2010

Differentiating Ethernet devices using Normal Link Pulse with efficient computation and the impacts on performance

Wade David Paustian
Iowa State University

Differentiating Ethernet devices using the Normal Link Pulse with efficient computation and the impacts on performance

by

Wade David Paustian

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Co-majors: Computer Engineering, Information Assurance

Program of Study Committee:
Thomas E. Daniels, Major Professor
Mani Mina
Joseph Zambreno

Iowa State University

Ames, Iowa

2010

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

The tolerances in manufacturing Ethernet devices cause detectable differences in the signals sent by two different devices. Here, the design space is examined for using the IEEE 802.3 Normal Link Pulse (NLP) as the signal to use for differentiating devices. A previously collected set of NLP records as well as new sets of NLP data are used for testing the detection algorithm. Further tests have been run to determine the possibility of reducing the sampling rate to the point where Analogue-to-Digital Converters (ADCs) are more readily available and inexpensive. Reduced precision at each decimation was also tested. The design space survey indicates that trimming the time domain NLP records is beneficial to a certain point, and tracking the changes or drift of the signal has a great benefit. The design space survey also showed both wavelet-based filtering and noise spectra density scaling are beneficial on their own, but noise spectra density scaling can impair our algorithm when wavelet filtering is also being used. The tests on reducing sample rate and precision of the collected NLP records yielded results showing that sample rate effected false negative (device falsely unauthenticated) rates noticeably at decimation factors 8 and 16. Furthermore, false positive (devise falsely authenticated) rates were mostly effected by reduced precision. It is also apparent that performance of the algorithm, as determined by the impostor minimum to authentic maximum power mean squared error ratio, decreases with increasing data decimation before there is an increase in false negatives.

CHAPTER 1. Overview and Introduction

The network physical layer lacks attention when it comes to intrusion detection. Above the physical layer, all the hardware required for intrusion detection exists, and only code is required for an Intrusion Detection System (IDS) to be made. This also allows the higher layer IDSs to be modified and quickly redistributed when new detection algorithms are invented. There has been work related to intrusion detection at the physical layer; however, the work done requires extra hardware in the detection device. Which means updating the detection algorithms is unlikely unless it can be done in firmware. This is a contributer to the lack of implementations seen.

1.1 Goals of Research

The research for use of the Normal Link Pulse (NLP) as a method for device authentication, Erbskorn, J. W. (2009), deserves a closer look. This should include expanding the simulation to run on data collected from new sets of computers. As well as examining the design space of the research. And exploring possibilities for making the design cost efficient. Before discussing these goals, an explanation of the NLP is given.

1.1.1 Normal Link Pulse

The Normal Link Pulse was referred to as the Link Integrity Test (LIT) pulse in 10BASE-T terminology. However, since the introduction of Auto-Negotiation the term Normal Link Pulse is more commonly used. Auto-Negotiation, which negotiates a connection up to higher bit rates, will also substitute the Fast Link Pulse (FLP) Burst in place of a LIT pulse. Currently, in order to collect NLPs, the transmitting device is configured or negotiated to use 10BASE-T only, so that it does not attempt to send FLP Bursts. However, the NLP is present at higher bit rates.

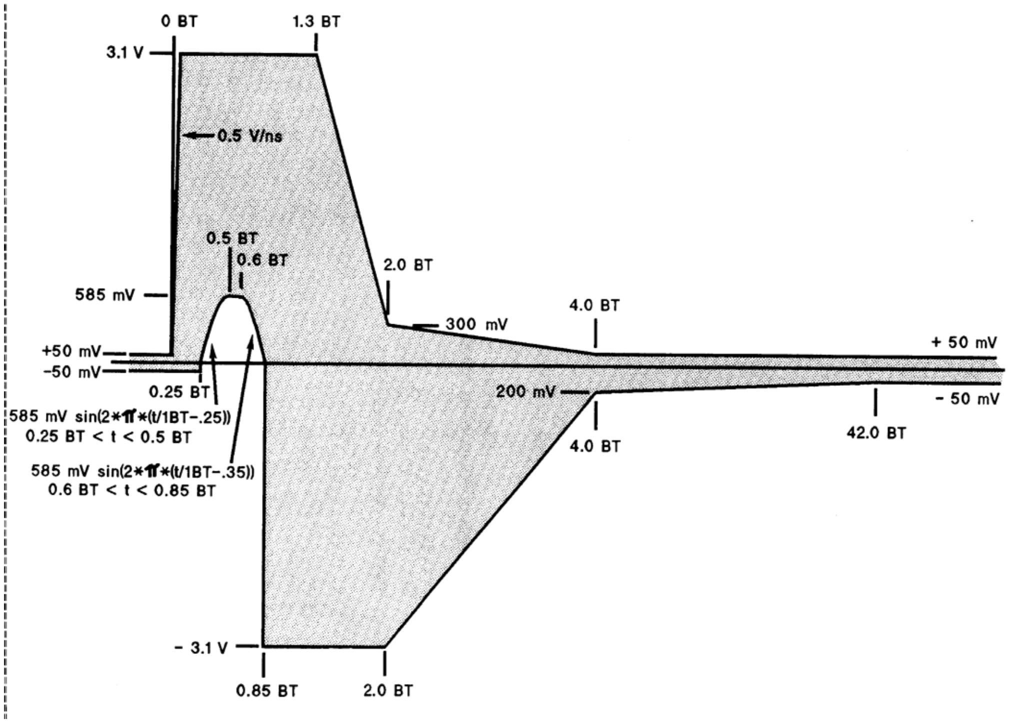


Figure 1.1: IEEE 802.3 10Base-T NLP [IEEE Standard 802.3 (2005)]

Figure 1.1 depicts the standard for an NLP. It is explained in IEEE Standard 802.3 (2005) that one Bit Time (BT) corresponds to $1 \times 10^{-7} seconds$ or 100 nanoseconds. The transmitter of an Ethernet device starts sending NLPs as soon as it is powered on, and an NLP is sent every 16 ms ± 8 ms while the data transmitter is idle. So, 41 to 125 NLP are transmitted each second. However, using the DILON mobile test system [Erbskorn, J. W. (2009)], collecting 100 NLP requires approximately 30 seconds. The NLPs are sent before any data is sent, so the authentication of a device may take place before allowing it to send data.

1.1.2 Expand Data Collected

Erbskorn, J. W. (2009) contains some previous work in the Detecting Intrusions at Layer ONe (DILON) project which involved running data collection on Dell Optiplex G1XP computers. And, one question brought up in previous DILON work has been whether the manufacturing processes for newer Ethernet cards may be more precise leading to less distinguishable differences present in the signals being sent.

To answer this question and increase the diversity of computers data has been collected from, the goal of collecting data on newer computers has been formed.

1.1.3 Examine Design Space

The design space as things were posed in Erbskorn, J. W. (2009) allows for choices in how the Normal Link Pulse (NLP) records are aligned, the method for minimizing the effect of noise on signal matching. By looking to Xiao, L. et al. (2008a), the possibility of updating the fingerprint as test records are authenticated also becomes part of the design space.

The goal of examining the design space with new methods to accomplish the same objectives is so find the optimal set of choices for the set of computers used in previous research, and test the hypothesis that the optimal set of choices for that set will also be optimal for the new sets of computers mentioned in the previous subsection.

1.1.4 Explore Efficient Design Possibilities

The options available in the design space can be examined by how cost efficient they are likely to be for implementation. Along with the cost to implement the algorithms, there is the cost of the Analogue to Digital Converter (ADC) to consider. If the sampling rate can be successfully lowered to 125-million samples per second, then the design could utilize the two 14-bit ADCs present on the Altera Cyclone II DSP Kit (DK-DSP-2C70N) which cost $56 in March of 2010. Making it inexpensive enough that hobbyists could purchase it. However, as of June 2010, the Altera Cyclone II DSP Kit has been moved to Altera's "Obsolete/Discontinued Part Numbers" category and the price is no longer listed.

Knowing one specific sampling rate the algorithm will still function at may be beneficial, but not as potentially helpful as exploring several sampling rates and bit precisions. Knowing the results at several sampling rates and bits of precision will allow selection of the ADC to take place based on performance, rather than the reverse.

1.2 General Algorithm

Before going on to explain the methods tested for signal alignment, noise effect minimization, and signal comparison, it would be beneficial to explain the general algorithm that these tools are incorporated into. The generic algorithm is depicted in figure 1.2 through a dataflow diagram.

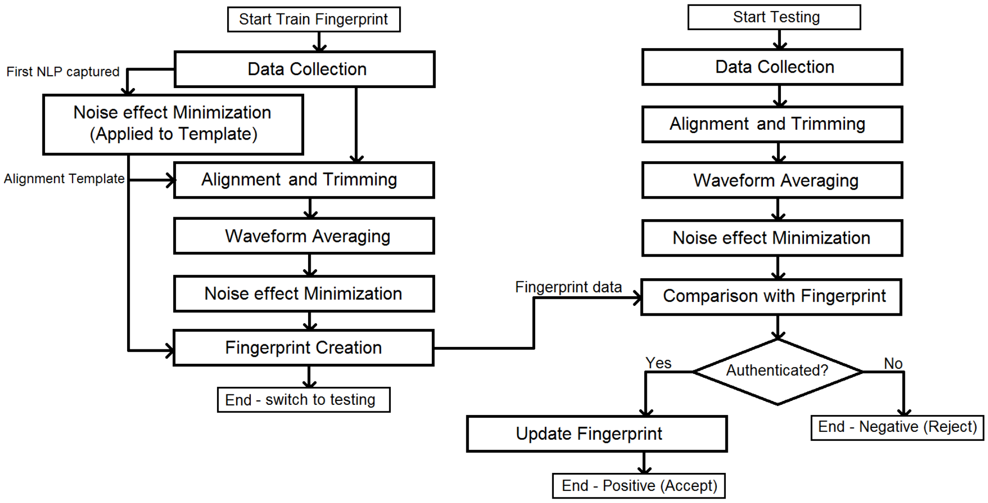


Figure 1.2: Dataflow diagram of fingerprint creation and matching algorithms.

The dataflow diagram above depicts on the left the process for creating the fingerprint of our device that we can later compare to. This process is the analog to the training process of a fingerprint scanner for biometric authentication to something such as a laptop. And on the right is the process used for comparing to the trained fingerprint. Which is the analog to the authentication process of a fingerprint scanner.

Fingerprint scanners used for biometric authentication have varied methods of performing the match detection. Similarly, there are many methods possible to perform detection of matching Ethernet devices. And in both cases, some methods perform better than others. Following chapters will discuss the changes made to the algorithm given for Ethernet device detection. The result of our algorithm is for an Ethernet device connecting to a switch to be authenticated much the same way as a person identifying themselves to a laptop using a fingerprint scanner.

In the following explanation of the algorithm, the optional data manipulation techniques are explained in chapter 3, and reasons for the specific number of NLP waveforms averaged together and other choices are explained as well as the specific algorithm in section 3.8.

The algorithm that trains the fingerprint collects a total of 500 NLP waveforms. The first NLP waveform is filtered using wavelet filtering, a low-pass frequency response filter, or left unfiltered then used as a template for alignment. Subsequent NLP waveforms are aligned to the template using cross-correlation to determine maximum alignment. Then the high-resolution alignment algorithm is, optionally, applied.

After alignment, the waveforms are averaged in groups of 100 NLP waveforms resulting in 5 waveforms. These 5 waveforms have their noise effects minimized using wavelet filtering, a low-pass frequency response filter, noise spectra density scaling, or left unfiltered. The same choice is later used in the noise effect minimization of NLPs to be tested. The fingerprint is created by averaging the 5 filtered waveforms and determining a threshold for accepting NLP records based on the differences between the 5 filtered waveforms.

The algorithm that tests for devices that match a fingerprint uses the same template waveform that was used in creating the fingerprint. This template is used again for alignment. The alignment is performed same as in fingerprint creation. Waveform averaging is then performed on groups of 100 NLP waveforms. These averages will be referred to as test records. The noise effect minimization is performed the same way it was in the fingerprint creation algorithm.

The comparison of the test record with the fingerprint is performed using the mean squared error of the power in the frequency domain. The result is determined to be authentic or an impostor device depending on if it fell within the threshold determined during fingerprint creation. If the test record is determined to be authentic and the signal is being tracked, the fingerprint is updated by replacing the oldest test record with this new test record. Test record also refers to the 5 initial averaged and filtered NLP records that compose the fingerprint.

CHAPTER 2. Review of Literature

There has been previous work related to this subject in the DILON project at Iowa State University, and at the Rutgers University Wireless Information Network Laboratory (WINLAB). Previous DILON work has focused on authenticating devices based on IEEE 802.3 signals. The WINLAB work has focused on authenticating IEEE 802.11 wireless devices. McGill and Dorfman performed work on the subject of achieving high resolution alignment and comparison of waveforms without greatly oversampling the signal. Which is related to our goals. And there has been work done before the DILON project in identifying radar, radios, and various wireless communications.

2.1 Previous DILON Work

Previous work done in DILON project has covered the possibilities of fingerprinting the synchronization signal of the Ethernet frame. There has also been work done on the suitability of the NLP which is sent as long as the device is operational. Both signals are IEEE 802.3 10BASE-T signals. There has also been some work in the possibility of fingerprinting IEEE 802.11 wireless signals which mostly focused on transient analysis.

Jackson, E. A. (2006) and Gerdes, R. M. (2006) took different approaches to using the synchronization signal for device authentication. In Jackson, E. A. (2006), a survey of fingerprinting methods were tested. These methods were the differential Fourier voiceprint, multifractal dimensional analysis, principle and independent component analysis and the Kolmogorov-Smirnov test. The results of the surveyed techniques were compared to the results of matched filtering.

In Gerdes, R. M. (2006), the matched filter was explored more deeply as an option applying variations in order to improve the capacity for the method to produce its desired effect. First, the signal is broken into sections. The transient, steady-state and source Medium Access Control (MAC) address sections are then handled separately, preventing effects in one section of the signal from overshadowing the characteristics of the other sections. Band pass filtering, normalization of the signal and time domain amplitude trimming are also applied.

In Erbskorn, J. W. (2009), the IEEE 802.3 Normal Link Pulse was examined as an additional signal that could be fingerprinted for device authentication. The problem was broken into distinguishing between different models of devices, and then discriminating between devices of the same model with more focus on the latter. For classification of devices that are the same model, sets of NLP records were aligned through cross-correlation and averaged. Then, had the details containing the most noise removed through using Daubechies D4 Discrete Wavelet Transform (DWT). And the Mean Squared Error (MSE) was used as a metric of similarity between fingerprint and test record.

2.2 WINLAB at Rutgers University Research

The research done by Xiao, Greenstein, Mandayam, and Trappe in the WINLAB has been presented in four papers. Xiao, L. et al. (2007) discusses scattering environment of the radio channel to determine whether current and prior communications originate from the same source. To evaluate feasibility, simulations were run with spatially variant channel responses in real environments utilizing the WiSE ray-tracing tool.

In Xiao, L. et al. (2008a) the work from Xiao, L. et al. (2007) is expanded to enhance authentication for mobile terminals. The work focuses on authenticating frames in a multiple frame burst. The authentication of the frames following the first in the burst is performed using the Neyman-Pearson hypothesis test or a least-squares adaptive channel estimator. Their simulation found the Neyman-Pearson test to be more robust against terminal mobility.

In Xiao, L. et al. (2008b) multiple-input multiple-output (MIMO) techniques are used to assist the channel-based wireless network authentication described in their previous publications. The same WiSE ray tracing tool is used to simulate a specific indoor environment, and the extra dimensions of channel estimation data from the use of multiple antennas of the MIMO system are employed to provide a security gain over single input single output (SISO) systems. Xiao, L. et al. (2008c) includes more information on the same things covered in their other publications with a greater pool of references.

2.3 High Resolution Alignment

The work by McGill and Dorfman is especially important for the idea of lowering the sampling rate so that the DILON project may be implemented on more readily available and inexpensive hardware. McGill, K. C., and Dorfman, L. J. (1984) shows how high resolution can be achieved efficiently from data sampled at the Nyquist rate. They present practical algorithms for aligning and comparing waveforms, locating peaks, resolving superimpositions, and averaging overlapping waveforms. They discuss how the algorithms they present are more accurate and efficient than techniques which employ continuous oversampling for many signal processing applications.

2.4 More Identification of Transmitters

The work in identification and classification of transmitters generally uses one of two sources of information to make the identification. Either digital differences caused by physical component differences, or differences in the analog characteristics of the signal which must be measured directly. In the first category, Kohno, T. et al. (2005) investigates the possibility to remotely fingerprint devices over the Internet by measuring clock skew through TCP timestamps option.

The second category, identifying transmitters by the analog characteristics of the signal, can be further divided into using the transient portion of the signal or the steady state portion, and whether the time, frequency, or wavelet domain is used for the analysis. Transient analysis has been most popular in the past; most early methods for radar identification used transient analysis. However, more in-depth transient analysis became required as higher frequency and faster responding circuits were introduced. The capabilities of the circuits have reached the point where transient analysis is used to identify transmitters of different makes rather than transmitters of the same make. An example of this is Ellis, K. J. and Serinken, N. (2001).

In Payal, Y. (1995) 4 push-to-talk devices, two of which were the same make, were identified using the Wavelet Transform through using the differences in the local extrema of the wavelet coefficients. In Barrere, W. G. et al. (1998), a differencing technique is used to identify cell phones where the differences between two signals are raised to an arbitrary power then summed to form the fingerprint.

CHAPTER 3. Design Reasoning

Specific reasons have gone into the design choices of the algorithm described in section 1.2. There are also reasons behind each technique being tested as part of the design space. Some of the techniques come from related works, and are expected to improve performance based on the outcomes in the related work. Other techniques may be expected to improve efficiency of implementation while still approaching the same performance. In the following sections each technique is introduced, an explanation for what the technique accomplishes, and an explaination for why it is expected to improve our results is given.

3.1 Daubechie's D4 Wavelet Transform

In our algorithm, the Daubechies D4 Wavelet Transform is used as a low-pass filter by applying the 2 or 4 level filter bank and reconstructing the approximation signal without the details. Because of how wavelet approximation works, the peaks in the signal and steep slopes are mostly uneffected. While quick oscillations around a local mean with a period of less than 4 or 16 are attenuated for the 2 and 4th level filter banks, respectively.

The Daubechies D4 Wavelet Transform belongs to a family of orthogonal discrete wavelet transforms. It has two vanishing moments relating to its ability to encode constant and linear signal components. Equations 3.1 and 3.2 [Strang, G. (2008)] show the Daubechies D4 Wavelet Transform scaling and wavelet functions, respectively.

$$s_i = \frac{1+\sqrt{3}}{4\sqrt{2}}x_{2i} + \frac{3+\sqrt{3}}{4\sqrt{2}}x_{2i+1} + \frac{3-\sqrt{3}}{4\sqrt{2}}x_{2i+2} + \frac{1-\sqrt{3}}{4\sqrt{2}}x_{2i+3} \tag{3.1}$$

$$d_i = \frac{1-\sqrt{3}}{4\sqrt{2}}x_{2i} - \frac{3-\sqrt{3}}{4\sqrt{2}}x_{2i+1} + \frac{3+\sqrt{3}}{4\sqrt{2}}x_{2i+2} - \frac{1+\sqrt{3}}{4\sqrt{2}}x_{2i+3} \tag{3.2}$$

The scaling function contains the lower frequencies of the signal while the wavelet function contains the higher frequencies. Perfect reconstruction of the signal can be obtained using the Inverse Wavelet Transform equations 3.3 and 3.4.

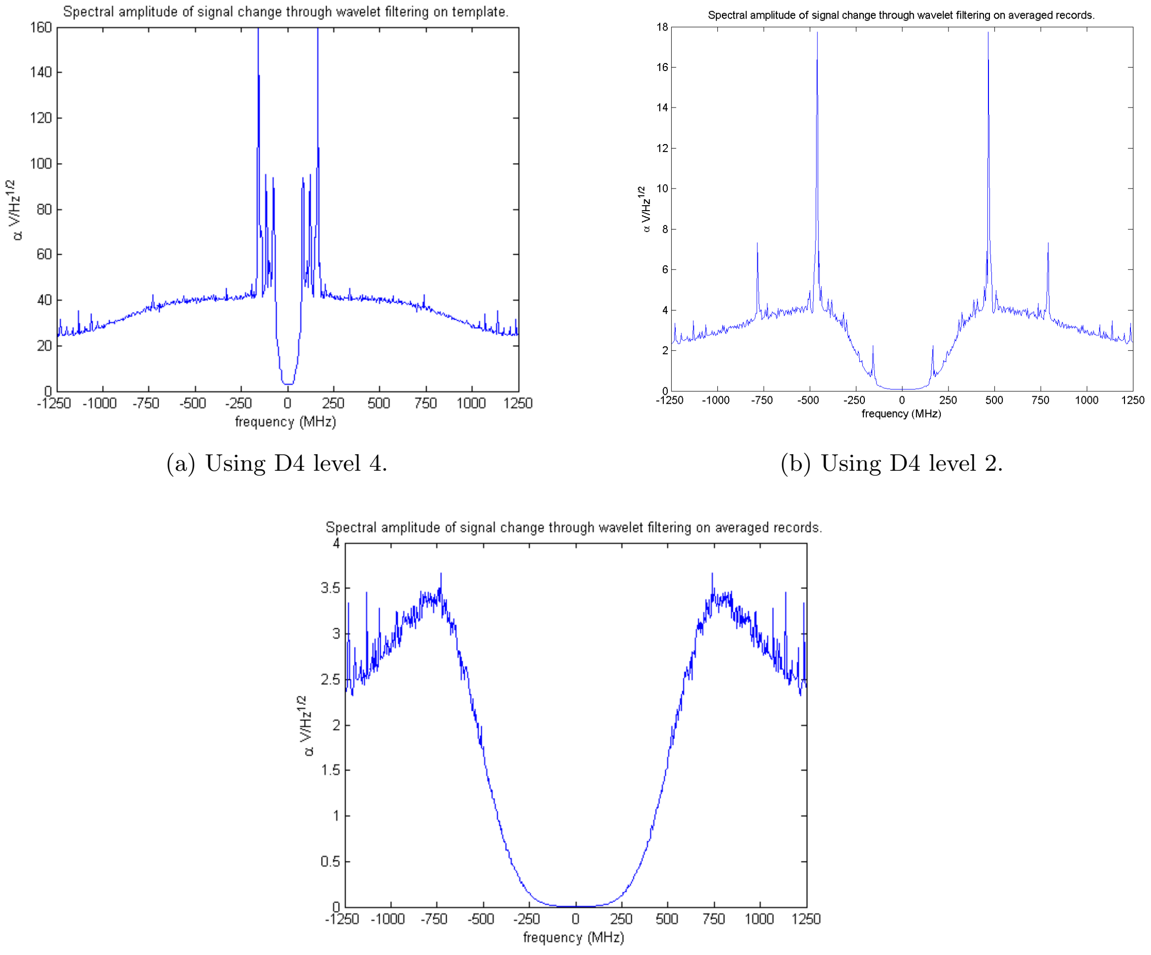
$$x_{2i} = \frac{3-\sqrt{3}}{4\sqrt{2}} s_{i-1} + \frac{1+\sqrt{3}}{4\sqrt{2}} s_i + \frac{3+\sqrt{3}}{4\sqrt{2}} d_{i-1} + \frac{1-\sqrt{3}}{4\sqrt{2}} d_i \tag{3.3}$$

$$x_{2i+1} = \frac{1-\sqrt{3}}{4\sqrt{2}} s_{i-1} + \frac{3+\sqrt{3}}{4\sqrt{2}} s_i - \frac{1+\sqrt{3}}{4\sqrt{2}} d_{i-1} - \frac{3-\sqrt{3}}{4\sqrt{2}} d_i \tag{3.4}$$

Erbskorn, J. W. (2009) began the use of Daubechies D4 Wavelet Transform in the DILON project because the 802.3 NLP being characterized is a pulse rather than a repeating wave. The reasoning was that the wavelets which are more pulse-like themselves would have a better chance of capturing the structure of the NLP than FFT techniques. Erbskorn, J. W. (2009) made use of wavelets to remove noise from the template NLP used for signal alignment and to remove noise from the test records before comparing them to the trained fingerprint record. Graps, A. L. (1995) describes previous use of wavelets for noise removal purposes.

The Daubechie's D4 Wavelet Transform is specifically used in the algorithm by decomposing the template record with a 4 level filter bank, and reconstructing from only the approximation signal (s). It is also used for removing noise from the averaged records by decomposing the record with a 2 level filter bank, and reconstructing from only the approximation signal. Figure 3.1 shows the spectral amplitude of the signal change made by the wavelet filtering.

These figures can be compared with figure 3.2 in upcoming section 3.3. Comparing figure 3.1a to figure 3.2b makes apparent that the spectral amplitude of the signal change created by the wavelet filtering on the template is a decent approximation of the noise spectral amplitude. This indicates that using the approximation signal from the 4th level wavelet decomposition is a good choice. And comparing figure 3.1b to figure 3.2b shows that the spectral amplitude of the noise theoretically remaining in the average is less than the signal change created by the wavelet filtering for using the approximation signal from the 2nd level wavelet decomposition. However, figure 3.1c shows that trying to use the 1st level does not match the shape of the noise spectral amplitude well indicating there is not a better level than the 2nd while using the Daubechie's D4 Wavelet.



(a) Using D4 level 4.

(b) Using D4 level 2.

(c) Using D4 level 1.

Figure 3.1: Spectral amplitude of the signal change made by wavelet filtering.

3.2 Low-Pass Filtering

The technique described by Graps, A. L. (1995) and the technique used by Erbskorn, J. W. (2009) remove high frequency components of a signal by substituting zero for values of d that were calculated using the wavelet function 3.2 in order to remove noise. It was hypothesized that a low-pass frequency response filter may provide similar results.

To perform the low-pass filtering, the FFT is taken of the signal, then above a certain frequency, the values of the FFT are set to zero, and the IFFT is taken to get the filtered signal. Mathematically, it is the same as applying a frequency response defined in equation

3.5, defined on the interval $[-\pi, \pi]$, to the FFT of the signal.

$$H(e^{j\omega}) = \begin{cases} 1 & : -\phi < \omega < \phi \\ 0 & : \omega < -\phi \ or \ \omega > \phi \end{cases} \tag{3.5}$$

During the study of the design space, several choices of ϕ were going to be tested. However, due to the already large number of possible configurations, the conservative choice of $\phi = \frac{\pi}{2}$ for smoothing the template, and $\phi = \frac{2\pi}{3}$ was chosen for filtering the averaged records. Viewing the averaged signal spectral amplitude presented ahead in figure 3.2a shows that at frequencies below 175 MHz, or $\phi = \frac{42\pi}{300}$, the spectral amplitude steeply increase from the frequencies above 175 MHz.

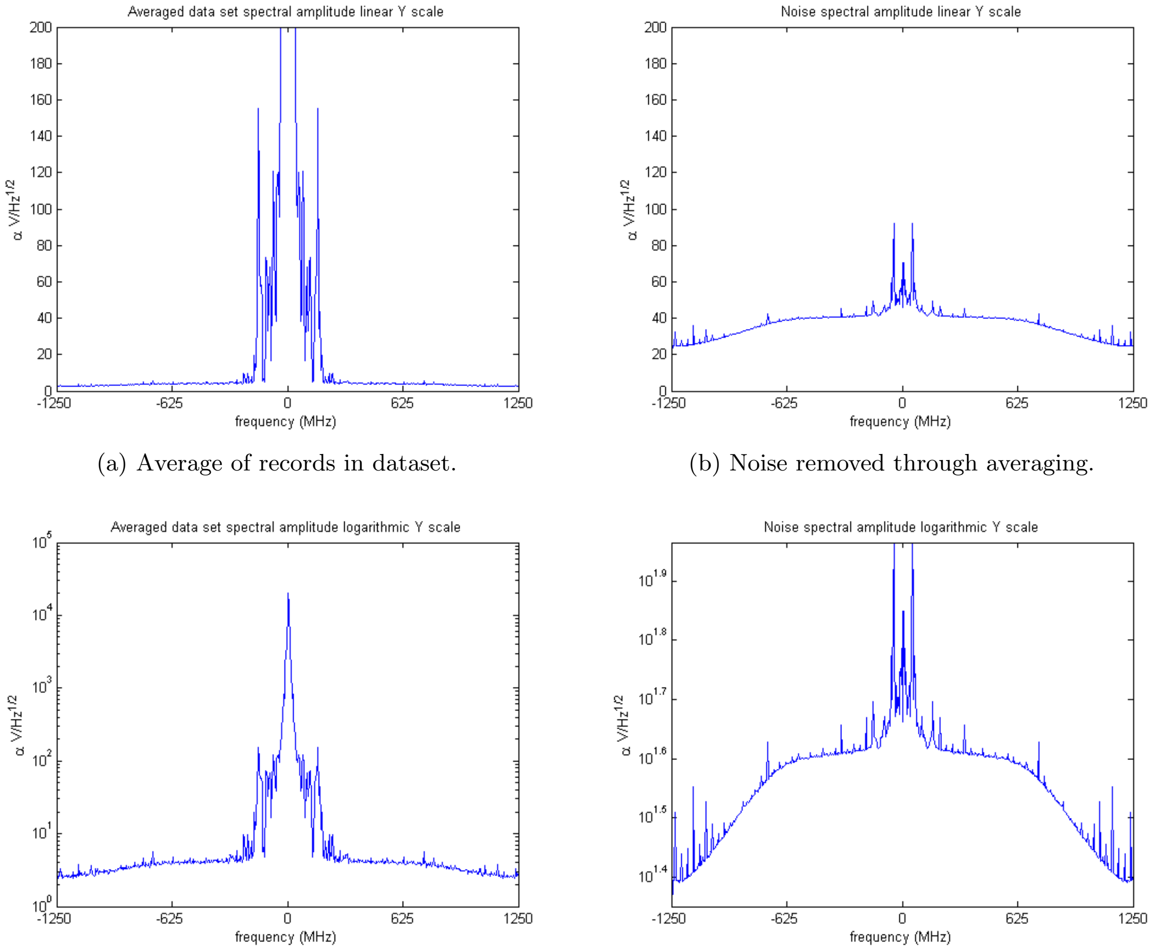
3.3 Scaling by Noise Spectral Amplitude

Using a low-pass filter to remove noise will remove parts of the signal as well as parts of the noise without any regard for how much noise underlying signal was represented by those frequencies. Which is what makes the choice of ϕ important. Taking a look at figure 3.2 we see that the frequency components that add the least to the signal also add the least noise. While the graphs in figures 3.2c and 3.2d allow us to see the full shapes of the spectral amplitudes, figures 3.2a and 3.2b allow direct comparison between the levels of signal and noise. It should be noted that figures 3.2b and 3.2d show the amount of noise present in one NLP record. The noise present after averaging should be approximately one hundredth the value shown in noise spectral amplitude figures. The data set used in the figure is Spring 2009, 3 hour collection, computer set 1 computer 1.

The reasoning behind dividing the spectral amplitude of an averaged signal by the spectral amplitude of the noise is to reduce the reliance on matching components of the signals that vary the greatest. This is shown in equation 3.6.

$$Y(e^{j\omega}) = \frac{\bar{X}(e^{j\omega})}{N(e^{j\omega})} \tag{3.6}$$

Where Y is our noise spectra scaled signal, $\bar{X}$ is the spectral amplitude of averaging a



(a) Average of records in dataset.

(b) Noise removed through averaging.

(c) Average of records in dataset.

(d) Noise removed through averaging.

Figure 3.2: Spectral amplitude of NLP record and noise removed through averaging.

number of original signal records divided element wise by N which is the spectral amplitude of the noise removed through averaging.

This technique provides an amount of flexibility by creating the noise spectral amplitude data to scale by while the fingerprint is being created, rather than using a universal threshold to cut the signal off at. This is limited with the idea that the noise environment will not change greatly from the time the device was fingerprinted. Consideration was given to using the noise spectral amplitude from the same time as the data to be tested was used; however, it was dismissed because of the possibility of a noise source being used to degrade performance of the algorithm.

3.4 Waveform Length Adjustment

Two attempts at adjusting the length of the time domain waveform have been made. Previous work by Erbskorn, J. W. (2009) trimmed the length of the waveform to a specific length based on the model of the device being fingerprinted. Termed here as model specified length adjustment.

3.4.1 Standard Length Adjustment

The first attempt at adjusting the length of the time domain waveform was to create a one standard length that all IEEE 802.3 device models could be fingerprinted with. This was done by looking at the standard for the NLP depicted in figure 1.1.

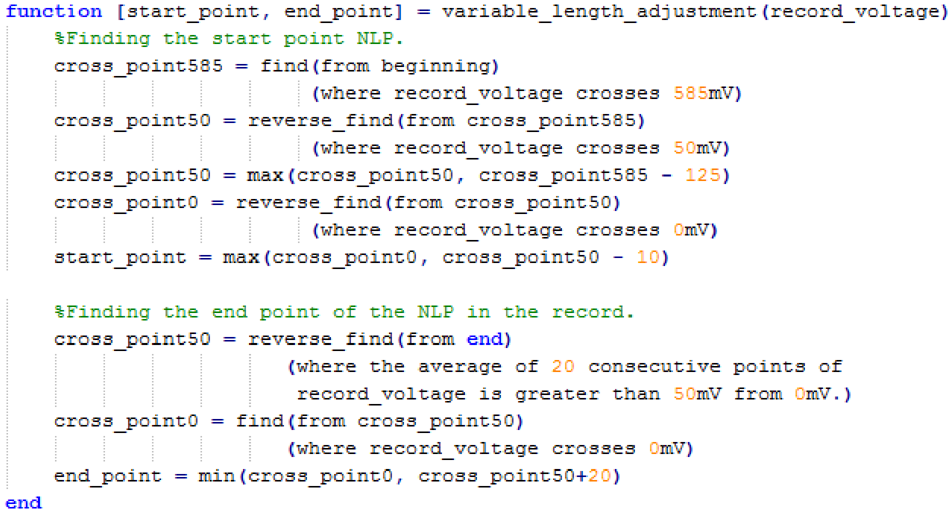
It is earlier explained in IEEE Standard 802.3 (2005) that one Bit Time (BT) corresponds to $1 \times 10^{-7} seconds$ or 100 nanoseconds. The majority of the variability allowed by the standard takes place between 0 BT and 4 BT. So to capture the full possible variety of NLP signals at our sampling rate of $2.5 \times 10^{9} samplepoints/second$ we need our waveform to be at least 1000 sample points long.

The simplistic algorithm used to find the section of interest among the extra sample points to be removed located the NLP by detecting the point where the record crosses 585mV. This can happen at any point between nearly 0 BT and 0.5 BT requiring that I add another 125 sample points from in front of where the record crosses 585mV. Totaling to at least 1125 sample points needed to be sure that any device may be fingerprinted with its full NLP represented.

3.4.2 Variable Length Adjustment

The second attempt at adjusting the length of the time domain waveform was to use a dynamic length for the record that removes as much as possible of the signal that is within noise levels.

Each point in the record is subject to noise. This translates to an increase in noise directly proportional to the length of the record. The dynamic determination of length allows any device to be fingerprinted without needing to select from a set of predetermined lengths while still removing as much noise as possible. The algorithm used to determine what part of the record to keep is depicted here in figure 3.3.



```
function [start_point, end_point] = variable_length_adjustment(record_voltage)
    %Finding the start point NLP.
    cross_point585 = find(from beginning)
                          (where record_voltage crosses 585mV)
    cross_point50 = reverse_find(from cross_point585)
                          (where record_voltage crosses 50mV)
    cross_point50 = max(cross_point50, cross_point585 - 125)
    cross_point0 = reverse_find(from cross_point50)
                          (where record_voltage crosses 0mV)
    start_point = max(cross_point0, cross_point50 - 10)

    %Finding the end point of the NLP in the record.
    cross_point50 = reverse_find(from end)
                        (where the average of 20 consecutive points of
                         record_voltage is greater than 50mV from 0mV.)
    cross_point0 = find(from cross_point50)
                        (where record_voltage crosses 0mV)
    end_point = min(cross_point0, cross_point50+20)
end
```

Figure 3.3: Algorithm for trimming the time of the actual NLP contained in the record

3.5 High Resolution Alignment

Previous works on the DILON project Gerdes, R. M. (2006), Jackson, E. A. (2006), and Erbskorn, J. W. (2009) have used cross-correlation for signal alignment. This returns the optimal alignment between two signals to within plus or minus one half of the sampling period. Intuitively, this potentially adds a difference to two otherwise exactly matching signals equal to one half the sampling period times the change in signal voltage positive and negative.

The work of McGill, K. C., and Dorfman, L. J. (1984) was used for aligning signals sampled near the Nyquist rate. However, their algorithm can also be used to reduce the difference between oversampled waveforms to the same resolution as promised for waveforms sampled near the Nyquist rate. The algorithm uses simple phase adjustments to the FFT of the signal described in equation 3.7.

$$X_{k,\phi} = X_k e^{\frac{j2\pi k\phi}{N}} \tag{3.7}$$

Where X_k is the k^{th} value of X, the FFT of signal x. N is the number of points in x, and ϕ is the phase adjustment. Equations 3.8 through 3.13 are used to minimize the error between X_ϕ and S, the signal to which it is being aligned, iteratively through Newton's method.

$$e^2 = \frac{1}{N}|X_0 - S_0|^2 + \frac{2}{N}\sum_{k=1}^{N/2-1}|X_{k,\phi} - S_k|^2 \tag{3.8}$$

$$\phi^{(p+1)} = \phi^{(p)} + \bar{u}^{(p)} \tag{3.9}$$

$$\bar{u}^{(p)} = \begin{cases} u^{(p)} \text{ if } |u^{(p)}| \leq \frac{1}{2} \text{ and } \left.\frac{d^2e^2}{d\phi^2}\right|_{\phi^{(p)}} > 0 \\ -\frac{1}{2}sign\left(\left.\frac{de^2}{d\phi}\right|_{\phi^{(p)}}\right) \text{ otherwise} \end{cases} \tag{3.10}$$

$$u^{(p)} = \frac{-\left.\frac{de^2}{d\phi}\right|_{\phi^{(p)}}}{\left.\frac{d^2e^2}{d\phi^2}\right|_{\phi^{(p)}}} \tag{3.11}$$

$$\frac{de^2}{d\phi} = \frac{4}{N}\sum_{k=1}^{N/2-1}\left(\frac{2\pi k}{N}\right)\Im\left\{X_{k,\phi}S_k^*\right\} \tag{3.12}$$

$$\frac{d^2e^2}{d\phi^2} = \frac{4}{N}\sum_{k=1}^{N/2-1}\left(\frac{2\pi k}{N}\right)^2\Re\left\{X_{k,\phi}S_k^*\right\} \tag{3.13}$$

McGill and Dorfman determined that for ideal low-pass noise, noise with a flat spectral density up to the folding frequency, the achievable resolution is $\sigma/\beta\sqrt{E}$ where σ is the rms noise amplitude, E is the waveform's energy and β is the waveform's normalized rms bandwidth. McGill and Dorfman give a conservative rule to stop the algorithm after computing $X_{k,\phi^{p+1}}$ when inequality 3.14 holds true.

$$\left|u^{(p)}\right| < \frac{\sigma}{\sqrt{E}} \tag{3.14}$$

The full mathematics of their work is explained in McGill, K. C., and Dorfman, L. J. (1984).

3.6 Base Signal Replacement

The work done previously in the DILON project all shows that the signal being recorded drifts away from its fingerprinted values as time progresses. One idea to handle this is derived from the simulation done by Xiao, Greenstein, Mandayam, and Trappe in Xiao, L. et al. (2008a). They showed that they should be able to track the change in the signal from a wireless mobile device by replacing their fingerprint record with the test record after each testing with a result of authentic. They also tested using a least squares estimator, but found the results were not as good.

To follow the drifting of the signal, the method explored in Xiao, L. et al. (2008a) of replacing the fingerprint record with test records that have an authentic result was chosen. In order to do this with the current method of creating a fingerprint, one change was made. The fingerprint record is an average of multiple test records. So, the records that compose the fingerprint are organized in a queue, each test record determined authentic is pushed onto the queue, and the oldest record is popped from the queue. Then the fingerprint is updated as the average of the records in the queue.

3.7 Waveform Record Decimation and Precision Removal

To explore the possibilities of using a lower sampling rate on the ADC the NLP data collected has been replicated with down sampled versions for tests to be run again. Also to see the effect of using fewer bits of precision, versions of the data have been created with an increasing number of Least Significant Bits (LSBs) removed.

The choice to down sample data that had already been collected was made to make the comparisons in performance as compatible as possible. It is expected that the sampling rate is higher than required to detect differences between devices. It is also expected that using the high resolution alignment mentioned in section 3.5 will be of greater benefit at lower sampling rates.

3.8 Algorithm Design Choices

A couple algorithm choices made remain to be explained. First, the reason for averaging 100 NLP records instead of 50 or 200. By looking at figure 3.4, it is apparent that the more NLP records that are averaged together, the smaller the calculated threshold needs to be. Which should lead to a smaller chance for an impostor device to falsely be accepted, or allows us to choose a smaller false reject rate while keeping the false accept rate the same. However, there is also the consideration of time.

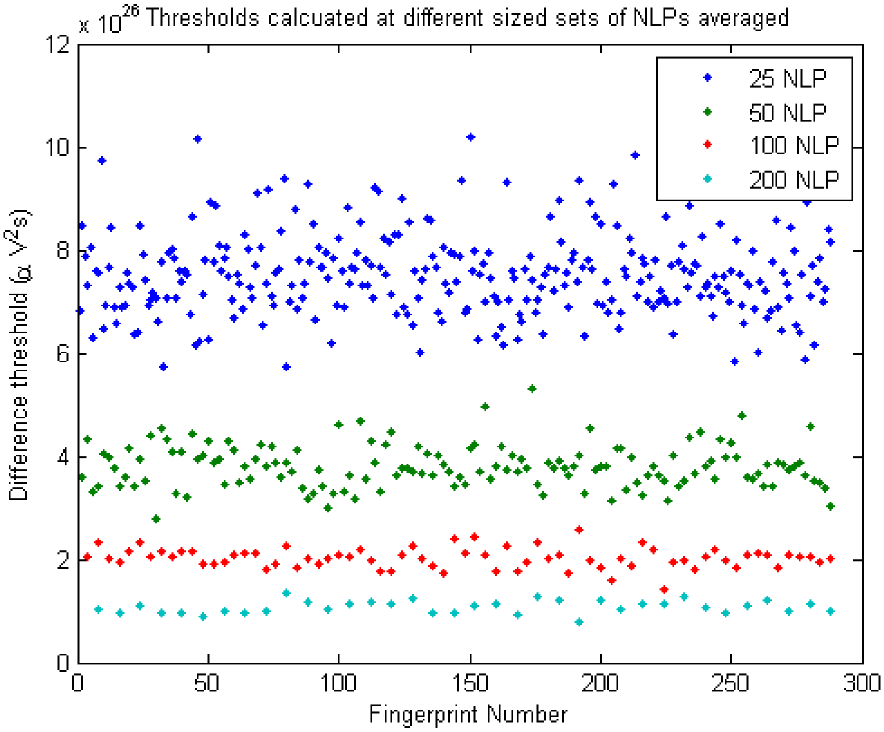


Figure 3.4: Threshold calculations for NLP records per averaging choices.

The interval between NLPs is 16ms±8ms [IEEE Standard 802.3 (2005)]. So, collecting 100 NLP should take between 0.8 and 2.4 seconds. However, using the DILON mobile test system [Erbskorn, J. W. (2009)], collecting 100 NLP requires approximately 30 seconds. The second decision made that is effected by time is the number of averaged records used in the fingerprint which was chosen to be 5. Collecting the 500 NLP to create the fingerprint requires 2.5 minutes with the DILON mobile test system, but theoretically could take just 4 to 12 seconds. For a system that could collect every NLP to meet that theoretical swiftness, it is likely desirable to use more than 100 NLP per averaged record, and more than 5 averaged records to create the fingerprint. However, with the time requirement already at 2.5 minutes to build a fingerprint of a device, the choice to stay at 100 and 5 was made.

The last design decision is the choice of trimming for each model in the model specified length adjustment. In figure 3.5 we see the trimming for 5 models of computers that have had data collected from them. All models other than the Mac Mini use the same trimming window. Although, it is apparent that the windows could be customized more to fit each models general waveform, the single trimming window was used to achieve greater simplicity in the design.

Including the choices made above and in the previous sections for use in the algorithm, the algorithm with all the specifics is given below. The last two bullet points are entry points of the algorithm which call on all the above procedures which are described in the order they are used. The procedures described can all be seen with the same name on figure 1.2.

- Noise Effect Minimization (Applied to Alignment Template)
    1. First NLP captured becomes alignment template
    2. If *model specified length adjustment*: record is trimmed based on the model of the Ethernet card
        - Dell Optiplex G1XP recored is trimmed to sample points 1001 to 1600
        - Mac Mini recored is trimmed to sample points 1001 to 1700
        - of the 2500 sample points collected by the oscilloscope
    3. Else-if *standard length adjustment*: record is trimmed as follows
        - Find where the record crosses 585mV
        - Trim the record to be 151 sample points before this point and 1000 sample points after this point; totaling 1152 points
    4. Else *variable length adjustment*: Algorithm described in figure 3.3 is followed
    5. If *wavelet filter*: Smooth average record using DWT fourth level Daubechies-4 approximate signal

6. If *low pass template filter*: Apply H shown in equation 3.5 with $\phi = \frac{\pi}{2}$

- Alignment and Trimming
  1. Align each raw record to the alignment template using cross-correlation.
  2. Trim each aligned record to the length of the alignment template.
- Waveform Averaging
  1. Average 100 aligned records into a single average record
  2. If *noise spectral amplitude scaling*: Calculate spectral amplitude of noise removed through averaging
- Noise Effect Minimization
  1. If *wavelet filter*: Smooth average record using DWT second level Daubechies-4 approximate signal
  2. If *low pass template filter*: Apply H shown in equation 3.5 with $\phi = \frac{2\pi}{3}$
  3. If *noise spectral amplitude scaling*: Scale the signal by the noise spectral amplitude
- Fingerprint Creation
  1. Push 5 noise minimized records into the base signal queue
  2. Average the records in the base signal queue for the fingerprint record
  3. Create a threshold (rejection region) from base signal queue
     (a) Determine distances for records in base signal queue using the Comparison with Fingerprint procedure
     (b) Map distances into χ_2^2 distribution
     (c) Calculate threshold using desired false negative rate, $\beta = 5\%$
- Comparison with Fingerprint
  1. Calculate MSE of the absolute value of fingerprint versus test record in frequency domain.
- Update Fingerprint
  1. Pop the oldest record from the base signal queue

    2. Push the new noise minimized record onto the end of the base signal queue.

    3. Average the records in the base signal queue for the fingerprint record.

- Fingerprint Creation Algorithm

    1. Noise effect Minimization (Applied to Template)

    2. Alignment and Trimming

    3. Waveform Averaging

    4. Nose effect Minimization

    5. Fingerprint Creation

- Decision Algorithm

    1. Alignment and Trimming

    2. Waveform Averaging

    3. Nose effect Minimization

    4. Comparison with Fingerprint

    5. If $Authenticated = No$: Negative result (reject)

    6. If $Authenticated = Yes$:

        - If *base signal replacement*: Update Fingerprint

        - Positive result (accept)

Based on the reasoning behind each of the techniques incorporated into the algorithm, most of the techniques are expected to improve performance. In the next chapter we look at how to measure performance, and examine the results.

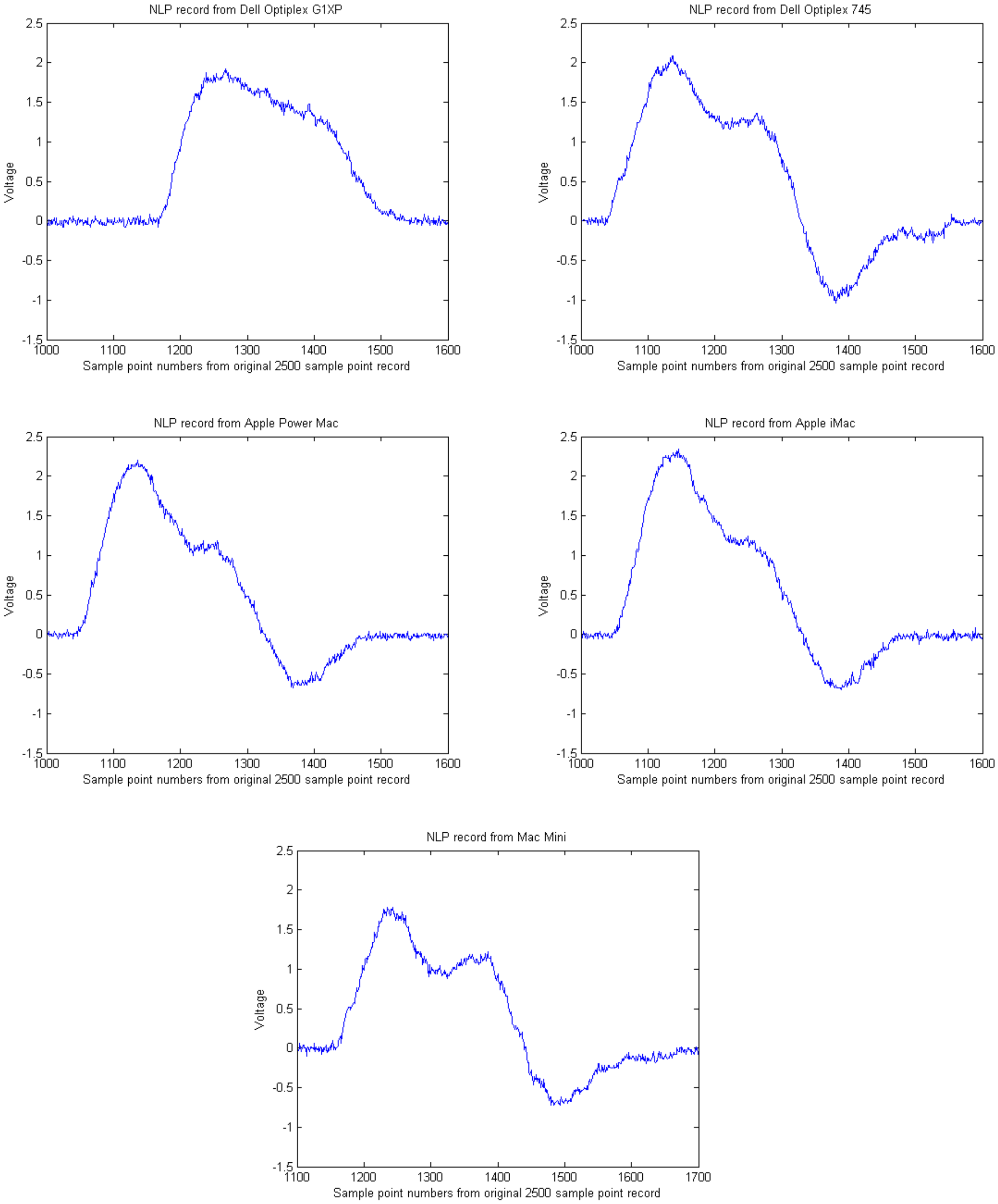


Figure 3.5: Model specified trimming choices.

CHAPTER 4. Results

How well the algorithm differentiates between Network Interface Controllers (NICs) of the first set of computers during the design space examination becomes the hypothesis for how well the algorithm will perform on the second and third sets of computers. And we test the algorithm at various sample set decimations expecting to see that at a certain point the performance of the algorithm will diminish. We then expect that by using the high resolution alignment algorithm as part of the algorithm given here, there will be a less pronounced fall in performance due to decimation.

The metric for evaluating performance of the algorithm used here has been borrowed from Erbskorn, J. W. (2009). It is the impostor minimum to authentic maximum (IMAM) ratio. This ratio is found by using the power mean squared error of comparing all the impostor devices to the authentic fingerprint, and dividing the minimum result by the maximum result of all the power mean squared error from comparing the authentic device data to the authentic fingerprint. The graphs here in chapter 4 all use IMAM ratio averaged over the whole computer set.

The algorithm is described in section 3.8 has been used with various combinations of techniques explained in chapter 3. The datasets the algorithm has been run on are given with their notation in table 4.1. The notation chosen has been $csXcY$ standing for “computer set X computer Y” where each computer has only one NIC, so there is no need to go further into saying $csXcYnicZ$.

In table 4.1, three sets of computers are listed, giving the notation and model of the machine in each set. However, each set has a different kind of unique identifier to each machine. The set of Dell Optiplex G1XP computers have their Serial Number (S/N) listed. The Dell Optiplex

Table 4.1: Dataset Notations

Notation	Machine	S/N
cs1c01	Dell Optiplex G1XP	24JWN
cs1c02	Dell Optiplex G1XP	3A70E
cs1c03	Dell Optiplex G1XP	8IVAY
cs1c04	Dell Optiplex G1XP	5SGU9
cs1c05	Dell Optiplex G1XP	3A70C
cs1c06	Dell Optiplex G1XP	6LBS0
cs1c07	Dell Optiplex G1XP	824ME
cs1c08	Dell Optiplex G1XP	824MU

Notation	Machine	Service Tag
cs2c01	Dell Optiplex 745	4C9L3D1
cs2c02	Dell Optiplex 745	GD9L3D1
cs2c03	Dell Optiplex 745	8C9L3D1
cs2c04	Dell Optiplex 745	4D9L3D1
cs2c05	Dell Optiplex 745	9D9L3D1
cs2c06	Dell Optiplex 745	7D9L3D1
cs2c07	Dell Optiplex 745	DD9L3D1
cs2c08	Dell Optiplex 745	GC9L3D1
cs2c09	Dell Optiplex 745	1D9L3D1
cs2c10	Dell Optiplex 745	6C9L3D1

Notation	Machine	ECpE Department Tag
cs3c01	Apple Power Mac	powermac01
cs3c02	Apple Power Mac	powermac02
cs3c03	Apple iMac	imac02
cs3c04	Apple iMac	imac01
cs3c05	Apple Power Mac	powermac04
cs3c06	Apple Power Mac	powermac03
cs3c07	Apple iMac	imac04
cs3c08	Apple iMac	imac03
cs3c09	Apple Power Mac	powermac05
cs3c10	Apple iMac	imac05

745 computers have their Service Tag identifiers listed. And the set of Mac computers lists the identifier tag information printed by the Electrical and Computer Engineering (ECpE) Department at Iowa State University.

It is also important to note that the NLP data collected for all the computers in computer set one was taken over the period of 3 hours per computer, while all the NLP data taken for all the computers in computer set two and three were taken over the period of 40 minutes per computer.

4.1 Design Space Examination Results

To perform the design space examination; first, the following techniques are tested:

- Standard waveform length adjustment

- Base signal replacement
- Wavelet filtering
- High resolution alignment
- Low pass filtering on the averaged records
- Low pass filtering on the alignment template

This gives us 6 binary variables, so 64 possible configurations. The remaining possible configuration choices were made to not use scaling by noise spectral amplitude, and to not use variable waveform length adjustment. The results of these configurations are shown in figure 4.1.

By looking at figure 4.1, we can see that the configuration that yields the highest IMAM ratio is to use low pass filtering on the averaged records, high resolution alignment, wavelet filtering, and base signal replacement while choosing not to use low pass filtering on the alignment template, and standard record length. The second highest set of configurations appears in the graph to be to use low pass filtering on the alignment template, wavelet filtering, and base signal replacement while not use low pass filtering on the averaged records, high resolution alignment and standard record length. A analysis of the impacts of each configuration choice is shown in table 4.2.

Table 4.2: Analysis of 64 design space configurations performed on computer set one.

Technique Used	**Ratio Impact Lower Bound**	**Ratio Impact Upper Bound**
Standard Record Length	-3.895063	-1.813130
Base Signal Replacement	0.110451	1.363972
Wavelet Filtering	1.846186	4.698396
High Resolution Alignment	-0.500944	0.500191
Low Pass Filter on Records	-0.500944	0.855517
Low Pass Filter on Template	-0.504249	0.496886

Table 4.2 lists the technique used as well as the greatest negative, or least positive, impact it has on performance as the “Ratio Impact Lower Bound,” and lists the greatest positive, or least negative, impact as “Ratio Impact Upper Bound.” A result where the lower and upper bounds are positive shows us the technique is beneficial even working with all the other varied

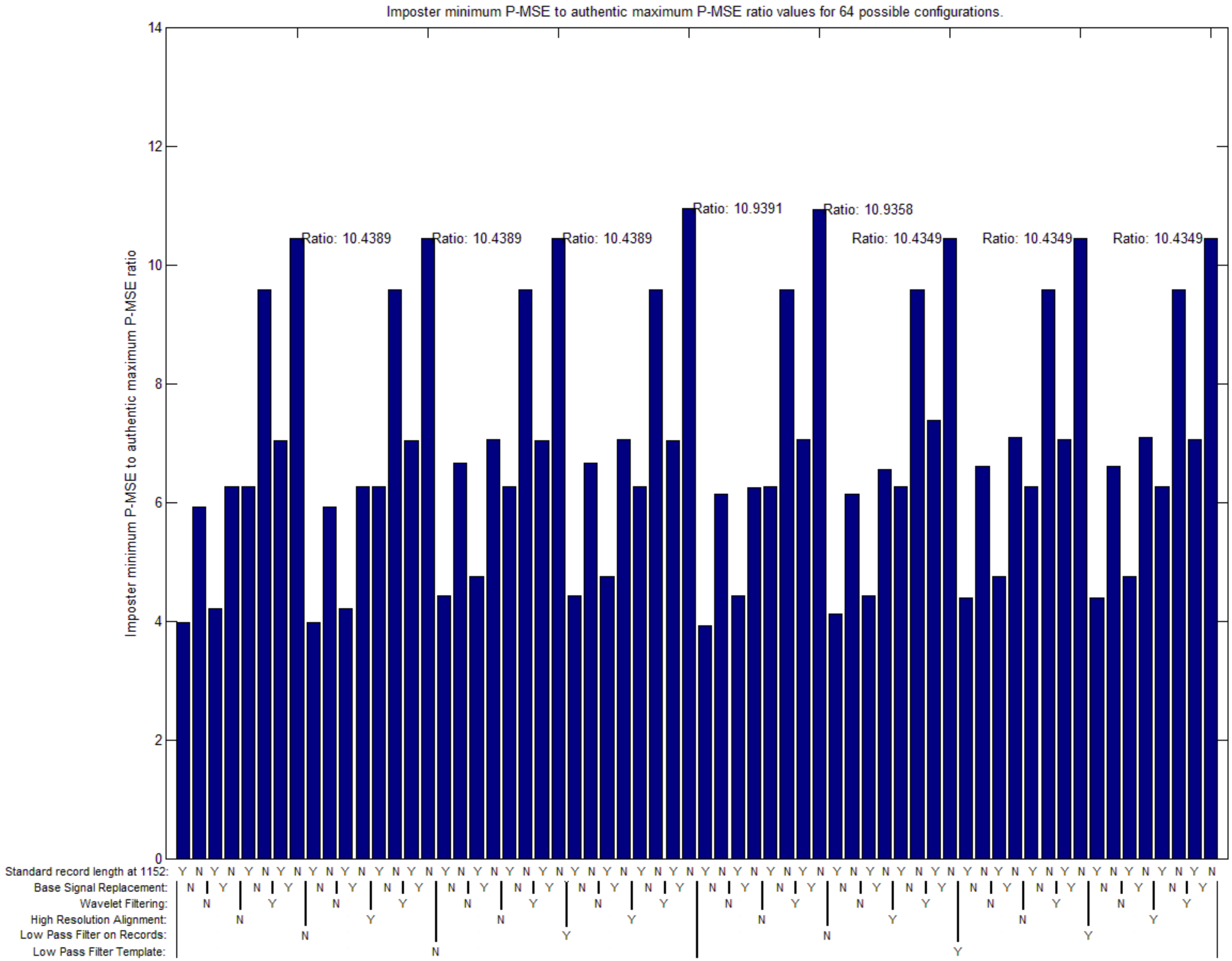


Figure 4.1: 64 design space configurations performed on computer set one.

techniques, and a result where the lower and upper bounds are negative shows us the technique is detrimental when working with any of the other varied techniques. However, when the result is negative to positive, it tells us the techniques is of unclear benefit.

What we see in table 4.2 is that three techniques benefit our algorithm independent of the other configurations that were varied. In order of greatest to least impact, these three techniques are wavelet filtering, not using the standard record length adjustment, and base signal replacement. The other techniques vary between improving and hindering our algorithm. Turning back to the graph in figure 4.1, we see that the 8 tallest bars are the ones that use these three greatest contributing techniques.

The second step was to test the configurations that had remained constant in order to test them as well. So the following techniques were tested:

- Scaling by noise spectral amplitude
- Variable waveform length adjustment
- Wavelet filtering

This gives us 3 binary variables so 8 possible configurations. The remaining possible configuration choices were made to not use low pass filtering on the averaged records and low pass filtering on the alignment template. The configuration choices were also made to use high resolution alignment and base signal replacement. Wavelet filtering was varied in both sets of configurations as it has shown to be the most interesting filter choice. The results of these configurations are shown in figure 4.2.

Figure 4.2, shows the configuration that yields the highest IMAM ratio is to use noise spectra scaling, dynamic time trimming, and wavelet filtering. And it would appear that the noise spectra filtering had a significant effect while wavelets were not being used, but was less effective than the dynamic time trimming when wavelets were also used. An analysis of the impacts of each configuration choice is shown in table 4.3.

We can see from the graph in figure 4.2 and table 4.3 that all three techniques improve our algorithm except one instance where the noise spectra filter degrades performance. We see the following techniques scoring reliably high performance on this computer set:

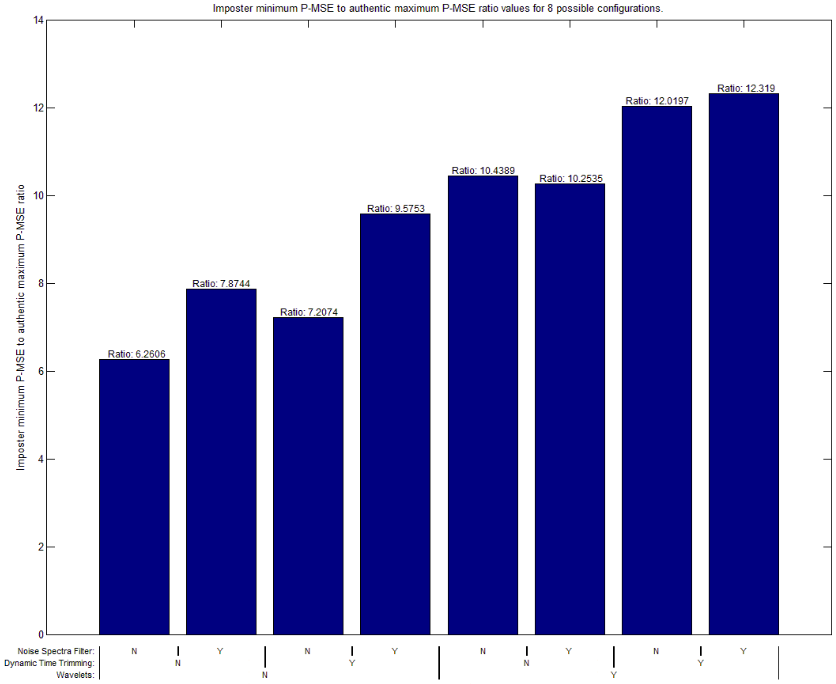


Figure 4.2: 8 design space configurations performed on computer set one.

Table 4.3: Analysis of 8 design space configurations performed on computer set one.

Technique Used	**Ratio Impact Lower Bound**	**Ratio Impact Upper Bound**
Noise Spectra Filter	-0.185432	2.367880
Dynamic Time Trimming	0.946873	2.065475
Wavelet Filtering	2.379102	4.812255

- Model specified length adjustment over standard length adjustment
- Variable length adjustment over model specific length adjustment
- Base signal replacement
- Wavelet filtering
- Noise spectra scaling, potentially

From this data we predict that the results from running our algorithm on new computer sets will result in the same techniques improving on performance.

4.2 Design Space Results Applied to new Data Sets

When we compare our hypothesis from above to figure 4.3 and table 4.5a, which indicate the performance of our algorithm with varied techniques on computer set two, we see that wavelet filtering and model specified length adjustment continue to be the highest performers. We do see that base signal replacement has a chance of lowering performance on table 4.5a. If we look at the graph, the eight highest performance configurations all use base signal replacement, but only perform slightly better than the eight configurations that do not utilize base signal replacement.

We now compare what we have previously seen in computer sets one and two with the results we get from computer set three shown in figure 4.4 and table 4.5b. What we see is that again wavelets have the strongest effect improving performance of our algorithm followed by model specified length adjustment and base signal replacement.

Table 4.4: Analysis of 64 design space configurations performed on new computer sets.

Technique Used	**Ratio Impact Lower Bound**	**Ratio Impact Upper Bound**
Standard Record Length	-1.157631	-0.555508
Base Signal Replacement	-0.012789	0.088546
Wavelet Filtering	1.013675	1.850254
High Resolution Alignment	-0.002358	0.000931
Low Pass Filter on Records	0.000000	0.342852
Low Pass Filter on Template	-0.012181	0.075636

(a) Computer set 2

Technique Used	**Ratio Impact Lower Bound**	**Ratio Impact Upper Bound**
Standard Record Length	-0.702924	-0.446122
Base Signal Replacement	0.299734	0.801687
Wavelet Filtering	0.427035	0.990244
High Resolution Alignment	0.000000	0.000000
Low Pass Filter on Records	0.000000	0.205447
Low Pass Filter on Template	-0.022785	0.017147

(b) Computer set 3

The graph for computer set three in figure 4.4 has a more definitive difference between the top eight performing configurations and the next eight. The eight that are top in this computer

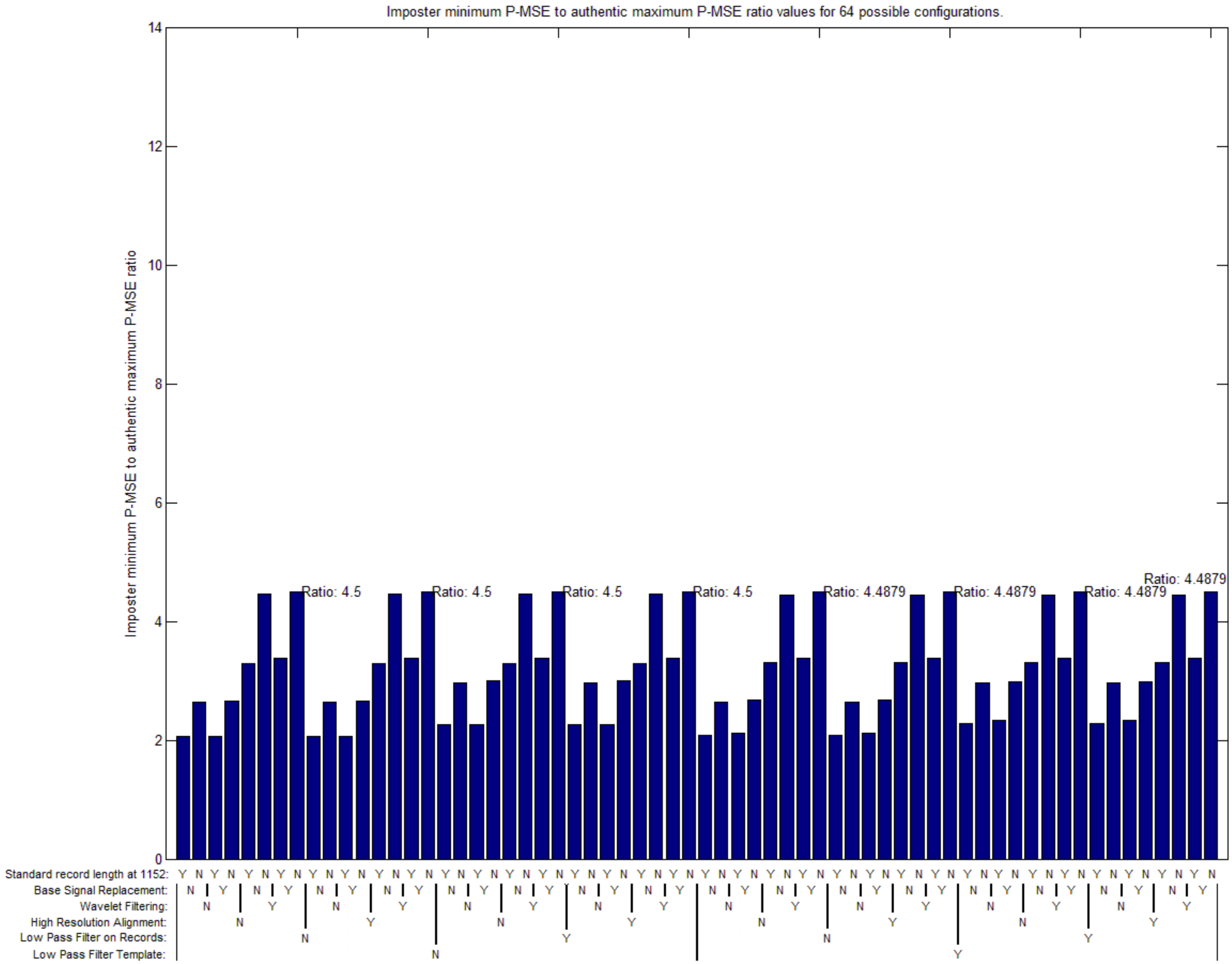


Figure 4.3: 64 design space configurations on one new data set.

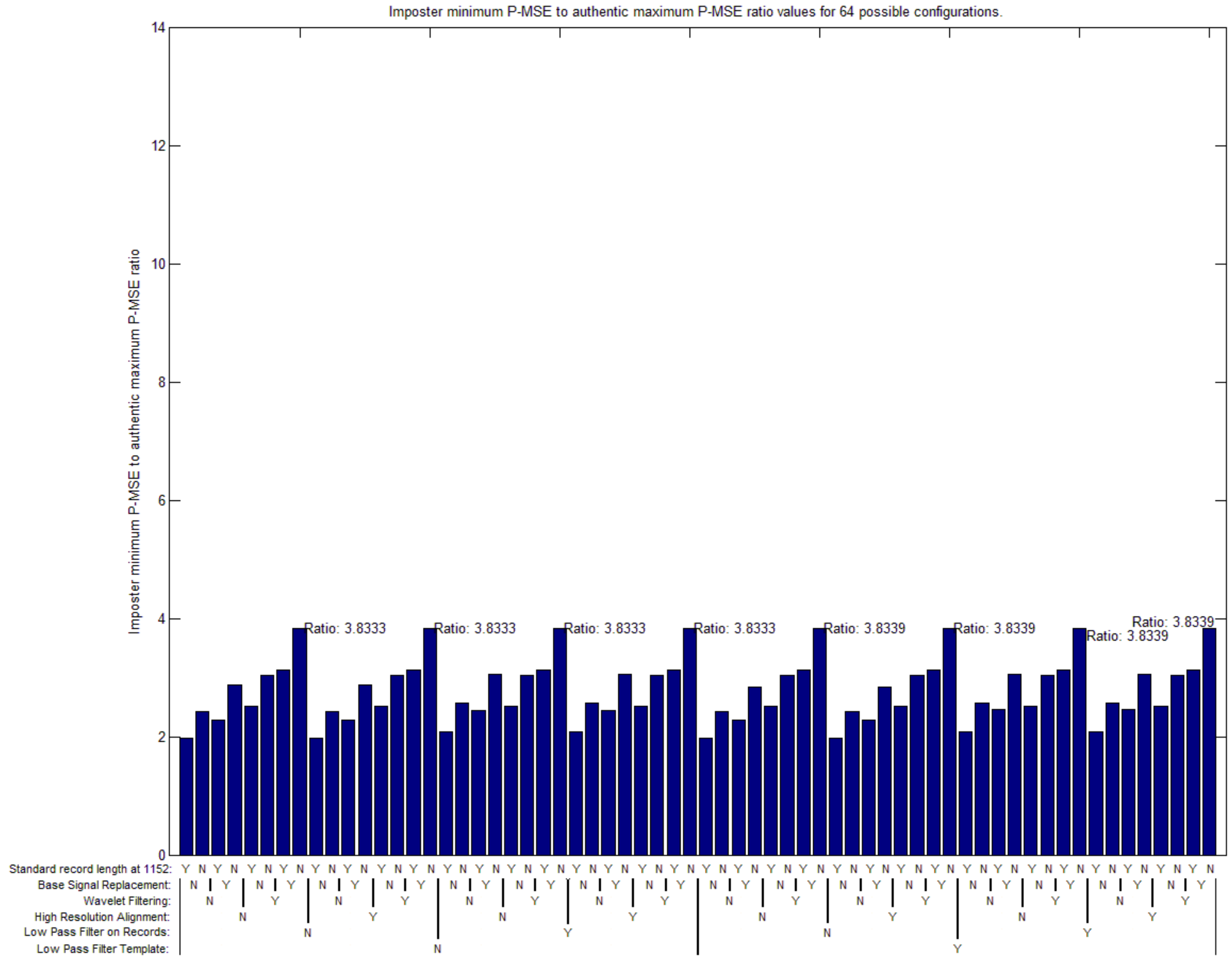
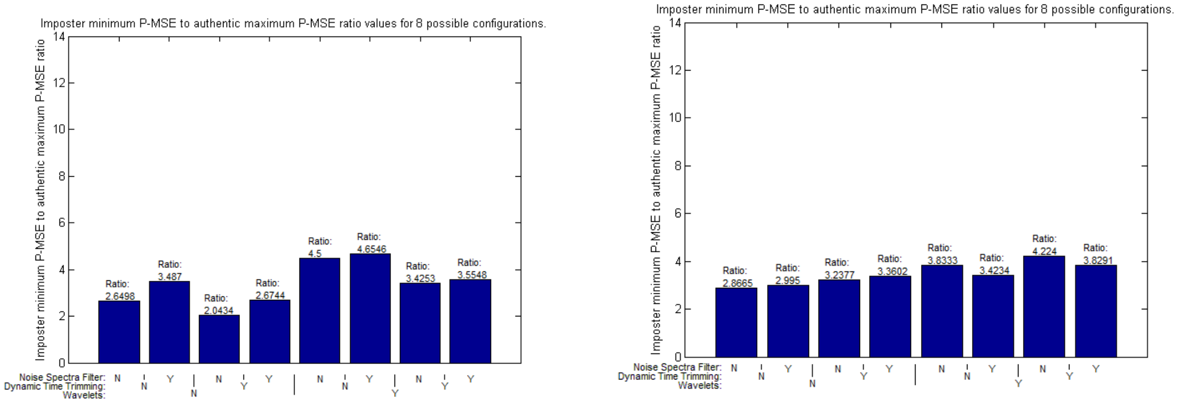


Figure 4.4: 64 design space configurations on a second new data set.

set all use model specified length adjustment, base signal replacement, and wavelet filtering.



(a) Computer set 2

(b) Computer set 3

Figure 4.5: 8 design space configurations on new data sets.

Table 4.5: Analysis of 8 design space configurations performed on new computer sets.

Technique Used	**Ratio Impact Lower Bound**	**Ratio Impact Upper Bound**
Noise Spectra Filter	0.129494	0.837256
Dynamic Time Trimming	-1.099802	-0.606407
Wavelet Filtering	0.880421	1.850254

(a) Computer set 2

Technique Used	**Ratio Impact Lower Bound**	**Ratio Impact Upper Bound**
Noise Spectra Filter	-0.409908	0.128518
Dynamic Time Trimming	0.365225	0.405641
Wavelet Filtering	0.428446	0.986258

(b) Computer set 3

Now we still must examine figure 4.5a and table 4.6a to determine if the variable length adjustment and noise spectra scaling still have a beneficial effect on scaling now that the algorithm has been run on the second computer set. We can see from the table and graph that for this data set, dynamic time trimming has a negative effect. Otherwise, the noise spectra scaling and wavelet filtering improve performance.

There is still the third computer set to check with in figure 4.5b and table 4.6b. From this table and graph, we see different results. That dynamic time trimming has always positive effect, and the noise spectra scaling may result in a negative effect. Wavelet filtering improve performance as before. By taking a close look at the previous graphs, we see that it is when noise spectra scaling is used with wavelets that the performance increase of noise spectra scaling suffers, and may potentially cause impairment.

From this data three things are concluded. It is concluded that with a more well designed algorithm, dynamic time trimming is likely to be most desirable. Base signal replacement is beneficial in most cases. Also, wavelet filtering and noise spectra scaling are beneficial when used exclusively one or the other.

4.3 Data Decimation Results

Decimation and precision lowering are performed on the computer set data before running the algorithm to give us an understanding of the sampling hardware needed for providing enough data to our algorithm. The configuration of techniques used consists of model specified length adjustment and wavelet filtering. All other techniques were not in use.

It should be noted that as data decimation factor is increased the number of NLP records that are averaged together, and the number of records that are used in creating the fingerprint are not increased. Theoretically, more samples can be substituted in for lower precision and for lower sampling rate. This experiment was for the purpose of understanding how much of the data we may be collecting at our current sampling rate of 2500 Mega-samples per second is actually aiding in detection.

From figure 4.6a we can see that the point where decimation starts effecting the false negative rate for computer set 1 is when the decimation factor is 8, and there is a large increase at decimation by 16. We also see that decimation effects the false negative rate more than the precision change does. Figure 4.6b shows us that precision has a greater effect on the false positive rates than the sampling rate does.

Based on this, the 8 bit precision data at decimation factors of 8 and 16 are chosen as interesting points to test your hypothesis that the high resolution alignment algorithm will improve our results at lower sampling rates. The points for decimation factors 1, 2, and 4 have

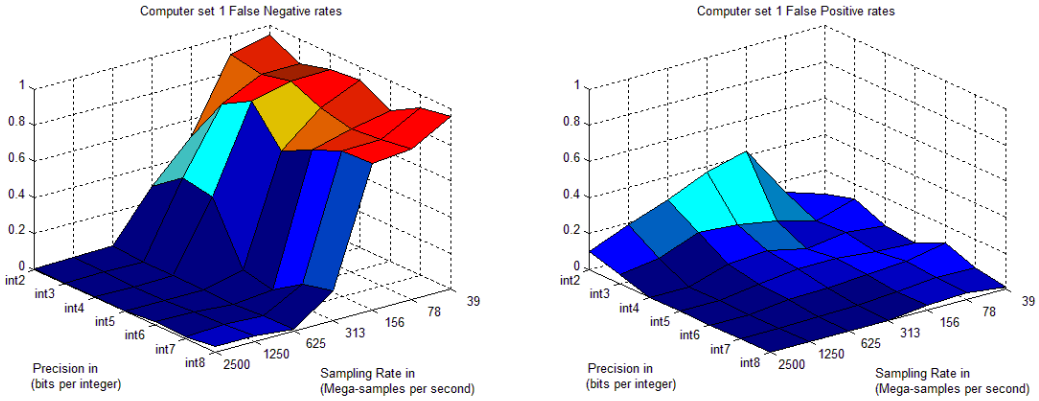


Figure 4.6: Computer set 1 false negative and false positive rates at multiple decimations and precisions.

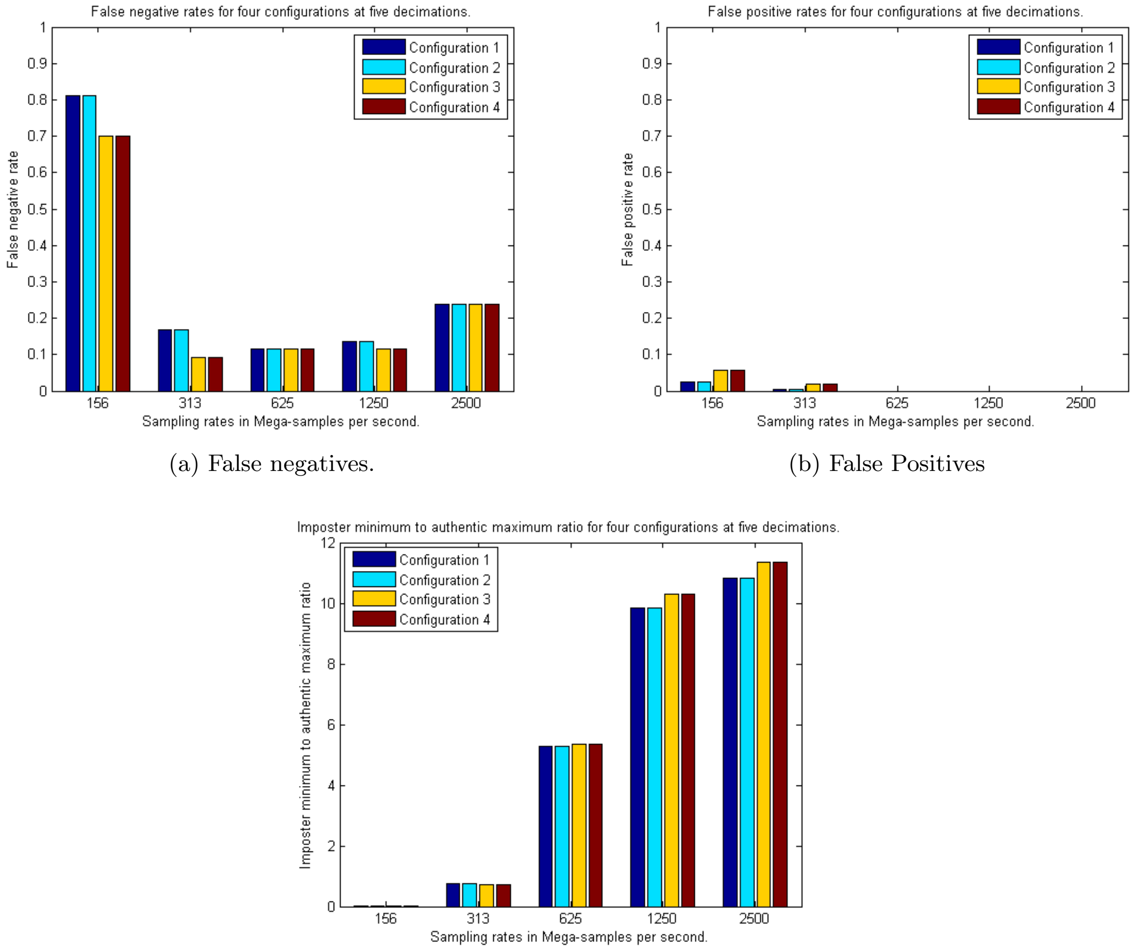
also been included for an inspection of the IMAM ratio trend.

4.4 Data Decimation with High Resolution Alignment Results

To test the benefits of the high resolution alignment algorithm as part of our algorithm, the results have been gathered for testing our algorithm on computer set 1 at decimation factors 8 and 16 using four different technique configurations. Configurations are as follows: Configuration 1 uses wavelet filtering and model specified length adjustment while all other techniques are not in use. Configuration 2 adds high resolution alignment to configuration 1. Configuration 3 uses wavelet filtering, base sample replacement, variable length adjustment, and scaling by noise spectral amplitude while all other techniques are not in use. And Configuration 4 adds high resolution alignment to configuration 3.

The graphs in figure 4.7 indicate that configurations 1 and 2 are equal in terms of false detections, and that configurations 3 and 4 are also equal. This shows that the high resolution alignment did not have enough effect to improve detection.

The graph in figure 4.7c again indicates that configurations 1 and 2 are equal, and configurations 3 and 4 are also equal this time in terms of the IMAM ratio. This shows that the high resolution alignment is not having any noticeable effect.



(a) False negatives.

(b) False Positives

(c) IMAM ratio

Figure 4.7: Computer set 1 false negative and false positive rates and IMAM ratios of multiple configurations at 5 decimations of interest.

CHAPTER 5. Summary and Discussion

This research has focused on using the tolerances in manufacturing Ethernet devices to detect differences in the signals sent by two different devices. Specifically, the signal used for device comparison is the IEEE 802.3 NLP which has been proposed previously in Erbskorn, J. W. (2009).

5.1 Conclusion

The NLP signal data collected has been expanded to include two new computer sets. And from those computer sets, we have results using the IMAM ratio that shows a promising improvement in performance by using: Wavelet filtering or noise spectra scaling (exclusive), variable length adjustment, and base signal replacement. And of noise spectra scaling and wavelet filtering, wavelet filtering showed the best results.

Therefore, with this algorithm, it is determined that wavelet filtering or noise spectra scaling may be used. Wavelet transformation requires $O(n \times log(n))$ multiplications where n is the length of the signal being transformed. However, the algorithm uses a specific number of levels, m, to the wavelet filter bank, the wavelet filtering requires $O(m \times n)$ multiplications as it involves applying equation 3.1 m times. Of course, m has an upper bound of $log(n)$, and is a constant, so it is also appropriate to write the complexity as $O(n)$ multiplications. This must be performed once per 100 NLP records that are averaged. It also offers a better performance increase over noise spectra scaling based on the numerical results in chapter 4.

A positive note for the noise spectra scaling is that it is conceptually simpler. As it involves scaling frequencies present in the signal so that the frequencies with the most fluctuation effect detection the least. However, in order to scale the signal by the calculated noise spectra, the

FFT must be performed to put the signal in the frequency domain which requires $O(n \times log(n))$ multiplications. And there is an initial cost when calculating the fingerprint; the noise spectral amplitude must also be calculated which requires calculating the FFT for each raw NLP record that goes into the fingerprint. Although not tested here, the noise spectra scaling should be more flexible for use in different noise environments. The wavelet filter removes noise at a certain threshold that does not change; however, the noise spectra scaling scales the signal based on the noise present at the time the fingerprint was created. Which provides an amount of flexibility limited with the idea that the noise environment will not change greatly from the time the device was fingerprinted.

The low pass filter used in this survey of the design space did not offer the same flexibility, the parameter ϕ was chosen ahead of time. The low pass filter is has a similar complexity to the noise spectral amplitude scaling, but does not have as high of an initial cost, and requires fewer operations after the FFT is performed since it either multiplies FFT values by one or zero. The conservative values for ϕ showed results that indicate this technique offered little improvement to performance. However a reexamination given in appendix C shows that a more aggressive choice of ϕ would possibly lead to better results.

Model specified length adjustment showed the greatest performance, but is the most complex of the length adjustments to implement. Standard length adjustment is the easiest, but has the worst performance. It is determined that the variable length adjustment is probably the best choice due to the fact that it is not much harder to implement than the standard length adjustment technique and requires not much more hardware, without being much worse in performance than the Model specified length adjustment. Also with a bit of modification, it may turn out to be as good as the model specified length adjustment.

No matter the choice of using variable length adjustment, standard length adjustment, or model specified length, they would all have the same hardware requirements for performing FFTs or DWTs. This is because at maximum the record should need to be 1125 points long at a sampling rate of 2.5×10^9 samples per second in order be sure to capture the section of the NLP that allows the most variation. So, the FFT hardware would be designed for 1152 since 1152 breaks down into $2^7 \times 3^2$ rather than $3^2 \times 5^3$ which 1125 breaks down into and there are more well known optimizations for FFTs with radix-2. For the cases where variable length adjustment and model specified length adjustment do not use the full 1152 points, the waveform can be padded with zeros before performing the FFT. The unique part of each method is the hardware required for choosing the data points that go into the FFT.

Standard length adjustment just has to find the point that crosses 585mV and copy all the correct values from before and after that point into their corresponding memory or register locations. The variable length adjustment only requires extra hardware to find that same point, then find the point ahead of there where the signal starts, and at the same time it should find the end of the signal by finding where the average of 20 points is greater than the expected noise which is 50mV. The model specified record length requires Read Only Memory (ROM) which will indicate the window of waveform points to use dependent on the model the waveform is detected to be from. Which means an initial round of detection is required to determine the model the signal came from. The actual algorithm for that determination is not covered here, but is available in Erbskorn, J. W. (2009).

Base signal replacement improves performance. And it only requires $4n$ additions and n divisions by 5 in order to average the five records that compose the base signal queue each time a record is authenticated.

The high resolution alignment algorithm did not appear to have any effect on performance. It was also the technique with largest hardware requirement when considering implementation. It requires an FFT to be performed on each raw NLP record for alignment, then error between the signal being aligned and the template is minimized using the Newton's method to vary the phase of the FFT values. Each iteration of the Newton's method requires $O(n)$ multiplications and depending on the representation of complex number used, may also require up to n CORDIC operations.

The possibility of reducing the sampling rate and precision of the collected NLP records to the point where Analogue-to-Digital Converters (ADCs) are more readily available and inexpensive has been explored. The results on reducing sample rate and precision of the collected

NLP records showed that sample rate effected false negative (device falsely unauthenticated) rates noticeably at decimation factors 8 and 16. Furthermore, false positive (devise falsely authenticated) rates were mostly effected by reduced precision. It is also apparent that performance of the algorithm, as determined by the IMAM ratio, decreases with increasing data decimation before there is an increase in false negatives.

In conclusion, what all our data really points to is that more data collected from the computers for creating the fingerprints and creating the averaged records to test against the fingerprint leads to better detection. Section 4.4 indicates that as soon as the data has been decimated, the performance of the algorithm suffers. Figure 3.4 shows us that the more raw NLP records averaged together the lower our detection threshold has to be indicating less difference between averaged records. The techniques that have had the greatest effect in improving performance of the algorithm have been wavelet filtering and scaling by the noise spectral amplitude which somewhat emulate the same effect as averaging a greater number of NLP records together. With a better data collection system that captures all NLPs at the full rate of 1 per 8ms to 24ms at the same or better sampling rate and precision, more NLP records could potentially be averaged together without requiring an increase in wait time, and we could potentially see greater still improvement in this algorithm. However based on the study in appendix C, although limited in size, it is shown that newer Ethernet devices are more difficult for making correct detections. Also, the obstacle illustrated in figure A.10c would still remain. In this figure we see a sharp jump in the difference between the NLPs sent and the fingerprint. This sort of behavior is what this algorithm was meant to detect as it may indicate a quick change in the cable impedance due to tampering, so if it occasionally happens without tampering, the algorithm as it is cannot make a correct decision on this behavior.

5.2 Future Work

This work does not cover a method in which the device NLP records are compared in the wavelet domain, which ought to have advantages over using the FFT on a pulse shaped waveform. In future research, that would be the focus of the changes dependent on the results of initial tests.

The work that has been done related to intrusion detection at the physical layer requires extra hardware in the detection device. This necessitates that any approach to implementing ought to include all promising methods of intrusion detection within reason because the hardware cannot be changed. However, firmware and software can be changed. A brilliant way to sell the idea of physical layer IDS is also likely needed to overcome the cost of the additional hardware cost. Also, before research can get that far, work on detection with signals at higher bit rates is likely to be necessary.

APPENDIX A. Results on Temperature Variation

One area of importance, which gives us a reason for using base signal replacement to track the signal, and may also lead to a technique better than base signal replacement, is to observe the change in the recorded signal as the ambient temperature of the sending device changes.

Temperature Equipment

The equipment used for examining the signal change due to temperature variation is shown here. In order of appearance, the depicted equipment is: Frigidaire temperature chamber, cs1c5, one-wire board with temperature probes, atmospheric agitation fan, the Ranco electronic temperature control.

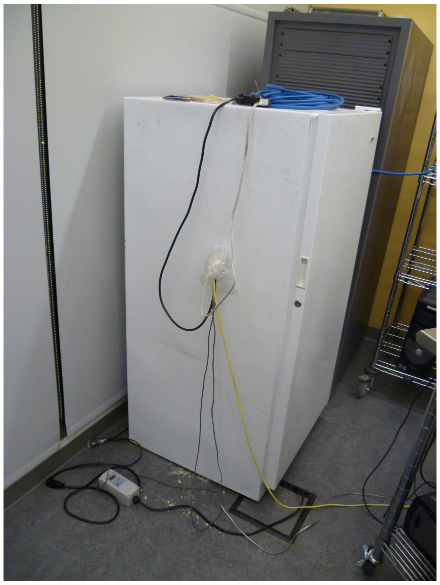


Figure A.1: Temperature chamber external view.

Figure A.1 displays the freezer unit used to cause temperature change, and next to it is the electronic temperature controller that controls power to the freezer. By changing the desired temperature on the temperature controller, we can externally change the temperature the freezer will run to.

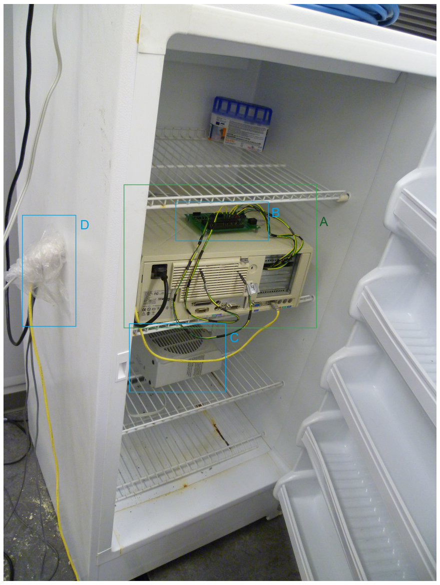


Figure A.2: Temperature chamber with contents marked.

Figure A.2 shows the equipment that is put into the freezer. (A) encloses the current computer in the temperature chamber, which is cs1c5 in this picture; we have a good look at its back end. (B) encloses the one-wire board which has five temperature probes reading temperature at different locations in and on cs1c5. (C) encloses the atmospheric agitation fan, which is continuously on to prevent stratification, and aid uniform temperature throughout the height of the temperature chamber. (D) encloses the plastic insulator that wraps around the cords exiting the temperature chamber to aid the efficiency of the freezer and uniformity of the temperature.

Figure A.3 provides a closer look at the equipment positioned inside the temperature chamber. In this photo it is now possible to see where all the wires exit the chamber, and the full atmospheric agitator is now in view.

Figure A.4 provides a closer look at the Ranco Electronic Temperature Control, which is set to 65 ^{o}F in this picture. Which was the approximate start and end temperature for each

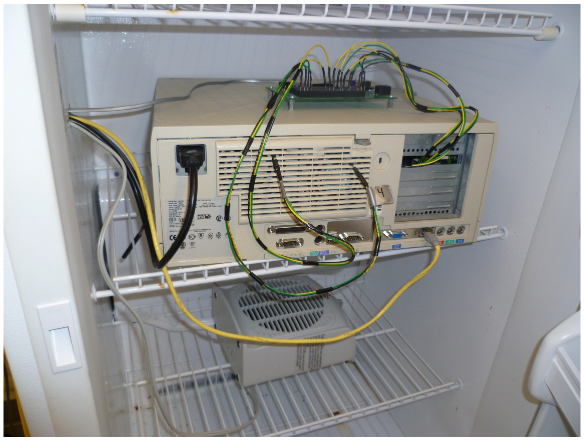


Figure A.3: Temperature chamber internals closer view.

Figure A.4: Temperature controller used: Ranco Electronic Temperature Control.

data collection session. This setting was chosen because it was slightly below the temperature of the room the chamber is situated in resulting in a stable starting and ending temperature for this experiment.

Figure A.5 provides a look at the entirety of the temperature variation test system. In the foreground is the DILON mobile test system described in more detail in Erbskorn, J. W. (2009). In the background is the temperature chamber with cs1c5 currently inside.

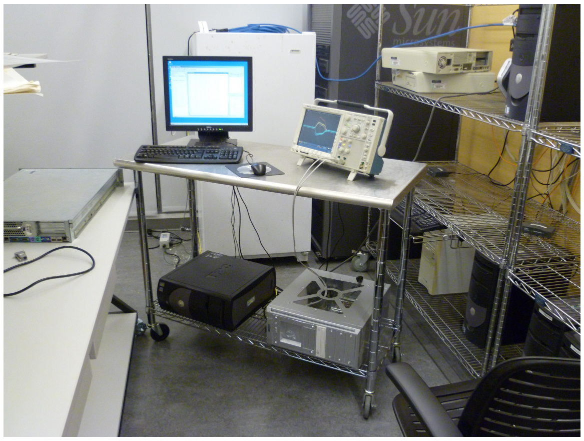


Figure A.5: Temperature chamber (back) and DILON mobile test system (front).

Temperature Results

The procedure used for taking temperature controlled data was to place the NLP sending computer in the temperature chamber with Ranco electronic temperature control set to 65 ^{o}F. The recording is started, and the temperature control is set to 36 ^{o}F, and the temperature is driven down towards that limit. At approximately hour 4, the control is set again to 65 ^{o}F, and allowed to passively increase in temperature back to that limit for 4 hours.

There are 5 temperature probes used to record the temperature during data recording. None of them appeared to be greatly more correlated to the result of matching the corresponding records to the fingerprint, so the probe closest to the Ethernet port was chosen.

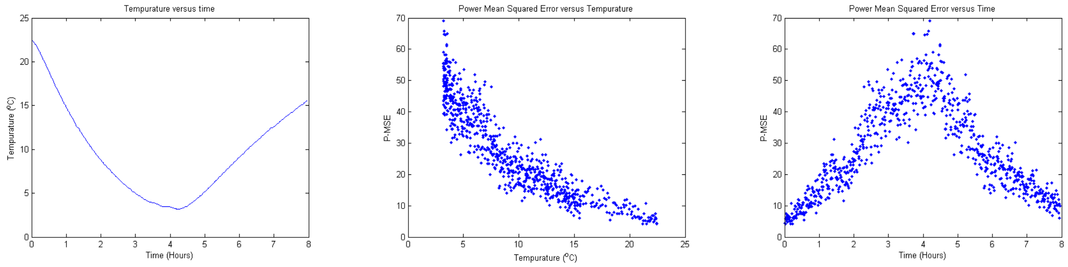


Figure A.6: Temperature results for computer set 1 computer 1.

Figure A.6 shows the temperature recorded across time, the power mean squared error result of comparing the later NLP records against its own fingerprint plotted against temperature, and the power mean squared error result plotted against time. These same graphs are given in the next figures for cs1c3, cs1c5, and cs1c7.

We can see in A.6 that the temperature correlates well with the power mean squared error result. We can also see that as the temperature approaches the original state, so does the result. And that the relationship appears to be quadratic.

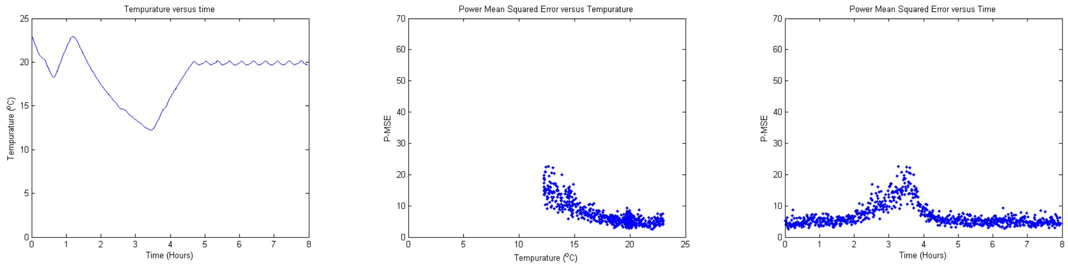


Figure A.7: Temperature results for computer set 1 computer 3.

In figure A.7, there is an anomaly with the temperature graph indicating that a little over half way through hour one, the temperature chamber warmed up again, then went back to cooling after the hour mark. The control was also changed back to 65 ^{o}F slightly early and increased in temperature rather quickly, so it spend a large amount of time at its high end limit. However, from the middle graph in A.7 it is apparent that the result corresponds well to the temperature, and the relationship again appears to be quadratic.

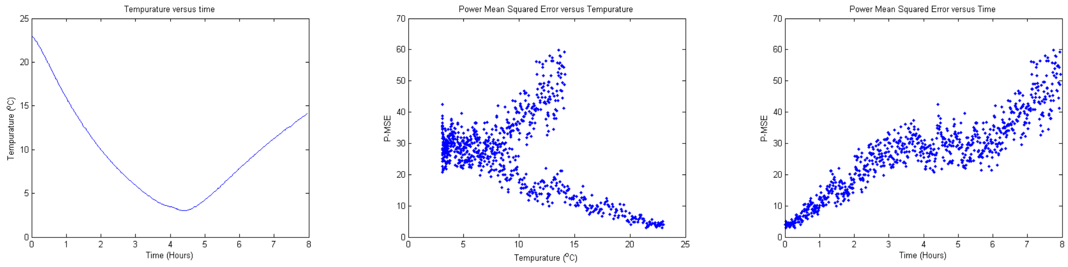


Figure A.8: Temperature results for computer set 1 computer 5.

Figure A.8 shows a different anomaly. The temperature graph indicates the temperature equipment worked as expected and has a similar curve to the one in figure A.6. However, the P-MSE result appears to follow the quadratic relationship as the temperature decreases, and again while the temperature increases, but then diverges after the temperature has increased back toward the original temperature for only a short time.

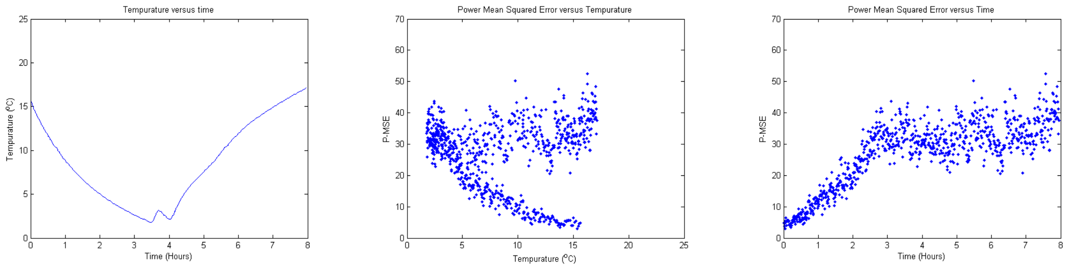


Figure A.9: Temperature results for computer set 1 computer 7.

In figure A.9 we see the same anomaly as in A.8. This anomaly is a large obstacle in trying to relate the change in a signal to change in temperature. To take a closer look cs1c5 had the procedure run on it again over 40 hours instead of 8. In order to prevent the amount of data collected from being too large, each set of 100 NLP records was taken over the time of 3 minutes rather than 30 seconds.

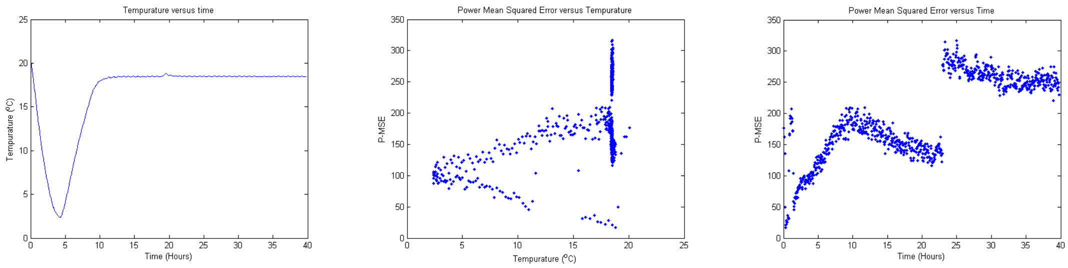


Figure A.10: Temperature results for computer set 1 computer 5 across 40 hours.

This time cs1c5 shows more discontinuous behavior in figure A.10. And it does not return to its original state in the 40 hours it was given. The large disconnects shown in the result graph are an indication that it may not be possible to authenticate a device based on the characteristics of its Ethernet signal in this way.

APPENDIX B. Alternate Evaluation Metrics

In chapter 4, the primary evaluation method was to use the IMAM ratio used in Erbskorn, J. W. (2009). There are other metrics that can also be used to indicate the performance of each technique used. Firstly, the IMAM ratio used to create the graphs in chapter 4 uses the mean of the IMAM ratio over all the authentic cards in each computer set. In detail, the impostor minimum result and authentic maximum result is found for each authentic card then the ratios are averaged. However, it is of interest to look at the IMAM ratio for the devices that are closest to matching exclusively so that we know which techniques change our worst case in positive ways.

The IMAM ratio metric focuses on preventing outliers of the algorithms from crossing a theoretical boundary that can be set as long as the ratio is greater than one. However it is possible to still differentiate devices from each other when there are outlying points that cross the theoretical boundary. Thus it is important to examine the number of standard deviations between impostor and authentic device.

The first metric using standard deviations looks at the number of standard deviations of the impostor devices results before the impostor device reaches the authentic devices mean value. This metric tells us how the techniques we are using effect the variability and distance of the impostor devices, so that we can determine which techniques make false positives more likely. The choice was also made to look at specifically the minimum values of standard deviations per computer set in order to focus on improving the worst case.

The second metric using standard deviations looks at the number of standard deviations of the authentic devices results before reaching the closest impostor devices mean value. This metric tells us how the techniques effect the variability of the authentic devices with respect to the distance to the impostor devices so that the likelihood of false negatives can be determined. The choice was made to examine closest matching authentic to impostor pair of devices in order to again focus on improving the worst case.

The results of all these metrics are displayed in table B.1. The average improvement or impairment of the algorithm is given with the recommendation to use or not use a technique is listed directly to the right of the average given for each metric. The IMAM Ratio for worst cases is shown under the IMAM column header. Impostor device standard deviations to authentic device mean is abbreviated as ISDA in the column headers. And authentic device standard deviations to impostor device mean is abbreviated ASDI. The reason that the average improvement or impairment of the algorithm is given is because it followed well with the determinations that could be made from the graphs, and does well to indicate the importance of using or not using a specific method.

The impostor device standard deviations to authentic device mean metric was selected for also showing the graphs of the effect each technique had on the algorithm. These graphs are shown in figure B.1 and B.2.

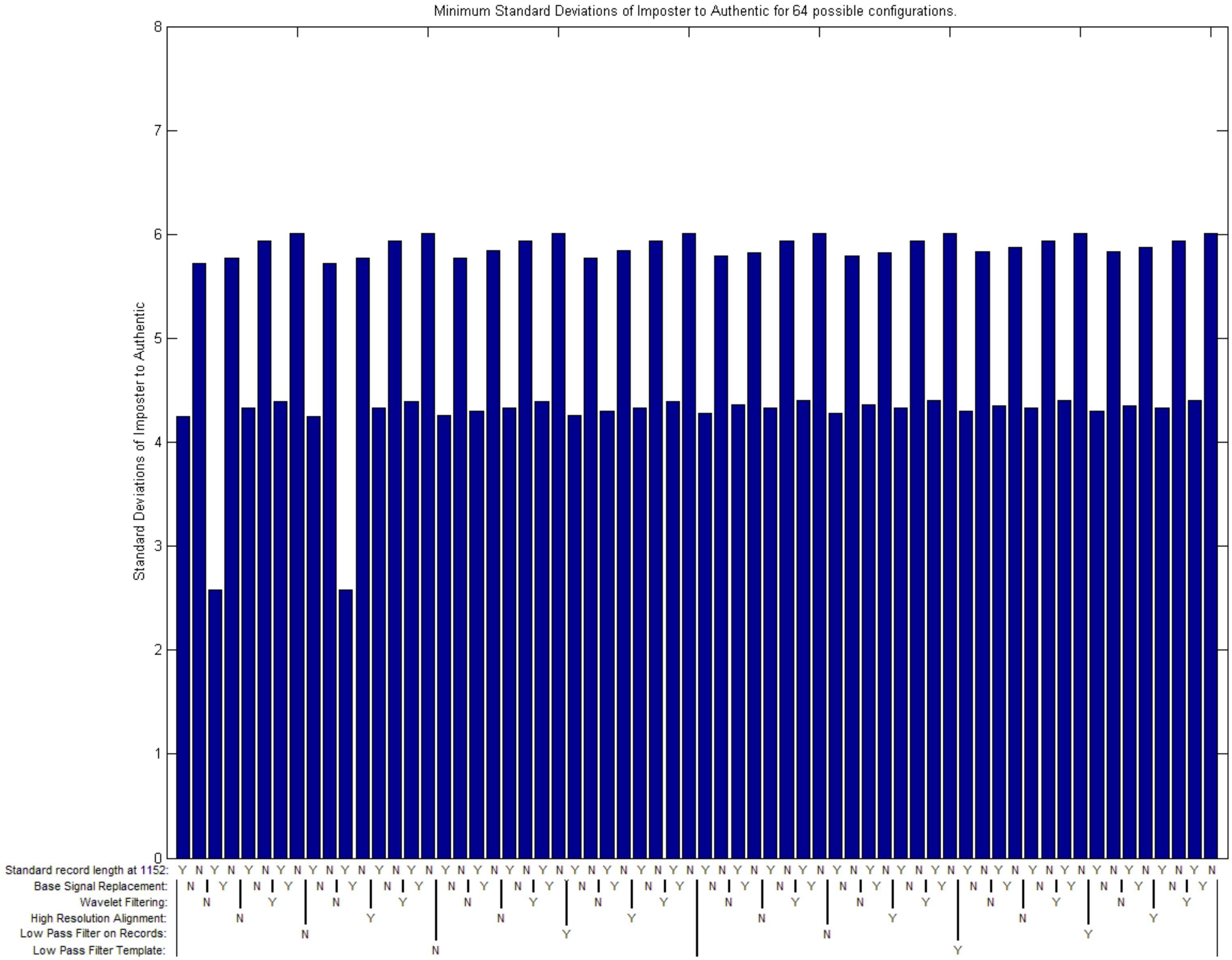


Figure B.1: 64 design space configurations performed on computer set one using impostor standard deviation to authentic mean metric.

Table B.1: Technique Usage Recommendations based on Different Evaluation Metrics

Set	Technique Used	IMAM	Use	ISDA	Use	ASDI	Use
CS1	Standard Record Length	0.419116	Yes	1.662229	Yes	3.346357	Yes
	Base Signal Replacement	0.128365	Yes	-0.047738	–	3.063711	Yes
	Wavelet Filtering	0.427035	Yes	0.225098	Yes	1.382983	Yes
	High Resolution Alignment	0.000000	–	0.000000	–	0.000000	–
	Low Pass Filter on Records	0.041638	–	0.122399	–	0.093434	–
	Low Pass Filter on Template	0.008932	–	0.132811	–	0.165898	–
	Noise Spectra Filter	-0.150518	No	-0.399893	No	-1.554990	No
	Dynamic Time Trimming	0.211363	Yes	0.415514	Yes	2.403407	Yes
CS2	Standard Record Length	-0.072373	–	0.231597	–	-0.701677	No
	Base Signal Replacement	-0.306393	No	-0.500187	No	-1.396652	No
	Wavelet Filtering	0.778338	Yes	0.232915	Yes	4.403474	Yes
	High Resolution Alignment	0.000000	–	0.000153	–	0.004285	–
	Low Pass Filter on Records	0.062166	–	0.016540	–	0.318399	Yes
	Low Pass Filter on Template	0.068023	–	0.102178	–	0.078464	–
	Noise Spectra Filter	0.554348	Yes	0.297993	Yes	6.551133	Yes
	Dynamic Time Trimming	-0.472136	No	-0.780934	No	-7.266581	No
CS3	Standard Record Length	0.024208	Yes	1.039929	Yes	0.082317	Yes
	Base Signal Replacement	-0.012411	No	-0.184450	No	-1.016726	No
	Wavelet Filtering	-0.030399	No	-0.027897	–	0.001218	Yes
	High Resolution Alignment	0.000000	–	0.000000	–	0.000000	–
	Low Pass Filter on Records	-0.006949	–	-0.186557	–	-0.011880	No
	Low Pass Filter on Template	0.000214	–	0.004858	–	0.000265	–
	Noise Spectra Filter	-0.012249	No	0.082074	–	0.017546	Yes
	Dynamic Time Trimming	-0.024398	No	-2.279157	No	-0.263113	No

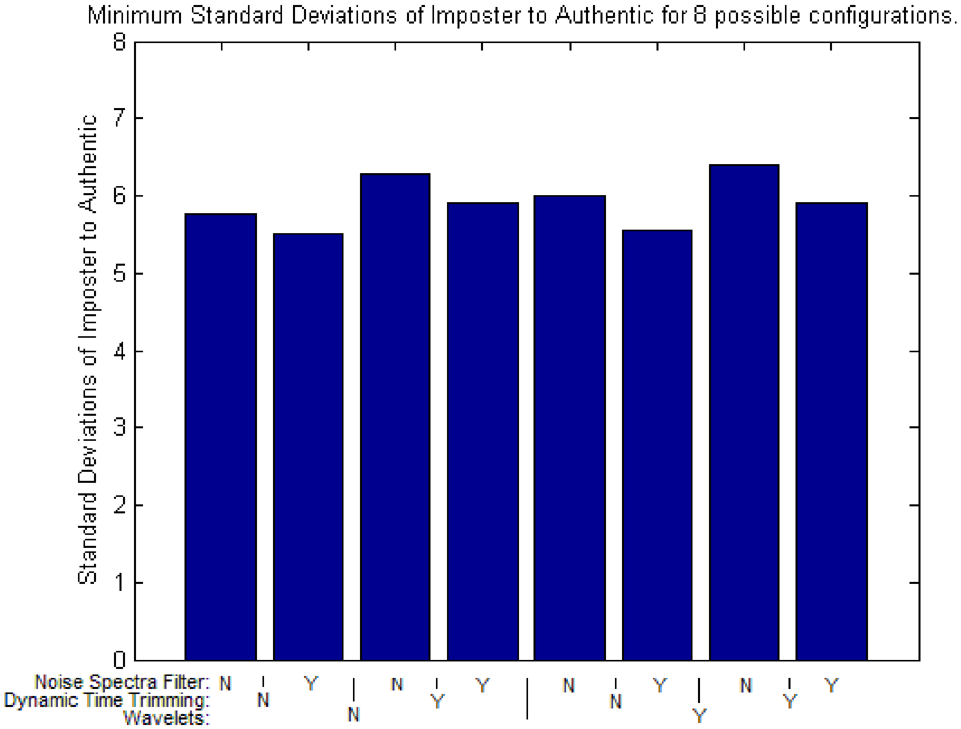


Figure B.2: 8 design space configurations performed on computer set one using impostor standard deviation to authentic mean metric.

APPENDIX C. Applied Results

Through applying the results found in chapter 4, the following results have been found, and formatted into tabular form. In each table a cell is the result of using the fingerprint of the computer in the row heading and testing the records from the computer in the column heading against that fingerprint. In the confusion matrices, the number indicate only false determinations. So, if it is the same computer in both column and row it is indicating false rejection rate (falsely un-authenticating), otherwise it is indicating the false accept rate (falsely authenticating).

All tables are for the results of the algorithm while using base signal replacement and model specific length adjustment. And In all tables the algorithm was not using high resolution alignment, nor the other options of length adjustment. The differences in the following tables is the use of wavelet filtering, noise spectral amplitude scaling, or aggressive low pass filtering, all exclusive of each other. Tables C.1 through C.3 show the confusion matrix for using wavelet filtering, and tables C.4 through C.6 show the IMAM ratio results for the same simulation of our algorithm. Then, tables C.7 through C.9 show the confusion matrix for using noise spectra scaling, and tables C.10 through C.12 show the IMAM ratio results for this same simulation of our algorithm. Finally, tables C.13 through C.15 show the confusion matrix for using low pass filtering with $\phi = \frac{42\pi}{300}$, while tables C.16 through C.18 show the IMAM ratio results for the same simulation of our algorithm.

We can see from these results that the simple low pass filter appears to have the best effect on both the false detection rate and the IMAM ratios. And may have been prematurely denounced based on the earlier conservative choice of ϕ. Through examining these tables, it can also be seen that it does appear to be more difficult to differentiate between cards of the

Table C.1: Confusion Matrix Result from Highest Performing Techniques

	cs2c1	cs2c2	cs2c3	cs2c4	cs2c5	cs2c6	cs2c7	cs2c8	cs2c9	cs2c10
cs2c1	0	0.640	0	0	0	0	0	0	0	0
cs2c2	0	0	0	0	0	0	0	0	0	0
cs2c3	0	0	0	0	0	0	0	0	0	0
cs2c4	0	0	0	0	0	0	0	0	0	0
cs2c5	0	0	0	0	0	0	0	0	0	0
cs2c6	0	0	0	0	0	0	0	0	0	0
cs2c7	0	0	0	0	0	0	0	0	0	0
cs2c8	0	0	0	0	0	0	0	0	0	0
cs2c9	0	0	0	0	0	0	0	0	0	0
cs2c10	0	0	0	0	0	0	0	0	0	0
cs3c1	0	0	0	0	0	0	0	0	0	0
cs3c2	0	0	0	0	0	0	0	0	0	0
cs3c3	0	0	0	0	0	0	0	0	0	0
cs3c4	0	0	0	0	0	0	0	0	0	0
cs3c5	0	0	0	0	0	0	0	0	0	0
cs3c6	0	0	0	0	0	0	0	0	0	0
cs3c7	0	0	0	0	0	0	0	0	0	0
cs3c8	0	0	0	0	0	0	0	0	0	0
cs3c9	0	0	0	0	0	0	0	0	0	0
cs3c10	0	0	0	0	0	0	0	0	0	0
cs1c1	0	0	0	0	0	0	0	0	0	0
cs1c2	0	0	0	0	0	0	0	0	0	0
cs1c3	0	0	0	0	0	0	0	0	0	0
cs1c4	0	0	0	0	0	0	0	0	0	0
cs1c5	0	0	0	0	0	0	0	0	0	0
cs1c6	0	0	0	0	0	0	0	0	0	0
cs1c7	0	0	0	0	0	0	0	0	0	0
cs1c8	0	0	0	0	0	0	0	0	0	0

Table C.2: Continuation of Table C.1

	cs3c1	cs3c2	cs3c3	cs3c4	cs3c5	cs3c6	cs3c7	cs3c8	cs3c9	cs3c10
cs2c1	0	0	0	0	0	0	0	0	0	0
cs2c2	0	0	0	0	0	0	0	0	0	0
cs2c3	0	0	0	0	0	0	0	0	0	0
cs2c4	0	0	0	0	0	0	0	0	0	0
cs2c5	0	0	0	0	0	0	0	0	0	0
cs2c6	0	0	0	0	0	0	0	0	0	0
cs2c7	0	0	0	0	0	0	0	0	0	0
cs2c8	0	0	0	0	0	0	0	0	0	0
cs2c9	0	0	0	0	0	0	0	0	0	0
cs2c10	0	0	0	0	0	0	0	0	0	0
cs3c1	0	0	0	0	0	0	0	0	0	0
cs3c2	0	0	0	0	0	0	0	0	0	0
cs3c3	0	0	0.133	0.880	0	0	0	0	0	0
cs3c4	0	0	0.307	0	0	0	0	0	0	0
cs3c5	0	0	0	0	0.600	0	0	0	0	0
cs3c6	0	0	0	0	0	0	0	0	0	0
cs3c7	0	0	0	0	0	0	0.013	0	0	0
cs3c8	0	0	0	0	0	0	0	0.907	0	0
cs3c9	0	0	0	0	0	0	0	0	0.587	0
cs3c10	0	0	0	0	0	0	0	0	0	0.680
cs1c1	0	0	0	0	0	0	0	0	0	0
cs1c2	0	0	0	0	0	0	0	0	0	0
cs1c3	0	0	0	0	0	0	0	0	0	0
cs1c4	0	0	0	0	0	0	0	0	0	0
cs1c5	0	0	0	0	0	0	0	0	0	0
cs1c6	0	0	0	0	0	0	0	0	0	0
cs1c7	0	0	0	0	0	0	0	0	0	0
cs1c8	0	0	0	0	0	0	0	0	0	0

same model that have been produced more recently. However, for making that determination, this is a very small study with only one computer set of older computers, specifically the Dell Optiplex G1XP. And two computer sets of newer computers, the Mac computer dataset and Dell Optiplex 745 computer set.

Table C.3: Continuation of Table C.2

	cs1c1	cs1c2	cs1c3	cs1c4	cs1c5	cs1c6	cs1c7	cs1c8
cs2c1	0	0	0	0	0	0	0	0
cs2c2	0	0	0	0	0	0	0	0
cs2c3	0	0	0	0	0	0	0	0
cs2c4	0	0	0	0	0	0	0	0
cs2c5	0	0	0	0	0	0	0	0
cs2c6	0	0	0	0	0	0	0	0
cs2c7	0	0	0	0	0	0	0	0
cs2c8	0	0	0	0	0	0	0	0
cs2c9	0	0	0	0	0	0	0	0
cs2c10	0	0	0	0	0	0	0	0
cs3c1	0	0	0	0	0	0	0	0
cs3c2	0	0	0	0	0	0	0	0
cs3c3	0	0	0	0	0	0	0	0
cs3c4	0	0	0	0	0	0	0	0
cs3c5	0	0	0	0	0	0	0	0
cs3c6	0	0	0	0	0	0	0	0
cs3c7	0	0	0	0	0	0	0	0
cs3c8	0	0	0	0	0	0	0	0
cs3c9	0	0	0	0	0	0	0	0
cs3c10	0	0	0	0	0	0	0	0
cs1c1	0	0	0	0	0	0	0	0
cs1c2	0	0	0	0	0	0	0	0
cs1c3	0	0	0.120	0	0	0	0	0
cs1c4	0	0	0	0	0	0	0	0
cs1c5	0	0	0	0	0.040	0	0	0
cs1c6	0	0	0	0	0	0	0	0
cs1c7	0	0	0	0	0	0	0	0
cs1c8	0	0	0	0	0	0	0	0

Table C.4: IMAM Ratio Matrix Result from Highest Performing Techniques

	cs2c1	cs2c2	cs2c3	cs2c4	cs2c5	cs2c6	cs2c7	cs2c8	cs2c9	cs2c10
cs2c1	–	1.844	13.128	23.395	23.081	13.265	10.646	4.957	7.184	4.823
cs2c2	2.709	–	6.146	14.644	22.164	13.433	8.799	5.651	9.453	4.465
cs2c3	10.685	6.115	–	11.016	48.270	28.701	17.099	18.383	27.103	11.537
cs2c4	21.578	17.907	11.133	–	81.343	58.744	41.528	21.155	40.775	9.835
cs2c5	15.693	18.062	39.873	62.201	–	4.648	14.374	17.331	6.771	29.408
cs2c6	11.535	12.415	27.226	51.490	5.124	–	4.318	16.897	9.165	23.642
cs2c7	8.516	8.435	16.386	36.656	17.309	4.140	–	14.223	12.505	16.185
cs2c8	4.683	6.595	20.618	21.419	26.088	20.995	17.147	–	5.544	3.955
cs2c9	5.522	8.893	26.709	35.530	8.953	9.215	11.695	4.119	–	9.683
cs2c10	4.217	4.661	10.780	8.783	33.390	22.883	16.290	3.318	10.265	–
cs3c1	2804.779	2885.839	2913.562	2983.154	2686.331	2699.591	2751.442	2830.145	2746.884	2843.940
cs3c2	2536.827	2609.298	2638.488	2702.500	2419.683	2436.732	2488.815	2559.329	2479.928	2574.298
cs3c3	1246.797	1273.686	1236.626	1228.754	1302.050	1263.007	1242.853	1256.841	1265.703	1233.125
cs3c4	4477.403	4571.236	4439.555	4410.854	4675.289	4533.408	4461.175	4510.950	4541.870	4426.867
cs3c5	698.534	718.021	718.790	732.867	680.193	678.092	686.671	704.282	688.663	704.631
cs3c6	3605.670	3709.066	3734.053	3818.869	3469.903	3477.647	3539.599	3638.070	3538.390	3650.776
cs3c7	2209.479	2253.552	2182.272	2163.855	2319.598	2243.934	2204.441	2226.476	2246.876	2180.832
cs3c8	772.112	789.581	768.982	764.721	801.968	779.856	769.082	777.921	781.312	764.111
cs3c9	1316.422	1354.081	1366.944	1399.745	1260.367	1266.428	1291.605	1327.743	1288.930	1335.256
cs3c10	572.136	584.420	569.165	567.933	592.933	576.875	569.425	576.899	578.688	566.943
cs1c1	43580.942	43826.433	43101.778	42724.197	44820.045	44128.986	43714.794	43675.228	44071.165	43224.087
cs1c2	62782.963	63085.345	62067.973	61489.112	64581.247	63623.540	63029.091	62896.386	63483.791	62252.495
cs1c3	25713.934	25828.636	25418.061	25180.626	26447.569	26062.450	25820.012	25758.973	26000.042	25498.035
cs1c4	50524.071	50783.659	49936.684	49513.989	51977.905	51147.647	50669.829	50644.362	51109.300	50115.464
cs1c5	31864.975	32028.948	31506.821	31201.570	32793.539	32302.672	31998.286	31920.623	32222.548	31590.342
cs1c6	48756.568	49046.135	48244.105	47765.584	50161.932	49422.783	48965.825	48830.899	49282.080	48334.975
cs1c7	59221.126	59504.773	58570.128	58004.115	60901.448	60037.632	59487.487	59312.608	59863.985	58725.070
cs1c8	49510.431	49751.576	48956.670	48506.503	50916.388	50166.222	49700.381	49601.066	50058.079	49095.969

Table C.5: Continuation of Table C.4

	cs3c1	cs3c2	cs3c3	cs3c4	cs3c5	cs3c6	cs3c7	cs3c8	cs3c9	cs3c10
cs2c1	4212.298	4400.286	4227.086	4240.117	3930.532	4203.726	4407.438	4235.275	4199.462	3980.885
cs2c2	3400.640	3551.162	3382.786	3394.211	3171.504	3394.578	3520.938	3393.800	3392.249	3188.774
cs2c3	3860.545	4033.380	3704.854	3720.848	3575.525	3844.462	3846.619	3727.075	3850.131	3506.987
cs2c4	4227.594	4421.737	3925.772	3942.499	3896.174	4204.643	4067.259	3952.116	4218.039	3731.331
cs2c5	3062.325	3188.904	3331.086	3336.774	2903.643	3069.802	3485.817	3323.255	3054.191	3109.283
cs2c6	3213.091	3349.639	3405.073	3412.476	3028.901	3215.121	3559.802	3401.669	3205.088	3186.398
cs2c7	3481.995	3636.661	3552.544	3562.488	3259.846	3478.745	3706.465	3555.973	3473.887	3340.824
cs2c8	4186.017	4374.521	4196.509	4207.857	3909.206	4180.780	4368.985	4202.024	4175.289	3954.799
cs2c9	3476.682	3627.776	3612.598	3620.341	3267.025	3476.550	3771.650	3609.896	3466.682	3389.801
cs2c10	3475.085	3633.257	3397.761	3409.076	3228.915	3465.375	3534.659	3408.057	3467.159	3208.456
cs3c1	–	5.040	936.127	910.503	43.411	3.489	1105.114	811.932	3.613	716.869
cs3c2	5.958	–	925.844	902.723	69.451	12.553	1078.425	808.485	9.451	727.115
cs3c3	392.360	446.783	–	0.442	229.225	353.187	6.692	2.880	380.522	5.931
cs3c4	1406.290	1602.337	1.203	–	825.200	1266.889	24.725	6.892	1365.012	22.380
cs3c5	11.088	19.148	156.072	151.167	–	7.356	193.549	132.690	9.523	109.154
cs3c6	2.557	10.978	1081.700	1050.961	36.767	–	1287.007	928.883	3.434	820.123
cs3c7	798.618	897.969	7.729	10.494	495.511	725.751	–	20.856	776.739	38.384
cs3c8	209.876	240.783	4.023	2.931	118.539	187.166	11.163	–	204.257	3.151
cs3c9	2.164	4.394	437.954	426.477	19.462	2.940	517.437	382.661	–	335.815
cs3c10	151.999	176.204	5.388	4.965	82.113	135.290	14.625	4.878	147.257	–
cs1c1	39493.281	39903.351	31236.895	31378.182	37571.512	38845.073	30448.353	31336.699	39392.859	32294.488
cs1c2	58377.627	59007.112	46251.885	46496.155	55575.855	57476.304	45128.558	46487.725	58240.057	47803.236
cs1c3	24216.116	24476.900	19247.755	19354.000	23071.147	23853.823	18793.861	19357.301	24160.057	19881.746
cs1c4	46227.912	46722.473	36468.044	36645.255	43959.234	45478.860	35548.533	36614.252	46107.435	37716.439
cs1c5	29618.802	29943.776	23400.583	23521.476	28179.348	29154.610	22823.368	23508.584	29546.980	24197.293
cs1c6	44401.621	44873.954	35114.951	35271.949	42244.365	43683.040	34231.153	35220.829	44298.038	36308.147
cs1c7	55405.991	55991.823	44110.103	44339.440	52803.255	54575.505	43062.626	44334.820	55277.472	45558.254
cs1c8	45969.312	46460.105	36519.843	36705.599	43789.419	45265.004	35638.389	36691.990	45862.365	37732.097

Table C.6: Continuation of Table C.5

cs2c1	51803.228	58919.409	63178.819	53786.228	56228.313	51214.530	62104.752	59287.202
cs2c2	40708.454	46251.949	49578.751	42250.251	44155.576	40248.701	48734.373	46537.332
cs2c3	45075.197	51219.796	54914.033	46775.762	48889.633	44562.730	53980.531	51541.890
cs2c4	47688.521	54191.235	58109.936	49486.996	51716.378	47140.456	57123.297	54539.129
cs2c5	39974.900	45453.869	48724.061	41506.918	43399.204	39531.866	47888.027	45725.131
cs2c6	41558.922	47286.763	50708.859	43161.362	45137.010	41089.484	49827.342	47562.829
cs2c7	43690.053	49706.894	53312.877	45362.922	47443.704	43195.638	52394.387	50009.813
cs2c8	51008.080	58010.790	62206.402	52958.298	55363.576	50426.977	61134.215	58359.393
cs2c9	43915.659	49961.691	53568.502	45605.707	47682.662	43424.792	52657.511	50268.088
cs2c10	41640.281	47357.782	50781.627	43230.906	45191.097	41167.318	49925.048	47657.400
cs3c1	31357.934	36572.045	39683.446	32849.171	34868.930	31144.603	38810.162	36761.979
cs3c2	27094.027	31620.169	34315.200	28393.604	30153.799	26917.230	33553.310	31781.231
cs3c3	11124.448	13001.202	14168.960	11625.499	12369.722	11046.021	13862.056	13101.836
cs3c4	39461.179	46127.944	50271.894	41244.456	43874.019	39177.565	49185.038	46486.772
cs3c5	8123.730	9469.085	10277.983	8503.342	9026.060	8069.139	10055.167	9524.189
cs3c6	39294.457	45882.294	49819.374	41177.592	43732.042	39030.059	48715.099	46125.835
cs3c7	18273.732	21371.475	23304.325	19096.234	20321.962	18146.228	22798.205	21540.876
cs3c8	6801.343	7968.274	8689.288	7114.639	7575.471	6753.332	8497.797	8027.289
cs3c9	14883.879	17349.552	18820.735	15587.289	16544.137	14786.188	18408.107	17440.078
cs3c10	5670.751	6609.587	7189.963	5923.317	6294.070	5632.071	7038.851	6659.811
cs1c1	–	385.328	894.979	60.513	205.291	18.281	686.839	381.514
cs1c2	487.682	–	139.271	209.284	60.514	508.082	68.252	19.852
cs1c3	440.500	55.827	–	262.898	126.496	449.807	13.638	59.742
cs1c4	68.390	186.881	596.956	–	72.087	82.515	417.240	178.149
cs1c5	149.304	31.116	166.670	49.273	–	157.526	102.152	30.625
cs1c6	21.144	456.621	1037.003	83.833	247.615	–	808.889	454.814
cs1c7	779.678	61.531	24.144	426.682	178.239	809.600	–	52.252
cs1c8	380.901	16.114	116.969	160.205	46.317	400.392	46.247	–

Table C.7: Confusion Matrix Result from using Noise Spectra Scaling

	cs2c1	cs2c2	cs2c3	cs2c4	cs2c5	cs2c6	cs2c7	cs2c8	cs2c9	cs2c10
cs2c1	0	0.053	0	0	0	0	0	0	0	0
cs2c2	0	0	0	0	0	0	0	0	0	0
cs2c3	0	0	0	0	0	0	0	0	0	0
cs2c4	0	0	0	0	0	0	0	0	0	0
cs2c5	0	0	0	0	0	0	0	0	0	0
cs2c6	0	0	0	0	0	0	0	0	0	0
cs2c7	0	0	0	0	0	0	0	0	0	0
cs2c8	0	0	0	0	0	0	0	0	0	0
cs2c9	0	0	0	0	0	0	0	0	0	0
cs2c10	0	0	0	0	0	0	0	0	0	0
cs3c1	0	0	0	0	0	0	0	0	0	0
cs3c2	0	0	0	0	0	0	0	0	0	0
cs3c3	0	0	0	0	0	0	0	0	0	0
cs3c4	0	0	0	0	0	0	0	0	0	0
cs3c5	0	0	0	0	0	0	0	0	0	0
cs3c6	0	0	0	0	0	0	0	0	0	0
cs3c7	0	0	0	0	0	0	0	0	0	0
cs3c8	0	0	0	0	0	0	0	0	0	0
cs3c9	0	0	0	0	0	0	0	0	0	0
cs3c10	0	0	0	0	0	0	0	0	0	0
cs1c1	0	0	0	0	0	0	0	0	0	0
cs1c2	0	0	0	0	0	0	0	0	0	0
cs1c3	0	0	0	0	0	0	0	0	0	0
cs1c4	0	0	0	0	0	0	0	0	0	0
cs1c5	0	0	0	0	0	0	0	0	0	0
cs1c6	0	0	0	0	0	0	0	0	0	0
cs1c7	0	0	0	0	0	0	0	0	0	0
cs1c8	0	0	0	0	0	0	0	0	0	0

Table C.8: Continuation of Table C.7

	cs3c1	cs3c2	cs3c3	cs3c4	cs3c5	cs3c6	cs3c7	cs3c8	cs3c9	cs3c10
cs2c1	0	0	0	0	0	0	0	0	0	0
cs2c2	0	0	0	0	0	0	0	0	0	0
cs2c3	0	0	0	0	0	0	0	0	0	0
cs2c4	0	0	0	0	0	0	0	0	0	0
cs2c5	0	0	0	0	0	0	0	0	0	0
cs2c6	0	0	0	0	0	0	0	0	0	0
cs2c7	0	0	0	0	0	0	0	0	0	0
cs2c8	0	0	0	0	0	0	0	0	0	0
cs2c9	0	0	0	0	0	0	0	0	0	0
cs2c10	0	0	0	0	0	0	0	0	0	0
cs3c1	0	0	0	0	0	0	0	0	0	0
cs3c2	0	0.027	0	0	0	0	0	0	0	0
cs3c3	0	0	0.107	0.880	0	0	0	0	0	0
cs3c4	0	0	0.640	0	0	0	0	0	0	0
cs3c5	0	0	0	0	0.600	0	0	0	0	0
cs3c6	0	0	0	0	0	0	0	0	0	0
cs3c7	0	0	0	0	0	0	0.040	0	0	0
cs3c8	0	0	0	0	0	0	0	0.907	0	0
cs3c9	0	0	0	0	0	0	0	0	0.667	0
cs3c10	0	0	0	0	0	0	0	0	0	0.720
cs1c1	0	0	0	0	0	0	0	0	0	0
cs1c2	0	0	0	0	0	0	0	0	0	0
cs1c3	0	0	0	0	0	0	0	0	0	0
cs1c4	0	0	0	0	0	0	0	0	0	0
cs1c5	0	0	0	0	0	0	0	0	0	0
cs1c6	0	0	0	0	0	0	0	0	0	0
cs1c7	0	0	0	0	0	0	0	0	0	0
cs1c8	0	0	0	0	0	0	0	0	0	0

Table C.9: Continuation of Table C.8

	cs1c1	cs1c2	cs1c3	cs1c4	cs1c5	cs1c6	cs1c7	cs1c8
cs2c1	0	0	0	0	0	0	0	0
cs2c2	0	0	0	0	0	0	0	0
cs2c3	0	0	0	0	0	0	0	0
cs2c4	0	0	0	0	0	0	0	0
cs2c5	0	0	0	0	0	0	0	0
cs2c6	0	0	0	0	0	0	0	0
cs2c7	0	0	0	0	0	0	0	0
cs2c8	0	0	0	0	0	0	0	0
cs2c9	0	0	0	0	0	0	0	0
cs2c10	0	0	0	0	0	0	0	0
cs3c1	0	0	0	0	0	0	0	0
cs3c2	0	0	0	0	0	0	0	0
cs3c3	0	0	0	0	0	0	0	0
cs3c4	0	0	0	0	0	0	0	0
cs3c5	0	0	0	0	0	0	0	0
cs3c6	0	0	0	0	0	0	0	0
cs3c7	0	0	0	0	0	0	0	0
cs3c8	0	0	0	0	0	0	0	0
cs3c9	0	0	0	0	0	0	0	0
cs3c10	0	0	0	0	0	0	0	0
cs1c1	0	0	0	0	0	0	0	0
cs1c2	0	0	0	0	0	0	0	0
cs1c3	0	0	0.267	0	0	0	0	0
cs1c4	0	0	0	0	0	0	0	0
cs1c5	0	0	0	0	0.080	0	0	0
cs1c6	0	0	0	0	0	0	0	0
cs1c7	0	0	0	0	0	0	0	0
cs1c8	0	0	0	0	0	0	0	0

Table C.10: IMAM Ratio Matrix Result from using Noise Spectra Scaling

	cs2c1	cs2c2	cs2c3	cs2c4	cs2c5	cs2c6	cs2c7	cs2c8	cs2c9	cs2c10
cs2c1	–	1.865	9.837	18.827	18.254	10.370	7.839	3.548	5.483	4.061
cs2c2	2.327	–	5.102	13.543	18.863	10.378	6.900	3.843	7.144	3.742
cs2c3	6.513	4.123	–	6.782	34.737	18.946	10.945	10.342	17.082	6.406
cs2c4	15.231	13.828	7.869	–	60.774	41.594	29.100	15.728	30.010	7.221
cs2c5	12.973	12.418	29.631	46.329	–	4.117	12.022	13.499	5.613	22.382
cs2c6	8.004	8.318	18.791	35.497	3.736	–	3.366	11.297	6.071	16.458
cs2c7	5.829	6.048	11.721	26.323	12.989	3.345	–	9.688	8.510	11.542
cs2c8	3.063	4.032	12.327	14.073	17.366	12.970	10.318	–	3.848	2.829
cs2c9	4.291	6.518	19.686	26.471	6.970	7.130	8.970	3.093	–	7.551
cs2c10	3.264	3.655	7.033	6.865	27.638	17.117	11.753	2.586	8.352	–
cs3c1	2155.117	2219.669	2239.728	2301.769	2059.260	2066.178	2107.957	2176.617	2108.966	2189.020
cs3c2	2052.445	2112.004	2118.911	2166.558	1985.263	1984.756	2015.661	2071.808	2019.338	2077.047
cs3c3	1113.001	1137.247	1103.731	1099.299	1161.975	1126.931	1107.592	1123.067	1130.444	1101.978
cs3c4	2813.233	2872.288	2770.634	2735.948	2977.455	2869.183	2809.440	2835.432	2869.121	2769.672
cs3c5	528.033	542.404	538.702	547.566	521.992	516.201	519.162	532.183	523.477	530.725
cs3c6	2602.875	2673.780	2682.894	2743.400	2516.732	2516.599	2555.659	2626.108	2559.910	2632.449
cs3c7	1884.800	1921.785	1850.867	1827.788	1996.257	1924.251	1884.483	1900.157	1925.505	1856.328
cs3c8	797.537	817.480	791.604	784.189	837.292	810.357	795.867	804.430	810.801	787.402
cs3c9	965.376	994.669	1002.061	1027.719	927.575	928.818	945.561	975.057	947.478	979.578
cs3c10	438.297	448.822	434.565	431.827	459.157	444.073	436.623	442.220	445.048	432.874
cs1c1	41392.320	41629.214	40880.378	40537.283	42640.647	41944.848	41511.227	41519.258	41902.202	41048.597
cs1c2	39248.630	39426.362	38759.452	38395.087	40433.040	39805.934	39411.961	39335.047	39715.821	38906.120
cs1c3	26749.826	26879.932	26434.338	26200.291	27522.754	27113.334	26851.170	26814.908	27056.673	26531.411
cs1c4	40086.052	40299.808	39603.793	39304.039	41238.758	40561.966	40165.288	40214.362	40566.045	39774.919
cs1c5	29853.296	30016.008	29501.313	29223.780	30748.292	30274.309	29973.290	29925.556	30205.672	29595.617
cs1c6	35451.197	35653.785	35052.184	34708.096	36504.029	35947.954	35608.180	35517.360	35848.968	35142.991
cs1c7	53306.921	53579.646	52703.082	52207.636	54849.343	54055.817	53538.029	53421.381	53908.532	52866.706
cs1c8	39070.712	39264.662	38615.360	38273.740	40200.292	39591.646	39216.381	39160.214	39515.094	38745.218

Table C.11: Continuation of Table C.10

	cs3c1	cs3c2	cs3c3	cs3c4	cs3c5	cs3c6	cs3c7	cs3c8	cs3c9	cs3c10
cs2c1	3265.385	3425.694	3110.690	3122.740	3002.872	3250.447	3239.944	3122.806	3252.608	2936.888
cs2c2	2599.384	2721.158	2794.849	2802.551	2419.972	2606.191	2938.790	2812.118	2587.021	2557.835
cs2c3	2434.225	2543.073	2638.072	2646.252	2272.889	2438.225	2775.069	2652.540	2422.876	2415.430
cs2c4	2919.069	3042.208	3063.604	3070.520	2745.028	2918.839	3201.295	3070.560	2911.707	2851.900
cs2c5	2138.509	2202.949	2821.218	2822.059	2111.967	2168.953	2982.181	2806.681	2131.658	2566.512
cs2c6	2151.577	2236.936	2475.907	2480.808	2053.087	2162.946	2602.305	2472.990	2144.338	2282.590
cs2c7	2564.666	2671.033	2810.224	2815.378	2438.593	2572.990	2941.958	2806.682	2558.983	2618.964
cs2c8	2671.718	2792.328	2764.867	2772.824	2502.041	2673.728	2888.385	2771.011	2663.824	2583.045
cs2c9	2620.865	2725.381	3013.339	3015.711	2523.335	2635.782	3162.542	3002.742	2613.649	2789.431
cs2c10	2512.584	2634.645	2551.689	2560.738	2318.704	2509.373	2674.599	2568.993	2500.819	2360.538
cs3c1	–	4.033	758.547	738.017	34.216	2.794	894.625	663.914	2.532	577.058
cs3c2	4.784	–	799.166	780.764	56.101	10.434	937.068	711.189	7.368	614.290
cs3c3	352.287	401.506	–	0.527	206.296	317.130	6.040	2.841	341.870	5.512
cs3c4	794.319	899.804	0.943	–	470.951	711.303	14.800	4.032	771.830	13.285
cs3c5	7.870	13.640	122.517	118.812	–	5.184	152.993	106.594	6.732	83.886
cs3c6	2.137	7.966	759.210	737.966	25.764	–	905.096	653.488	2.651	573.937
cs3c7	612.201	684.844	5.799	7.809	382.137	554.146	–	14.995	596.280	29.060
cs3c8	219.286	250.293	3.929	2.723	125.430	194.840	10.800	–	213.911	3.323
cs3c9	1.549	3.160	357.442	348.241	15.427	2.394	423.759	318.139	–	269.112
cs3c10	117.659	136.102	4.685	4.299	63.571	104.142	12.274	4.209	114.193	–
cs1c1	37415.395	37805.670	29352.961	29491.674	35562.959	36789.812	28582.155	29461.823	37325.289	30376.116
cs1c2	37193.635	37630.332	29034.500	29210.148	35295.369	36606.157	28301.611	29227.309	37101.691	30064.494
cs1c3	24741.092	24990.311	19728.001	19826.386	23607.895	24373.652	19255.603	19822.708	24690.764	20370.861
cs1c4	35997.359	36344.893	28517.294	28640.475	34289.918	35414.346	27787.939	28610.337	35909.999	29478.943
cs1c5	27583.388	27878.735	21712.406	21822.994	26247.630	27148.734	21159.219	21814.111	27523.518	22466.141
cs1c6	32499.343	32866.335	25434.064	25557.693	30845.828	31961.349	24778.510	25532.749	32420.383	26328.610
cs1c7	49339.452	49839.582	39252.553	39445.175	47053.289	48598.347	38300.007	39438.551	49236.597	40545.182
cs1c8	36180.736	36562.769	28651.929	28797.907	34453.148	35623.145	27946.598	28790.735	36100.028	29617.236

Table C.12: Continuation of Table C.11

	cs1c1	cs1c2	cs1c3	cs1c4	cs1c5	cs1c6	cs1c7	cs1c8
cs2c1	38326.461	43882.871	47169.558	39878.286	41802.608	37916.530	46308.437	44108.744
cs2c2	45937.876	52235.275	55882.789	47649.235	49817.167	45511.258	55003.065	52554.993
cs2c3	44088.224	50197.363	53710.326	45749.340	47837.685	43682.303	52867.892	50510.461
cs2c4	43411.570	49382.500	52852.139	45050.908	47067.760	42959.487	52012.873	49689.991
cs2c5	40978.916	46426.874	49556.006	42467.288	44310.001	40578.386	48823.424	46723.860
cs2c6	35137.920	39993.949	42805.345	36470.113	38135.201	34791.396	42108.382	40211.089
cs2c7	36767.943	41840.656	44807.961	38158.938	39892.100	36380.720	44071.150	42070.602
cs2c8	37194.193	42354.892	45381.597	38605.232	40377.234	36812.255	44620.957	42574.748
cs2c9	38756.267	43950.583	46984.061	40175.499	41924.782	38338.124	46257.211	44192.016
cs2c10	40194.118	45802.182	49062.345	41721.548	43656.152	39812.781	48274.737	46081.454
cs3c1	26655.747	31164.052	33815.531	27938.107	29668.411	26488.624	33072.367	31303.078
cs3c2	28195.860	32800.716	35459.773	29491.242	31218.950	28019.101	34749.924	32970.319
cs3c3	9910.784	11643.236	12695.839	10369.834	11049.386	9850.060	12413.655	11708.042
cs3c4	27391.144	31942.522	34658.297	28580.920	30300.384	27198.546	33992.931	32173.318
cs3c5	7305.865	8485.047	9176.842	7630.741	8074.806	7256.561	8996.451	8533.966
cs3c6	27854.786	32561.276	35336.906	29184.416	30998.356	27679.109	34561.151	32701.058
cs3c7	15865.541	18562.939	20190.673	16569.556	17607.276	15765.598	19781.286	18697.910
cs3c8	8022.078	9406.987	10236.958	8391.135	8920.776	7966.086	10023.400	9471.630
cs3c9	13791.213	16075.650	17405.205	14435.487	15300.665	13705.095	17042.449	16154.750
cs3c10	5160.507	6034.266	6554.719	5393.932	5727.281	5126.844	6422.274	6074.087
cs1c1	–	341.938	795.399	51.388	177.986	14.640	616.833	341.269
cs1c2	323.925	–	93.130	139.165	37.126	335.069	44.563	11.917
cs1c3	398.809	49.172	–	240.559	115.306	412.956	10.775	50.902
cs1c4	46.146	132.487	419.467	–	48.053	55.712	300.332	130.401
cs1c5	123.577	25.159	144.007	39.842	–	129.571	90.876	26.171
cs1c6	14.983	347.150	787.109	63.494	188.282	–	612.049	343.347
cs1c7	638.267	49.644	16.928	357.631	148.999	665.403	–	41.594
cs1c8	292.327	11.276	87.791	124.641	35.194	307.864	35.183	–

Table C.13: Confusion Matrix Result from Aggressive Low Pass Filtering

	cs2c1	cs2c2	cs2c3	cs2c4	cs2c5	cs2c6	cs2c7	cs2c8	cs2c9	cs2c10
cs2c1	0	0	0	0	0	0	0	0	0	0
cs2c2	0	0	0	0	0	0	0	0	0	0
cs2c3	0	0	0	0	0	0	0	0	0	0
cs2c4	0	0	0	0	0	0	0	0	0	0
cs2c5	0	0	0	0	0	0	0	0	0	0
cs2c6	0	0	0	0	0	0	0	0	0	0
cs2c7	0	0	0	0	0	0	0	0	0	0
cs2c8	0	0	0	0	0	0	0	0	0	0
cs2c9	0	0	0	0	0	0	0	0	0	0
cs2c10	0	0	0	0	0	0	0	0	0	0
cs3c1	0	0	0	0	0	0	0	0	0	0
cs3c2	0	0	0	0	0	0	0	0	0	0
cs3c3	0	0	0	0	0	0	0	0	0	0
cs3c4	0	0	0	0	0	0	0	0	0	0
cs3c5	0	0	0	0	0	0	0	0	0	0
cs3c6	0	0	0	0	0	0	0	0	0	0
cs3c7	0	0	0	0	0	0	0	0	0	0
cs3c8	0	0	0	0	0	0	0	0	0	0
cs3c9	0	0	0	0	0	0	0	0	0	0
cs3c10	0	0	0	0	0	0	0	0	0	0
cs1c1	0	0	0	0	0	0	0	0	0	0
cs1c2	0	0	0	0	0	0	0	0	0	0
cs1c3	0	0	0	0	0	0	0	0	0	0
cs1c4	0	0	0	0	0	0	0	0	0	0
cs1c5	0	0	0	0	0	0	0	0	0	0
cs1c6	0	0	0	0	0	0	0	0	0	0
cs1c7	0	0	0	0	0	0	0	0	0	0
cs1c8	0	0	0	0	0	0	0	0	0	0

Table C.14: Continuation of Table C.13

	cs3c1	cs3c2	cs3c3	cs3c4	cs3c5	cs3c6	cs3c7	cs3c8	cs3c9	cs3c10
cs2c1	0	0	0	0	0	0	0	0	0	0
cs2c2	0	0	0	0	0	0	0	0	0	0
cs2c3	0	0	0	0	0	0	0	0	0	0
cs2c4	0	0	0	0	0	0	0	0	0	0
cs2c5	0	0	0	0	0	0	0	0	0	0
cs2c6	0	0	0	0	0	0	0	0	0	0
cs2c7	0	0	0	0	0	0	0	0	0	0
cs2c8	0	0	0	0	0	0	0	0	0	0
cs2c9	0	0	0	0	0	0	0	0	0	0
cs2c10	0	0	0	0	0	0	0	0	0	0
cs3c1	0	0	0	0	0	0	0	0	0	0
cs3c2	0	0	0	0	0	0	0	0	0	0
cs3c3	0	0	0.133	0.800	0	0	0	0	0	0
cs3c4	0	0	0.133	0	0	0	0	0	0	0
cs3c5	0	0	0	0	0.587	0	0	0	0	0
cs3c6	0	0	0	0	0	0	0	0	0	0
cs3c7	0	0	0	0	0	0	0.040	0	0	0
cs3c8	0	0	0	0	0	0	0	0.907	0	0
cs3c9	0	0	0	0	0	0	0	0	0.587	0
cs3c10	0	0	0	0	0	0	0	0	0	0.667
cs1c1	0	0	0	0	0	0	0	0	0	0
cs1c2	0	0	0	0	0	0	0	0	0	0
cs1c3	0	0	0	0	0	0	0	0	0	0
cs1c4	0	0	0	0	0	0	0	0	0	0
cs1c5	0	0	0	0	0	0	0	0	0	0
cs1c6	0	0	0	0	0	0	0	0	0	0
cs1c7	0	0	0	0	0	0	0	0	0	0
cs1c8	0	0	0	0	0	0	0	0	0	0

Table C.15: Continuation of Table C.14

	cs1c1	cs1c2	cs1c3	cs1c4	cs1c5	cs1c6	cs1c7	cs1c8
cs2c1	0	0	0	0	0	0	0	0
cs2c2	0	0	0	0	0	0	0	0
cs2c3	0	0	0	0	0	0	0	0
cs2c4	0	0	0	0	0	0	0	0
cs2c5	0	0	0	0	0	0	0	0
cs2c6	0	0	0	0	0	0	0	0
cs2c7	0	0	0	0	0	0	0	0
cs2c8	0	0	0	0	0	0	0	0
cs2c9	0	0	0	0	0	0	0	0
cs2c10	0	0	0	0	0	0	0	0
cs3c1	0	0	0	0	0	0	0	0
cs3c2	0	0	0	0	0	0	0	0
cs3c3	0	0	0	0	0	0	0	0
cs3c4	0	0	0	0	0	0	0	0
cs3c5	0	0	0	0	0	0	0	0
cs3c6	0	0	0	0	0	0	0	0
cs3c7	0	0	0	0	0	0	0	0
cs3c8	0	0	0	0	0	0	0	0
cs3c9	0	0	0	0	0	0	0	0
cs3c10	0	0	0	0	0	0	0	0
cs1c1	0	0	0	0	0	0	0	0
cs1c2	0	0	0	0	0	0	0	0
cs1c3	0	0	0.133	0	0	0	0	0
cs1c4	0	0	0	0	0	0	0	0
cs1c5	0	0	0	0	0.040	0	0	0
cs1c6	0	0	0	0	0	0	0	0
cs1c7	0	0	0	0	0	0	0	0
cs1c8	0	0	0	0	0	0	0	0

Table C.16: IMAM Ratio Matrix Result from Aggressive Low Pass Filtering

cs2c1	–	3.044	16.873	28.964	29.758	16.166	13.074	4.656	7.261	4.891
cs2c2	3.656	–	8.348	20.555	32.482	18.969	12.498	7.278	12.312	5.242
cs2c3	13.199	7.251	–	11.460	61.542	35.753	21.126	21.720	32.452	11.565
cs2c4	28.200	23.289	12.590	–	113.729	81.108	56.342	28.600	57.144	13.303
cs2c5	20.260	23.449	52.161	81.817	–	5.562	18.345	22.584	8.128	37.485
cs2c6	13.591	15.013	33.303	64.088	5.740	–	5.061	20.337	10.359	28.913
cs2c7	13.691	13.468	27.277	61.955	28.998	6.357	–	22.672	19.206	26.668
cs2c8	4.800	7.422	25.814	28.439	35.072	27.379	21.774	–	7.016	4.381
cs2c9	5.570	11.015	35.536	50.784	11.471	12.000	15.546	5.001	–	13.414
cs2c10	4.055	4.753	11.176	10.649	41.145	27.755	19.405	3.268	12.215	–
cs3c1	3412.790	3511.201	3542.853	3629.146	3266.759	3282.176	3347.117	3441.142	3340.395	3461.275
cs3c2	3021.745	3107.804	3140.693	3219.101	2880.270	2900.913	2964.048	3045.957	2952.076	3066.779
cs3c3	1473.984	1505.817	1461.117	1452.526	1538.488	1493.221	1468.928	1485.202	1495.796	1458.087
cs3c4	5774.920	5896.223	5721.859	5688.661	6027.989	5845.814	5751.931	5816.143	5856.672	5711.533
cs3c5	740.666	761.388	761.672	777.043	720.891	718.628	727.826	746.386	729.794	747.443
cs3c6	4890.537	5030.670	5061.099	5179.296	4703.764	4713.117	4799.906	4932.124	4797.097	4953.119
cs3c7	2539.394	2590.489	2506.624	2486.865	2664.621	2578.921	2532.868	2557.778	2581.475	2507.192
cs3c8	808.672	827.041	804.946	800.986	839.661	816.872	805.493	814.437	818.064	800.532
cs3c9	1512.037	1554.965	1569.098	1607.573	1446.556	1453.844	1483.251	1523.782	1479.540	1533.939
cs3c10	599.660	612.683	596.341	595.207	621.127	604.713	596.832	604.351	606.340	594.415
cs1c1	71763.148	72164.931	70967.835	70353.333	73802.805	72660.717	71976.595	71916.601	72566.696	71177.270
cs1c2	92910.089	93357.779	91845.195	90993.258	95570.295	94151.269	93274.070	93075.695	93943.605	92129.574
cs1c3	29938.041	30070.273	29591.070	29316.005	30791.105	30342.092	30060.373	29988.507	30270.177	29687.557
cs1c4	74853.179	75237.911	73979.727	73357.464	77002.886	75776.926	75062.482	75032.685	75722.663	74255.208
cs1c5	37718.191	37911.331	37291.945	36931.740	38816.298	38235.362	37876.093	37782.584	38140.892	37394.674
cs1c6	73369.591	73803.468	72596.585	71876.899	75483.255	74370.120	73684.302	73478.148	74159.219	72739.263
cs1c7	86279.425	86689.318	85328.250	84504.057	88724.954	87469.177	86669.870	86410.138	87214.626	85560.257
cs1c8	71227.174	71573.068	70424.269	69783.052	73247.576	72166.363	71497.330	71357.185	72013.118	70633.078

Table C.17: Continuation of Table C.16

	cs3c1	cs3c2	cs3c3	cs3c4	cs3c5	cs3c6	cs3c7	cs3c8	cs3c9	cs3c10
cs2c1	5654.428	5905.763	5675.080	5691.586	5275.689	5643.256	5915.946	5685.721	5637.961	5344.527
cs2c2	5096.980	5321.873	5070.310	5086.780	4753.480	5087.434	5277.247	5087.363	5084.673	4779.483
cs2c3	5010.492	5234.659	4808.789	4828.879	4640.470	4989.850	4992.371	4837.555	4997.320	4551.586
cs2c4	5969.001	6243.043	5542.883	5566.141	5500.538	5936.716	5742.439	5580.567	5956.323	5268.468
cs2c5	4073.925	4241.503	4432.224	4438.826	3861.349	4083.916	4637.144	4421.548	4063.030	4136.350
cs2c6	4069.621	4242.265	4313.453	4322.234	3835.548	4072.382	4508.788	4309.855	4059.688	4036.098
cs2c7	6066.659	6335.407	6189.965	6206.831	5678.793	6061.570	6457.991	6197.312	6052.641	5820.616
cs2c8	5744.223	6002.368	5759.572	5773.962	5363.809	5737.419	5995.601	5766.958	5729.839	5427.361
cs2c9	5019.406	5236.964	5216.522	5227.727	4715.738	5019.393	5445.796	5212.996	5005.305	4895.336
cs2c10	4420.957	4621.512	4322.446	4336.640	4107.336	4408.117	4496.668	4335.858	4410.513	4081.910
cs3c1	–	5.953	1140.144	1108.936	52.528	3.962	1345.983	988.686	4.206	872.790
cs3c2	6.990	–	1103.718	1076.156	82.582	14.753	1285.769	963.693	11.097	866.659
cs3c3	464.309	528.694	–	0.410	271.190	417.902	7.799	3.336	450.281	6.908
cs3c4	1815.611	2068.848	1.472	–	1065.129	1635.520	31.377	8.625	1762.417	28.491
cs3c5	11.725	20.273	165.683	160.488	–	7.739	205.457	140.816	10.060	115.830
cs3c6	3.098	14.542	1468.513	1427.081	49.590	–	1747.368	1260.910	4.255	1113.180
cs3c7	918.885	1033.209	8.640	11.920	570.002	834.940	–	23.830	893.702	43.920
cs3c8	220.077	252.479	4.191	3.013	124.246	196.249	11.649	–	214.152	3.237
cs3c9	2.354	4.970	503.573	490.375	22.265	3.245	594.943	439.928	–	386.038
cs3c10	159.482	184.896	5.623	5.159	86.148	141.955	15.290	5.060	154.534	–
cs1c1	65036.506	65712.656	51439.485	51672.553	61872.753	63967.692	50140.838	51605.899	64871.483	53180.908
cs1c2	86397.540	87327.503	68451.599	68811.525	82251.010	85062.549	66789.531	68799.904	86192.674	70747.879
cs1c3	28195.233	28498.533	22410.633	22534.206	26862.193	27773.582	21881.854	22537.929	28129.612	23148.776
cs1c4	68495.488	69227.556	54034.250	54296.148	65134.267	67386.107	52671.351	54250.480	68316.254	55882.661
cs1c5	35062.384	35446.567	27700.845	27844.259	33358.054	34512.576	27017.749	27828.689	34976.894	28644.151
cs1c6	66821.507	67533.787	52845.723	53080.726	63577.717	65740.213	51516.294	53005.242	66666.004	54642.355
cs1c7	80726.938	81581.901	64269.432	64602.996	76935.605	79517.091	62743.459	64596.397	80539.878	66379.646
cs1c8	66137.469	66843.197	52542.171	52808.954	63001.123	65124.561	51273.791	52788.849	65983.628	54286.455

Table C.18: Continuation of Table C.17

	cs1c1	cs1c2	cs1c3	cs1c4	cs1c5	cs1c6	cs1c7	cs1c8
cs2c1	69603.624	79167.973	84890.711	72268.491	75551.971	68812.414	83449.171	79664.009
cs2c2	61068.788	69388.370	74378.208	63384.163	66242.473	60377.489	73110.873	69815.299
cs2c3	58583.441	66571.661	71374.728	60798.114	63544.151	57917.834	70160.609	66991.989
cs2c4	67393.670	76585.637	82125.266	69937.795	73088.109	66618.674	80728.874	77075.621
cs2c5	53254.853	60553.291	64912.804	55295.968	57818.024	52664.892	63795.757	60914.467
cs2c6	52706.460	59974.097	64312.133	54740.384	57247.738	52111.362	63197.419	60322.501
cs2c7	76220.156	86716.043	93010.608	79140.009	82766.570	75355.287	91403.776	87246.789
cs2c8	70097.699	79725.899	85490.879	72780.737	76087.239	69300.662	84017.026	80200.173
cs2c9	63474.566	72214.377	77429.644	65918.195	68919.620	62766.463	76112.077	72659.219
cs2c10	53010.402	60289.691	64647.870	55034.499	57531.047	52408.740	63557.375	60672.801
cs3c1	38202.530	44555.444	48344.182	40019.228	42480.638	37944.105	47284.644	44786.395
cs3c2	32307.771	37705.582	40917.905	33857.817	35956.641	32097.474	40010.465	37897.514
cs3c3	13167.770	15388.535	16770.862	13760.023	14641.050	13074.698	16407.448	15507.487
cs3c4	50958.275	59569.128	64918.311	53260.716	56658.900	50591.520	63517.698	60031.655
cs3c5	8625.673	10054.291	10913.154	9028.804	9584.328	8567.927	10676.673	10112.743
cs3c6	53360.872	62308.820	67653.014	55918.827	59388.337	53003.195	66157.100	62638.321
cs3c7	21028.386	24593.011	26817.233	21975.091	23386.077	20880.521	26236.053	24788.742
cs3c8	7133.316	8356.912	9113.094	7461.696	7945.110	7082.719	8912.355	8418.801
cs3c9	17117.209	19952.691	21644.490	17925.840	19026.972	17004.439	21170.572	20057.508
cs3c10	5952.148	6937.697	7546.701	6217.449	6606.551	5911.675	7388.441	6990.610
cs1c1	–	633.946	1473.261	99.367	337.556	29.638	1130.376	627.736
cs1c2	721.095	–	205.644	309.109	89.103	751.580	100.064	28.583
cs1c3	512.654	64.909	–	305.847	147.077	523.497	15.411	69.262
cs1c4	100.847	276.093	883.707	–	106.276	121.696	617.859	263.391
cs1c5	176.567	36.466	197.053	57.973	–	186.334	120.564	35.812
cs1c6	31.341	686.875	1559.966	125.786	372.116	–	1216.876	683.973
cs1c7	1135.573	88.940	34.198	621.326	259.202	1179.004	–	75.767
cs1c8	547.335	22.767	167.325	230.042	66.305	575.684	66.239	–

BIBLIOGRAPHY

Barrere, W. G., Kaplan, D., and Green, E. R. (1998). Adaptive waveform matching for use in transmitter identification. United States Patent 5,715,518.

Ellis, K. J. and Serinken, N. (2001). Characteristics of radio transmitter fingerprints. *Radio Science*, *36*, pages 585-598.

Erbskorn, J. W. (2009). Detection of intrusions at layer one: A preliminary performance analysis of the IEEE 802.3 normal link pulse as a means of host-to-network authentication and a survey of environmental effects. Master's thesis, Iowa State University, Ames, IA.

Gerdes, R. (2006). Differentiating Ethernet devices through SNR maximization. Master's thesis, Iowa State University, Ames, IA.

Gerdes, R. M., Daniels, T. E., Mina, M., and Russel, S. (2006). Device identification via analog signal fingerprinting: A matched filter approach. In *Proceedings of The 13th Annual Network and Distributed System Security (NDSS) Symposium*, San Diego, California. NDSS.

Graps, A. L. (1995). An Introduction to Wavelets. *IEEE Computational Sciences and Engineering*, *2*(2), pages 50-61.

Institute of Electrical and Electronics Engineers (2005). IEEE Standard 802.3-2005 - Part 3: Carrier Sense Multiple Access With Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications. *IEEE Standard 802.3-2005*. IEEE, New York.

Jackson, E. A. (2006). Detecting intrusions at layer one: device fingerprinting for network access authorization. Master's thesis, Iowa State University, Ames, IA.

Kohno, T., Broido, A., and Claffy, K. C. (2005). Remote physical device fingerprinting. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 211-225. IEEE.

McGill, K. C., and Dorfman, L. J. (1984). High-resolution alighment of sampled waveforms. *IEEE Transactions on Biomedical Engineering*, *31*(6). IEEE.

Payal, Y. (1995). Identification of push-to-talk transmitters using wavelets. Masters thesis, Naval Postgraduate School, Monterey, CA.

Strang, G. (2008). 18.085 Computational Science and Engineering I. *MIT OpenCourseWare*, lectures 24–28. MIT.

Xiao, L., Greenstein, L., Mandayam, N., and Trappe, W. (2007). Finterprints in the ether: Using the physical layer for wireless authentication. In *Proceedings of the IEEE International Conference on Communications (ICC)*, pages 4646–4651, Glasgow, Scotland. IEEE.

Xiao, L., Greenstein, L., Mandayam, N., and Trappe, W. (2008). A physical-layer technique to enhance authentication for mobile terminals. In *Proceedings of the IEEE International Conference on Communications (ICC)*, pages 1520–1524, Beijing, China. IEEE.

Xiao, L., Greenstein, L., Mandayam, N., and Trappe, W. (2008). MIMO-assisted channel-based authentication in wireless networks. In *Proceedings of Conference on Information Sciences and Systems (CISS)*, pages 642–646, Princeton, NJ. CISS.

Xiao, L., Greenstein, L., Mandayam, N., and Trappe, W. (2008). Using the physical layer for wireless authentication in time-variant channels. In *IEEE Transactions on Wireless Communications*, *6*(1), pages 2571–2579. IEEE.