De La Salle University
College of Engineering
Electronics and Communications Engineering

LBYEC4A — EK1

Term-End Final Project:

Comparison of Digital Filters in Denoising Electrocardiogram Signals using MATLAB

Submitted by:
ALVARO, Kimberly Gayle M.
BARREDO, Nathan Matthew T.
YCONG, Ckyle Emanuele M.

Date Submitted:
April 17, 2023

Submitted to:
Ruiz, Ramon Stephen L.

I. ABSTRACT

This study compares various digital filters for electrocardiogram (ECG) signal denoising with the help of MATLAB. ECG readings are susceptible to noise and artifacts, which can make it difficult to diagnose cardiac problems accurately, which is why digital filters are frequently used to denoise ECG signals. This study examines the efficacy of four distinct digital filters, including the lowpass Butterworth filter, Chebyshev Types I and II filters, and a lowpass FIR filter, as well as the four different window methods (Hanning, Hamming, Bartlett, and Blackman). The outcomes demonstrated that the Chebyshev Type 2 filter and Blackman window outperformed the competition in terms of noise removal and preservation.

II. INTRODUCTION

Using nodes positioned all over the body, an electrocardiogram (ECG) is a medical instrument that tracks the electrical activity of the heart. It monitors the SA node of the heart's spontaneous depolarization (SD) wave. Segments, waves, and intervals are used to group the deflection waves, where the wave represents the actual deflection of the ECG signal, while the straight line separating each segment and interval indicates their respective intervals. P-waves signify the depolarization of the atria, while T-waves signify the repolarization of the ventricles [1]. The depolarization of the ventricles is represented by the three waves (Q, R, and T) of the QRS complex, which is the densely packed negative-positive-negative deflection in an ECG signal. The deflections visible in the ECG graph are caused by the SD flow through the carefully positioned nodes on the body [2]. However, the nodes are susceptible to interference from various factors, and corresponding frequency ranges where the noises/interference can be found exist.

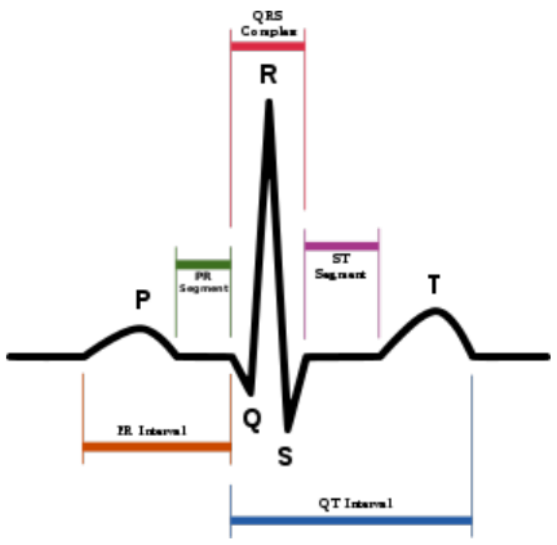


Figure 2.1. ECG Signal at a Time Interval [3]

Several external influences that influence the electrical nodes distributed throughout the body can be blamed for noise or interference in an ECG signal. Low- and high-frequency interference are two subcategories of these noises. Baseline wander (BW) is a type of low-frequency interference which happens when the impedance between the electrodes and the patient's skin, breathing, or movement is incorrect. The ECG signal is impacted by baseline wander when the baseline is not steady and is moved upward. This interference frequently happens around 0.5-0.6 Hz, and their frequency rises with movement. On the other hand, powerline interference and Electromyogram (EMG) noise are examples of high-frequency interference. When a powerline is situated close to the subject, powerline interference (PLI) happens. The frequency range of this noise is between 50 and 60 Hz. When muscle electrical activity outside of the heart is detected by the nodes, EMG noise occurs. This often happens when there is patient movement, and occurs at frequencies greater than 100 Hz [3].

III. THEORETICAL CONSIDERATION

1. **DIGITAL FILTERS:** A digital filter uses a digital processor to perform numerical calculations on sampled values of the signal. The processor may be a general-purpose computer such as a PC, or a specialized DSP (Digital Signal Processor) chip.

   a. **Low-pass filter -** A low-pass filter (LPF) is a circuit that only passes signals below its cutoff frequency while attenuating all signals above it. It is the complement of a high-pass filter, which only passes signals above its cutoff frequency and attenuates all signals below it. It has applications in anti-aliasing, reconstruction, and speech processing, and can be used in audio amplifiers, equalizers, and speakers.

   b. **FIR filter -** a filter whose impulse response is of finite period, as a result of it settles to zero in finite time. This is often in distinction to IIR filters, which can have internal feedback and will still respond indefinitely. The impulse response of an Nth order discrete time FIR filter takes precisely N+1 samples before it then settles to zero. FIR filters are the most popular kind of filters executed in software and these filters can be continuous time, analog or digital and discrete time.

   c. **IIR Filter**

      i. **Butterworth filter -** a filter associated with the passband region having a frequency response that is flat. Ideally, the order of the butterworth determines the frequency response of filter design, a higher order will have a steeper roll-off region which when further cascaded will have an increase in the order will resemble more the brick-type ideal frequency response of the filter. [4]

      ii. **Chebyshev Types I and II filters -** filter designs that have steeper roll-off as compared to that of the Butterworth filter. Because of this steeper roll-off it has a ripple on either the passband region or in the attenuation region or the stopband region. The downside of Chebyshev filters is the ripples in the passband, which increases the error between idealized and real filter characteristics along the range of filters. [4]

2. **Window Methods** - involves multiplying the ideal impulse response with a window function to generate a corresponding filter, which tapers the ideal impulse response. Like the frequency sampling method, the windowing method produces a filter whose frequency response approximates a desired frequency response. The windowing method, however, tends to produce better results than the frequency sampling method [5].

   a. **Hanning Window** - It is a type of window method wherein it processes data through hanning window before applying a fast fourier transform. It calculates the first half or the positive values of cosine [6]. In one dimension, it is mathematically defined as:

$$result(i) = \frac{1}{2}\left(1 - \cos\left(\frac{2\pi i}{n-1}\right)\right)$$

Figure 2.1 Hanning Window Mathematical Definition

   b. **Hamming Window** - It is a type of window method which is a taper created by a raised cosine with endpoints of non-zero in which it was designed to reduce the closest side lobe [7]. It is mathematically defined as:

$$w(n) = 0.54 - 0.46\cos\left(\frac{2\pi n}{M-1}\right) \qquad 0 \leq n \leq M-1$$

Figure 2.2 Hamming Window Mathematical Definition

   c. **Bartlett Window** - is kind of identical to the triangular window, although the difference is that the endpoints are zero. It is frequently employed in signal processing to taper a signal while minimizing a frequency domain ripple [8]. It is mathematically defined as:

$$w(n) = \frac{2}{M-1}\left(\frac{M-1}{2} - \left|n - \frac{M-1}{2}\right|\right)$$

Figure 2.3 Bartlett Window Mathematical Definition

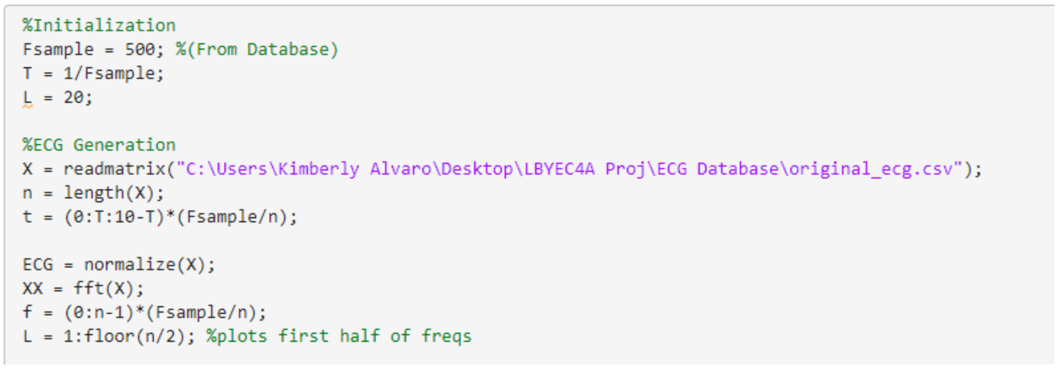
d. **Blackman Window -** the first three terms of cosine summation are used to create this type of window method. Through this, it was knowingly made so that almost little to no leakage would happen as much as possible. Moreover, the Blackman window is much more efficient and optimal out of all the window methods [9]. It is mathematically defined as:

$$w(n) = 0.42 - 0.5\cos(2\pi n/M) + 0.08\cos(4\pi n/M)$$

Figure 2.4 Blackman Window Mathematical Definition

IV. METHODOLOGY

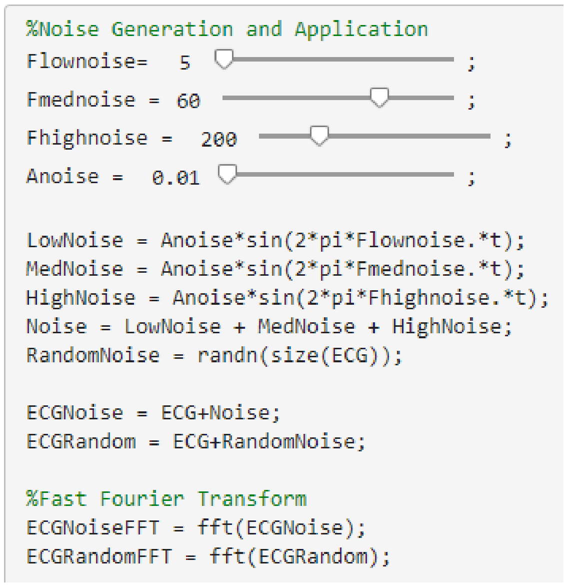
The first step is to identify and acquire a signal source, in this case an ECG signal that has been inputted from an online ECG database. Once the signal has been chosen, it will be processed using the various filters and window methods that have been chosen in order to compare and determine which one is the most effective at denoising electrocardiogram signals.



```
%Initialization
Fsample = 500; %(From Database)
T = 1/Fsample;
L = 20;

%ECG Generation
X = readmatrix("C:\Users\Kimberly Alvaro\Desktop\LBYEC4A Proj\ECG Database\original_ecg.csv");
n = length(X);
t = (0:T:10-T)*(Fsample/n);

ECG = normalize(X);
XX = fft(X);
f = (0:n-1)*(Fsample/n);
L = 1:floor(n/2); %plots first half of freqs
```

Figure 4.1: ECG Generation MATLAB CODE

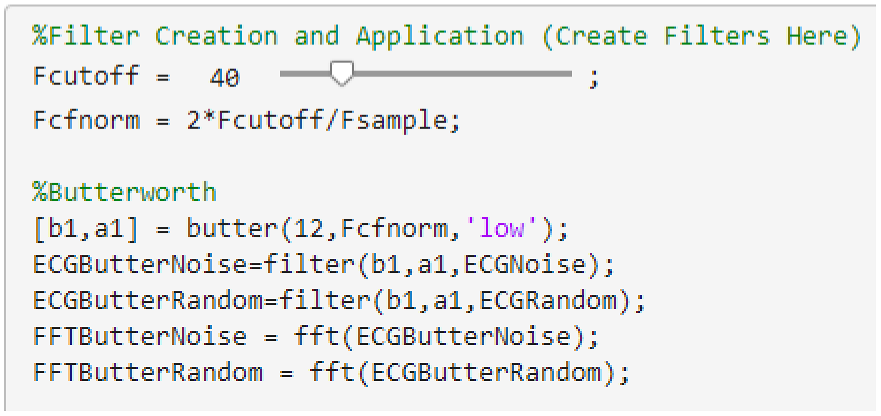
The next step is to introduce noise to the signal, which can be done through various methods such as adding random noise and using signal jammers. This step is crucial as it helps to simulate real-world scenarios where signals are often contaminated with noise. Here, four kinds of noises are introduced to the signal; high frequency noise that can come from an electromyogram (can have frequencies that exceed 100 Hz), mid frequency noise which can come from powerline interference (50-60 Hz), low frequency noise from the baseline wander (0.5 Hz), and a random signal that will be generated in MATLAB. The noise is then applied to the signal to create a noisy signal, which is then put through the selected filters and window methods.



```
%Noise Generation and Application
Flownoise=   5   ;
Fmednoise =  60   ;
Fhighnoise =   200   ;
Anoise =   0.01   ;

LowNoise = Anoise*sin(2*pi*Flownoise.*t);
MedNoise = Anoise*sin(2*pi*Fmednoise.*t);
HighNoise = Anoise*sin(2*pi*Fhighnoise.*t);
Noise = LowNoise + MedNoise + HighNoise;
RandomNoise = randn(size(ECG));

ECGNoise = ECG+Noise;
ECGRandom = ECG+RandomNoise;

%Fast Fourier Transform
ECGNoiseFFT = fft(ECGNoise);
ECGRandomFFT = fft(ECGRandom);
```

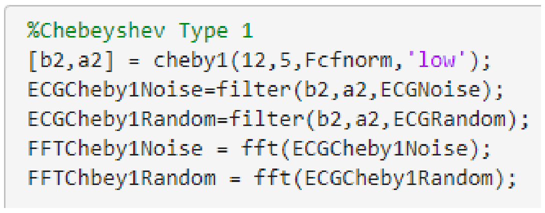
Figure 4.2: Noise Generation and Application MATLAB CODE

To improve the quality of the noisy signal, the next step is to apply a filter. The filter can be designed using various techniques, such as finite impulse response (FIR) or infinite impulse response (IIR) filters. For this paper, the students have chosen the lowpass Butterworth filter, Chebyshev Types I and II filters, and the four different window methods (Hanning, Hamming, Bartlett, and Blackman), and applied each to the noisy signal to both reduce the amount of noise present in the signal, and to enhance the quality of the signal. In order to apply these filters and process the noisy signal, MATLAB software was utilized in order to produce the filtered signal for each filter and compare the results of the experiment.

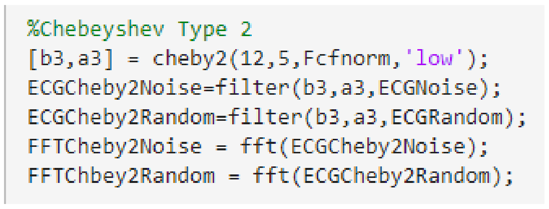


```
%Filter Creation and Application (Create Filters Here)
Fcutoff =   40                   ;
Fcfnorm = 2*Fcutoff/Fsample;

%Butterworth
[b1,a1] = butter(12,Fcfnorm,'low');
ECGButterNoise=filter(b1,a1,ECGNoise);
ECGButterRandom=filter(b1,a1,ECGRandom);
FFTButterNoise = fft(ECGButterNoise);
FFTButterRandom = fft(ECGButterRandom);
```

Figure 4.3: Butterworth Filter MATLAB CODE

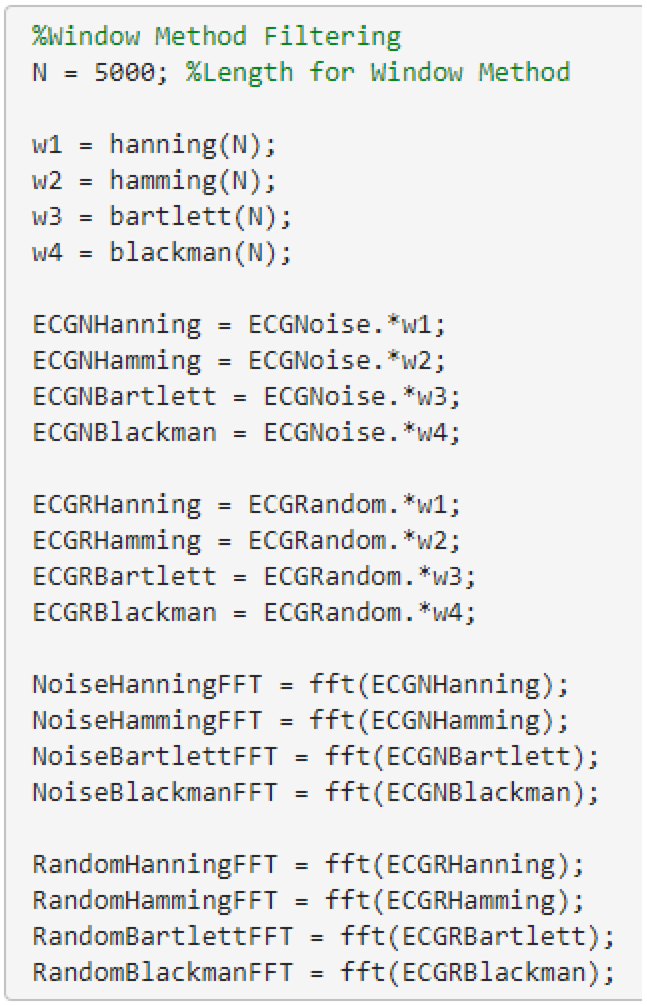


```
%Chebeyshev Type 1
[b2,a2] = cheby1(12,5,Fcfnorm,'low');
ECGCheby1Noise=filter(b2,a2,ECGNoise);
ECGCheby1Random=filter(b2,a2,ECGRandom);
FFTCheby1Noise = fft(ECGCheby1Noise);
FFTChbey1Random = fft(ECGCheby1Random);
```



```
%Chebeyshev Type 2
[b3,a3] = cheby2(12,5,Fcfnorm,'low');
ECGCheby2Noise=filter(b3,a3,ECGNoise);
ECGCheby2Random=filter(b3,a3,ECGRandom);
FFTCheby2Noise = fft(ECGCheby2Noise);
FFTChbey2Random = fft(ECGCheby2Random);
```

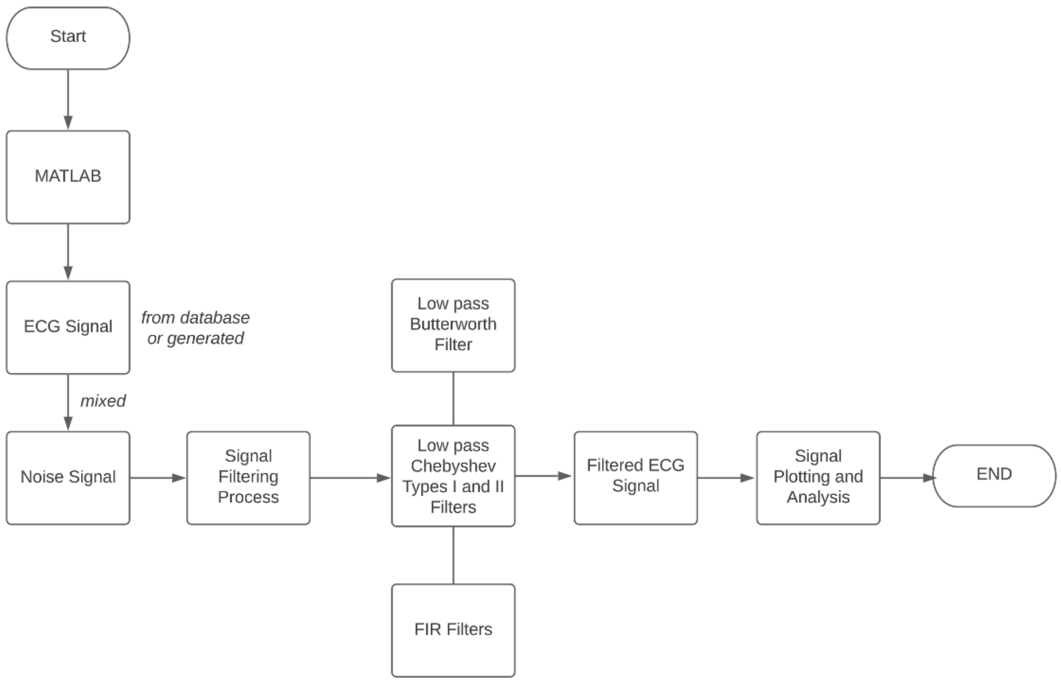
Figure 4.4: Chebyshev Types 1 and 2 MATLAB CODE



```
%Window Method Filtering
N = 5000; %Length for Window Method

w1 = hanning(N);
w2 = hamming(N);
w3 = bartlett(N);
w4 = blackman(N);

ECGNHanning = ECGNoise.*w1;
ECGNHamming = ECGNoise.*w2;
ECGNBartlett = ECGNoise.*w3;
ECGNBlackman = ECGNoise.*w4;

ECGRHanning = ECGRandom.*w1;
ECGRHamming = ECGRandom.*w2;
ECGRBartlett = ECGRandom.*w3;
ECGRBlackman = ECGRandom.*w4;

NoiseHanningFFT = fft(ECGNHanning);
NoiseHammingFFT = fft(ECGNHamming);
NoiseBartlettFFT = fft(ECGNBartlett);
NoiseBlackmanFFT = fft(ECGNBlackman);

RandomHanningFFT = fft(ECGRHanning);
RandomHammingFFT = fft(ECGRHamming);
RandomBartlettFFT = fft(ECGRBartlett);
RandomBlackmanFFT = fft(ECGRBlackman);
```

Figure 4.5: Window Method Filter MATLAB CODE

Finally, the results obtained from applying the filter are compared to the noisy original signal to determine the effectiveness of the filter. The respective FFTs of the filtered signals are compared to the respective noise (noisy and random noise) by utilizng a built-in function of MATLAB. Through that, the researchers would be able to observe the difference between the filtered signals and the noisy ECG signals. Same goes for the window methods as well, so that it could observe the differences between the FFTs of the window filtered signals to the noisy/random noise ECG signals, in order to conclude which of the filtering methods produces the best results.



Start
MATLAB
ECG Signal
from database or generated
mixed
Noise Signal
Signal Filtering Process
Low pass Butterworth Filter
Low pass Chebyshev Types I and II Filters
FIR Filters
Filtered ECG Signal
Signal Plotting and Analysis
END

Figure 4.6. Flowchart of Experiment

V. RESULT

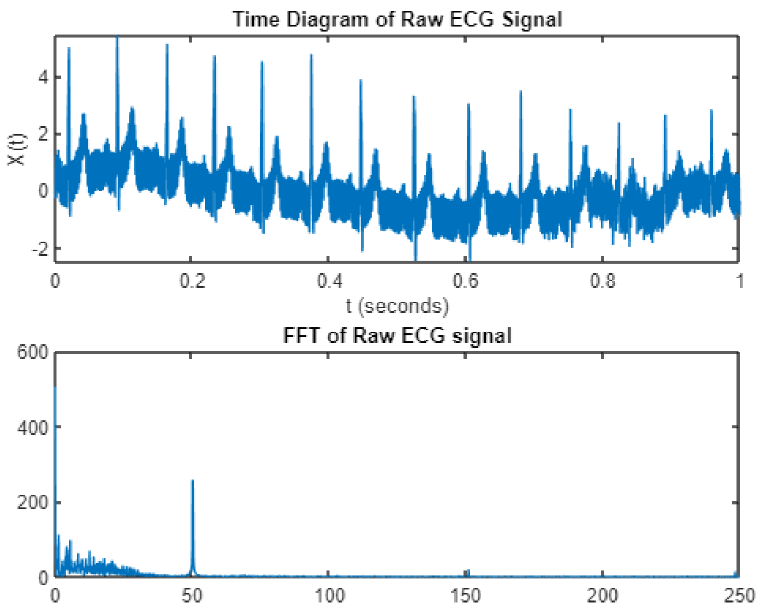


Figure 5.1: Original ECG Signal from Database

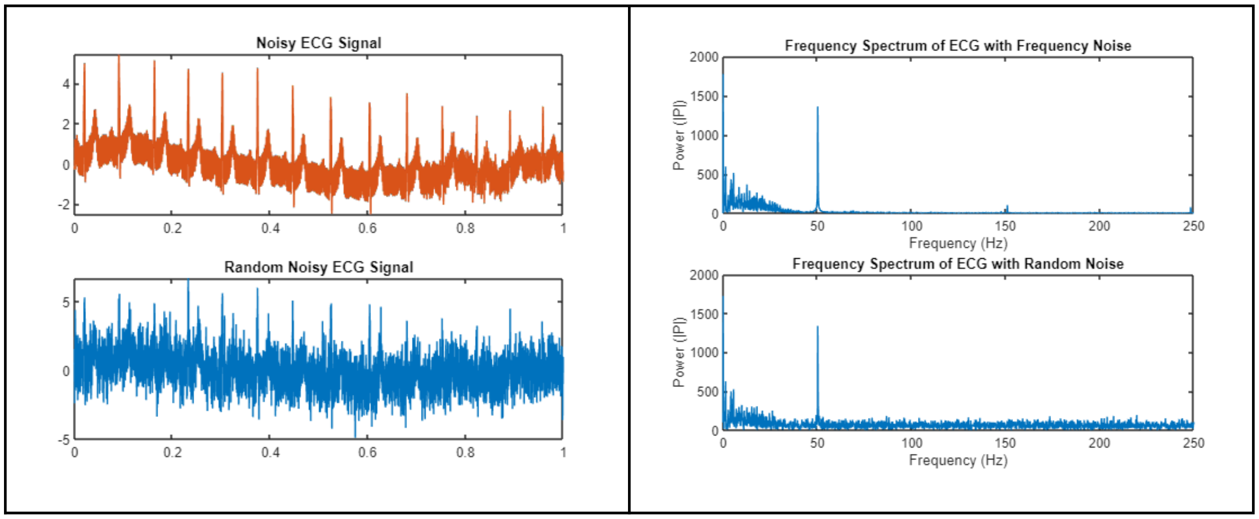


Figure 5.2: Applied Noise to ECG Signal with respective FFTs

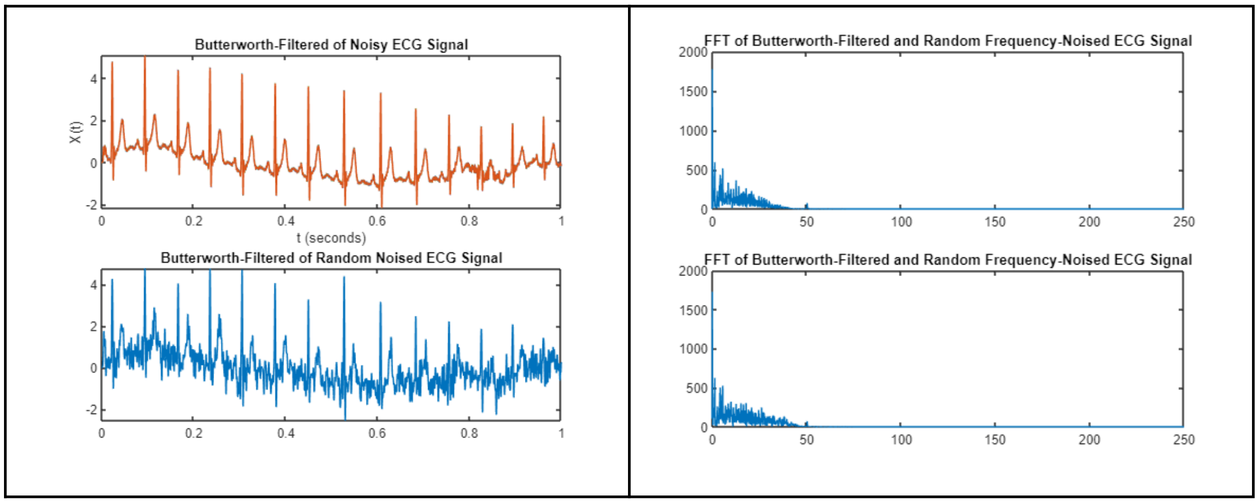


Figure 5.3: Butterworth Filtered Signal with respective FFTs

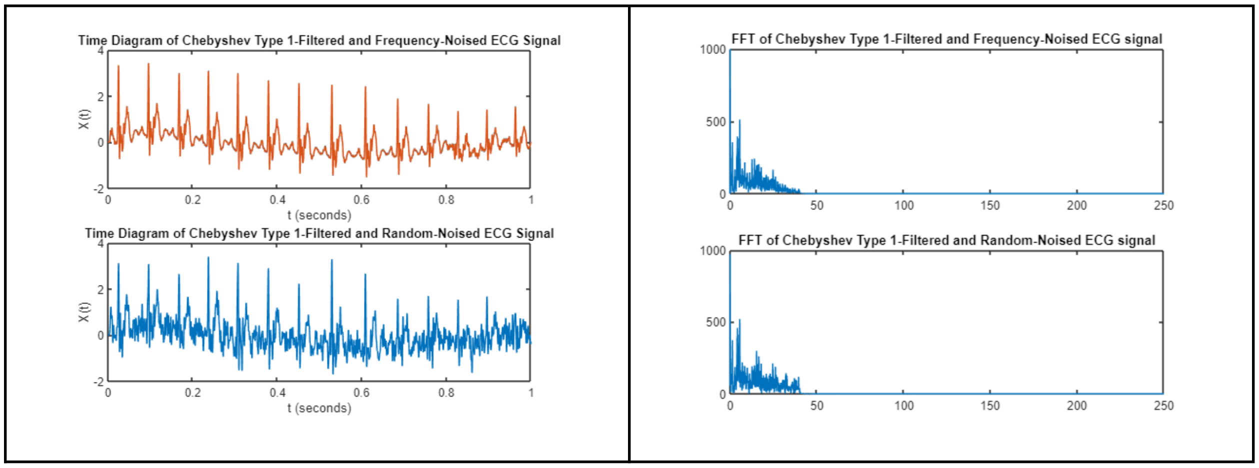


Figure 5.4: Chebyshev Type 1 Filtered Signal with respective FFTs

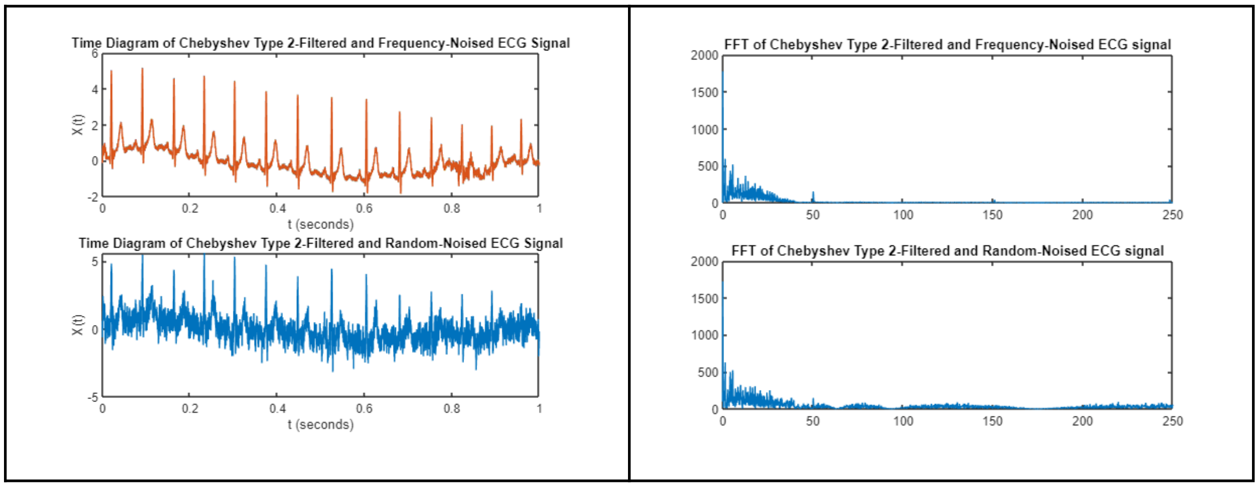
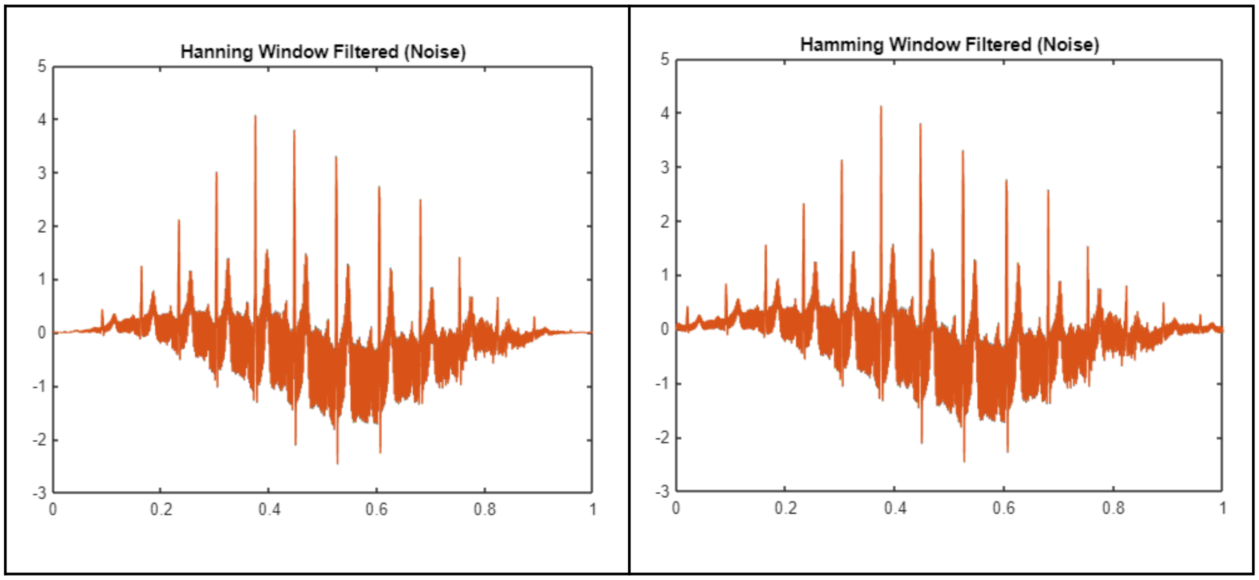
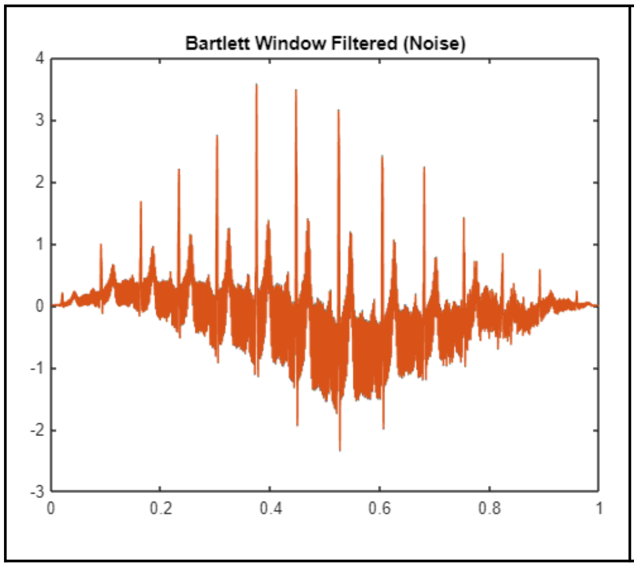


Figure 5.5: Chebyshev type 2 Filtered Signal with respective FFTs





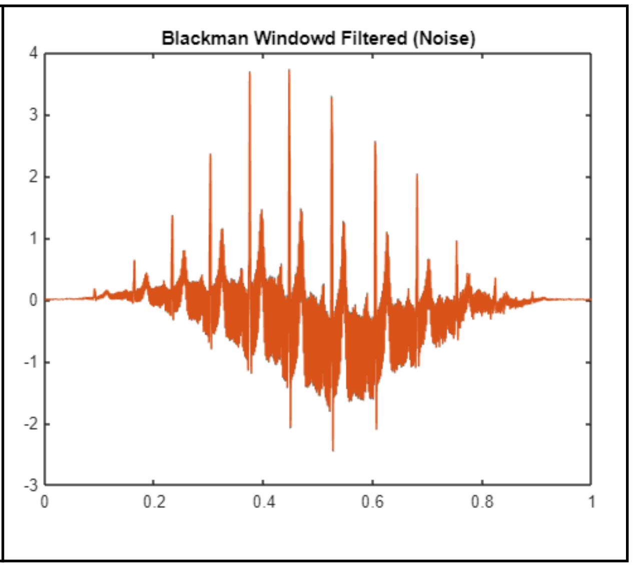
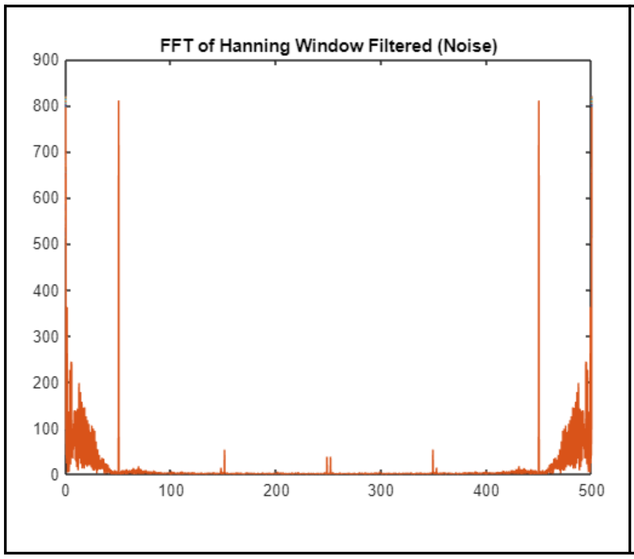
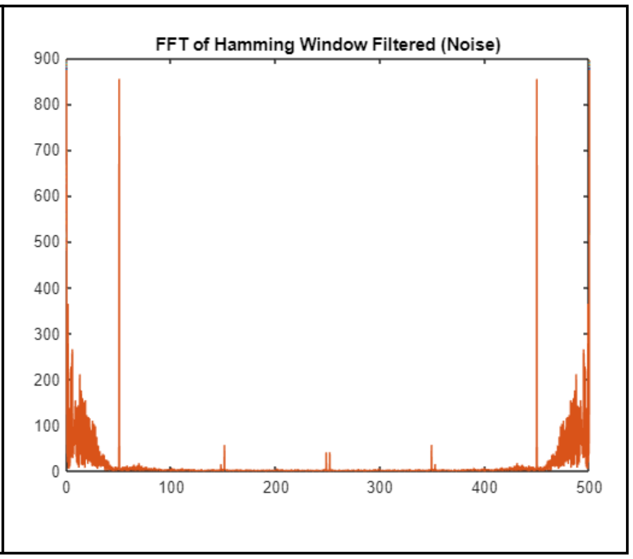


Figure 5.6: Window Method Filtered Signal (Noisy)





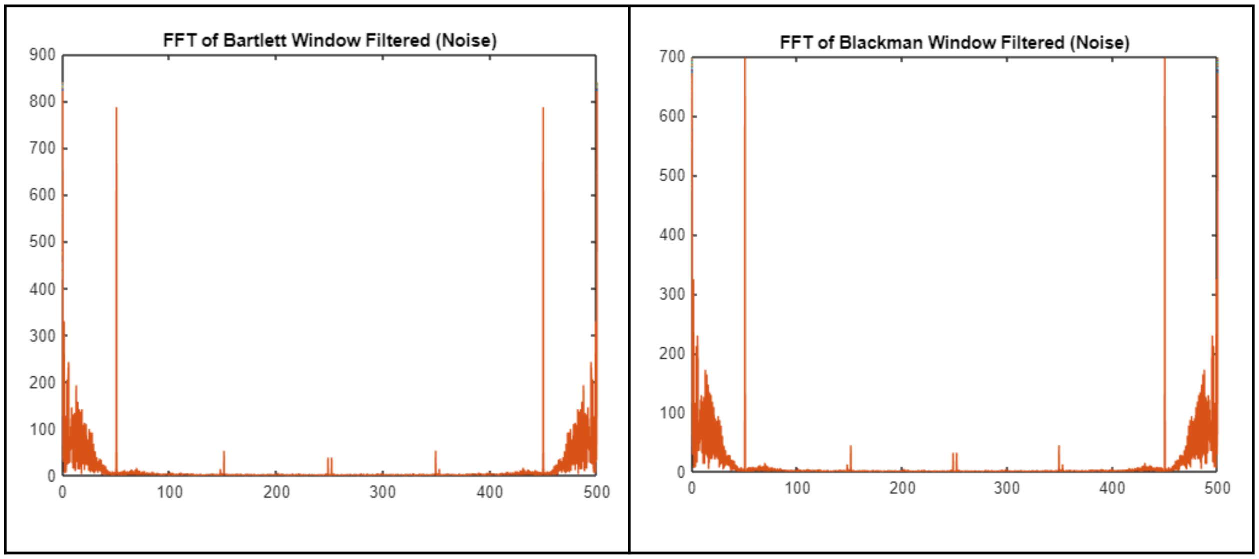
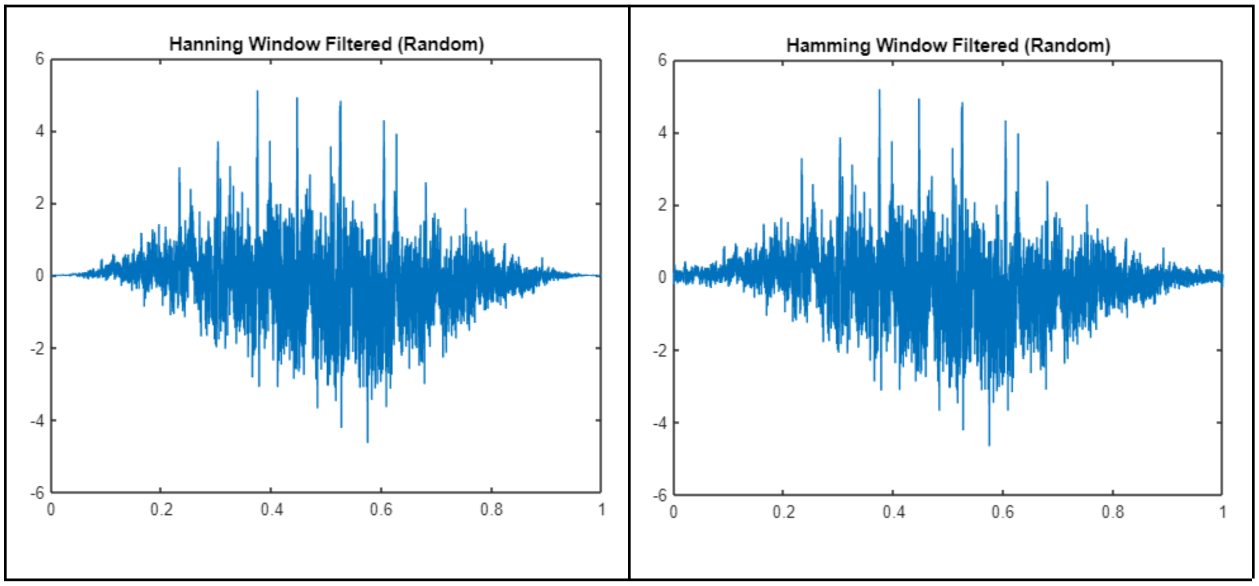


Figure 5.7: FFTs of Window Method Filtered Signal (Noisy)



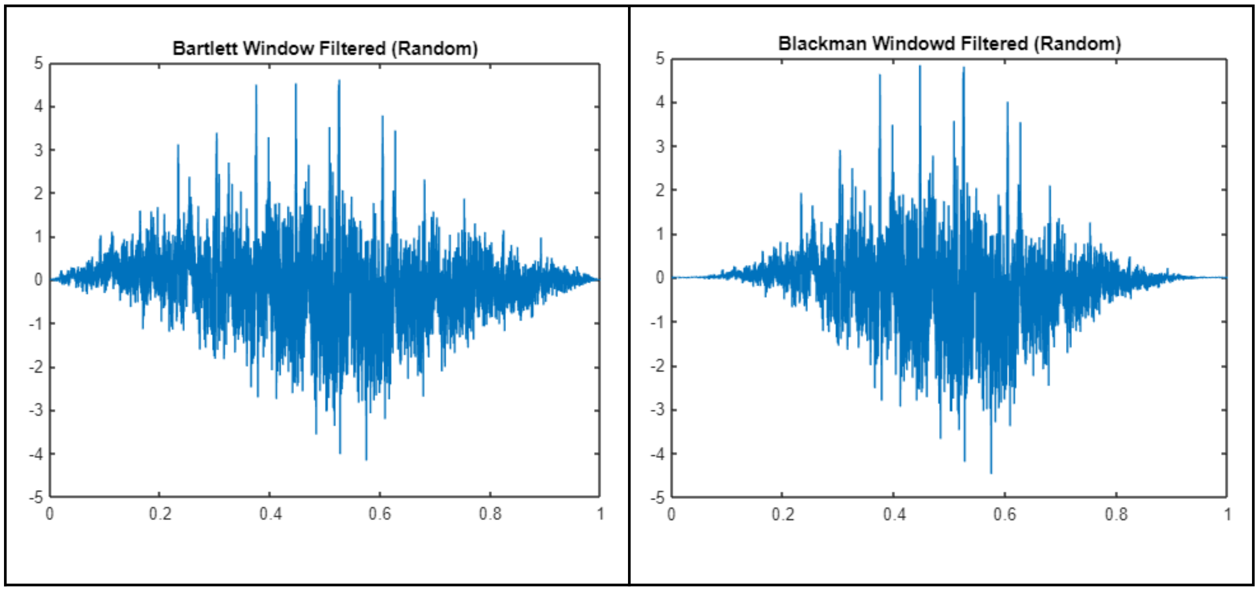
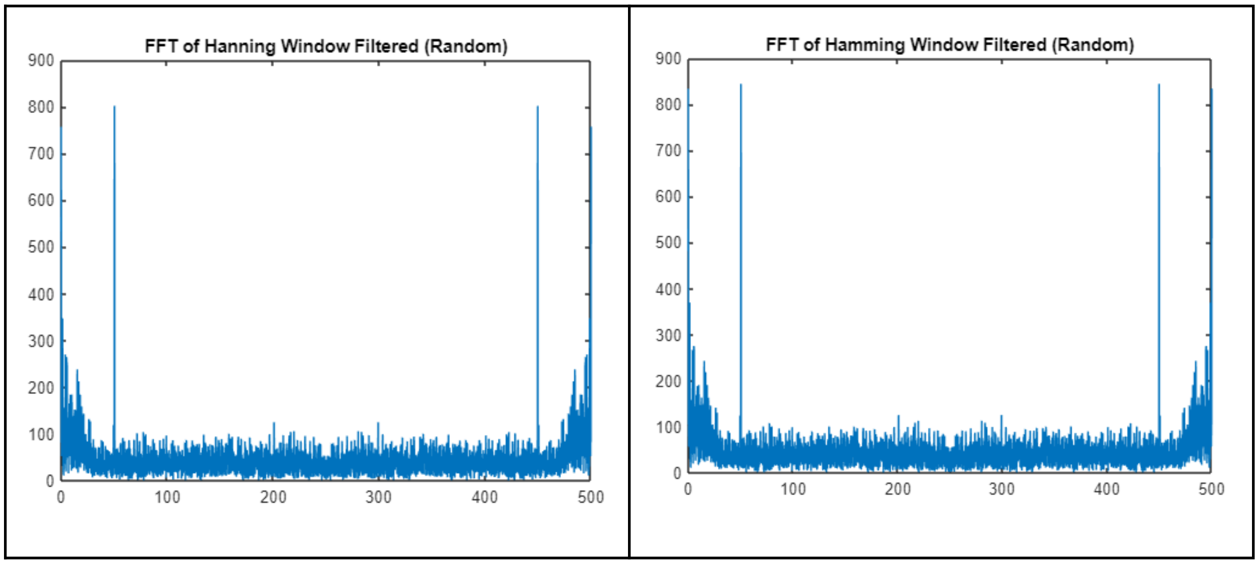


Figure 5.8: Window Method Filtered Signal (Random)



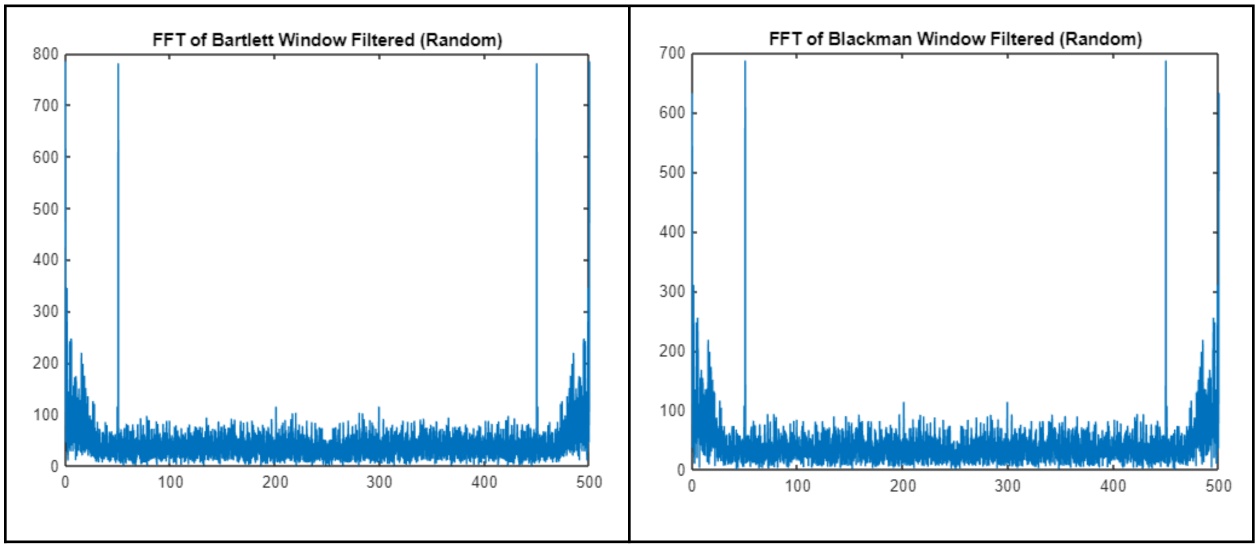


Figure 5.9: FFTs of Window Method Filtered Signal (Random)

VI. DISCUSSION

In the initialization part of the code, instead of generating an ECG signal within MATLAB, the ECG signal is obtained from a database called the "PhysioBank ATM". Wherein the ECG signals came from real persons with different kinds of heart conditions. There are a lot of different options to choose from, however, the researchers opted for the regular Raw ECG signal so that there would be no abnormalities or errors in the filtering process. The ECG signal was then exported as a .csv file. The exported file contains both the unfiltered and filtered ECG signal. The researchers only utilized the unfiltered ECG signal because the main purpose of the project is to filter out the noise from the ECG signal. The sampling frequency is also provided from the database as well, which is at 500Hz. With a period of 1/500Hz (or 0.002) and a Length (L) of 20.

By using the built-in function *readmatrix,* the Raw ECG signal is then defined within MATLAB. Then, a time vector was established according to the length of the Raw ECG signal. A length for the frequency is also established as it would be utilized when plotting for the respective FFTs of the signals. To generate the FFT of a signal, simply use the built-in function *fft* of MATLAB. When plotting the FFT of a signal, use the function *abs* to generate the magnitude of the FFT in the frequency domain. The plotted FFT only shows the first half of the graph for less processing and it would be easier to differentiate with other signals. The original ECG signal is then plotted together with its FFT as shown in figure 5.1.

For the noise generation and application part of the code, firstly four (4) different noises are generated: low-frequency noise, medium-frequency noise, high-frequency noise, and random noise. For the low, medium, and high-frequency noises, a sinusoidal equation was defined with different frequency ranges. As seen in figure 4.2, the established frequency for low, medium, and high-frequency noises is shown. After defining each of the aforementioned noises, the three are then combined into one single noise to lessen the length of the filtering process of the code. Aside from that, random noise was generated by using the *randn* function with respect to the length of the original ECG signal. Keep in mind that the length for each signal should be the same so that there would be no errors when they are combined. Two kinds of noisy signals are now generated, Noisy ECG signal and Random Noise ECG signal with their respective FFTs also established. The graphs are shown in figure 5.2.

Onto the filtering part of the code, firstly a cutoff frequency is established at 40Hz, and normalize the cutoff frequency with an equation of 2*cutoff freq/sampling freq. The first filter used is the Lowpass Butterworth filter. By using the built-in function *butter* with a 12th filter order to generate the two coefficients (b1 and a1) that would be used in the next step. Then by using the *filter* function and integrating the two coefficients (b1 and a1) in order to filter out the noise that is present in both the Noisy ECG signal and Random Noise ECG signal. Their respective FFTs are also established and plotted. The plotted graphs from the Butterworth filtered signals and respective FFTs are shown in figure 5.3. This goes the same with the next filters, Chebyshev types 1 and 2. The built-in functions used are *cheby1* and *cheby2*, respectively. The parameters used in Butterworth are also used in both Chebyshev types 1 and 2, the only addition is the passband ripple which is 5. Then the filtering process is also the same after obtaining the respective coefficients and establishing their respective FFTs. The plotted graphs of Chebyshev type 1 and 2 are seen in figure 5.4 and 5.5, respectively. After establishing all of the IIR filters, in order to differentiate the researchers utilize a built-in function *hold on* and *hold off* when plotting. As seen in figure 6.1, it would be observed that the peak at 50Hz is minimized in all filters. However, the filter that produced the clean signal wherein the noise is little to no noise, is the Chebyshev type 1 filter for the noisy ECG signal. As observed in figure 6.2 and 5.4, it is seen that the produced filtered signal is cleaner and the form of the ECG signal is clearly seen. This also applies to the random noise ECG signal, the Chebyshev type 1 has a cleaner output than the rest of the implemented IIR filters. Therefore, the Chebyshev type 1 proved to be an efficient filter when it comes to denoising ECG signals.

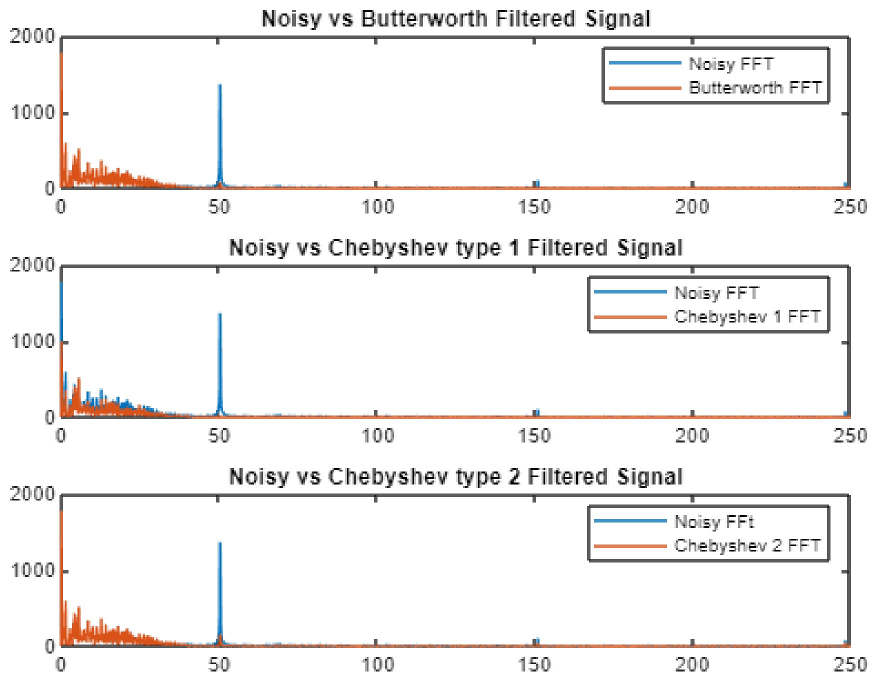


Figure 6.1: Difference between the FFTs of filtered signals to Noisy ECG signal

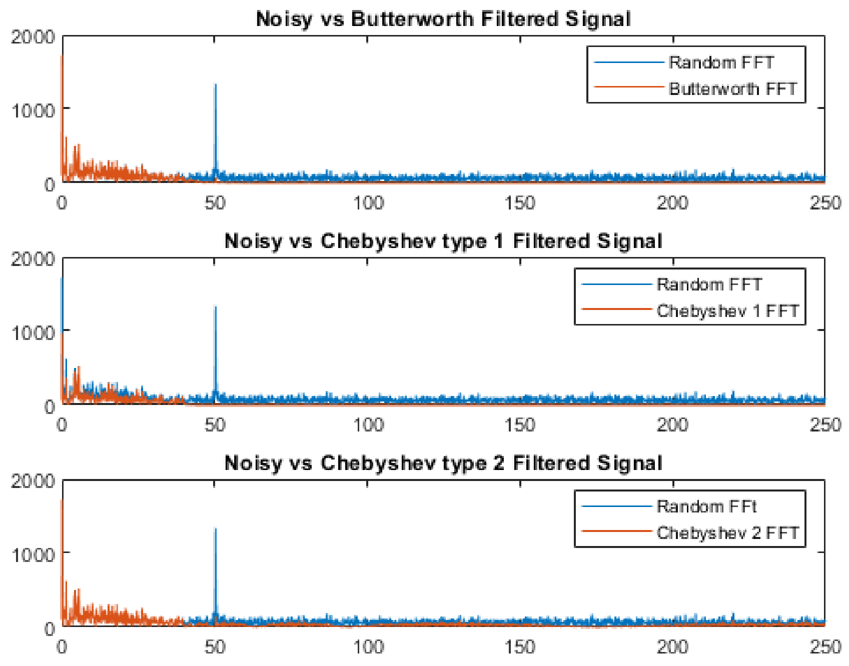


Figure 6.2: Difference between FFTs of filtered signals to Random Noise ECG Signal

For the next part of the filtering process, FIR filters are utilized specifically the window methods. These are the Hamming, Hanning, Bartlett, and Blackman window methods. There are built-in functions of these window methods within MATLAB namely: *hamming(N), hanning(N), bartlett(N),* and *blackman(N)*. Wherein the N is the length of the window. Before establishing the length (N), it must be the same length as the ECG signal. With the help of the workspace portion of MATLAB software, one could be able to see the size of each variable. According to the workspace, the size or length of both noisy and random noise ECG signals is 5000. Thus the length (N) for the window method is 5000 as well. Afterward, integrate the window filters to the noisy and random noise by using an element-by-element operation ". *". Afterward, each of its respective FFTs is also established. As you would observe in figure 5.6, the filtering process is only applied to the side portions of the ECG signals, leaving the middle portion unfiltered. Moreover, if you would differ the FFTs of each window method, you would observe that there is not much of a difference done by the window method to filter the noise out as seen in figures 6.3 and 6.4. However, the filter that denoises the most at the sides out of all the window methods is the Blackman window. Although the difference is only a little but it is the only window method that filters most of the noise present at the side portions. This also applies to the random noise ECG signal when the window method filters are implemented.

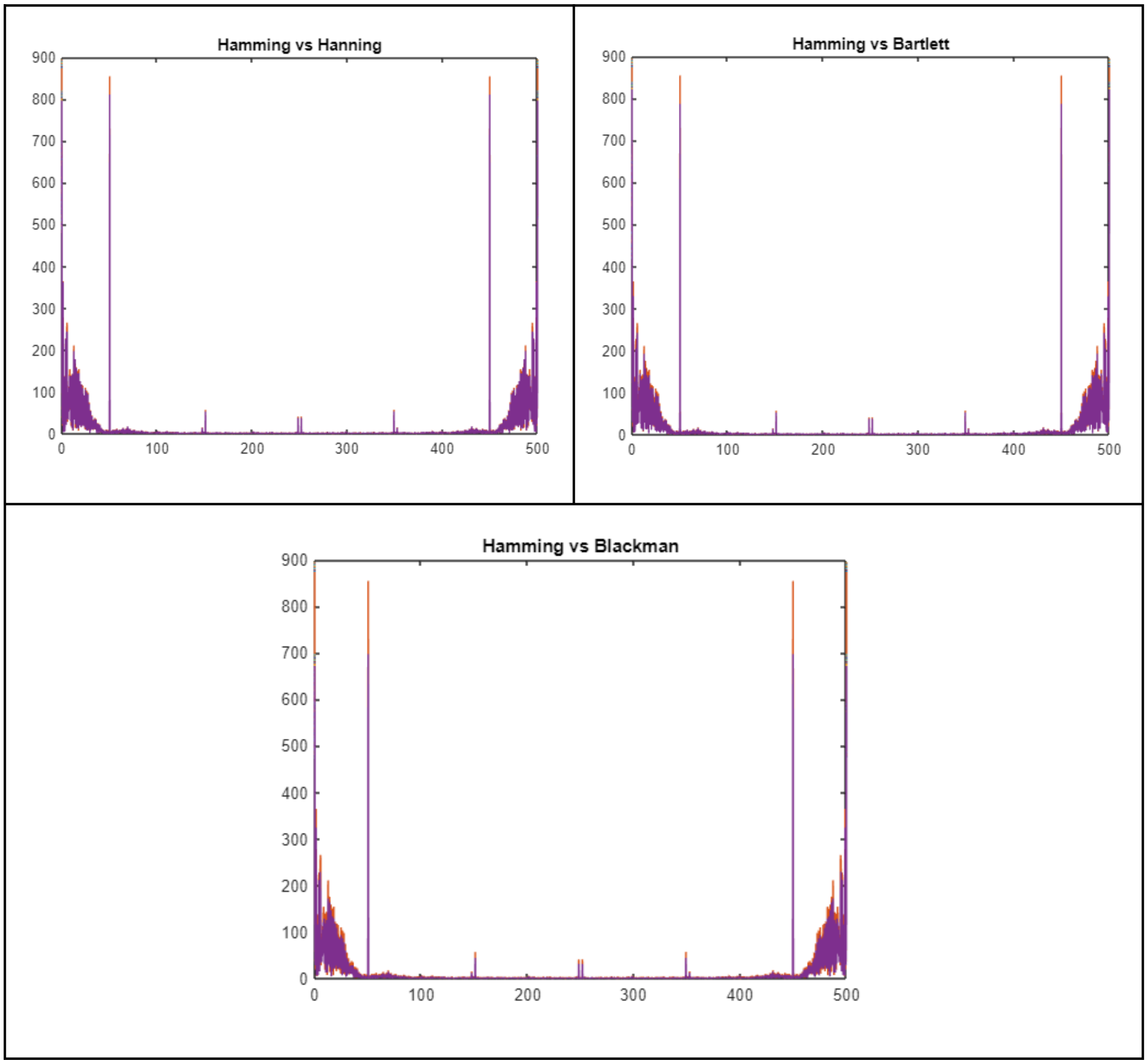


Figure 6.3: Difference of the FFTs of Window Method Filtered Signals (Noisy)

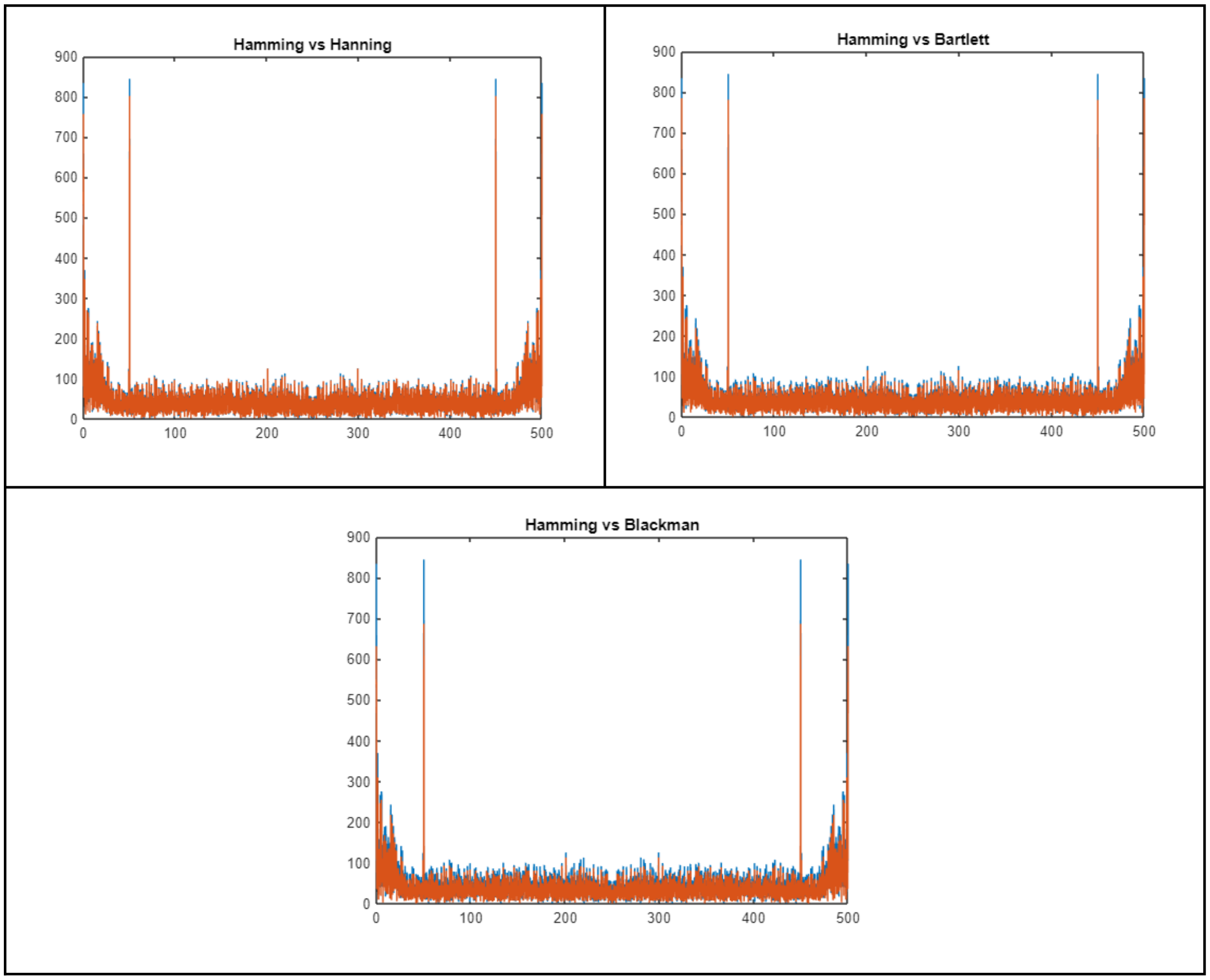


Figure 6.4: Difference of the FFTs of Window Method Filtered Signals (Random Noise)

VII. CONCLUSION

To conclude, the implemented IIR filters such as the low pass Butterworth, and Chebyshev types 1 and 2 efficiently denoise the present noise in the ECG signal. As shown in the results above, all of the IIR filters produced a clean and denoised ECG signal. Furthermore, when they are differentiated with the original FFT of the noisy/random noisy ECG signal, you would be able to observe the distinction between the two as seen in figures 6.1 and 6.2. The Chebyshev type 1 filter produced the best results when it comes to denoising ECG signals as it has little to no noise in the filtered signal. This implies that the generated codes for the IIR filters were working perfectly. However, when it comes to the FIR filters, there is a large portion in the middle of the signal that was untouched by the window method to filter. Furthermore, there was not much that the window method filtered out as observed in the respective FFTs of the filtered signals through the window method. There could be another method or way to improve the code for implementing the window method to effectively filter out all of the noise that is present in the ECG signal.

Another recommendation that could be made is to have another method to analyze the signals such as the Mean Squared Error (MSE) or correlation between the two signals to have a further analysis of the difference between the signals. And could see the effectiveness of each filter method in denoising the noisy ECG signals.

VIII. AUTHORS CONTRIBUTION

- ALVARO, Kimberly Gayle - MATLAB Code, Results, Discussion, and Conclusion
- BARREDO, Nathan Matthew - Introduction, and Theoretical Considerations
- YCONG, Ckyle Emanuele - MATLAB Code, and Conclusion

IX. REFERENCES

[1] L. Potter, “Understanding an ECG: ECG interpretation,” Geeky Medics, 12-Nov-2021. [Online]. Available: https://geekymedics.com/understanding-an-ecg/. [Accessed: 13-Mar-2023].

[2] D. Price, “How to read an electrocardiogram (ECG). part One: Basic principles of the ECG. the normal ECG,” South Sudan Medical Journal. [Online]. Available: http://www.southsudanmedicaljournal.com/archive/may-2010/how-to-read-an-electrocardiogram-ecg.-part-one-basic-principles-of-the-ecg.-the-normal-ecg.html. [Accessed: 13-Mar-2023].

[3] R. Kher, “Signal Processing Techniques for Removing Noice from ECG Signals,” J Biomed Eng 1: 1-9. [Accessed: 01-May-2022].

[4] Rastogi, N., & Mehra, R. (2013). Analysis of butterworth and Chebyshev filters for ECG Denoising Using Wavelets. Retrieved July 6, 2022, from https://iosrjournals.org/iosr-jece/papers/Vol6-Issue6/G0663744.pdf

[5] “Windowing Method,” *Windowing method :: Linear Filtering and filter design (Image Processing Toolbox User's Guide)*. [Online]. Available:

http://matlab.izmiran.ru/help/toolbox/images/linfil11.html#:~:text=The%20windowing%20method%20involves%20multiplying,approximates%20a%20desired%20frequency%20response. [Accessed: 16-Apr-2023].

[6] *PV-wave online help*. [Online]. Available: https://help.perforce.com/pv-wave/12.0/PVWAVE_Online_Help/pvwave.html#page/Foundation/hanning.html. [Accessed: 16-Apr-2023].

[7] "Scipy.signal.hamming¶," *scipy.signal.hamming - SciPy v0.19.1 Reference Guide*. [Online]. Available: https://docs.scipy.org/doc/scipy-0.19.1/reference/generated/scipy.signal.hamming.html#:~:text=The%20Hamming%20window%20is%20a,an%20empty%20array%20is%20returned. [Accessed: 16-Apr-2023].

[8] "Scipy.signal.windows.bartlett#," *scipy.signal.windows.bartlett - SciPy v1.10.1 Manual*. [Online]. Available: https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.windows.bartlett.html. [Accessed: 16-Apr-2023].

[9] "Scipy.signal.windows.blackman#," *scipy.signal.windows.blackman - SciPy v1.10.1 Manual*. [Online]. Available: https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.windows.blackman.html. [Accessed: 16-Apr-2023].

[10] gfaitech, "Correlation," Correlation in Signal Processing. [Online]. Available: https://www.gfaitech.com/knowledge/faq/correlation-in-signal-processing#:~:text=The%20correlation%20of%20two%20real,y%20(%20t%20−%20τ%20)%20. [Accessed: 13-Mar-2023].