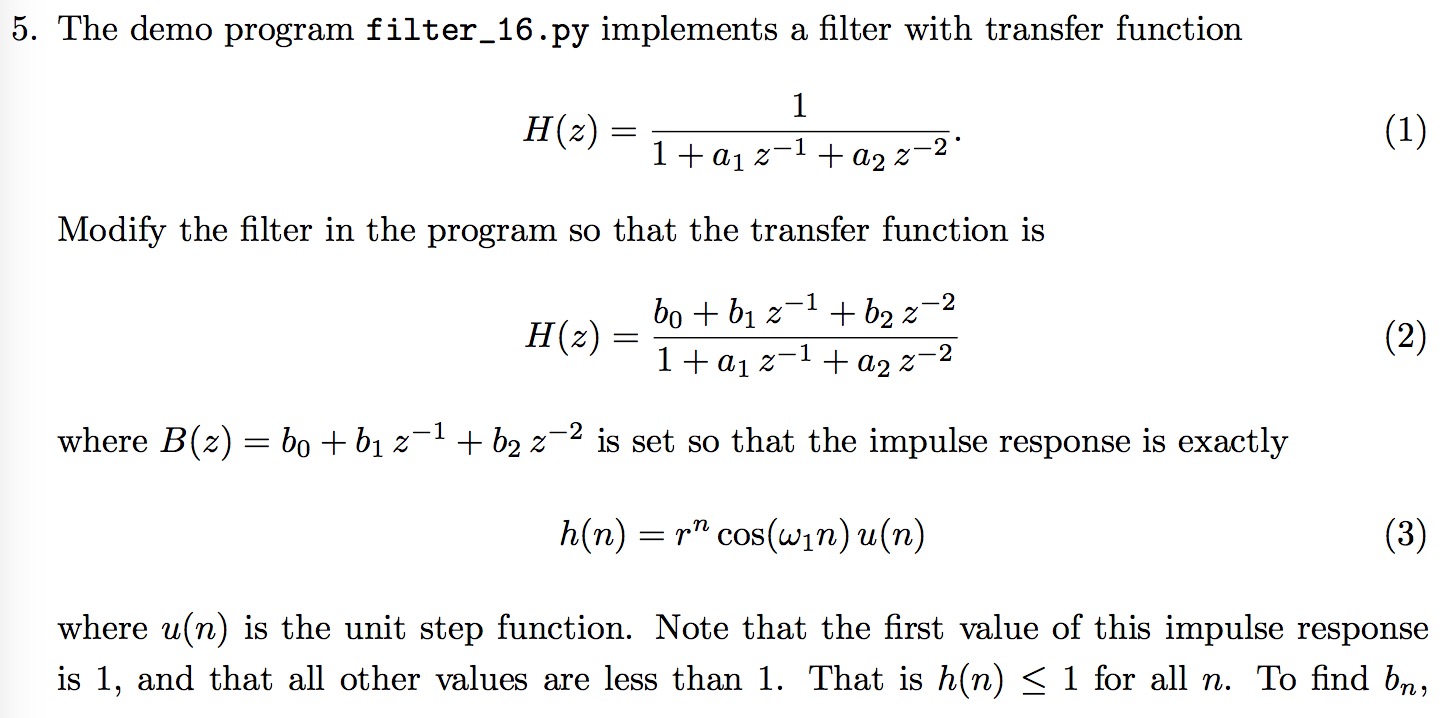
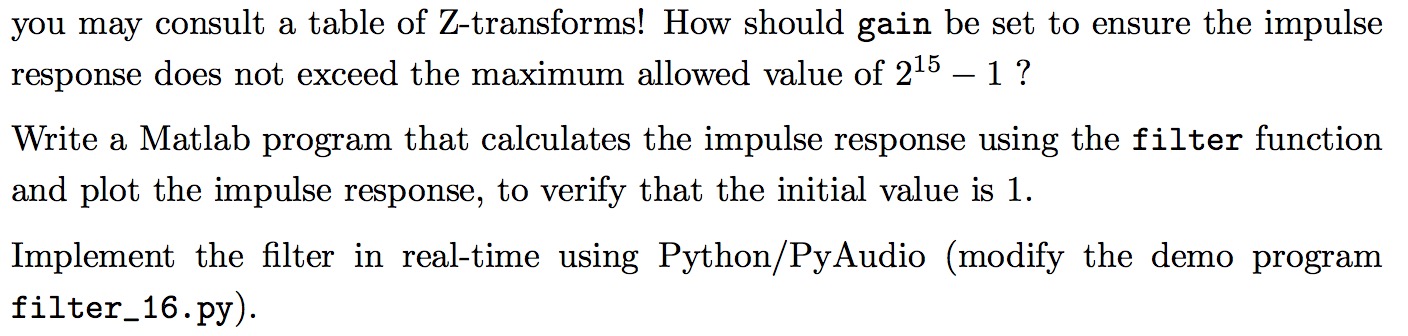
5. The demo program filter\_16.py implements a filter with transfer function

**Python code: 5\_my.py**

#16 bit/sample

#y(n) = x(n) -a1 y(n-1) -a2 y(n-2)

from math import cos, pi

import struct

import pyaudio

Fs = 8000

T = 1

N = T \* Fs

f = 800

om = 2.0 \* pi \* float(f) / Fs

r = 0.998

a0 = 1

a1 = -2 \* r \* cos(om)

a2 = r\*\*2

b0 = 1

b1 = -r \* cos(om)

b2 = 0

x1 = 0.0

y1 = 0.0

y2 = 0.0

gain = 10000.0

p = pyaudio.PyAudio()

stream = p.open(format = pyaudio.paInt16, #16 bits

channels = 1,

rate = Fs,

input = False,

output = True)

#

for n in range(0, N):

if n == 0:

x0 = 1.0

else:

x0 = 0.0

y0 = x0 + b1 \* x1 - a1 \* y1 - a2 \* y2

#delays

x1 = x0

y2 = y1

y1 = y0

output\_value = gain \* y0

if output\_value > 2\*\*15-1:

output\_value = 2\*\*15-1

elif output\_value < -2\*\*15:

output\_value = -2\*\*15

output\_string = struct.pack('h', int(output\_value))

stream.write(output\_string)

print(" \* finished \* ")

stream.stop\_stream()

stream.close()

p.terminate()

**Matlab code: q5.m**

%% 5.Using the filter function and plot the impulse response

clc

clear all

f = 800;

Fs = 8000;

om = 2 \* pi \* f / Fs;

r = 0.998;

a0 = 1;

a1 = -2 \* r \* cos(om);

a2 = r^2;

b0 = 1;

b1 = -r \* cos(om);

b2 = 0;

a = [a0 a1 a2]

b = [b0 b1 b2]

n = 0:100;

x = ( n==0 );

% make the filter

y = filter(b, a, x);

g = r.^n;

figure(1)

plot(n, y, 'o-', n, g, '--')

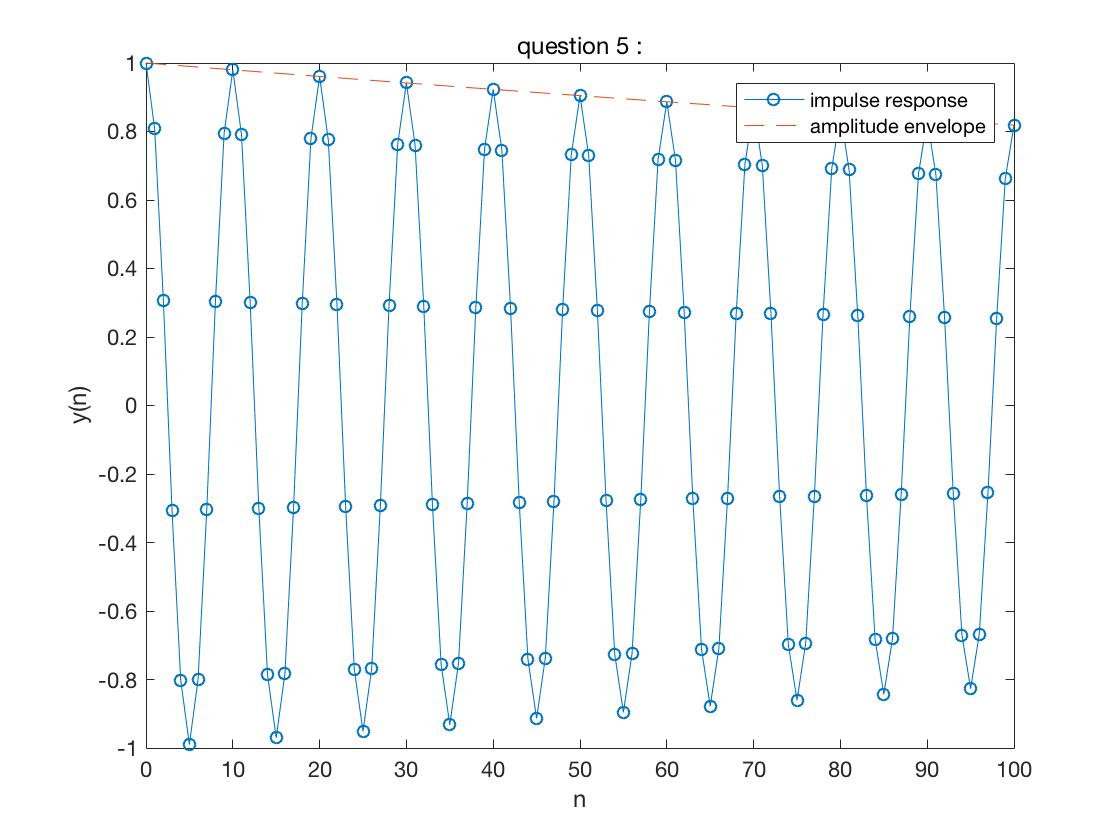
legend('impulse response', 'amplitude envelope')

xlabel('n')

ylabel('y(n)')

title('question 5 :')

**Figure:**

****

**Comment:**

**I looked up the z-transform in the table and then found the coefficient of H(z). Then I assignment a and b to make them become z-transform of h(n). I assumed the frequency is 800 Hz and r is 0.998. Then I used filter to verify the figure, it was begin with 1 and then attenuated as 0.998n. Since the figure is attenuation, gain should be set to ensure the initial output value less than 215-1.**