In [92]:
```
import os
import matplotlib.pyplot as plt
import numpy as np
from scipy.io.wavfile import read
from scipy.io.wavfile import write
from IPython.display import Audio
from scipy import signal
%matplotlib inline
print(os.getcwd())
```

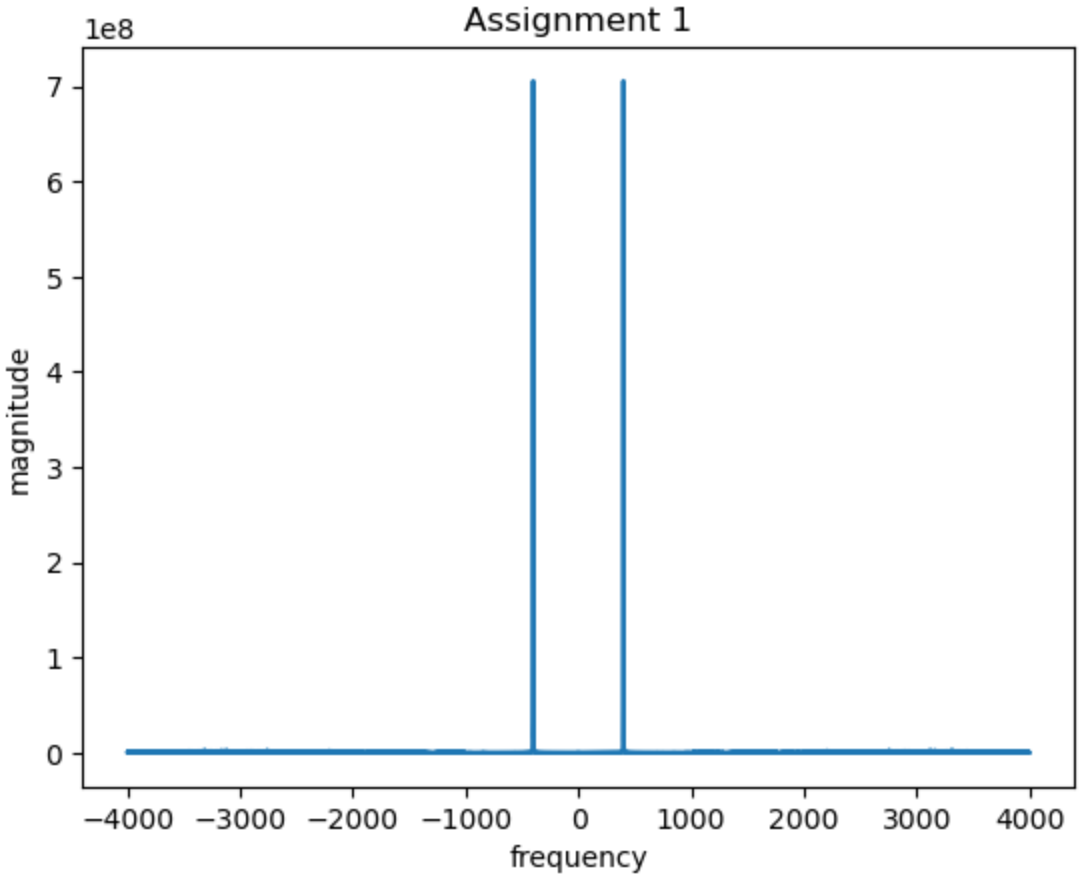
```
C:\Users\Yicheng Zhou\OneDrive\Desktop\UIUC\ECE 420\Lab 02
```

In [3]:
```
sampling_rate, data = read('with_hum.wav')
Audio('with_hum.wav')
```

Out[3]:

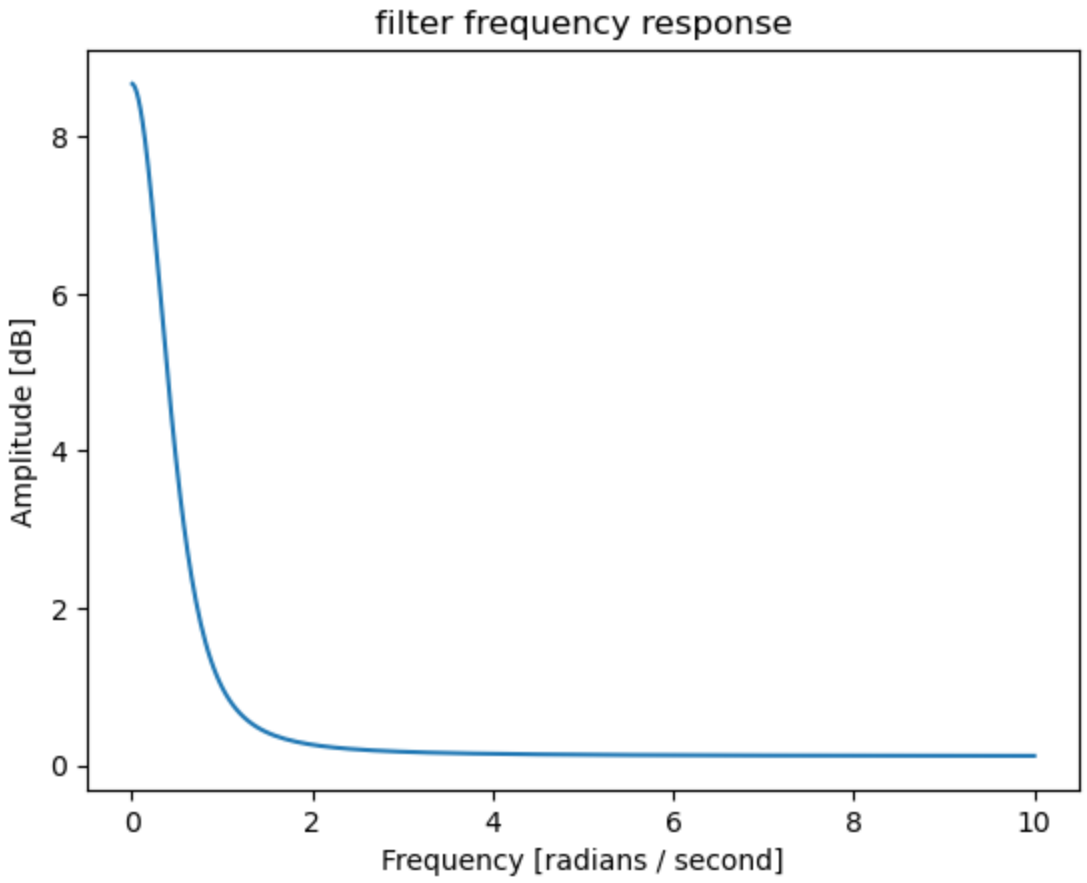
0:00 / 0:09

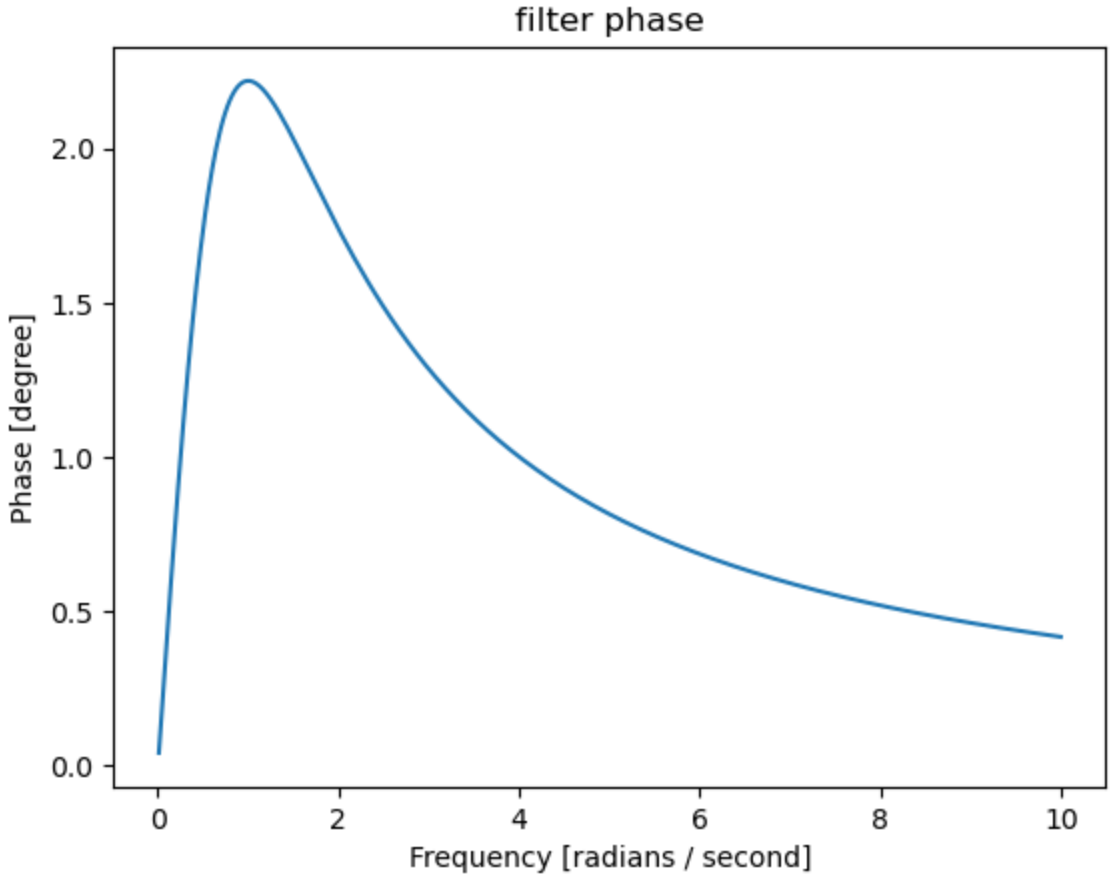
In [12]:
```
sp = np.fft.fft(data)
freq = np.fft.fftfreq(len(data),1/sampling_rate)
plt.plot(freq,abs(sp))
plt.title('Assignment 1')
plt.ylabel('magnitude')
plt.xlabel('frequency')
plt.show()
```



In [83]:
```
b,a = signal.butter(10,420,'highpass',fs=sampling_rate*0.5)
w,h=signal.freqs(b,a)
plt.plot(w,abs(h))
plt.title('filter frequency response')
plt.xlabel('Frequency [radians / second]')
plt.ylabel('Amplitude [dB]')
plt.figure()
plt.plot(w,np.angle(h))
plt.title('filter phase')
plt.xlabel('Frequency [radians / second]')
plt.ylabel('Phase [degree]')
print (sampling_rate)
```

```
8000
```

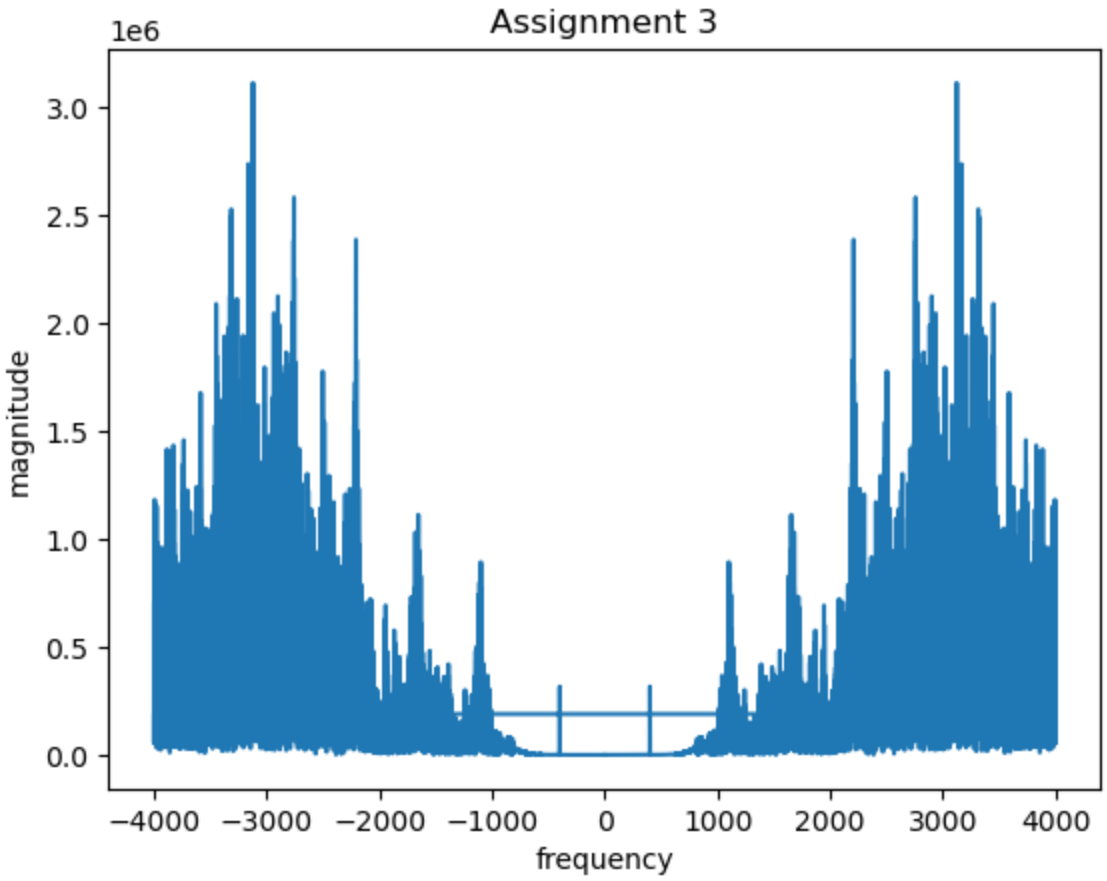




In [85]:
```
#The design has 10 taps, this is due to with lower taps, the 400HZ signal is n
#If we raise the cut_off freq in the butter function, we would be able to redu
#but we might lose useful signals too.
#with lower taps, the delay and resource utilization is less.
```

In [86]:
```
data_out = signal.lfilter(b,a,data)
data_out_fft = np.fft.fft(data_out)
data_freq = np.fft.fftfreq(data_out_fft.size,1/sampling_rate)

plt.plot(data_freq,abs(data_out_fft))
plt.title('Assignment 3')
plt.ylabel('magnitude')
plt.xlabel('frequency')
plt.show()
print(abs(data_out))
print(abs(data))
```




```
[2614.50874689 8648.45673571 4504.66051599 ... 1585.62029562 2027.32029658
  570.29930667]
[22680 20520 12859 ... 21027 19391 13993]
```

In [87]:
```
Audio('without_hum.wav')
```

Out[87]:

0:01 / 0:09

In [100]:
```
write("processed.wav",sampling_rate,data_out.astype(np.int16))
Audio('processed.wav')
```

Out[100]:

0:02 / 0:09