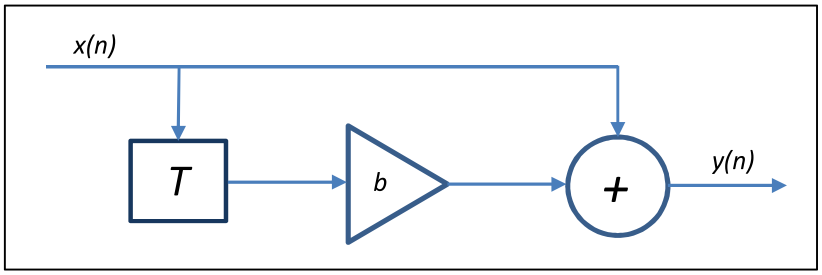
Poles & Zeros, Cut-Off Frequency (FIR 1st Order)

Here we shall investigate the determination of the cut-off frequency of both low-pass and high-pass FIR 1st order filters. We shall also look at how to determine the filter tap coefficient for a given cut-off frequency specification. A brief examination of the pole/zero diagram will also be done.

1st order filter (FIR)

Consider the block diagram of the 1st order finite impulse response filter shown below:



$$y(n) = x(n) + bx(n-1) \Rightarrow Y(z) = X(z) + bX(z)z^{-1} \Rightarrow H(z) = \frac{Y(z)}{X(z)} = 1 + bz^{-1}$$

In general, when we wish to obtain the frequency response of any digital filter, we make the simple substitution:

$$z = e^{j\theta}, \theta = \frac{2\pi f_a}{f_s} = 2\pi f_a T = \omega T$$

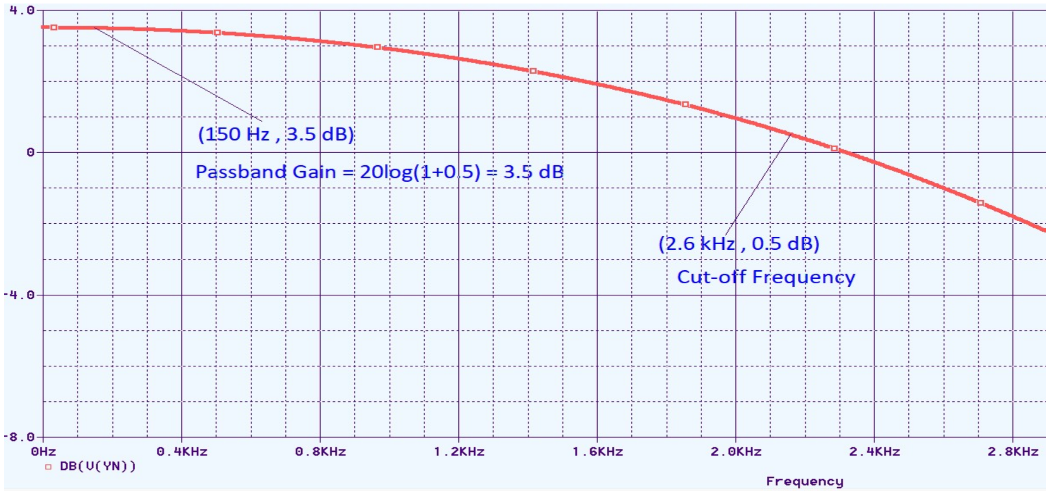
$$f_a = analogue\ frequency, f_s = sampling\ frequency$$

$$s = j\omega \quad and \quad z = e^{j\theta} = e^{j\omega T}$$

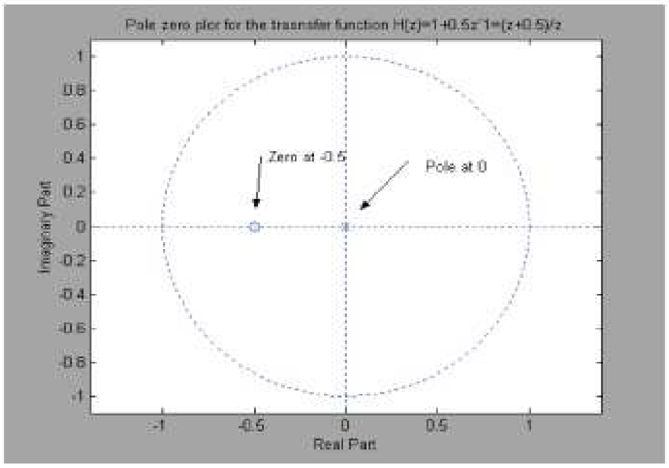
$$H(\theta) = 1 + be^{-j\theta} \quad H(0) = 1 + be^{0} = 1 + b$$

$$e.g.\ if\ b = 0.5, H(0) = 1.5 \quad H_{dB}(0) = 20\log(1.5) = 3.5\ dB$$

Note that to find the D.C. gain of a filter you always substitute θ=0. This is because the D.C. case corresponds to ω=0 rads/sec. Note also that θ=0 corresponds to z=1.



$$H(z)=\frac{Y(z)}{X(x)}=1+0.5z^{-1}=\frac{z+0.5}{z}$$



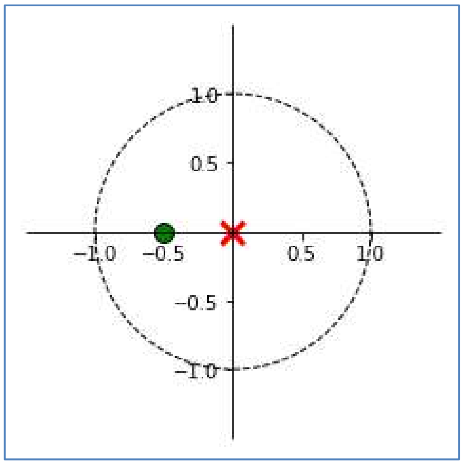
The above diagram was created in Matlab as follows:

```
>>bb=[1 0.5]
>>aa=1
>>zplane (bb,aa)
```

Alternatively we can use Python:

```
import numpy as np
# If the code is in a file called plot_zplane.py
from plot_zplane import zplane
bb = np.array([1, 0.5])
aa = np.array([1,0])
zplane(bb,aa)
```

The program plot_zplane.py is attached to the end of this document. It should be saved into the default directory where Python saves programs.



You should type these commands into Matlab or Python yourself to verify that the correct pole/zero diagram is obtained. It shows poles (x's) and zeros (o's). A system will be BIBO stable if the poles are inside the unit circle and unstable if they are outside. If the poles are on the circle it is *marginally* stable. This is the case for oscillators.

Cut-off frequency (LPF FIR)

$$H(z)=\frac{Y(z)}{X(z)}=1+bz^{-1}\colon H(\theta)=1+be^{-j\theta}$$

$$But\ e^{j\theta}=\cos(\theta)+j\sin(\theta)\ and\ e^{-j\theta}=\cos(\theta)-j\sin(\theta)$$

This is the euler identity

$$\therefore\ H(\theta)=1+b\{\cos(\theta)-j\sin(\theta)\}$$

$$|H(\theta)|^2=\{1+b\cos(\theta)\}^2+\{b\sin(\theta)\}^2=1+2b\cos(\theta)+b^2\{\cos^2(\theta)+\sin^2(\theta)\}$$

$$|H(\theta)|^2=1+2b\cos(\theta)+b^2$$

Sample Problem:

A *Finite Impulse Response* filter has a difference equation:

$$y(n)=x(n)+bx(n-1)$$

Show that the magnitude squared frequency response of the filter is defined as:

$$|H(\theta)|^2=1+2b\cos(\theta)+b^2$$

If b=0.6 draw up a table of the *magnitude squared* frequency response for values of ϑ between 0 and π in steps of π/8 and use this table to draw an approximate graph of the filter's frequency response. From the graph state the values of:

(i) $|H(0)|^2$

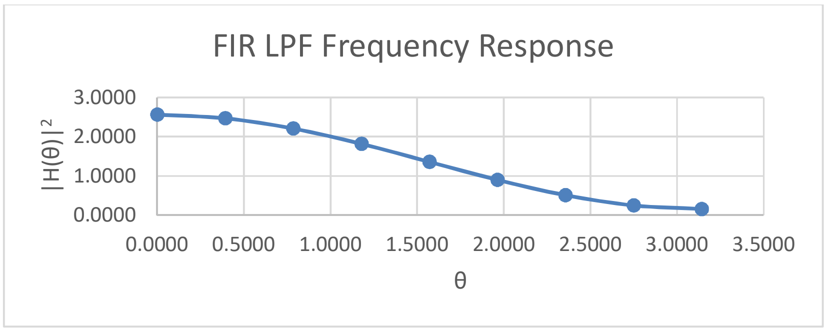
(ii) $|H(\pi)|^2$

If ϑ_c represents the digital cut-off frequency can you estimate ϑ_c from the graph? Show by derivation that ϑ_c is defined as:

$$\theta_c=\cos^{-1}\left\{\frac{2b-1-b^2}{4b}\right\}$$

What is the cut-off frequency of this filter if the sampling frequency is 16000 Hz?

θ	0.0000	0.3927	0.7854	1.1781	1.5708	1.9635	2.3562	2.7489	3.1416
$\lvert H(\theta)\rvert^2$	2.5600	2.4687	2.2085	1.8192	1.3600	0.9008	0.5115	0.2513	0.1600



From the graph we can see that:

(i) $\left|H(0)\right|^2$ =2.56

(ii) $\left|H(\pi)\right|^2$ =0.16

The cut-off frequency of the filter is at the half-power point. Thus:

$$\left|H(\theta_c)\right|^2 = \frac{\left|H(0)\right|^2}{2} = 1.125$$

From the table this occurs between 4π/8 and 5π/8, so a crude approximation for the cut-off point is half way between these values or 4.5π/8.

$$f_C = \frac{\theta_c f_S}{2\pi} = \frac{(4.5\pi/8)\times 16000}{2\pi} = 4500\ Hz$$

$$\frac{\left|H(0)\right|^2}{2} = \left|H(\theta_c)\right|^2 \Rightarrow \frac{1+2b+b^2}{2} = 1+2b\cos(\theta_c)+b^2$$

$$1+2b+b^2 = 2+4b\cos(\theta_c)+2b^2$$

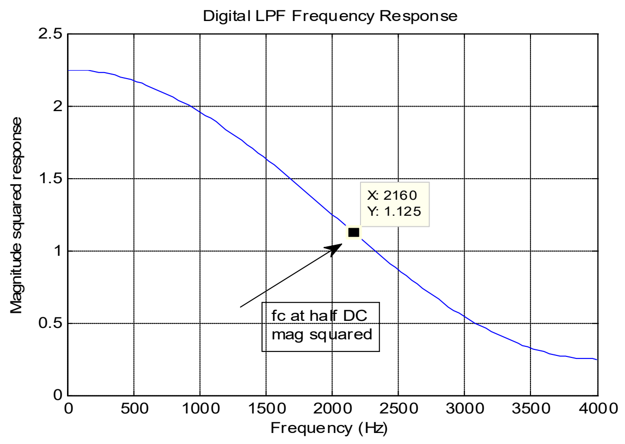
$$4b\cos(\theta_c) = 2b-1-b^2$$

$$\theta_c = \cos^{-1}\left\{\frac{2b-1-b^2}{4b}\right\} = \cos^{-1}\left\{\frac{2(0.6)-1-(0.6)^2}{4(0.6)}\right\} = \cos^{-1}\left\{\frac{-0.16}{2.4}\right\} = 1.6375$$

$$f_C = \frac{\theta_c f_S}{2\pi} = \frac{1.6375\times 16000}{2\pi} = 4170\ Hz$$

Now we use Matlab to sketch this frequency response.

```
b=0.5
fs=8000;
theta=0:pi/100:pi
freq=theta*fs/(2*pi);
magH_sqr=(1+2*b*cos(theta)+b*b);
plot(freq,magH_sqr);
title 'Digital LPF Frequency Response'
xlabel 'Frequency (Hz)'
ylabel 'Magnitude squared response'
grid
thetac=acos((2*b-1-b*b)/(4*b));
fc=(fs/(2*pi))*acos((2*b-1-b*b)/(4*b))
```



The above plot assumes a sampling frequency of 8 kHz.

$$|H(\theta)|^2 = 1 + 2b\cos(\theta) + b^2$$
$$|H(0)|^2 = 1 + 2b + b^2 = 1 + 2(0.5) + (0.5)^2 = 2.25$$
$$|H(\pi)|^2 = 1 - 2b + b^2 = 1 - 2(0.5) + (0.5)^2 = 0.25$$

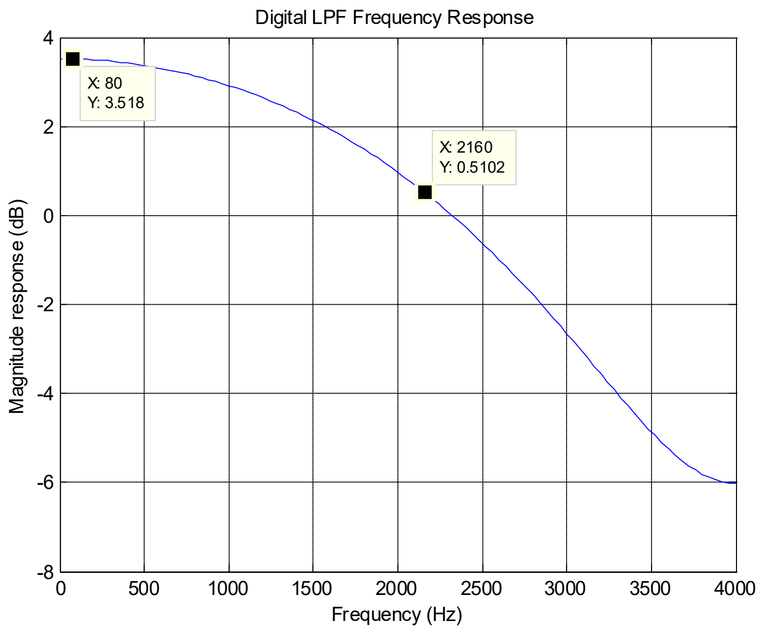
You can see from the plot above that the filter is a 1st order FIR LPF. You could have seen this also by examining the values of *H(0)* and *H(π)* and observing that the magnitude is at a maximum for *H(0).* Now the cut-off frequency is that point when the gain is down by 3dB or the gain squared is halved.

$$\frac{|H(0)|^2}{2} = |H(\theta_c)|^2 \Rightarrow \frac{1+2b+b^2}{2} = 1 + 2b\cos(\theta_c) + b^2$$
$$1 + 2b + b^2 = 2 + 4b\cos(\theta_c) + 2b^2$$
$$4b\cos(\theta_c) = 2b - 1 - b^2$$
$$\theta_c = \cos^{-1}\left\{\frac{2b-1-b^2}{4b}\right\} = \cos^{-1}\left\{\frac{2(0.5)-1-(0.5)^2}{4(0.5)}\right\} = \cos^{-1}\left\{\frac{-0.25}{2}\right\} = 1.69$$
$$f_C = \frac{\theta_c f_S}{2\pi} = \frac{1.69 \times 8000}{2\pi} = 2160\ Hz$$
$$|H(\theta_C)| = \frac{|H(0)|^2}{2} = \frac{2.25}{2} = 1.125$$

	Cut-off frequency of FIR LPF			
	theta=ARCCOS((2*b-1-b^2)/(4*b))			
fs	b	theta (rads	fc	
8000.0000	0.5000	1.6961	2159.5724	

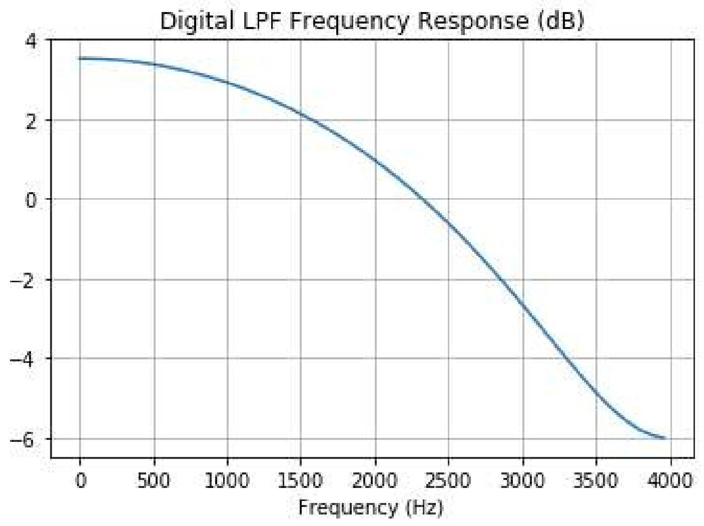
We can also plot the gain in dB.

```
b=0.5
fs=8000;
theta=0:pi/100:pi
freq=theta*fs/(2*pi);
magH_sqr=(1+2*b*cos(theta)+b*b);
magH_sqr_dB=10*log10(magH_sqr);
plot(freq,magH_sqr_dB);
title 'Digital LPF Frequency Response'
xlabel 'Frequency (Hz)'
ylabel 'Magnitude response (dB)'
grid
thetac=acos((2*b-1-b*b)/(4*b));
fc=(fs/(2*pi))*acos((2*b-1-b*b)/(4*b))
```



We can also plot this in Python.

```
import matplotlib.pyplot as plt
import numpy as np
b=0.5
fs=8000
theta=np.arange(0,np.pi,np.pi/100)
f=theta*fs/(2*np.pi);
magH_sqr=(1+2*b*np.cos(theta)+b*b)
HdB=10*np.log10(magH_sqr)
plt.plot(f,HdB);
plt.title ('Digital LPF Frequency Response (dB)')
plt.xlabel ('Frequency (Hz)')
plt.grid()
plt.show()
thetac=np.arccos((2*b-1-b*b)/(4*b));
fc=(fs/(2*np.pi))*np.arccos((2*b-1-b*b)/(4*b))
for i in range (len(f)):
   print(f'{f[i]:6.2f} ==> {HdB[i]:6.2f}')
```



```
 0.00 ==>  3.52
 40.00 ==>  3.52
 80.00 ==>  3.52
120.00 ==>  3.51
160.00 ==>  3.51
2000.00 ==>  0.97
2040.00 ==>  0.86
2080.00 ==>  0.75
2120.00 ==>  0.63
2160.00 ==>  0.51
2200.00 ==>  0.39
2240.00 ==>  0.26
2280.00 ==>  0.14
2320.00 ==>  0.01
```

Normalising the DC gain

$$\theta = 0 \Rightarrow z = e^{j0} = 1$$

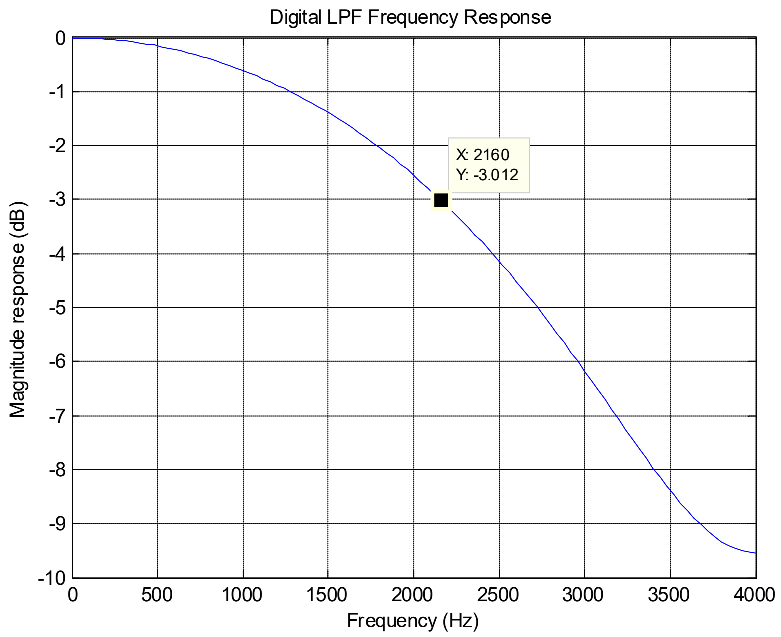
$$H(z) = 1 + bz^{-1}$$

$$H(z=1) = 1 + be^{-j0} = 1 + b$$

$$Let\ H_N(z) = \frac{H(z)}{H(z=1)} = \frac{1+bz^{-1}}{1+b}$$

```
b=0.5
fs=8000;
theta=0:pi/100:pi
freq=theta*fs/(2*pi);
magH_sqr=(1+2*b*cos(theta)+b*b);
magH_sqr=(1+2*b*cos(theta)+b*b)/((1+b)^2);
magH_sqr_dB=10*log10(magH_sqr);
plot(freq,magH_sqr_dB);
title 'Digital LPF Frequency Response'
xlabel 'Frequency (Hz)'
ylabel 'Magnitude response (dB)'
```

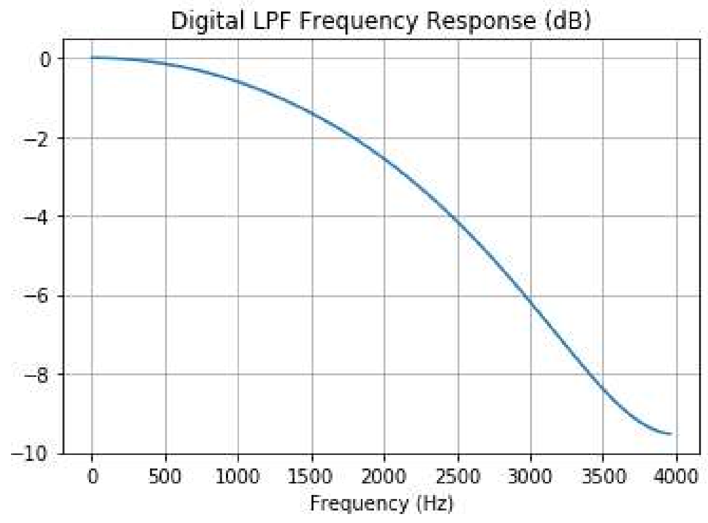
```
grid
thetac=acos((2*b-1-b*b)/(4*b));
fc=(fs/(2*pi))*acos((2*b-1-b*b)/(4*b))
```



Note that the dc gain is now 0 dB corresponding to a unity dc gain.

Python can also be used here.

```
import matplotlib.pyplot as plt
import numpy as np
b=0.5
fs=8000
theta=np.arange(0,np.pi,np.pi/100)
f=theta*fs/(2*np.pi);
magH_sqr=(1+2*b*np.cos(theta)+b*b)/((1+b)**2);
HdB=10*np.log10(magH_sqr)
plt.plot(f,HdB);
plt.title ('Digital LPF Frequency Response (dB)')
plt.xlabel ('Frequency (Hz)')
plt.grid()
plt.show()
thetac=np.arccos((2*b-1-b*b)/(4*b));
fc=(fs/(2*np.pi))*np.arccos((2*b-1-b*b)/(4*b))
for i in range (len(f)):
   print(f'{f[i]:6.2f} ==> {HdB[i]:6.2f}')
```



```
2080.00 ==>  -2.78
2120.00 ==>  -2.89
2160.00 ==>  -3.01
2200.00 ==>  -3.13
2240.00 ==>  -3.26
2280.00 ==>  -3.39
2320.00 ==>  -3.52
```

Alternatively, we can solve for b in terms of the cut-off frequency:

$$1+2b+b^2 = 2+4b\cos(\theta_c)+2b^2$$

$$b^2+4b\cos(\theta_c)-2b+1=0$$

$$b^2+2b\left(2\cos(\theta_c)-1\right)+1=0$$

$$b=\frac{-2\left(2\cos(\theta_c)-1\right)\pm\sqrt{4\left(2\cos(\theta_c)-1\right)^2-4}}{2}$$

$$b=\left(1-2\cos(\theta_c)\right)\pm\sqrt{\left(2\cos(\theta_c)-1\right)^2-1}$$

$$\theta_c=1.69\Rightarrow b=\left(1-2\cos(1.696)\right)\pm\sqrt{\left(2\times\cos(1.696)-1\right)^2-1}$$

$$=1.25\pm0.75=2,\,0.5$$

$$\textit{Since solution of quadratic } Ax^2+Bx+C \textit{ is}:$$

$$\frac{-B\pm\sqrt{B^2-4AC}}{2A}$$

Note that this particular equation is very sensitive to round off errors. You need to put in the angle with around 4 decimal places. The spreadsheet is very useful in this case.

	Cut-off frequency of FIR LPF				
	theta=ARCCOS((2*b-1-b^2)/(4*b))				
fs	b	theta (rads	fc		
8000.0000	0.5000	1.6961	2159.5724		
1-2cos(theta)	sqrt((2cos(theta)-1)^2-1)			b1	b2
1.25	0.75			2	0.5

Note that there are 2 solutions to the problem in this case. Putting b=0.5 or 2 will give exactly the same frequency response.

Problem 1

The output *y(n)* of a digital *FIR* low-pass filter is defined by: $y(n) = x(n) + bx(n-1)$ where *x(n)* is the sampled input signal. Use the z-transformation to derive an expression for the filter transfer function. From the transfer function of the filter, show that the cut-off frequency of the filter (ϑ_c) is defined by:

$$\theta_c = \cos^{-1}\left\{\frac{2b-1-b^2}{4b}\right\}$$

Determine θc if *b*=0.25. What is the cut-off frequency of this digital filter if the sampling frequency is 16 kHz? Draw an approximate sketch of the pole zero diagram of the filter.

You can see from the spreadsheet that the cut-off frequency is 5.5 kHz. Or using a calculator:

$$\theta_c = \cos^{-1}\left\{\frac{2b-1-b^2}{4b}\right\} = \cos^{-1}\left\{\frac{2(0.25)-1-(0.25)^2}{4(0.25)}\right\} = 2.17\ rads$$

$$\theta_c = \frac{2\pi f_c}{f_s} \Rightarrow f_c = \frac{\theta_c f_s}{2\pi} = \frac{2.17\times 16000}{2\pi} = 5.5\ kHz$$

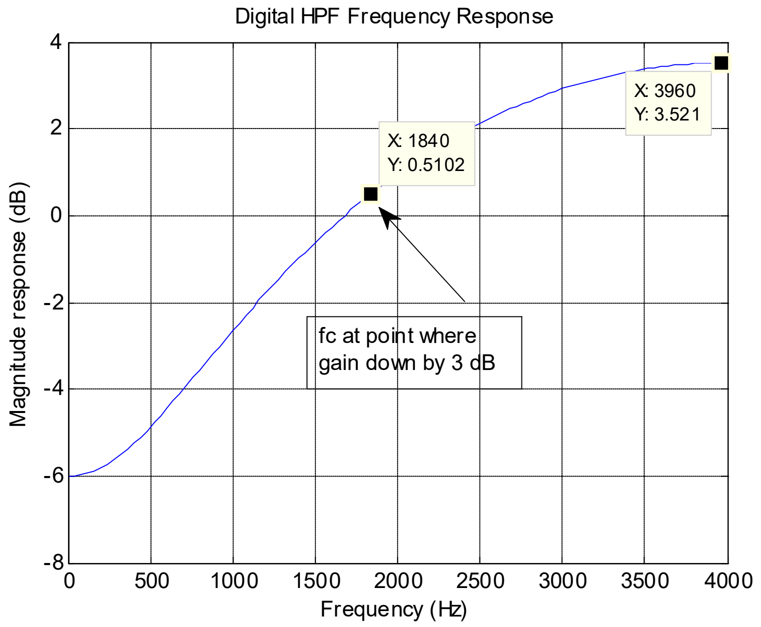
Problem 2

Repeat problem 1 for a=0.4 and the sampling frequency is 32 kHz. Use Matlab to plot the frequency response of this filter. Measure the 3 dB frequency in Matlab. How does this compare to theory?

Now we use Matlab to sketch this frequency response again except we make b negative.

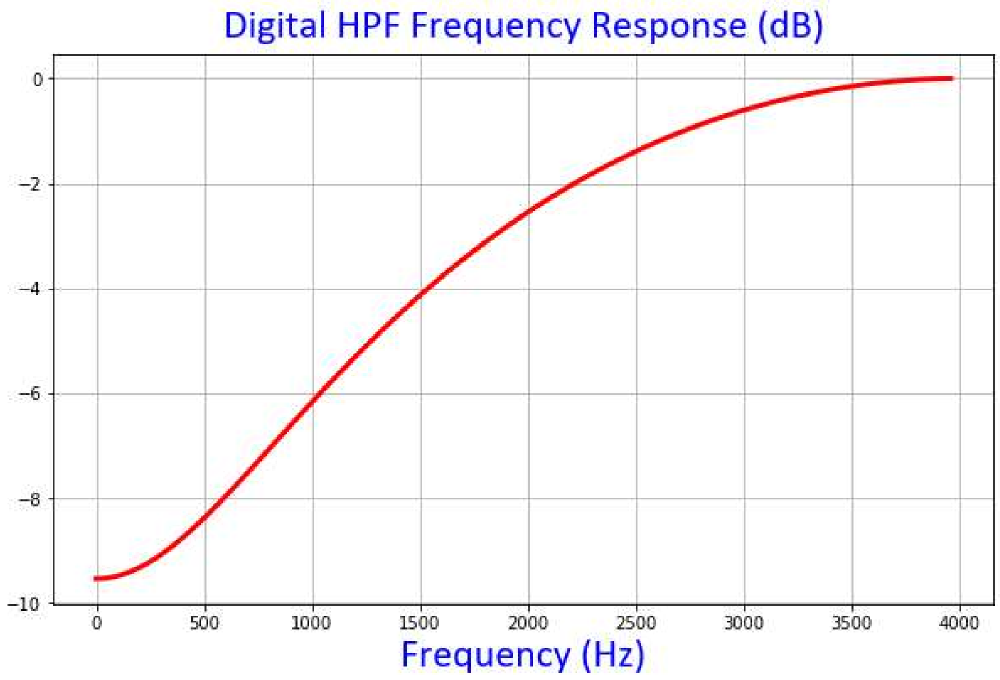
```
b=-0.5
fs=8000;
theta=0:pi/100:pi
freq=theta*fs/(2*pi);
magH_sqr=(1+2*b*cos(theta)+b*b);
magH_sqr_dB=10*log10(magH_sqr);
plot(freq,magH_sqr_dB);
title 'Digital HPF Frequency Response'
xlabel 'Frequency (Hz)'
```

```
ylabel 'Magnitude response (dB)'
grid
thetac=acos((-2*b-1-b*b)/(4*b));
fc=(fs/(2*pi))*thetac
```



```
import matplotlib.pyplot as plt
import numpy as np
b=-0.5
fs=8000
theta=np.arange(0,np.pi,np.pi/100)
f=theta*fs/(2*np.pi);
magH_sqr=(1+2*b*np.cos(theta)+b*b)/((1-b)**2)
HdB=10*np.log10(magH_sqr)

plt.figure(figsize=(10,6))
title_font = {'fontname':'Calibri', 'size':'24', 'color':'blue', 'weight':'normal',
          'verticalalignment':'bottom'} # Bottom vertical alignment for more space
axis_font = {'fontname':'Calibri', 'size':'24', 'color':'blue'}
plt.plot(f,HdB, color="red",linewidth=3.0 )
ax = plt.gca()
#ax.set_facecolor('#CCFFFF')
#https://www.rapidtables.com/web/color/RGB_Color.html
plt.title ('Digital HPF Frequency Response (dB)', **title_font)
plt.xlabel ('Frequency (Hz)',**axis_font)
plt.grid()
plt.show()
thetac=np.arccos((-2*b-1-b*b)/(4*b));
fc=(fs/(2*np.pi))*thetac
if(0):
   for i in range (len(f)):
      print('{:07.2f} ==> {:04.2f}'.format(f[i],HdB[i]))
```



```
fc=1840.43

1680.00 ==> -3.52
1720.00 ==> -3.39
1760.00 ==> -3.26
1800.00 ==> -3.13
1840.00 ==> -3.01
1880.00 ==> -2.89
1920.00 ==> -2.78
1960.00 ==> -2.66
```

The analysis for determining the cut-off frequency is slightly different for the high-pass filter. In this case we need to find the gain at π rather than at 0.

$$\frac{\left|H(\pi)\right|^2}{2} = \left|H(\theta_c)\right|^2 \Rightarrow \frac{1-2b+b^2}{2} = 1+2b\cos(\theta_c)+b^2$$

$$1-2b+b^2 = 2+4b\cos(\theta_c)+2b^2$$

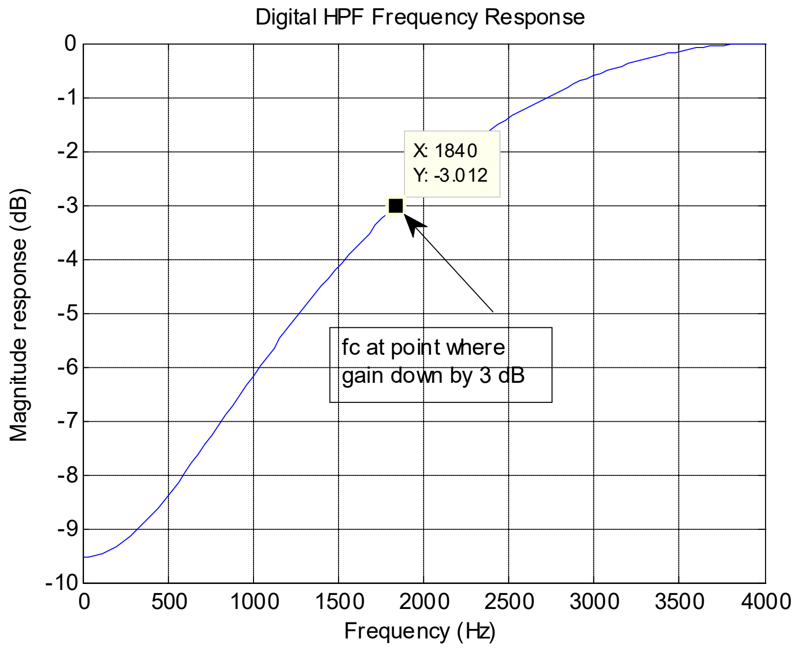
$$4b\cos(\theta_c) = -2b-1-b^2$$

$$\theta_c = \cos^{-1}\left\{-\frac{2b+1+b^2}{4b}\right\} = \cos^{-1}\left\{-\frac{2(-0.5)+1+(0.5)^2}{4(-0.5)}\right\} = \cos^{-1}\left\{\frac{0.25}{2}\right\} = 1.4455$$

$$f_C = \frac{\theta_c f_S}{2\pi} = \frac{1.4455\times 8000}{2\pi} = 1840\ Hz$$

Normalising the High-Frequency gain

$$\theta = \pi \Rightarrow z = e^{j\pi} = -1$$
$$H(z) = 1 + bz^{-1}$$
$$H(z=-1) = 1 - b$$
$$Let\ H_N(z) = \frac{H(z)}{H(z=-1)} = \frac{1+bz^{-1}}{1-b}$$



Alternatively we can solve for b in terms of the cut-off frequency:

$$1 - 2b + b^2 = 2 + 4b\cos(\theta_c) + 2b^2$$
$$b^2 + 4b\cos(\theta_c) + 2b + 1 = 0$$
$$b^2 + 2b(2\cos(\theta_c) + 1) + 1 = 0$$
$$b = \frac{-2(2\cos(\theta_c)+1) \pm \sqrt{4(2\cos(\theta_c)+1)^2 - 4}}{2}$$
$$b = -(2\cos(\theta_c)+1) \pm \sqrt{(2\cos(\theta_c)+1)^2 - 1}$$
$$For\ \theta_c = 1.4455$$
$$b = -(2\cos(1.4455)+1) \pm \sqrt{(2\cos(1.4455)+1)^2 - 1}$$
$$= -1.25 \pm 0.75 = -0.5, -2$$
$$Since\ solution\ of\ quadratic\ Ax^2 + Bx + C\ is:$$
$$\frac{-B \pm \sqrt{B^2 - 4AC}}{2A}$$

	Cut-off frequency of FIR HPF				
	theta=ARCCOS((-2*b-1-b^2)/(4*b))				
fs	b	theta (rads	fc		
8000.0000	-0.5000	1.4455	1840.4276		
(1+2cos(theta))	sqrt((2cos(theta)+1)^2-1)			b1	b2
-1.25	0.75			-0.5	-2

```
#
# Copyright (c) 2011 Christopher Felton
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU Lesser General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU Lesser General Public License for more details.
#
# You should have received a copy of the GNU Lesser General Public License
# along with this program.  If not, see <http://www.gnu.org/licenses/>.
#

# The following is derived from the slides presented by
# Alexander Kain for CS506/606 "Special Topics: Speech Signal Processing"
# CSLU / OHSU, Spring Term 2011.

import numpy as np
import matplotlib.pyplot as plt
from  matplotlib import patches

def zplane(b,a,filename=None):
    """Plot the complex z-plane given a transfer function.
    """

    # get a figure/plot
    ax = plt.subplot(111)

    # create the unit circle
    uc = patches.Circle((0,0), radius=1, fill=False,
                        color='black', ls='dashed')
    ax.add_patch(uc)

    # The coefficients are less than 1, normalize the coeficients
    if np.max(b) > 1:
        kn = np.max(b)
        b = b/float(kn)
    else:
        kn = 1

    if np.max(a) > 1:
        kd = np.max(a)
    a = a/float(kd)
else:
    kd = 1

# Get the poles and zeros
p = np.roots(a)
z = np.roots(b)
k = kn/float(kd)

# Plot the zeros and set marker properties
t1 = plt.plot(z.real, z.imag, 'go', ms=10)
plt.setp( t1, markersize=10.0, markeredgewidth=1.0,
          markeredgecolor='k', markerfacecolor='g')

# Plot the poles and set marker properties
t2 = plt.plot(p.real, p.imag, 'rx', ms=10)
plt.setp( t2, markersize=12.0, markeredgewidth=3.0,
          markeredgecolor='r', markerfacecolor='r')

ax.spines['left'].set_position('center')
ax.spines['bottom'].set_position('center')
ax.spines['right'].set_visible(False)
ax.spines['top'].set_visible(False)

# set the ticks
r = 1.5; plt.axis('scaled'); plt.axis([-r, r, -r, r])
ticks = [-1, -.5, .5, 1]; plt.xticks(ticks); plt.yticks(ticks)

if filename is None:
    plt.show()
else:
    plt.savefig(filename)


return z, p, k
```