Python code:

#######################

#Willard Wider

#7/20/18

#ELEC4400

#Lab 10

#######################

import matplotlib.pyplot as plt

import numpy as np

from scipy.fftpack import fft, ifft

from scipy.signal import lfilter, freqz

#specify the stype of the plot to use

plt.style.use('ggplot')

#problem 1 - FFT and IFFT

#make the time domain signal. 1000 data points, 2 seconds in duration

x\_index = np.linspace(0,2,1000)

#define the input signal (xt)

input\_singal = np.cos(2\*np.pi\*10\*x\_index) + np.cos(2\*np.pi\*100\*x\_index) + np.cos(2\*np.pi\*200\*x\_index)

#the fft of the input signal

x\_index\_frequency = np.linspace(0,500,1000)

fft\_input\_out = fft(input\_singal)

#the inverse fft of the fft (meta)

i\_fft\_input\_out = ifft(fft\_input\_out)

#create three graphs for plotting

#graph of first plot

f,xarr = plt.subplots(3)

plt.tight\_layout()

xarr[0].set\_title('noisy signal')

xarr[0].plot(x\_index,input\_singal)

#graph of fft of first

xarr[1].set\_title('fft out')

#https://stackoverflow.com/questions/15858192/how-to-set-xlim-and-ylim-for-a-subplot-in-matplotlib

xarr[1].set\_xlim([0,250])

xarr[1].plot(x\_index\_frequency,np.abs(fft\_input\_out))

#graph of inverse of fftr

xarr[2].set\_title('ifft')

xarr[2].plot(x\_index,i\_fft\_input\_out)

#https://stackoverflow.com/questions/13228254/subplots-are-overlapping-axis-labels

plt.tight\_layout()

plt.show()

#problem 2 - time domain filtering

N = 15

b = (1/N)\*np.ones(N)

#input\_singal=xt

filtered\_output = lfilter(b,1,input\_singal)

p2\_f,p2\_xarr = plt.subplots(2)

#signal first

p2\_xarr[0].set\_title('noisy signal')

p2\_xarr[0].plot(x\_index,input\_singal)

p2\_xarr[1].set\_title('filtered signal')

p2\_xarr[1].plot(x\_index, filtered\_output)

plt.tight\_layout()

plt.show()

#probmel 3 - frequency domain filtering

#check for sampled rate

hz,H = freqz(b,1,worN=x\_index\_frequency\*2\*np.pi/500)

hzTrue = hz\*500/(2\*np.pi)

fft\_input\_out\_filtered = fft\_input\_out\*H

#create three graphs for plotting

#graph of first plot

p3\_f,p3\_xarr = plt.subplots(3)

p3\_xarr[0].set\_title('noisy fft signal')

p3\_xarr[0].set\_xlim([0,250])

p3\_xarr[0].plot(x\_index\_frequency,np.abs(fft\_input\_out))

#graph of filter frequency response

p3\_xarr[1].set\_title('filter frequency response')

p3\_xarr[1].set\_xlim([0,250])

p3\_xarr[1].plot(hzTrue,H)

#graph result of frequncy domain filtering

p3\_xarr[2].set\_title('frequency domain filtered signal')

p3\_xarr[2].set\_xlim([0,250])

#xarr[1].plot(x\_index\_frequency,np.abs(fft\_input\_out))

p3\_xarr[2].plot(x\_index\_frequency,np.abs(fft\_input\_out\_filtered))

plt.tight\_layout()

plt.show()

#problem 4 - Prestige

p4\_f,p4\_xarr = plt.subplots(4)

#graph of time domain filtering method

p4\_xarr[0].set\_title('time domain filtering, time domain')

p4\_xarr[0].plot(x\_index, filtered\_output)

#graph of fft of time domain filter

p4\_xarr[1].set\_title('time domain filtering, frequency domain')

time\_domain\_fft = fft(filtered\_output)

p4\_xarr[1].set\_xlim([0,250])

p4\_xarr[1].plot(x\_index\_frequency, np.abs(time\_domain\_fft))

#graph of frequency domain filter, feqeucncy domain

p4\_xarr[2].set\_title('frequency domain filtering, frequency domain')

p4\_xarr[2].set\_xlim([0,250])

p4\_xarr[2].plot(x\_index\_frequency,np.abs(fft\_input\_out\_filtered))

#graph of frequency domain filter, time domain

p4\_xarr[3].set\_title('frequency domain filtering, time domain')

time\_domain\_ifft = ifft(fft\_input\_out\_filtered)

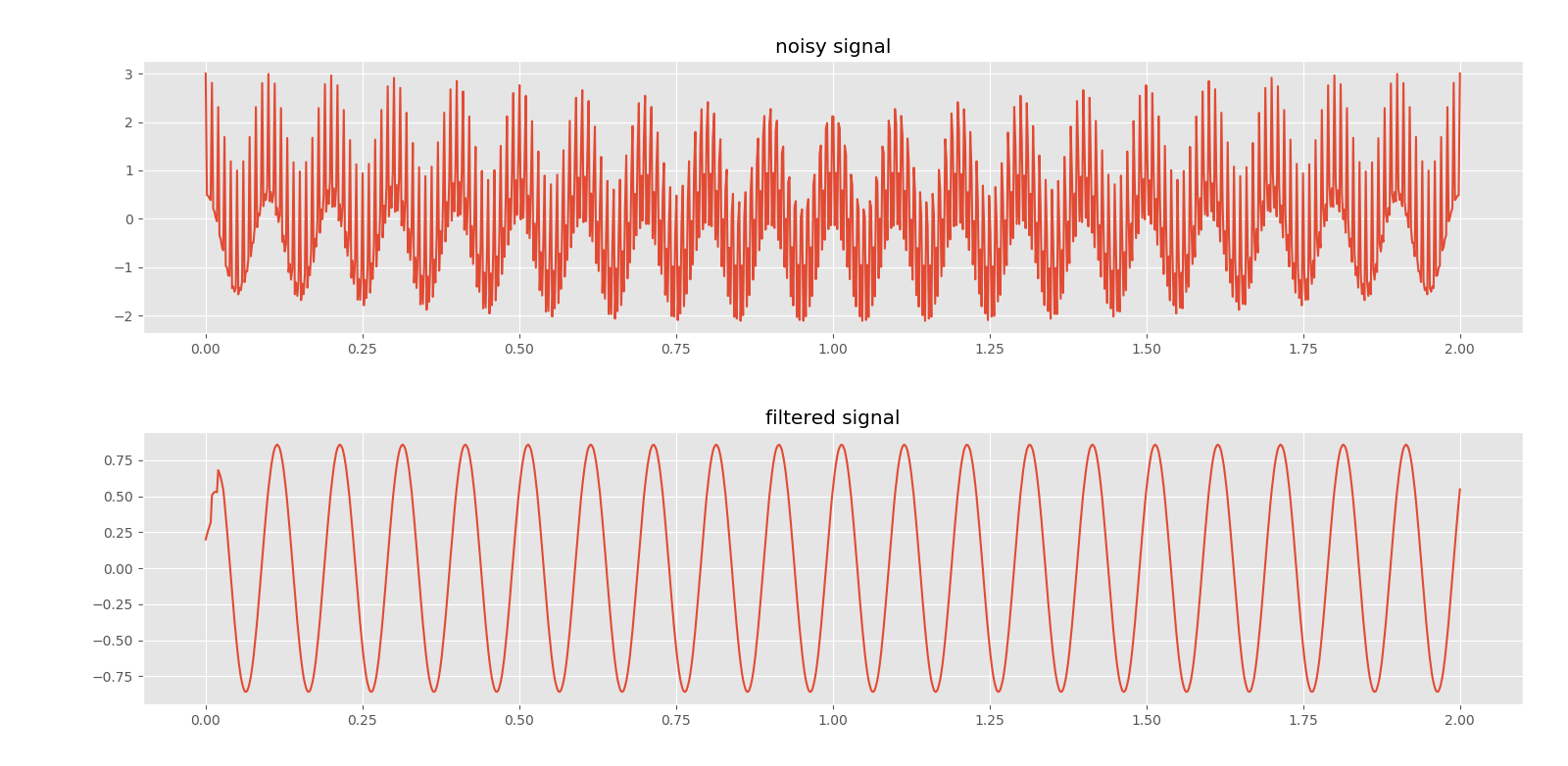
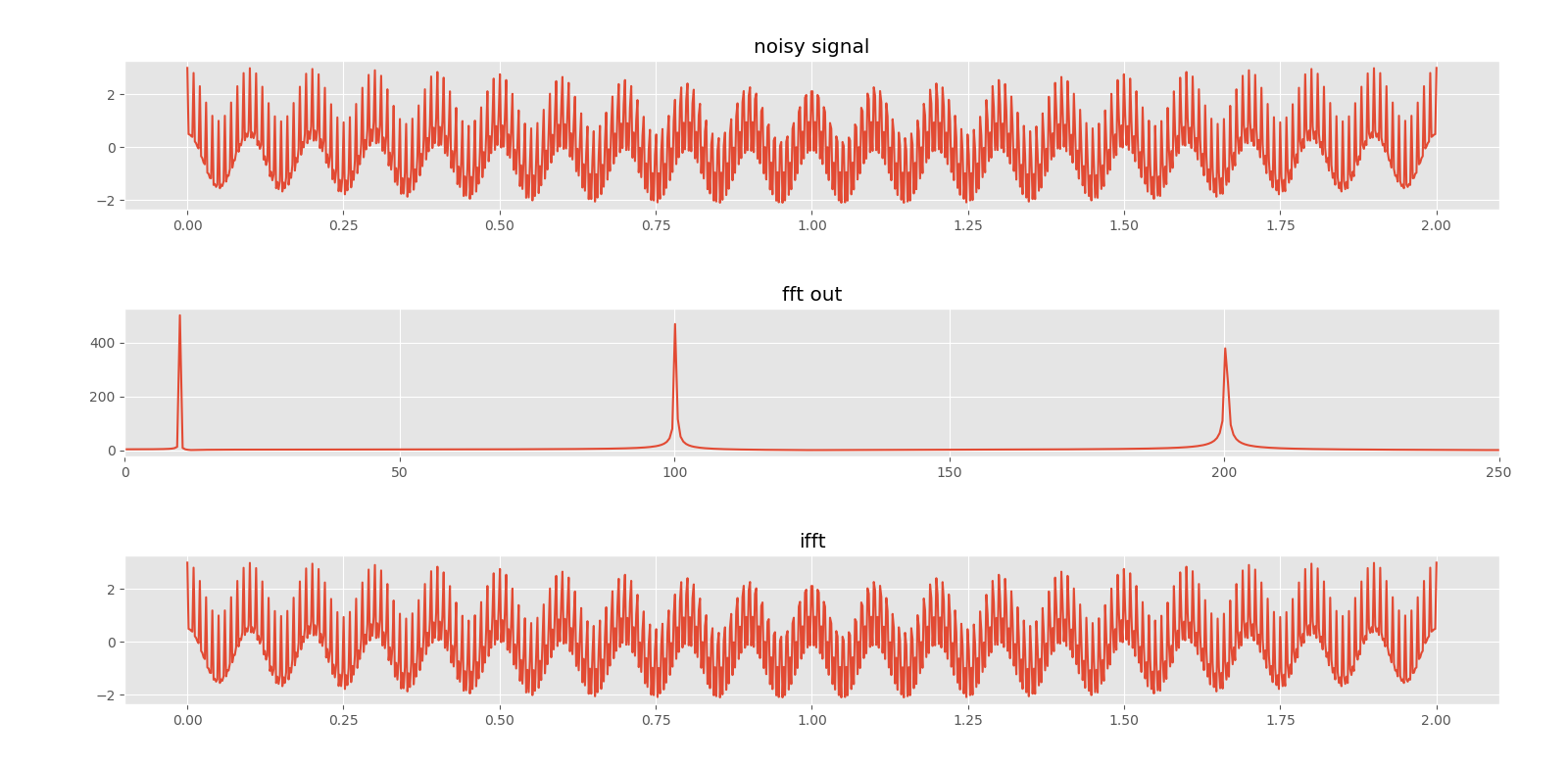
p4\_xarr[3].plot(x\_index,time\_domain\_ifft)

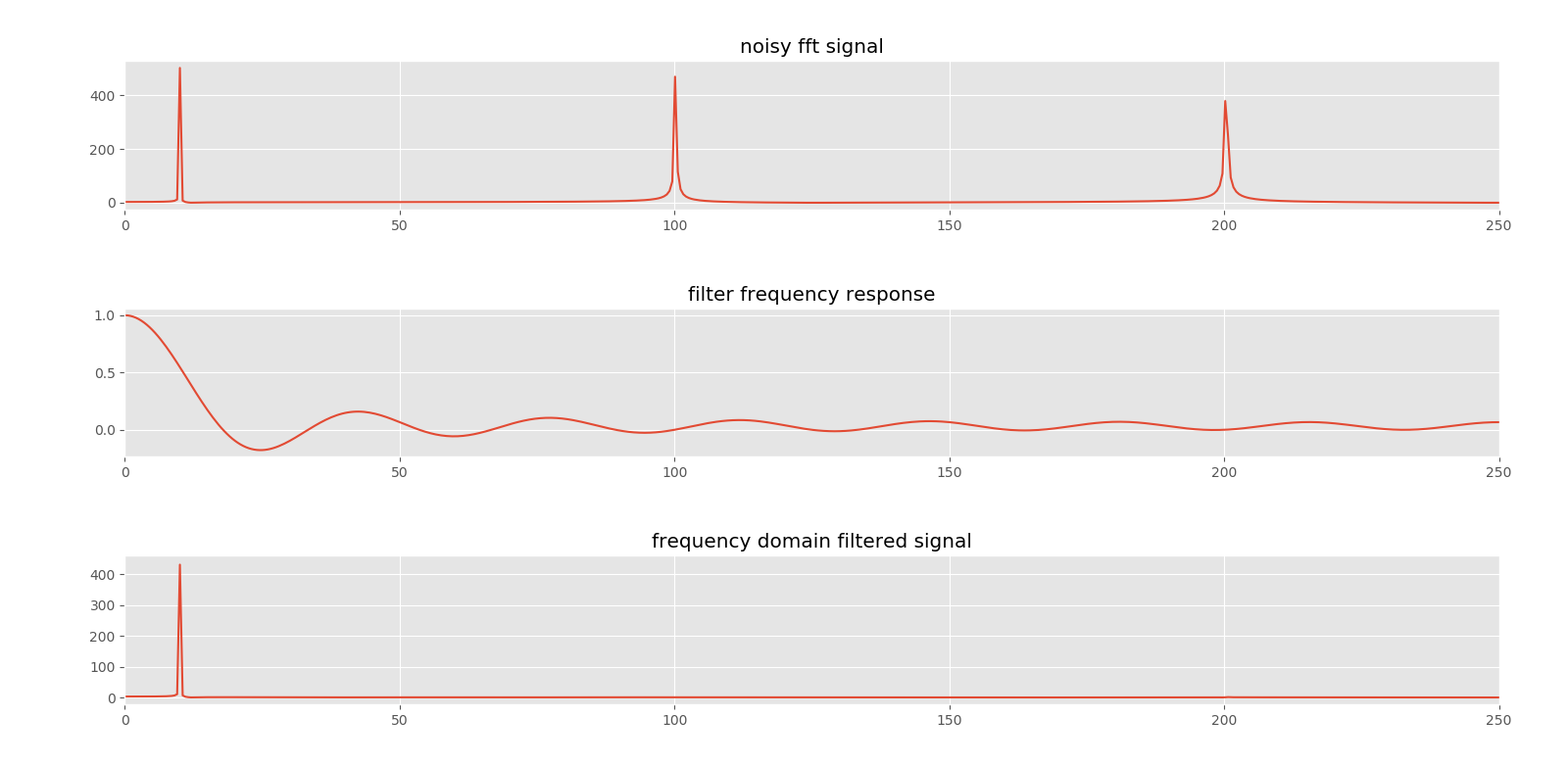
plt.tight\_layout()

plt.show()

plots:

Plot 1

Plot 2

Plot 3

Plot 4