*Electromyography Signal Processing*

Devin Santos  
Department of Electrical and Computer Engineering California State University,ChicoChico, United States  
dsantos11@mail.csuchico.edu

*Abstract*—This paper discusses the process of acquiring an EMG signal and processing it to get a desired signal. The goal of this work is to be able to successfully implement a filter design learned in my DSP course and use it to filter my EMG signal.

Keywords—EMG, Electromyogram, filter, design, DSP

# Introduction

Electromyogram (EMG) signals are electrical signals that travel through our muscles when there is any movement or muscular activities [3]. These signals occur any time we think about moving a certain part of our body and the proper muscle group responds. For example, when the arm is moved in a curl motion the nervous system is sending signals to those muscles, telling them to contract, and the EMG signal can be recorded [1]. EMG signal processing is needed in the medical field today because patients who have lost a part of their body can have the ability to regain an artificial attachment, such as a prosthetic limb. With new technology advances we can monitor EMG signals and allow to the user to control the prosthetic limb.

To capture my own EMG signal, I purchased a MyoWare sensor made by Advancer Technologies. The sensor consists of three electrodes that are placed on the surface of the skin over the muscle of interest. For this project I placed the sensor near my forearm arm area to record the EMG signal when I make a fist and squeeze. The sensor does not come with pins installed so I had to solder pins to the board to connect jumper wires. To get the data to my computer I used the Tiva LaunchPad’s ADC and UART functions.

After the data has been recorded, I saved it in a file and uploaded it to the MATLAB environment. Once the data was in MATLAB, I could begin analyzing the frequency spectrum of my raw EMG signal. Since EMG signals are noise corrupted it is desired to filter out certain frequencies. From researching I found that typical desired EMG signals lie between 5Hz to 40Hz [2]. This gave me a starting point for designing my lowpass filter.

# Myoware sensor

The MyoWare sensor by Advancer Technologies was bought from Amazon and delivered to my house. They sell for about $40 per sensor and require the user to solder on pins for securing jumper wires. I bought a total of three sensors in anticipation to record the EMG signal in three different locations. However, my soldering skills are still at the beginner level and I could only get one sensor to successfully record data.Diagram, schematic

Description automatically generated

Figure 1. The MyoWare sensor by Advancer Technologies

This is the sensor that was used to capture my EMG signal. I powered the sensor using my 3.3V pin from my Tiva LaunchPad and grounded it using my GND pin. The EMG signal was recorded using the “RAW” pin on the sensor. The pin “SIG” on the sensor is an already processed EMG signal, so that signal was only used for a comparison to my filtered signal.

# Tiva LaunchPad

A picture containing text, electronics, scoreboard

Description automatically generatedThe Tiva LaunchPad was used as the data acquisition device, since I already had it in my possession. Other microcontrollers can be used and will achieve the same goal.

Figure 2. TM4C123 Tiva LaunchPad.

A picture containing text, indoor

Description automatically generatedSome basic C programming was used to initialize pin PE2 as an ADC. The sampling rate of the ADC was set to 125K samples per second. The interrupt was triggered by software so I could use SysTick as my sampling frequency. The sampling frequency was set to 500Hz. I initialized UART to communicate with my computer through the serial port. Every time my SysTick interrupt occurred, my ADC would sample. Then that sample was sent to my computer using UART and PuTTy.

Figure 3. Data acquisition set-up

Chart, histogram

Description automatically generatedAs you can see in the figure above, I have attached the sensor to my forearm area. The sensor is fed 3.3V and 0V from the Tiva and PE2 is connected to the “RAW” pin to sample the EMG signal. Then UART is used to send the digital sample values to my computer through the USB port.

# MATLAB

Graphical user interface

Description automatically generatedOnce I had the data samples on PuTTy, I was able to copy and paste them onto an excel spreadsheet, saving it as a .csv file. I was then able to use the “xlsread” command in MATLAB to extract the data from the file.

Figure 4. Raw EMG signal plotted on MATLAB.

After obtaining the plot shown in Figure 4, I needed to apply a Fast Fourier Transform (FFT) to look at the frequency spectrum. Chart

Description automatically generated

Figure 5. Plot after applying FFT

You can see I have strong small frequency signals and a lot of higher frequency signals. I want to design a filter that will eliminate these smaller and higher frequencies from my signal. Before designing my filter, I took a couple more steps in processing the signal. I rectified and removed the DC offset from the EMG signal to allow only positive voltage starting from 0V. This is done because if we were to connect a servo motor to the signal, we would have some meaningful data to move the motor [2].

Figure 6. Rectified EMG signal with no DC offset

To remove the DC offset I subtracted the mean of the signal from the original signal [2]. To rectify it I took the absolute value of the signal. This was done using the “mean” and “abs” MATLAB commands.

The next step was to design of the filter. First, I needed to decide what type of filter to choose. I had the options of choosing from a FIR or IIR filter. I was initially going to go with the FIR filter for a simpler design, however when running the raw EMG signal through the filter, I was getting some unwanted signal. So, I chose to further analyze the IIR filter design. My two Chart, line chart

Description automatically generatedoptions for the IIR filters were between Butterworth or Chebyshev. In the end I chose an IIR Butterworth filter because I wanted a flat response in the passband and stopband. If I chose the Chebyshev design, my filter would have oscillation in the Diagram

Description automatically generatedpassband and it would affect my filtered signal.

Figure 7. Zero and Pole distribution in z-plane

Chart

Description automatically generatedBy looking at the poles and zeros in the z-plane we can see the filter is stable since all poles lie within the unit circle.

Figure 8. Bode plot of Butterworth filter

As you can see, I have a lowpass filter by means of the Butterworth design. The passband is smooth and flat, as for the stopband as well. You can also see that we are only letting in lower frequencies and attenuating the higher frequencies.

Figure 9. Logarithmic Magnitude Response

Chart, line chart

Description automatically generated

Figure 10. Impulse Response

By looking at figures 7-10 we can see the characteristics of my 11th order Butterworth lowpass filter. My cutoff frequency is around 50Hz and my digital stopband magnitude is at 0.01.

wp = (2\*pi\*Fp)/Fs (1)

ws = (2\*pi\*Fst)/Fs (2)

Omega\_p = 2\*Fs\*tan(wp/2) (3)

Omega\_s = 2\*Fs\*tan(ws/2) (4)

delta1 = 1 – (1 – passband ripple)^2 (5)

delta2 = (stopband ripple)^2 (6)

N = (1/2)\*(log10( ((1/1-delta1)-1)/((1/delta2)-1)) / (7) log(Omega\_p/Omega\_s))

Omega\_c = Omega\_p / (1 / (1-delta1) -1)^(1/(2\*N)) (8)

k = 0:N-1; (9)

Sk = Omega\_c\*exp(j\*pi\*((2\*k+1)/2/N + ½)) (10)

Poles = (1 + ((Sk./Fs)./2))./(1 – ((Sk./Fs)./2)) (11)

These are the equations I used to design my IIR lowpass Butterworth filter on MATLAB.

# Results

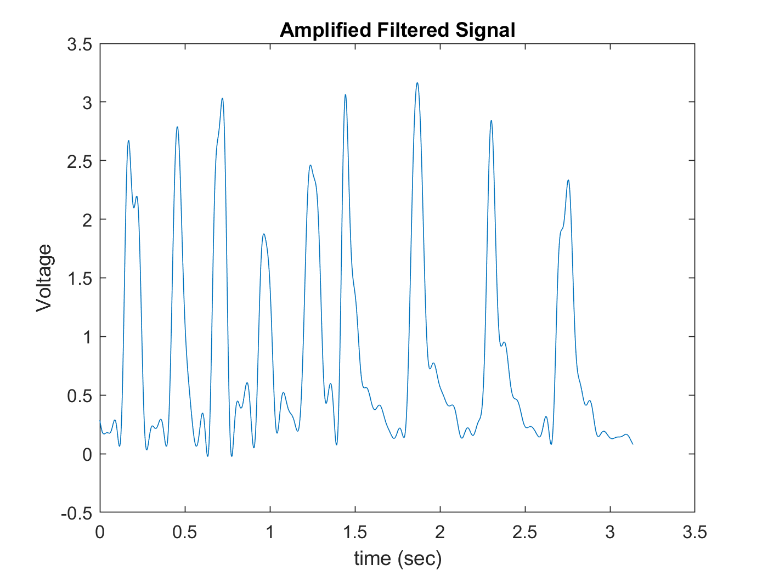
 After the filter equations were solved using MATLAB, I got the numerator and denominator coefficients of the filter using the “zp2tf” command. Using the “num” and “den” coefficients I was able to run my signal through the filter using the “filtfilt” command. I was able to obtain a filtered EMG signal.

Figure 11. Filtered EMG signal

Graphical user interface, chart

Description automatically generatedAs you can see, my filtered EMG signal no longer contains the higher frequencies. I have captured what I believe to be the

“envelope” of my EMG signal. To get a rough comparison, I was able to sample the “SIG” pin on the sensor while making a fist and squeezing. Even though figure 11 and 12 are from different samples in time, you can see they are comparable.

# Conclusion

To conclude my work, I would like to mention that I was not able to implement an analog or digital frequency transformation. I attempted to make it work several times, but I kept getting a slightly altered lowpass filter instead of a bandpass filter. Besides this hiccup, I have been able to implement the IIR Butterworth lowpass filter design that was previously learned from EECE 465 computer assignment 2. With some research I was able to obtain the raw EMG signal and process this signal using MATLAB. The previous figures show my work through the process and my results show that the signal has been successfully filtered. Even though I was not able to perform a successful frequency transformation, this project taught me a lot about how to design and implement a digital filter using MATLAB.

##### Acknowledgment

I would like to thank my DSP professor, Dr. Salehi, for teaching us the basics of digital signal processing and providing a foundation for our careers.

##### References

[1] Muhammad Zahak Jamal, Dong-Hyun Lee, and Dong Jin Hyun. Real Time Adaptive Filter based EMG Signal Processing and Instrumentation Scheme for Robust Signal Acquisition Using Dry EMG Electrodes. IEEE Xplore

[2] William Rose. Electromyogram Analysis. www1.udel.edu/biology/rosewc/kaap686/notes/EMG%20analysis.pdf

[3] Mohammad Manzur Murshid, Hassan S. Salehi, "Electromyography signal analysis with real-time support vector machine," Proc. SPIE 11423, Signal Processing, Sensor/Information Fusion, and Target Recognition XXIX, 1142313 (18 May 2020); doi: 10.1117/12.2558962

Figure 12. SIG signal from sensor