Lab 15 & 16

ECG Circuit – Analog Filtering and A/D Conversion

and

Digital Signal Manipulation of the ECG Signal

Julia Rasmussen

Instructor: Iman Salama

Lab Partner: Anfisa Bogdanenko

7 December 2022

1. Introduction

In this lab, two filters were added to the instrumentation amplifier built in the previous lab in order to filter the obtained ECG signal properly. Various signals of non-elevated and elevated heart rates were then obtained using the A/D device. These signals were imported into MATLAB and modified digitally from there. The purpose behind this lab was to learn how to make an ECG device that reads a clean ECG signal from a patient, and learn how to use MATLAB in order to modify it and derive basic data from it. This lab assumed a basic knowledge of waves, ECGs, op-amps, capacitors, and resistors. A working knowledge of the oscilloscope and MATLAB were also assumed. The main constraints to this lab were the sensitivity of the oscilloscope and electrodes, which both determined how clear the resulting ECG signal would be, as well as the sampling rate of the A/D.

2. Preliminary

We are working with an A/D device that ranges in voltage from -10 V to 10 V, with an op-amp that can only output a -1.5 V to 1.5 V signal. There are a total of 16384 levels, so the number of levels within the 3 V window our op-amp uses is $\frac{3\,V}{20\,V} = \frac{x\ levels}{16394\ levels}$, $x = \frac{3}{20} \cdot 16384\ levels = 2457.6\ levels \approx 2458\ levels$. The spacing between each level would be $\frac{20\,V}{16384\ levels} = 0.00814\ \frac{V}{level}$.

The average heart rate is 60 to 100 bpm, so a good lower bound for the low pass filter would be 0.5 Hz. Given that the frequencies for the FFT of ECGs in previous labs seemed to taper off after 100 Hz or so, a high cutoff of 100 Hz for the highpass filter could also work.

3. Results and Analysis

We decided to start with no gain for the high-pass filter and then increase R_S as needed depending on the outputted ECG signal once everything had been connected up. We put four 1 μF capacitors in parallel, for a total capacitance of 4 μF, as well as a R_F value of 200 kΩ, as can be seen below in Figure 1. With this, the cutoff frequency is $\frac{1}{200000 \cdot 4 \cdot 10^{-6} \cdot 2\pi} = 0.199\ Hz$. The filter was then tested using a sine wave generated by the function generator, with the input and output signals both routed to the oscilloscope. For a 1 Hz sine wave, the input voltage was 1.08 V and the output was 1.03 V, as seen in Figure 2. For a 500 mHz sine wave, the input voltage was 1.08 V and the output was 940 mV, seen in Figure 3. At 200 mHz, the input voltage stayed the same, but the output was 690 mV. At 100 mHz, the input stayed the same again, and the output was 440 mV, seen in Figure 4. These tests indicate that the filter worked as intended, as the value found at the cutoff frequency, 690 mV, was quite close to the expected value of about 707 mV, and once the input reached 100 mHz, the output tapered off quite a bit.

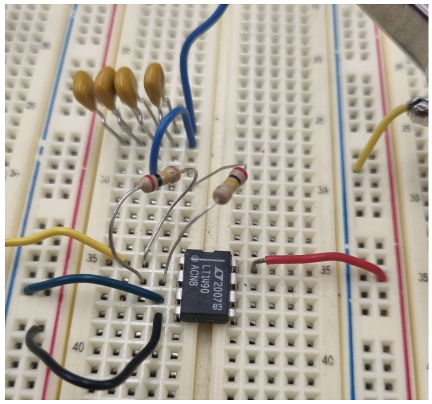

Figure 1: The built 0.199 Hz high-pass filter.

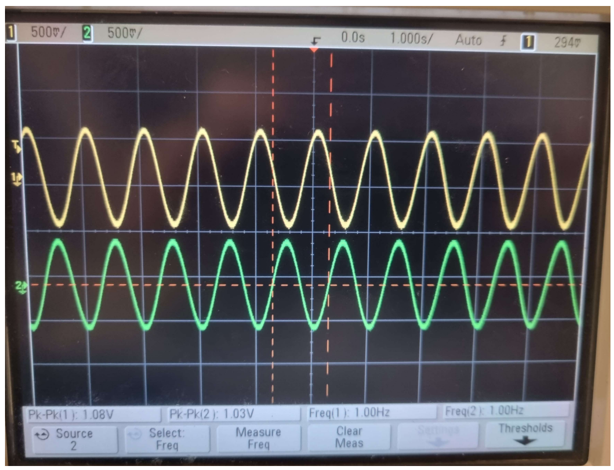

Figure 2: The oscilloscope output for a 1 Hz sine wave through the high-pass filter.

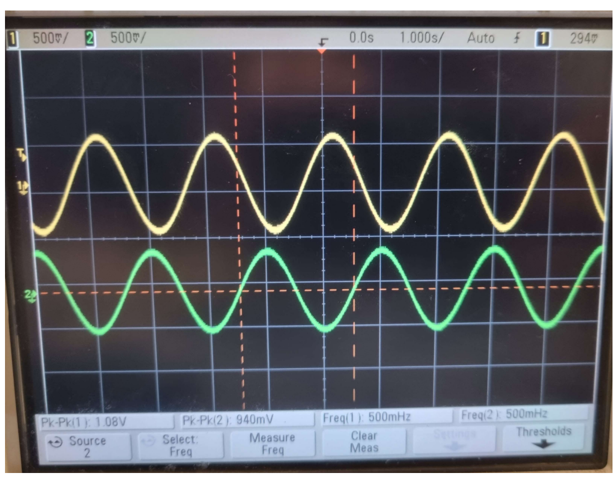

Figure 3: The oscilloscope output for a 500 mHz sine wave through the high-pass filter.

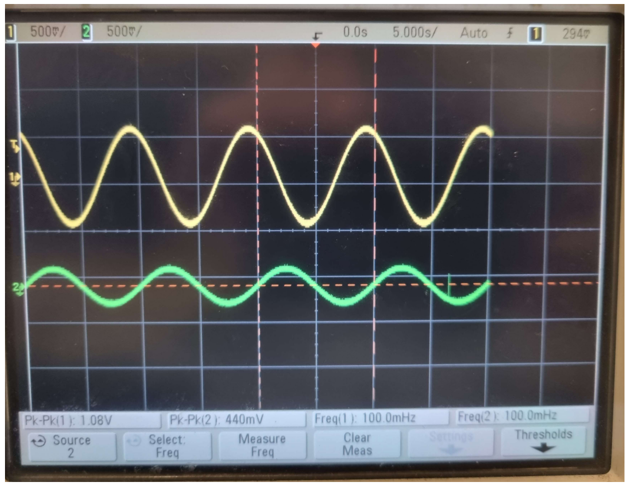

Figure 4: The oscilloscope output for a 100 mHz sine wave through the high-pass filter.

We also decided to start with no gain for the low-pass filter and then increase R_F as needed depending on the outputted ECG signal once everything had been connected up. We decided to use a 7.5 kΩ resistor for R_C and a 0.2 μF capacitor for C, as can be seen below in Figure 5. Thus, the cutoff frequency was $\frac{1}{7500\cdot 0.2\cdot 10^{-6}\cdot 2\pi} = 106.1\ Hz$, or an angular frequency of 666.7 rad/s. The filter was then tested using a sine wave generated by the function generator, with the input and output signals both routed to the oscilloscope. At 10 Hz, the input voltage was 1.06 V, and the output was also 1.06 V, as can be seen below in Figure 6. At 100 Hz, the output was 719 mV, as can be seen in Figure 7, which aligns with the expected value given that it is the cutoff frequency. At 200 Hz, the output was 463 mV.

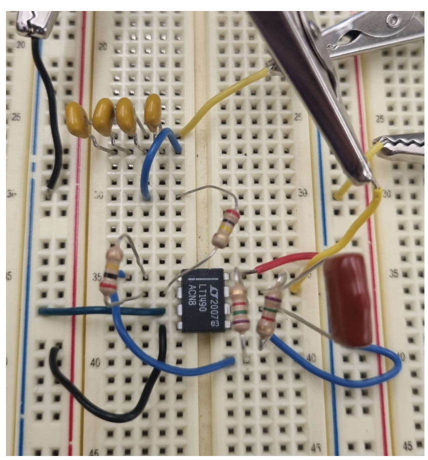

Figure 5: The built 0.199 Hz high-pass filter and 106.1 Hz low-pass filters, to the left and right, respectively.

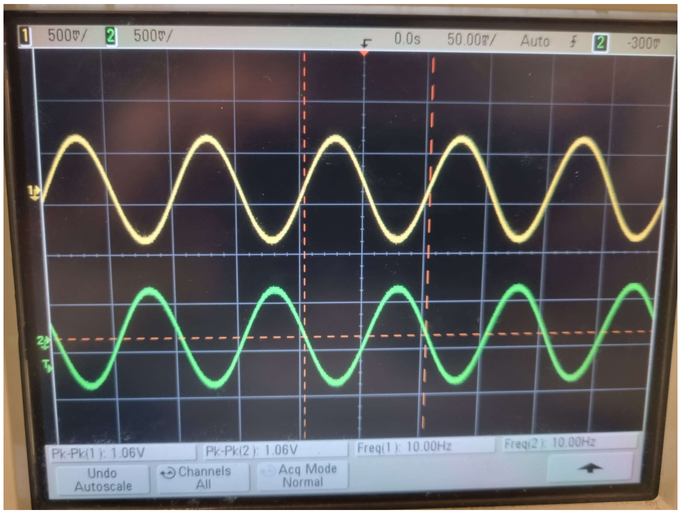

Figure 6: The oscilloscope output for a 10 Hz sine wave through the low-pass filter.

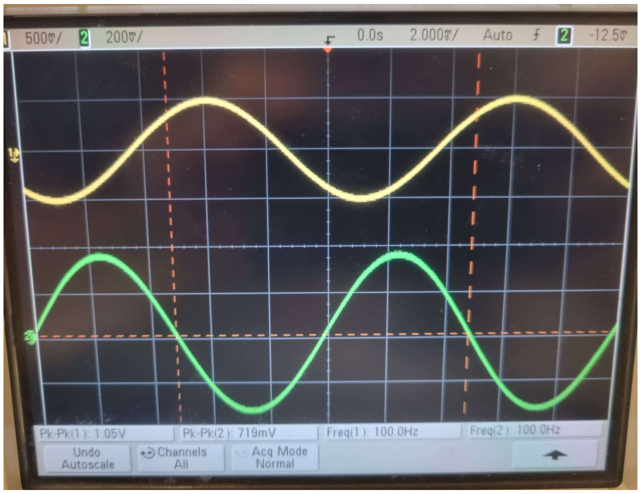

Figure 7: The oscilloscope output for a 100 Hz sine wave through the low-pass filter.

We connected the fully built circuit, seen in Figure 8, to electrodes on the chest, and then used the A/D device to upload it to MATLAB, using a sampling frequency of 500 Hz. We adjusted the value of R_S for the high-pass filter and R_F for the low-pass filter, until the output on the oscilloscope had a peak to peak value of 1.31 V, as can be seen below in Figure 9. The value for R_F for the high-pass filter ended up being 2 MΩ, and the value for R_s in the low-pass filter was adjusted to 1.5 kΩ. The full diagram for the circuit is as shown below in Figure 10. We saved some ECG signals into MATLAB, such as the one shown in Figure 11, from which we then did our analyses. The rescaled ECG signal, based on the overall gain factor, is shown in Figure 12.

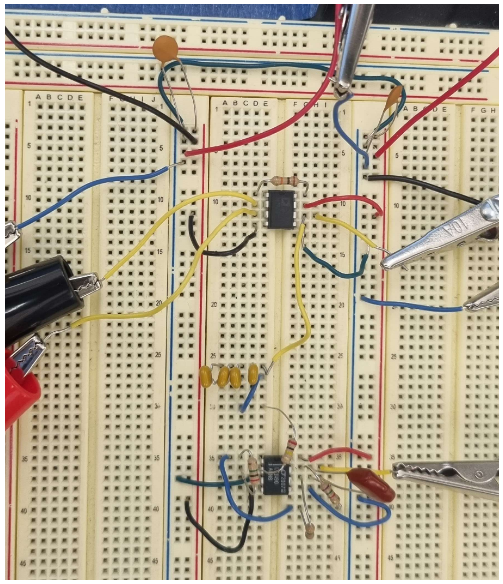

Figure 8: The fully built circuit.

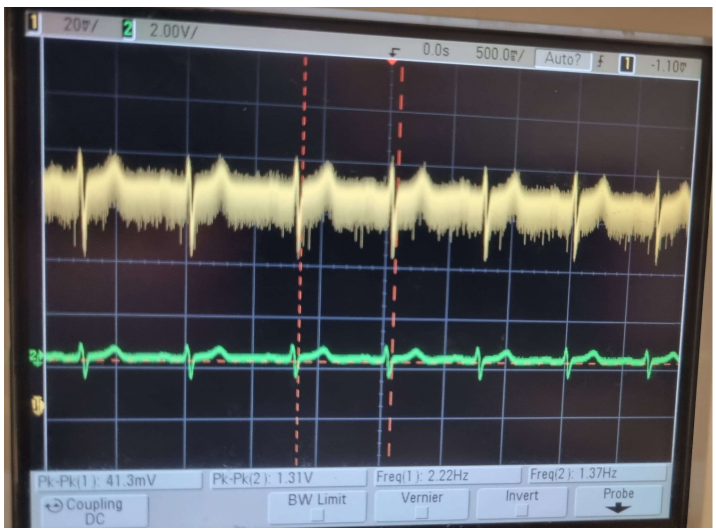

Figure 9: The oscilloscope output from the circuit, while hooked up to electrodes placed on the chest.

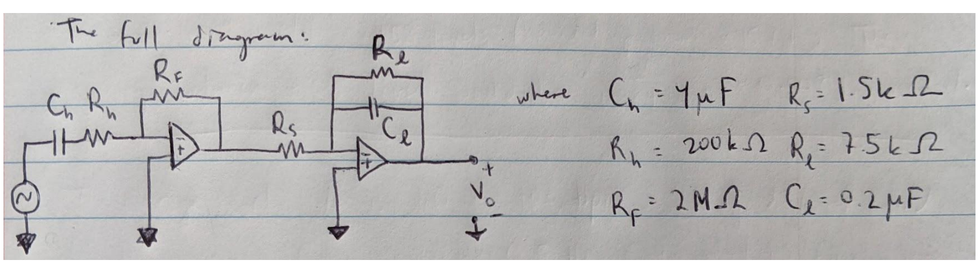

Figure 10: The drawn out diagram for the circuit.

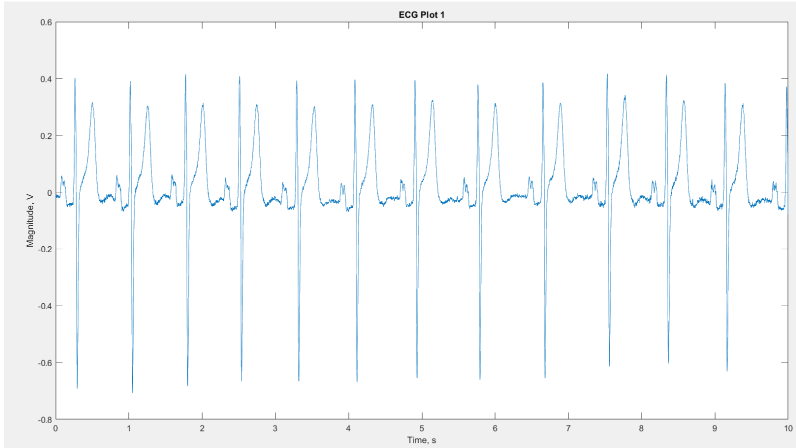

Figure 11: The imported ECG signal, plotted in MATLAB.

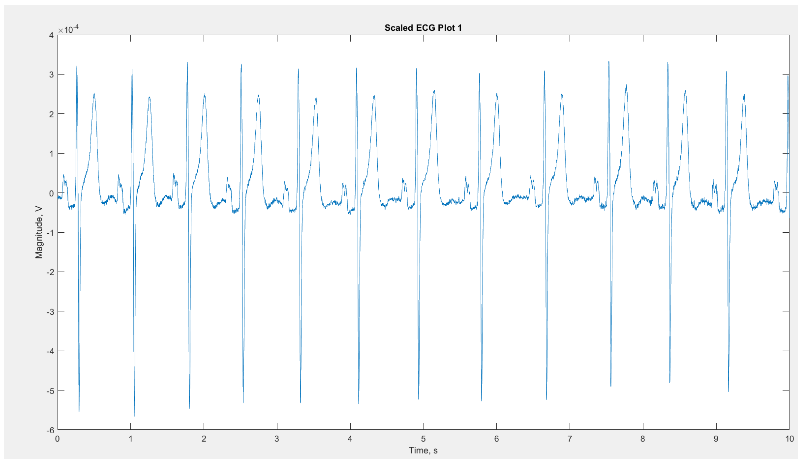

Figure 12: The rescaled imported ECG signal, plotted in MATLAB.

These signals were then modified within MATLAB. First, a low-pass filter with a cutoff frequency of 200 Hz and a total cutoff frequency of 250 Hz was made, as can be seen in Figure 13 and Figure 14 below. This was applied to the signal, in order to get rid of any potential aliasing, and as most of the data for the ECG signal was between about 1 Hz to 100 Hz, this filter mainly removed noise. However, we felt that it removed too little from the original signal, so we tested two other lowpass filters, one with a cutoff frequency of 5 Hz and a total cutoff frequency of 20 Hz, seen below in Figure 15, and then one with a cutoff frequency of 80 Hz and a total cutoff frequency of 100 Hz, as seen in Figure 16. We chose to keep the last one as we felt it got rid of most of the noise while keeping the integrity of the ECG signal.

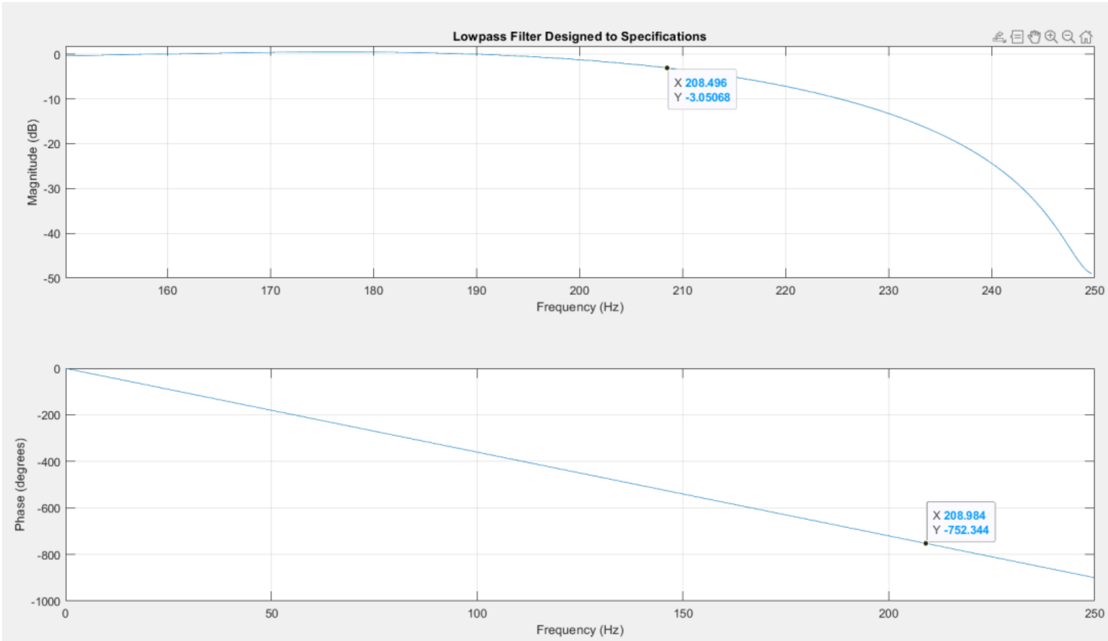

Figure 13: The low-pass filter designed using the firpmord and firprm functions, from 200 Hz to 250 Hz.

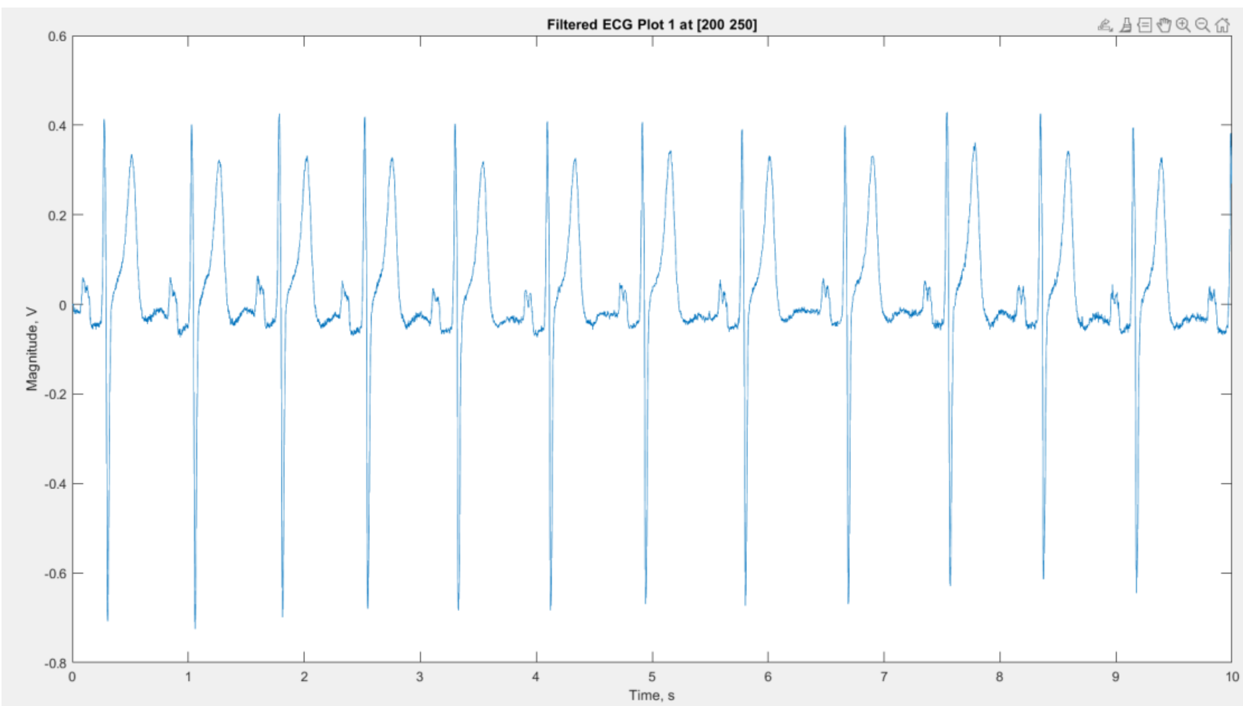

Figure 14: The ECG signal with the 200 Hz to 250 Hz low-pass filter applied.

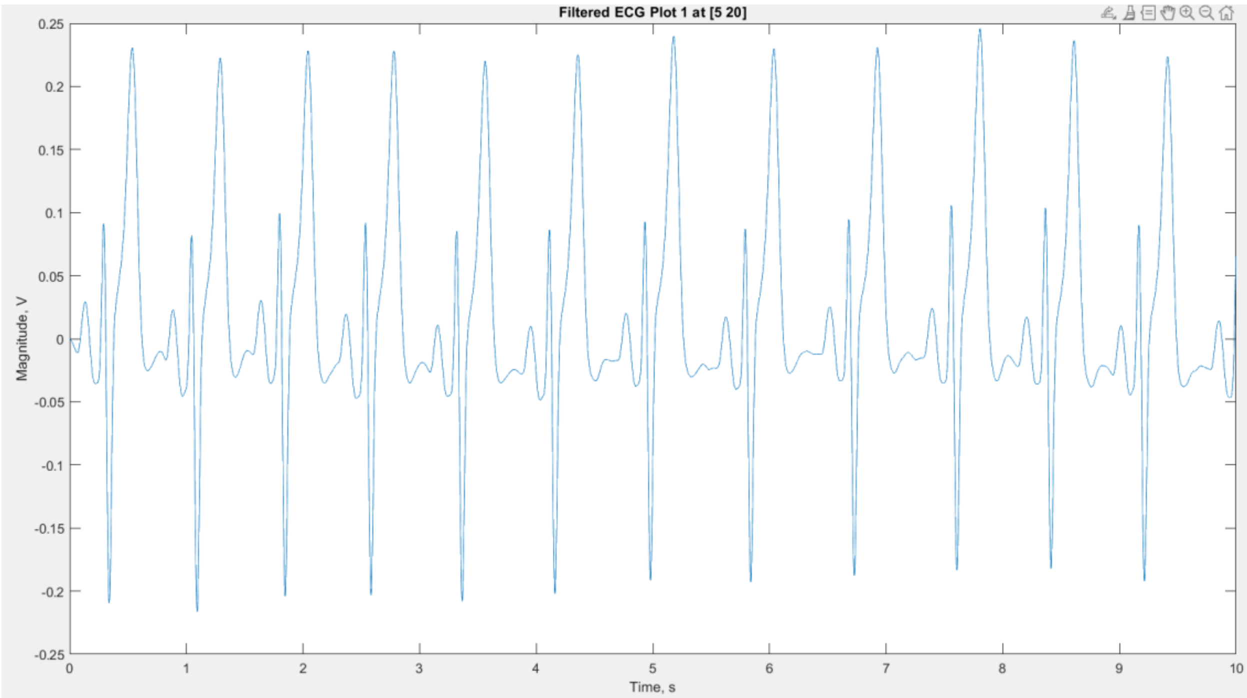

Figure 15: The ECG signal with the 5 Hz to 20 Hz low-pass filter applied.

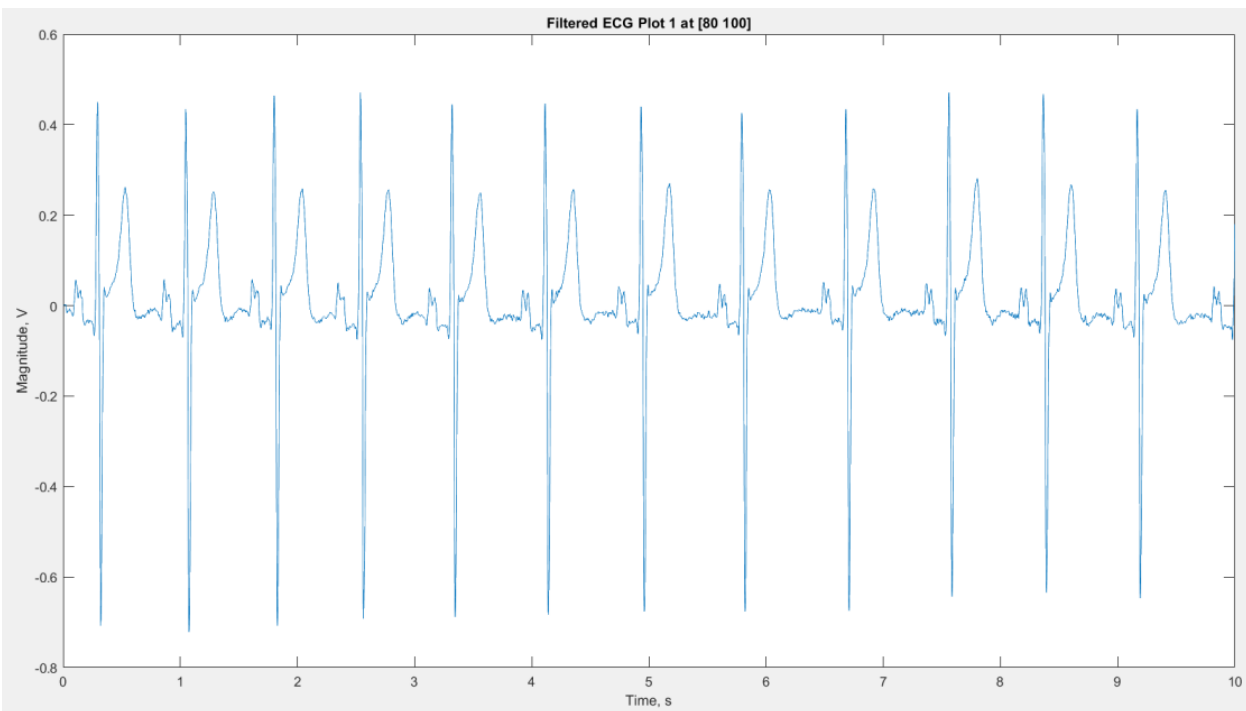

Figure 16: The ECG signal with the 80 Hz to 100 Hz low-pass filter applied.

Next, a notch filter was made using the notch function within MATLAB. The magnitude response of the filter can be seen below in Figure 17, and the signal with and without the filter applied can be seen in Figure 18. The FFTs of the ECG signal with and without the notch filter can be seen in Figure 20 and Figure 19, respectively.

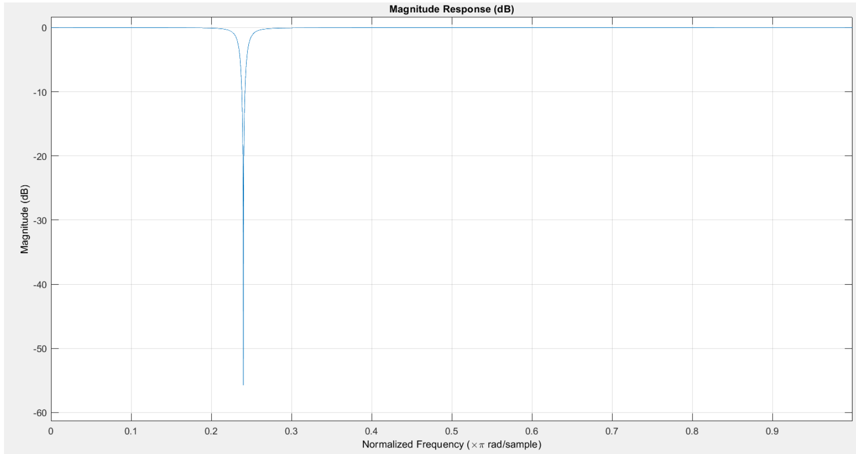

Figure 17: The magnitude response of the notch filter.

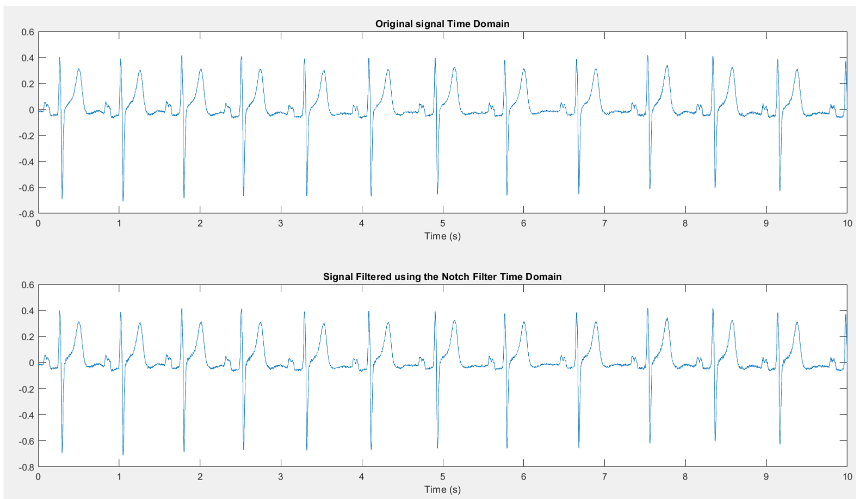

Figure 18: The signal with and without the notch filter.

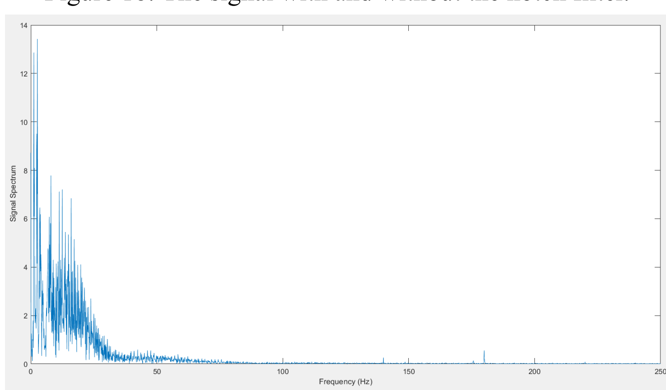

Figure 19: FFT of the ECG signal without the notch filter.

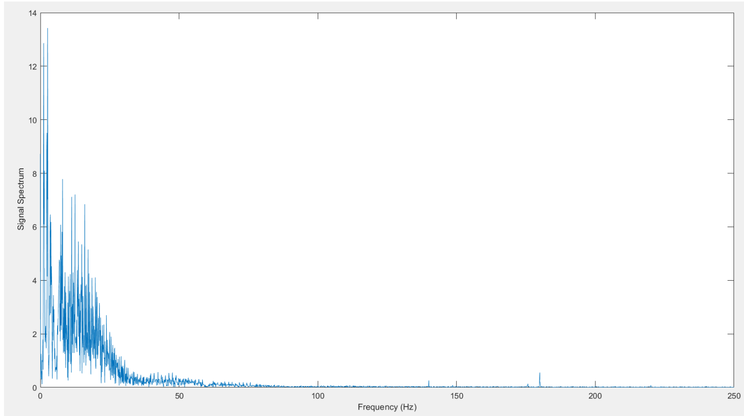

Figure 20: FFT of the ECG signal with the notch filter.

Finally the heart rate was calculated, using the code shown below in Figures 21 and 22. First, the signal was inverted, then everything below a cutoff voltage of 0.2 V was gotten rid of, after which all the peaks of the signal were found using the findpeaks() function. The length of the list of peaks was then found, divided by the length of the plot, which in this case was 10 seconds, and then multiplied by 60 seconds to find the number of peaks, or beats, per minute. This code was then incorporated into the function calc_hr(), after which three sampled ECG signals with elevated heart rates were also inputted into the function in order to see whether it gave the anticipated results, as can be seen in Figures 23, 24, and 25.

```
c5 = -d1
c5(c5 < 0.2) = 0
pks = findpeaks(c5)
```

Figure 21: MATLAB code for detecting the heart rate of the ECG signal.

```
>> (length(pks)/(10))*60

ans =

    72
```

Figure 22: Additional MATLAB code for detecting the heart rate of the ECG signal.

```
>> calc_hr(d3, 0.55, 10)

ans =

   126
```

Figure 23: The calc_hr() function on the first elevated heart rate signal.

```
>> calc_hr(d4, 0.3, 10)

ans =

    96
```

Figure 24: The calc_hr() function on the second elevated heart rate signal.

```
>> calc_hr(d5, 0.3, 10)

ans =

    84
```

Figure 25: The calc_hr() function on the third elevated heart rate signal.

4. Discussions and Conclusions

Through this lab, a deeper understanding of how an instrumentation amplifier can collect an ECG signal, and how that signal can then be filtered with RC op-amp circuitry and filtered and analyzed in MATLAB was developed. First a low-pass and a high-pass filter were built and applied to the incoming ECG signal measured by electrodes. This signal was then uploaded to MATLAB using the A/D device, after which another low-pass filter was put on the signal, a notch filter centering around 60 Hz was applied, and the heart rate was calculated using a script. This lab assumed a familiarity with oscilloscopes, waves, op-amps, resistors, capacitors, ECGs, and instrumentation amplifiers, all experience gained over the course of previous labs. This lab also assumed an understanding of MATLAB. This lab gave a solid foundation for an understanding of how to build an ECG and how such devices actually work. It also gave a window into signal processing, both digitally and with analog circuitry, which has various applications in all sorts of fields, including the medical field, music technology, and radio technology.