

Figure 2: y-direction magnitude by time plot and corresponding Fourier transform

Figure 1: x-direction magnitude by time plot and corresponding Fourier transform

Figure 3: z-direction magnitude by time plot and corresponding Fourier transform

1. **import** numpy as np
2. **import** scipy as sy
3. **import** scipy.fftpack as syfp
4. **import** matplotlib.pyplot as plt
6. array = np.loadtxt("D:\\Desktop\\accelxyz.csv", delimiter=',')
7. column\_num = 2  #0 means x-axis, 1 means y-axis and 2 means z-axis
9. length = len(array[:,column\_num]) #Number of data points
10. x = sy.linspace(0.005, length\*0.008, num=length) #Return evenly spaced numbers as x-axis values
12. yf = syfp.fft(array[:,column\_num]) #Discrete Fourier transform of array
13. f = syfp.fftfreq(length, np.mean(np.diff(x))) #Return the Discrete Fourier Transform sample frequencies
15. plt.subplot(211) #Create signal magnitude by time plot
16. plt.plot(x, array[:,column\_num]) #Plot values
17. plt.subplot(212) #Create magnitude by frequency plot
18. plt.plot(abs(f), abs(yf)) #Plot values
19. plt.show()

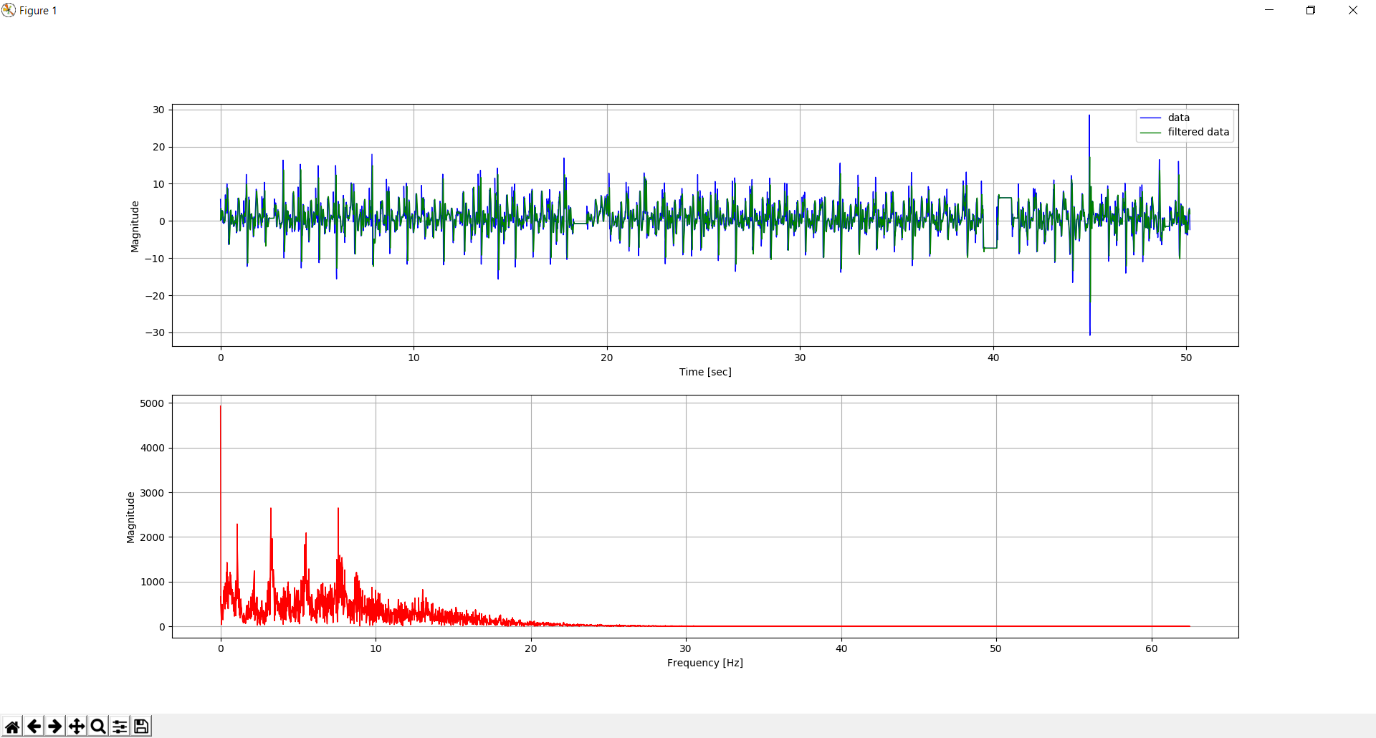
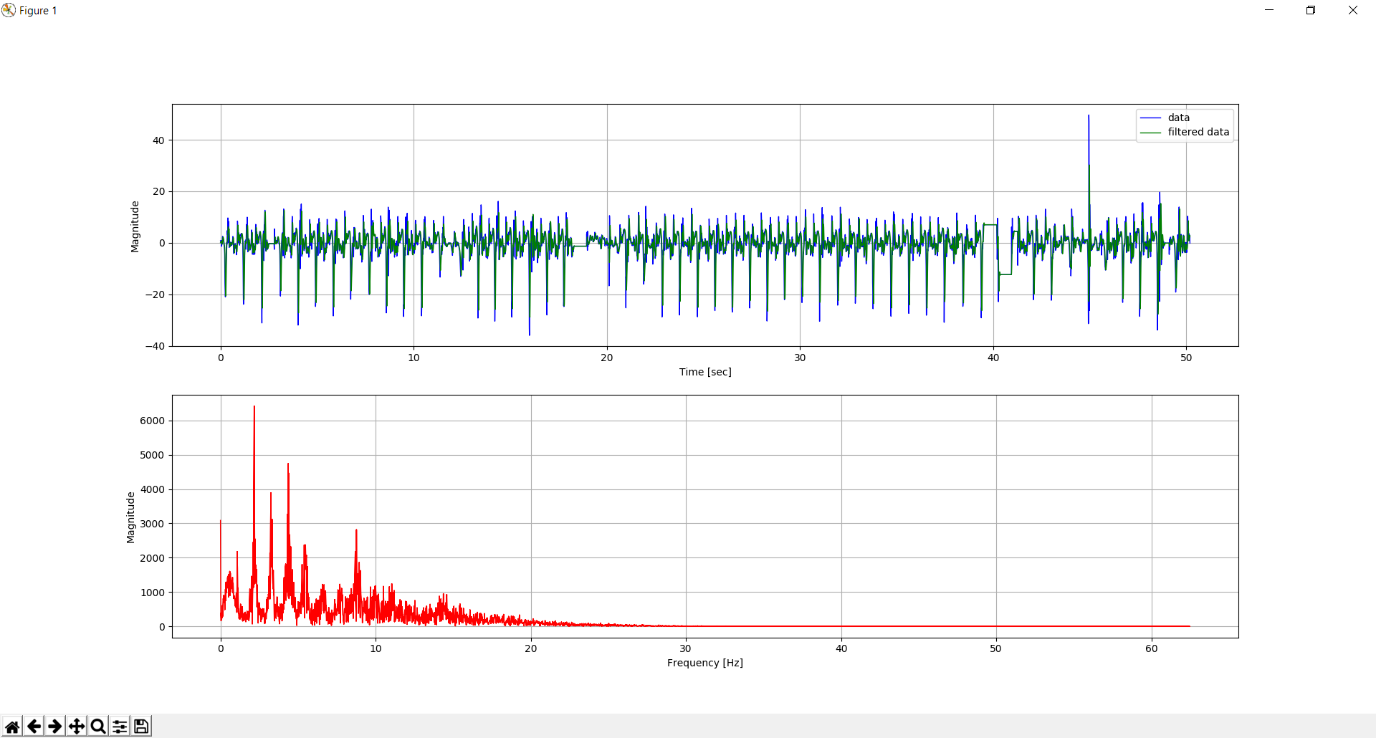
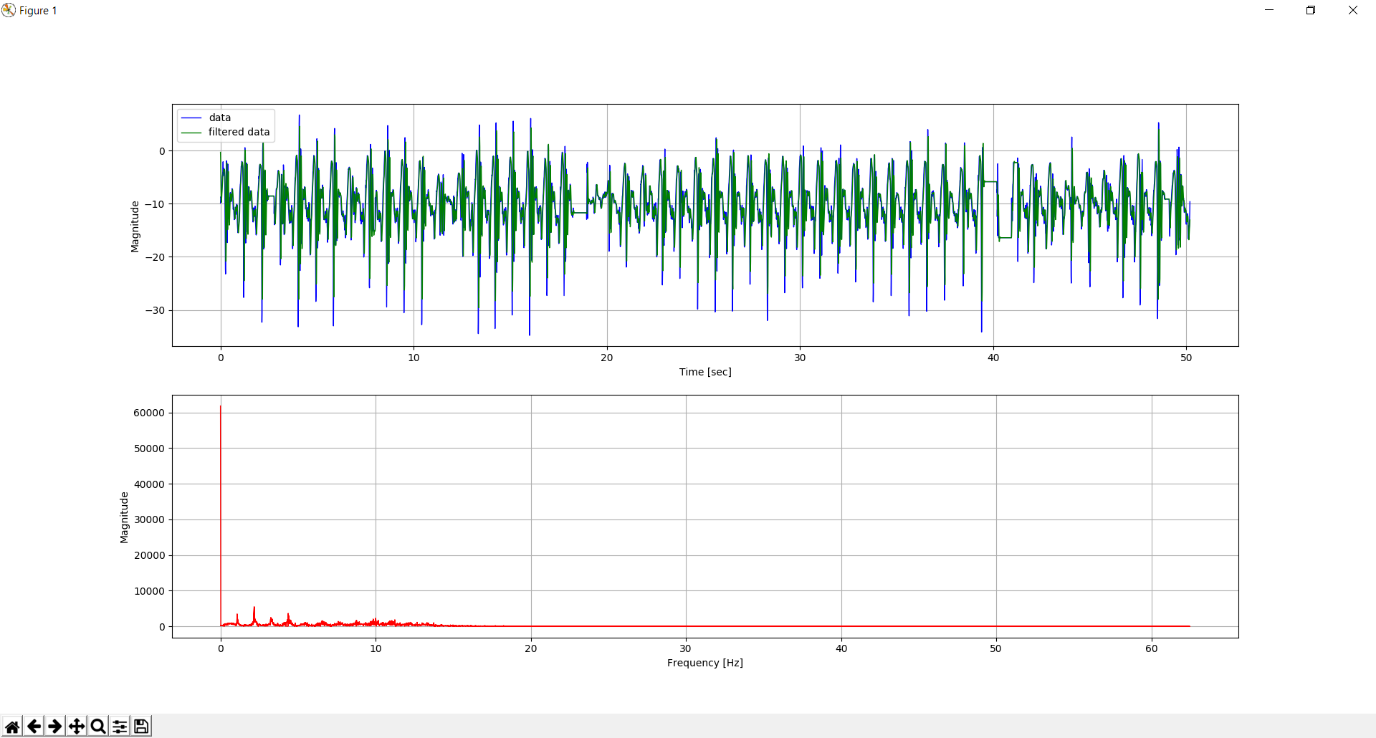


Figure 6: Original and filtered sensor data in the z-direction along with Fourier transform of filtered sensor data

Figure 5: Original and filtered sensor data in the y-direction along with Fourier transform of filtered sensor data

Figure 4: Original and filtered sensor data in the x-direction along with Fourier transform of filtered sensor data

1. **import** numpy as np
2. **import** scipy as sy
3. **import** scipy.fftpack as syfp
4. **import** matplotlib.pyplot as plt
5. **from** scipy.signal **import** butter, lfilter, freqz
7. **def** butter\_lowpass(cutoff, fs, order=3):
8. nyq = 0.5 \* fs
9. normal\_cutoff = cutoff / nyq
10. b, a = butter(order, normal\_cutoff, btype='low', analog=False)
11. **return** b, a
13. **def** butter\_lowpass\_filter(data, cutoff, fs, order=3):
14. b, a = butter\_lowpass(cutoff, fs, order=order)
15. y = lfilter(b, a, data)
16. **return** y
18. array = np.loadtxt("D:\\Desktop\\accelxyz.csv", delimiter=',')
19. column\_num = 0 #0 means x-axis, 1 means y-axis and 2 means z-axis
21. order = 3 # filter order
22. fs = 125.0 # sample rate, Hz
23. cutoff = 15 # cut-off frequency
25. length = len(array[:,column\_num]) #Number of data points
26. x = sy.linspace(0.005, length\*0.008, num=length) #Return evenly spaced numbers as x-axis values
28. y = butter\_lowpass\_filter(array[:,column\_num], cutoff, fs, order)
30. plt.subplot(2, 1, 1)
31. plt.plot(x, array[:,column\_num], 'b-', linewidth=1, label='data')
32. plt.plot(x, y, 'g-', linewidth=1, label='filtered data')
33. plt.xlabel('Time [sec]')
34. plt.ylabel('Magnitude')
35. plt.grid()
36. plt.legend()
38. yf = syfp.fft(y) #Discrete Fourier transform of array
39. f = syfp.fftfreq(length, np.mean(np.diff(x))) #Return the Discrete Fourier Transform sample frequencies
41. plt.subplot(212) #Create magnitude by frequency plot
42. plt.plot(abs(f), abs(yf), 'r-', linewidth=1) #Plot values
43. plt.xlabel('Frequency [Hz]')
44. plt.ylabel('Magnitude')
45. plt.grid()
46. plt.show()

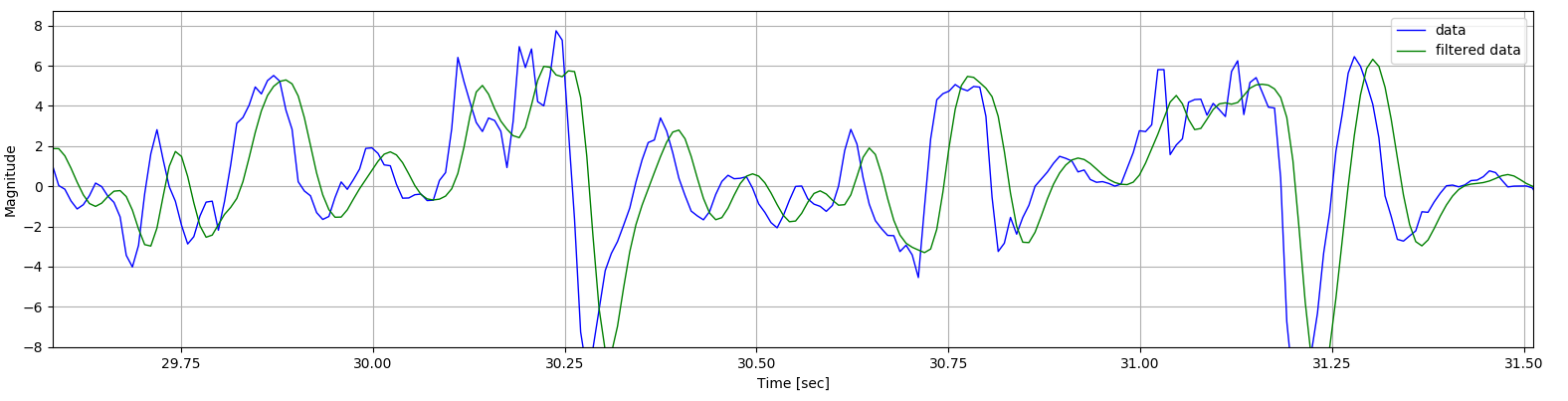


Figure 7: Closer look at raw data and filtered data