PROJECT NAME:

EMG Signal Acquisition and Data Processing

PROJECT TEAM

Arif Emre YILDIZ
Esra Firkat YILMAZ
Samet YILMAZ
Talha ÜSTÜNDAĞ

Table of Contents

Introduction

A signal which is collected from any organ that is showing a physical activity is named a biomedical signal. It is a function of time and can be defined in terms of phase, frequency, and amplitude. As a type of biomedical signal, Electromyography, myoelectric activity, can be exemplified. EMG is used in the way of computing electrical currents produced in muscles while presenting neuromuscular activities. Activities of the muscles are controlled by the nervous system so EMG signals are directly related to that system. In order to collect signals from varying motor units, an EMG detector is placed on top of the skin. (Raez, et al., 2006) When a healthy muscle is resting it releases no seek able current, but in the existence of a disease or abnormality muscles emit impulses according to the features of that problem. These impulses can be studied by the created sonic and oscilloscopic patterns. (Miller, 1958) In this project, the aim was to create an EMG system that is able to collect electrical signals from muscles using electrodes. And after filtering the signal transferring the signal to the computer by using Arduino and MATLAB. For the Arduino part, the AD8232 heart rate module has been used and coding has been completed. Values that have been taken from Arduino, have been imported to MATLAB. Other coding issues has been completed and the signal which is obtained for 10 seconds has been graphed digitally. Other components which are like FFT, filtering and RMS have been done in the MATLAB part.

Placing Electrodes

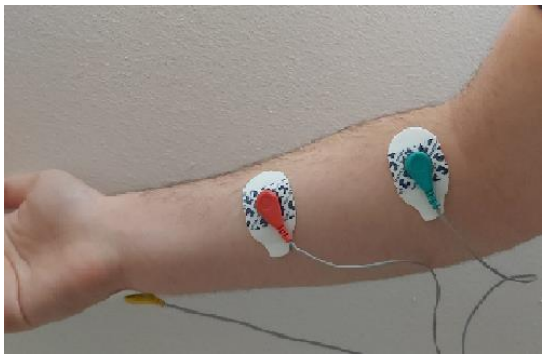


Figure 2.1 Electrode Placement for Forearm

To get EMG signals of a muscle, 3 electrodes should be replaced. In the project, two of the peds were located between the muscles and one is located on wrist as a ground. To find the best condition, two replacement condition were tested. In the first case the peds were located between the forearm interior muscle as in figure 2.1. In second case the, they were located between forearm exterior muscle as in figure 5.1.2.1. Data coming from muscles were collected by using Arduino and MATLAB and plotted with their Fourier Transforms. According to the graphs, in figure 2.5 the amplitudes of the signals are so low when it is compared with the magnitudes of signals in figure 2.3. Even the noise element which is 50Hz seems so high and the sinusoidal looking is weak in the second case (Fig. 2.4), the first case (Fig. 2.2) is the best signal to work on it so the placing the electrodes as in the Fig 2.1 is the best placement.

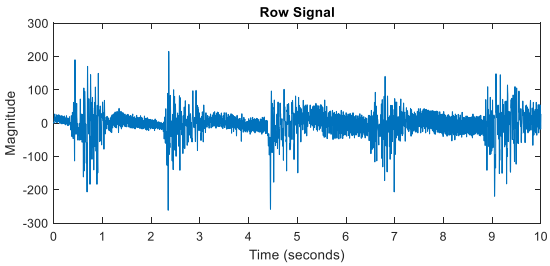


Figure 2.2 Row Signal

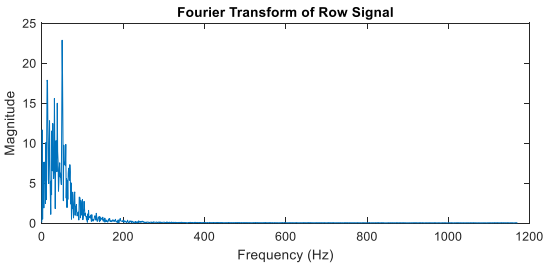


Figure 2.3 FFt of Row Signal

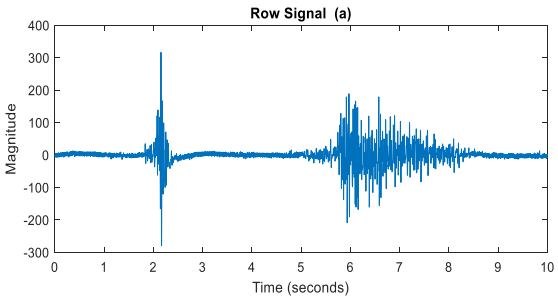


Figure 2.4 Notch Filter for Row Signal

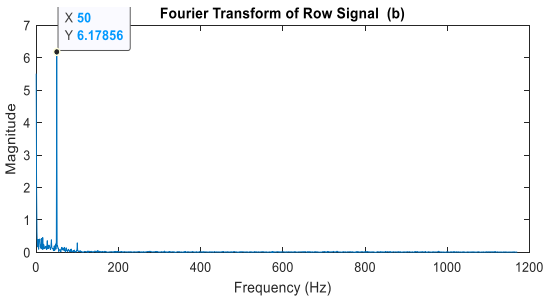


Figure 2.5 Notch Filter Fft of Row Signal

Acquiring Datas

3.1 Connecting and Obtaining Datas from Arduino Uno

The working logic of the Arduino Uno algorithm to get EMG data:

```
int LOn = 9;
int LOp = 10;
int LOp_v;
int LOn_v;
void setup() {
Serial.begin(115200); // We chose the Baud rate as 115200 to set the number of
emg data.
pinMode(LOn, INPUT);
pinMode(LOp, INPUT);
pinMode(A0, INPUT);
}

void loop() {
  LOn_v = digitalRead(LOn);
  LOp_v = digitalRead(LOp);
if ((LOp_v==1)||(LOn_v==1)){
  Serial.println("!!!!Check your electrode connections!!!");
  }
else{
  Serial.println(analogRead(A0));
  int  t=micros();
  }
}
```

EMG signals were measured using the Ad8232 board with the Arduino code seen above. Data was taken by analog reading and EMG signals were digitized using Arduino. The baud rate has been increased from 9600 to 115200 in order to obtain sufficient data. Thus, the number of data was made sufficient for the measurement of EMG signals.

3.2 Recording and plotting by serial communication on Matlab

The MATLAB code used to get the data from the Arduino and measure for 10 seconds:

```
clear all
clc;
x = serialport('COM6', 115200); % Serial
communication with Arduino Uno
%veri21 = zeros(1,30000);
time = []; %
t = 0;
a = 0;
i = 1;
flag = true;
while t < 10 % It is defined to take up to 10
seconds of data
tic; % We started the time
veri = readline(x); % We used it to read data
from arduino

veri21(i) = str2double(veri); % We converted
string to numbers because readline command takes
values as strings.
i = i + 1;
if flag
time = [time ,a];
flag = false;
else
time   =[time ,a+time(end)];
end

a = toc;
t = t + a;
end
save veri21 % The name of EMG signals
```

EMG data received from Arduino was transferred to the MATLAB environment using serial communication. Serial communication with MATLAB was started by selecting the correct port of Arduino Uno. The elapsed time was calculated using the tic toc command to measure 10 seconds of data. The readline command was used to transfer the data to MATLAB, and then the incoming data

was converted from text to number due to the working principle of the readline command. Then, up to 10 seconds of data acquisition was provided with the obtained data. The save command was used to save the received data as a mat file.

Plotting Differents EMG Signals

4.1 Differences Between Graphs

4.1.1 Main Graph of EMG Signal

```
clc;
clear all;
load veri18;
veri18 = veri18(~isnan(veri18)); %If there is any
NAN value when reading data, we used it to remove
it from our data
s = veri18;  fs = 2335; %Our sample number is
23567 so that is how we defined fs approximately.
s = s-mean(s); % We used it to align EMG signals
from the 0 value.
t = (0:size(s,2)-1)/fs;
subplot 321, plot(t',s)
xlabel('time (seconds)'), ylabel('Magnitude')
%ylim([-350 250]);
xlim([0 10]);
```

The coding has been finished in MATLAB. First, data from the Arduino portion was imported into the MATLAB environment. A mat file named "veri18" was utilized for this signal. After that, a specific command called "isnan" was used to ensure that there were no missing values. This command looks for numbers. When reading data, if there is a NAN value, it is erased from our data using this embedded function. The sample frequency, denoted as "fs" in code, and the time required to generate a graph have been computed. In the last section, this information was utilized to generate the desired graph.

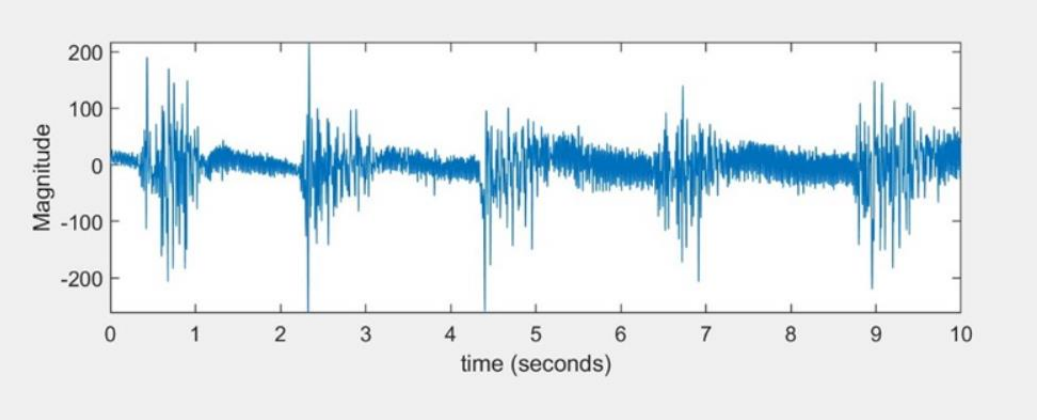


Figure 4.1.1.1 Main Graph of EMG Signal

This EMG signal was taken by making a fist. Figure 4.1.1.1 depicts the EMG signal obtained from Arduino data. This data was collected for ten seconds as the arm muscle relaxed and contracted three times. The greatest magnitude attained for this graph is around 200. In addition, the number of data received in 10 seconds for this EMG signal is 23567. Therefore, the sampling frequency (fs) is set at 2335.

4.1.2 Graph Which Have Higher Amplitude

```
clc;
clear all;
load veri12;
veri12 = veri12(~isnan(veri12)); %If there is any
NAN value when reading data, we used it to remove
it from our data
s = veri12;  fs = 2345; %Our sample number is
23583 so that is how we defined fs approximately.
s = s-mean(s); % We used it to align EMG signals
from the 0 value.
t = (0:size(s,2)-1)/fs;
subplot 321, plot(t',s)
xlabel('time (seconds)'), ylabel('Magnitude')
%ylim([-350 250]);
xlim([0 10]);
```

The coding was accomplished in MATLAB. To begin, data from signals obtained by the Arduino portion has been loaded into the MATLAB environment. This signal was created using a mat file called "veri12." After then, a specific command called "isnan" was used to ensure that there was no

missing value. This command checks for the presence of numbers. When reading data, if there is a NAN value, this embedded function deletes it from our data. The sample frequency, denoted as "fs" in code, and the time required to generate a graph have been determined. As a final step, this data was utilized to generate the desired graph.

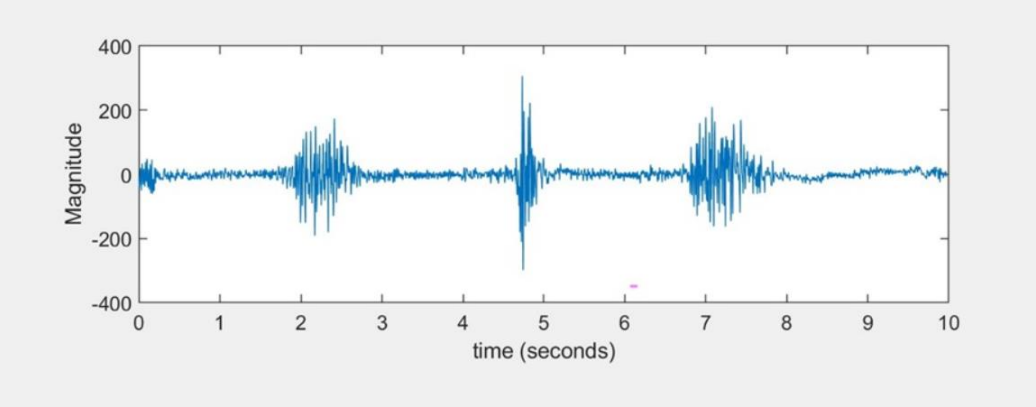


Figure 4.1.2.1 Higher Amplitude Graph

The graph in Figure 4.1.1.2 depicts the EMG signal obtained from Arduino data. This data was collected for 10 seconds as the arm muscle relaxed and contracted three times. The greatest achieved magnitude for this graph is around 300. The number of data received in 10 seconds for this EMG signal is 23583. That is why, the sampling frequency (fs) is 2345.

4.1.3 Graph of Different Places of Arm Muscle

```
clc;
clear all;
load veri9;
veri9 = veri9(~isnan(veri9)); %If there is any
NAN value when reading data, we used it to remove
it from our data
s = veri9;  fs = 2350; %Our sample number is
23656 so that is how we defined fs approximately.
s = s-mean(s); % We used it to align EMG signals
from the 0 value.
t = (0:size(s,2)-1)/fs;
subplot 321, plot(t',s)
xlabel('time (seconds)'), ylabel('Magnitude')
%ylim([-350 250]);
xlim([0 10]);
```

Coding has been completed in MATLAB. Firstly, data of signals which are acquired from the Arduino part has been imported to the MATLAB environment. For this signal, a mat file which is called "veri9" has been used. After, for not having any missing value, a special command which is "isnan" has been used. This command checks for numbers. When reading data, if there is a NAN value, it is being deleted from our data by using this embedded function. Sample frequency which is "fs" in code and the time that is crucial for drawing a graph have been determined. As the last part, this information has been used for obtaining the wanted graph.

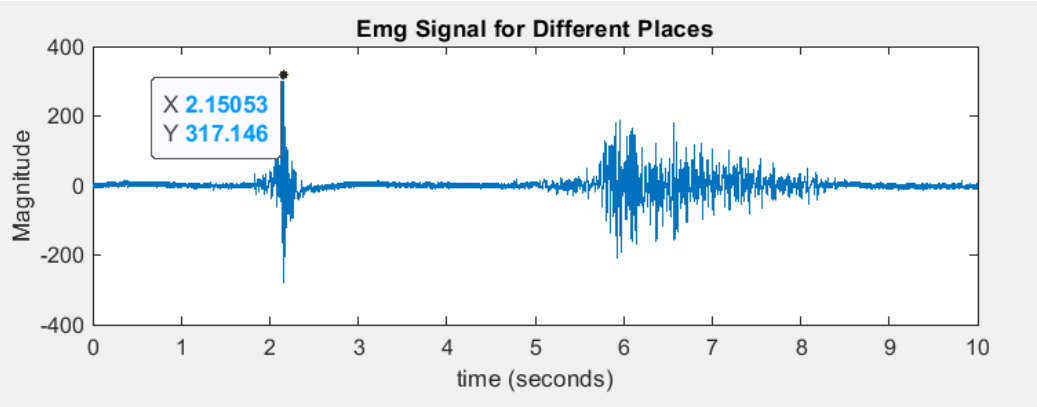


Figure 4.1.3.1 EMG Signal Graph for Different Places

The graph is shown in Figure 4.1.1.3 illustrate the EMG signal which is acquired from Arduino data. This data has been recorded for 10 seconds while arm muscle relaxing and contracting. For this graph, the maximum reached magnitude is 317.146. In addition, the number of data received in 10 seconds for this signal is 23656. Therefore, the sampling frequency (fs) is 2350.

4.1.4 Comparison of Three Graphs

TABLE 1. COMPARISON BETWEEN 3 GRAPHICS

Figure 4.1.1.1	**Figure 4.1.1.2**	**Figure 4.1.1.3**
Main graph.	Graph with high amplitude.	Graph of different place arm muscle.
Obtained EMG signal.	Obtained EMG signal.	Obtained EMG signal.
10 seconds signal.	10 seconds signal.	10 seconds signal.
Graph of arm muscle while relaxing and contracting.	Graph of arm muscle while relaxing and contracting.	Graph of arm muscle while relaxing and contracting.
The maximum reached magnitude is around 200.	The maximum reached magnitude is around 300.	The maximum reached magnitude is around 317.
This graph has the most noise (50 Hz).	This EMG graph has noise values between the other two graphs.	There is the least noise in this graph (50 Hz).

4.2 Noisy Graph by Shaking Arm Muscle

```
clc;
clear all;
load veri_shaking;
%load time1;
veri_shaking =
veri_shaking(~isnan(veri_shaking)); %If there is
any NAN value when reading data, we used it to
remove it from our data
s = veri_shaking;  fs = 2390; %Our sample number
is 23787 so that is how we defined fs
approximately.
s = s-mean(s); % We used it to align EMG signals
from the 0 value.
%t  = time1*(10^-6);
t = (0:size(s,2)-1)/fs;
subplot 321, plot(t',s)
xlabel('time (seconds)'), ylabel('Magnitude')
%ylim([-350 250]);
xlim([0 10]);
% FFT of the signal
```

```
F = abs(fft(s,fs+1));
Fs = F(1:fs/2+1)/(fs/2);

subplot 322, plot(0:fs/2,Fs)
xlabel('frequency (Hz)'), ylabel('Magnitude')
```

MATLAB was used to code everything. First, data from the Arduino component was imported into the MATLAB environment. A mat file called "veri shaking" was utilized to generate this signal. Following that, a specific command called "isnan" was used to ensure that there were no missing values. This command searches for numeric values. If there is a NAN value when reading data, it is erased from our data by utilizing this embedded function. The sample frequency, denoted by "fs" in code, as well as the time required to generate a graph, have been calculated. This data was utilized to generate the desired graph in the last section.

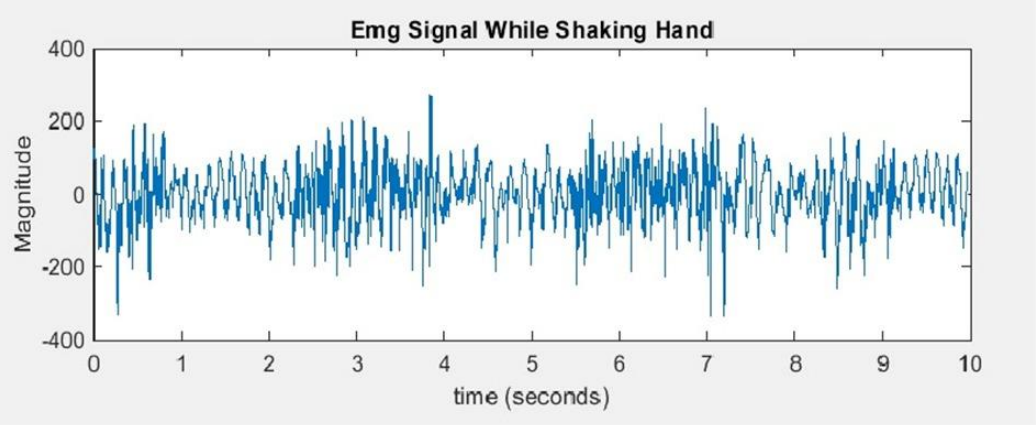


Figure 4.2.1 Shaking Hand Graph

Figure 4.2.1 depicts an EMG signal obtained from Arduino data. This data was collected for ten seconds as the arm shake. The greatest magnitude attained for this graph is around 250.

This graph has been obtained while shaking arm muscle. While recording, arm muscle was constructed so there is a noisy graph according to the signals which have shown in 4.1 Because of having a noisy graph and not filtering, a clear graph should not be obtained.

Filtering EMG Signals

5.1 Appliying Notch Filter and Fast Fourier Transform Graph

Matlab algorithm part:

```
clc;
clear all;
load veri18; %Loading the signal data
veri18 = veri18(~isnan(veri18)); %Finding and
deleting the NaN values
s = veri18;  fs = 2340; % EMG fs
s = s-mean(s); %Determining avarage of the values
and shifting down to make the center of  the
values at zero.
t = (0:size(s,2)-1)/fs; %Definition of time array

% FFT of the original signal
F = abs(fft(s,fs+1)); %Taking the fourier
Transform and their absolute values.
Fs = F(1:fs/2+1)/(fs/2); % In this line we delete
mirror image of fourier transform, +1 is for the
0Hz. in the plot.

% Notch Filtering
Wo = 50/(fs/2); %Cutoff Frequency f (50Hz)
converted to w (0.042).
BW = Wo/35; % increasing the number 35 decreases
the effect of filter, decreasing the number
increases the bandwidth
[b,a] = iirnotch(Wo,BW); %Numerator and
denominator coefficients were produced.
fvtool(b,a); title('Bode plot of Notch Filter
With cutoff 50Hz.'); %Shows bode plot of the
filter.
ns = filter(b,a,s); %Applying filter to the
signal.

% FFT of notch filtered signal
Fn = abs(fft(ns,fs+1));
Fns = Fn(1:fs/2+1)/fs;
%Plots
```

```
subplot 221, plot(t',s); title('Row Signal
(a)'); xlabel('Time (seconds)');
ylabel('Magnitude'); xlim([0 10]);
subplot 222, plot(0:fs/2,Fs); title('Fourier
Transform of Row Signal  (b)'); xlabel('Frequency
(Hz)'), ylabel('Magnitude');
subplot 223, plot(t',ns); title('The Signal After
The Notch Filter Was Applied,Cutoff is 50 Hz.
(c)'); xlabel('Time (seconds)');
ylabel('Magnitude'); ylim([-350 250]); xlim([0
10]);
subplot 224, plot(0:fs/2,Fns); title('Fourier
Transform of Filtered Signal  (d)');
xlabel('Frequency (Hz)'); ylabel('Magnitude');
```

When 10 second signal was recorded, it can be predicted that approximately where the constructions happen. However, there were obvious noises in most of the places and they block determining the exact construction time and their amplitudes. The noisy signal can be seen in figure 5.1.1. To see which frequency components are included in the row signal, the Fourier transform was applied by using “fft” function in MATLAB. In figure 5.1.2, all the frequency components vs. their amplitudes were plotted. There is a point which is greater than others. This point represents the 50Hz frequency component with magnitude of 22.86. In fact, this outcome was expected, because the places where measurements were done is not clear in terms of signals. For instance, there are cables inside of the walls and there are electronic devices around the place. All these cause a noise and most powerful one is the power line interface whose frequency is 50Hz in most of the countries. To get rid of this 50Hz signal, notch filter is the best filter option, because it only removes the chosen frequency component and mostly keep the others as it is. The bode plot of the notch filter (Fig. 5.1.5) shows how the frequency components will be affected from the filter. In figure 5.1.3, the signal was purified from the 50Hz noise and difference between Fig. 5.1.1 and Fig. 5.1.3 can be clearly seen. Also, Fourier Transform of filtered signal shows that the magnitude of 50Hz dropped to nearly one-fourth of its magnitude in row signal. To plot all the graphs in figure 5.1.1, figure 5.1.2,

figure 5.1.3, figure 5.1.4 and applying the notch filter, the codes above were used.

In the codes, first, the relevant data record was loaded and checked if there is any NaN or not. Because there may be some NaNs in our data, and they cause problem when running the m file. Therefore, these NaNs were determined and deleted. Since the record is 10 second, the sample frequency was calculated through dividing length of data by 10. Second, the center of the signal was needed to move to zero. Mean value of the data was calculated and the signal was shifted down amount mean value. Then, the Fourier Transform of the signal was get and its absolute value computed. However, the "fft" function gives a mirrored image and that mirrored part was removed. In filtering part, the cut-off frequency was assigned as 50 to decrease the effect of the 50Hz. noise. As the row signal, the notch filtered signal's Fourier Transform was computed. And last, the row signal, filtered signal and their Fourier Transforms were plotted.

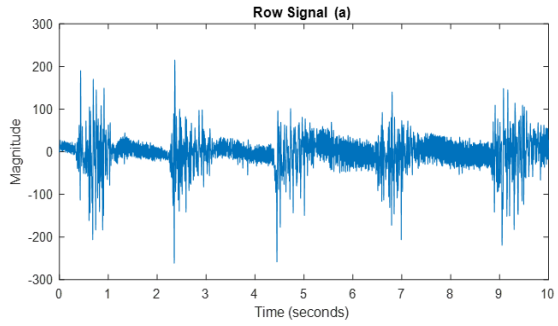


Figure 5.1.1 Figure caption

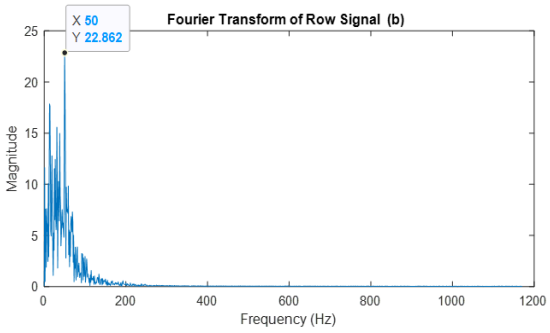


Figure 5.1.2 Figure caption

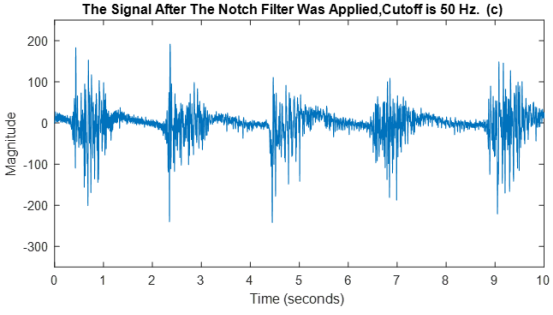


Figure 5.1.3 Figure caption

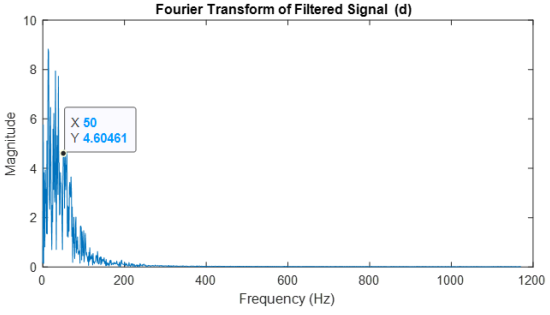


Figure 5.1.4 Figure caption

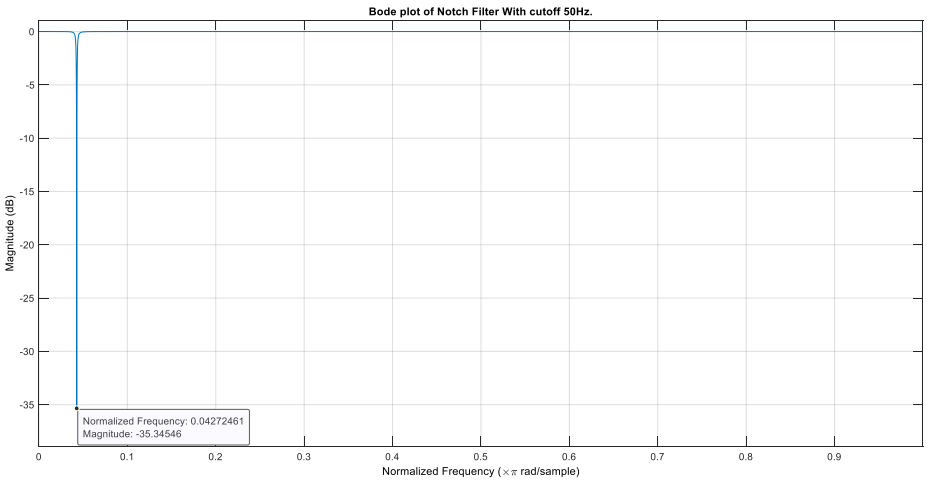


Figure 5.1.5 Figure caption

5.1.1 Hand Shaking Filtering

It is aimed to reduce the noise in the signal by applying notch and butterworth filters for the handshaking EMG signal.

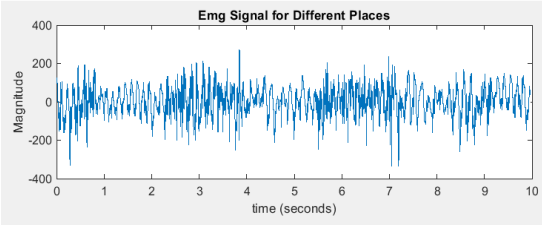


Figure 5.1.1.1 Raw Filter for Hand Shaking

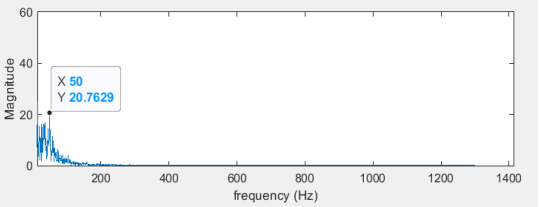


Figure 5.1.1.2 Raw Fft Filter for Hand Shaking

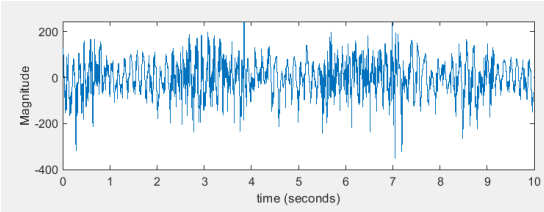


Figure 5.1.1.3 Notch Filter for Hand Shaking

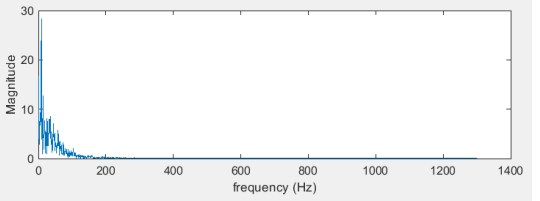


Figure 5.1.1.4 Notch Fft Filter for Hand Shaking

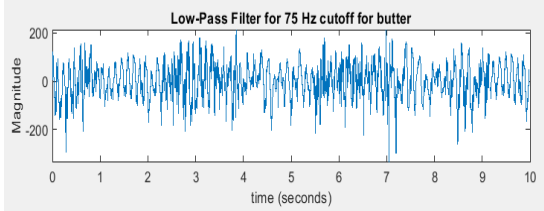


Figure 5.1.1.5 Low-pass Filter

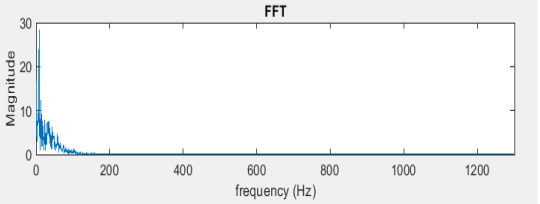


Figure 5.1.1.6 Low-pass Fft Filter

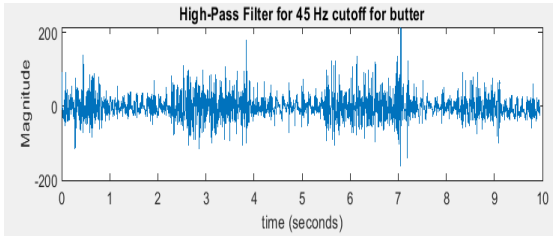


Figure 5.1.1.7 High-pass Filter

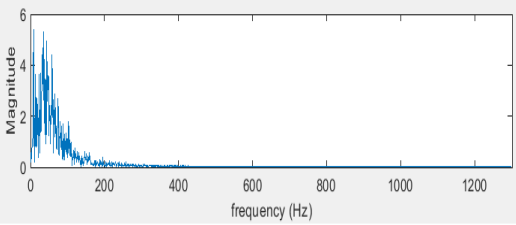


Figure 5.1.1.8 High-pass Fft Filter

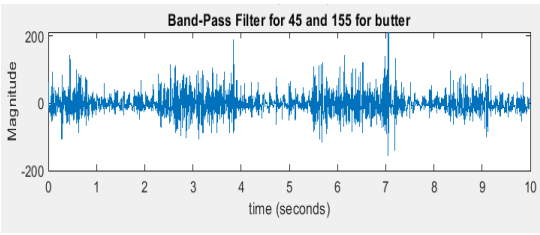


Figure 5.1.1.9 Band-pass Filter

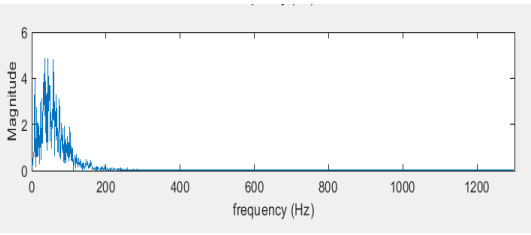


Figure 5.1.1.10 Band-pass Fft Filter

Shaking arm signals cause artificial artifacts and therefore the resulting signal sounds louder than normal. However, when filtering is done, especially when the band-pass Butterworth filter is applied, the signals become more pronounced when noise is eliminated. As seen in Figure 5.1.1.6, it can be seen approximately where the contraction is or where the relaxation is.

5.1.2 Different Place Filtering (Forearm)

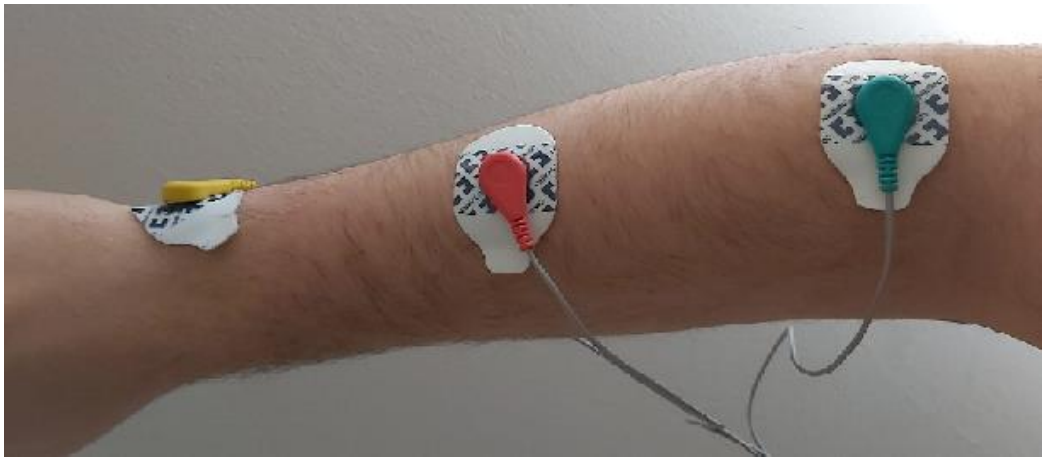


Figure 5.1.2.1 Electrode Places for Upper Forearm

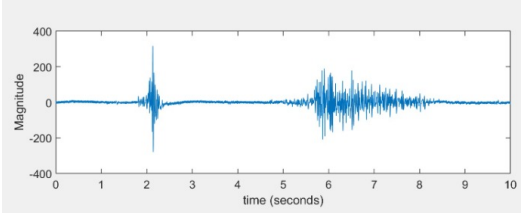


Figure 5.1.2.2 Raw Filter for Forearm

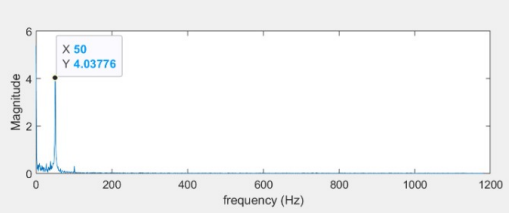


Figure 5.1.2.3 Raw Fft Filter for Forearm

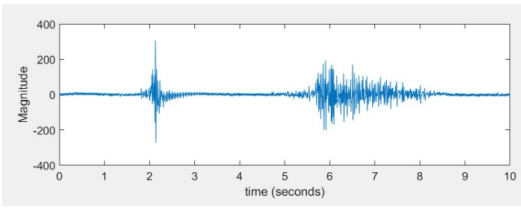


Figure 5.1.2.4 Notch Filter for Forearm

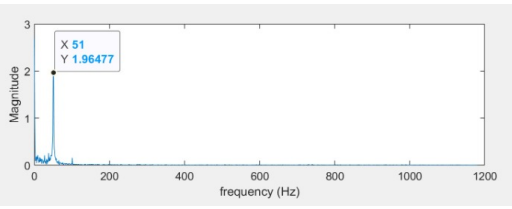


Figure 5.1.2.5 Raw Fft Filter for Forearm

After applying Notch Filter, new filter which is "Butterworth" has been used to get more clear results. To compare results, Low-Pass Filter, High-Pass Filter, Band-Pass Filter has been used.

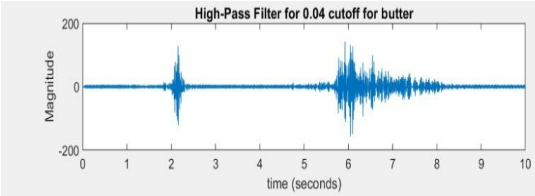


Figure 5.1.2.6 High-pass Filter

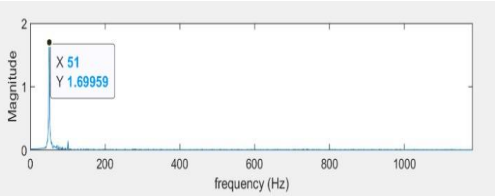


Figure 5.1.2.7 High-pass Fft Filter

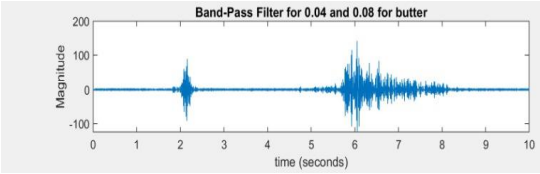


Figure 5.1.2.8 Band-pass Filter

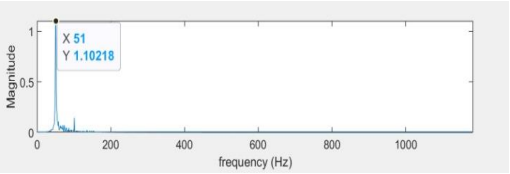


Figure 5.1.2.9 Band-pass Fft Filter

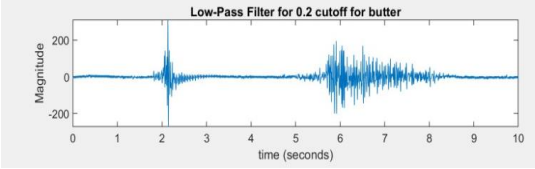


Figure 5.1.2.10 Low-pass Filter

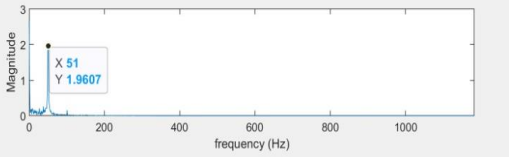


Figure 5.1.2.11 Low-pass Fft Filter

As can be seen from Figure 5.1.2.8 in the signals received from the upper wrist, the magnitude of the contractions is not strong because it is said that the upper arm muscle is weaker than the other muscles.

5.2 Appliying Different Types of Filter and Cut-off Frequencies

Ideal filters and their optimum cut-off frequency values determined by mainly testing. It is known that if the degree of a filter increases the sharpen it will be. Thus, the degree of the filters increased until the bode plots of each filter deteriorated as shown in the Figure 5.2.1. When the bode plot of it deteriorate, it meant that the degree of filter was not ideal.

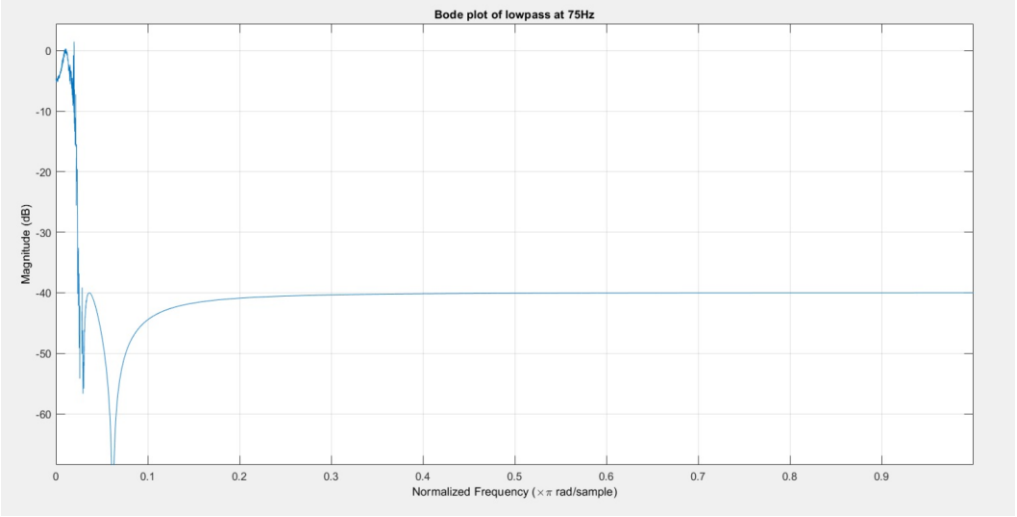


Figure 5.2.1 Example of Deteriorated Signal

5.2.1 Butterworth Filtering

Low-pass: For Butterworth low-pass 10th order filter used because of the deteriorated in the bode plot after starts from 11th order.

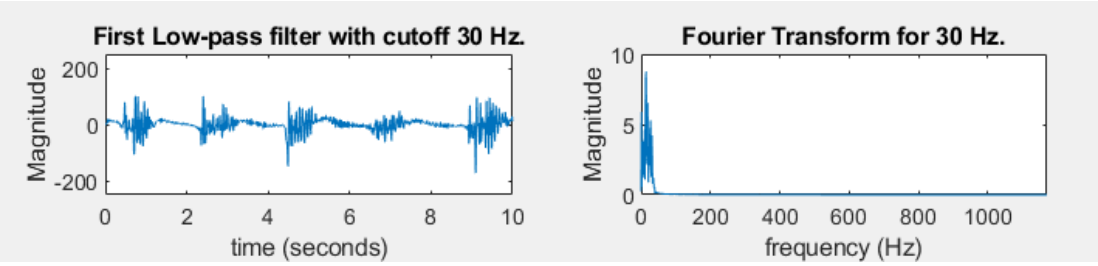


Figure 5.2.1.1 Low-pass Filter for 30 Hz

Figure 5.1.2.2 Low-pass Fft Filter for 30 Hz

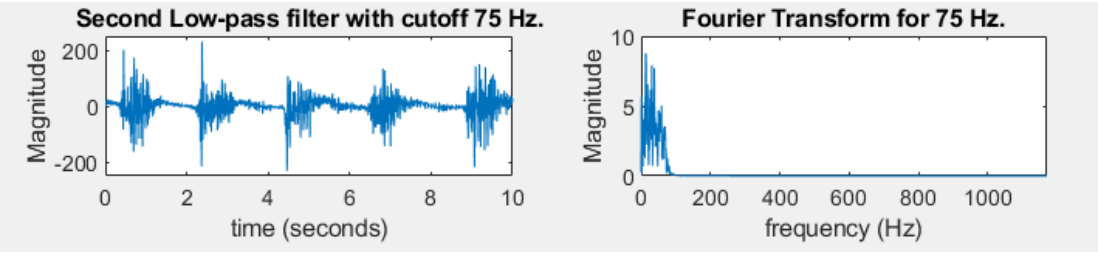


Figure 5.2.1.3 Low-pass Filter for 75 Hz

Figure 5.1.2.4 Low-pass Fft Filter for 75 Hz

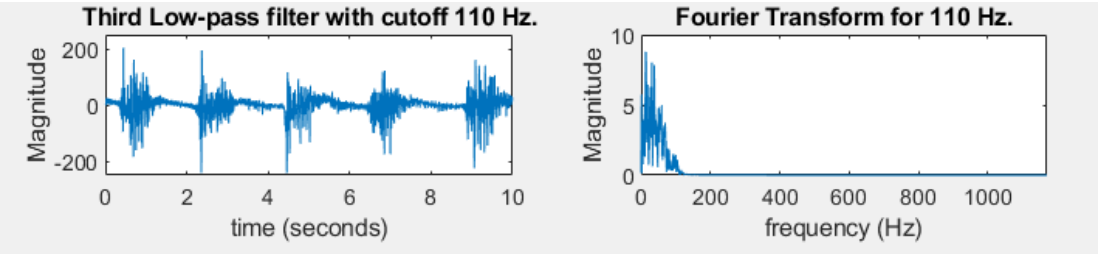


Figure 5.2.1.5 Low-pass Filter for 110 Hz

Figure 5.1.2.6 Low-pass Fft Filter for 110 Hz

As seen in the Figure 5.2.1.1, Figure 5.2.1.3, and Figure 5.2.1.5 three different cut-offs compared to each other to obtain accurate result. In the Fig. 5.2.1.1, because of the loss of amplitude, 30Hz is not a good option for the low-pass Butterworth. In comparison of 75Hz (Fig. 5.2.1.3) and 110Hz (Fig. 5.2.1.5), it can be said that the 75Hz low pass is better because the signal seems thinner

than the 110Hz case. At the end, 75Hz cut-off frequency was used as a main one.

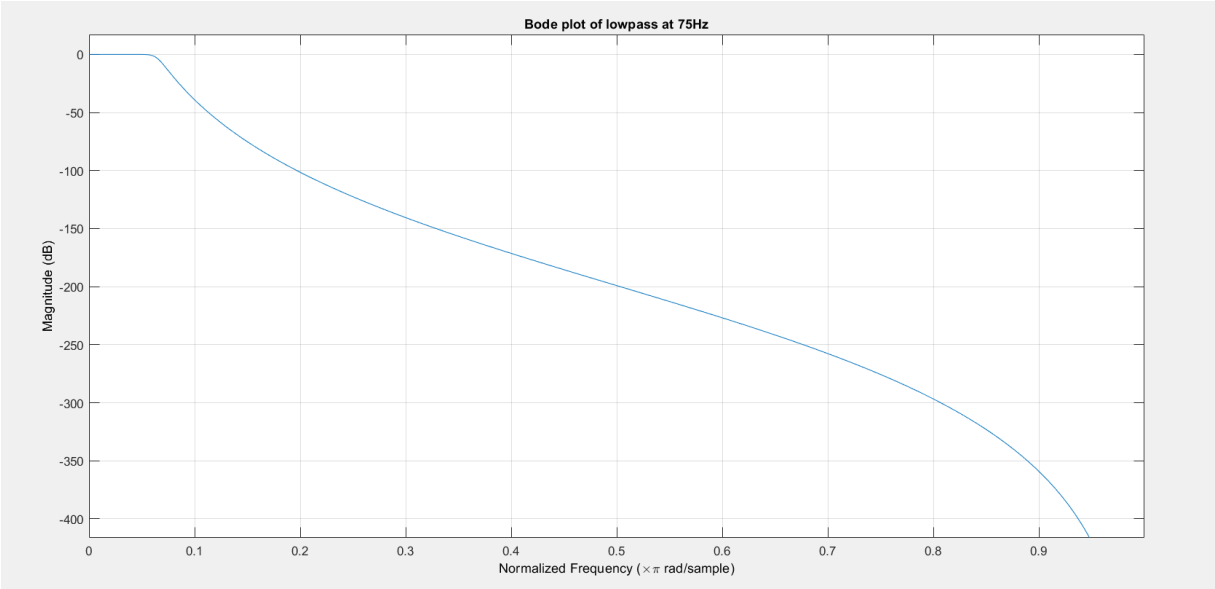


Figure 5.2.1.7 Bode Plot of 75Hz Butterworth Low-pass Filter

High-pass: In high-pass filter of Butterworth, also 8th order filter used for the same reasons with low-pass.

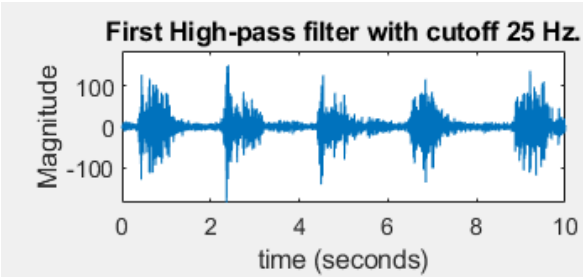


Figure 5.2.1.8 High-pass Filter for 25 Hz

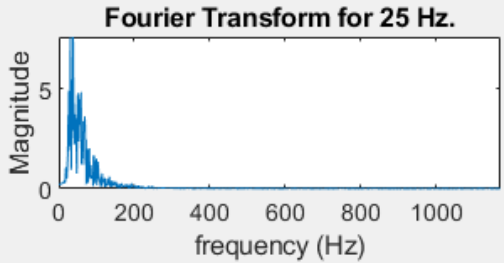


Figure 5.1.2.9 High-pass Fft Filter for 25 Hz

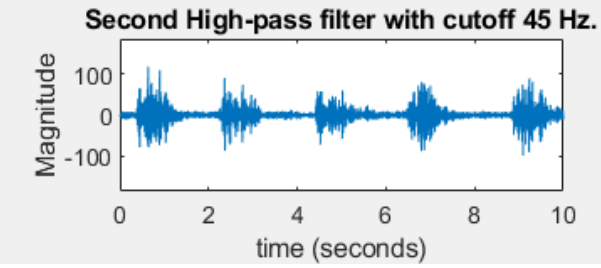


Figure 5.2.1.10 High-pass Filter for 45 Hz

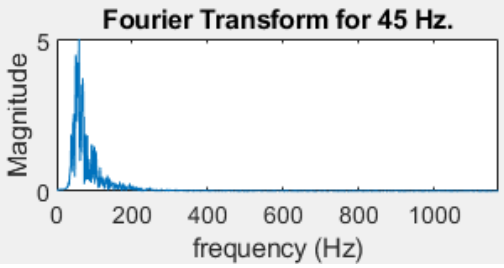
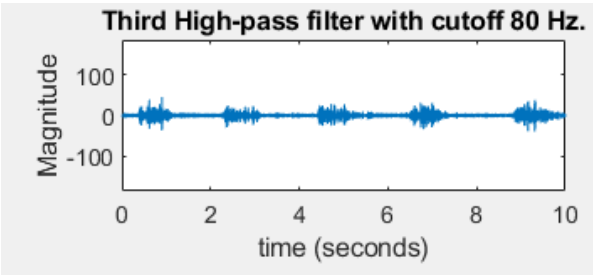


Figure 5.1.2.11 High-pass Fft Filter for 45 Hz



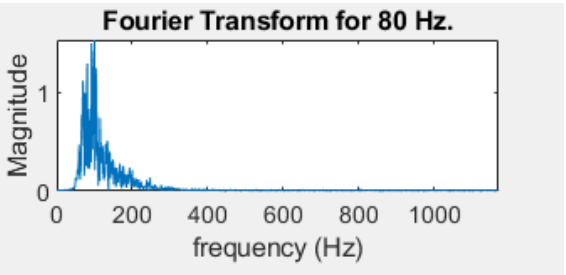


Figure 5.2.1.12 High-pass Filter for 80 Hz

Figure 5.1.2.13 High-pass Fft Filter for 80 Hz

As a high-pass filter cutt-of frequencies, also three different values compared to each other which are 25Hz, 45Hz and 80Hz. Then, 45Hz cut-off frequency was used as a high-pass filter frequency. The reason behind it is the fact that our EMG signal mainly starts at 50 Hz. That is why the 80Hz (Fig. 5.2.1.12) high-pass filter causes decreasing in the magnitude of the signal and the 25Hz passes the noise between 45-25Hz.

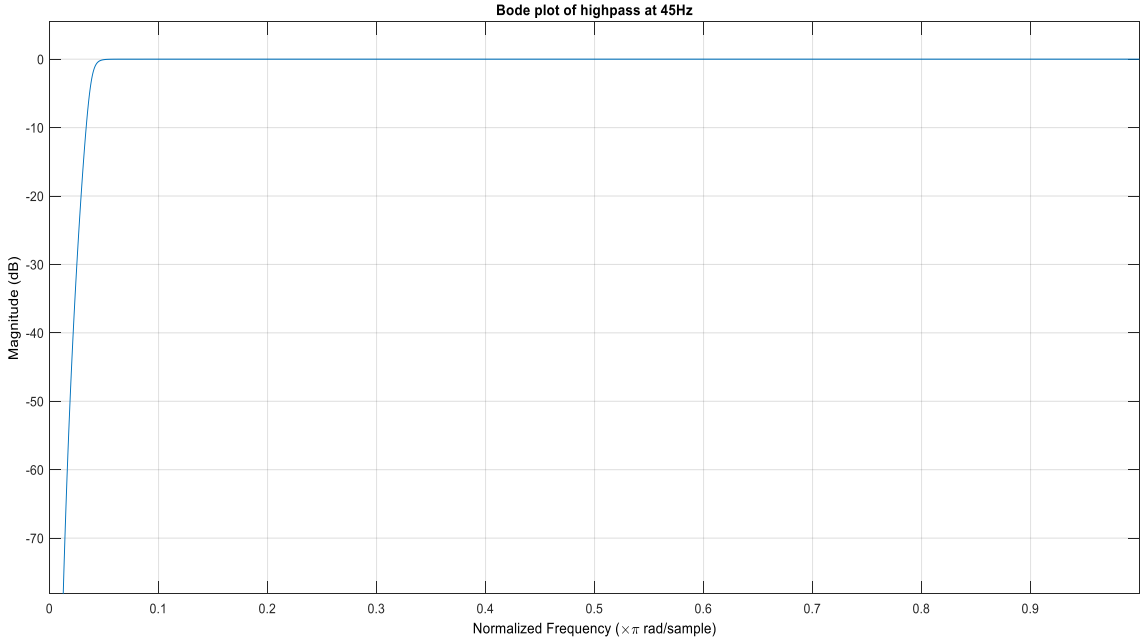
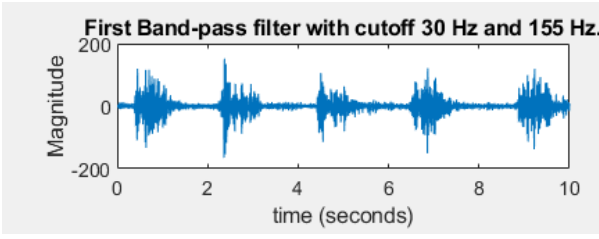


Figure 5.2.1.14 Bode Plot of 45 Hz Butterworth High-pass Filter

Band-pass: Lastly, because the bode plot of these filter starts to deteriorate at 8^{th} order, 7^{th} order filters used for band-pass Butterworth filters.



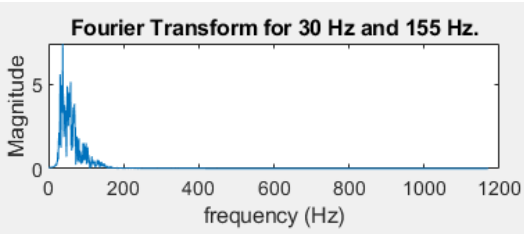


Figure 5.2.1.15 Band-pass Filter for 30 and 155 Hz

Figure 5.1.2.16 High-pass Fft Filter for 30 and 155 Hz

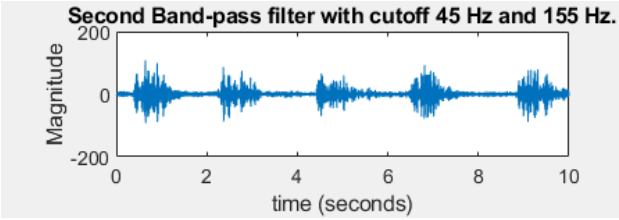


Figure 5.2.1.17 Band-pass Filter for 45 and 155 Hz

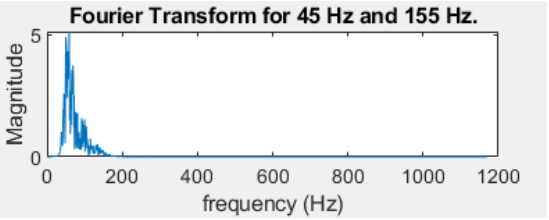


Figure 5.1.2.18 High-pass Fft Filter for 45 and 155 Hz

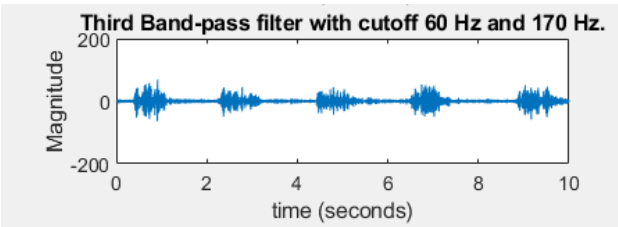


Figure 5.2.1.19 Band-pass Filter for 60 and 170 Hz

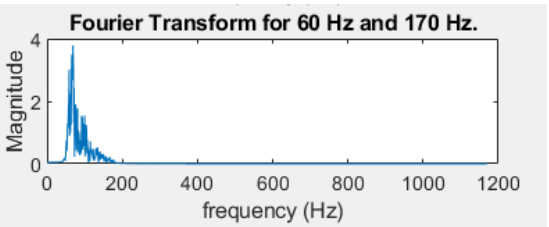


Figure 5.1.2.20 High-pass Fft Filter for 60 and 170 Hz

After these comparisons between 30Hz-155Hz, 45Hz-155Hz, 60Hz-170Hz, 45Hz-155Hz bandwidth was used. The 45-155 was chosen as the best bandwidth for the band-pass Butterworth filter. The reason for choosing 45Hz to 155Hz is because our EMG signal mainly exist between 50 to 150Hz. For instance, the magnitudes of the signal were pretty affected from the 60Hz and decreased in the Figure 5.2.1.19. The difference between the Figure 5.2.1.15 and Figure 5.2.1.17 is that the small constructions can be seen better in the Figure 5.2.1.17. Some of these small constructions can be seen between third and fourth construction, between 5-6 seconds, in the Figure 5.2.1.17.

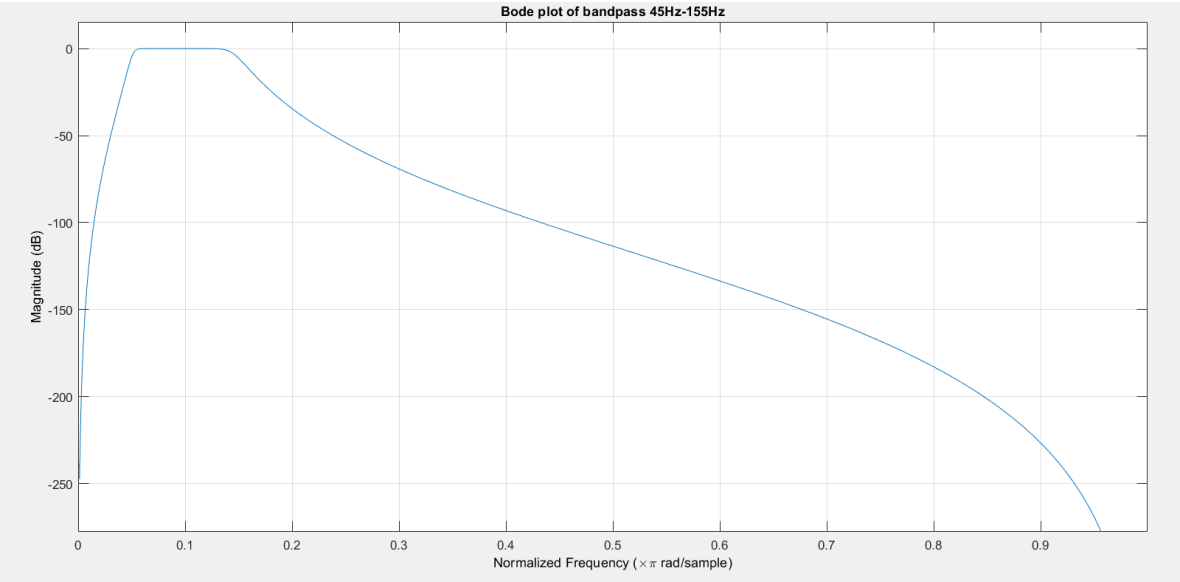


Figure 5.2.1.21 Bode Plot of 45 Hz and 155 Hz Butterworth Band-pass Filter

5.2.2 Chebyshev Filtering

Low-pas: For low-pass Chebyshev filter 9th order filter used due to deteriorations which starts at 10th order.

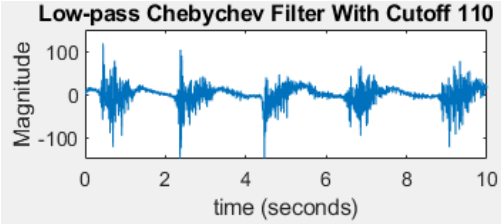


Figure 5.2.2.1 Low-pass Filter for 110 Hz

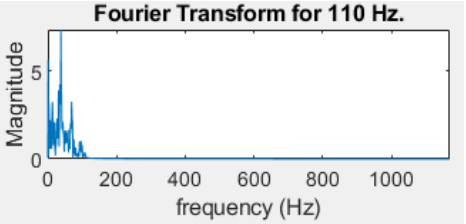


Figure 5.2.2.2 Low-pass Fft Filter for 110 Hz

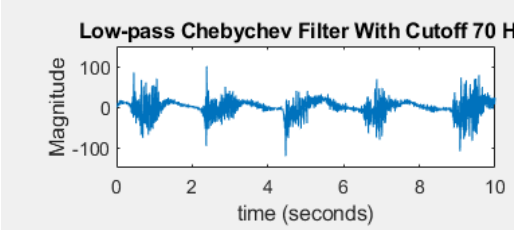


Figure 5.2.2.3 Low-pass Filter for 70 Hz

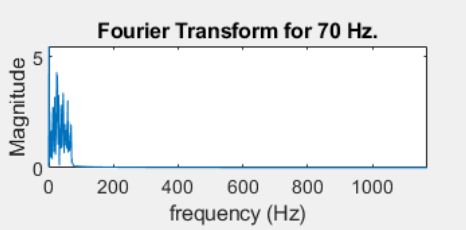


Figure 5.2.2.4 Low-pass Fft Filter for 70 Hz

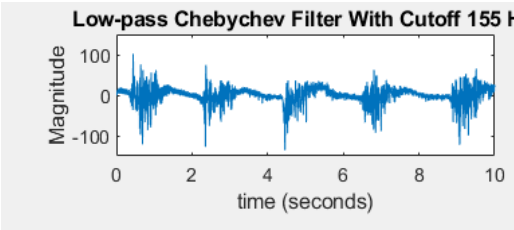


Figure 5.2.2.5 Low-pass Filter for 155 Hz

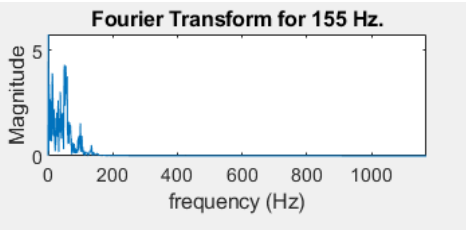


Figure 5.2.2.6 Low-pass Fft Filter for 155 Hz

As it can be seen in the Figure 5.2.2.1 there is a noise peak at 38Hz but in the Figure 5.2.2.3 the noise element is suppressed due to the shape of bode plot for the 70Hz cut-off frequency. The signal in Figure 5.2.2.5 is noisy when there is no construction. That is why 70 Hz cut-off frequency was chosen as best option.

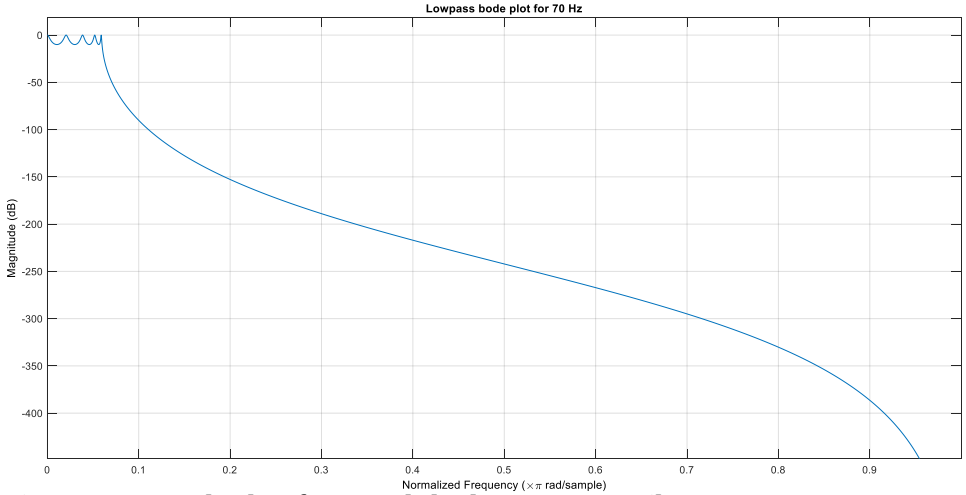


Figure 5.2.2.7 Bode Plot of 70 Hz Chebyshev Low-pass Filter

High pass: This part of Chebyshev filters, also 9th order filter used due to the same deterioration problems occurs at 10th order.

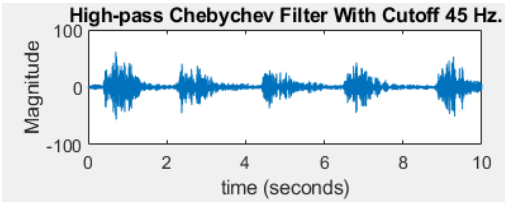


Figure 5.2.2.8 High-pass Filter for 45 Hz

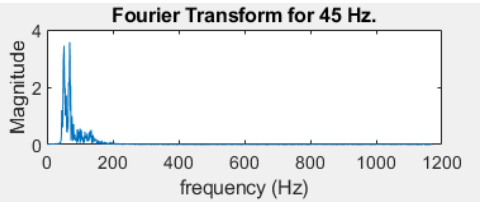


Figure 5.2.2.9 High-pass Fft Filter for 45 Hz

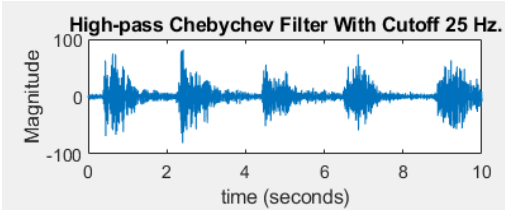


Figure 5.2.2.10 High-pass Filter for 25 Hz

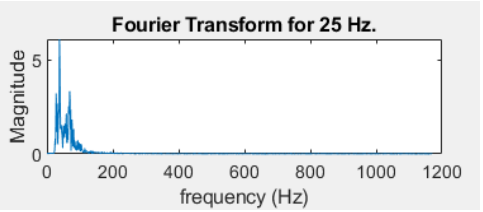


Figure 5.2.2.11 High-pass Fft Filter for 25 Hz

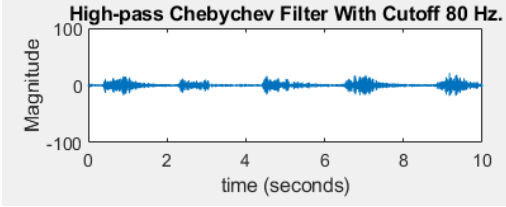


Figure 5.2.2.12 High-pass Filter for 80 Hz

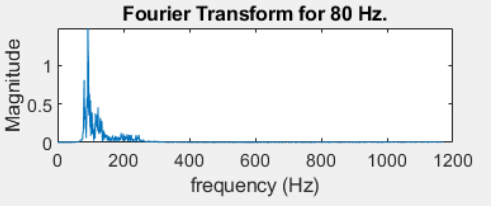


Figure 5.2.2.13 High-pass Fft Filter for 80 Hz

Because the EMG signal mainly occurs in between 50 - 150Hz, choosing 80Hz as cut-off frequency of high-pass filter affects the amplitudes badly and decrease

it (Fig. 5.2.2.12). There is not so much difference between Figure 5.2.2.8 and Figure 5.2.2.10 in terms of amplitude but the low-pass filter which has 45Hz cut-off is better than 25Hz cut-off. Therefore, analyzing the signal in Figure 5.2.2.8 will be easier than the signal in Figure 5.2.2.10. Through the comparisons, 45Hz cut-off frequency were chosen.

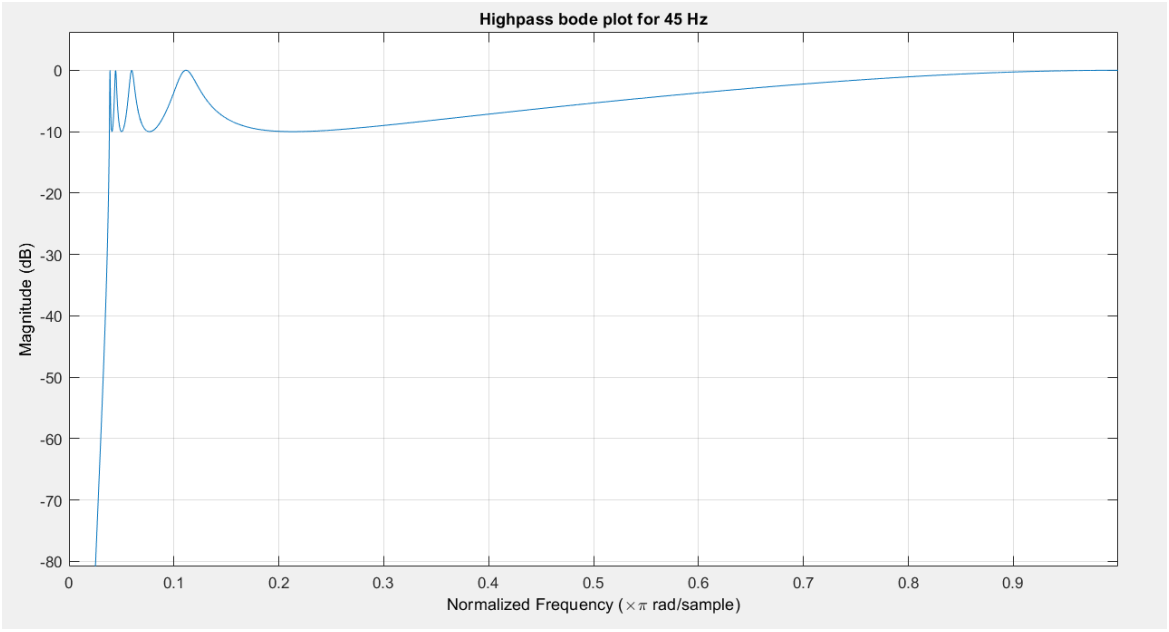


Figure 5.2.2.14 Bode Plot of 45 Hz Chebyshev High-pass Filter

Band-pass: Lastly, for Chebyshev band-pass filters 6th order was used due to deteriorations which starts at 7th order.

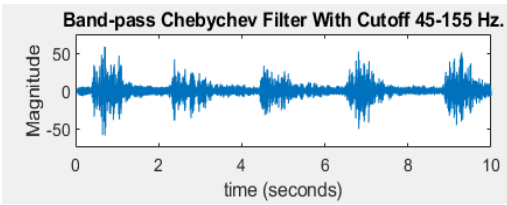


Figure 5.2.2.15 Band-pass for 45 and 155 Hz

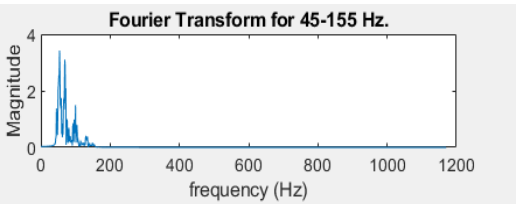


Figure 5.2.2.16 High-pass Fft for 45 and 155 Hz Hz

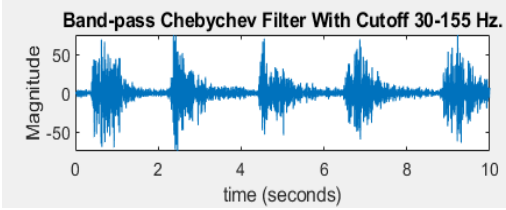


Figure 5.2.2.17 Band-pass for 30 and 155 Hz

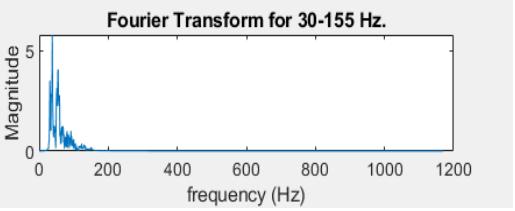


Figure 5.2.2.18 High-pass Fft for 30 and 155 Hz Hz

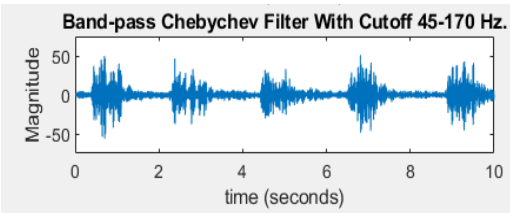


Figure 5.2.2.19 Band-pass for 45 and 170 Hz

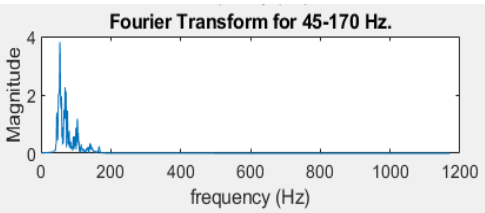


Figure 5.2.2.20 High-pass Fft for 45 and 170 Hz Hz

In Figure 5.2.2.15, there is an important noise element which has amplitude of 5. This noise element was decresed in the Figure 5.2.2.15 by chosing the cut-off frequencies as 45-155. In addition, the magnitude of the singal in Figure 5.2.2.15 is higher than the magnitude of the singal in the Figure 5.2.2.19. That is why 45 Hz to 155 Hz bandwidth decided to used as best option for the band-pass chebychev filter.

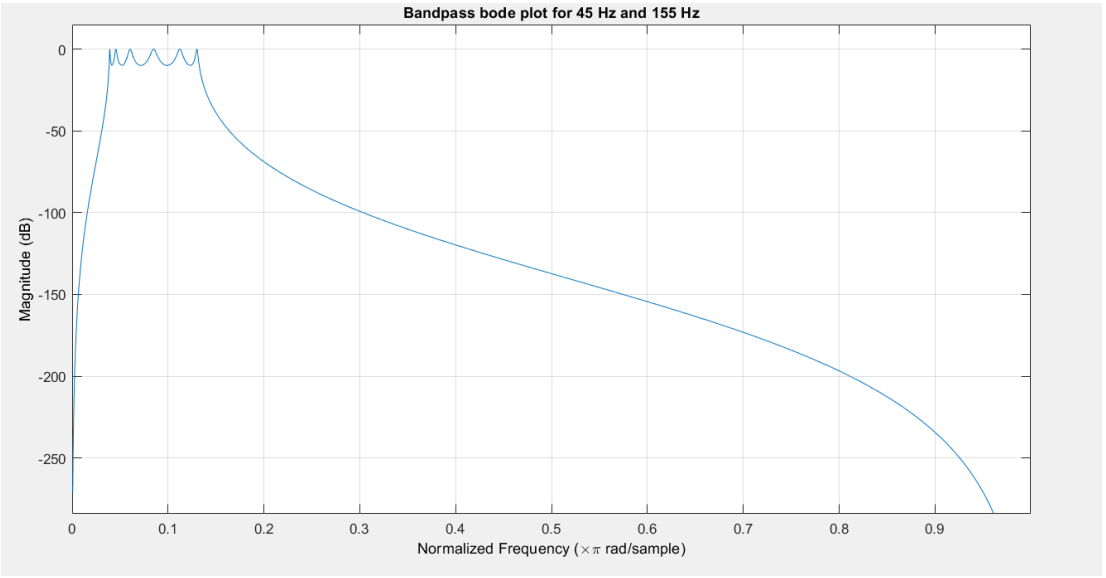


Figure 5.2.2.21 Bode Plot of 45 to 155 Hz Chebyshev Band-pass Filter

5.2.3 Elliptic Filtering

For all elliptic filters has ripple that has 1dB peak-to-peak. Because when the peak-to-peak dB of a ripple increase the passband of the any filter gets more unstable in terms of passing capability.

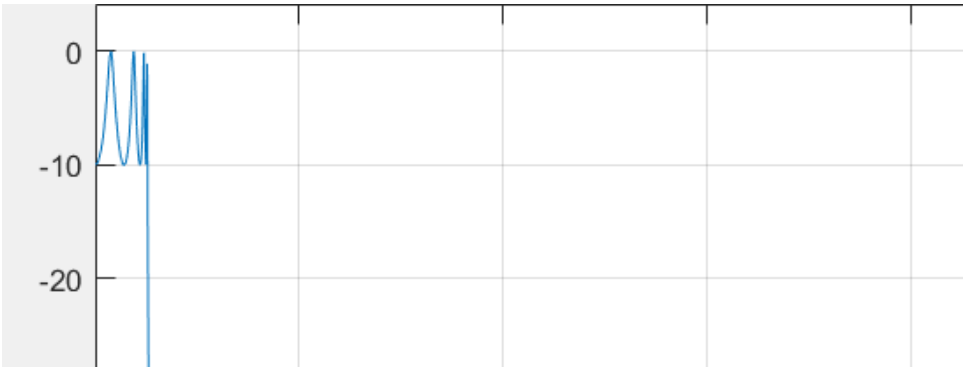


Figure 5.2.3.1 Example of 10dB Peak-to-Peak Ripple

Figure 5.2.3.2 Example of 1dB Peak-to-Peak Ripple

In addition to that, it seen that until 70dB stopband attenuation stopband capability has been increased. After 70dB even though stopband capability increased, the decay of filter has been increased as well. Thus, 70dB stopband attenuation decided to use in Elliptic filters.

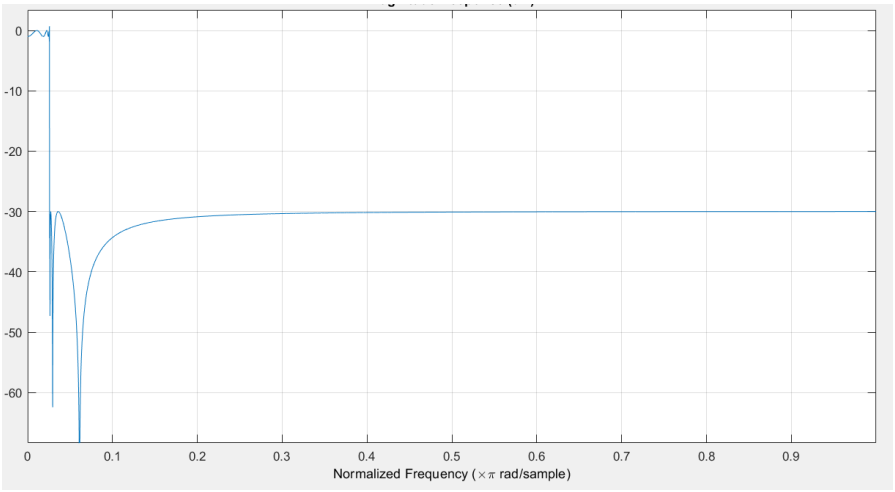


Figure 5.2.3.3 Example of 30dB Stopband Attenuation

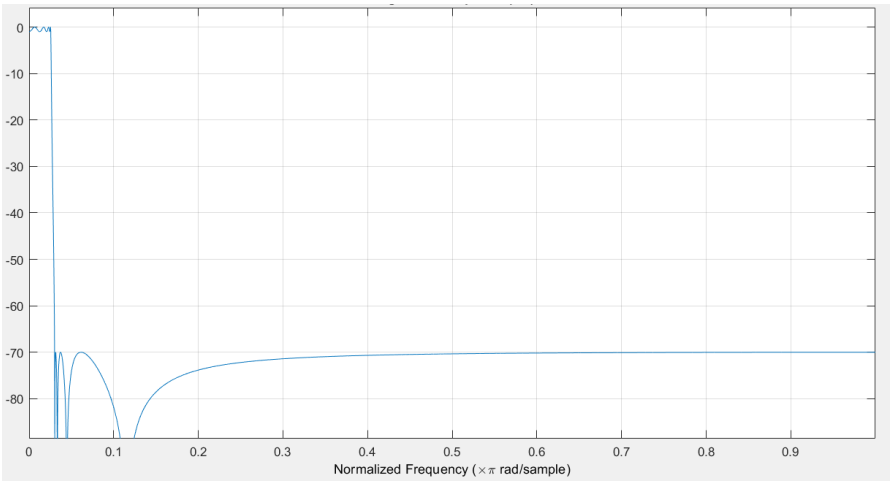


Figure 5.2.3.4 Example of 70dB Stopband Attenuation

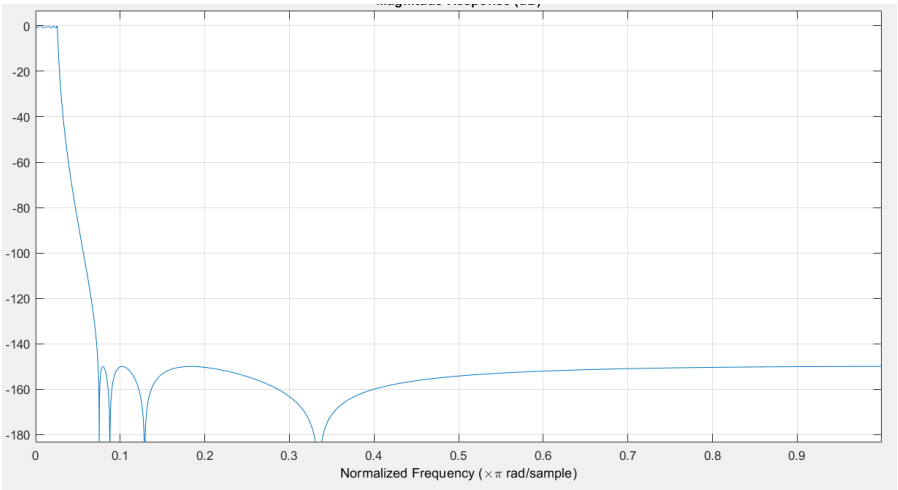


Figure 5.2.3.5 Example of 150dB Stopband Attenuation

Low-pass: As low-pass Elliptic filter, 8^{th} order filters used because of the same deterioration problem.

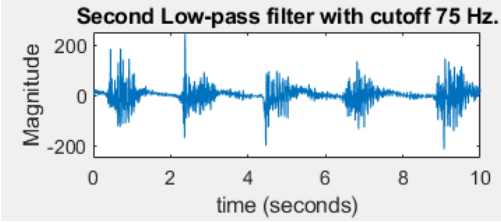


Figure 5.2.3.6 Low-pass Filter for 75 Hz

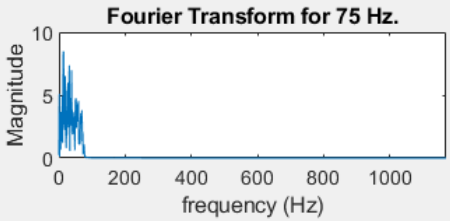


Figure 5.2.3.7 Low-pass Fft Filter for 75 Hz

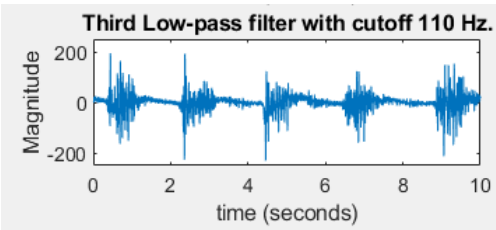


Figure 5.2.3.8 Low-pass Filter for 110 Hz

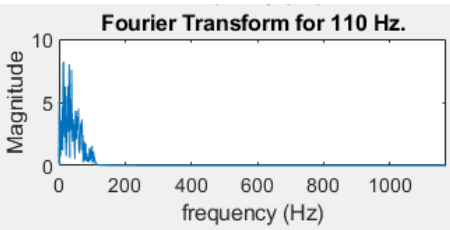


Figure 5.2.3.9 Low-pass Fft Filter for 110 Hz

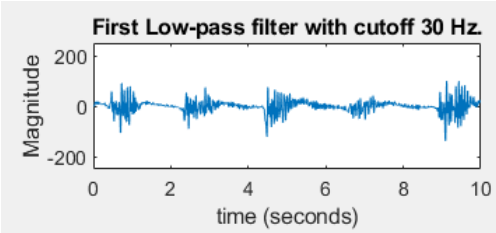


Figure 5.2.3.10 Low-pass Filter for 30 Hz

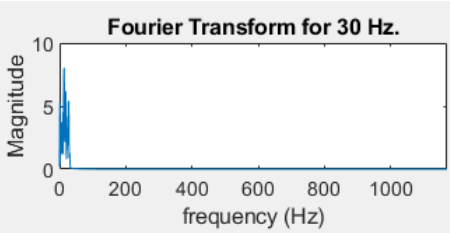


Figure 5.2.3.11 Low-pass Fft Filter for 30 Hz

The frequencies between 50 and 150 Hz are important for observing an EMG signal. Its importance can be seen in the Figure 5.2.3.10. Choosing the cut-off frequency as 30 Hz leaves mostly the noise elements. Thus, a bit high cut-off for the low-pass elliptic filter is better. Yet, increasement in the cut-off may not be helpful after some value. For instance, the signal in Figure 5.2.3.6 is better looking than the signal in Figure 5.2.3.8. The comparisons shows that 75 Hz cut-off frequency is the optimum one.

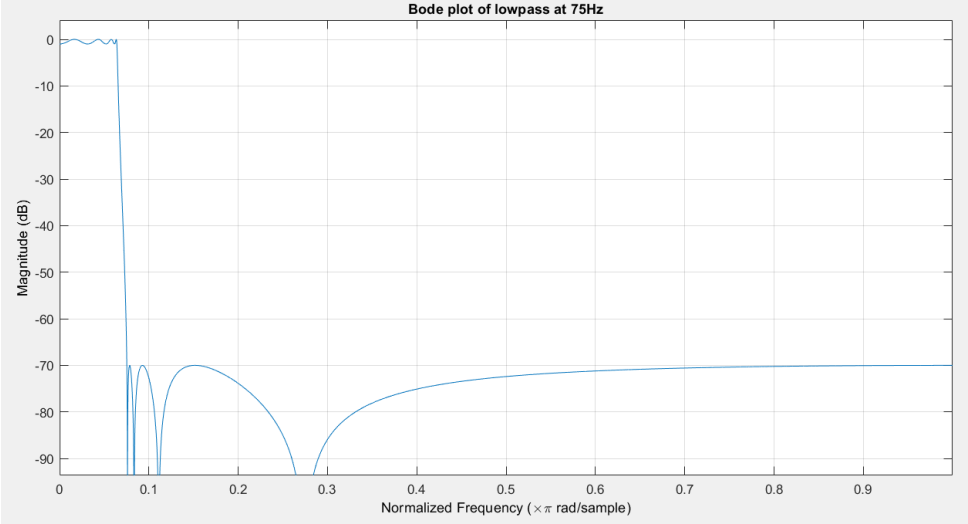


Figure 5.2.3.12 Bode Plot of 75 Hz Elliptic Low-pass Filter

High-pass: For high-pass filter, also 8th order filters used because of the deterioration which occurs after 8th order filters.

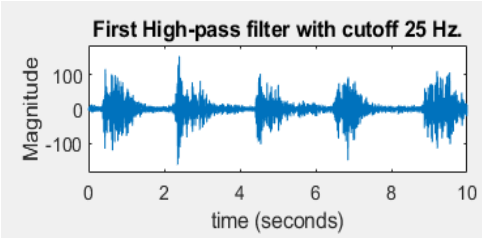


Figure 5.2.3.13 High-pass Filter for 25 Hz

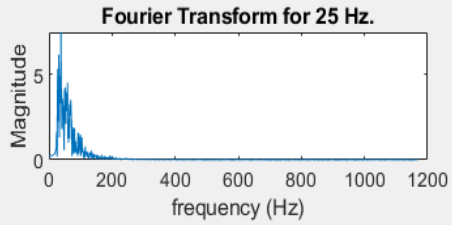


Figure 5.2.3.14 High-pass Fft Filter for 25 Hz

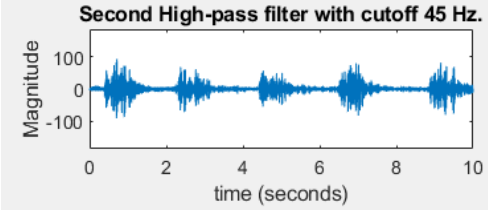


Figure 5.2.3.15 High-pass Filter for 45 Hz

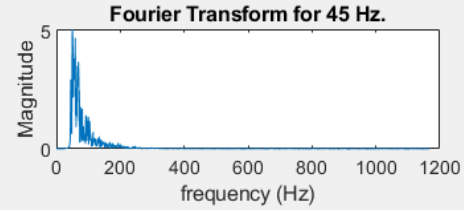


Figure 5.2.3.16 High-pass Fft Filter for 80 Hz

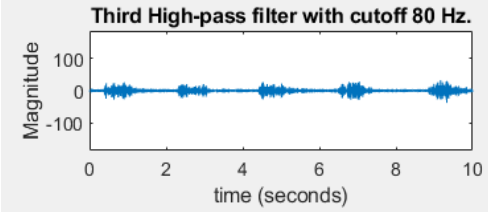


Figure 5.2.3.17 High-pass Filter for 80 Hz

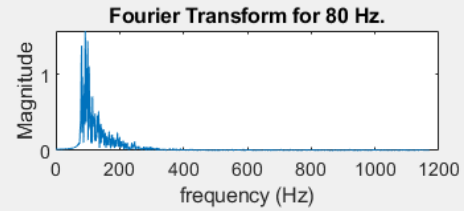


Figure 5.2.3.18 High-pass Fft Filter for 80 Hz

Cut-off frequency and magnitude of the signal are like two side of seesaw. Increase in one side decreases other side. That is why 80Hz is not an optimum option. Even the magnitude of the signal in Figure 5.2.3.13 is greater than the magnitude in Figure 5.2.3.15, the signal in Figure 5.2.3.15 has more clear view.

It can be concluded that the noise elements were removed by choosing cut-off as 45Hz.

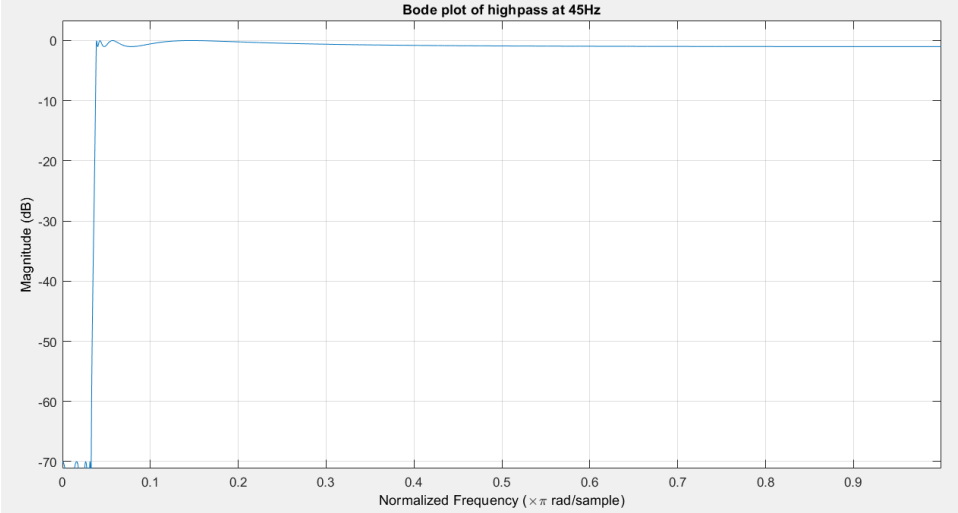


Figure 5.2.3.19 Bode Plot of 45 Hz Elliptic High-pass Filter

Band-pass: 5th order filter used as order of Elliptic band-pass filter, the reason behind it is the deterioration which occur after 5th order.

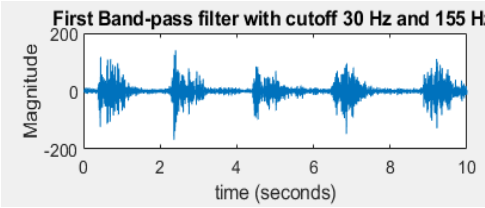


Figure 5.2.3.20 Band-pass for 30 and 155 Hz

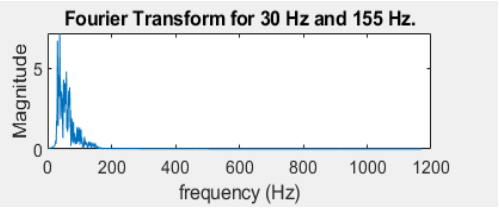


Figure 5.2.3.21 Band-pass Fft for 30 and 155 Hz

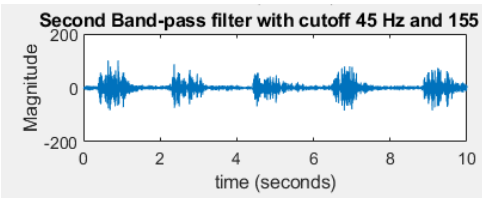


Figure 5.2.3.22 Band-pass for 45 and 155 Hz

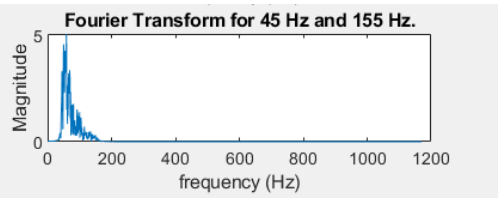


Figure 5.2.3.23 Band-pass Fft for 45 and 155 Hz

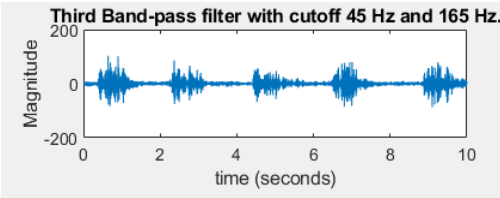


Figure 5.2.3.24 Band-pass for 45 and 165 Hz

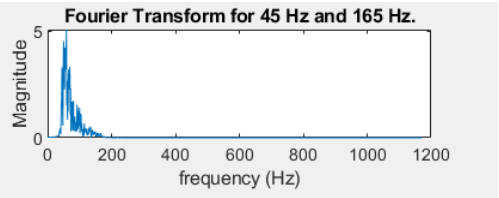


Figure 5.2.3.25 Band-pass Fft for 45 and 165 Hz

Because the filter is not ideal and the EMG signals are generally exist between 50 and 150Hz, choosing 45 and 155Hz as cut-off frequencies for the band-pass elliptic filter is the best option.

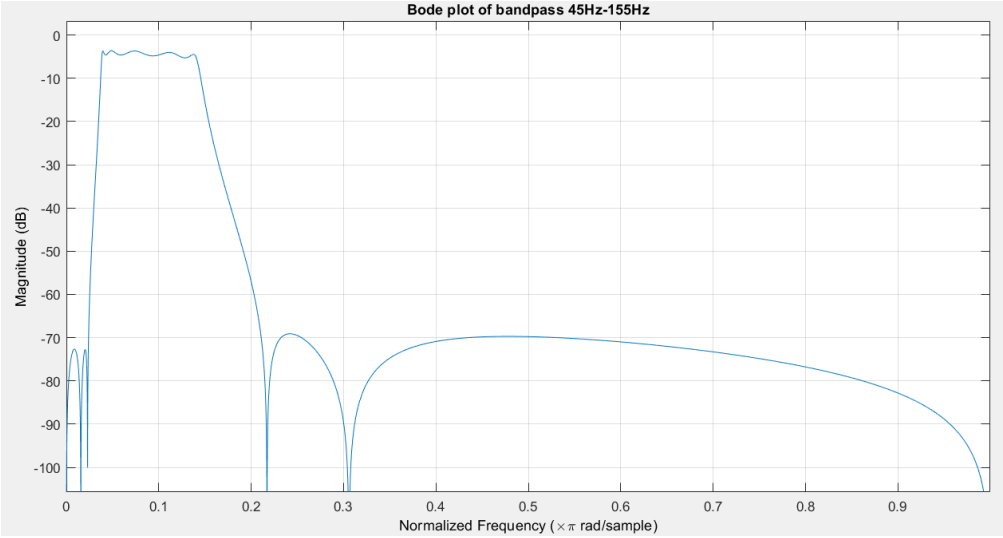


Figure 5.2.3.26 Bode Plot of 45 to 155 Hz Elliptic Band-pass Filter

General Overview for Filtering

Butterworth, Chebyshev, and Elliptic filters compared to each other to observe best filter to obtain better EMG signal. At the end, Elliptic filter was used to obtain accurate EMG signal because of the sharpness of the filter type. In addition to that, while deciding cut-off frequencies, the fact that EMG signals mainly exist in 50Hz to 150Hz and noise components exist mainly in 0 to 25Hz, which is motion artifacts, was considered. That is why in all band-pass filters 45 to 155Hz bandwidth used, in high pass filters 45 Hz used. When it comes to low-pass filters, these kinds of filters are not optimal to use it for signals which the noise components have lower frequencies than main signal components. Because main purpose of using filters is to passing main signal components, while suppressing the noises and in low pass filter it is impossible to pass higher frequencies, while suppressing the lower ones. Best filter type to use it for obtaining precise EMG signal is Band-pass filters because it satisfies the main purpose of filtering process more than high-pass filters.

Matlab Codes For Filtering:

```
clc;
clear all;
load veri18;
veri18 = veri18(~isnan(veri18));
s = veri18;  fs = 2340; % EMG fs
s = s-mean(s);
t = (0:size(s,2)-1)/fs;
% FFT of the signal
F = abs(fft(s,fs+1));
Fs = F(1:fs/2+1)/(fs/2);
% Notch Filter
Wo = 50/(fs/2);
BW = Wo/35;
[b,a] = iirnotch(Wo,BW);
fvtool(b,a); title("Bode Plot of Notch Filter")
ns = filter(b,a,s);
% FFT of filtered signal
Fn = abs(fft(ns,fs+1));
Fns = Fn(1:fs/2+1)/fs;
%Raw Graph
subplot 321, plot(t',s);
xlabel('time (seconds)'),
ylabel('Magnitude'),title("Raw Signal")
xlim([0 10]);
% FFT Graph
subplot 322, plot(0:fs/2,Fs) ;
xlabel('frequency (Hz)'),
ylabel('Magnitude'),title("FFT of Raw Signal")
%Notch Filtered Signal
subplot 323, plot(t',ns);
xlabel('time (seconds)'),
ylabel('Magnitude'),title("Notch Filtered
Signal")
xlim([0 10]);
%FFT of Notch Filtered Signal
subplot 324, plot(0:fs/2,Fns),
xlabel('frequency (Hz)'),
ylabel('Magnitude'),title("FFT of Notch Filtered
Signal")
%BUTTERWORTH%
% Cutoff freq.'s for lowpass
```

```
but_fc_lp_1  = 30;
but_fc_lp_2  = 75;
but_fc_lp_3  = 110;
% Cutoff freq.'s for highpass
but_fc_hp_1 = 25;
but_fc_hp_2 = 45;
but_fc_hp_3 = 80;
% Cutoff freq.'s for bandpass
but_fc_bp_lf_1 = 30;
but_fc_bp_lf_2 = 45;
but_fc_bp_lf_3 = 60;
but_fc_bp_hf_1 = 155;
but_fc_bp_hf_2 = 155;
but_fc_bp_hf_3 = 170;
%Cutoff arrangement for using them properly into
codes
but_wc_lp_1 = but_fc_lp_1/(fs/2);
but_wc_lp_2 = but_fc_lp_2/(fs/2);
but_wc_lp_3 = but_fc_lp_3/(fs/2);
but_wc_hp_1 = but_fc_hp_1/(fs/2);
but_wc_hp_2 = but_fc_hp_2/(fs/2);
but_wc_hp_3 = but_fc_hp_3/(fs/2);
but_wc_bp_lf_1 = but_fc_bp_lf_1/(fs/2);
but_wc_bp_hf_1 = but_fc_bp_hf_1/(fs/2);
but_wc_bp_lf_2 = but_fc_bp_lf_2/(fs/2);
but_wc_bp_hf_2 = but_fc_bp_hf_2/(fs/2);
but_wc_bp_lf_3 = but_fc_bp_lf_3/(fs/2);
but_wc_bp_hf_3 = but_fc_bp_hf_3/(fs/2);
% Butterworth lowpass
[k_1,l_1] = butter(10,but_wc_lp_1,'low'); % 8th
order low-pass filter   0.2 mean 0.2*fs/2 Hz
[k_2,l_2] = butter(10,but_wc_lp_2,'low'); % 8th
order low-pass filter   0.2 mean 0.2*fs/2 Hz
[k_3,l_3] = butter(10,but_wc_lp_3,'low'); % 8th
order low-pass filter   0.2 mean 0.2*fs/2 Hz
% Butterworth highpass
[c_1,v_1] = butter(8,but_wc_hp_1,'high'); % 8th
order high-pass filter
[c_2,v_2] = butter(8,but_wc_hp_2,'high'); % 8th
order high-pass filter
[c_3,v_3] = butter(8,but_wc_hp_3,'high'); % 8th
order high-pass filter
% Butterworth bandpass
```

```
[p_1,o_1] = butter(7,[but_wc_bp_lf_1
but_wc_bp_hf_1]); % 8th order high-pass filter
[p_2,o_2] = butter(7,[but_wc_bp_lf_2
but_wc_bp_hf_2]); % 8th order high-pass filter
[p_3,o_3] = butter(7,[but_wc_bp_lf_3
but_wc_bp_hf_3]); % 8th order high-pass filter
%To filtering our notched signal for each filter
that we obtain above
kl_1 = filter(k_1,l_1,ns);  % filtering
(convolution)
kl_2 = filter(k_2,l_2,ns);  % filtering
(convolution)
kl_3 = filter(k_3,l_3,ns);  % filtering
(convolution)
cv_1 = filter(c_1,v_1,ns);  % filtering
(convolution)
cv_2 = filter(c_2,v_2,ns);  % filtering
(convolution)
cv_3 = filter(c_3,v_3,ns);  % filtering
(convolution)
po_1 = filter(p_1,o_1,ns);  % filtering
(convolution)
po_2 = filter(p_2,o_2,ns);  % filtering
(convolution)
po_3 = filter(p_3,o_3,ns);  % filtering
(convolution)
%FFT of each double filtered signal(notch and
butterworth)
fft_kl_1 = abs(fft(kl_1,fs+1));
fft_kl_1_s = fft_kl_1(1:fs/2+1)/fs;
fft_kl_2 = abs(fft(kl_2,fs+1));
fft_kl_2_s = fft_kl_2(1:fs/2+1)/fs;
fft_kl_3 = abs(fft(kl_3,fs+1));
fft_kl_3_s = fft_kl_3(1:fs/2+1)/fs;
fft_cv_1 = abs(fft(cv_1,fs+1));
fft_cv_1_s = fft_cv_1(1:fs/2+1)/fs;
fft_cv_2 = abs(fft(cv_2,fs+1));
fft_cv_2_s = fft_cv_2(1:fs/2+1)/fs;
fft_cv_3 = abs(fft(cv_3,fs+1));
fft_cv_3_s = fft_cv_3(1:fs/2+1)/fs;
fft_po_1 = abs(fft(po_1,fs+1));
fft_po_1_s = fft_po_1(1:fs/2+1)/fs;
fft_po_2 = abs(fft(po_2,fs+1));
```

```
fft_po_2_s = fft_po_2(1:fs/2+1)/fs;
fft_po_3 = abs(fft(po_3,fs+1));
fft_po_3_s = fft_po_3(1:fs/2+1)/fs;
v=250;
% Lowpass Plot
subplot 321; plot(t',kl_1); xlabel('time
(seconds)'), ylabel('Magnitude'); title(['First
Low-pass filter with cutoff
',num2str(but_fc_lp_1),' Hz.']);
xlim([0,10]);ylim([-v v]);
subplot 322; plot(0:fs/2,fft_kl_1_s);
xlabel('frequency (Hz)'), ylabel('Magnitude');
title(['Fourier Transform for
',num2str(but_fc_lp_1),' Hz.']);
subplot 323; plot(t',kl_2); xlabel('time
(seconds)'), ylabel('Magnitude'); title(['Second
Low-pass filter with cutoff
',num2str(but_fc_lp_2),' Hz.']); xlim([0,10]);
ylim([-v v]);
subplot 324; plot(0:fs/2,fft_kl_2_s);
xlabel('frequency (Hz)'); ylabel('Magnitude');
title(['Fourier Transform for
',num2str(but_fc_lp_2),' Hz.']);
subplot 325; plot(t',kl_3); xlabel('time
(seconds)'), ylabel('Magnitude'); title(['Third
Low-pass filter with cutoff
',num2str(but_fc_lp_3),' Hz.']); xlim([0,10]);
ylim([-v v]);
subplot 326; plot(0:fs/2,fft_kl_3_s);
xlabel('frequency (Hz)'), ylabel('Magnitude');
title(['Fourier Transform for
',num2str(but_fc_lp_3),' Hz.']);
%Bode plot of Butterworth Lowpass filters
fvtool(k_1,l_1) % filter visualization tool
fvtool(k_2,l_2);title("Bode plot of lowpass at
75Hz") % filter visualization tool
fvtool(k_3,l_3) % filter visualization tool
v=186;
% Highpass Plot
subplot 321; plot(t',cv_1); xlabel('time
(seconds)'); ylabel('Magnitude'); title(['First
High-pass filter with cutoff
```

```
',num2str(but_fc_hp_1),' Hz.']); xlim([0,10]);
ylim([-v v]);
subplot 322; plot(0:fs/2,fft_cv_1_s);
xlabel('frequency (Hz)'); ylabel('Magnitude');
title(['Fourier Transform for
',num2str(but_fc_hp_1),' Hz.']);
subplot 323; plot(t',cv_2); xlabel('time
(seconds)'); ylabel('Magnitude'); title(['Second
High-pass filter with cutoff
',num2str(but_fc_hp_2),' Hz.']); xlim([0,10]);
ylim([-v v]);
subplot 324; plot(0:fs/2,fft_cv_2_s);
xlabel('frequency (Hz)'); ylabel('Magnitude');
title(['Fourier Transform for
',num2str(but_fc_hp_2),' Hz.']);
subplot 325; plot(t',cv_3); xlabel('time
(seconds)'); ylabel('Magnitude'); title(['Third
High-pass filter with cutoff
',num2str(but_fc_hp_3),'
Hz.']);xlim([0,10]);ylim([-v v]);
subplot 326; plot(0:fs/2,fft_cv_3_s);
xlabel('frequency (Hz)');
ylabel('Magnitude');title(['Fourier Transform for
',num2str(but_fc_hp_3),' Hz.']);
%Bode plot of Butterworth Highpass filters
fvtool(c_1,v_1) % filter visualization tool
fvtool(c_2,v_2);title("Bode plot of highpass at
45Hz") % filter visualization tool
fvtool(c_3,v_3) % filter visualization tool
v=200;
% Bandpass
subplot 321; plot(t',po_1);xlabel('time
(seconds)'); ylabel('Magnitude');title(['First
Band-pass filter with cutoff
',num2str(but_fc_bp_lf_1),' Hz and ',
num2str(but_fc_bp_hf_1),'
Hz.']);xlim([0,10]);ylim([-v v]);
subplot 322;
plot(0:fs/2,fft_po_1_s);xlabel('frequency (Hz)');
ylabel('Magnitude');title(['Fourier Transform for
',num2str(but_fc_bp_lf_1),' Hz and ',
num2str(but_fc_bp_hf_1),' Hz.']);
```

```
subplot 323; plot(t',po_2); xlabel('time
(seconds)'); ylabel('Magnitude');title(['Second
Band-pass filter with cutoff
',num2str(but_fc_bp_lf_2),' Hz and ',
num2str(but_fc_bp_hf_2),' Hz.']); xlabel('time
(seconds)'),
ylabel('Magnitude');xlim([0,10]);ylim([-v v]);
subplot 324; plot(0:fs/2,fft_po_2_s);
xlabel('frequency (Hz)');
ylabel('Magnitude');title(['Fourier Transform for
',num2str(but_fc_bp_lf_2),' Hz and ',
num2str(but_fc_bp_hf_2),' Hz.']);
subplot 325; plot(t',po_3); xlabel('time
(seconds)'); ylabel('Magnitude');title(['Third
Band-pass filter with cutoff
',num2str(but_fc_bp_lf_3),' Hz and ',
num2str(but_fc_bp_hf_3),'
Hz.']);xlim([0,10]);ylim([-v v]);
subplot 326; plot(0:fs/2,fft_po_3_s);
xlabel('frequency (Hz)');
ylabel('Magnitude');title(['Fourier Transform for
',num2str(but_fc_bp_lf_3),' Hz and ',
num2str(but_fc_bp_hf_3),' Hz.']);
%Bode plot of Butterworth Bandpass filters
fvtool(p_1,o_1), % filter visualization tool
fvtool(p_2,o_3), title("Bode plot of bandpass
45Hz-155Hz");Fronsize=40;% filter visualization
tool
fvtool(p_3,o_3), % filter visualization tool
%CHEBYCHEV%
%Graphs' y-axis limits
v_1 = 150; % limits for Lowpass
v_2 = 100; %Limits for highpass
v_3 = 75; % Limits fo bandpass
%Low-pass Cutoffs'
ch_fc_lp_1 = 110;
ch_fc_lp_2 = 70;
ch_fc_lp_3 = 155;
%High-pass Cutoffs'
ch_fc_hp_1 = 25;
ch_fc_hp_2 = 45;
ch_fc_hp_3 = 80;
%Band-pass Cutoffs'
```

```
ch_graph1_fc_bp_1 = 30;
ch_graph1_fc_bp_2 = 155;

ch_graph2_fc_bp_1 = 45;
ch_graph2_fc_bp_2 = 155;

ch_graph3_fc_bp_1 = 45;
ch_graph3_fc_bp_2 = 170;

ch_wc_lp_1 = ch_fc_lp_1/(fs/2);
ch_wc_lp_2 = ch_fc_lp_2/(fs/2);
ch_wc_lp_3 = ch_fc_lp_3/(fs/2);
ch_wc_hp_1 = ch_fc_hp_1/(fs/2);
ch_wc_hp_2 = ch_fc_hp_2/(fs/2);
ch_wc_hp_3 = ch_fc_hp_3/(fs/2);
ch_graph1_wc_bp_1 = ch_graph1_fc_bp_1/(fs/2);
ch_graph1_wc_bp_2 = ch_graph1_fc_bp_2/(fs/2);
ch_graph2_wc_bp_1 = ch_graph2_fc_bp_1/(fs/2);
ch_graph2_wc_bp_2 = ch_graph2_fc_bp_2/(fs/2);
ch_graph3_wc_bp_1 = ch_graph3_fc_bp_1/(fs/2);
ch_graph3_wc_bp_2 = ch_graph3_fc_bp_2/(fs/2);
%-----------------------------------------
a = 10;% db değeri
%Low-pass filters, ffts' and plots
[h,g] = cheby1(9,a,ch_wc_lp_1,'low');
[k,l] = cheby1(9,a,ch_wc_lp_2,'low');
[m,n] = cheby1(9,a,ch_wc_lp_3,'low');
fvtool(h,g);title(['Lowpass bode plot for
',num2str(ch_fc_lp_1),' Hz']);
fvtool(k,l);title(['Lowpass bode plot for
',num2str(ch_fc_lp_2),' Hz']);
fvtool(m,n);title(['Lowpass bode plot for
',num2str(ch_fc_lp_3),' Hz']);
lp_filtered_1 = filter(h,g,ns);
lp_filtered_2 = filter(k,l,ns);
lp_filtered_3 = filter(m,n,ns);

fft_hp_filtered_1 = abs(fft(lp_filtered_1,fs+1));
fft_lp_filtered_1_s =
fft_hp_filtered_1(1:fs/2+1)/fs;
fft_bp_filtered_2 = abs(fft(lp_filtered_2,fs+1));
fft_lp_filtered_2_s =
fft_bp_filtered_2(1:fs/2+1)/fs;
fft_hp_filtered_3 = abs(fft(lp_filtered_3,fs+1));
```

```
fft_lp_filtered_3_s =
fft_hp_filtered_3(1:fs/2+1)/fs;

subplot 321; plot(t',lp_filtered_1); xlabel('time
(seconds)'); ylabel('Magnitude');title(['Low-pass
Chebychev Filter With Cutoff
',num2str(ch_fc_lp_1),' Hz.']); ylim([-v_1 v_1]);
xlim([0 10]);
subplot 322; plot(0:fs/2,fft_lp_filtered_1_s);
xlabel('frequency (Hz)');
ylabel('Magnitude');title(['Fourier Transform for
',num2str(ch_fc_lp_1),' Hz.']);
subplot 323; plot(t',lp_filtered_2); xlabel('time
(seconds)'); ylabel('Magnitude');title(['Low-pass
Chebychev Filter With Cutoff
',num2str(ch_fc_lp_2),' Hz.']); ylim([-v_1 v_1]);
xlim([0 10]);
subplot 324; plot(0:fs/2,fft_lp_filtered_2_s);
xlabel('frequency (Hz)');
ylabel('Magnitude');title(['Fourier Transform for
',num2str(ch_fc_lp_2),' Hz.']);
subplot 325; plot(t',lp_filtered_3); xlabel('time
(seconds)'); ylabel('Magnitude'); title(['Low-
pass Chebychev Filter With Cutoff
',num2str(ch_fc_lp_3),' Hz.']); ylim([-v_1 v_1]);
xlim([0 10]);
subplot 326; plot(0:fs/2,fft_lp_filtered_3_s);
xlabel('frequency (Hz)'); ylabel('Magnitude');
title(['Fourier Transform for
',num2str(ch_fc_lp_3),' Hz.']);
%High-pass filters, ffts' and plots
[hh,gg] = cheby1(9,a,ch_wc_hp_1,'high');
[kk,ll] = cheby1(9,a,ch_wc_hp_2,'high');
[mm,nn] = cheby1(9,a,ch_wc_hp_3,'high');
fvtool(hh,gg);title(['Highpass bode plot for
',num2str(ch_fc_hp_1),' Hz']);
fvtool(kk,ll);title(['Highpass bode plot for
',num2str(ch_fc_hp_2),' Hz']);
fvtool(mm,nn);title(['Highpass bode plot for
',num2str(ch_fc_hp_3),' Hz']);
hp_filtered_1 = filter(hh,gg,ns);
hp_filtered_2 = filter(kk,ll,ns);
hp_filtered_3 = filter(mm,nn,ns);
```

```
fft_hp_filtered_1 = abs(fft(hp_filtered_1,fs+1));
fft_hp_filtered_1_s =
fft_hp_filtered_1(1:fs/2+1)/fs;
fft_bp_filtered_2 = abs(fft(hp_filtered_2,fs+1));
fft_hp_filtered_2_s =
fft_bp_filtered_2(1:fs/2+1)/fs;
fft_hp_filtered_3 = abs(fft(hp_filtered_3,fs+1));
fft_hp_filtered_3_s =
fft_hp_filtered_3(1:fs/2+1)/fs;

subplot 321; plot(t',hp_filtered_1);xlabel('time
(seconds)'); ylabel('Magnitude');title(['High-
pass Chebychev Filter With Cutoff
',num2str(ch_fc_hp_1),' Hz.']); ylim([-v_2 v_2]);
xlim([0 10]);
subplot 322; plot(0:fs/2,fft_hp_filtered_1_s);
xlabel('frequency (Hz)');
ylabel('Magnitude');title(['Fourier Transform for
',num2str(ch_fc_hp_1),' Hz.']);
subplot 323; plot(t',hp_filtered_2); xlabel('time
(seconds)'); ylabel('Magnitude');title(['High-
pass Chebychev Filter With Cutoff
',num2str(ch_fc_hp_2),' Hz.']); ylim([-v_2 v_2]);
xlim([0 10]);
subplot 324; plot(0:fs/2,fft_hp_filtered_2_s);
xlabel('frequency (Hz)');
ylabel('Magnitude');title(['Fourier Transform for
',num2str(ch_fc_hp_2),' Hz.']);
subplot 325; plot(t',hp_filtered_3); xlabel('time
(seconds)'); ylabel('Magnitude');title(['High-
pass Chebychev Filter With Cutoff
',num2str(ch_fc_hp_3),' Hz.']); ylim([-v_2 v_2]);
xlim([0 10]);
subplot 326; plot(0:fs/2,fft_hp_filtered_3_s);
xlabel('frequency (Hz)');
ylabel('Magnitude');title(['Fourier Transform for
',num2str(ch_fc_hp_3),' Hz.']);
%Band-pass filters, ffts' and plots
[hhh,ggg] = cheby1(6,a,[ch_graph1_wc_bp_1
ch_graph1_wc_bp_2]);
[kkk,lll] = cheby1(6,a,[ch_graph2_wc_bp_1
ch_graph2_wc_bp_2]);
[mmm,nnn] = cheby1(6,a,[ch_graph3_wc_bp_1
ch_graph3_wc_bp_2]);
```

```
fvtool(hhh,ggg);title(['Bandpass bode plot for
',num2str(ch_graph1_fc_bp_1),' Hz and
',num2str(ch_graph1_fc_bp_2),' Hz']);
fvtool(kkk,lll);title(['Bandpass bode plot for
',num2str(ch_graph2_fc_bp_1),' Hz and
',num2str(ch_graph2_fc_bp_2),' Hz']);
fvtool(mmm,nnn);title(['Bandpass bode plot for
',num2str(ch_graph3_fc_bp_1),' Hz and
',num2str(ch_graph3_fc_bp_2),' Hz']);
bp_filtered_1 = filter(hhh,ggg,ns);
bp_filtered_2 = filter(kkk,lll,ns);
bp_filtered_3 = filter(mmm,nnn,ns);

fft_bp_filtered_1 = abs(fft(bp_filtered_1,fs+1));
fft_bp_filtered_1_s =
fft_bp_filtered_1(1:fs/2+1)/fs;
fft_bp_filtered_2 = abs(fft(bp_filtered_2,fs+1));
fft_bp_filtered_2_s =
fft_bp_filtered_2(1:fs/2+1)/fs;
fft_bp_filtered_3 = abs(fft(bp_filtered_3,fs+1));
fft_bp_filtered_3_s =
fft_bp_filtered_3(1:fs/2+1)/fs;
subplot 321; plot(t',bp_filtered_1); xlabel('time
(seconds)'); ylabel('Magnitude');title(['Band-
pass Chebychev Filter With Cutoff
',num2str(ch_graph1_fc_bp_1),'-
',num2str(ch_graph1_fc_bp_2),' Hz.']); ylim([-v_3
v_3]); xlim([0 10]);
subplot 322;
plot(0:fs/2,fft_bp_filtered_1_s);xlabel('frequenc
y (Hz)'); ylabel('Magnitude');title(['Fourier
Transform for ',num2str(ch_graph1_fc_bp_1),'-
',num2str(ch_graph1_fc_bp_2),' Hz.']);
subplot 323; plot(t',bp_filtered_2); xlabel('time
(seconds)'); ylabel('Magnitude');title(['Band-
pass Chebychev Filter With Cutoff
',num2str(ch_graph2_fc_bp_1),'-
',num2str(ch_graph2_fc_bp_2),' Hz.']);ylim([-v_3
v_3]); xlim([0 10]);
subplot 324; plot(0:fs/2,fft_bp_filtered_2_s);
xlabel('frequency (Hz)');
ylabel('Magnitude');title(['Fourier Transform for
',num2str(ch_graph2_fc_bp_1),'-
',num2str(ch_graph2_fc_bp_2),' Hz.']);
```

```
subplot 325; plot(t',bp_filtered_3); xlabel('time
(seconds)'); ylabel('Magnitude');title(['Band-
pass Chebychev Filter With Cutoff
',num2str(ch_graph3_fc_bp_1),'-
',num2str(ch_graph3_fc_bp_2),' Hz.']); ylim([-v_3
v_3]); xlim([0 10]);
subplot 326; plot(0:fs/2,fft_bp_filtered_3_s);
xlabel('frequency (Hz)');
ylabel('Magnitude');title(['Fourier Transform for
',num2str(ch_graph3_fc_bp_1),'-
',num2str(ch_graph3_fc_bp_2),' Hz.']);
%ELLİPTİC%
%Graphs' y-axis limits
ell_v_1 = 150; % limits for Lowpass
ell_v_2 = 100; %Limits for highpass
ell_v_3 = 75; % Limits fo bandpass
%Low-pass Cutoffs'
ell_ch_fc_lp_1 = 70;
ell_ch_fc_lp_2 = 110;
ell_ch_fc_lp_3 = 155;
%High-pass Cutoffs'
ell_ch_fc_hp_1 = 25;
ell_ch_fc_hp_2 = 45;
ell_ch_fc_hp_3 = 80;
%Band-pass Cutoffs'
ell_ch_graph1_fc_bp_1 = 30;
ell_ch_graph1_fc_bp_2 = 155;
ell_ch_graph2_fc_bp_1 = 45;
ell_ch_graph2_fc_bp_2 = 155;
ell_ch_graph3_fc_bp_1 = 45;
ell_ch_graph3_fc_bp_2 = 170;

ell_ch_wc_lp_1 = ell_ch_fc_lp_1/(fs/2);
ell_ch_wc_lp_2 = ell_ch_fc_lp_2/(fs/2);
ell_ch_wc_lp_3 = ell_ch_fc_lp_3/(fs/2);
ell_ch_wc_hp_1 = ell_ch_fc_hp_1/(fs/2);
ell_ch_wc_hp_2 = ell_ch_fc_hp_2/(fs/2);
ell_ch_wc_hp_3 = ell_ch_fc_hp_3/(fs/2);
ell_ch_graph1_wc_bp_1 =
ell_ch_graph1_fc_bp_1/(fs/2);
ell_ch_graph1_wc_bp_2 =
ell_ch_graph1_fc_bp_2/(fs/2);
ell_ch_graph2_wc_bp_1 =
ell_ch_graph2_fc_bp_1/(fs/2);
```

```
ell_ch_graph2_wc_bp_2 =
ell_ch_graph2_fc_bp_2/(fs/2);
ell_ch_graph3_wc_bp_1 =
ell_ch_graph3_fc_bp_1/(fs/2);
ell_ch_graph3_wc_bp_2 =
ell_ch_graph3_fc_bp_2/(fs/2);
%-----------------------------------------
a = 10;% db değeri
%Low-pass filters, ffts' and plots
[h,g] = cheby1(9,a,ell_ch_wc_lp_1,'low');
[k,l] = cheby1(9,a,ell_ch_wc_lp_2,'low');
[m,n] = cheby1(9,a,ell_ch_wc_lp_3,'low');
fvtool(h,g);title(['Lowpass bode plot for
',num2str(ell_ch_fc_lp_1),' Hz']);
fvtool(k,l);title(['Lowpass bode plot for
',num2str(ell_ch_fc_lp_2),' Hz']);
fvtool(m,n);title(['Lowpass bode plot for
',num2str(ell_ch_fc_lp_3),' Hz']);
lp_filtered_1 = filter(h,g,ns);
lp_filtered_2 = filter(k,l,ns);
lp_filtered_3 = filter(m,n,ns);

fft_hp_filtered_1 = abs(fft(lp_filtered_1,fs+1));
fft_lp_filtered_1_s =
fft_hp_filtered_1(1:fs/2+1)/fs;
fft_bp_filtered_2 = abs(fft(lp_filtered_2,fs+1));
fft_lp_filtered_2_s =
fft_bp_filtered_2(1:fs/2+1)/fs;
fft_hp_filtered_3 = abs(fft(lp_filtered_3,fs+1));
fft_lp_filtered_3_s =
fft_hp_filtered_3(1:fs/2+1)/fs;

subplot 321; plot(t',lp_filtered_1); xlabel('time
(seconds)'); ylabel('Magnitude');title(['Low-pass
Chebychev Filter With Cutoff
',num2str(ell_ch_fc_lp_1),' Hz.']); ylim([-
ell_v_1 ell_v_1]); xlim([0 10]);
subplot 322; plot(0:fs/2,fft_lp_filtered_1_s);
xlabel('frequency (Hz)');
ylabel('Magnitude');title(['Fourier Transform for
',num2str(ell_ch_fc_lp_1),' Hz.']);
subplot 323; plot(t',lp_filtered_2); xlabel('time
(seconds)'); ylabel('Magnitude');title(['Low-pass
Chebychev Filter With Cutoff
```

```
',num2str(ell_ch_fc_lp_2),' Hz.']); ylim([-
ell_v_1 ell_v_1]); xlim([0 10]);
subplot 324; plot(0:fs/2,fft_lp_filtered_2_s);
xlabel('frequency (Hz)');
ylabel('Magnitude');title(['Fourier Transform for
',num2str(ell_ch_fc_lp_2),' Hz.']);
subplot 325; plot(t',lp_filtered_3); xlabel('time
(seconds)'); ylabel('Magnitude');title(['Low-pass
Chebychev Filter With Cutoff
',num2str(ell_ch_fc_lp_3),' Hz.']); ylim([-
ell_v_1 ell_v_1]); xlim([0 10]);
subplot 326; plot(0:fs/2,fft_lp_filtered_3_s);
xlabel('frequency (Hz)');
ylabel('Magnitude');title(['Fourier Transform for
',num2str(ell_ch_fc_lp_3),' Hz.']);
%High-pass filters, ffts' and plots
[hh,gg] = cheby1(9,a,ell_ch_wc_hp_1,'high');
[kk,ll] = cheby1(9,a,ell_ch_wc_hp_2,'high');
[mm,nn] = cheby1(9,a,ell_ch_wc_hp_3,'high');
fvtool(hh,gg);title(['Highpass bode plot for
',num2str(ell_ch_fc_hp_1),' Hz']);
fvtool(kk,ll);title(['Highpass bode plot for
',num2str(ell_ch_fc_hp_2),' Hz']);
fvtool(mm,nn);title(['Highpass bode plot for
',num2str(ell_ch_fc_hp_3),' Hz']);
hp_filtered_1 = filter(hh,gg,ns);
hp_filtered_2 = filter(kk,ll,ns);
hp_filtered_3 = filter(mm,nn,ns);

fft_hp_filtered_1 = abs(fft(hp_filtered_1,fs+1));
fft_hp_filtered_1_s =
fft_hp_filtered_1(1:fs/2+1)/fs;
fft_bp_filtered_2 = abs(fft(hp_filtered_2,fs+1));
fft_hp_filtered_2_s =
fft_bp_filtered_2(1:fs/2+1)/fs;
fft_hp_filtered_3 = abs(fft(hp_filtered_3,fs+1));
fft_hp_filtered_3_s =
fft_hp_filtered_3(1:fs/2+1)/fs;

subplot 321; plot(t',hp_filtered_1); xlabel('time
(seconds)'); ylabel('Magnitude');title(['High-
pass Chebychev Filter With Cutoff
',num2str(ell_ch_fc_hp_1),' Hz.']); ylim([-
ell_v_2 ell_v_2]); xlim([0 10]);
```

```
subplot 322; plot(0:fs/2,fft_hp_filtered_1_s);
xlabel('frequency (Hz)');
ylabel('Magnitude');title(['Fourier Transform for
',num2str(ell_ch_fc_hp_1),' Hz.']);
subplot 323; plot(t',hp_filtered_2); xlabel('time
(seconds)'); ylabel('Magnitude');title(['High-
pass Chebychev Filter With Cutoff
',num2str(ell_ch_fc_hp_2),' Hz.']); ylim([-
ell_v_2 ell_v_2]); xlim([0 10]);
subplot 324; plot(0:fs/2,fft_hp_filtered_2_s);
xlabel('frequency (Hz)');
ylabel('Magnitude');title(['Fourier Transform for
',num2str(ell_ch_fc_hp_2),' Hz.']);
subplot 325; plot(t',hp_filtered_3); xlabel('time
(seconds)'); ylabel('Magnitude');title(['High-
pass Chebychev Filter With Cutoff
',num2str(ell_ch_fc_hp_3),' Hz.']); ylim([-
ell_v_2 ell_v_2]); xlim([0 10]);
subplot 326; plot(0:fs/2,fft_hp_filtered_3_s);
xlabel('frequency (Hz)');
ylabel('Magnitude');title(['Fourier Transform for
',num2str(ell_ch_fc_hp_3),' Hz.']);
%Band-pass filters, ffts' and plots
[hhh,ggg] = cheby1(6,a,[ell_ch_graph1_wc_bp_1
ell_ch_graph1_wc_bp_2]);
[kkk,lll] = cheby1(6,a,[ell_ch_graph2_wc_bp_1
ell_ch_graph2_wc_bp_2]);
[mmm,nnn] = cheby1(6,a,[ell_ch_graph3_wc_bp_1
ell_ch_graph3_wc_bp_2]);
fvtool(hhh,ggg);title(['Bandpass bode plot for
',num2str(ell_ch_graph1_fc_bp_1),' Hz and
',num2str(ell_ch_graph1_fc_bp_2),' Hz']);
fvtool(kkk,lll);title(['Bandpass bode plot for
',num2str(ell_ch_graph2_fc_bp_1),' Hz and
',num2str(ell_ch_graph2_fc_bp_2),' Hz']);
fvtool(mmm,nnn);title(['Bandpass bode plot for
',num2str(ell_ch_graph3_fc_bp_1),' Hz and
',num2str(ell_ch_graph3_fc_bp_2),' Hz']);
bp_filtered_1 = filter(hhh,ggg,ns);
bp_filtered_2 = filter(kkk,lll,ns);
bp_filtered_3 = filter(mmm,nnn,ns);

fft_bp_filtered_1 = abs(fft(bp_filtered_1,fs+1));
```

```
fft_bp_filtered_1_s =
fft_bp_filtered_1(1:fs/2+1)/fs;
fft_bp_filtered_2 = abs(fft(bp_filtered_2,fs+1));
fft_bp_filtered_2_s =
fft_bp_filtered_2(1:fs/2+1)/fs;
fft_bp_filtered_3 = abs(fft(bp_filtered_3,fs+1));
fft_bp_filtered_3_s =
fft_bp_filtered_3(1:fs/2+1)/fs;

subplot 321; plot(t',bp_filtered_1); xlabel('time
(seconds)'); ylabel('Magnitude');title(['Band-
pass Chebychev Filter With Cutoff
',num2str(ell_ch_graph1_fc_bp_1),'-
',num2str(ell_ch_graph1_fc_bp_2),' Hz.']);
ylim([-ell_v_3 ell_v_3]); xlim([0 10]);
subplot 322;
plot(0:fs/2,fft_bp_filtered_1_s);xlabel('frequenc
y (Hz)'); ylabel('Magnitude');title(['Fourier
Transform for ',num2str(ell_ch_graph1_fc_bp_1),'-
',num2str(ell_ch_graph1_fc_bp_2),' Hz.']);
subplot 323; plot(t',bp_filtered_2); xlabel('time
(seconds)'); ylabel('Magnitude');title(['Band-
pass Chebychev Filter With Cutoff
',num2str(ell_ch_graph2_fc_bp_1),'-
',num2str(ell_ch_graph2_fc_bp_2),' Hz.']);ylim([-
ell_v_3 ell_v_3]); xlim([0 10]);
subplot 324; plot(0:fs/2,fft_bp_filtered_2_s);
xlabel('frequency (Hz)'),
ylabel('Magnitude');title(['Fourier Transform for
',num2str(ell_ch_graph2_fc_bp_1),'-
',num2str(ell_ch_graph2_fc_bp_2),' Hz.']);
subplot 325; plot(t',bp_filtered_3); xlabel('time
(seconds)'), ylabel('Magnitude');title(['Band-
pass Chebychev Filter With Cutoff
',num2str(ell_ch_graph3_fc_bp_1),'-
',num2str(ell_ch_graph3_fc_bp_2),' Hz.']);
ylim([-ell_v_3 ell_v_3]); xlim([0 10]);
subplot 326; plot(0:fs/2,fft_bp_filtered_3_s);
xlabel('frequency (Hz)'),
ylabel('Magnitude');title(['Fourier Transform for
',num2str(ell_ch_graph3_fc_bp_1),'-
',num2str(ell_ch_graph3_fc_bp_2),' Hz.']);
```

5.3 Graphing Root-Mean Square (RMS)

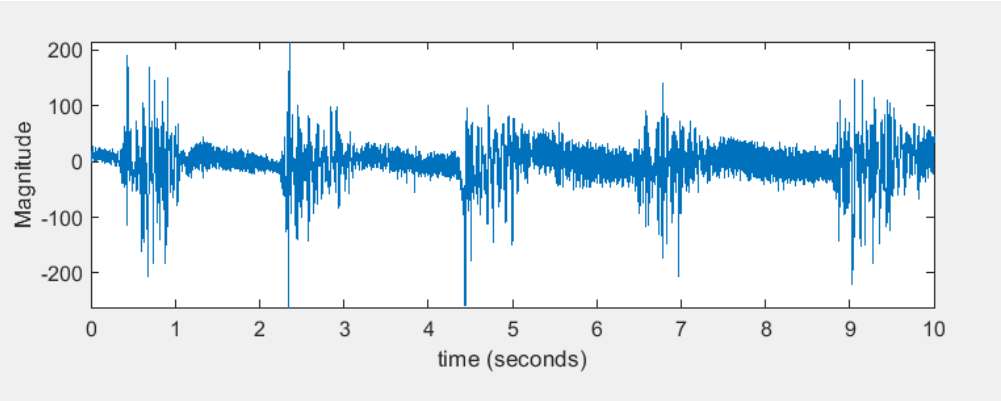


Figure 5.3.1 Unfiltered EMG Graph

The graph shown in Figure 5.3.1 is the graph on which the rms calculations are applied. RMS calculation was made by applying determined and calculated filters to this graph.

Below is the MATLAB code used to measure the contraction time, that is, the start time of the contraction, the ending time, and the contraction time, and to calculate the contraction strength.

MATLAB algorithm for RMS:

```
load veri18;
veri18 = veri18(~isnan(veri18)); %If there is any
NAN value when reading data, we used it to remove
it from our data.
s = veri18;  fs = 2345; % Our sample number is
23767 so that is how we defined fs approximately.
s = s-mean(s); % We used it to align EMG signals
from the 0 value.
fc_bp1 = 45; % First cutoff frequency value for
bandpass filter
fc_bp2 = 155; % Second cutoff frequency value for
bandpass filter
wc_bp1 = fc_bp1/(fs/2); % Value to be entered in
butterworth filter
wc_bp2 = fc_bp2/(fs/2);
t = (0:size(s,2)-1)/fs; % time component
subplot 321, plot(t',s)
xlabel('time (seconds)'), ylabel('Magnitude')
%ylim([-350 250]);
```

```
xlim([0 10]);
% FFT of the signal
F = abs(fft(s,fs+1));
Fs = F(1:fs/2+1)/(fs/2);

subplot 322, plot(0:fs/2,Fs)
xlabel('frequency (Hz)'), ylabel('Magnitude')
% Filtering
Wo = 50/(fs/2); % We defined as 50 Hz because of
city grid frequency
BW = Wo/35;
[b,a] = iirnotch(Wo,BW); %notch filter
fvtool(b,a); % To examine the filter structure
applied
ns = filter(b,a,s); % Filter purified from noise
at 50 Hz
subplot 323, plot(t',ns)
xlabel('time (seconds)'), ylabel('Magnitude')
%ylim([-350 250]);
xlim([0 10]);
title('Notch Filter for 50 Hz frequency')
% FFT of filtered signal
Fn = abs(fft(ns,fs+1));
Fns = Fn(1:fs/2+1)/fs;
subplot 324, plot(0:fs/2,Fns),
xlabel('frequency (Hz)'), ylabel('Magnitude')


[n,m] = butter(7,[wc_bp1 wc_bp2]);  % 7th order
band-pass filter for butterworth
ns_1 = filter(n,m,ns);  % filtering (convolution)

plot(t',ns_1)
title(['Band-Pass Filter for ' num2str(wc_bp1) '
and ' num2str(wc_bp2) ' for butter'])
xlabel('time (seconds)'), ylabel('Magnitude'),
xlim([0 10])
% FFT of filtered (butterworth) signal
fft_ns_1 = abs(fft(ns_1,fs+1));
fft_ns_1_s = fft_ns_1(1:fs/2+1)/fs;
plot(0:fs/2,fft_ns_1_s);
xlim([0 fs/2+1]), xlabel('frequency (Hz)'),
ylabel('Magnitude')
```

```
% RMS (Root-Mean Square)
N = 300; % Window
t_2 = (0:23467-1)/fs; % Time component for veri18
sizeemg = size(ns_1,2);
movingrms = zeros(sizeemg-N,1);
for i=1:sizeemg-N
    wins = ns_1(i:i+N-1);
    movingrms(i,1) = sqrt(sum(wins.^2)/N);
end
subplot 211, plot(t_2',movingrms)
hold on;
title('Root-Mean-Square')
xlabel('time (seconds)'), ylabel('Magnitude'),
xlim([0 10])
sz = size(t_2);
threshhold = 11.75; %12.48; % The limit value we
apply to determine the contraction time
y = threshhold*ones(sz); % Threshhold line to see
the limit
plot(t_2,y)


count = 1;
pp = 1;
xx = 1;
cc = 1;
ll = 1;
sn = []; % Sample number for each contraction
startTime = [];
stopTime = [];

for n=1:numel(movingrms) % Returns the number of
elements
    if movingrms(n) > threshhold
        if count == 1
        y = ['Time when the contraction starts
is',': ',num2str(n/fs),' s'];
        count = count + 1;
        disp(y);
        Ini_time = n/fs; % It gives the time
values at the start of the contraction
        k1 = n; % It gives the sample numbers at
the start of the contraction
```

```
         startTime(xx) = n/fs; % Gives the start
times as an array
         xx = xx + 1;
        end
    elseif movingrms(n) < threshhold
        if count == 2
        y1 = ['Time when the contraction ends
is',': ',num2str(n/fs),' s'];
        disp(y1);
        count = 1;
        End_time = n/fs; % It gives the time
values at the end of the contraction
        k2 = n; % It gives the sample numbers at
the end of the contraction
        stopTime(cc) = n/fs; % % Gives the end
times as an array
         cc = cc + 1;


        vns = End_time - Ini_time; % Time between
the initiation of the contraction until the
muscle is relaxed
        sn(ll) = vns;
        ll = ll + 1;
        vict = ['Contraction duration',':
',num2str(vns),' s'];
        disp(vict);
        rr(pp,1) = vns;
        Moc = movingrms(k1:k2,1); % moment of
contraction
        Svc(pp,1) = sum(Moc); % sum of values in
contraction

        end
    end
end
% CALCULATION OF POWER %
avaragePowerSize = length(startTime); % The
length of the contraction values when the
contraction starts
avaragePowers= ones(1,avaragePowerSize); % We
created the array to calculate avarage power

for i = 1:length(startTime)
```

```
    calculation_n = round(startTime(i)*fs); % It
gives the sample numbers.(Round was used for ease
of operation)
    start_n = startTime(i)*fs;
    end_n = stopTime(i)*fs;
    sum_of_values = 0;
    while (calculation_n <= end_n)
        sum_of_values = sum_of_values +
(movingrms(calculation_n))^2; % To calculate the
average power we squared each value at the moment
of contraction. Then we summed these values
        calculation_n = calculation_n + 1;
    end
    diff_n = end_n - start_n; % Number of samples
at contraction time
    avaragePowers(i) = sum_of_values/diff_n; %
Average power values
    answer = ['Contraction Power',':
',num2str(avaragePowers(i)),' mW'];
    disp(answer);
end
```

Command Window:

threshhold =

  11.7500

Time when the contraction starts is: 0.30874 s
Time when the contraction ends is: 1.1531 s
Contraction duration: 0.84435 s
Time when the contraction starts is: 2.1991 s
Time when the contraction ends is: 3.0597 s
Contraction duration: 0.86055 s
Time when the contraction starts is: 4.3301 s
Time when the contraction ends is: 5.0887 s
Contraction duration: 0.75864 s
Time when the contraction starts is: 6.4154 s
Time when the contraction ends is: 7.1041 s
Contraction duration: 0.6887 s
Time when the contraction starts is: 8.7569 s
Time when the contraction ends is: 9.5885 s
Contraction duration: 0.83156 s

Contraction Power: 940.4522 mW
Contraction Power: 546.7638 mW
Contraction Power: 403.8834 mW
Contraction Power: 897.8756 mW
Contraction Power: 775.7693 mW

A notch filter was applied to the EMG signal to calculate the root-mean-square value. Thus, the noises at 50 Hz caused by the electricity grid were eliminated. Next, a Butterworth Band-pass filter was applied. The cut-off frequency values of the band-pass filter were determined as 45 Hz and 155 Hz. To apply RMS, the window value was determined as 300 samples. It was set to process every 300 samples. As can be seen in Figure 5.3.3**,** the threshold value was determined to find the contraction time. This value was determined according to the places where the beginning and end of the contractions were smoother. The algorithm written in MATLAB was used to determine the start time, end time, contraction time, and power of the contraction. All desired results can be achieved from the MATLAB algorithm.

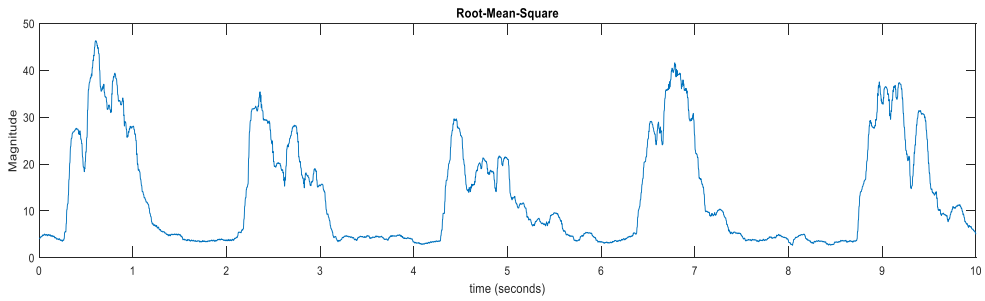


Figure 5.3.2 Rms Graph with Threshold line

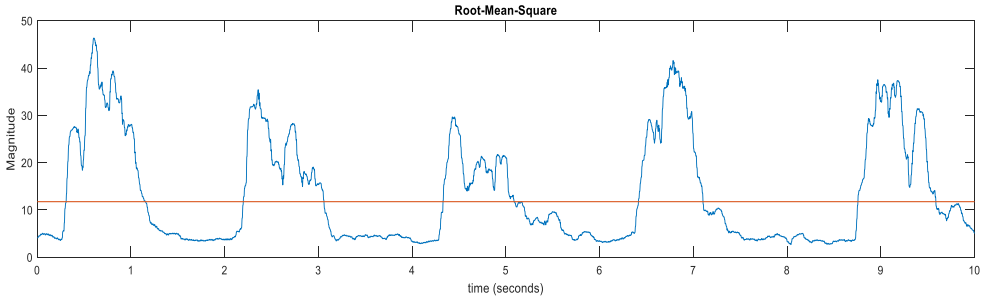


Figure 5.3.3 Rms Graph with Threshold line

5.3.1 Calculating Contraction Duration

The duration of the contractions was calculated with the help of the algorithm written in MATLAB and the results can be viewed from Table 1.

TABLE 2. CONTRACTION DURATION RESULTS

Contraction Order	Contraction Duration (s)
First Contraction	0.84435 s
Second Contraction	0.86055 s
Third Contraction	0.75864 s
Fourth Contraction	0.6887 s
Fifth Contraction	0.83156 s

While calculating the power of contraction, the average power was used [1].

$Average\ power = \frac{1}{N} \sum_{-N/2}^{N/2} |x[n]|^2$ [1]

N is the number of samples at the time of contraction.

The power of each contraction was calculated using the average power formula [1]. The power value of each contraction can be seen from the above MATLAB algorithm. As seen in Figure 5.3.2, there are 5 contractions in total. The values of these contractions are respectively:

First contraction: 940.4522 mW

Second contraction: 546.7638 mW

Third contraction: 403.8834 mW

Fourth contraction: 897.8756 mW

Fifth contraction: 775.7693 mW

5.3.2 Calculating Contraction Power

As can be seen in Figure 5.3.2 the contraction with the highest power is the first contraction since most area is in the first contraction. If we evaluate the found

contraction powers as percentiles, the results in the Table 3 below can be examined.

TABLE 3. CONTRACTION POWER RESULTS

Contraction Order	Calculation of Percent of Contraction Power
First Contraction	100%
Second Contraction	58.1383934%
Third Contraction	42.9456595%
Fourth Contraction	95.4727524%
Fifth Contraction	82.4889665%

When the results in Table 3 are observed, the contraction with the highest contraction power is the first contraction, while the least contraction is the third contraction. From these results, it can be observed which of the contractions are strong and which are weak and for how long they contract.

Results

Firstly, to obtain EMG signal data AD8232 and Arduino Uno connected to each other and data, which between 200 to 500 in terms of Arduino Serial port values, was obtained. While these data acquired different placements were tried. Then via coding, these data transferred into MATLAB environment. After, transferring data into MATLAB, it seen that there is a 50Hz peek due to the noise and it suppressed with Notch filter into one sixed of the original 50Hz noise component that can be seen in Figure-2.3 and main data was chosen according to Notch filtered signals. Then extra noise was tried to apply into signal via shaking arm and obtained and extra motion noise which occurs between 0 to 25Hz. Thus, 45 Hz high-pass filter used for suppressing the motion artefacts and more accurate EMG signal was obtained.

When it comes to second filtering for main EMG signal, firstly three different filter type compared, and best options were chosen. For Butterworth the best

options are 75, 45, and 45-155Hz for the low-pass, high-pass, and band pass respectively. Low-pass, high-pass, and band pass cut-off frequencies for the Chebyshev are 70, 45, and 45-155Hz were chosen respectively. For the elliptic filters, the optimum cut-off frequency for the low-pass is 75Hz and for the high-pass is 45Hz. As in the Butterworth and Chebyshev filters cut-off frequencies for the bandpass are 45 and155Hz. Among these 3 filters, the Elliptic was chosen due to its sharpness. Because of our aim which is filtering the main EMG signal, mainly exist between 50Hz and 150Hz, bandpass is the optimum filtering type to remove noisy components.

After that RMS of signal calculated and using that RMS, number of contractions that is 5 and contraction durations, which were 0.84435sec., 0.86055sec., 0.75864sec., 0.6887sec., 0.6887sec., obtained with a MATLAB algorithm. After, via another MATLAB algorithm, average power obtained that are 940.4522 W, 546.7638 W, 546.7638 W, 897.8756 W, 775.7693 W for each construction.

Conclusion

Creating an EMG system that is able to collect electrical signals produced by muscles by using electrodes was the aim of this project. For this purpose, AD8232, Arduino, MATLAB, and surface electrodes were used. In the beginning, Arduino and AD8232 were connected, and the surface electrodes were placed on different muscles to find the best place to obtain better results. Then, codes for the Arduino were written to obtain the data in serial port and via MATLAB codes, the serial port was read for 10 seconds, stored as a mat file, and analyzed. After that, 50Hz noise that was seen in the analyzes was suppressed with the help of a notch filter. In addition to that, different virtual filters with different cut-off frequencies and orders were applied on the notch filtered signal to determine the most applicable filter for obtaining the clearest signal. After the signal was filtered by the chosen filter, the clear signal was analyzed in terms of several constructions, construction time, and power. In the

end, all the hardware and software information were gathered to analyze an EMG signal on MATLAB.

References

[1] MILLER R. V., Jr (1958). Electromyography; uses and limitations. California medicine, 89(4), 250–252.
[2] Raez, M. B., Hussain, M. S., & Mohd-Yasin, F. (2006). Techniques of EMG signal analysis: detection, processing, classification and applications. Biological procedures online, 8, 11–35. https://doi.org/10.1251/bpo115