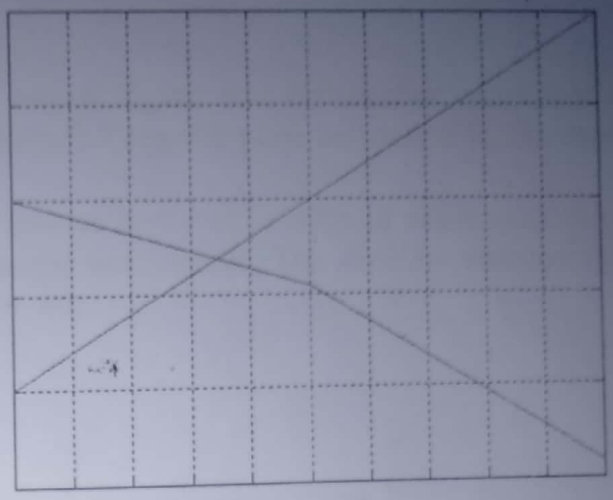
Now,
Calculate the point of intersection of both the Signals. More precise value can be obtained by Zoom In.

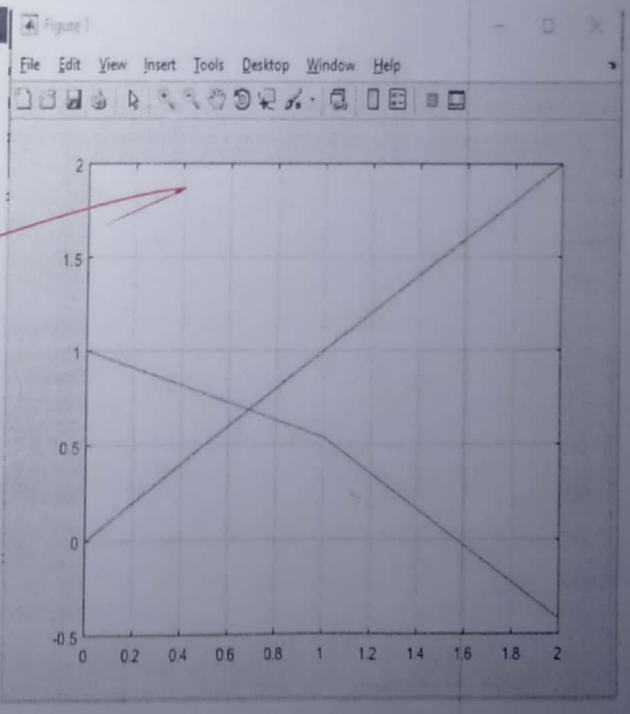
Point of intersection is
0.68



Example:

Command Window

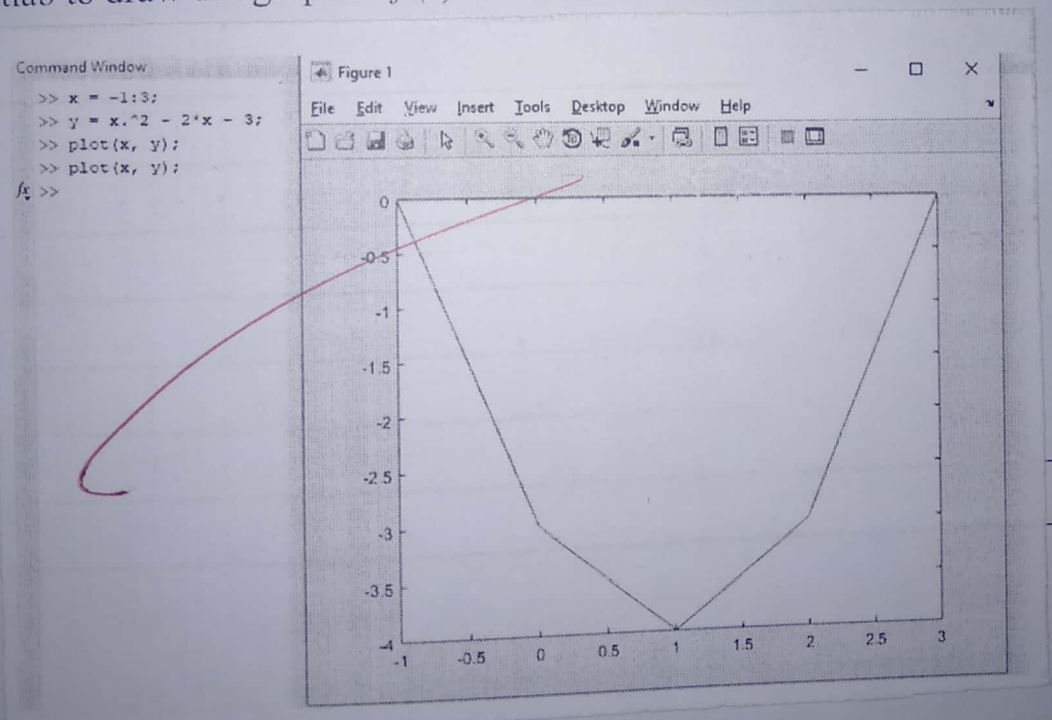
```
>> clear all
>> x = 0:2

x =

     0     1     2

>> y = cos(x);
>> plot(x, y);
>> Y = cos(x);
>> y=x;
>> plot(x,y,xY);
Undefined function or variable 'xY'.

>> plot(x,y,x,Y);
>> grid on;
>>
```



Scanned by CamScanner

EXERCISES 1

1. Use Matlab to draw the graph of $f(x) = x^2 - 2x - 3$ on the interval $[-1, 3]$.



Scanned by CamScanner

EXERCISE 2

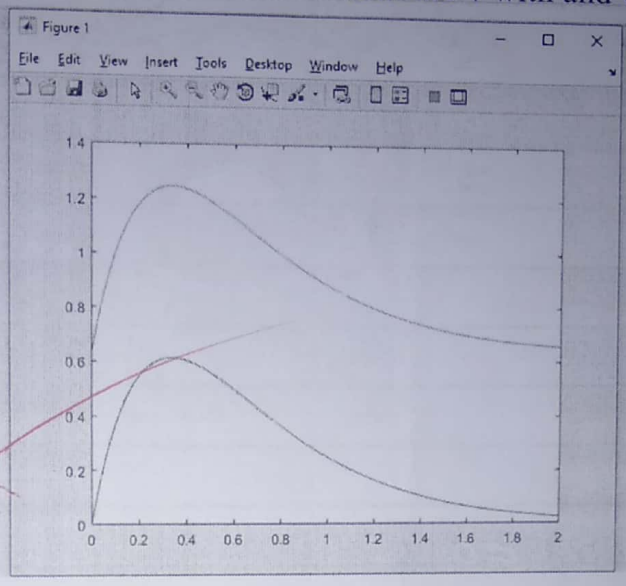
1. Create a **symbol plot** with the data generated by adding noise to the given function: y = 5*x.*exp(-3*x)

   Where x = 0:0.01:2;

Hint: In order to add noise study randint and rand functions using help command.

It's a multiple plot question; two possible input functions are **v** with and without noise (say y1).

```
Command Window
>> x = 0:0.01:2;
>> y=5*x.*exp(-3*x);
>> y1 = y + rand();
>> plot(x, y, x, y1);
fx >>
```



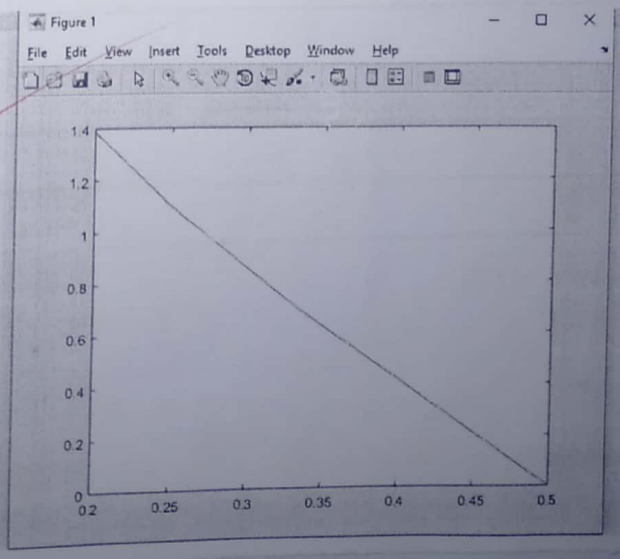
Orange: with noise

Blue: without noise

1. Sketch the graphs of $f(x) = 1/x$ and $g(x) = \ln(x - 1)$ on the interval $[2, 5]$.

Exercise 2.2:

```
Command Window
>> clear all;
>> x = 2:5;
>> y = 1./x

y =

    0.5000    0.3333    0.2500    0.2000

>> z = log(x - 1);
>> plot(y, z);
fx >>
```



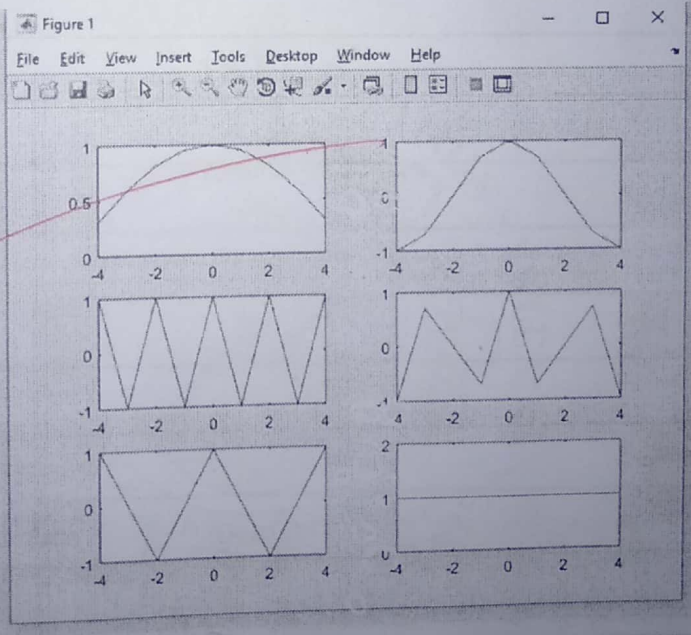
Scanned by CamScanner

EXERCISE 3

1. Write a sequence of MATLAB commands in the space below to plot the discrete time sinusoid $x(n) = \cos\omega n \ (-5 < n < 5)$ *for the following values of* angular frequency. You must divide your figure into 6 subplots. What's the conclusion?

- $\omega = 0.10\pi$
- $\omega = 0.25\pi$
- $\omega = \pi$
- $\omega = 1.25\pi$
- $\omega = 1.50\pi$
- $\omega = 2\pi$

```
>> n = -4:4;
w = 0.10 * pi;
subplot(3,2,1);
x = cos(w .* n);
plot(n, x);
plot(x, n);
plot(n, x);
w = 0.25 * pi;
x = cos(w .* n);
subplot(3,2,2);
plot(n, x);
subplot(3,2,3);
w = pi;
x = cos(w .* n);
plot(n, x);
w = 1.25 * pi;
x = cos(w .* n);
subplot(3,2,4);
plot(n, x);
subplot(3,2,5);
w = 1.5 * pi;
x = cos(w .* n);
plot(n, x);

subplot(3,2,6);
w = 2 * pi;
x = cos(w .* n);
plot(n, x);
```



Scanned by CamScanner

Generate a time vector **t** from -0.5 to 0.5 with a step size of 0.001. Implement the function $x(t) = e^{-t/p}$ where $p=0.1$. In order to simulate an A/D converter, perform the following tasks:

1. Plot x and its magnitude spectrum in a two-panel figure window. What is the bandwidth of x?

2. Set the sampling frequency to twice the bandwidth of the signal **x** (which is approximately 25Hz). Generate a rectangular pulse train starting at -0.5 to 0.5 where the step size is *1/ (sampling frequency)* and with a duration of 0.0001. In a two-panel figure window, plot the pulse train and its magnitude spectrum.

3. What is the relation between the time and frequency domain representation of the pulse train?

a time-domain graph shows how a signal changes over time where as Frequency domain graph shows how much of the signal lies with in each frequency band over a range of frequency.

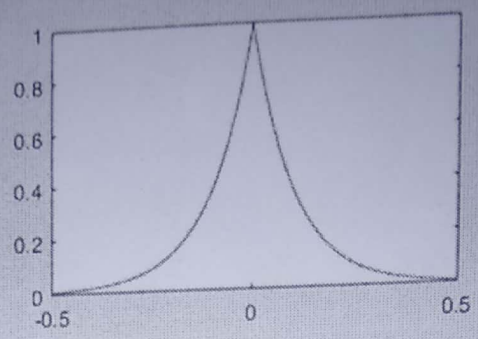
4. Sample x using the pulse train and plot the resulting sampled version of x. Also, plot its magnitude spectrum. What can you observe from both plots?

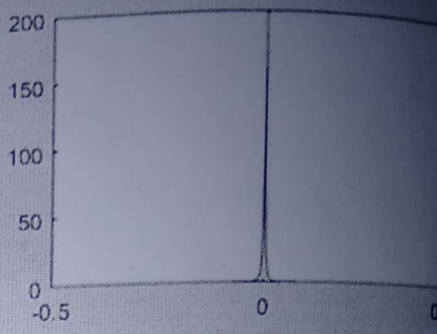
The magnitude of spectrum we observed now must energy in ~~contributed~~ distributed in freq domain.

Scanned by CamScanner

Command Window

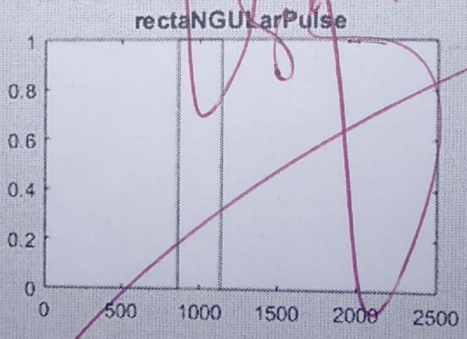
```
>> t = -0.5:0.001:0.5;
>> p = 0.1;
>> x = exp(-abs(t)/p);
>> subplot(2,2,1),plot(t,x);
>> y = fftshift(abs(fft(x)));
>> subplot(2,2,2),plot(t,y);
>> [lower,upper] = bandwidth(x);
>> lower

lower =

     0

>> upper

upper =

        1000

fx >>
```

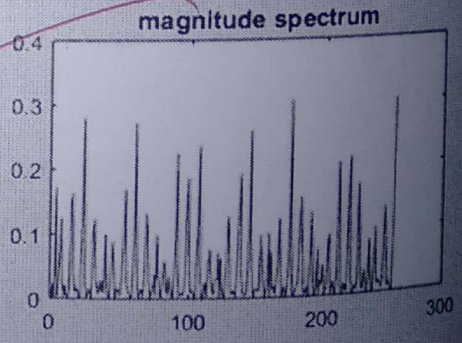


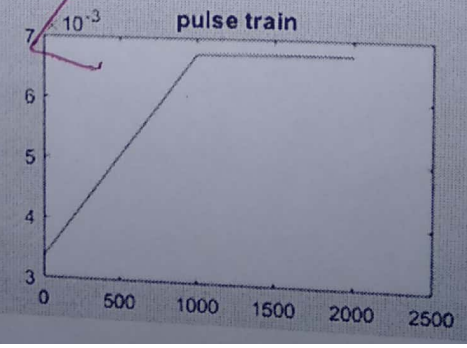


Command Window

```
>> t = -0.5:0.001:0.5;
>> p = 0.1;
>> x = exp(-abs(t)/p);
>> y = bandwidth(x,'upper');
>> fs = y*2;
>> a = -0.5:1/fs:0.5;
>> z = exp(-abs(a)/p);
>> b = reactangularPulse(z);
Undefined function or variable

Did you mean:
>> b = rectangularPulse(z);
>> mag = mscohere(a,b);
>> d = 0.0001;
>> c = pulstran(a,d,z);
>> subplot(2,2,1),plot(b);
>> title('rectaNGULarPulse');
>> subplot(2,2,2),plot(mag);
>> title('magnitude spectrum')
>> subplot(2,2,3),plot(c);
>> title('pulse train');
fx >>
```





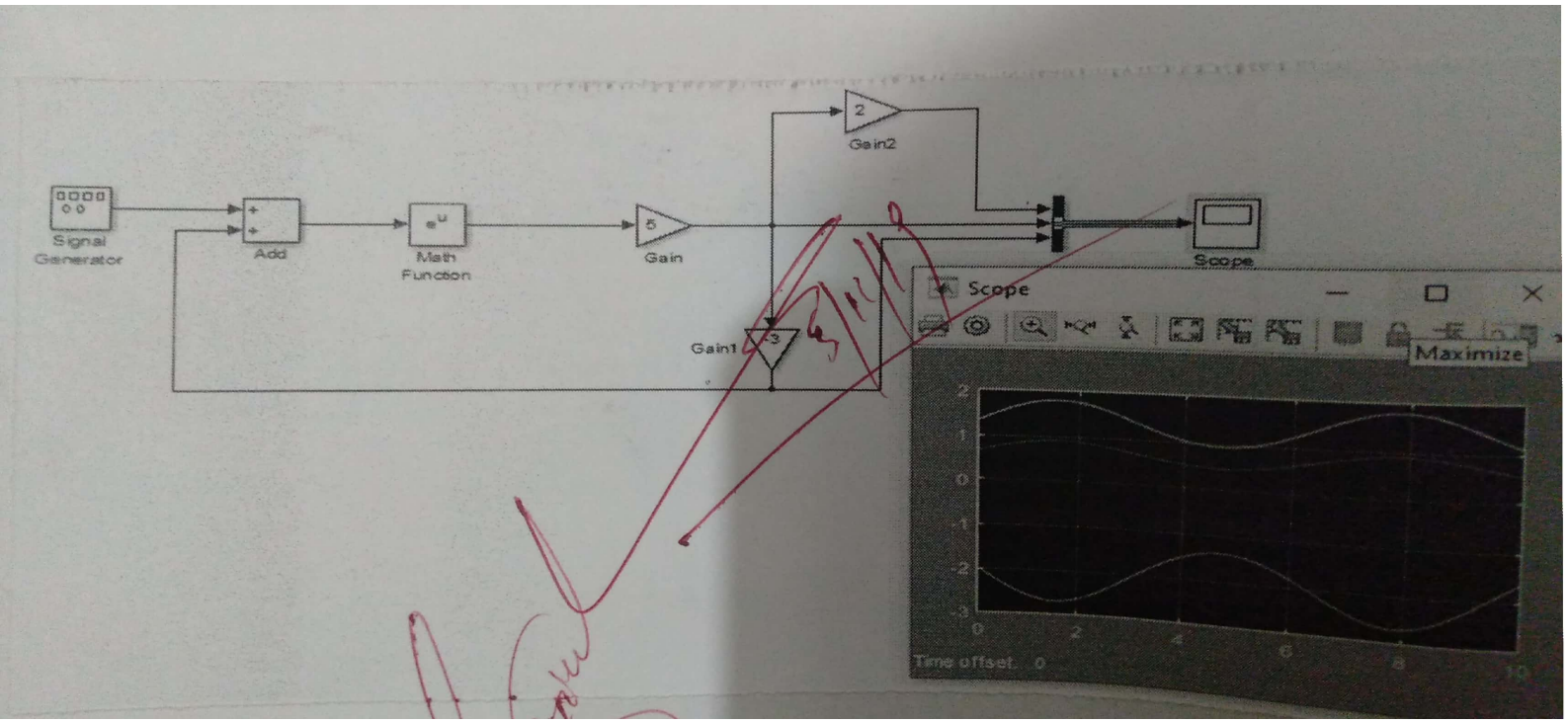


Scanned by CamScanner

EXERCISE

1. Apply gain of +5 to an exponential signal, part of which is propagated ahead to get a further gain of +2 and another part is fed back to the source after experiencing a loss of -3. Attaches scopes at appropriate points and obtain the wave form of the resultant signal. Attach the figures and the plot.

Scanned by CamScanner



Scanned by CamScanner

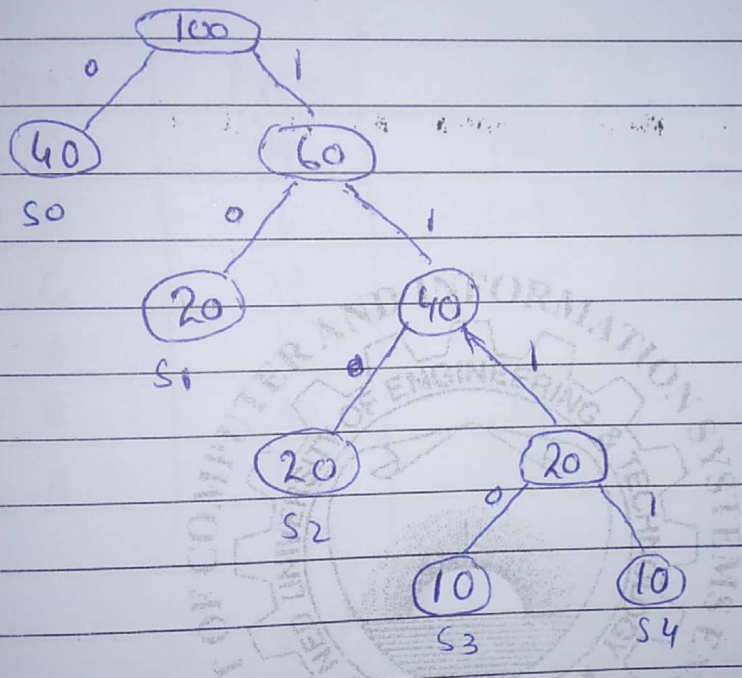
1. Five source symbols of the alphabet of a source and their probabilities are shown

Symbol	Probability
S0	0.4
S1	0.2
S2	0.2
S3	0.1
S4	0.1

a) Build the Huffman code tree for the message and find the codeword for each character.

Scanned by CamScanner

	Char	freq	Code	Code length
1	S4	10	1111	4
2	S3	10	1110	4
3	S2	20	110	3
4	S1	20	10	2
5	S0	40	0	1



) Write MATLAB code to obtain Huffman coding for the given characters.

```
Sig = repmat ([1 2 3 3 4 4 5 5 5 5]) ;
Symbols = [1 2 3 4 5];
P = [0.1 0.1 0.2 0.2 0.4];
dict = huffmandict (symbols, P);
hcode = huffmanenco (sig, dict);
dhsig = huffman deco (hcode, dict);
```

Scanned by CamScanner

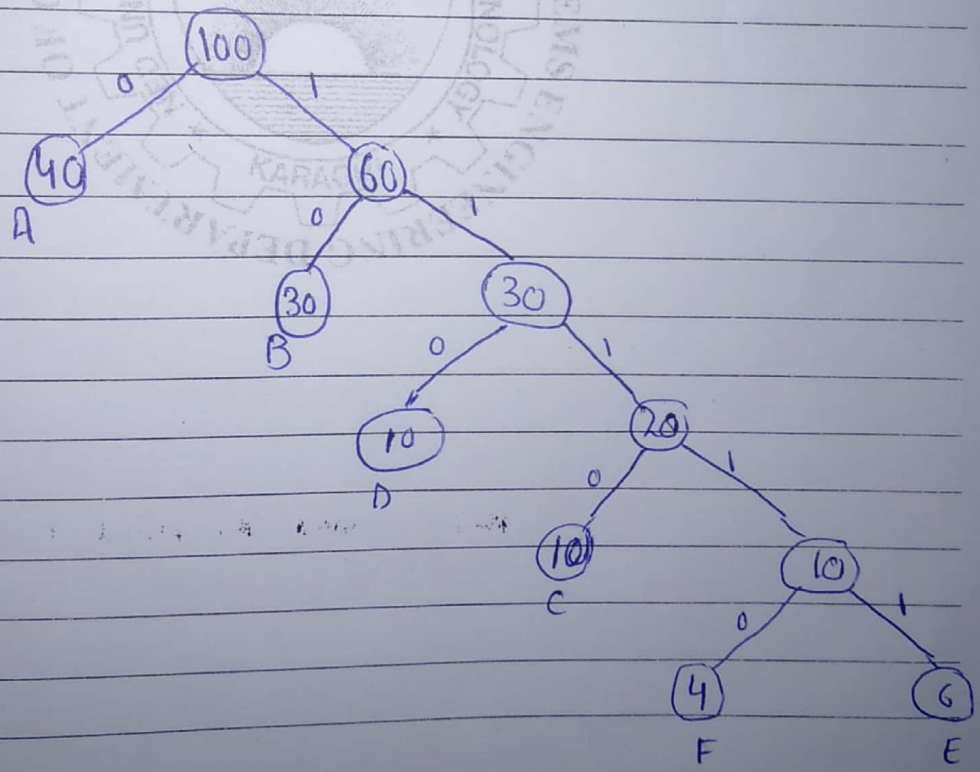
Construct a Huffman code for the source X emitting symbols with probabilities as shown below.

Symbol	Probability	Symbol	Probability
A	0.4	B	0.3
C	0.1	D	0.1
E	0.06	F	0.04

Determine:

i) The code word for each symbol.

	Char	freq	Code	Code length
1	F	4	1110	5
2	E	6	1111	5
3	C	10	110	4
4	D	10	10	3
5	B	30	0	2
6	A	40		1



ii) The average number of transmitted binary digits per code word.

Total no of bits = freq x Code length

= (4×5) + (6×5) + (10×4) + (10×3) + (30×2) + (40×1)

Total no of bits. = 220

Scanned by CamScanner

Avg. no. of bits = Total bits / highest no

= 220 / 100

Avg no of bits. = 2.2 bits

c) Write MATLAB code to obtain Huffman code for the given source.

```
sig = repmat
symbols = [1 2 3 4 5 6];
dict = huffmandict (symbols, p);
hcode = huffmanenco (sig, dict);
dhsig = huffmandeco (hcode, dict);
```

current value > sum value

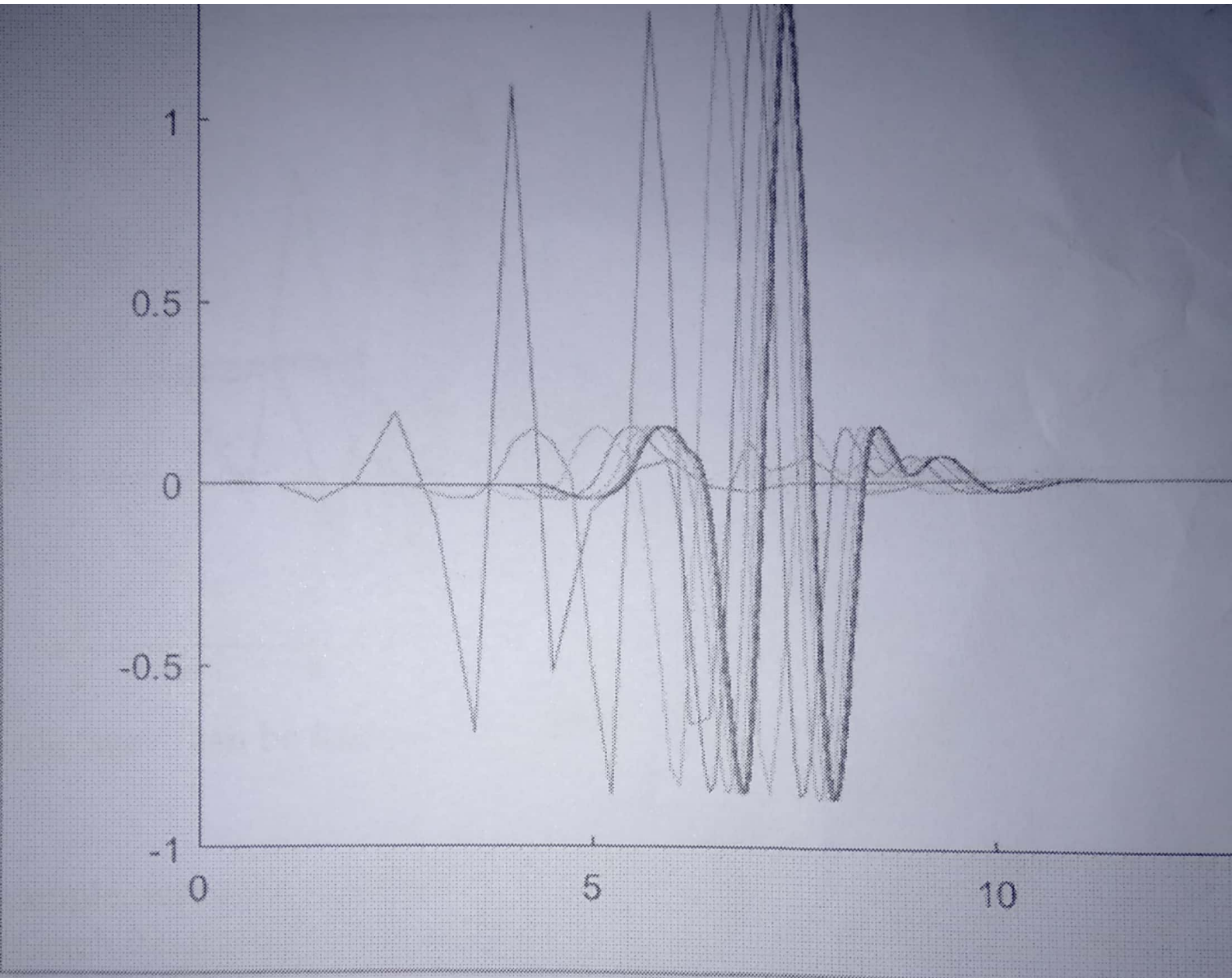
```
Symbol = {'A' 'B' 'C' 'D' 'E' 'F'};
P = [0.4 0.3 0.1 0.1 0.06 0.04];
dict = huffmandict (symbol, P);
hcode = huffmanenco (
```

Scanned by CamScanner

EXERCISES

1. Using the above example, write the code for 25 piecewise linear approximations of the sym8 wavelet obtained after each iteration of the cascade algorithm. Attach the plot.

Scanned by CamScanner

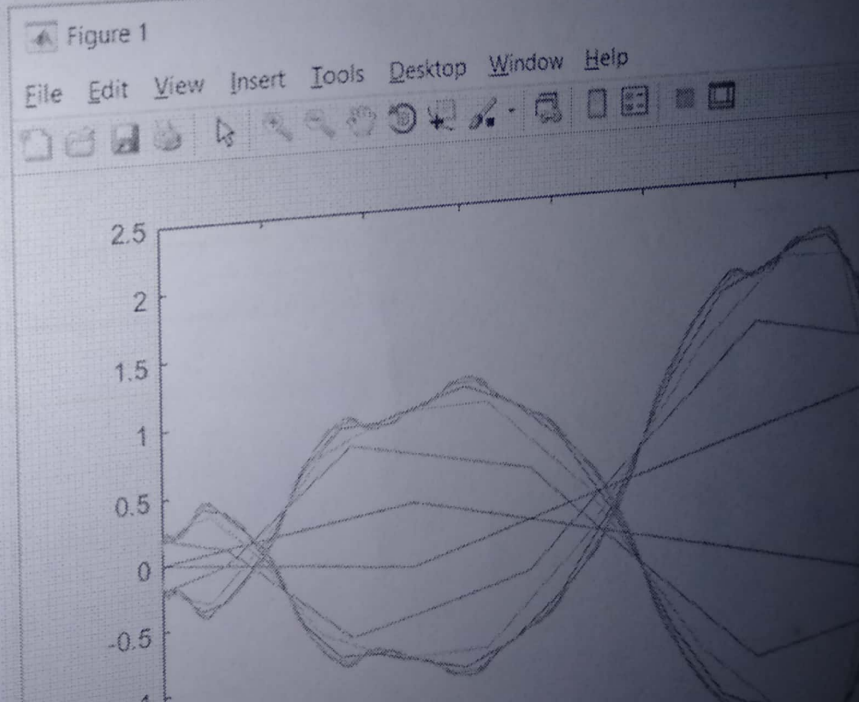



```
>> iter2=25;
>> wav='sym8';
>> for i=1:iter2
[phi,psi,xval]=wavefun(wav,i);
plot(xval,psi);
hold on
end
```

Scanned by CamScanner

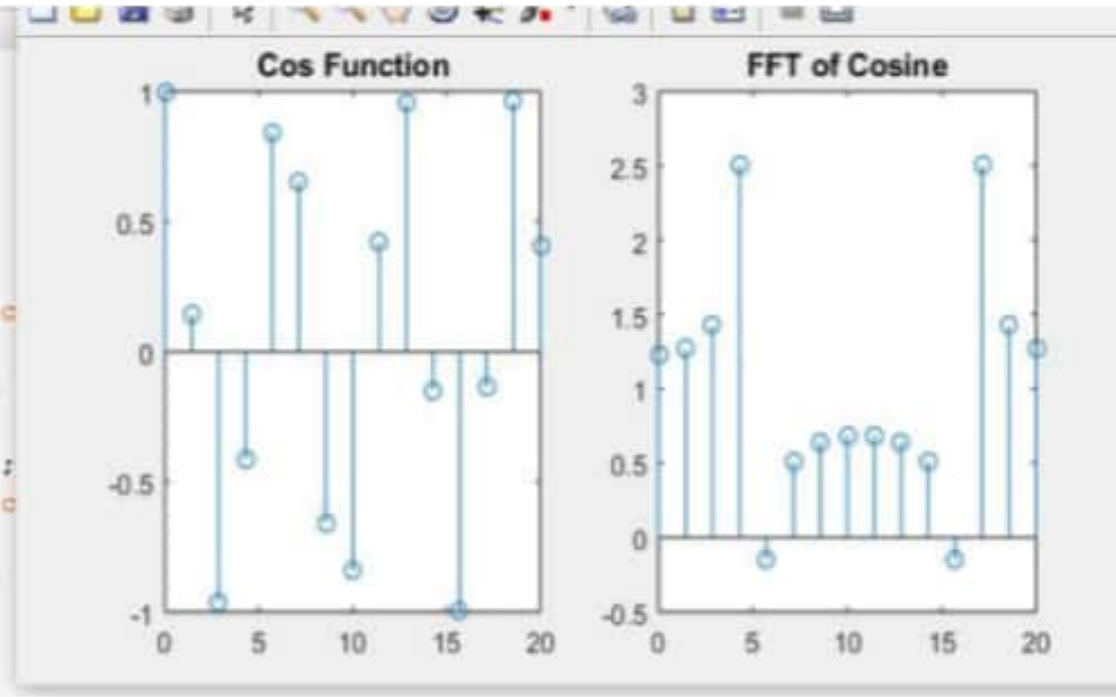
Referring to the given example, execute the biorthogonal function and give the code below.
Note: The value of the variable 'wav' will be 'bior' for biorthogonal functions and the range
must be seen from MATLAB help]

```
>>
>>
>>
>>
>>
>>
>> iter2=10;
>> wav='bior3.7';
>> for i=1:iter2
[phi,psi1,xval]=wavefun(wav,i);
plot(xval,psi1);
hold on
end
>>
>>
>>
>>
```

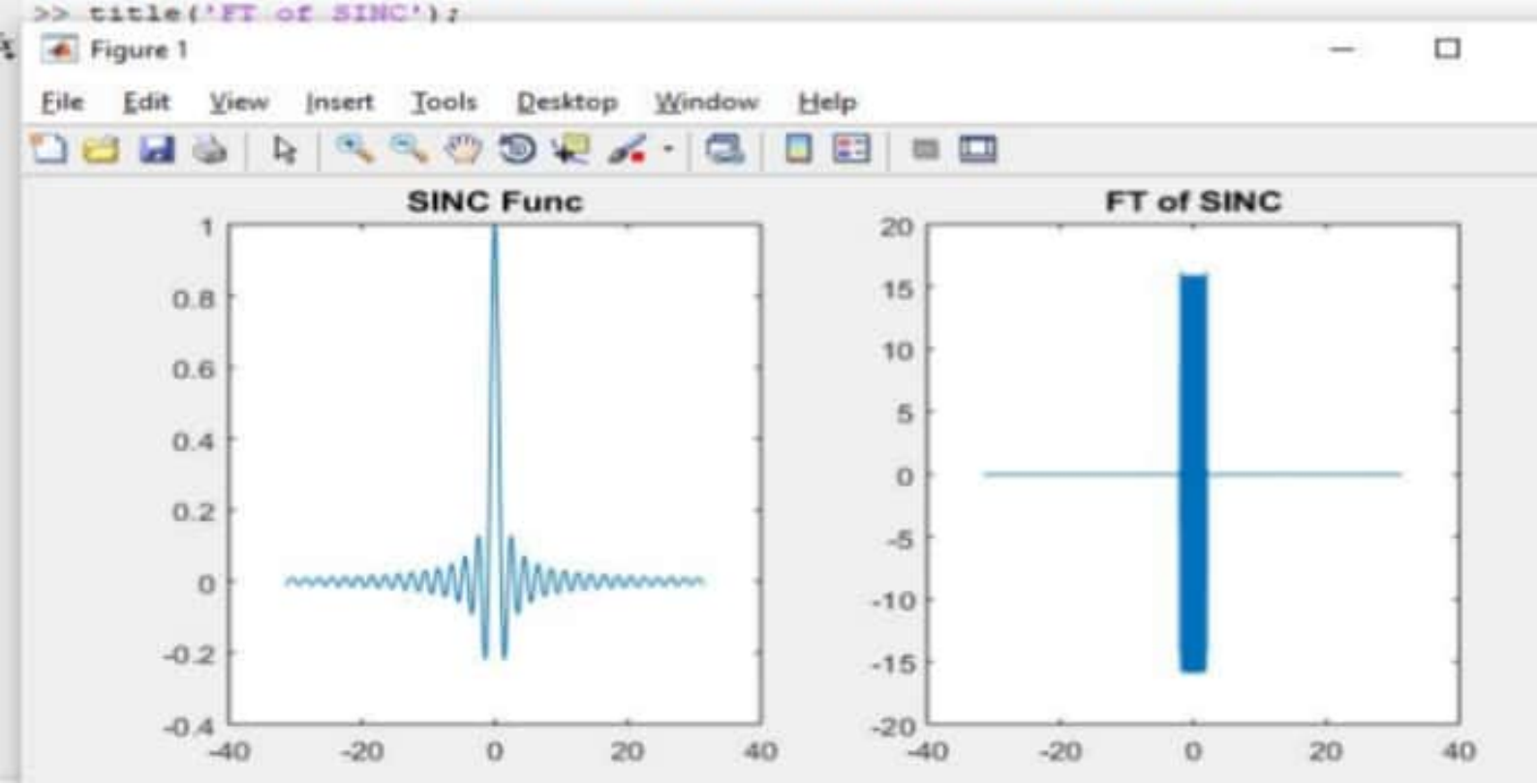


Scanned by CamScanner

```
>> x = linspace(0,20,15);
>> s = cos(x);
>> subplot(1,2,1),stem(x,s);
>> title('Cos Function');
>> F = fft(s);
>> subplot(1,2,2),stem(x,F);
Warning: Using only the real c
> In getRealData (line 14)
  In stemHGUsingMATLABClasses
  In stem (line 29)
>> subplot(1,2,2),stem(x,(F));
Warning: Using only the real c
> In getRealData (line 14)
  In stemHGUsingMATLABClasses
  In stem (line 29)
>> title('FFT of Cosine');
```




```
>> x = linspace((-10*pi),(10*pi),1000);
>> s = sinc(x);
>> subplot(1,2,1),plot(x,s);
>> title('SINC Func');
>> f = fft(s);
>> subplot(1,2,2),plot(x,fftshift(f));
Warning: Imaginary parts of complex X and/or Y arguments ignored
>> title('FT of SINC');
```



Scanned by CamScanner

EXERCISES

1. Only for positive values of t , if bit 1 is represented by a square wave with +5 Volts and bit 0 with -5 Volts, and if the duration of a bit is 1 second, then plot a time graph for the following sequence: 1011010001

Scanned by CamScanner

QUESTION 1:

```
1 -   a=5;
2 -   t=1;
3 -   wav=[1:10]
4 -   seq=[1,0,1,1,0,1,0,0,0,1];
5 -   x=length(seq);
6 -   y=[1:t:10];
7 -   for i=1:x;
8 -       if seq(i)==0
9 -           wav(i)=-a
10 -      else
11 -          wav(i)=a
12 -      end
13 -  end
14 -  stairs(wav)
15 -  axis([0 10 -5 5])
16
```

5 -5 5 5 -5 5

wav =

Scanned by CamScanner