note: All values in Table 4.1 have been rounded to 4 decimal places.

From Table 4.1 and the knowledge that a signal is always present throughtout the tested signal's dataset (Wi-Fi 6 MCS7 Signal 1), it is possible to infer that:

- As L increases and N_s is constant, the Eigenvalue Ratios tend to increase, the Threshold Constant (γ) decreases and the likelihood of detecting a signal increases.
- As N_s increases and L is constant, the Eigenvalue Ratio range (between minimum and maximum ratios) reduces, the Threshold Constant (γ) reduces and the likelihood of detecting a signal increases.
- $N_s = 1\ 000$ is prone to producing large maximum eigenvalue ratios and has failed to produce a minimum eigenvalue ratio.
- $N_s = 1\ 000$ does not fully detect the present signal.

Regarding the failure to produce a minimum eigenvalue ratio, it was identified within the eigenvalue/vector theory section (Section 2.3.3) that it is possible for all eigenvalues to be zero, complex eigenvalues to be present or to have a minimum eigenvalue of zero.

In the case of 'all eigenvalues are zero', the code sets the eigenvalue ratio to be zero. This is the only case that can result in a minimum eigenvalue ratio of zero. All other cases are handled by the python code or have not appeared during the running of the code.

Therefore, $N_s = 1\ 000$ calculates all eigenvalues to be zero for some unknown frames of the signal. This is evidently an undesirable outcome from running the code, and may have resulted in incorrect 'no signal detected' determinations. Further evaluation of the signal detection results was deemed necessary to determine if these determinations were correct and within regions of low signal activity.

However, at the creation of Table 4.1, it was expected that the low signal activity sections of the signal (Wi-Fi 6 MCS7 Signal 1 as Figure 4.13 on Page 49) would not have a signal present. Later investigation would show that these low activity sections contained a signal from both visual examination and signal detection results. To further examine the pattern of signal detection, Wi-Fi 6 MCS7 Signal 1 was graphed and overlayed with the regions where the signal was detected for each test. These figures (Figures 4.16 to 4.24) can be found on the following pages (Pages 54 to 58).

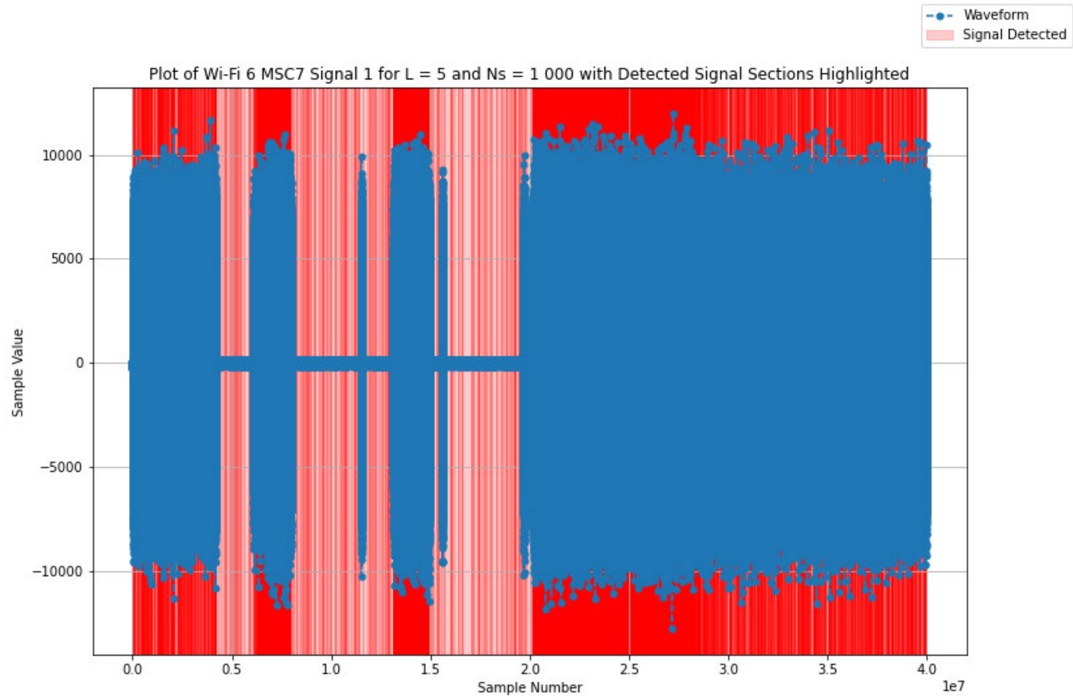

Figure 4.16: Wi-Fi 6 MCS7 Signal 1 with signals detected in highlighted sections using $L = 5$ and $N_s = 1000$

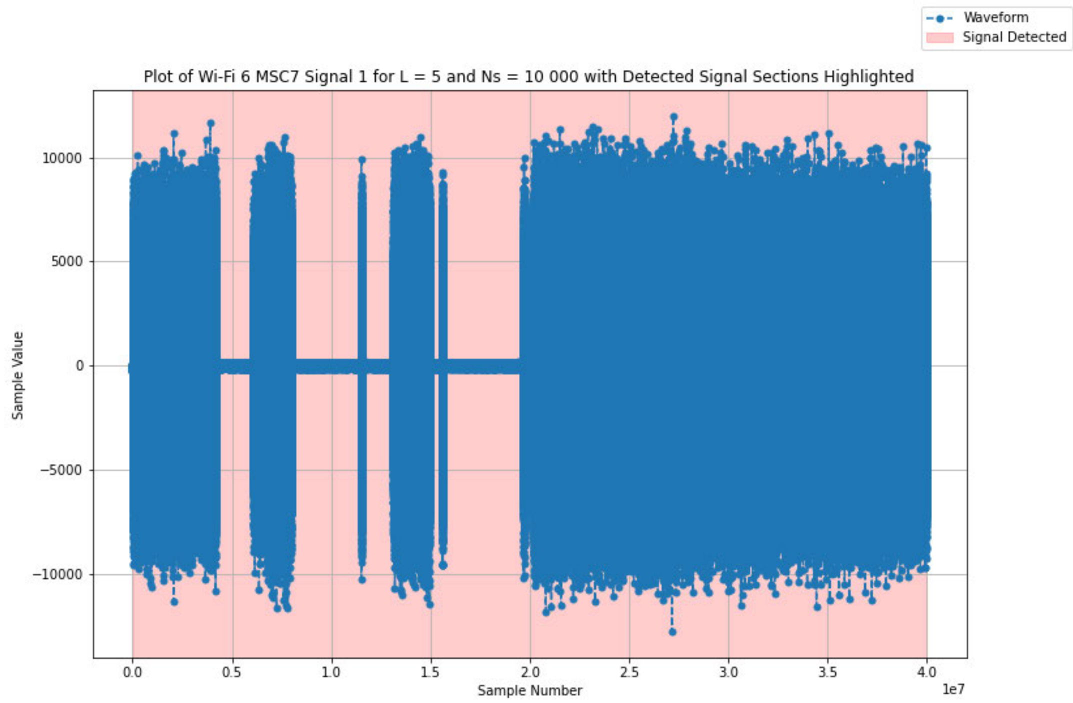

Figure 4.17: Wi-Fi 6 MCS7 Signal 1 with signals detected in highlighted sections using $L = 5$ and $N_s = 10\ 000$

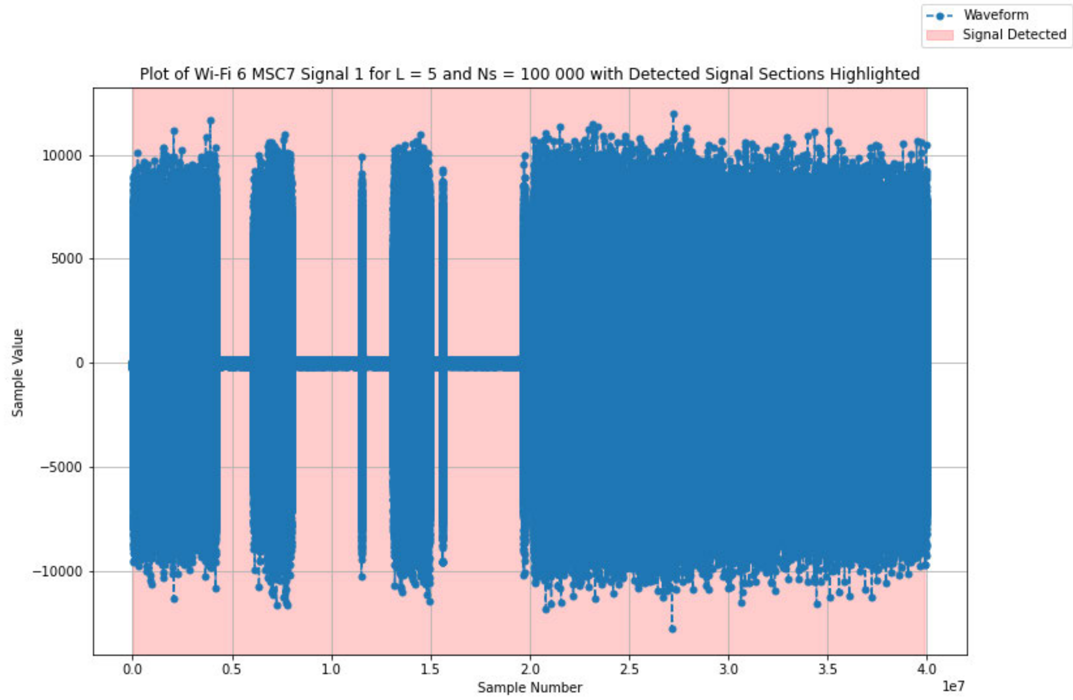

Figure 4.18: Wi-Fi 6 MCS7 Signal 1 with signals detected in highlighted sections using $L = 5$ and $N_s = 100\ 000$

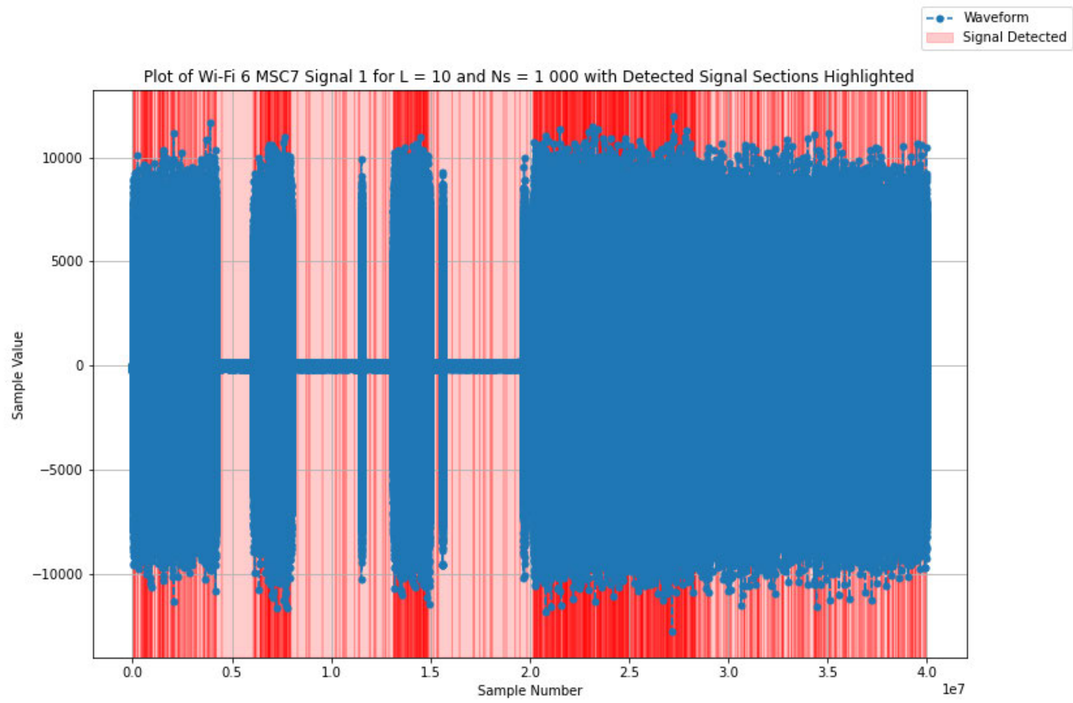

Figure 4.19: Wi-Fi 6 MCS7 Signal 1 with signals detected in highlighted sections using $L = 10$ and $N_s = 1000$

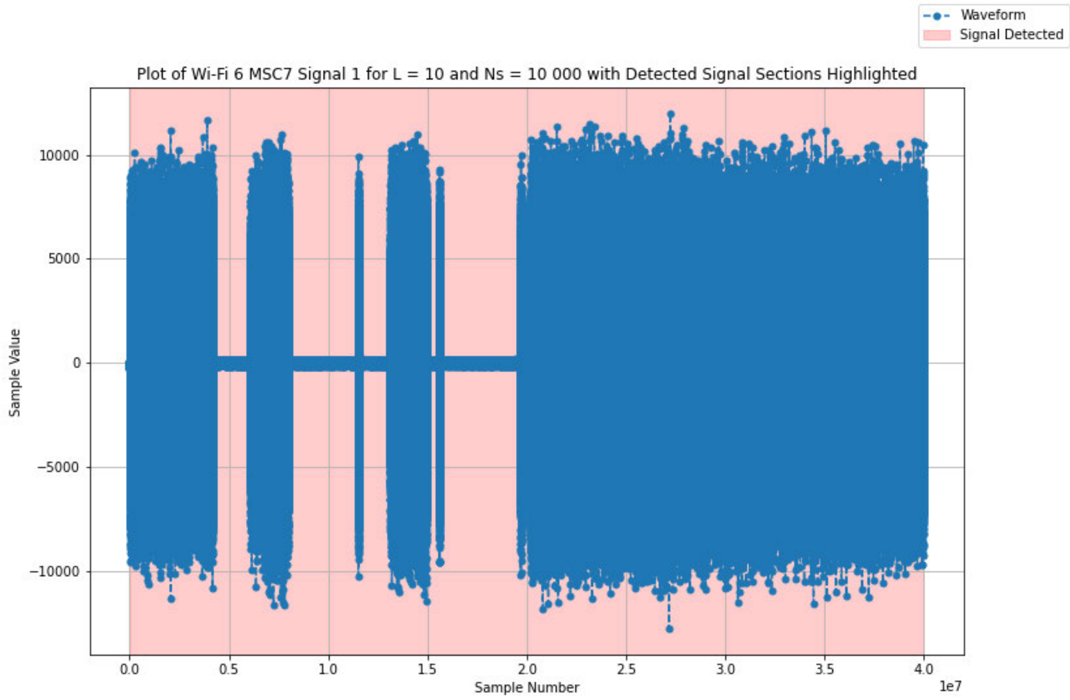

Figure 4.20: Wi-Fi 6 MCS7 Signal 1 with signals detected in highlighted sections using $L = 10$ and N_s = 10 000

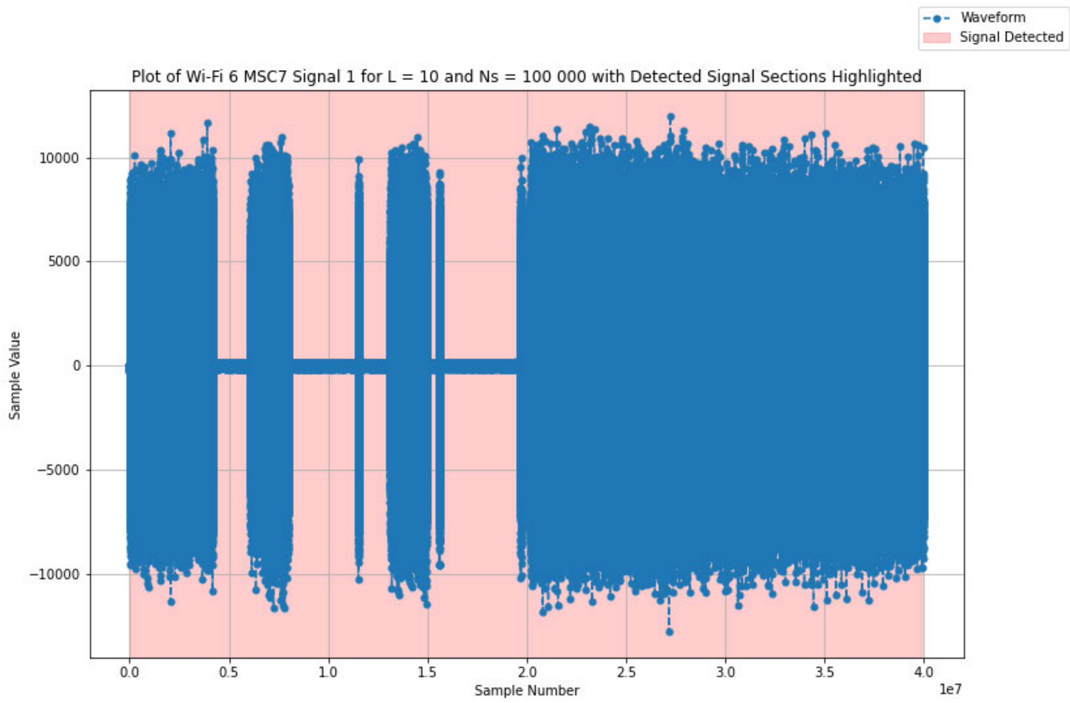

Figure 4.21: Wi-Fi 6 MCS7 Signal 1 with signals detected in highlighted sections using $L = 10$ and N_s = 100 000

Figure 4.22: Wi-Fi 6 MCS7 Signal 1 with signals detected in highlighted sections using $L = 15$ and $N_s = 1000$

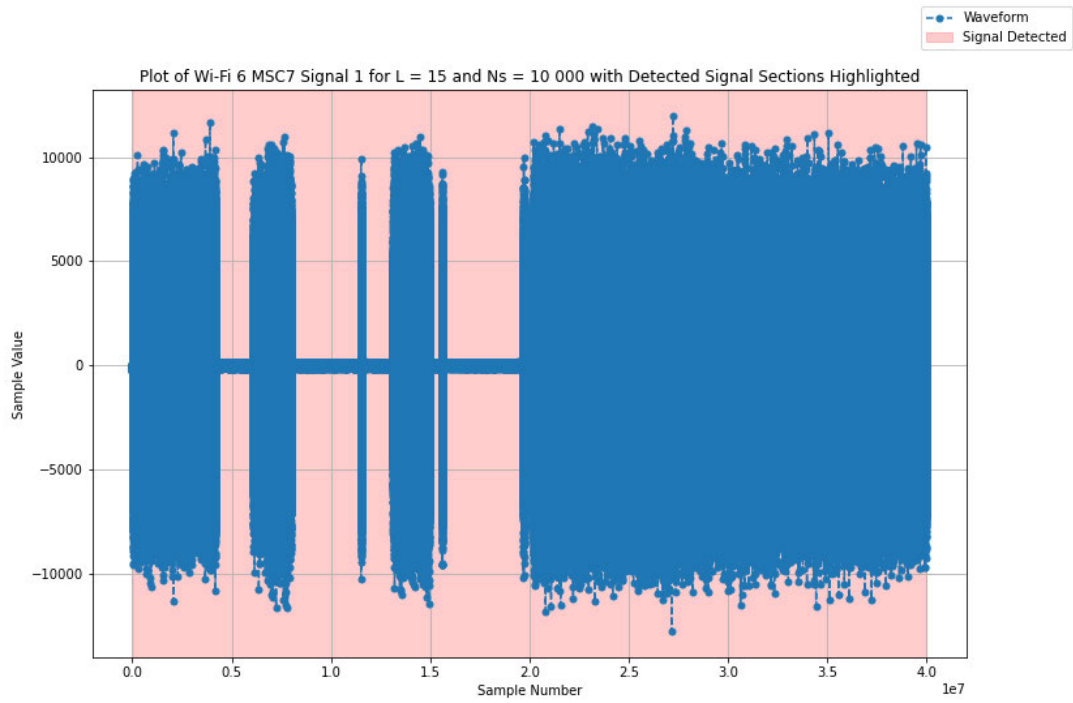

Figure 4.23: Wi-Fi 6 MCS7 Signal 1 with signals detected in highlighted sections using $L = 15$ and $N_s = 10\ 000$

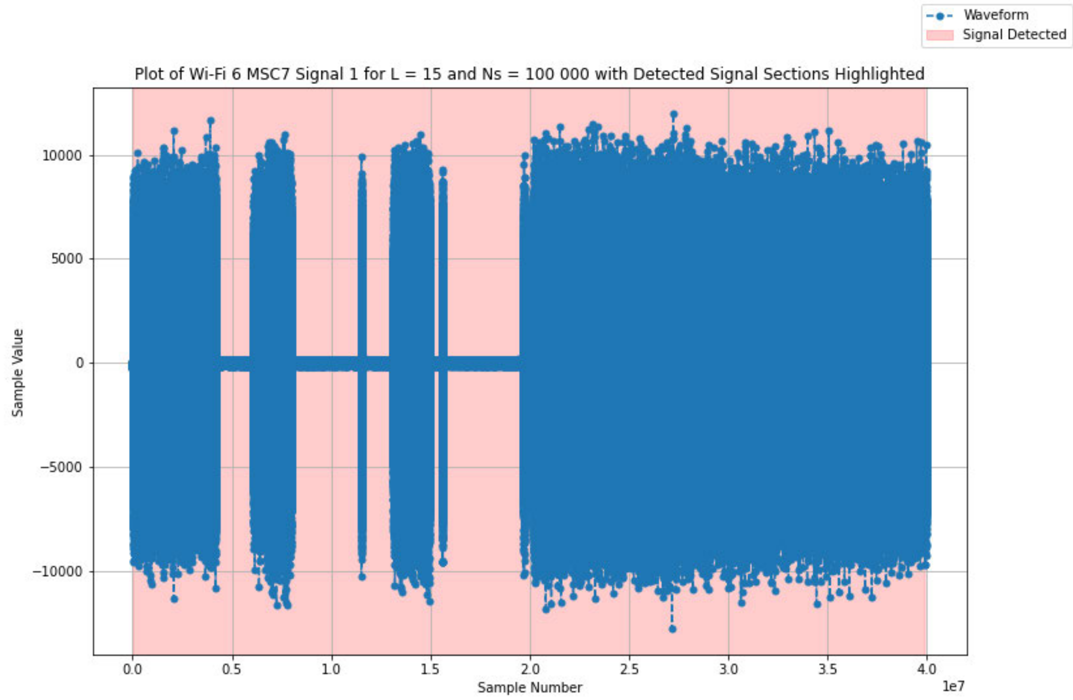

Figure 4.24: Wi-Fi 6 MCS7 Signal 1 with signals detected in highlighted sections using $L = 15$ and $N_s = 100\ 000$

Note that Figures 4.16 to 4.24 on Pages 54 to 58 have the sections where the signal was detected highlighted as a transparent pink block. Sections where the signal was not detected are not highlighted. Sections that appear red are the result of inconsistent signal detection.

From these figures, it is evident that the signal detection for $N_s = 1\ 000$ is flawed. Judging by the striped red sections of Figures 4.16, 4.19 and 4.22, areas of high signal activity (large sample values) are being incorrectly evaluated as having no signal present. To form these stripes, the results must alternate rapidly between the signal being present or the signal being missing over small segements of the overall signal waveform. As such, the signal detection for $N_s = 1\ 000$ is unstable and either $N_s = 10\ 000$ or $N_s = 100\ 000$ should be used.

However, despite the unstable signal detection, the results did not support the assumption that low signal activity sections of the waveform do not contain a signal. This assumption was a result of the visual examination of the waveform, as graphed previously in Figure 4.13 on Page 49.

The eigenvalue ratio was suspected to be the main contributer to the instability of the signal detection, as signal presence is determined by comparing the ratio to the threshold. To investigate how the eigenvalues changed throughout the signal, the eigenvalue ratio results were grouped by the N_s used to generate the results and graphed. The resulting figures (Figures 4.25 to 4.27) can be found on the following pages (Pages 60 to 61).

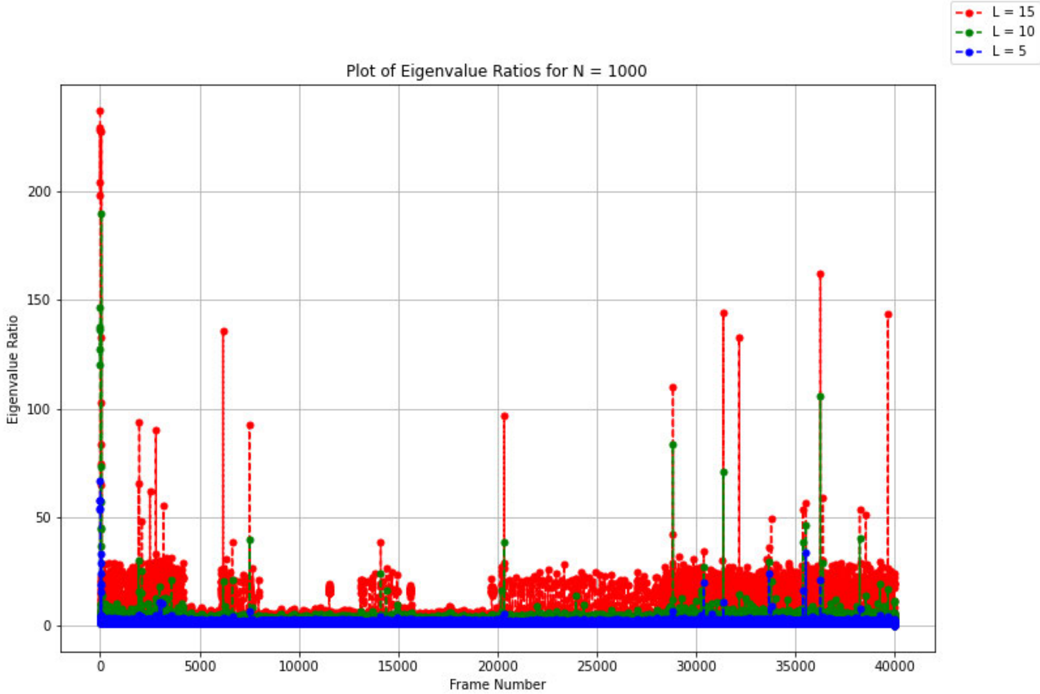

Figure 4.25: Graph of eigenvalue ratios per frame for various L and $N_s = 1000$

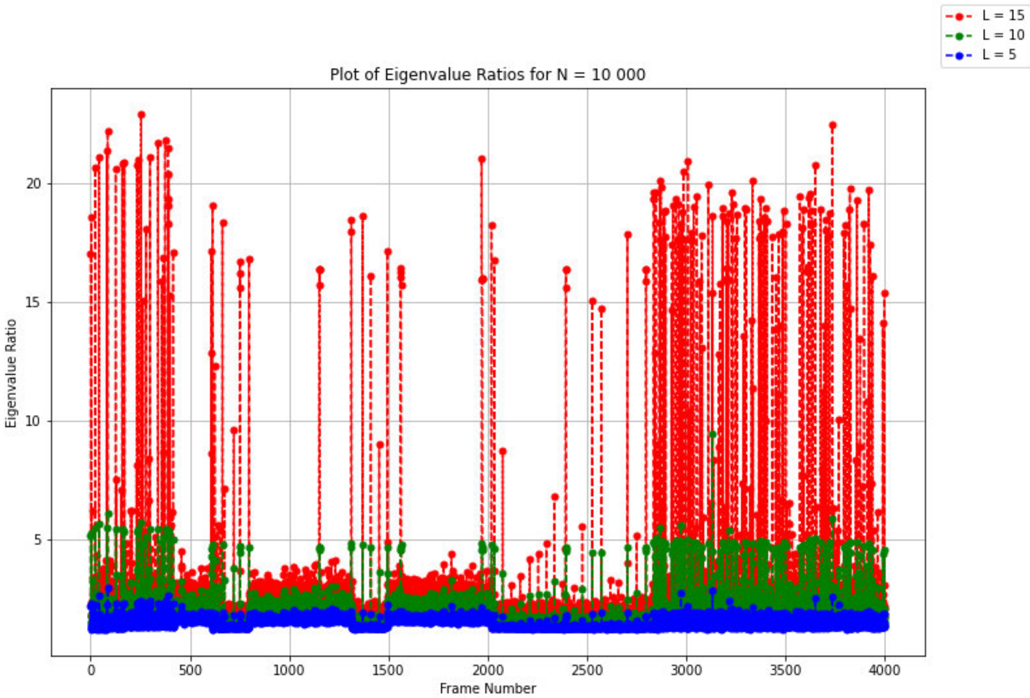

Figure 4.26: Graph of eigenvalue ratios per frame for various L and $N_s = 10\ 000$

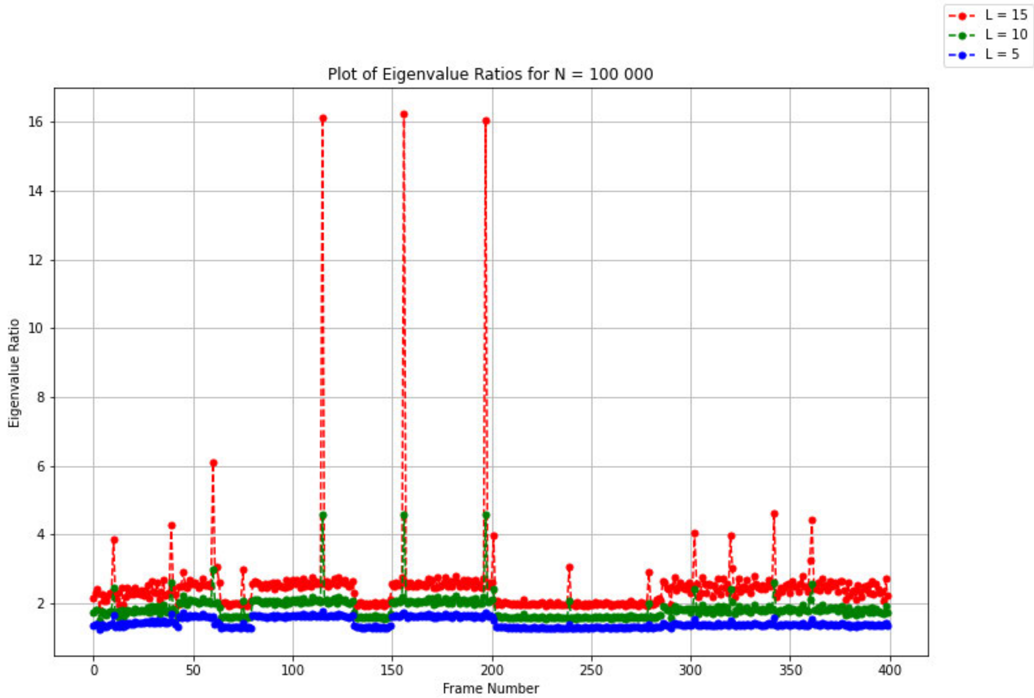

Figure 4.27: Graph of eigenvalue ratios per frame for various L and $N_s = 100\ 000$

From Figure 4.27 (on Page 4.27), it is evident that increasing L can increase the eigenvalue ratios, while maintaining the same distribution shape. By examination of the all figures (Figure 4.25 to 4.27 on Page 60 to 61), it appears that increasing L can result in significant spikes in the eigenvalue ratios. Since the signal detection was identified previously to be unstable for $N_s = 1\ 000$, resulting in the stripes appearing on the signal detection graphs (Figure 4.16, 4.19 and 4.22), $N_s = 1\ 000$ cannot be chosen as a calibration value, due to the minimum eigenvalue ratio being zero and the large spikes in eigenvalue ratios throughout Figure 4.25 on Page 60.

Due to the smaller amount and reduced value of the eigenvalue spikes for $N_s = 100$ 000, $N_s = 100\ 000$ was chosen. This should avoid the minimum eigenvalue problems of $N_s = 1\ 000$ and the high maximums and/or large eigenvalue ratio spikes of $N_s = 100$ 000. Additionally, since increasing L results in both larger eigenvalue ratios and spikes in eigenvalue ratios, $L = 5$ was chosen to minimise the number of spikes in eigenvalue ratios.

Thus, to ensure that the signal detection is stable (no rapid alternating between signal absence and signal presence), $N_s = 100000$ and $L = 5$ were chosen for further testing.

However, despite the identification of better values for L and N_s, the signal detection instability for $N_s = 1\ 000$ could not be explained by the eigenvalue figures (Figures 4.25 to 4.27 on Pages 60 to 61). This lead to the creation of custom signal waveforms (using Wi-Fi 6 MCS7 Signal 1), to confirm the signal detection code correctly determines the presence of a signal in ideal circumstances and to confirm the absence of a signal within the low signal activity sections of Wi-Fi 6 MCS7 Signal 1 (areas of low sample value on Figure 4.13 on Page 49). This process is covered in the following section (Section 4.4) and succesfully confirms that the code is able to detect the presence/absence of a signal in ideal conditions as well as identifying the presence of a signal within the low signal activity sections of Wi-Fi 6 MCS7 Signal 1.

Thus, using the findings of the following section to validate the signal detection code and constant signal presence throughout Wi-Fi 6 MCS7 Signal 1, as well as the findings discussed in this section, $N_s = 100\ 000$ and $L = 5$ should avoid the 'all eigenvalues are zero' problem identified previously (during the discussion on Table 4.1 results) and allow for more consistent eigenvalue ratios. $N_s = 100\ 000$ and $L = 5$ is used in all relevant following sections.

4.4 Calibration of Signal Detection using Custom Waveforms

Having successfully identified $L = 5$ and $N_s = 100\ 000$ as suitable values to stabilise the eigenvalue ratio and potentially signal detection, a custom signal was created to determine whether the signal detection code was functioning properly.

Using the first 200 000 samples of Wi-Fi MCS7 Signal 1, Custom Wave 1 featured in Figure 4.28 was created. Custom Wave 1 has been padded with 100 000 samples with an amplitude of zero, before and after the 200 000 samples of Wi-Fi MCS7 Signal 1. This was intended to test whether the signal detection code could accuractely determine the presence and absence of a signal in ideal conditions.

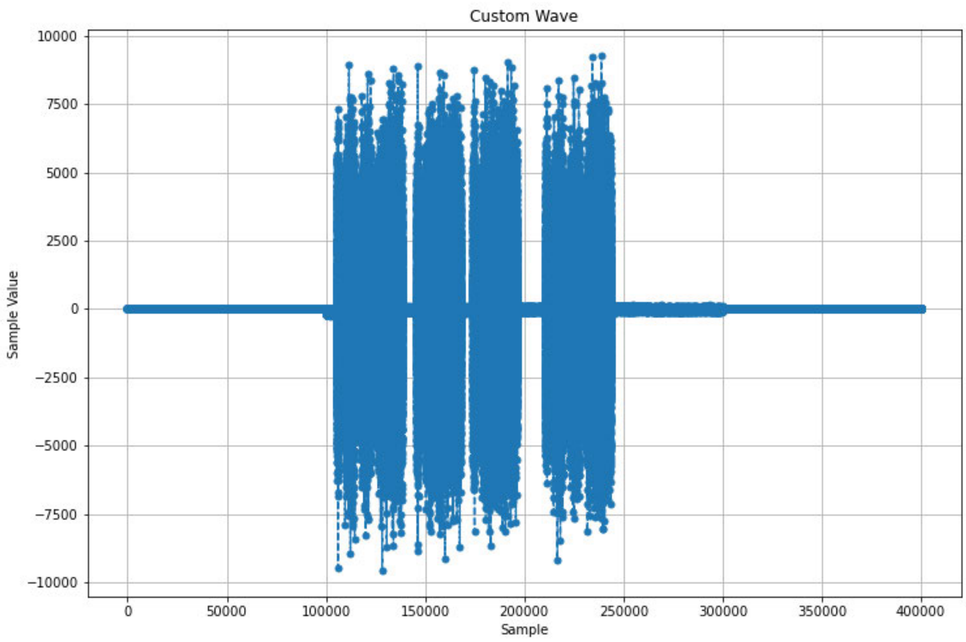

Figure 4.28: Custom Wave 1, created by padding 100 000 zero samples before and after the first 200 000 samples of Wi-Fi MCS7 Signal 1

Applying the signal detection code to Custom Wave 1, the signal detection results have been represented graphically in Figure 4.29 below. By inspection of Figure 4.29, the signal detection test was successful, as the zero sample areas of the custom waveform have been successfully identified to not contain a signal. Thus, additional testing was required to determine the presence of a signal within the low signal activity sections of Wi-Fi MCS7 Signal 1.

Note: $L = 5$ and $N_s = 100\ 000$ was used.

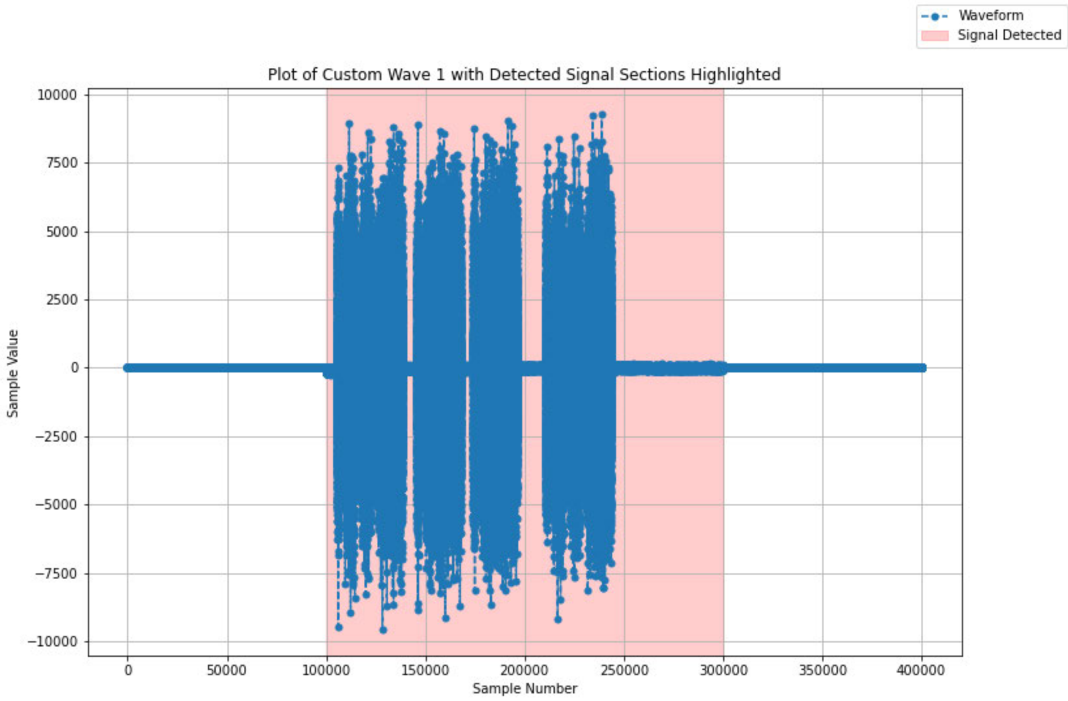

Figure 4.29: Custom Wave 1 with the detected signal areas highlighted

With the signal detection code confirmed to be functional, Custom Wave 2 was created to evaluate the low signal activity areas of Wi-Fi MCS7 Signal 1. Taking a 3 million sample long slice of Wi-Fi MCS7 Signal 1, from sample 16 million to sample 19 million, Custom Wave 2 was created and graphed in Figure 4.30 below.

By inspection of Figure 4.30, it was apparent that within the low signal activity sections of Wi-Fi MCS7 Signal 1, some form of signal or noise was present. As the Subray (2023) dataset does not provide information regarding the noise levels of the dataset signals, it is assumed that a signal is present within the low signal activity sections of Wi-Fi MCS7 Signal 1 (and other dataset signals). Regardless, signal detection was performed on Custom Wave 2.

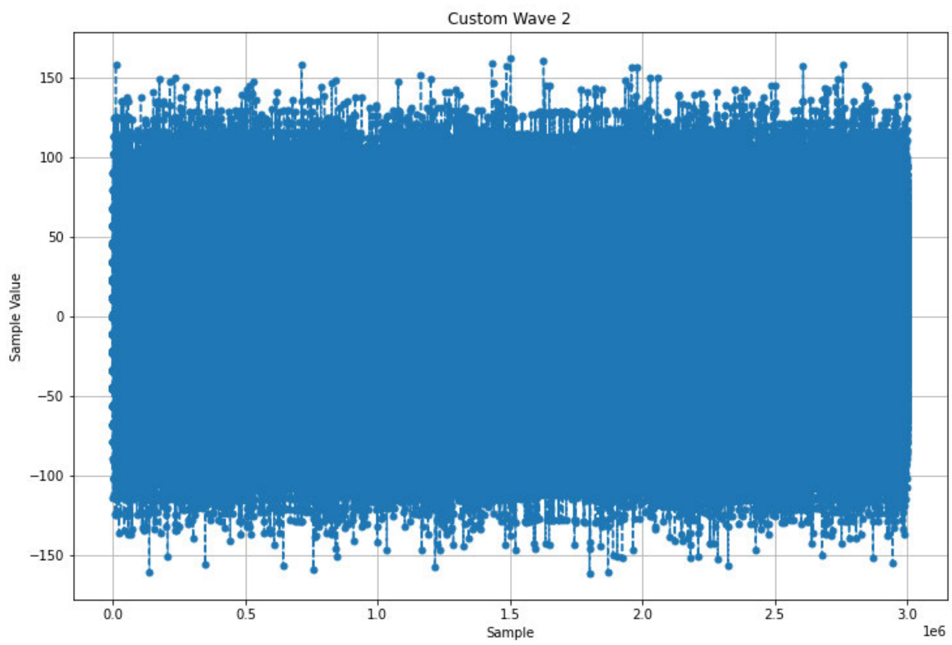

Figure 4.30: Custom Wave 2, a 3 million sample slice of Wi-Fi MCS7 Signal 1

Having performed signal detection on Custom Wave 2, Figure 4.31 was created (shown below), highlighting the sections where a signal has been detected. Due to poor paritioning, the signal detection code did not operate on the last set of samples.

Otherwise, the signal detection code confirmed the presence of a signal in the low-activity section of Wi-Fi MCS7 Signal 1 from sample 16 million to sample 19 million. It is inferred that other low signal activity sections in Wi-Fi 6 MCS7 Signal 1 and other signals from the Subray (2023) dataset also contain a signal.

Note: $L = 5$ and $N_s = 100\ 000$ was used.

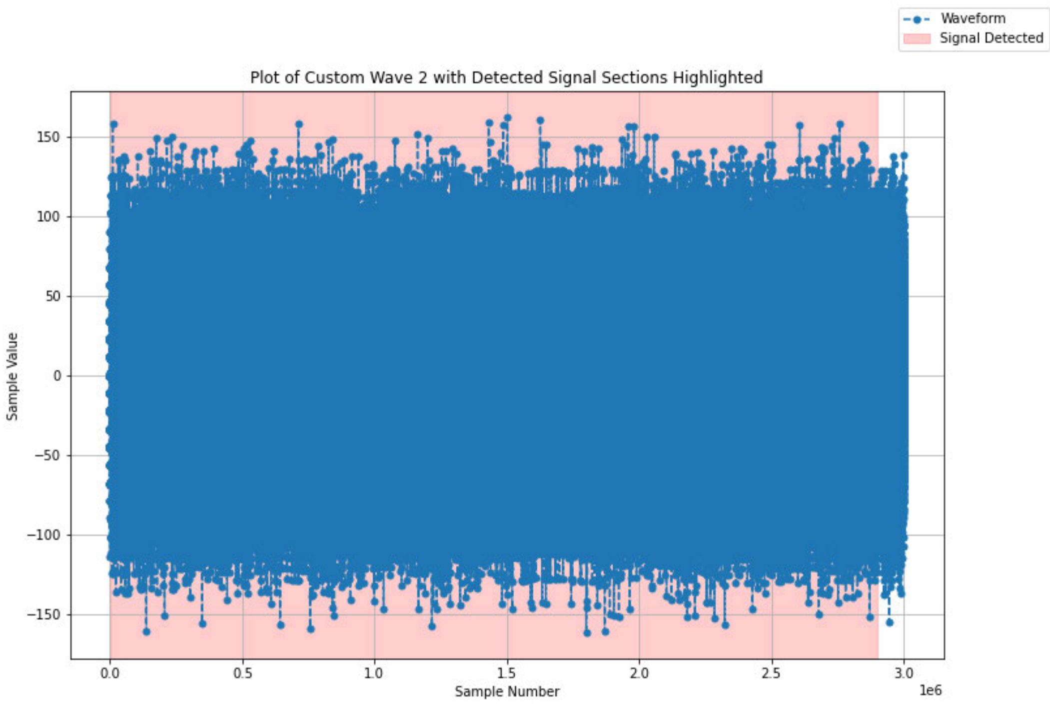

Figure 4.31: Custom Wave 2 with the detected signal areas highlighted

Thus, the signal detection code has been successfully calibrated for signal detection using $M = 1$, $P_{fa} = 10\%$, $L = 5$ and $N_s = 100\ 000$.

4.5 General Signal Detection

With calibration successfully completed, all of the signals provided in the Subray (2023) dataset were extracted and tested using the signal detection code. The following variables were used:

$$L = 5 \qquad N_s = 100000$$

$$M = 1 \qquad P_{fa} = 10\%$$

The signal detection results have been graphed using the original signal waveforms (shown previously as Figures 4.1 to 4.15 on Pages 43 to 50) by highlighting the relevant sections where a signal was detected in pink.

These highlighted graphs can be seen as Figures 4.32 to 4.46 on Pages 67 to 74.

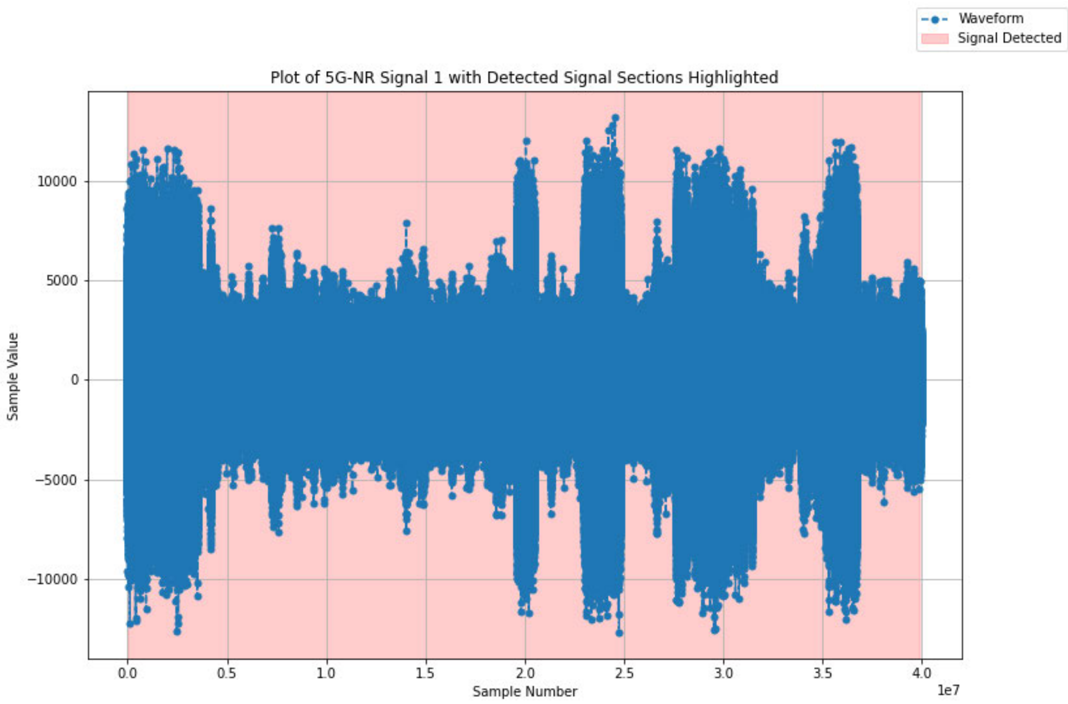

Figure 4.32: 5G NR Signal 1 with highlighted sections depicting the presence of a signal

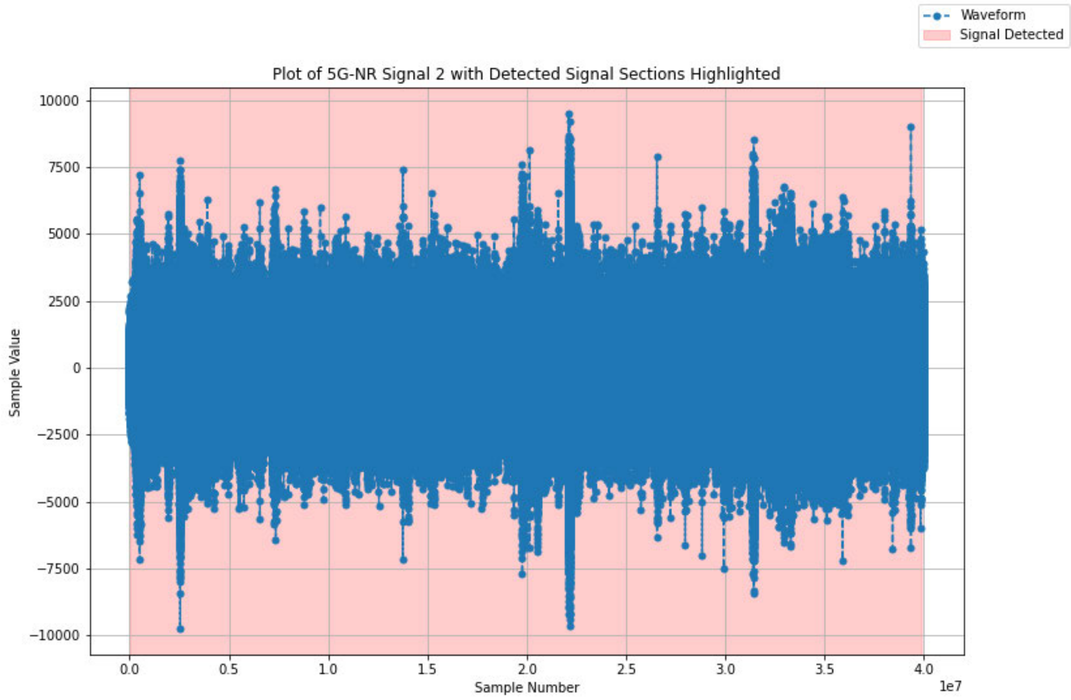

Figure 4.33: 5G NR Signal 2 with highlighted sections depicting the presence of a signal

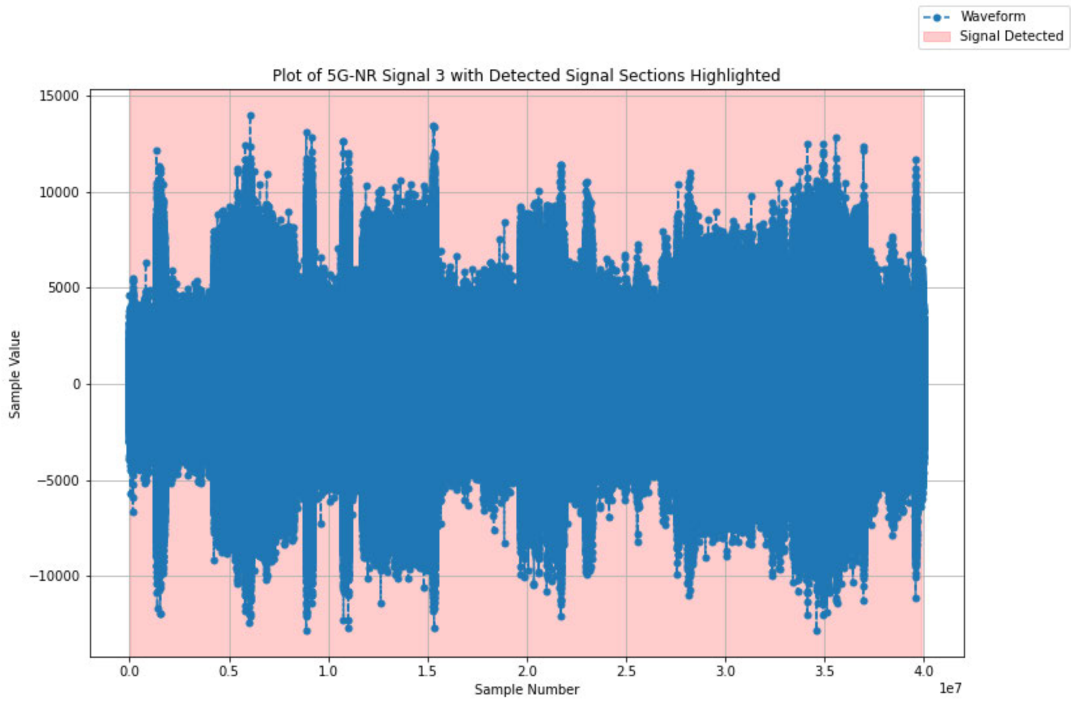

Figure 4.34: 5G NR Signal 3 with highlighted sections depicting the presence of a signal

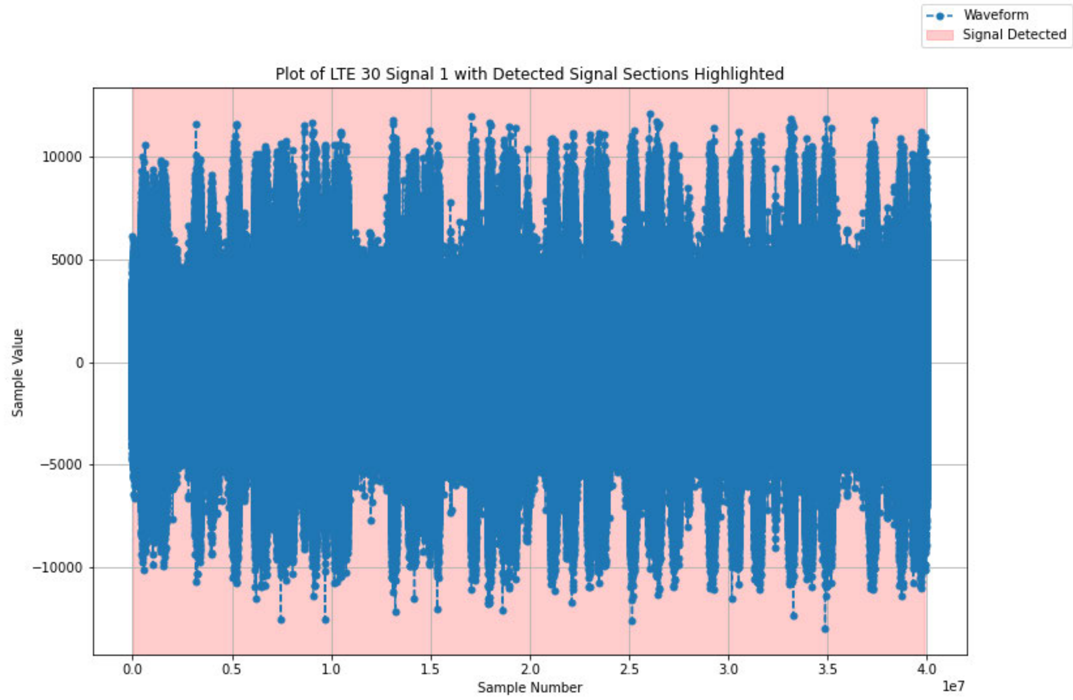

Figure 4.35: LTE (30mbps) Signal 1 with highlighted sections depicting the presence of a signal

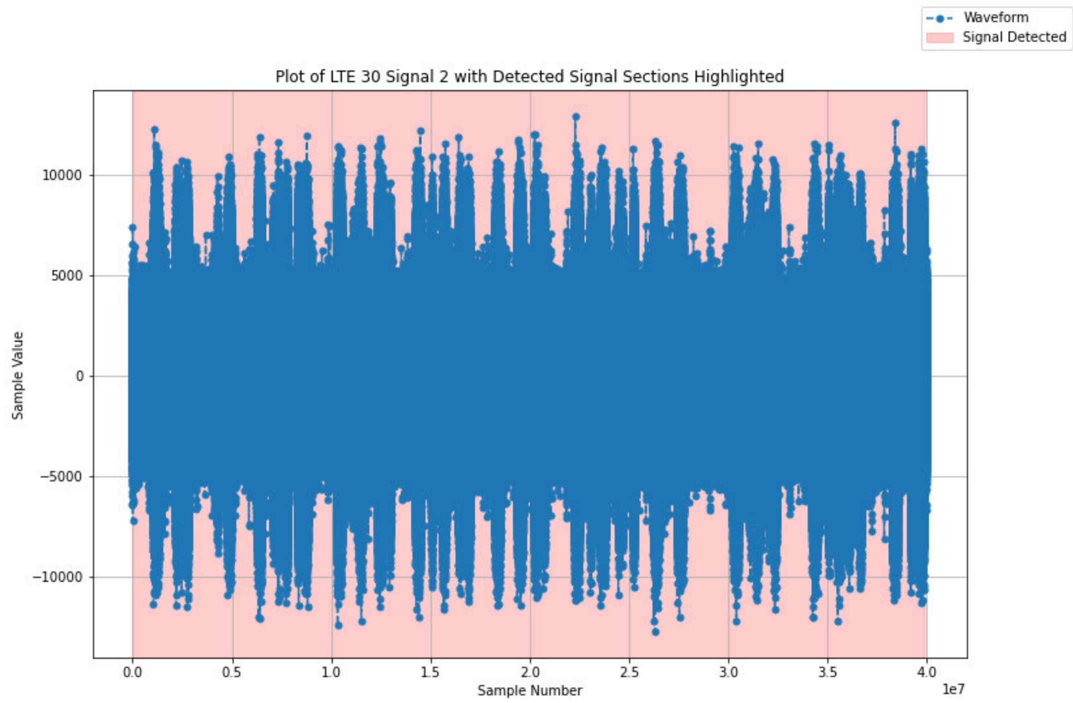

Figure 4.36: LTE (30mbps) Signal 2 with highlighted sections depicting the presence of a signal

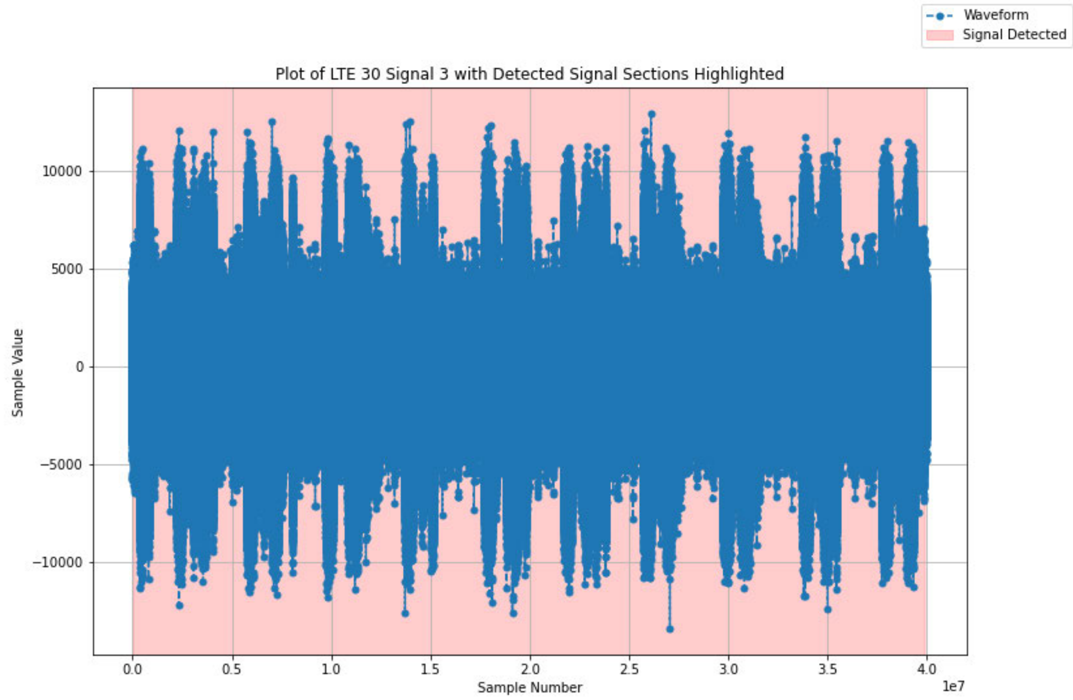

Figure 4.37: LTE (30mbps) Signal 3 with highlighted sections depicting the presence of a signal

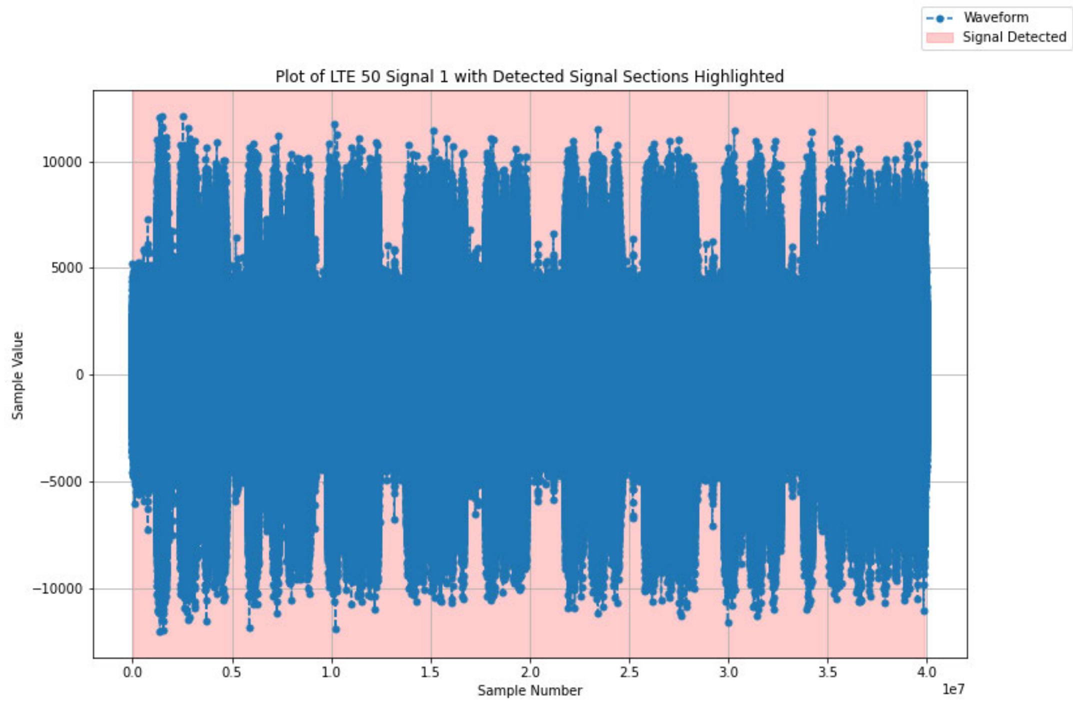

Figure 4.38: LTE (50mbps) Signal 1 with highlighted sections depicting the presence of a signal

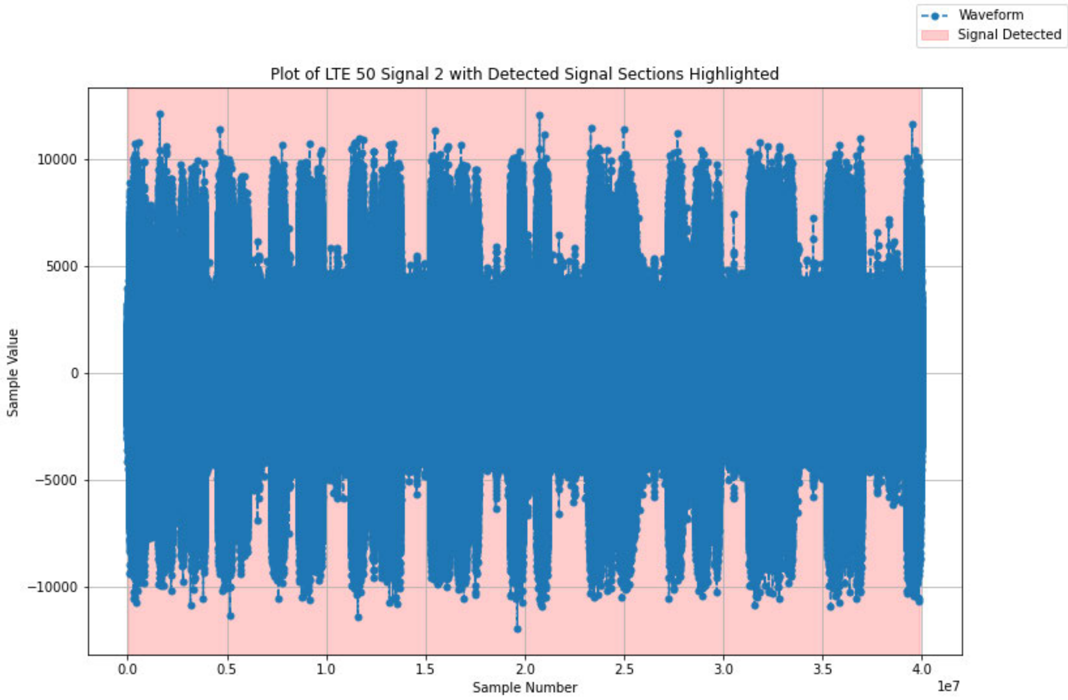

Figure 4.39: LTE (50mbps) Signal 2 with highlighted sections depicting the presence of a signal

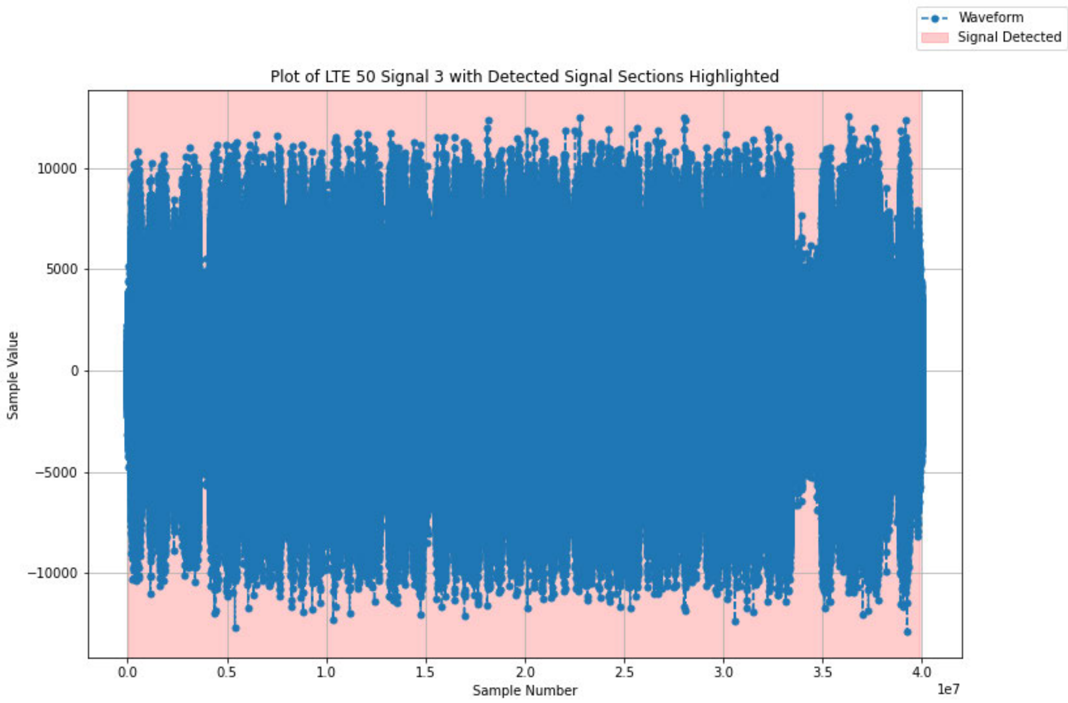

Figure 4.40: LTE (50mbps) Signal 3 with highlighted sections depicting the presence of a signal

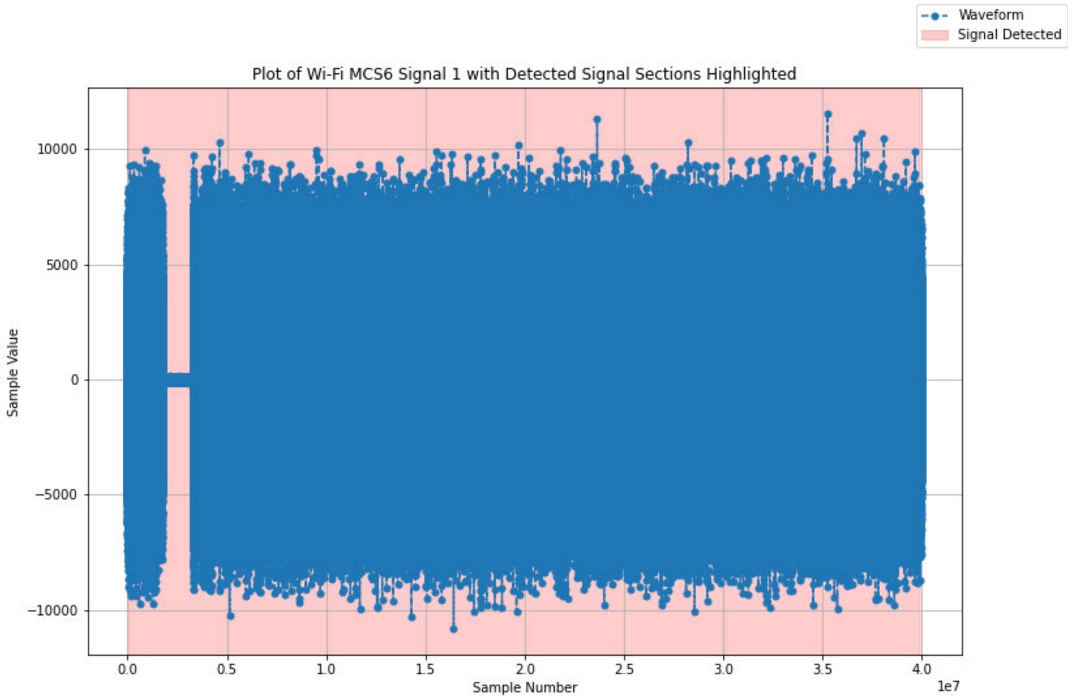

Figure 4.41: Wi-Fi 6 MCS6 Signal 1 with highlighted sections depicting the presence of a signal

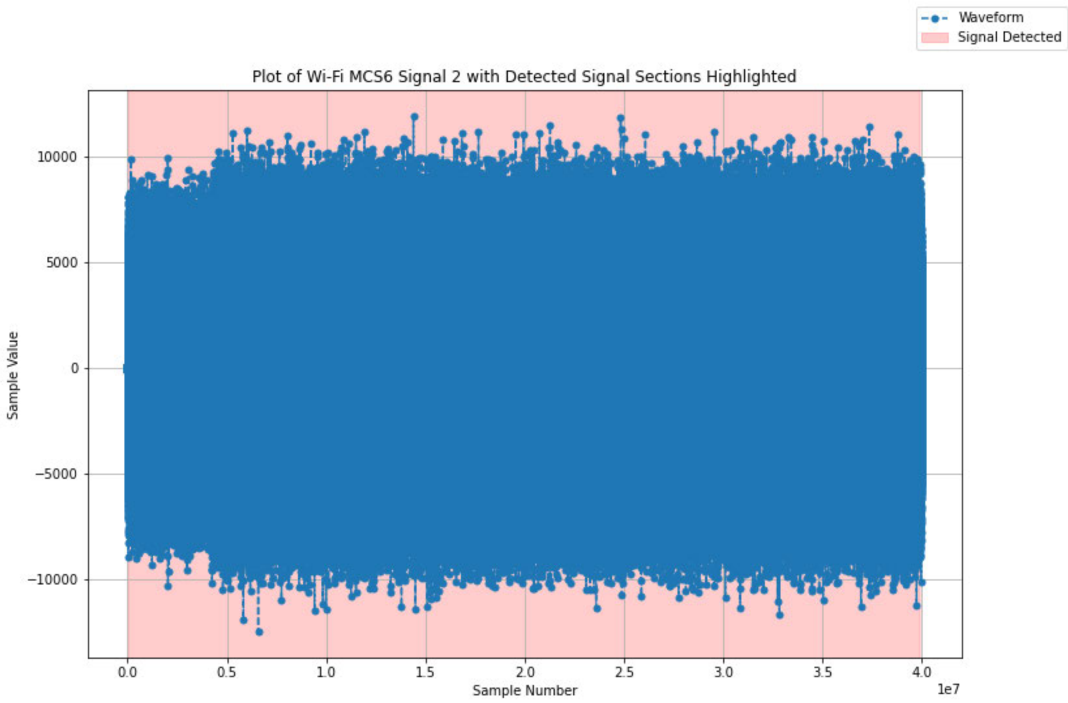

Figure 4.42: Wi-Fi 6 MCS6 Signal 2 with highlighted sections depicting the presence of a signal

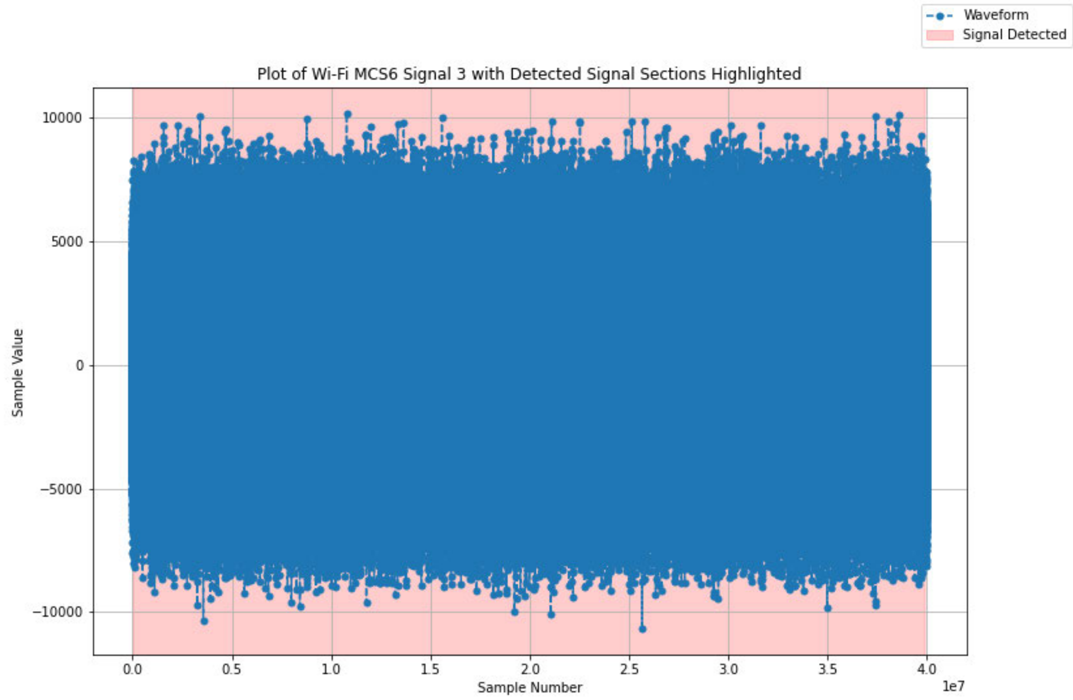

Figure 4.43: Wi-Fi 6 MCS6 Signal 3 with highlighted sections depicting the presence of a signal

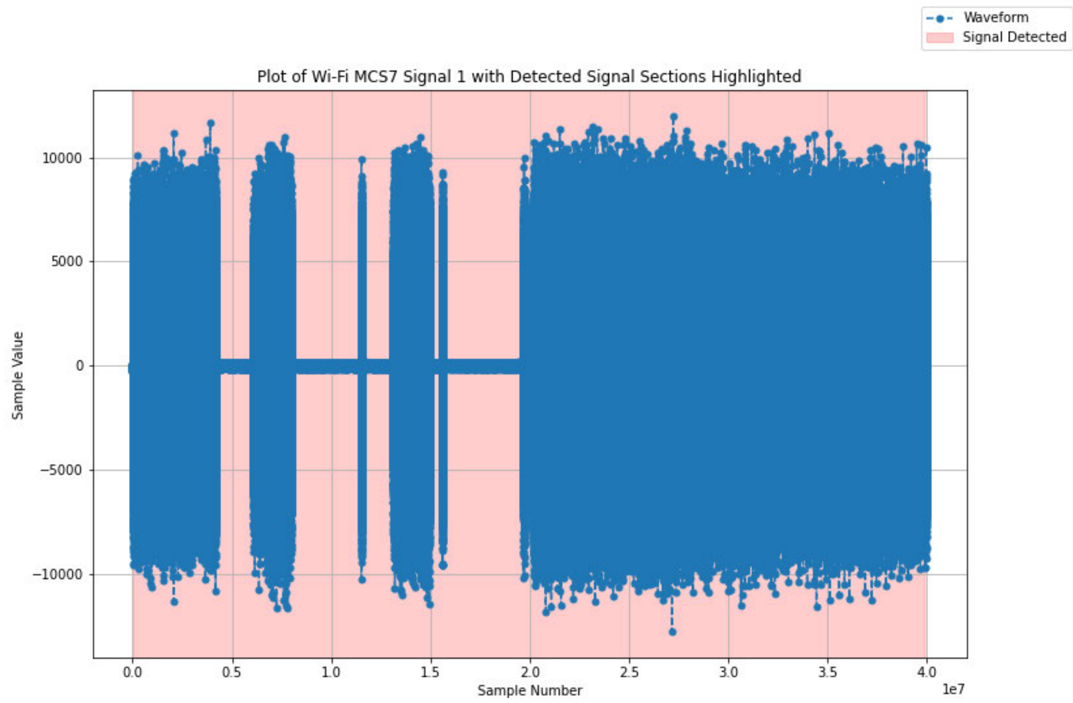

Figure 4.44: Wi-Fi 6 MCS7 Signal 1 with highlighted sections depicting the presence of a signal

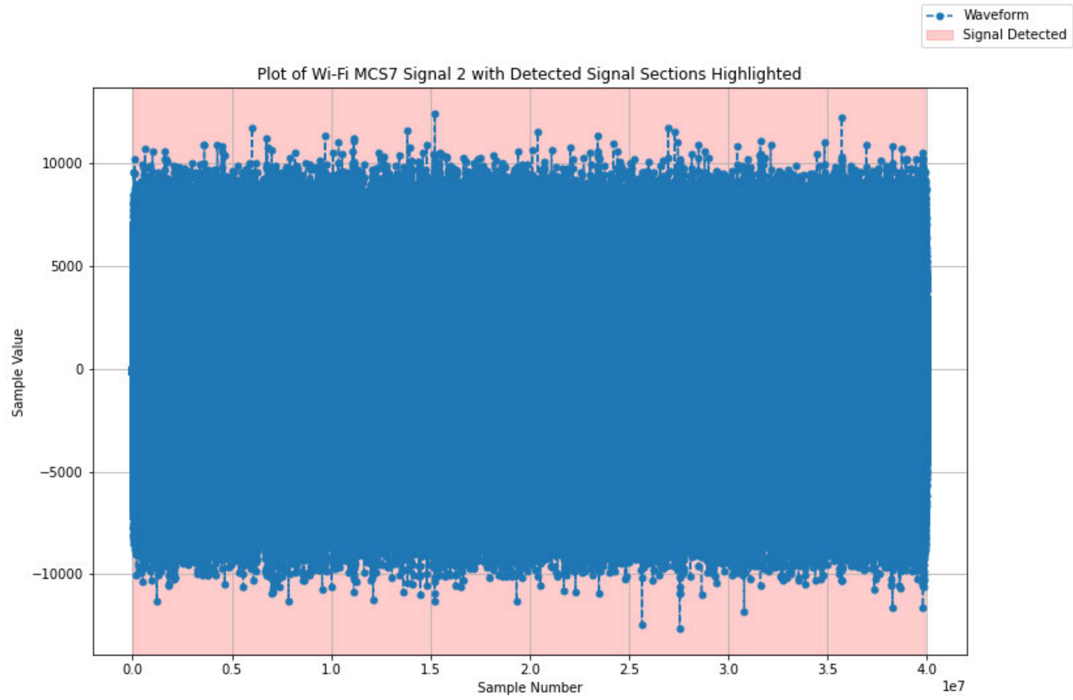

Figure 4.45: Wi-Fi 6 MCS7 Signal 2 with highlighted sections depicting the presence of a signal

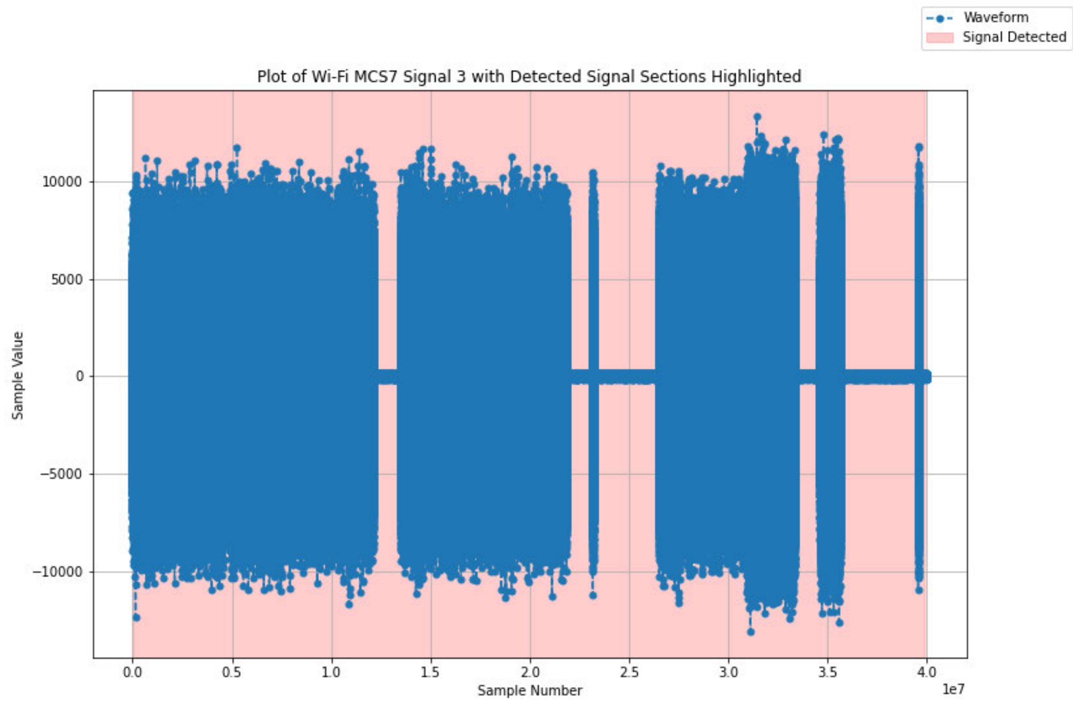

Figure 4.46: Wi-Fi 6 MCS7 Signal 3 with highlighted sections depicting the presence of a signal

By inspection of the signals with highlighted sections, Figures 4.32 to 4.46 on Pages 67 to 74, it appears that signal detection is successful, as the areas of relatively low signal activity (low sample value) being identified to contain a signal.

This is under the assumption that the Subray (2023) dataset contains no additive or correlative noise, suggesting that areas of relatively low signal activity (sample value) contains a signal. This assumption was discussed previously in Section 4.4 (calibration using custom waveforms).

Thus, with the Figures 4.32 to 4.46 visually examined that a signal is present throughout all of the Subray (2023) dataset's signal waveforms, further examination is required of the numerical results to confirm a signal is detected for the entire signal waveform and to identify differences between signal types.

The numerical results minimums, maximums and averages (where relevant) have been listed in Table 4.2 and Table 4.3 on the following pages (Page 76 and Page 77).

Table 4.2: Tabulated Statistical Results from 5G NR and LTE Signal Detection Tests

Signal	Eigenvalue Ratio ($\lambda_{max}/\lambda_{min}$)			Threshold Constant (γ)	Signal Presence (Boolean, 0 or 1) Average
	Minimum	Maximum	Average		
5G-NR 1	2.4451	22.8274	4.3373	1.0298	1.0
5G-NR 2	2.9557	51.8365	8.4339	1.0298	1.0
5G-NR 3	2.5915	28.5279	6.3321	1.0298	1.0
LTE (30mbps) 1	1.1378	3.4516	1.6513	1.0298	1.0
LTE (30mbps) 2	1.2666	2.6212	1.6102	1.0298	1.0
LTE (30mbps) 3	1.1704	3.7619	1.6053	1.0298	1.0
LTE (50mbps) 1	1.1879	3.3823	1.6776	1.0298	1.0
LTE (50mbps) 2	1.1261	3.4439	1.4811	1.0298	1.0
LTE (50mbps) 3	1.1274	4.5145	1.8293	1.0298	1.0

Note: Due to how the Threshold (γ) is calculated, all signals will share the same threshold value.

Note 2: Table 4.2 contains all the relevant numerical results for all of the 5G NR, LTE (30mbps) and LTE (50mbps) signals. Table 4.3 contains all the relevant numerical results for all of the Wi-Fi 6 MCS6 and MCS7 signals.

Note 3: All values are rounded to 4 decimal places.

Table 4.3: Tabulated Statistical Results from Wi-Fi 6 MCS6 and MCS7 Signal Detection Tests

Signal	Eigenvalue Ratio ($\lambda_{max}/\lambda_{min}$)			Threshold Constant (γ)	Signal Presence (Boolean, 0 or 1) Average
	Minimum	Maximum	Average		
Wi-Fi 6 MCS6 1	1.2349	2.1233	1.3766	1.0298	1.0
Wi-Fi 6 MCS6 2	1.2342	2.6028	1.4550	1.0298	1.0
Wi-Fi 6 MCS6 3	1.2587	1.9030	1.4071	1.0298	1.0
Wi-Fi 6 MCS7 1	1.2404	1.7885	1.4480	1.0298	1.0
Wi-Fi 6 MCS7 2	1.2179	1.9982	1.3903	1.0298	1.0
Wi-Fi 6 MCS7 3	1.2280	2.6351	1.5668	1.0298	1.0

Note: Due to how the Threshold (γ) is calculated, all signals will share the same threshold value.

Note 2: Table 4.2 contains all the relevant numerical results for all of the 5G NR, LTE (30mbps) and LTE (50mbps) signals. Table 4.3 contains all the relevant numerical results for all of the Wi-Fi 6 MCS6 and MCS7 signals.

Note 3: All values are rounded to 4 decimal places.

Table 4.4: Signal Group and Average Minimum Eigenvalue Ratio from Table 4.2 and Table 4.3

Signal Group	Avg. Min. Eigenvalue Ratio
5G-NR	2.664
LTE (30mbps)	1.192
LTE (50mpbs)	1.147
Wi-Fi 6 MCS6	1.243
Wi-Fi 6 MCS7	1.229

Inspecting Table 4.2 and Table 4.3, it appears that the signal detection code has successfully detect the presence of a signal throughout all samples, confirming expectations from visual inspection of the figures (Figures 4.32 to 4.46 on Pages 67 to 74).

By inspection, the maximum eigenvalue ratios from Table 4.2 for the 5G-NR signals appear to be outliers. For all other signals, the maximum and average eigenvalue ratios appear to be consistent and within a reasonable range, suggesting that LTE will be detected more often than Wi-Fi 6 Signals due to a higher average eigenvalue ratio.

On the other hand, by comparing the minimum eigenvalue ratios with the threshold constant, it appears that 5G-NR signals are significantly more likely to be detected than LTE and Wi-Fi 6 signals, due to the higher minimum eigenvalue ratios. If only minimum eigenvalue ratios and threshold constant is considered, then Wi-Fi 6 signals will be easier to detect than LTE signals. The numerical average minimum eigenvalue ratio for each signal group can be seen below in Table 4.4, which supports this observation.

Considering that consistent and accurate signal detection is desired, the minimum eigenvalue ratio will be considered as a comparable measure of signal detectability, if a signal is present for all samples of the Subray (2023) dataset signals.

Therefore, in the absence of other statistics, it appears that 5G-NR signals are the easiest to detect followed by Wi-Fi 6 signals and LTE signals using Maximum-Minimum Eigenvalue detection.

4.6 Signal Detection and Noise

Due to time constraints, only two trials could be completed. To investigate the effects of noise on the signal detection code, Gaussian noise was multiplied and added to Wi-Fi 6 MCS7 Signal 1 in two seperate tests.

It was expected that the correlative (multiplicative) noise and added noise would negatively affect the signal detection code's ability to detect the signal. The correlative and additive noise signal detection graphs can be seen in Figure 4.47 and Figure 4.48 respectfully on the following page. The numerical results for the additive, correlative and original Wi-Fi 6 MCS7 Signal 1 can be seen in Table 4.5.

By inspection of Figure 4.47 and Figure 4.48, it appears that additive and correlative noise had no effect on the signal detection.

However, by comparison of the eigenvalues of in Table 4.5, it is clear that correlative noise reduces the eigenvalue ratio while additive noise increases the eigenvalue ratio.

This suggests, that additive noise increases the chance of a false alarm while correlative noise decreases the chance of detection. Thus, in low signal-to-noise ratios, additive noise will be detected as a signal while correlative noise will reduce the visibility of a transmitted signal.

Table 4.5: Wi-Fi 6 MCS7 Signal 1 and Noise

Statistic	Original	Correlated	Additive
Min. Eigenvalue Ratio.	1.2404	1.1820	5.0798
Max. Eigenvalue Ratio.	1.7885	1.5292	16.3330
Avg. Eigenvalue Ratio.	1.4480	1.3190	9.9817
Signal Presence	1.0	1.0	1.0

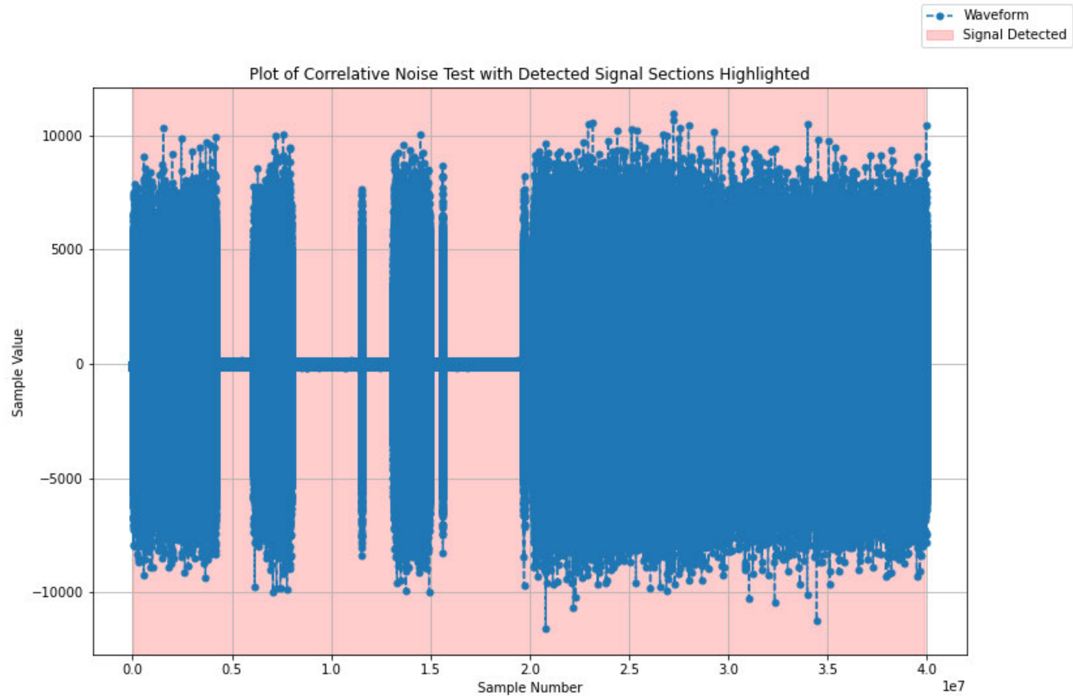

Figure 4.47: Wi-Fi 6 MCS7 Signal 1 with Multiplicative Noise

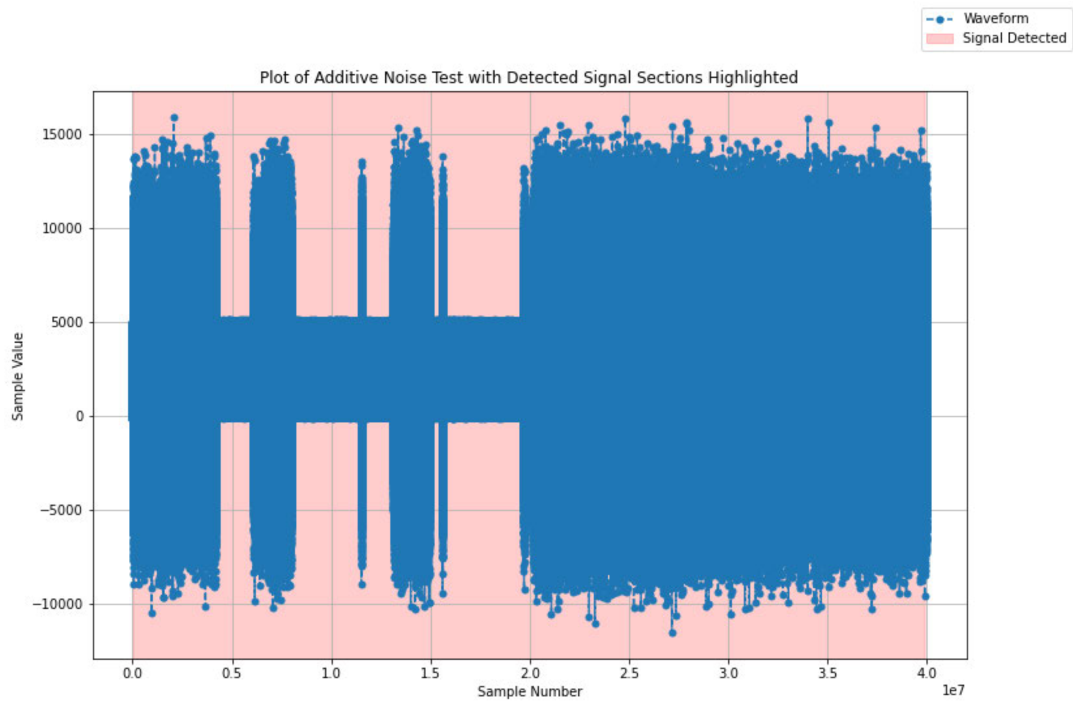

Figure 4.48: Wi-Fi 6 MCS7 Signal 1 with Additive Noise

4.7 Discussion and Summary of Results

4.7.1 Summary

1. Code Completeness Statement (Section 4.1)
    - Extraction, Graphing and MME Algorithm operational.
    - Added noise code was implemented and tested briefly.
    - Automated creation of custom signals not implemented.
2. Extraction of Data (Section 4.2)
    - Data extraction successful.
    - Data graphing successful.
    - Selected Wi-Fi 6 MCS7 Signal 1 for calibration due to low sample value zones.
3. Calibration of Signal Detection using Wi-Fi 6 MCS7 Signal 1 (Section 4.3)
    - Detected signal highlighting on Signal Graph is successful.
    - Signal detection stripes present (alternating between detected and absent signal).
    - Low sample value zones appear to contain a signal.
    - Identified N_s =100 000 and $L = 5$ as ideal values for testing.
4. Calibration of Signal Detection using Custom Waveforms (Section 4.4)
    - Verified the signal detection code correctly determining absence of signal.
    - Discovered signal present in low sample value zones.
5. General Signal Detection (Section 4.5)
    - All signals correctly detected.
    - 5G-NR prone to large maximum eigenvalue ratios.
    - LTE has higher average eigenvalue ratio compared to Wi-Fi 6.
    - Wi-Fi 6 has higher minimum eigenvalue ratio compared to LTE.
    - 5G-NR has highest minimum eigenvalue ratios.
    - 5G-NR found to be easiest to detect.
    - LTE found to be hardest to detect (min. eigenvalue ratio)
    - Wi-Fi 6 slight easier than LTE to detect (min. eigenvalue ratio).

4.7.2 Discussion

Unfortunately, due to a lack of time, additional datasets could not be integrated into the project. As a result, statistical methods of evaluation, such as t-tests, where not conducted due to a small sample size.

Within the small sample size, the results show that the 5G New Radio signals (as extracted from the dataset) are the easiest to detect, followed by Wi-Fi 6 signals in Maximum-Minimum Eigenvalue (MME) detection.

As the Australian Communications and Media Authority plans to have 5G and Wi-Fi 6 operate in the same frequency band (Communications & Authority 2024, pp. 20-21), the limited results suggest that a cognitive radio methodology or device could be applied to reduce Wi-Fi 6 and 5G interference.

However, this is reliant on 5G data not being influenced by additive or correlative noise, which may have resulted in the large maximum eigenvalue ratios.

Future investigations into cognitive radio applications in the microwave spectrum should include the construction of signal simulators for 5G, Wi-Fi 6 and other common microwave spectrum signals, as the use of an online dataset that does not provide information regarding noise levels or method of data capture can hamper calibration of the signal detection code.

It was assumed previously that the low sample values zone or low signal activity areas of the dataset signals contained the presence of a signal. This assumption was made in the absence of practical information regarding how the dataset was captured/generated.

No real-world issues have been observed with LTE signals, but this is likely due to LTE signals operating outside the ISM bands (comparison between known LTE operating frequencies in Section 2.4.2 and ISM bands in Section 2.4.4). Further testing of LTE and Wi-Fi 6 with a larger dataset may suggest that LTE and Wi-Fi 6 can be detected at similar eigenvalue ratios. I.E., significantly large datasets of LTE and Wi-Fi 6 may show an insignficant difference in eigenvalue ratios.

As such, it is possible to apply cognitive radio techniques to both Wi-Fi 6 and 5G in their shared band of operation.

Chapter 5

Conclusions and Further Work

5.1 Conclusions

With the upcoming changes to 5G and Wi-Fi 6 within the microwave spectrum, a review of applicable cognitive radio signal detection techniques has been completed.

Using a maximim-minimum eigenvalue method and Python code, signal detection testing was conducted on an online dataset.

Successful application of the maximum-minimum eigenvalue detection method has been completed, with the results suggesting a cognitive radio system could be applied to 5G and/or Wi-Fi 6 within their shared frequency band.

Limited testing of the effect of noise on maximum-minimum eigenvalue detection has been completed, suggesting both correlative and additive Gaussian noise can negatively impact the signal detection code.

Further work is required to evaluate LTE, 5G New Radio and Wi-Fi 6 signals as the online dataset provides limited information regarding noise presence.

5.2 Achievement of Project Objectives

5.2.1 Discussion

The following subsections are the listed project objectives from Chapter 1 Section 1.2

Identification of signal detection techniques for microwave spectrum signals

This objective has been achieved.

The literature review has identified several signal detection techniques that have and can be applied to microwave spectrum signals. Although some were not discussed in detail, cooperative methods or signal detection methods like cyclostationary detection that requires prior knowledge could be applied to microwave spectrum signals.

For most signal detection techniques, it is a question of how the signal detection technique was applied. In this case, while non-cooperative signal detection or spectrum sensing techniques were chosen for testing, the identified cooperative and non-cooperative signal detection techniques could be applied in further work.

Identification of frequency bands within the microwave spectrum that are currently underutilised or at high-risk of interference

This objective has been partially achieved.

All ISM bands have been identified, as well as upcoming changes to both 5G and Wi-Fi 6. However, underutilisation or interference was not investigated aside from the ISM band at 2.4 GHz.

Identification of signals operating within high-risk or underutilised microwave frequency bands

This objective has been partially achieved.

The upcoming shared frequency allocation for 5G and Wi-Fi 6 represents two signals potentially operating in a over-utilised frequency band. Additional signals should have been identified.

Creation or collection of a dataset containing signals from the microwave spectrum

This objective has been achieved.

The Subray (2023) dataset contains signals from the microwave spectrum. However, additional datasets containing other microwave spectrum signals is needed.

Creation of a signal detection testing environment

This objective has been achieved.

Python code has been successful in testing signal detection.

Successful detection of a microwave frequency signal using a chosen signal detection technique

This objective has been achieved.

The results section (Section 4) contain several signal waveforms that have been successfully identified as containing a signal.

Verification of the impact of noise in signal detection

This objective has been partially achieved.

Limited testing has been conducted to examine the effect of noise on Maximum-Minimum Eigenvalue detection.

Statistical analysis of signal detection results

This objective has not been achieved.

Statistical analysis requires larger datasets than the Subray (2023) dataset used in this project.

Creation and testing of a cognitive radio system in a real-world environment

This objective has not been achieved.

This objective has been abandoned in favour of simulation testing, due to a lack of time.

5.3 Further Work

5.3.1 Simulation Work

Further simulation work applying cognitive radio techniques to microwave spectrum signals is required as only three signals within the microwave spectrum were examined.

Additionally, the dataset used in this project was limited and provided insufficient information regarding the presence of noise and the signal recording method.

Further work could be conducted using alternative datasets or constructed signal generators to ensure the noise presence or signal recording method is known.

Alternatively, further work with 5G and Wi-Fi 6 could be conducted to determine if a cognitive radio system within a shared frequency band could accurately detect simulated 5G and Wi-Fi 6 signals.

5.3.2 Real-World Testing

With Wi-Fi 6 and 5G being allocated a shared frequency band, further work could be conducted to confirm the previous findings that 5G signals were the easiest to detect. Alternatively, real-world testing or data collection could be completed collect real-world signal datasets and to estimate current transmitting environments and conditions from collected data.

References

Ali, S. S., Zhao, W., Jin, M. & Yoo, S.-J. (2019), Enhanced Maximum-Minimum Eigenvalue Based Spectrum Sensing, 2019 International Conference on Information and Communication Technology Convergence (ICTC), IEEE, Jeju, Korea (South).

Althaf, C. I. M. & Prema, S. C. (2018), Covariance and eigenvalue based spectrum sensing using USRP in real environment, 2018 10th International Conference on Communication Systems & Networks (COMSNETS), IEEE, Bengaluru, India.

Armi, N., Saad, N. M., Arshad, M. & Y, M. Z. (2010), Cooperative spectrum sensings and signal detection in congitive radio, 2010 International Conference on Intelligent and Advanced Systems, IEEE, Kuala Lumpur, Malaysia.

Australian Communications and Media Authority (2024), 'Five-year spectrum outlook 2024-29', https://www.acma.gov.au/five-year-spectrum-outlook. Government Report.

Australian Radiofrequency Spectrum Plan (2021). (Cwlth).

Captain, K. M. & Joshi, M. V. (2018), SNR wall for cooperative spectrum sensing using generalized energy detector, 2018 10th International Conference on Communication Systems & Networks (COMSNETS), IEEE, Bengaluru, India.

Communications, A. & Authority, M. (2024), 'Australia Radiofrequency Spectrum Plan', https://www.acma.gov.au/australian-radiofrequency-spectrum-plan. viewed 11/11/2024.

CueMath (2024), 'Covariance Matrix', https://www.cuemath.com/algebra/covariance-matrix/. viewed 11/11/2024.

de Carvalho, F. B. S., Lopes, W. T. A. & Alencar, M. S. (2015*a*), Spectrum sensing via statistic tests in Rayleigh fading channels, 2015 SBMO/IEEE MTT-S International Microwave and Optoelectronics Conference (IMOC), IEEE, Porto de Galinhas, Brazil.

de Carvalho, F. B. S., Lopes, W. T. A. & Alencar, M. S. (2015*b*), Spectrum sensing via statistic tests in Rayleigh fading channels, SBMO/IEEE MTT-S International Conference on Microwave and Optoelectronics, IEEE, Porto de Galinhas, Brazil. DOI: 10.1109/IMOC.2015.7369084.

Gills, A. S., Jones, D. & Beaver, K. (2024), 'LTE (Long-Term Evolution)', `https://www.techtarget.com/searchmobilecomputing/definition/Long-Term-Evolution-LTE`. viewed 11/11/2024.

Google (2024), 'What is Deep Learning?', `https://cloud.google.com/discover/what-is-deep-learning`.

Guiliano, A., Hilal, W., Alsadi, N., Gadsen, S. A. & Yawney, J. (2022), A Review of Cognitive Dynamic Systems and Cognitive IoT, 2022 IEEE International IOT, Electronics and Mechatronics Conference (IEMTronics), IEEE, Toronto, ON, Canada.

Hanna, S. A. & Sydor, J. (2012), Distributed sensing of spectrum occupancy and interference in outdoor 2.4 GHz Wi-Fi networks, 2012 IEEE Global Communications Conference (GLOBECOM), IEEE, Anaheim, CA, USA.

Haykin, S. (2005), 'Cognitive radio: brain-empowered wireless communications', *IEEE Journal on Selected Areas in Communications* **vol. 23, no. 2**, pp. 201–220.

Intel Technologies (2024), 'What Is WiFi 6?', `https://www.intel.com/content/www/us/en/gaming/resources/wifi-6.html`. viewed 11/11/2024.

Ivanenko, S. & Bezruk, V. (2016), Methods of detecting of unknown signals in cognitive radio networks, 2016 Third International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T), IEEE, Kharkiv, Ukraine.

James, G., Burley, D., Clements, D., Dyke, P., Searl, J. & Wright, J. (2015), fifth edn, Pearson Education Limited, Harlow, United Kingdom, chapter The eigenvalue problem, pp. 389 – 409.

Kakalou, I., Psannis, K., Goudos, S. K., Yioultsis, T. V., Kantartzis, N. V. & Ishibashi, Y. (2019), Radio Environment Maps for 5G Cognitive Radio Network, 2019 8th International Conference on Modern Circuits and Systems Technologies (MOCAST), IEEE, Thessaloniki, Greece.

Kalamkar, S. S. & Banerjee, A. (2013), On the performance of generalized energy detector under noise uncertainity in cognitive radio, 2013 National Conference on Communications (NCC), IEEE, New Delhi, India.

Kot, A. C., Teh, K. C., Razul, S. G. & Su, Z. (2022), 'Deep Non-Cooperative Spectrum Sensing Over Rayleigh Fading Channel', *IEEE Transactions on Vehicular Technology* **vol. 71, no. 4**, pp. 4460–4464. DOI: 10.1109/TVT.2021.3138593.

Luo, J., Wang, S., Zhang, E. & Luo, J. (2015), Non-cooperative signal detection in alpha stable noise via Kolmogorov-Smirnov test, 2015 8th International Congress on Image and Signal Processing (CISP), IEEE, Shenyang, China.

Mucchi, L. & Carpini, A. (2014), ISM band aggregate interference in BAN-working environments, 2014 8th International Symposium on Medical Information and Communication Technology (ISMICT), IEEE, Firenze, Italy.

National Institute of Standards and Technology (2022), 'The Spectrum Crunch', `https://www.nist.gov/advanced-communications/spectrum-crunch`.

North American Space Administration (2010), 'Microwaves', `https://science.nasa.gov/ems/06_microwaves/`.

Ramyea, R. & Kasthuri, N. (2018), Performance Analysis of Non Cooperative Spectrum Sensing Schemes in 5G Cognitive Radio Networks, 2018 International Conference on Intelligent Computing and Communication for Smart World (I2C2SW), IEEE, Erode, India.

RF Wireless World (2024*a*), '802.11ax MCS Table — 802.11ax MCS Rates', `https://www.rfwireless-world.com/Terminology/802.11ax-MCS-Rates-Table.html`. viewed 11/11/2024.

RF Wireless World (2024*b*), 'WLAN 802.11ax Tutorial: WiFi 6 Basics', `https://www.rfwireless-world.com/Tutorials/WLAN-802-11ax-tutorial.html`. viewed 11/11/2024.

Sharma, S. K., Chatzinotas, S. & Ottersten, B. (2014), Maximum Eigenvalue detection for spectrum sensing under correlated noise, 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, Florence, Italy.

Shekhawat, G. K. & Yadav, R. P. (2015), Non-cooperative spectrum sensing techniques in cognitive radio-a survey, 2015 IEEE Technological Innovation in ICT for Agriculture and Rural Development (TIAR), IEEE, Chennai, India.

Shekhawat, G. K. & Yadav, R. P. (2021), Review on Classical to Deep Spectrum Sensing in Cognitive Radio Networks, 2021 Sixth International Conference on Wireless Communications, Signal Processing and Network (WiSPNET), IEEE, Chennai, India.

Subray, S. (2023), 'Real-World Wireless Communication Dataset: IEEE 802.11AX, LTE, AND 5G-NR SIGNALS', https://www.kaggle.com/datasets/siddss/real-world-wireless-communication-dataset/data. Used under an Apache License, Version 2.0, (https://www.apache.org/licenses/LICENSE-2.0).

Wei, F., Zheng, S., Zhou, X., Zhang, L., Lou, C., Zhao, Z. & Yang, X. (2022), 'Detection of Direct Sequence Spread Spectrum Signals Based on Deep Learning', *IEEE Transactions on Cognitive Communications and Networking* **vol. 8, no. 3**, pp. 1399–1410. DOI: 10.1109/TCCN.2022.3174609.

Wikipedia (2024*a*), '5G NR — Wikipedia, the Free Encyclopedia', https://en.wikipedia.org/wiki/5G_NR. viewed 11/11/2024.

Wikipedia (2024*b*), 'Covariance Matrix', https://en.wikipedia.org/wiki/Covariance_matrix. viewed 11/11/2024.

Wikipedia (2024*c*), 'List of 5G NR Networks — Wikipedia, the Free Encyclopedia', https://en.wikipedia.org/wiki/List_of_5G_NR_networks. viewed 11/11/2024.

Wikipedia (2024*d*), 'LTE frequency bands — Wikipedia, the Free Encyclopedia', https://en.wikipedia.org/wiki/LTE_frequency_bands. viewed 11/11/2024.

Wituski, J. & Dietl, G. (2020), Interference Investigation of Wireless ISM Band Devices in Rescue Helicopters, 2020 IEEE 5th International Symposium on Smart and Wireless Systems within the Conferences on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS-SWS), IEEE, Dortmund, Germany.

Yawada, P. S., Wei, A. J. & Kiki, M. M. (2015), Performance evaluation of energy detection based on non-cooperative spectrum sensing in cognitive radio network, 2015 4th International Conference on Computer Science and Network Technology (ICCSNT), IEEE, Harbin, China.

Zeng, Y. & Liang, Y.-C. (2009), 'Eigenvalue-based spectrum sensing algorithms for cognitive radio', *IEEE Transactions on Communications* **vol. 57, no. 6**, pp. 1784 – 1793.

Zeng, Y., Liang, Y.-C. & Koh, C. L. (2008), Maximum Eigenvalue Detection: Theory and Application, 2008 IEEE International Conference on Communications, IEEE, Beijing, China.

Appendix A

Project Specification

Project:

Project Title: Cognitive Radio for Microwave Spectrum Access

Name: Joshua Knipe

Student ID: [redacted]

Supervisor: John Leis

Specifications:

Introduction and Background:

Within the microwave spectrum, there is a continuous demand for increased frequency band allocations to new and emerging devices. In particular, the emergence of 5G communications and Internet of Things (IoT) devices has caused the Australian Communications and Media Authority to review and reduce existing bandwidth allocations to accommodate 5G and IoT devices (ACMA 2019, pp. 13 - 17).

In the roll-out of these changes, some frequency bands have been identified for optimisation to reduce their required spectrum band (ACMA 2019, p. 15). Thus, there is a need for spectrum optimisation methods within the microwave spectrum (300 MHz to 300 GHz) to allow for further reduction in allocated bandwidths.

One such method of reducing required bandwidth, is the use of cognitive dynamic systems (cognitive radio). These systems are 'smart' systems able to determine available transmission frequencies in a complex transmission environment by observing and learning from the transmission environment (Alsadi et al. 2022). Various methods of cognitive radio have been simulated and may prove effective at identifying valid frequencies for transmission (Kasthuri & Ramyea 2018) in a simulated transmission environment. In theory, cognitive radio should allow for the detection and utilisation of unused frequencies within a selected frequency band.

Additionally, cognitive radio could be used to avoid collisions/interference with high-risk devices or frequency bands, such as 5G and radio altimeters used in aircraft. Although there is no sign of interference between 5G and radio altimeters in Australia despite significant concerns in the United States of America (Department of Infrastructure, Transport, Regional Development, Communications and the Arts 2022), the Civil Aviation Safety Authority (2023) has stated that they will have a mitigation frequency band. This is one such example that could benefit from the use of cognitive radio techniques or systems to reduce the potential of interference significantly, aside from the use of frequency mitigation bands.

Thus, it is hoped that this project will be able to identify problematic areas within the microwave spectrum and construct a viable dynamic cognitive system (cognitive radio system) able to correctly identify valid transmission frequency ranges within problematic frequency bands.

Objective and Aims:

Using cognitive radio principles and techniques, the project will aim to create a cognitive radio system for use in problematic bands of the microwave spectrum. It is expected that industrial, scientific, and medical (ISM) frequency bands within the microwave spectrum, as well as 5G and IoT operating bands will be the primary focus of investigation.

Objectives:

- Identify available signal detection techniques for microwave spectrum signals and frequency bands.
- Identify high-traffic and high-risk frequency bands within the microwave spectrum.
- Identify communication schemes used in high-use and high-risk channels within the microwave spectrum.
- Collect or create a dataset of various microwave frequency signals for simulation.
- Create a simulated transmission environment.
- Apply identified signal detection techniques to selected microwave frequency channels via controlled simulation.
- Obtain a programmable VNA and perform real-world signal detection testing for a limited number of transmitting devices.
- Perform statistical analysis on both simulated and real-world tests.

Expected Outcomes:

- Identification of cognitive radio techniques applicable to the microwave spectrum.
- Understanding the high-use and high-risk channels within the microwave frequencies and the expected signal behaviour.
- Creation of a dataset and simulated testing environment containing various signals operating on various frequencies within a band.
- Successful identification of available frequency space for transmission in a simulated and real-world transmission environment.
- Recommendations for applicable cognitive radio methods and for further research.

Work Plan:

Timeline:

Month	Goal	Content
Jan - Feb	Choose and Plan Project	• Choose project. • Complete generalised work plan.
February	Review Detection Methods	• Review literature regarding cooperative and non-cooperative signal detection. • Review literature regarding cognitive radio applications. • Review literature regarding current spectrum sharing methodologies.
March	Review Microwave Spectrum and Draft Literature Review	• Review literature regarding both 5G and IoT in the context of transmissions, signals, spectrum sharing and expected behaviour of signals within relevant bands. • Review and determine high traffic and/or high-risk communication bands within the microwave spectrum to determine if cognitive radio applications are relevant. • Review additional or required material for literature review. • Draft literature review.
April	Finalise Literature Review and Construct Experimental Methodology	• Finalise literature review. • Apply signal detection methods reviewed in February/March to methodology design. • Determine if valid datasets exist for cognitive radio detection in the microwave spectrum. • If no valid datasets exist, create method to simulate a dataset (via real-world data capture or simulation of known signal behaviour).
May	Finalise Methodology	• Communicate with supervisor and obtain relevant permissions (risk-assessment and other material). • Construct the methodology report.

June - August	Collect data and perform analysis	• If needed, collect real-world data from laboratory or anechoic chamber (June/July). • Create environment for signal detection to test cognitive radio application (June/July). • Test cognitive radio method(s) on created simulation of transmission environment (July/Aug). • Perform statistical analysis (if possible) on cognitive radio methods to determine most viable method for selected frequency band (if multiple methods and/or frequency bands have been selected).
August - September	Draft Dissertation	• Construct a Dissertation draft for submission, communicating with supervisor regarding quality of draft.
September - October	Dissertation finalisation, Reflection and Presentation	• If needed, continue, and finalise on-going data collection/analysis. • Finalise Dissertation when possible. • Complete reflection assignment. • Construct presentation, preferably using available data. • Practise presentation.
Late October - November	Finalise Dissertation	• Finalise the Dissertation for submission, communicating where needed to supervisor.
November - December	Await Results	• Await submission results. • Submit Dissertation if not already submitted.

Note: The above work-plan is generalised and subject to change according to UniSQ notifications (e.g. change of presentation dates/month etc.).

Required Resources:

At this stage, the required resources listed below are for the worst-case scenario where transmission datasets cannot be obtained from available academic or industrial sources. Possible dataset sources are listed for posterity.

Equipment:

- Programmable Vector Network Analyser (VNA) and appropriate peripherals to collect transmission data and test cognitive radio techniques.
- Various (microwave frequency) antennas and/or (microwave frequency) transmission equipment (e.g radios, Bluetooth devices, IoT devices) for testing cognitive radio techniques and data collection.
- Access to the UniSQ Toowoomba Anechoic Chamber or engineering laboratories.

Software:

- MATLAB or Python (with NumPy, SciPy and other relevant modules) for simulation and data analysis.
- Microsoft Office or LaTeX to create and finalise reports.

Access:

- IEEE Xplore (Database) for literature review.
- ScienceDirect (Database) for literature review.
- United States of America, National Library of Medicine, PubMed Central for literature review.
- EBSCOHost Megafile Ultimate (Database) for literature review.
- IEEE Dataport (and other data sources) as data sources for simulation.

References:

Australian Communications and Media Authority 2019, *Five-year spectrum outlook 2019-2023*, Government Report, Australian Government, viewed 09/02/2024, <https://www.acma.gov.au/publications/2019-09/plan/five-year-spectrum-outlook-2019-23>

Alsadi, N, Gadsden, SA, Giuliano, A, Hilal, W & Yawney, J 2022, 'A Review of Cognitive Dynamic Systems and Cognitive IoT', *IEEE International IOT, Electronics and Mechatronics Conference 2022*, 01-04 June 2022, Toronto, Canada, <https://ieeexplore.ieee.org/document/9795834>

Civil Aviation Safety Authority 2023, *CASA to work with aviation sector on 5G*, Government News Article, Australian Government, viewed 09/02/2024, <https://www.casa.gov.au/about-us/news-media-releases-and-speeches/casa-work-aviation-sector-5g>

Department of Infrastructure, Transport, Regional Development, Communications and the Arts 2022, *5G no issue for aircraft in Australian airspace*, Government News Article, Australian Government, viewed 09/02/2024, <https://www.infrastructure.gov.au/department/media/news/5g-no-issue-aircraft-australian-airspace>

Kasthuri, N & Ramyea, R 2018, 'Performance Analysis of Non Cooperative Spectrum Sensing Schemes in 5G Cognitive Radio Networks', *International Conference on Intelligent Computing and Communication for Smart World 2018*, 14-15 December 2018, Erode, India, <https://ieeexplore.ieee.org/document/8997126>

Appendix B

Risk Assessment

5622	RISK DESCRIPTION	STATUS	TREND	CURRENT	RESIDUAL
	ENP4111 - Cognitive Radio for Microwave Spectrum Access	Live		Low	Very Low

RISK OWNER	RISK IDENTIFIED ON	LAST REVIEWED ON	NEXT SCHEDULED REVIEW
Joshua Knipe	30/08/2024	06/11/2024	06/11/2025

RISK FACTOR(S)	EXISTING CONTROL(S)	CURRENT	PROPOSED CONTROL(S)	TREATMENT OWNER	DUE DATE	RESIDUAL
Coding, testing and simulation for the completion of the project is completed on a custom computer, which exhibits strange behavior at times, resulting in rare Blue Screen of Death crashing. There exists a risk of loss of data during a crash.	Control: Work is saved regularly and backed up to physical storage devices daily.	Very Low	No Control:			Very Low
Project work can only be completed at a computer station due to the nature of online journals and signal detection simulation. There exists small risks from prolonged static postures.	Control: Ergonomic workspace aids and equipment are present and used at the computer station.	Very Low	No Control:			Very Low
Project work requires the use of available digital datasets obtainable from internet repositories. There exists a risk of misuse due to student inexperience and|or illegal uploading of datasets.	Control: Investigation to the relevant requirements for common | free use clauses regarding datasets, and|or the appropriate accreditation.	Low	Use of the correct accreditation within existing UniSQ requirements and specified requirements on the given dataset's download location should eliminate the risk, if the accreditation is completed correctly.		04/11/2024	Very Low

Appendix C

Python Code

The Python Code used to generate the results in Chapter 4 is listed on the following pages.

The code is split into individual functions.

To generate the results, the Python code will execute the *main()* function first, and run through the functions *extract_and_graph()*, *man_calibrate()*, *test_custom_wave()* and *extract_and_test()* in the order listed.

Function *extract_and_graph()* corresponds to the results in Chapter 4 Section 4.2.

Function *man_calibrate()* corresponds to the results in Chapter 4 Section 4.3.

Function *test_custom_wave()* corresponds to the results in Chapter 4 Section 4.4.

Function *extract_and_test()* corresponds to the results in Chapter 4 Section 4.5.

To stop these functions from executing, locate the *main()* function and comment out the undesired function(s) using # which represents a single line comment in Python.

Listing C.1: Python Code

```
# -*- coding: utf-8 -*-
"""
@author: Joshua John Knipe [redacted]
"""

import numpy as np
import scipy.linalg as spilin
import matplotlib.pyplot as plt
from TracyWidom import TracyWidom

def loadTracyWidom(betaInput):
    """
    Code to load Tracy-Widom distribution based on Beta value.

    Parameters
    ----------
    betaInput : Integer
        Value of 1,2 or 4, to allow select of correct T-W Distribution.

    Returns
    -------
    TWDist : Object(?)
        Tracy-Widom Distribution for selected beta value.
    """
    if betaInput == 1:
        # Distribution Order 1
        TWDist = TracyWidom(betaInput)
        return TWDist
    elif betaInput == 2:
        # Distribution Order 2
        TWDist = TracyWidom(betaInput)
        return TWDist
    elif betaInput == 4:
        # Distribution Order 4
        TWDist = TracyWidom(betaInput)
        return TWDist
    else:
        TypeError("Tracy-Widom: Beta outside required values!")
        return None

def create_cov_mat(xn, L):
    """
    Code to generate the covariance matrix for MME Signal Detection.

    Parameters
    ----------
    xn : NumPy Array
        Truncated sample vector
    L : Integer
        Smoothing factor
```

```
    Returns
    -------
    Rx : NumPy Array
        Square Sample Covariance Matrix

    """
    L_arr = np.arange(1, L)
    xhat = np.zeros([L, xn.size])
    xn_len = xn.size
    xhat[0, :] = xn
    for i in L_arr:
        xn_temp = np.zeros(xn.size)
        xn_len = xn[i:].size
        xn_temp[0:xn_len] = xn[i:]
        xhat[i, :] = xn_temp
    xhat_tpose = xhat.transpose()
    Rx = np.zeros([L, L])
    xn_len = xn.size
    L_arr = np.arange(0, xn_len)
    for i in L_arr:
        Rx = Rx + (xhat[:, i].reshape(L, 1)*xhat_tpose[i, :].reshape(1, L))
    Rx = Rx/xn_len
    return Rx

def MaxMinEigAlgo(data,frameSize,M,L,Pfa):
    """
    Code to seperate dataset into frames and perform MME
    detection on each frame.

    Parameters
    ----------
    data : NumPy Array
        Extracted data from Dataset
    frameSize : Integer
        Ns: Number of samples per frame
    M : Integer
        Number of Inputs (not relevant, deprecated)
    L : Integer
        Smoothing factor.
    Pfa : TYPE
        Probability of False Alarm

    Returns
    -------
    [Eigenvalue Ratio Array, Threshold Array, Signal Presence Array]

    """
    # Pre-load TracyWidom
    TWDistMME = loadTracyWidom(1)
    # Setup Rx
    Ns_all = np.size(data)
```

```
    No_Of_Frame = np.arange(0, np.floor(Ns_all/frameSize).astype(int))
    ratioEig = np.zeros((No_Of_Frame.size,1))
    gam1 = np.zeros((No_Of_Frame.size,1))
    signal_pres = np.zeros((No_Of_Frame.size,1))
    for i in No_Of_Frame:
        frameStart = i*frameSize
        frameEnd = frameStart+frameSize
        temp_data = data[frameStart:frameEnd]
        Ns = np.size(temp_data)
        Xn = temp_data
        Rx = create_cov_mat(Xn,L)
        # Get EIGENVALUES
        EigVecVal = spilin.eigh(Rx, eigvals_only = True, overwrite_a = True)
        if np.size(EigVecVal[EigVecVal != 0]) != 0:
            maxEig = np.max(EigVecVal)
            minEig = np.min(EigVecVal[EigVecVal != 0])
            # SETUP RATIO
            ratioEig[i] = maxEig/minEig
        else:
            ratioEig[i] = 0
        # SETUP THRESHOLD
        gam1p1 = ((np.sqrt(Ns) + np.sqrt(M*L))**2)/((np.sqrt(Ns) - np.sqrt(M*L
        gam1p2 = 1 + (((np.sqrt(Ns) + np.sqrt(M*L))**(-2/3))/((Ns*M*L)**(1/6))
        gam1[i] = gam1p1 * gam1p2
        # COMPARE RATIO AND THRESHOLD
        if ratioEig[i] > gam1[i]:
            signal_pres[i] = True
        else:
            signal_pres[i] = False
    return [ratioEig, gam1, signal_pres]


def graphdata(data, dataname):
    """
    Code to generate graph of dataset.

    Parameters
    ----------
    data : NumPy Array
        DESCRIPTION.
    dataname : String
        Name of File / Graph.

    Returns
    -------
    None.

    """
    fig = plt.figure(figsize=(12, 8))
    ax = plt.subplot()
    ax.plot(data, linestyle='--', marker='.', markersize=10)
    ax.set_title(dataname)
```

```
    ax.set_xlabel('Sample')
    ax.set_ylabel('Sample Value')
    ax.grid()
    filename = dataname + '.png'
    fig.savefig(filename)

def savedata(data, dataname):
    """
    Code to save complete dataset, minimum, maximum and average.

    Parameters
    ----------
    data : NumPy Array
        Data to be saved to disk.
    dataname : String
        Name of datafile.

    Returns
    -------
    None.

    """
    with open(dataname, 'w+') as fi:
        for line in data:
            fi.write('{}\n'.format(line))
        fi.write('{}\n'.format(np.average(data)))
        fi.write('{}\n'.format(np.min(data)))
        fi.write('{}\n'.format(np.max(data)))
    fi.close()

def scale_data(data, span):
    """
    Code to rescale data for graphing.

    Parameters
    ----------
    data : NumPy Array
        Data to be rescaled.
    span : Integer
        Size to rescale data to.

    Returns
    -------
    new_data : NumPy Array
        Rescaled dataset.

    """
    new_data = np.zeros(span)
    data_part = np.linspace(0, span, data.size+1, dtype=int)
    for i in np.arange(0, data.size -1):
        new_data[data_part[i]:data_part[i+1]] = data[i]
    return new_data
```

```
def signal_graph(wave_data, signal_data, wave_title, graph_title):
    """
    Code to graph waveform and detected signal regions.

    Parameters
    ----------
    wave_data : NumPy Array
        Waveform dataset/array.
    signal_data : NumPy Array
        Detected Signal Data.
    wave_title : String
        Name of Saved Figure
    graph_title : String
        Graph Title

    Returns
    -------
    None.

    """
    #
    # Scale Data
    #
    X_span = np.linspace(0, wave_data.size-1, wave_data.size)
    signal_plot = scale_data(signal_data,wave_data.size)
    #
    # Setup Strings
    #
    fig_title = ('Plot of ' + graph_title
                 + ' with Detected Signal Sections Highlighted')
    figname = wave_title + '.png'
    #
    # Graph
    #
    fig = plt.figure(figsize=(12, 8))
    ax = plt.subplot()
    ax.plot(wave_data, linestyle='--', marker='.', markersize=10)
    ax.fill_between(X_span, 0, 1, where=signal_plot>0, color = 'r',
                    alpha = 0.2, transform = ax.get_xaxis_transform())
    ax.set_title(fig_title)
    ax.set_xlabel('Sample Number')
    ax.set_ylabel('Sample Value')
    ax.grid()
    fig.legend(['Waveform', 'Signal Detected'], loc = 'upper right')
    fig.savefig(figname)

def main():
    """
    Code to execute all code used to generate results.

    Returns
```

```
    -------
    None.

    """
    # When using this code, make sure to tag these out one by one,
    # Because it takes a while to run!
    extract_and_graph()
    man_calibrate()
    test_custom_wave()
    extract_and_test()

def noise_test():
    """
    Code to generate correlative and additive noise test results.

    Returns
    -------
    None.

    """
    #
    # Setup Variables
    #
    M = 1
    L = 5
    Pfa = 0.1
    Ns = 100000
    #
    # Setup RNG
    #
    rng = np.random.default_rng(12312112024)
    #
    # Load Wi-Fi 6 MCS7 Signal 1
    #
    cal_wave = np.fromfile('32274CA-015-D20211123T131300M012278.data',np.int16
    #
    # Noise 1 - Correlative
    #
    corr_wave = cal_wave*rng.random(cal_wave.size)
    ratio_corr, gam_corr, signal_corr = MaxMinEigAlgo(corr_wave, Ns, M, L, Pfa
    savedata(ratio_corr, 'ratio_noise_test_1.txt')
    savedata(gam_corr, 'gamma_noise_test_1.txt')
    savedata(signal_corr, 'signal_noise_test_1.txt')
    #
    # Noise 2 - Additive
    #
    signal = cal_wave + (rng.random(cal_wave.size)*5000)
    ratio_noise, gam_noise, signal_noise = MaxMinEigAlgo(signal,Ns, M, L, Pfa)
    savedata(ratio_noise, 'ratio_noise_test_2.txt')
    savedata(gam_noise, 'gamma_noise_test_2.txt')
    savedata(signal_noise, 'signal_noise_test_2.txt')
    #
```

```
    #
    #
    signal_graph(signal, signal_noise , 'Noise_Additive',
                 'Additive Noise Test')
    signal_graph(corr_wave, signal_corr , 'Noise_Correlated',
                 'Correlative Noise Test')


def man_calibrate():
    """
    Code to generate calibration results (non-custom waves).

    Returns
    -------
    None.

    """
    #
    # Setup Variables
    #
    M = 1
    L1 = 5
    L2 = 10
    L3 = 15
    Pfa = 0.1
    frameSize1 = 1000
    frameSize2 = 10000
    frameSize3 = 100000
    #
    # Wifi MCS7
    #
    cal_wave = np.fromfile('32274CA-015-D20211123T131300M012278.data',
                           np.int16)
    #
    # L1, Ns1
    #
    ratio_L1N1, gam_L1N1, signal_L1N1 = MaxMinEigAlgo(cal_wave, frameSize1,
                                                      M, L1, Pfa)
    savedata(ratio_L1N1, 'L1N1_ratio.txt')
    savedata(gam_L1N1, 'L1N1_gamma.txt')
    savedata(signal_L1N1, 'L1N1_signal.txt')
    #
    # L1, Ns2
    #
    ratio_L1N2, gam_L1N2, signal_L1N2 = MaxMinEigAlgo(cal_wave, frameSize2,
                                                      M, L1, Pfa)
    savedata(ratio_L1N2, 'L1N2_ratio.txt')
    savedata(gam_L1N2, 'L1N2_gamma.txt')
    savedata(signal_L1N2, 'L1N2_signal.txt')
    #
    # L1, Ns3
    #
```

```
ratio_L1N3, gam_L1N3, signal_L1N3 = MaxMinEigAlgo(cal_wave, frameSize3,
                                                  M, L1, Pfa)
savedata(ratio_L1N3, 'L1N3_ratio.txt')
savedata(gam_L1N3, 'L1N3_gamma.txt')
savedata(signal_L1N3, 'L1N3_signal.txt')
#
# L2, Ns1
#
ratio_L2N1, gam_L2N1, signal_L2N1 = MaxMinEigAlgo(cal_wave, frameSize1,
                                                  M, L2, Pfa)
savedata(ratio_L2N1, 'L2N1_ratio.txt')
savedata(gam_L2N1, 'L2N1_gamma.txt')
savedata(signal_L2N1, 'L2N1_signal.txt')
#
# L2, Ns2
#
ratio_L2N2, gam_L2N2, signal_L2N2 = MaxMinEigAlgo(cal_wave, frameSize2,
                                                  M, L2, Pfa)
savedata(ratio_L2N2, 'L2N2_ratio.txt')
savedata(gam_L2N2, 'L2N2_gamma.txt')
savedata(signal_L2N2, 'L2N2_signal.txt')
#
# L2, Ns3
#
ratio_L2N3, gam_L2N3, signal_L2N3 = MaxMinEigAlgo(cal_wave,
                                                  frameSize3, M, L2, Pfa)
savedata(ratio_L2N3, 'L2N3_ratio.txt')
savedata(gam_L2N3, 'L2N3_gamma.txt')
savedata(signal_L2N3, 'L2N3_signal.txt')
#
# L3, Ns1
#
ratio_L3N1, gam_L3N1, signal_L3N1 = MaxMinEigAlgo(cal_wave,
                                                  frameSize1, M, L3, Pfa)
savedata(ratio_L3N1, 'L3N1_ratio.txt')
savedata(gam_L3N1, 'L3N1_gamma.txt')
savedata(signal_L3N1, 'L3N1_signal.txt')
#
# L3, Ns2
#
ratio_L3N2, gam_L3N2, signal_L3N2 = MaxMinEigAlgo(cal_wave,
                                                  frameSize2, M, L3, Pfa)
savedata(ratio_L3N2, 'L3N2_ratio.txt')
savedata(gam_L3N2, 'L3N2_gamma.txt')
savedata(signal_L3N2, 'L3N2_signal.txt')
#
# L3, Ns3
#
ratio_L3N3, gam_L3N3, signal_L3N3 = MaxMinEigAlgo(cal_wave,
                                                  frameSize3, M, L3, Pfa)
savedata(ratio_L3N3, 'L3N3_ratio.txt')
savedata(gam_L3N3, 'L3N3_gamma.txt')
```

```
savedata(signal_L3N3, 'L3N3_signal.txt')
#
# Setup X-axis for detection testing
#
X_N1 = np.floor(np.linspace(0, cal_wave.size, ratio_L1N1.size))
X_N2 = np.floor(np.linspace(0, cal_wave.size, ratio_L1N2.size))
X_N3 = np.floor(np.linspace(0, cal_wave.size, ratio_L1N3.size))
#
#
#
signal_graph(cal_wave, signal_L1N1, 'L1N1_SigDet',
             'Wi-Fi 6 MCS7 Signal 1 for L = 5 and Ns = 1 000')
signal_graph(cal_wave, signal_L1N2, 'L1N2_SigDet',
             'Wi-Fi 6 MCS7 Signal 1 for L = 5 and Ns = 10 000')
signal_graph(cal_wave, signal_L1N3, 'L1N3_SigDet',
             'Wi-Fi 6 MCS7 Signal 1 for L = 5 and Ns = 100 000')
signal_graph(cal_wave, signal_L2N1, 'L2N1_SigDet',
             'Wi-Fi 6 MCS7 Signal 1 for L = 10 and Ns = 1 000')
signal_graph(cal_wave, signal_L2N2, 'L2N2_SigDet',
             'Wi-Fi 6 MCS7 Signal 1 for L = 10 and Ns = 10 000')
signal_graph(cal_wave, signal_L2N3, 'L2N3_SigDet',
             'Wi-Fi 6 MCS7 Signal 1 for L = 10 and Ns = 100 000')
signal_graph(cal_wave, signal_L3N1, 'L3N1_SigDet',
             'Wi-Fi 6 MCS7 Signal 1 for L = 15 and Ns = 1 000')
signal_graph(cal_wave, signal_L3N2, 'L3N2_SigDet',
             'Wi-Fi 6 MCS7 Signal 1 for L = 15 and Ns = 10 000')
signal_graph(cal_wave, signal_L3N3, 'L3N3_SigDet',
             'Wi-Fi 6 MCS7 Signal 1 for L = 15 and Ns = 100 000')
#
# Graph Detection - L = 5
#
det_fig1 = plt.figure(figsize=(12, 8))
det_ax1 = plt.subplot()
det_ax1.plot(cal_wave, color='b', linestyle='--',
             marker='.', markersize=10)
det_ax1.plot(X_N3, signal_L1N3*6000, color='c',
             linestyle='--', marker='.', markersize=10)
det_ax1.plot(X_N2, signal_L1N2*4000, color='g',
             linestyle='--', marker='.', markersize=10)
det_ax1.plot(X_N1, signal_L1N1*2000, color='r',
             linestyle='--', marker='.', markersize=10)
det_ax1.set_title('Plot with overlayed points of detected signal for L = 5
det_ax1.set_xlabel('Sample Number')
det_ax1.set_ylabel('Sample Value')
det_ax1.grid()
det_fig1.legend(['Original Waveform', 'Ns = 100 000',
                 'Ns = 10 000', 'Ns = 1000'], loc='upper right')
det_fig1name = 'CalSigDetFigL1.png'
det_fig1.savefig(det_fig1name)
#
# Graph Detection - L = 10
#
```

```
det_fig2 = plt.figure(figsize=(12, 8))
det_ax2 = plt.subplot()
det_ax2.plot(cal_wave, color='b', linestyle='--',
             marker='.', markersize=10)
det_ax2.plot(X_N3, signal_L2N3*6000, color='c',
             linestyle='--', marker='.', markersize=10)
det_ax2.plot(X_N2, signal_L2N2*4000, color='g',
             linestyle='--', marker='.', markersize=10)
det_ax2.plot(X_N1, signal_L2N1*2000, color='r',
             linestyle='--', marker='.', markersize=10)
det_ax2.set_title('Plot with overlayed points of detected signal for L = 1
det_ax2.set_xlabel('Sample Number')
det_ax2.set_ylabel('Sample Value')
det_ax2.grid()
det_fig2.legend(['Original Waveform','Ns = 100 000',
                 'Ns = 10 000','Ns = 1000'], loc = 'upper right')
det_fig2name = 'CalSigDetFigL2.png'
det_fig2.savefig(det_fig2name)
#
# Graph Detection - L = 15
#
det_fig3 = plt.figure(figsize=(12, 8))
det_ax3 = plt.subplot()
det_ax3.plot(cal_wave, color = 'b', linestyle='--',
             marker='.', markersize=10)
det_ax3.plot(X_N3, signal_L3N3*6000, color = 'c',
             linestyle='--', marker='.', markersize=10)
det_ax3.plot(X_N2, signal_L3N2*4000, color = 'g',
             linestyle='--', marker='.', markersize=10)
det_ax3.plot(X_N1, signal_L3N1*2000, color = 'r',
             linestyle='--', marker='.', markersize=10)
det_ax3.set_title('Plot with overlayed points of detected signal for L = 1
det_ax3.set_xlabel('Sample Number')
det_ax3.set_ylabel('Sample Value')
det_ax3.grid()
det_fig3.legend(['Original Waveform','Ns = 100 000','Ns = 10 000',
                 'Ns = 1000'], loc = 'upper right')
det_fig1name = 'CalSigDetFigL3.png'
det_fig3.savefig(det_fig1name)
#
# Compare L - Figure 1 (Ns = 1000)
#
fig1 = plt.figure(figsize=(12, 8))
ax1 = plt.subplot()
ax1.plot(ratio_L3N1, color = 'r', linestyle='--',
         marker='.', markersize=10)
ax1.plot(ratio_L2N1, color = 'g', linestyle='--',
         marker='.', markersize=10)
ax1.plot(ratio_L1N1, color = 'b', linestyle='--',
         marker='.', markersize=10)
ax1.set_title('Plot of Eigenvalue Ratios for N = 1000')
ax1.set_xlabel('Frame Number')
```

```python
    ax1.set_ylabel('Eigenvalue Ratio')
    ax1.grid()
    fig1.legend(['L = 15','L = 10','L = 5'], loc = 'upper right')
    fig1name = 'L1N1toL3N1.png'
    fig1.savefig(fig1name)
    #
    # Compare L - Figure 2 (Ns = 10 000)
    #
    fig2 = plt.figure(figsize=(12, 8))
    ax2 = plt.subplot()
    ax2.plot(ratio_L3N2, color = 'r', linestyle='--',
             marker='.', markersize=10)
    ax2.plot(ratio_L2N2, color = 'g', linestyle='--',
             marker='.', markersize=10)
    ax2.plot(ratio_L1N2, color = 'b', linestyle='--',
             marker='.', markersize=10)
    ax2.set_title('Plot of Eigenvalue Ratios for N = 10 000')
    ax2.set_xlabel('Frame Number')
    ax2.set_ylabel('Eigenvalue Ratio')
    ax2.grid()
    fig2.legend(['L = 15','L = 10','L = 5'], loc = 'upper right')
    fig2name = 'L1N2toL3N2.png'
    fig2.savefig(fig2name)
    #
    # Compare L - Figure 3 (Ns = 100 000)
    #
    fig3 = plt.figure(figsize=(12, 8))
    ax3 = plt.subplot()
    ax3.plot(ratio_L3N3, color = 'r', linestyle='--',
             marker='.', markersize=10)
    ax3.plot(ratio_L2N3, color = 'g', linestyle='--',
             marker='.', markersize=10)
    ax3.plot(ratio_L1N3, color = 'b', linestyle='--',
             marker='.', markersize=10)
    ax3.set_title('Plot of Eigenvalue Ratios for N = 100 000')
    ax3.set_xlabel('Frame Number')
    ax3.set_ylabel('Eigenvalue Ratio')
    ax3.grid()
    fig3.legend(['L = 15','L = 10','L = 5'], loc = 'upper right')
    fig3name = 'L1N3toL3N3.png'
    fig3.savefig(fig3name)

def test_custom_wave():
    """
    Code to generate Custom Wave Calibration Results.

    Returns
    -------
    None.

    """
    #
```

```
    #
    #
    M = 1
    L = 5
    Pfa = 0.1
    Ns = 100000
    #
    #
    #
    cal_wave = np.fromfile('32274CA-015-D20211123T131300M012278.data',np.int16
    custom_wave = np.append(np.zeros(100000),cal_wave[0:200000])
    custom_wave = np.append(custom_wave,np.zeros(100000))
    #
    #
    #
    graphdata(custom_wave, 'Custom Wave')
    #
    #
    #
    ratio_custom, gam_custom, signal_custom = MaxMinEigAlgo(custom_wave, Ns, M
    savedata(ratio_custom, 'custom_wave_ratio.txt')
    savedata(gam_custom, 'custom_wave_gamma.txt')
    savedata(signal_custom, 'custom_wave_signal.txt')
    #
    #
    #
    signal_graph(custom_wave, signal_custom, 'CustomWave1', 'Custom Wave 1')

    #
    #
    #
    custom_wave = cal_wave[16000000:19000000]
    #
    #
    #
    graphdata(custom_wave, 'Custom Wave 2')
    #
    #
    #
    ratio_custom, gam_custom, signal_custom = MaxMinEigAlgo(custom_wave, Ns, M
    savedata(ratio_custom, 'custom_wave_2_ratio.txt')
    savedata(gam_custom, 'custom_wave_2_gamma.txt')
    savedata(signal_custom, 'custom_wave_2_signal.txt')
    #
    #
    #
    signal_graph(custom_wave, signal_custom, 'CustomWave2', 'Custom Wave 2')

def extract_and_graph():
    """
    Code to extract and graph all datasets.
```

```
    Returns
    -------
    None.

    """
    #
    # 5G Data
    #
    print('Extracting 5G Data')
    data_1_5G = np.fromfile(
        '32274CF-dell-latitude-D20211201T191100M024240.data',np.int16)
    data_2_5G = np.fromfile(
        '32274CF-dell-latitude-D20211201T191200M056171.data',np.int16)
    data_3_5G = np.fromfile(
        '32274CF-dell-latitude-D20211201T191300M048183.data',np.int16)
    print('Graphing 5G Data')
    graphdata(data_1_5G, "Recorded 5G-NR Signal 1")
    print('Graph Complete')
    graphdata(data_2_5G, "Recorded 5G-NR Signal 2")
    print('Graph Complete')
    graphdata(data_3_5G, "Recorded 5G-NR Signal 3")
    print('Graph Complete')
    print('Deleting 5G Data')
    del data_1_5G, data_2_5G, data_3_5G
    #
    # LTE 30mbps
    #
    print('Extracting LTE 30 Data')
    data_1_LTE_30 = np.fromfile(
        '32274CF-dell-latitude-D20211125T150800M004475.data',np.int16)
    data_2_LTE_30 = np.fromfile(
        '32274CF-dell-latitude-D20211125T150900M059227.data',np.int16)
    data_3_LTE_30 = np.fromfile(
        '32274CF-dell-latitude-D20211125T151000M056507.data',np.int16)
    print('Graphing LTE 30 Data')
    graphdata(data_1_LTE_30, "Recorded LTE 30mbps Signal 1")
    print('Graph Complete')
    graphdata(data_2_LTE_30, "Recorded LTE 30mbps Signal 2")
    print('Graph Complete')
    graphdata(data_3_LTE_30, "Recorded LTE 30mbps Signal 3")
    print('Graph Complete')
    print('Deleting LTE 30 Data')
    del data_1_LTE_30, data_2_LTE_30, data_3_LTE_30
    #
    # LTE 50mbps
    #
    print('Extracting LTE 50 Data')
    data_1_LTE_50 = np.fromfile(
        '32274CF-dell-latitude-D20211125T152000M052549.data',np.int16)
    data_2_LTE_50 = np.fromfile(
        '32274CF-dell-latitude-D20211125T152100M004125.data',np.int16)
    data_3_LTE_50 = np.fromfile(
```

```
    '32274CF-dell-latitude-D20211125T152400M046037.data',np.int16)
print('Graphing LTE 50 Data')
graphdata(data_1_LTE_50, "Recorded LTE 50mbps Signal 1")
print('Graph Complete')
graphdata(data_2_LTE_50, "Recorded LTE 50mbps Signal 2")
print('Graph Complete')
graphdata(data_3_LTE_50, "Recorded LTE 50mbps Signal 3")
print('Graph Complete')
print('Deleting LTE 50 Data')
del data_1_LTE_50, data_2_LTE_50, data_3_LTE_50
#
# Wi-fi 802.11ax MCS6 30mbps
#
print('Extracting Wi-Fi MCS6 Data')
data_1_802_11ax_MCS6_30 = np.fromfile(
    '32274CA-015-D20211123T132300M011437.data',np.int16)
data_2_802_11ax_MCS6_30 = np.fromfile(
    '32274CA-015-D20211123T132400M058863.data',np.int16)
data_3_802_11ax_MCS6_30 = np.fromfile(
    '32274CA-015-D20211123T133000M028048.data',np.int16)
print('Graphing Wi-Fi MCS6 Data')
graphdata(data_1_802_11ax_MCS6_30,
          "Recorded Wi-fi 802.11ax MCS6 30mbps Signal 1")
print('Graph Complete')
graphdata(data_2_802_11ax_MCS6_30,
          "Recorded Wi-fi 802.11ax MCS6 30mbps Signal 2")
print('Graph Complete')
graphdata(data_3_802_11ax_MCS6_30,
          "Recorded Wi-fi 802.11ax MCS6 30mbps Signal 3")
print('Graph Complete')
print('Deleting Wi-Fi MCS6 Data')
del data_1_802_11ax_MCS6_30, data_2_802_11ax_MCS6_30, data_3_802_11ax_MCS6
#
# Wi-fi 802.11ax MCS7 50mbps
#
print('Extracting Wi-Fi MCS7 Data')
data_1_802_11ax_MCS7_50 = np.fromfile(
    '32274CA-015-D20211123T131300M012278.data',np.int16)
data_2_802_11ax_MCS7_50 = np.fromfile(
    '32274CA-015-D20211123T131400M050226.data',np.int16)
data_3_802_11ax_MCS7_50 = np.fromfile(
    '32274CA-015-D20211123T131500M053479.data',np.int16)
print('Graphing Wi-Fi MCS7 Data')
graphdata(data_1_802_11ax_MCS7_50,
          "Recorded Wi-fi 802.11ax MCS7 50mbps Signal 1")
print('Graph Complete')
graphdata(data_2_802_11ax_MCS7_50,
          "Recorded Wi-fi 802.11ax MCS7 50mbps Signal 2")
print('Graph Complete')
graphdata(data_3_802_11ax_MCS7_50,
          "Recorded Wi-fi 802.11ax MCS7 50mbps Signal 3")
print('Graph Complete')
```

```
    print('Deleting Wi-Fi MCS7 Data')
    del data_1_802_11ax_MCS7_50, data_2_802_11ax_MCS7_50, data_3_802_11ax_MCS7

def extract_and_test():
    """
    Code to extract all datasets and perform signal detection.

    Returns
    -------
    None.

    """
    #
    # Start of Function - Print
    #
    print('Start of Extracting and Testing')
    #
    # Setup Variables
    #
    Ns = 100000
    L = 5
    M = 1
    Pfa = 0.1
    #
    # 5G Data
    #
    print('Extracting 5G Data')
    data_1_5G = np.fromfile(
        '32274CF-dell-latitude-D20211201T191100M024240.data',np.int16)
    data_2_5G = np.fromfile(
        '32274CF-dell-latitude-D20211201T191200M056171.data',np.int16)
    data_3_5G = np.fromfile(
        '32274CF-dell-latitude-D20211201T191300M048183.data',np.int16)
    # Signal 1
    print('Testing 5G Signal 1')
    ratio_data, gam_data, signal_data = MaxMinEigAlgo(data_1_5G, Ns, M, L, Pfa
    print('Saving 5G Signal 1 Data')
    savedata(ratio_data, '5G_1_ratio.txt')
    savedata(gam_data, '5G_1_gam.txt')
    savedata(signal_data, '5G_1_signal.txt')
    print('Saved 5G Signal 1 Data')
    # Signal 1 Graphing
    signal_graph(data_1_5G, signal_data, '5G NR Signal 1', '5G NR Signal 1')
    # Signal 2
    print('Testing 5G Signal 2')
    ratio_data, gam_data, signal_data = MaxMinEigAlgo(data_2_5G, Ns, M, L, Pfa
    print('Saving 5G Signal 2 Data')
    savedata(ratio_data, '5G_2_ratio.txt')
    savedata(gam_data, '5G_2_gam.txt')
    savedata(signal_data, '5G_2_signal.txt')
    print('Saved 5G Signal 2 Data')
    # Signal 2 Graphing
```

```
signal_graph(data_2_5G, signal_data, '5G-NR-Signal-2', '5G-NR-Signal-2')
# Signal 3
print('Testing-5G-Signal-3')
ratio_data, gam_data, signal_data = MaxMinEigAlgo(data_3_5G, Ns, M, L, Pfa
print('Saving-5G-Signal-3-Data')
savedata(ratio_data, '5G_3_ratio.txt')
savedata(gam_data, '5G_3_gam.txt')
savedata(signal_data, '5G_3_signal.txt')
print('Saved-5G-Signal-3-Data')
# Signal 3 Graphing
signal_graph(data_3_5G, signal_data, '5G-NR-Signal-3', '5G-NR-Signal-3')
# Delete Data
print('Clearing-5G-Data')
del data_1_5G, data_2_5G, data_3_5G, ratio_data, gam_data, signal_data
print('Cleared-5G-Data')
#
# LTE 30mbps
#
print('Extracting-LTE-30-Data')
data_1_LTE_30 = np.fromfile(
    '32274CF-dell-latitude-D20211125T150800M004475.data',np.int16)
data_2_LTE_30 = np.fromfile(
    '32274CF-dell-latitude-D20211125T150900M059227.data',np.int16)
data_3_LTE_30 = np.fromfile(
    '32274CF-dell-latitude-D20211125T151000M056507.data',np.int16)
# Signal 1
print('Testing-LTE-30-Signal-1')
ratio_data, gam_data, signal_data = MaxMinEigAlgo(data_1_LTE_30, Ns, M, L,
print('Saving-LTE-30-Signal-1-Data')
savedata(ratio_data, 'LTE_30_1_ratio.txt')
savedata(gam_data, 'LTE_30_1_gam.txt')
savedata(signal_data, 'LTE_30_1_signal.txt')
print('Saved-LTE-30-Signal-1-Data')
# Signal 1 Graphing
signal_graph(data_1_LTE_30, signal_data, 'LTE-30-Signal-1', 'LTE-30-Signal
# Signal 2
print('Testing-LTE-30-Signal-2')
ratio_data, gam_data, signal_data = MaxMinEigAlgo(data_2_LTE_30, Ns, M, L,
print('Saving-LTE-30-Signal-2-Data')
savedata(ratio_data, 'LTE_30_2_ratio.txt')
savedata(gam_data, 'LTE_30_2_gam.txt')
savedata(signal_data, 'LTE_30_2_signal.txt')
print('Saved-LTE-30-Signal-2-Data')
# Signal 2 Graphing
signal_graph(data_2_LTE_30, signal_data, 'LTE-30-Signal-2', 'LTE-30-Signal
# Signal 3
print('Testing-LTE-30-Signal-3')
ratio_data, gam_data, signal_data = MaxMinEigAlgo(data_3_LTE_30, Ns, M, L,
print('Saving-LTE-30-Signal-3-Data')
savedata(ratio_data, 'LTE_30_3_ratio.txt')
savedata(gam_data, 'LTE_30_3_gam.txt')
savedata(signal_data, 'LTE_30_3_signal.txt')
```

```
print('Saved LTE 30 Signal 3 Data')
# Signal 3 Graphing
signal_graph(data_3_LTE_30, signal_data, 'LTE 30 Signal 3', 'LTE 30 Signal
# Delete Data
print('Clearing LTE 30 Data')
del data_1_LTE_30, data_2_LTE_30, data_3_LTE_30, ratio_data, gam_data, sig
print('Cleared LTE 30 Data')
#
# LTE 50mbps
#
print('Extracting LTE 50 Data')
data_1_LTE_50 = np.fromfile(
    '32274CF-dell-latitude-D20211125T152000M052549.data',np.int16)
data_2_LTE_50 = np.fromfile(
    '32274CF-dell-latitude-D20211125T152100M004125.data',np.int16)
data_3_LTE_50 = np.fromfile(
    '32274CF-dell-latitude-D20211125T152400M046037.data',np.int16)
# Signal 1
print('Testing LTE 50 Signal 1')
ratio_data, gam_data, signal_data = MaxMinEigAlgo(data_1_LTE_50, Ns, M, L,
print('Saving LTE 50 Signal 1 Data')
savedata(ratio_data, 'LTE_50_1_ratio.txt')
savedata(gam_data, 'LTE_50_1_gam.txt')
savedata(signal_data, 'LTE_50_1_signal.txt')
print('Saved LTE 50 Signal 1 Data')
# Signal 1 Graphing
signal_graph(data_1_LTE_50, signal_data, 'LTE 50 Signal 1', 'LTE 50 Signal
# Signal 2
print('Testing LTE 50 Signal 2')
ratio_data, gam_data, signal_data = MaxMinEigAlgo(data_2_LTE_50, Ns, M, L,
print('Saving LTE 50 Signal 2 Data')
savedata(ratio_data, 'LTE_50_2_ratio.txt')
savedata(gam_data, 'LTE_50_2_gam.txt')
savedata(signal_data, 'LTE_50_2_signal.txt')
print('Saved LTE 50 Signal 2 Data')
# Signal 2 Graphing
signal_graph(data_2_LTE_50, signal_data, 'LTE 50 Signal 2', 'LTE 50 Signal
# Signal 3
print('Testing LTE 50 Signal 3')
ratio_data, gam_data, signal_data = MaxMinEigAlgo(data_3_LTE_50, Ns, M, L,
print('Saving LTE 50 Signal 3 Data')
savedata(ratio_data, 'LTE_50_3_ratio.txt')
savedata(gam_data, 'LTE_50_3_gam.txt')
savedata(signal_data, 'LTE_50_3_signal.txt')
print('Saved LTE 50 Signal 3 Data')
# Signal 3 Graphing
signal_graph(data_3_LTE_50, signal_data, 'LTE 50 Signal 3', 'LTE 50 Signal
# Delete Data
print('Clearing LTE 50 Data')
del data_1_LTE_50, data_2_LTE_50, data_3_LTE_50, ratio_data, gam_data, sig
print('Cleared LTE 50 Data')
#
```

```
# Wi-fi 802.11ax MCS6 30mbps
#
print('Extracting Wi-Fi MCS6 Data')
data_1_802_11ax_MCS6_30 = np.fromfile(
    '32274CA-015-D20211123T132300M011437.data',np.int16)
data_2_802_11ax_MCS6_30 = np.fromfile(
    '32274CA-015-D20211123T132400M058863.data',np.int16)
data_3_802_11ax_MCS6_30 = np.fromfile(
    '32274CA-015-D20211123T133000M028048.data',np.int16)
# Signal 1
print('Testing Wi-Fi MCS6 Signal 1')
ratio_data, gam_data, signal_data = MaxMinEigAlgo(data_1_802_11ax_MCS6_30,
                                                  Ns, M, L, Pfa)
print('Saving Wi-Fi MCS6 Signal 1 Data')
savedata(ratio_data, 'WiFi_MCS6_1_ratio.txt')
savedata(gam_data, 'WiFi_MCS6_1_gam.txt')
savedata(signal_data, 'WiFi_MCS6_1_signal.txt')
print('Saved Wi-Fi MCS6 Signal 1 Data')
# Signal 1 Graphing
signal_graph(data_1_802_11ax_MCS6_30, signal_data, 'Wi-Fi MCS6 Signal 1',
             'Wi-Fi MCS6 Signal 1')
# Signal 2
print('Testing Wi-Fi MCS6 Signal 2')
ratio_data, gam_data, signal_data = MaxMinEigAlgo(data_2_802_11ax_MCS6_30,
                                                  Ns, M, L, Pfa)
print('Saving Wi-Fi MCS6 Signal 2 Data')
savedata(ratio_data, 'WiFi_MCS6_2_ratio.txt')
savedata(gam_data, 'WiFi_MCS6_2_gam.txt')
savedata(signal_data, 'WiFi_MCS6_2_signal.txt')
print('Saved Wi-Fi MCS6 Signal 2 Data')
# Signal 2 Graphing
signal_graph(data_2_802_11ax_MCS6_30, signal_data, 'Wi-Fi MCS6 Signal 2',
             'Wi-Fi MCS6 Signal 2')
# Signal 3
print('Testing Wi-Fi MCS6 Signal 3')
ratio_data, gam_data, signal_data = MaxMinEigAlgo(data_3_802_11ax_MCS6_30,
                                                  Ns, M, L, Pfa)
print('Saving Wi-Fi MCS6 Signal 3 Data')
savedata(ratio_data, 'WiFi_MCS6_3_ratio.txt')
savedata(gam_data, 'WiFi_MCS6_3_gam.txt')
savedata(signal_data, 'WiFi_MCS6_3_signal.txt')
print('Saved Wi-Fi MCS6 Signal 3 Data')
# Signal 3 Graphing
signal_graph(data_3_802_11ax_MCS6_30, signal_data, 'Wi-Fi MCS6 Signal 3',
             'Wi-Fi MCS6 Signal 3')
# Delete Data
print('Clearing Wi-Fi MCS6 Data')
del data_1_802_11ax_MCS6_30, data_2_802_11ax_MCS6_30
del data_3_802_11ax_MCS6_30, ratio_data, gam_data, signal_data
print('Cleared Wi-Fi MCS6 Data')
#
# Wi-fi 802.11ax MCS7 50mbps
```

```
#
print('Extracting Wi-Fi MCS7 Data')
data_1_802_11ax_MCS7_50 = np.fromfile(
    '32274CA-015-D20211123T131300M012278.data',np.int16)
data_2_802_11ax_MCS7_50 = np.fromfile(
    '32274CA-015-D20211123T131400M050226.data',np.int16)
data_3_802_11ax_MCS7_50 = np.fromfile(
    '32274CA-015-D20211123T131500M053479.data',np.int16)
# Signal 1
print('Testing Wi-Fi MCS7 Signal 1')
ratio_data, gam_data, signal_data = MaxMinEigAlgo(data_1_802_11ax_MCS7_50,
                                                  Ns, M, L, Pfa)
print('Saving Wi-Fi MCS7 Signal 1 Data')
savedata(ratio_data, 'WiFi_MCS7_1_ratio.txt')
savedata(gam_data, 'WiFi_MCS7_1_gam.txt')
savedata(signal_data, 'WiFi_MCS7_1_signal.txt')
print('Saved Wi-Fi MCS7 Signal 1 Data')
# Signal 1 Graphing
signal_graph(data_1_802_11ax_MCS7_50, signal_data, 'Wi-Fi MCS7 Signal 1',
             'Wi-Fi MCS7 Signal 1')
# Signal 2
print('Testing Wi-Fi MCS7 Signal 2')
ratio_data, gam_data, signal_data = MaxMinEigAlgo(data_2_802_11ax_MCS7_50,
                                                  Ns, M, L, Pfa)
print('Saving Wi-Fi MCS7 Signal 2 Data')
savedata(ratio_data, 'WiFi_MCS7_2_ratio.txt')
savedata(gam_data, 'WiFi_MCS7_2_gam.txt')
savedata(signal_data, 'WiFi_MCS7_2_signal.txt')
print('Saved Wi-Fi MCS7 Signal 2 Data')
# Signal 2 Graphing
signal_graph(data_2_802_11ax_MCS7_50, signal_data, 'Wi-Fi MCS7 Signal 2',
             'Wi-Fi MCS7 Signal 2')
# Signal 3
print('Testing Wi-Fi MCS7 Signal 3')
ratio_data, gam_data, signal_data = MaxMinEigAlgo(data_3_802_11ax_MCS7_50,
                                                  Ns, M, L, Pfa)
print('Saving Wi-Fi MCS7 Signal 3 Data')
savedata(ratio_data, 'WiFi_MCS7_3_ratio.txt')
savedata(gam_data, 'WiFi_MCS7_3_gam.txt')
savedata(signal_data, 'WiFi_MCS7_3_signal.txt')
print('Saved Wi-Fi MCS7 Signal 3 Data')
# Signal 3 Graphing
signal_graph(data_3_802_11ax_MCS7_50, signal_data, 'Wi-Fi MCS7 Signal 3',
             'Wi-Fi MCS7 Signal 3')
# Delete Data
print('Clearing Wi-Fi MCS7 Data')
del data_1_802_11ax_MCS7_50, data_2_802_11ax_MCS7_50
del data_3_802_11ax_MCS7_50, ratio_data, gam_data, signal_data
print('Cleared Wi-Fi MCS7 Data')
#
# End of Function - Print
#
```

```
    print('End of Extracting and Testing')


if __name__ == '__main__':
    main()
```