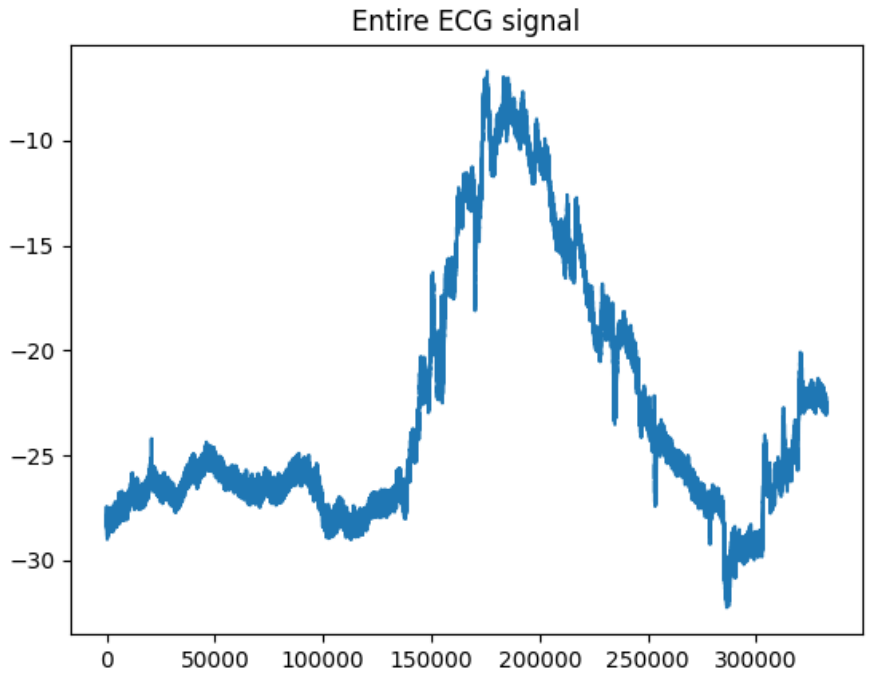
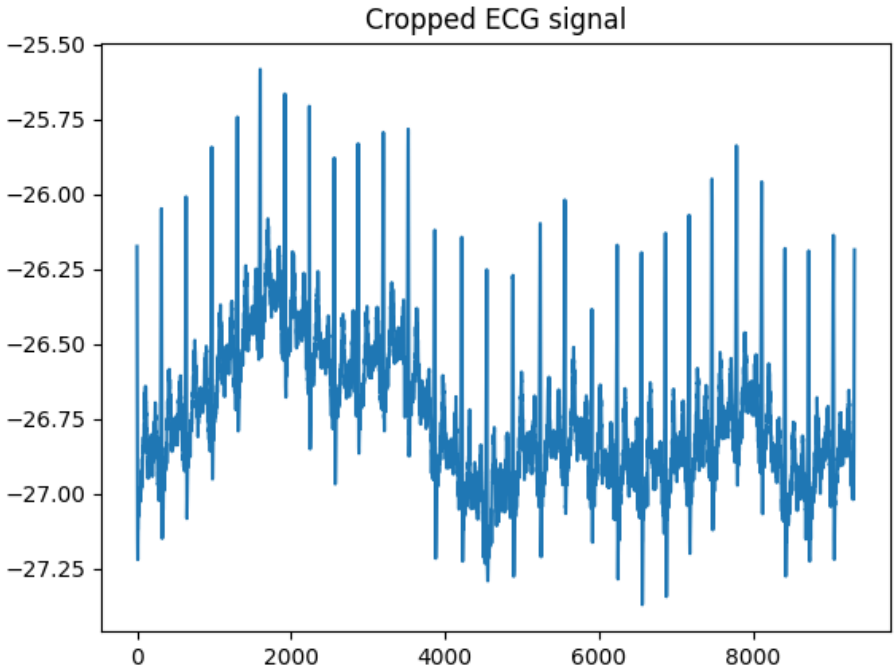
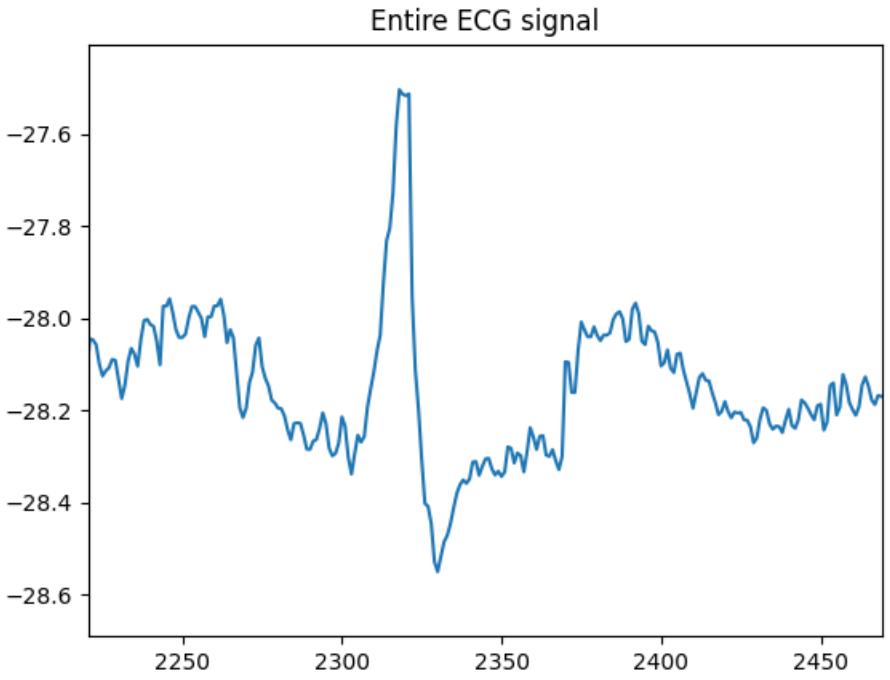
Lab 2 – Kay Burnham – ID 20220414

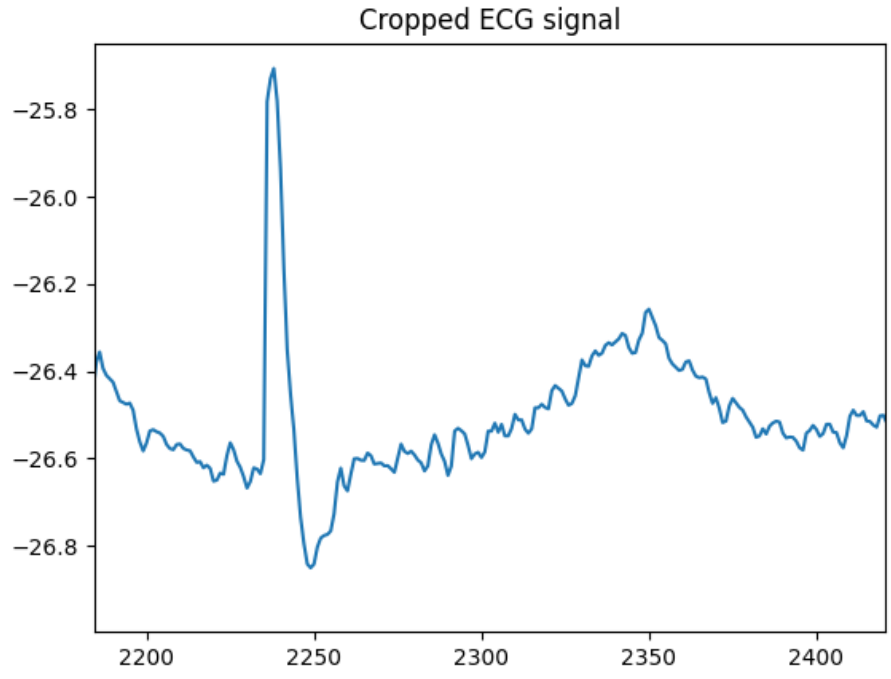
1





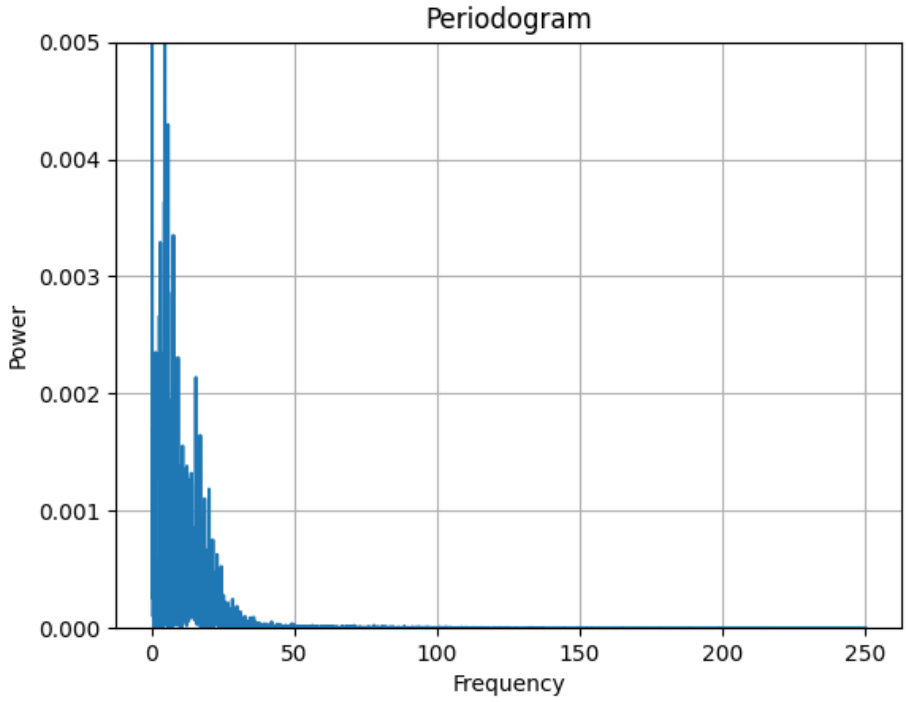
2

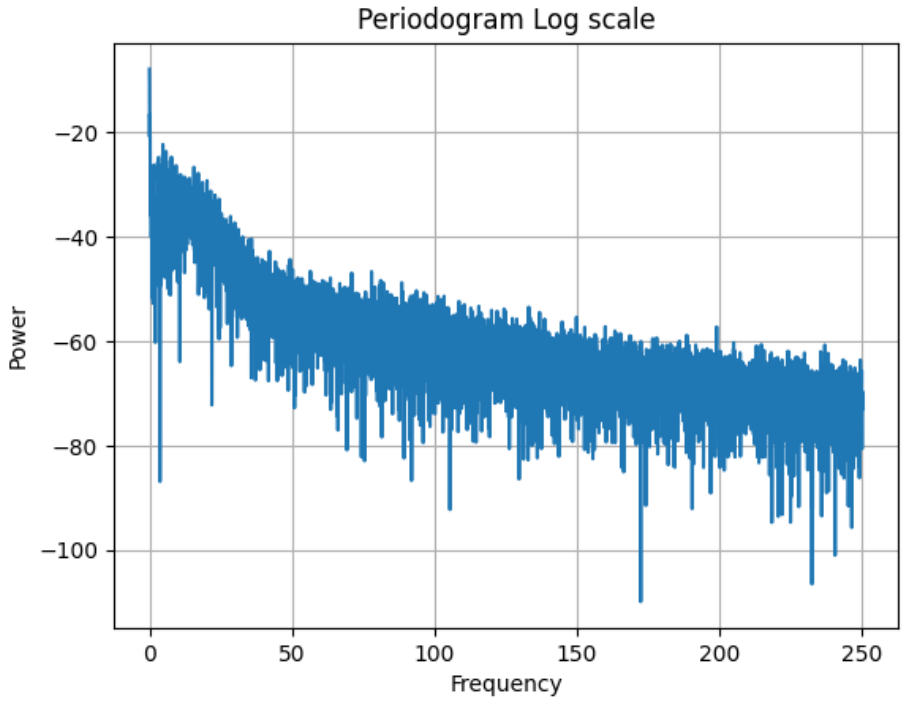




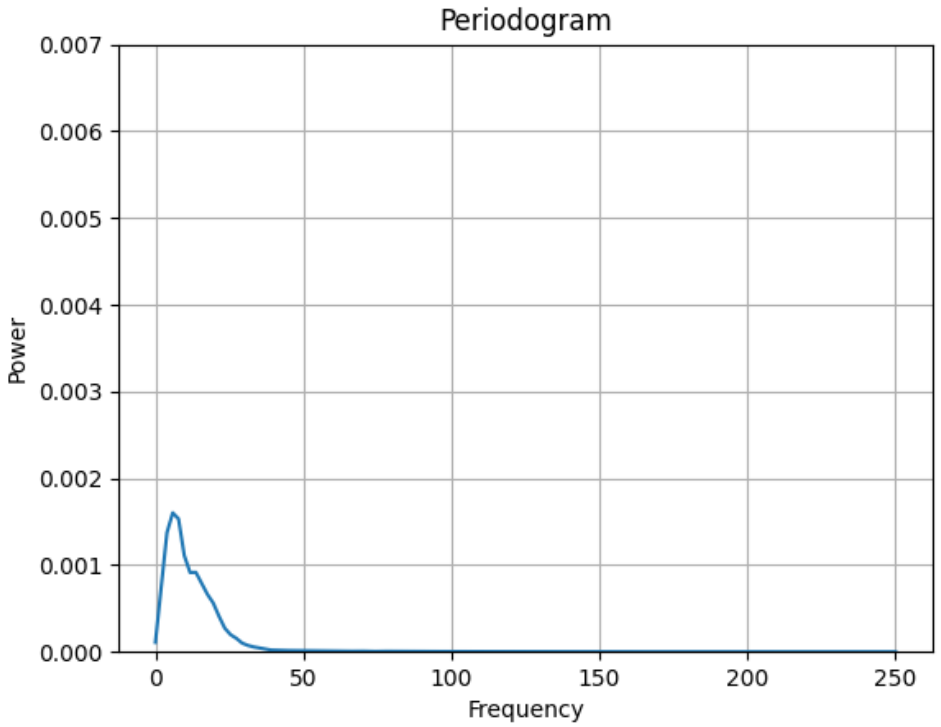
(zoomed in on both plots to the same time frame – interesting that plot is time-shifted?)

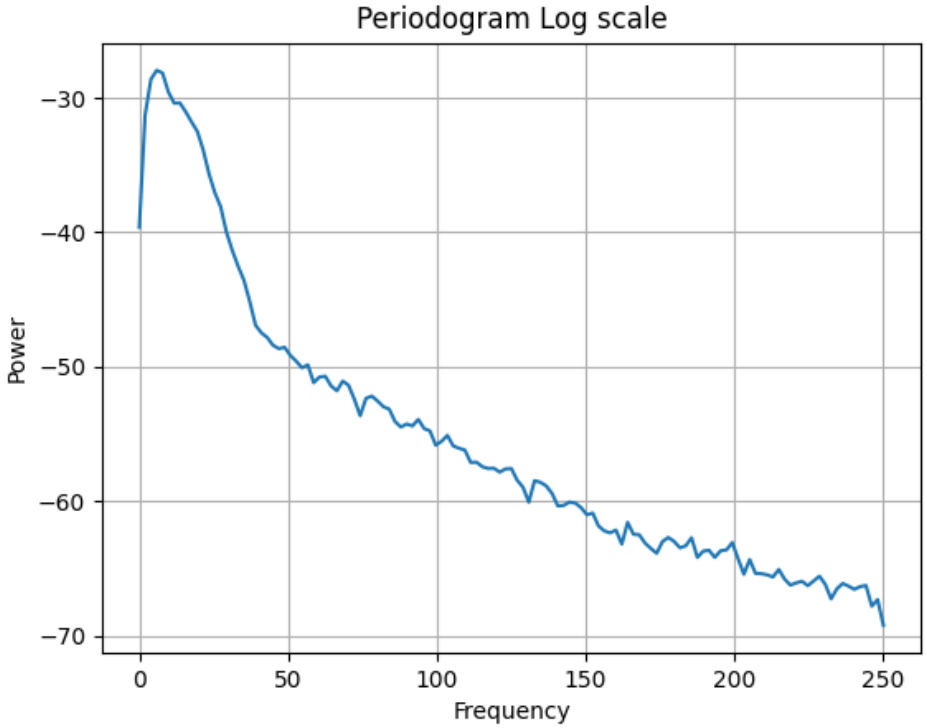
3


Periodogram
0.005
0.004
0.003
0.002
0.001
0.000
Power
0
50
100
150
200
250
Frequency


Periodogram Log scale
−20
−40
−60
−80
−100
Power
0
50
100
150
200
250
Frequency

4


Periodogram
0.007
0.006
0.005
0.004
0.003
0.002
0.001
0.000
Power
0
50
100
150
200
250
Frequency


Periodogram Log scale
−30
−40
−50
−60
−70
Power
0
50
100
150
200
250
Frequency

5

Periodogram (in part 3) has very noisy signal, especially when on a log scale. Difficult to estimate power value at any given frequency. Welch method (in part 4) is much smoother with less variance and is easier to read.

6

The spike corresponds to low-frequency but high-power signals. This is from the bias/offset in the ECG data.

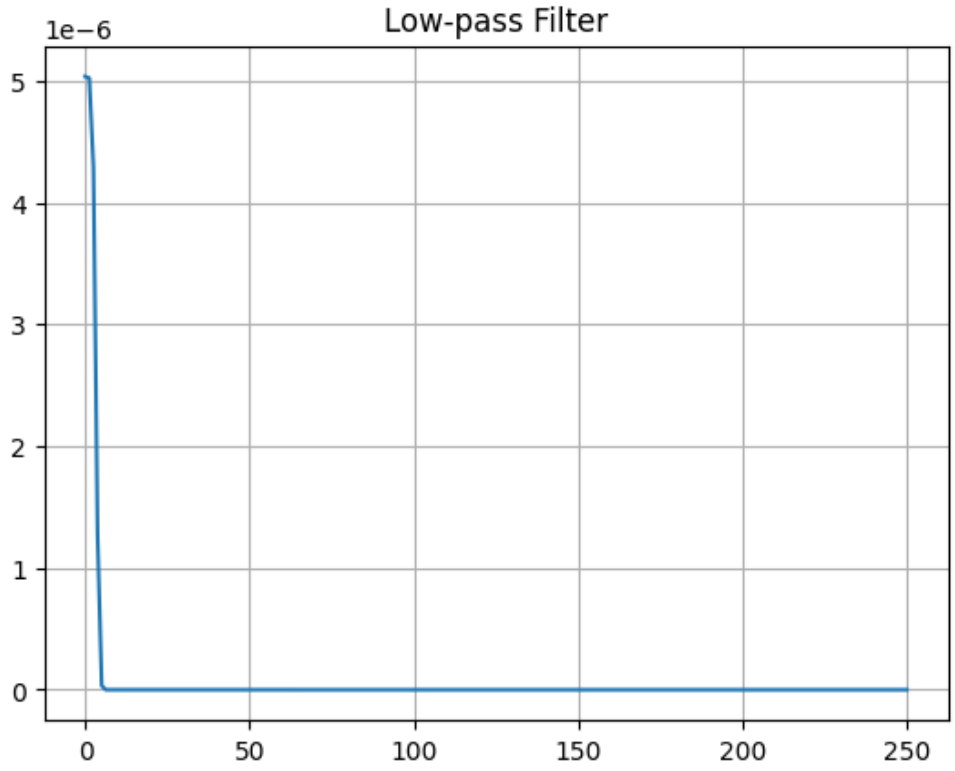
7

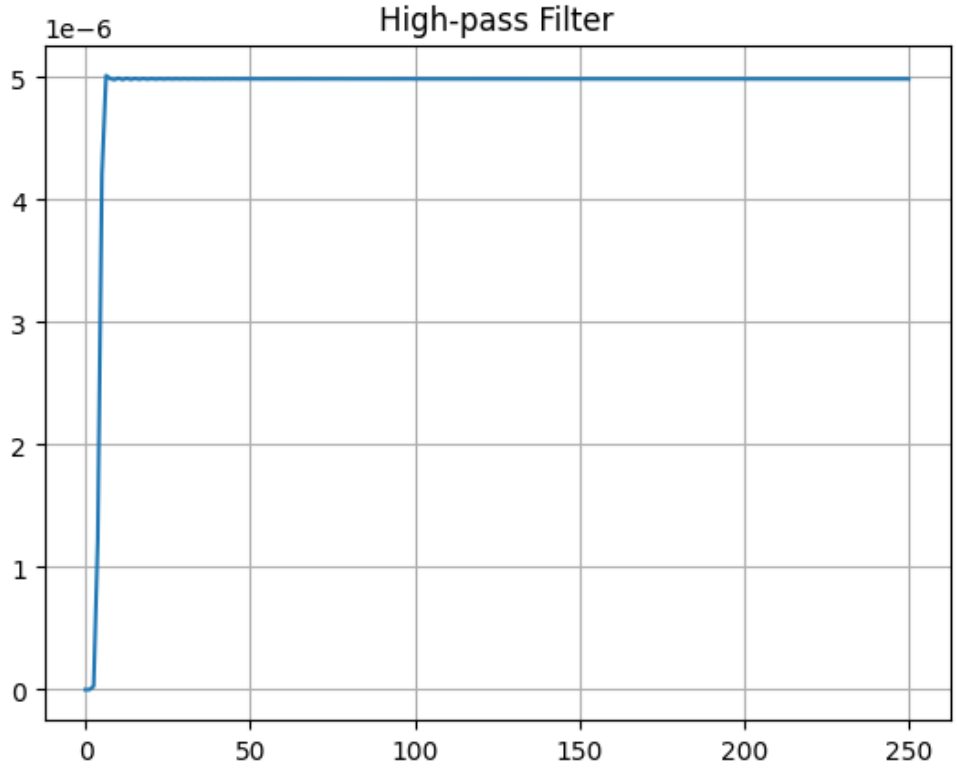
Code for part 7a, 7b, and 8:

```
import matplotlib.pyplot as plt
import lab3 as l3
import numpy as np
import scipy.signal as sci

window_size = 401
cutoff = 5
sampling_freq = 500 # same as previous questions

#lowpass filter:
filter_coefficients_low = sci.firwin(window_size, cutoff, window='hamming',
pass_zero=True,
scale=True, fs=sampling_freq)

# plot the created filter:
freq, pow1 = l3.psd(filter_coefficients_low, sampling_freq)
plt.figure()
plt.plot(freq, pow1)
plt.title('Low-pass Filter')
plt.grid()
plt.show()

#highpass filter:
filter_coefficients_high = sci.firwin(window_size, cutoff, window='hamming',
pass_zero=False,
scale=True, fs=sampling_freq)

# plot the created filter:
freq, pow1 = l3.psd(filter_coefficients_high, sampling_freq)
plt.figure()
plt.plot(freq, pow1)
```

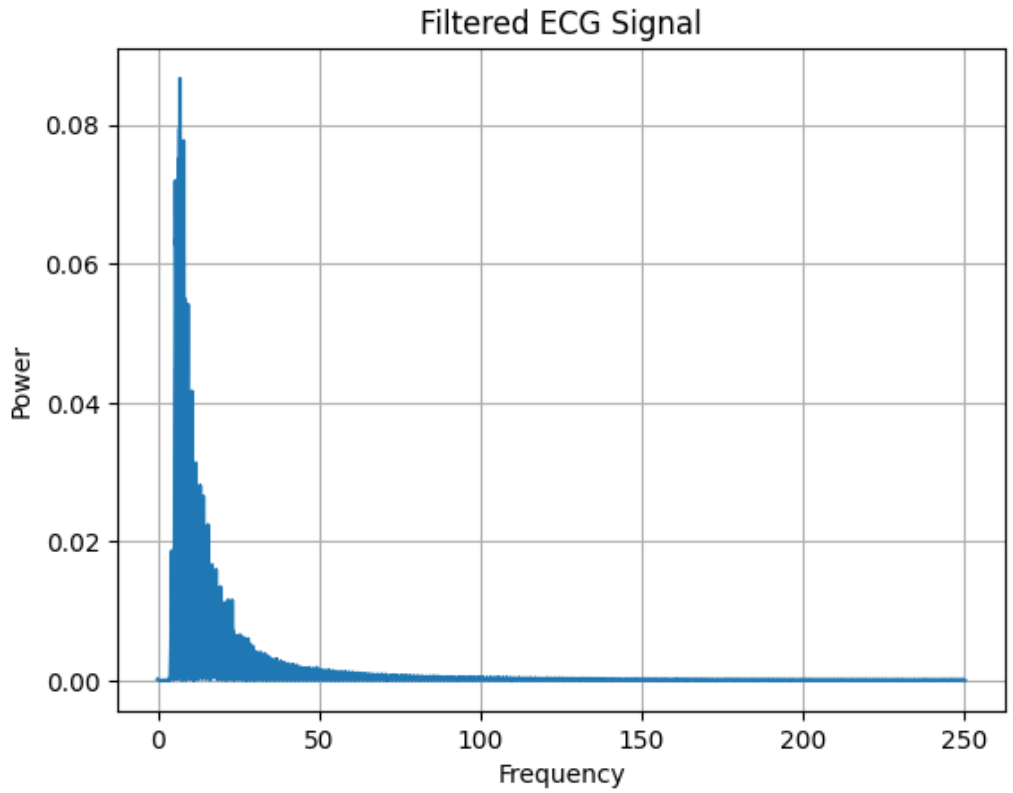
```
plt.title('High-pass Filter')
plt.grid()
plt.show()

# import x and convolve with filter: use high pass to cut offset (spike at beginning)
ecg_timeseries_full=[]

with open('ecg.csv') as f:
    for line in f.readlines():
        ecg_timeseries_full.append(float(line.strip()))

x = ecg_timeseries_full[71740:81060]
x_filtered = np.convolve(x, filter_coefficients_high)

# print filtered ECG signal
freq, pow1 = l3.psd(x_filtered, sampling_freq)
plt.figure()
plt.plot(freq, pow1)
plt.title('Filtered ECG Signal')
plt.xlabel('Frequency')
plt.ylabel('Power')
plt.grid()
plt.show()
```

Filter plots (7a and 7b):





8

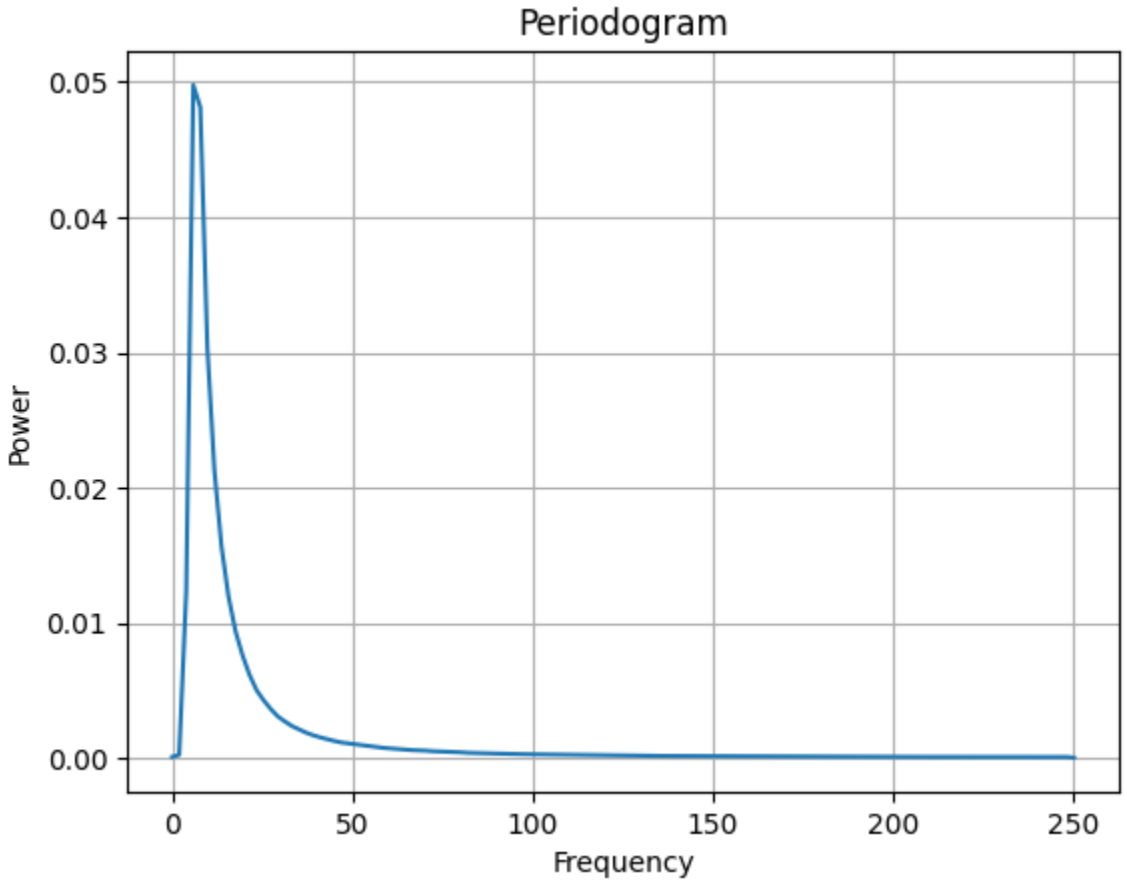


The highpass filter was chosen to eliminate the bias spike at very low frequency.

9

Compared to the original signal x, this filter results in a periodogram with fewer small spikes, indicating that the hamming window has performed as expected to highlight certain (desired) frequencies. However, the graph is still difficult to read for precise values.

10



11

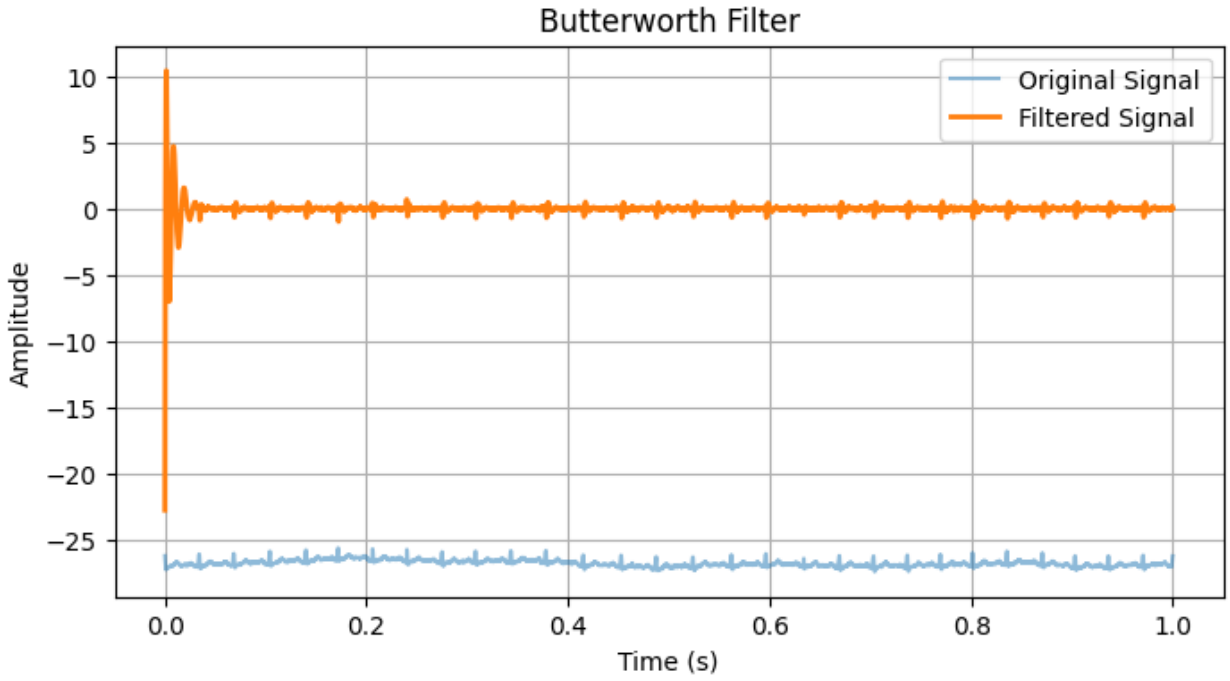
The periodogram now reads much more clearly than before. As stated, there is no need to limit the y-axis, as the initial spike (bias) has been removed by the filter.

12

Code for the Butterworth IIR filter:

```
filter_order = 7
Wn = 5

b, a = sci.butter(filter_order, Wn, btype='high', analog=False, output='ba',
fs=sampling_freq) # sampling is same as previous questions (500 Hz)
```

13



With plt.ylim([-2, 2]) :