QUESTION 1

(a) Firstly use the Laplace transform

Y(s) = 3*sX(s) – 0.5*sY(s)

And move Y(s) to the same side

(1+0.5s)Y(s) = 3*X(s)

Due to H(s) = $\frac{Y(s)}{X(s)}$,so the transfer function is

H(s) = $\frac{3s}{1+0.5s}$

When 1+0.5s equals 0,s equals -2,so the pole is s = -2.

When 3s equals 0,s equals 0,so the zero is s = 0.

(b) The input pulse signal can be written as

x(t) = u(t) – u(t-T)

Therefore

X(s) = $\frac{1}{s} - e^{-sT}\left(\frac{1}{s}\right) = \frac{1-e^{-sT}}{s}$

And

Y(s) = H(s)X(s) = $\frac{3s(1-e^{-sT})}{(1+0.5s)s} = \frac{3(1-e^{-sT})}{1+0.5s} = 6(\frac{1}{s+2} - \frac{e^{-sT}}{s+2})$

Y(t) = L^{-1} [Y(s)] = $6u(t)e^{-2t} - 6u(t-T)e^{-2(t-T)}$

(c) Use the ilaplace function to change Y(s) into another function related to t.Then plot the output signal.*(Since use the syms s,so need to add subs in plot sentence)

And here is the code.

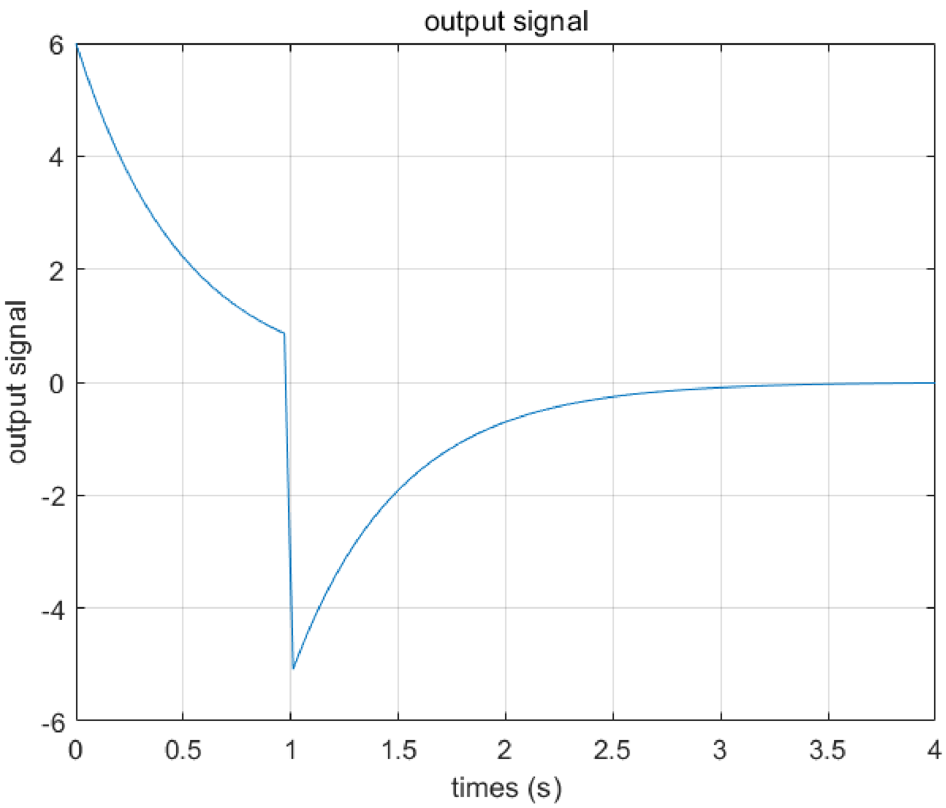
this does not plot the answer as obtained under (b)

DSP_Q1.m

```
syms s
T = 1;
% % % A = 3 - 3*(exp(-s*T));
% % % B = 1+0.5*s;
Y = (3 - 3*(exp(-s*T)))/(1+0.5*s);
Y_T = ilaplace(Y);


t = linspace(0, 4);
plot(t,subs(Y_T));


xlabel('times (s)');
ylabel('output signal');
title('output signal');
grid;
```



QUESTION 2

(a) We have

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

Thus

$Y(z) = \frac{(a+z^{-1})X(z)}{1+az^{-1}}$

And eliminate the denominator

$Y(z)\ (1+az^{-1}) = (a+z^{-1})X(z)$

or

$Y(z) = (a+z^{-1})X(z) - Y(z)az^{-1}$

So now we can transform this equation to the time domain by taking the inverse z-transform

y(n) = ax(n)+x(n-1)-ay(n-1) ✓

(b) Multiply both numerator and denominator with z

$H(z) = \frac{az+1}{z+a}$

The poles are the roots of the denominator

z+a = 0

which has the solution

z = -a ✓

The poles of this system thus lie at z = -a which mean a ∈ [-1, 1] (-1,1)

(c) The complex frequency response is

$$H(\omega) = H(z)|_{z=e^{j\omega T}} = \frac{a+e^{-j\omega T}}{1+ae^{-j\omega T}}$$

And use the Euler's formula

$$H(\omega) = \frac{a+cos\omega T - jsin\omega T}{1+acos\omega T - ajsin\omega T}$$

So

$$|H(\omega)|^2 = |\frac{(a+\cos\omega T)^2+\sin^2\omega T}{(1+a\cos\omega T)^2+(a\sin\omega T)^2}|$$ ✓

$$= \left|\frac{a^2+2\,a\cos\omega T+\cos^2\omega T+\sin^2\omega T}{1+2a\,cos\,\omega T+a^2\cos^2\omega T+a^2\sin^2\omega T}\right|$$

$$= \left|\frac{a^2+2\,a\cos\omega T+1}{1+2a\,cos\,\omega T+a^2}\right|$$

$$= 1$$

(d) If x(t) $=\sin\omega t$,X(z) = $\frac{z\sin\omega T}{z^2-2z\cos\omega T+1}$

So

Y(z) = H(z)X(z) = $\frac{a+z^{-1}}{1+az^{-1}}\times\frac{z\sin\omega T}{z^2-2z\cos\omega T+1}$

$$=\frac{az\sin\omega T+\sin\omega T}{z^2-2z\cos\omega T-2a\cos\omega T+az^{-1}+az+1}$$

this does not clarify how y is different from x;

(e) When a = 0.5 then y(n) = 0.5x(n)+x(n-1)-0.5y(n-1)

which is

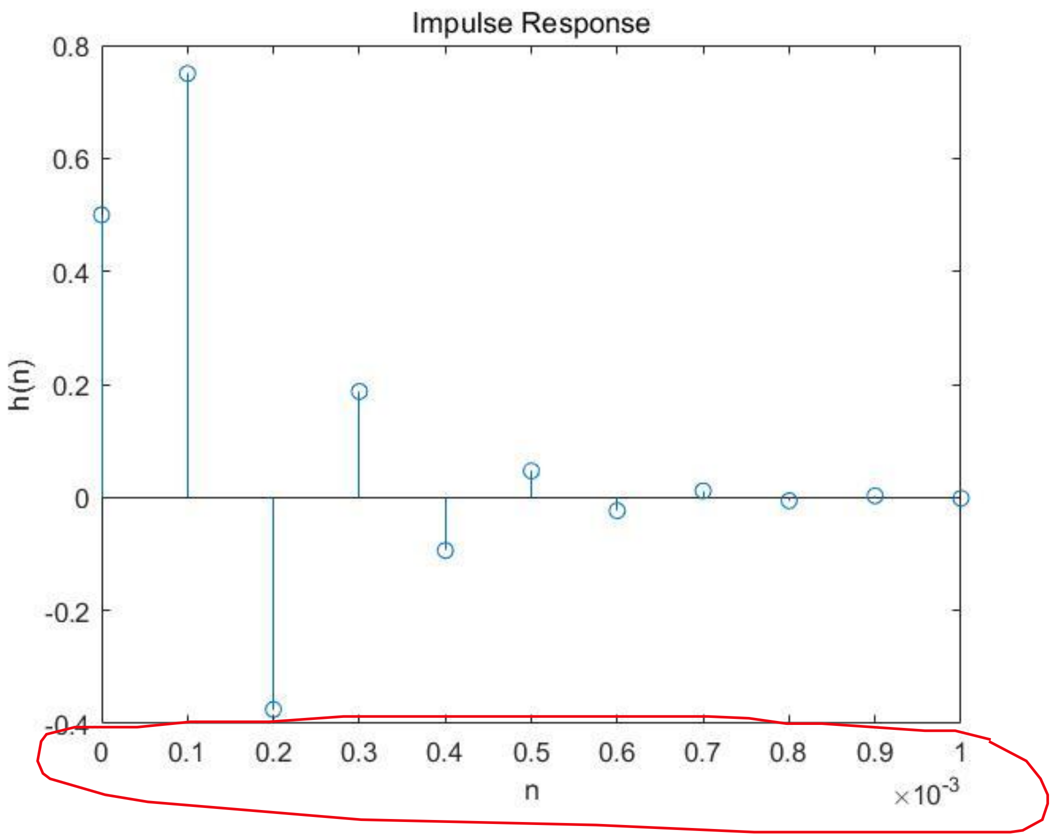
y(n)+ 0.5y(n-1) = 0.5x(n)+x(n-1)

Use the impz(b,a,n) function where a is the difference equation coefficient of output y,b is the coefficient of input x

DSP_Q1.m | DSP_Q2.m | DSP_(

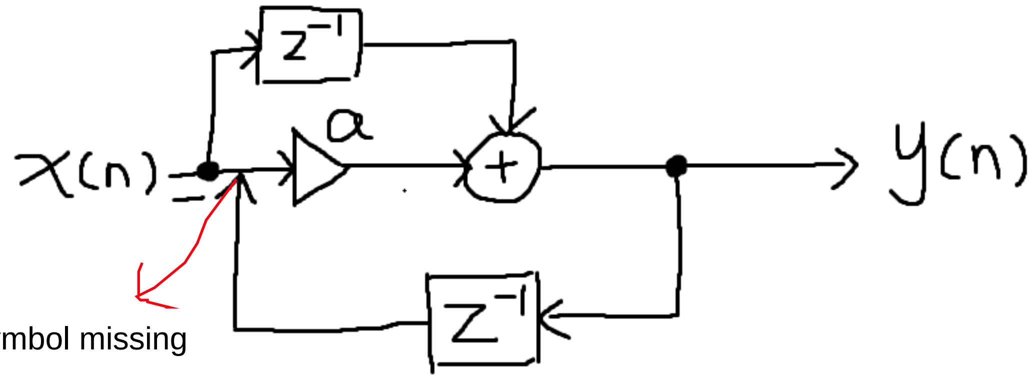
```
b = [0.5, 1];
a = [1, 0.5];
n = 0:10;
[h,n] = impz(b,a,n,10000)

stem(n,h);
title('Impulse Response');
xlabel('n');
ylabel('h(n)');
```



time axis not in

(f) $y(n) = ax(n)+x(n-1)-ay(n-1)$



+ symbol missing

QUESTION 3

(a)

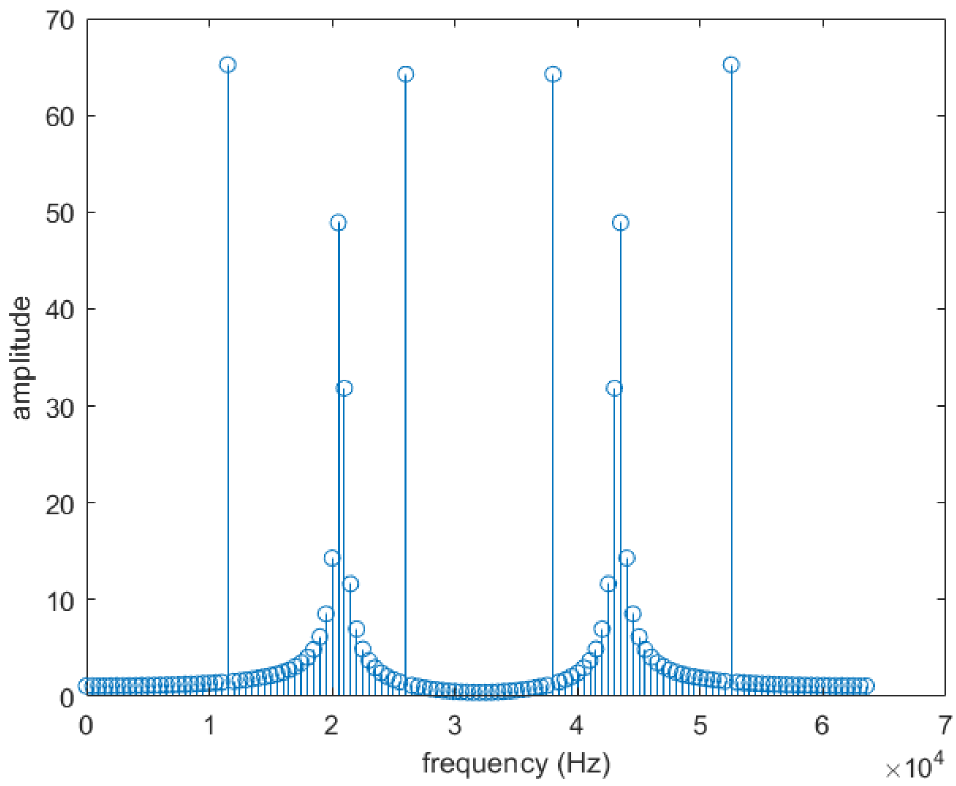
DSP_Q1.m DSP_Q2.m DSP_Q3.m

```
Fs = 64000;
T = 1/Fs;
Ns = 128;
t = (0:Ns-1)*T;
f = (0:Ns-1)*(Fs/Ns);

x = cos(2*pi*11500*t) + cos(2*pi*20700*t) + cos(2*pi*38000*t);

X = fft(x);

stem(f,abs(X));
xlabel('frequency (Hz)');
ylabel('amplitude');
```



where?

(b) Leakage happens when that frequency component does not have an integer number of periods within the analysed window of N samples.

As a result, the energy of the signal is ‘spread out’ out over neighbouring bins.

And Leakage effects can be alleviated by applying a (more smooth) window.

question asks "eliminate"

(c)
~~(d)~~

Leakage effects can be alleviated by applying a (more smooth) window.

In order to reduce spectrum energy leakage, different interception functions can be used to truncate the signal, which is called window function, or window for short.

explain further

(d)
~~(e)~~ DFT multiplication complexity is

O(N*N) = O(16384)

FFT multiplication complexity is

$O(\frac{N}{2} * \log_2 N)$ = $O(64 * \log_2 128)$ = O(64*7) = O(448)

So

$$\frac{O(N*N)}{O(\frac{N}{2}*\log_2 N)} = \frac{O(16384)}{O(448)} = 36.57$$

QUESTION 4

(a)

x_1	1	0	3	2						
x_2				1	-1	0	2	1		
Y	2									
x_1		1	0	3	2					
x_2				1	-1	0	2	1		
Y	3-2=1									

x_1			1	0	3	2			
x_2				1	-1	0	2	1	
Y	0-3+0=-3								
x_1				1	0	3	2		
x_2				1	-1	0	2	1	
Y	1+0+0+4=5								
x_1					1	0	3	2	
x_2				1	-1	0	2	1	
Y	-1+0+6+2=7								
x_1						1	0	3	2
x_2				1	-1	0	2	1	
Y	0+0+3=3								
x_1				1	0	3	2		
x_2	1	-1	0	2	1				
Y	2+0=2								
x_1					1	0	3	2	
x_2	1	-1	0	2	1				
Y	1								

So

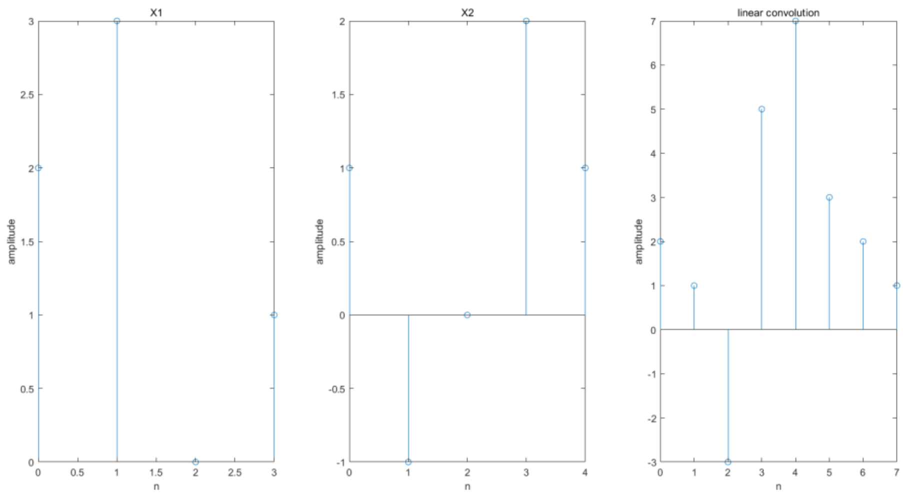
Y = [2, 1, -3, 5, 7, 3, 2, 1] ✓

(b)

DSP_Q1.m DSP_Q2.m DSP_Q3.m DSP_Q4.m

```
x1 = [2,3,0,1];
nx1=[0:length(x1)-1];
x2 = [1,-1,0,2,1];
nx2=[0:length(x2)-1];
y=conv(x1,x2);
ny=[0:1:length(y)-1];
subplot(1,3,1)
stem(nx1,x1);
xlabel('n')
ylabel('amplitude')
title('X1')
subplot(1,3,2)
stem(nx2,x2)
xlabel('n')
ylabel('amplitude')
title('X2')
subplot(1,3,3)
stem(ny,y);
xlabel('n')
ylabel('amplitude')
title('linear convolution')
```

-> this does not use DFTs



(c) Segmentation of signal into fixed-size data blocks, to avoid transform of very long sequences. Avoids large memory requirements and long lag in case of on-line processing

OVERLAP-SAVE

OVERLAP-ADD