ECE 450 - Homework #14

December 3, 2019

1 ECE 450 - Homework #13

1.0.1 Package Imports

In [1]:
```
import numpy as np
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt
from scipy import signal as sig
from control import margin, tf
import warnings
warnings.filterwarnings('ignore')
```

1.1 Generic function to convolve any number of equations

In [2]:
```
def convolve_all(values):
    temp_conv = values[0]
    if len(values) > 1:
        for next_val in values[1:]:
            temp_conv = np.convolve(temp_conv, next_val)

    return temp_conv
```

1.2 Generic function for generating a Blackman and Half-Blackman Window

In [3]:
```
# The lowpass function in the frequency domain
def blackman(N):
    A, B, C = 0.42, 0.5, 0.08
    return [A - B * np.cos(2 * np.pi * k / N) + C * np.sin(2 * np.pi * k / N) for k in

def half_blackman(N, type="low"):
    A, B, C = 0.42, 0.5, 0.08
    window = []
    for k in range(N):
        if k <= N / 2 and type is "low":
            window.append(1)
        elif k > N / 2 and type is "low":
            window.append(A - B * np.cos(2 * np.pi * k / N) + C * np.sin(2 * np.pi * k
```

```
            elif k <= N / 2 and type is "high":
                window.append(A - B * np.cos(2 * np.pi * k / N) + C * np.sin(2 * np.pi * k
            else:
                window.append(1)

        return window
```

1.2.1 Generic function to plot the responses of a system

```
In [76]: # Color list for multiple lines on each subplot
         colors = ["red", "blue", "green", "gray", "purple", "orange"]
         step_size = 0.005

         # Generic Function to create a plot
         def create_plot(x, y, xLabel=["X-Values"], yLabel=["Y-Values"],
                         title=[("Plot", )], num_rows=1, size=(18, 14), logx=False):
             plt.figure(figsize=size, dpi=300)
             for c, (x_vals, y_vals, x_labels, y_labels, titles) in enumerate(zip(x, y, xLabel
                 for c2, (y_v, t) in enumerate(zip(y_vals, titles)):
                     plt.subplot(num_rows, 1, c + 1)
                     # Add a plot to the subplot, use transparency so they can both be seen
                     plt.plot(x_vals, y_v, label=t, color=colors[c2], alpha=0.70, marker='o')
                     plt.ylabel(y_labels)
                     plt.xlabel(x_labels)
                     plt.grid(True)
                     plt.legend(loc='lower right')
                     if logx:
                         plt.xscale("log")

             plt.show()
```

1.3 Problem 9.4.A

Passband: $f < 3000Hz$, Stopband: $f > 4500Hz$

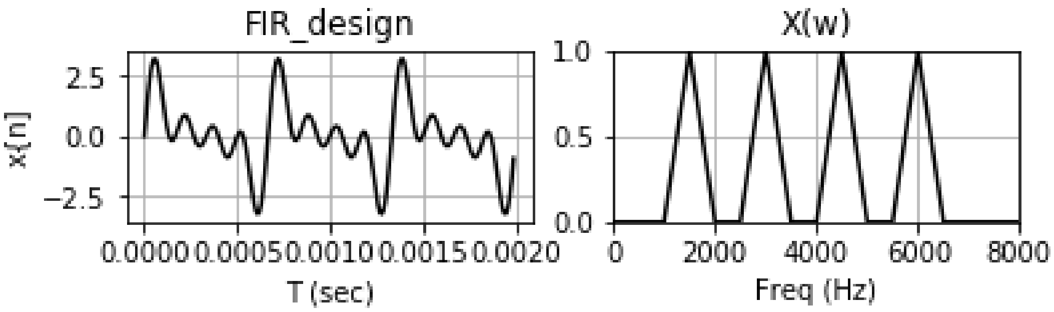
10% attenuation in the passband, 90% in the stopband. Minimize the number of points.

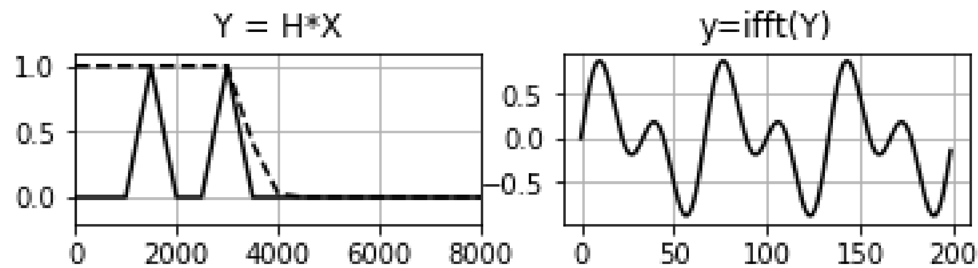
```
In [180]: x_t = lambda t: np.sin(2 * np.pi * 1500 * t) + np.sin(2 * np.pi * 3000 * t) + np.sin

          NN = 200
          N2 = int(NN / 2)
          x = np.zeros(NN)
          y = np.zeros(NN)

          dt = 0.00001
          TT = np.linspace(0, dt * (NN - 1), NN)
          DF = 1/(dt*NN)
          FF = np.linspace(0,DF*(NN-1),NN)
```

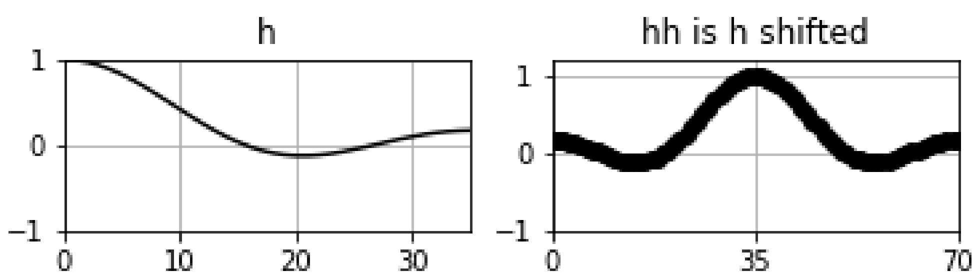
```
x = x_t(TT)

plt.subplot(321)
plt.plot(TT,x,'k')
plt.title('FIR_design')
plt.ylabel('x{n]')
plt.xlabel('T (sec)')
plt.grid()

X = (2/NN)*np.fft.fft(x)

plt.subplot(322)
plt.plot(FF,abs(X),'k')
plt.axis([0, 8000, 0, 1])
plt.grid()
plt.xlabel('Freq (Hz)')
# plt.savefig('x.png',dpi=300)
plt.title('X(w)')
plt.show()

alpha = 0.75
H = [1 if n < 6 else np.exp(-0.5 * (((n - 6) / alpha) ** 2)) for n in range(NN)]
H = [0 if n > N2 else val for n, val in enumerate(H)]

Y = np.multiply(H, X)

plt.subplot(323)
plt.plot(FF,H,'k--')
plt.plot(FF,abs(Y),'k')
plt.axis([0, 8000, -.2, 1.1])
plt.title('Y = H*X')
plt.grid()

y = (NN/2)*np.fft.ifft(Y).real

plt.subplot(324)
plt.plot(y,'k')
plt.title('y=ifft(Y)')
plt.grid()
plt.show()

h = (NN/2)*np.fft.ifft(H).real

hmax = max(h)

for n in range(NN):
    h[n] = h[n]/hmax
```

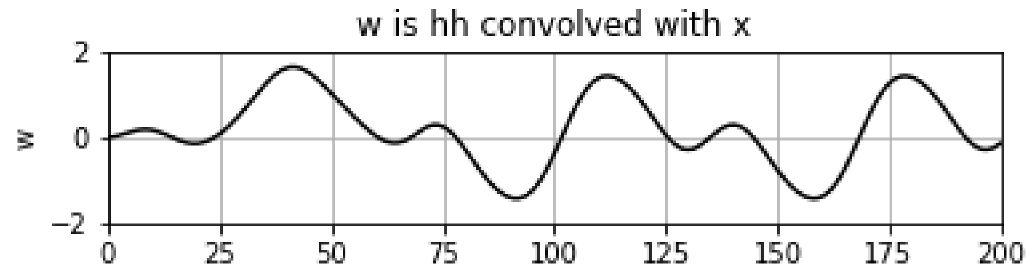
```
MM = 35
plt.subplot(325)
plt.plot(h,'k')
plt.axis([0,MM,-1,1])
plt.grid()
plt.title('h')

hh = np.zeros(2*MM+1)
for n in range(0,MM+1):
    hh[n+MM] = h[n]

for n in range(0,MM):
    hh[n] = h[NN-MM+n]

plt.subplot(326)
plt.plot(hh,'ko')
plt.axis([0 ,2*MM,-1,1.2])
plt.xticks([0,MM,int(2*MM)])
plt.title('hh is h shifted')
plt.grid()
plt.show()

hsum = sum(abs(h))
K = 2.099
w= (K/hsum)*np.convolve(hh,x)
plt.subplot(311)
plt.plot(w,'k')
plt.axis([0, 200, -2, 2])
plt.ylabel('w')
plt.title('w is hh convolved with x')
plt.grid()
plt.show()

z = np.zeros(NN)
for n in range(NN):
    z[n] = w[n+MM]

plt.subplot(311)
plt.plot(z,'k')
plt.ylabel('z')
#plt.plot(x,'m--')
#plt.axis([0,50,-1,1])
plt.title('z is w shifted and truncated')
plt.grid()

Z = (2/NN)*np.fft.fft(z)
plt.subplot(313)
plt.plot(FF,abs(Z),'k')
```

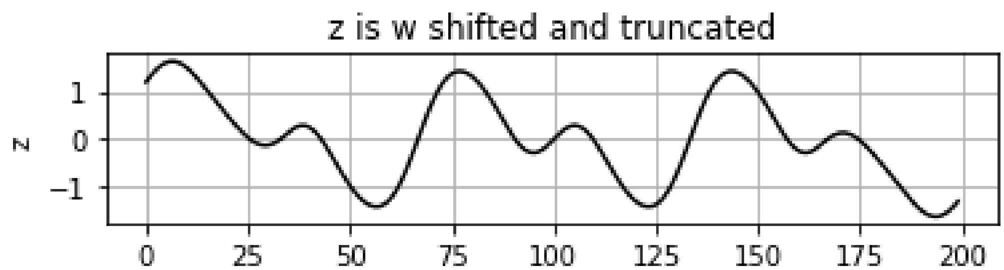
```
plt.ylabel('|Z(w)|')
plt.axis([0, 8000, 0, 1])
plt.title('Z = fft(z)')
plt.grid()
plt.show()
```

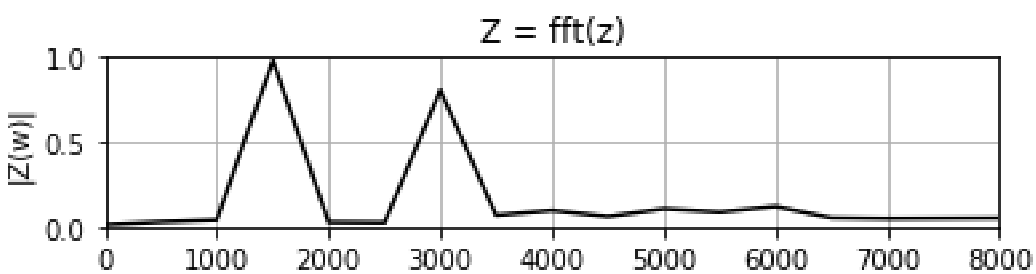












As evident in the FFT(Z), the undesireable frequencies are attenuated to an acceptable amount, and this was accomplished with 35 points.

1.4 Problem 9.4.B

Passband: $3000 < f < 4500Hz$, and Stopband: $f < 1500Hz$, $f > 6000Hz$

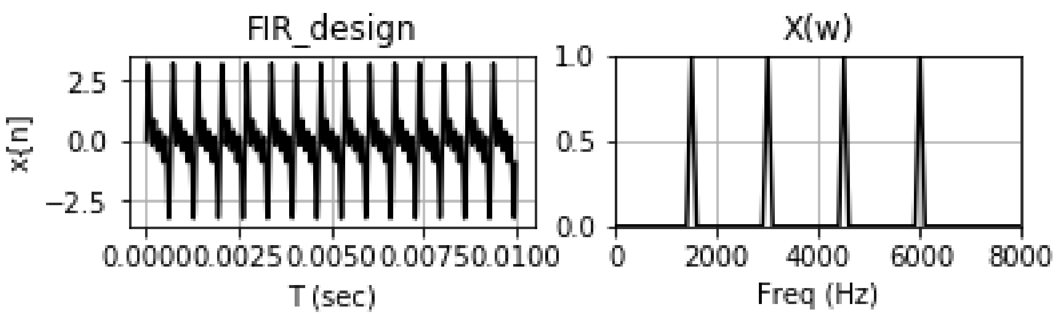
No more than 10% attenuation in the passband, 90% attenuation in the stopband.

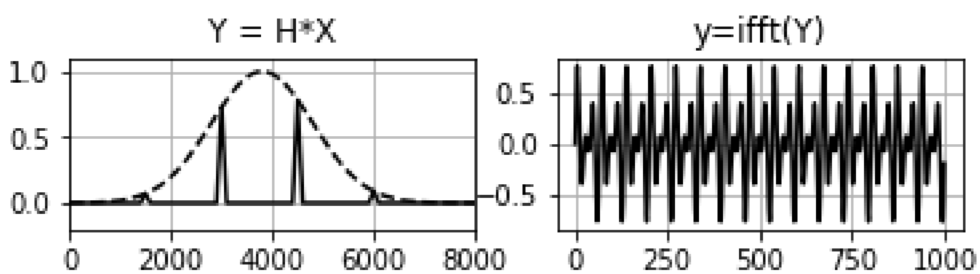
```
In [181]: NN = 1000
          N2 = int(NN/2)
          x = np.zeros(NN)
          y = np.zeros(NN)

          dt = .00001
```

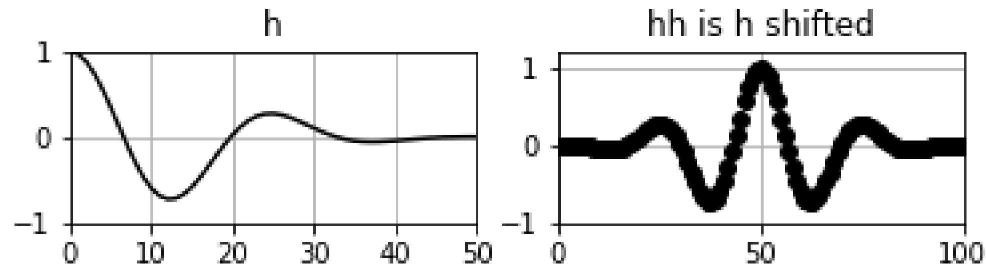
```
TT = np.linspace(0,dt*(NN-1),NN)
DF = 1/(dt*NN)
FF = np.linspace(0,DF*(NN-1),NN)

x = x_t(TT)

plt.subplot(321)
plt.plot(TT,x,'k')
plt.title('FIR_design')
plt.ylabel('x{n]')
plt.xlabel('T (sec)')
plt.grid()

X = (2/NN)*np.fft.fft(x)

plt.subplot(322)
plt.plot(FF,abs(X),'k')
plt.axis([0,8000,0,1 ])
plt.grid()
plt.xlabel('Freq (Hz)')
plt.title('X(w)')
plt.show()

H = np.zeros(NN)

nc = int(3875/DF) #100
sigma = 10 #10
for n in range(0,N2):
    H[n] = np.exp(-0.5*( ((n-nc)/sigma)**2))

Y = X
for n in range(0,NN):
    Y[n] = H[n]*X[n]

plt.subplot(323)
plt.plot(FF,H,'k--')
plt.plot(FF,abs(Y),'k')
plt.axis([0,8000,-.2,1.1])
plt.title('Y = H*X')
plt.grid()

y = (NN/2)*np.fft.ifft(Y).real

plt.subplot(324)
plt.plot(y,'k')
plt.title('y=ifft(Y)')
plt.grid()
plt.show()
```

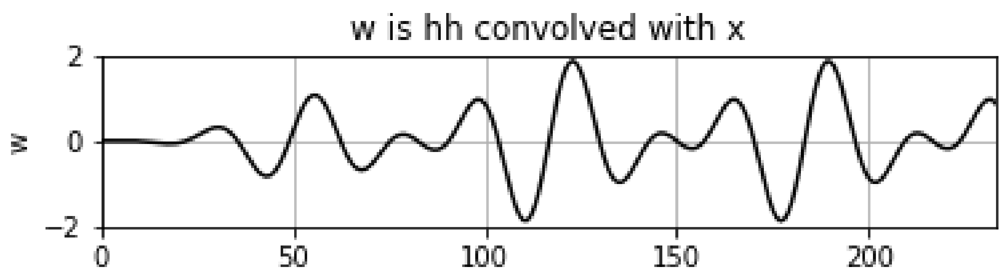
```
h = (NN/2)*np.fft.ifft(H).real

hmax = max(h)

for n in range(NN):
    h[n] = h[n]/hmax

MM = 50 #13
plt.subplot(325)
plt.plot(h,'k')
plt.axis([0,MM,-1,1])
plt.grid()
plt.title('h')

#  Shift h  to hh

""" The shifted filter is 2*MM + 1 points long. """
hh = np.zeros(2*MM+1)
for n in range(0,MM+1):
    hh[n+MM] = h[n]

for n in range(0,MM):
    hh[n] = h[NN-MM+n]

plt.subplot(326)
plt.plot(hh,'ko')
plt.axis([0 ,2*MM,-1,1.2])
plt.xticks([0,MM,int(2*MM)])
plt.title('hh is h shifted')
plt.grid()
plt.show()

# convolve hh with x

hsum = sum(abs(h))
K = 1.55     # Band pass, 1.35
w= (K/hsum)*np.convolve(hh,x)
plt.subplot(311)
plt.plot(w,'k')
plt.axis([0,233,-2,2])
plt.ylabel('w')
plt.title('w is hh convolved with x')
plt.grid()
plt.show()

# Trim the ends and take the FT.
```

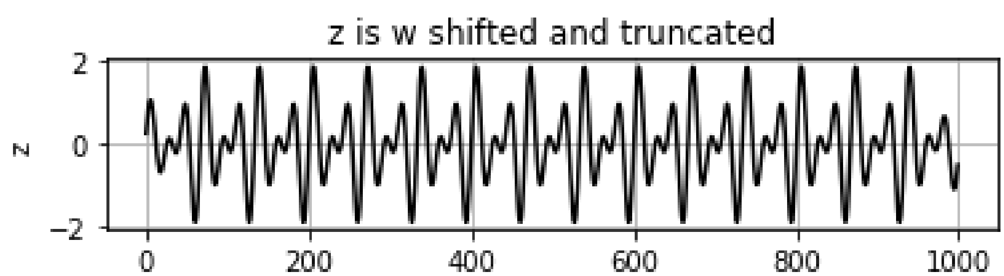
```
z = np.zeros(NN)
for n in range(NN):
    z[n] = w[n+MM]

plt.subplot(311)
plt.plot(z,'k')
plt.ylabel('z')
plt.title('z is w shifted and truncated')
plt.grid()

Z = (2/NN)*np.fft.fft(z)
plt.subplot(313)
plt.plot(FF,abs(Z),'k')
plt.ylabel('|Z(w)|')
plt.axis([0,8000,0,1.])
plt.title('Z = fft(z)')
plt.grid()
plt.show()
```

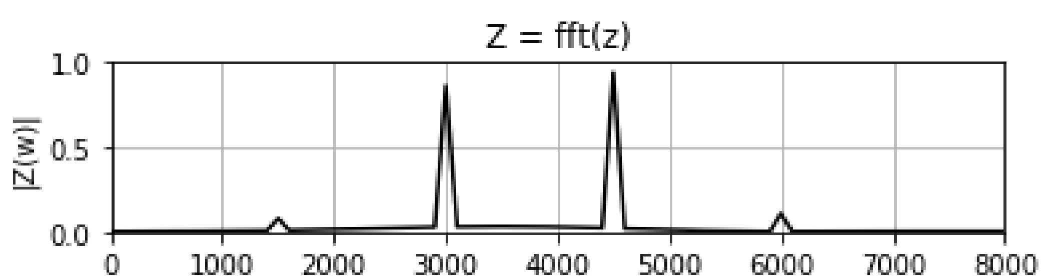












As evident in the FFT(Z), the passband frequencies are passed with an acceptable amount of attenuation while rejecting the stopband signals. This was accomplished with 50 samples, which is relatively high as the passband is quite wide, and the rejection criteria for the stopband is quite restrictive. Sampling less (around 30) attenuates the passband too much.