Expt. No. 1a GENERATION OF SEQUENCES

AIM:

To write a program to generate different waveforms using MATLAB

SOFTWARE REQUIRED:

MATLAB Software

ALGORITHM:

1. Clear command window.
2. Get the choice from user to select the waveform to be generated.
3. Use switch case to execute the code for different waveforms.
4. Generate
   i) Unit impulse: Iterate n from -20 to 20. Generate output=1 when n = 0. ii) Unit step: Iterate n from -20 to 20. Generate output=1 when n >= 0. iii) Ramp: Iterate n from -20 to 20. Generate output=n when n >= 0. iv) Exponential: Iterate n from -20 to 20. Generate output=exp(n) when n >= 0. v) Sine: Iterate n from 0 to 4*pi. Generate output = sin(n) vi) Cosine: Iterate n from 0 to 4*pi. Generate output = cos(n) vii) Triangular: Iterate n from 0 to 20 in steps of 0.2.
   Generate output=sawtooth(n,0.5)
   viii) Sawtooth: Iterate n from 0 to 20 in steps of 0.2.
   Generate output=sawtooth(n,1)
5. Plot the signal.
6. Get the input from user if another waveform needs to be generated.
7. If yes, jump to Step 4, else terminate the program.

FLOWCHART:

B

ch = choice

A

If ch = 1-4

Start

Ye
~~s~~

No

1 2 3 4

with output = 1, when
n >= 0

Display the signal

Iterate n from -20 to
20.

Plot the signal

Display the signal

Generate impulse
signal with
output=1, when n =
0

Iterate n from -20 to
20.

Plot the signal

Generate step signal

Iterate n from -20 to 20.

Generate ramp signal with output = n, when n >= 0

Display the signal

Plot the signal

Iterate n from -20 to 20.

Generate exponential signal with output = exp(n), when n >= 0

Display the signal

Plot the signal

C

A

5 6 7 8

Iterate n from 0 to 4*pi.

Generate sinusoidal signal with output=sin(n)

Display the signal

Plot the signal

Iterate n from 0 to 4*pi.

Generate cosine signal with output=cos(n)

Display the signal

Plot the signal

C

Iterate n from 0 to 20 in steps of 0.2

Generate triangular signal with output=sawtooth(n,0.5)

Display the signal

Plot the signal

Iterate n from 0 to 20 in steps of 0.2

Generate triangular signal with output=sawtooth(n,1)

Display the signal

Plot the signal

Stop

If ch = 1-8

Get user choice ‘ch’ to continue / terminate

Ye s
B

No

NOTE: Pls include the 9^{th} one which is the random signal

PROGRAM:

```
%WAVEFORM GENERATOR
clc
clear all
close all
disp('Program for Waveform generation');
opt=1;

while(opt==1)
 disp('Which waveform you want to generate?');
 disp('1.Impulse,2.Step,3.Ramp,4.Exponential,5.Sine,6.Cosine,7.Triangle,8.Sawtooth,
9.Random Signal');
 k=input('ENTER YOUR CHOICE:');
 switch k

%IMPULSE WAVEFORM
  case 1
  n = [-20:1:20];
 for k=1:1:length(n)
 if(n(k)==0)
 x(k)=1;
 else
 x(k)=0;
 end
 end
% disp(x);

 subplot(5,2,1)
 stem(n,x);
 xlabel('n -->');
 ylabel('amplitude');
 title('UNIT IMPULSE SIGNAL');
```

```
%STEP WAVEFORM
case 2
n =[-20:1:20];
for k=1:1:length(n)
 if(n(k)>=0)
 x(k)=1;
 else
 x(k)=0;
 end
end
% disp(x);
subplot(5,2,2)
stem(n,x);
xlabel('n -->');
ylabel('amplitude');
title('UNIT STEP SIGNAL');

%RAMP WAVEFORM
case 3
n = [-20:1:20];
for k=1:1:length(n)
 if(n(k)>=0)
 x(k)=n(k);
 else
 x(k)=0;
 end
 end
% disp(x);
 subplot(5,2,3)
 stem(n,x);
 xlabel('n -->');
 ylabel('amplitude');
 title('RAMP SIGNAL');


 %EXPONENTIAL WAVEFORM
 case 4
 n = [-20:1:20];
 for k=1:1:length(n)
 if(n(k)>=0)
 x(k)=exp(n(k));  else
 x(k)=0;
 end
 end
% disp(x);
 subplot(5,2,4)
 stem(n,x);
 xlabel('n -->');
 ylabel('amplitude');
 title('EXPONENTIAL
SIGNAL');

 %SINE WAVEFORM
 case 5
 n = [0:(pi/32):(4*pi)];
 x = sin(n);
```

```
% disp(x);
 subplot(5,2,5)
 stem(n,x);
 xlabel('n -->');
 ylabel('amplitude');
 title('SINE SIGNAL');


%COSINE WAVEFORM
case 6
n = [0:(pi/32):(4*pi)];
x = cos(n);
% disp(x);
subplot(5,2,6)
stem(n,x);
xlabel('n -->');
ylabel('amplitude');
title('COSINE SIGNAL');


%TRIANGULAR WAVEFORM
 case 7
 n = [0:0.2:20];
 x =sawtooth(n,0.5);
% disp(x);
subplot(5,2,7)
stem(n,x);
xlabel('n -->');
ylabel('amplitude');
title('TRIANGULAR SIGNAL');


%SAWTOOTH WAVEFORM
case 8
n = [0:0.2:20];
x=sawtooth(n,1);
% disp(x);
subplot(5,2,8)
stem(n,x);
xlabel('n -->');
ylabel('amplitude');
title('SAWTOOTH SIGNAL');


%RANDOM SIGNAL
case 9
r=10
x=rand(r,1)
% disp(x);
subplot(5,2,[9 10])
stem(x,'k')
xlabel('r->')
ylabel('X->')
title('RANDOM SIGNAL’)

 otherwise
```

```
 disp('INVALID CHOICE');
 end
 disp('Do you want to continue?');
 opt=input('If YES, press 1:');
end
```

RESULT:
Thus, the program to generate different waveforms using MATLAB is executed and the outputs are verified.

Expt. No. 2. AUTO CORRELATION and CROSS CORRELATION

Expt. No. 2a. AUTO CORRELATION

AIM:
To write a program to obtain auto correlation of the given sequence using MATLAB.

SOFTWARE REQUIRED:
MATLAB Software

ALGORITHM:
1. Start the program.
2. Give the input sequence.
3. Obtain the autocorrelation of the input sequence using the built in function, xcorr(x,x).
4. Display and plot the output.
5. Terminate the program.

FLOWCHART:

Start

Get the input sequence

Obtain Auto-Correlation of the input sequence using inbuilt function

Display and Plot the output signal

Stop

PROGRAM:
```
clc
clear all
close all
x=input('enter the input sequence x')
c=xcorr(x,x) %correlation using the function 'xcorr'
subplot(2,1,1)
```

```
stem(x)
xlabel('n')
ylabel('x(n)')
title('input x')
disp('auto correlated sequence')
disp(c)
subplot(2,1,2)
stem(c)
xlabel('n')
ylabel('c(n)')
title('auto correlated sequence')
```

RESULT:

Thus, the program to find the auto correlation of the given sequences using MATLAB is executed and the output is verified.

Expt. No. 2b. CROSS CORRELATION

AIM:

To write a program to obtain cross correlation of the given sequences using MATLAB.

SOFTWARE REQUIRED:

MATLAB Software

ALGORITHM:

1. Start the program.
2. Give the two input sequences.
3. Obtain the autocorrelation of the input sequence using the built in function, xcorr(x,x).
4. Display and plot the output.
5. Terminate the program.

FLOWCHART:

Start

Get the input sequence

Calculate the length of the sequences

Perform zero padding for the lesser length sequence to make the lengths equal

Obtain Cross-Correlation of the input sequence using inbuilt function

Display and Plot the output signal

Stop

PROGRAM:

```
clc
clear all
close all
x=input('enter the input sequence x')
y=input('enter the input sequence y')
m=length(x) %length of x
n=length(y) %length of x
if (m-n) ~= 0
 if m>n
 y=[y zeros(1,(m-n))] %append m-n number of zeros to the sequence ‘y’  n=m
 else
 x=[x zeros(1,(n-m))] %append n-m number of zeros to the sequence ‘x’  m=n
 end
end
c=xcorr(x,y) %correlation using the function ‘xcorr’
subplot(3,1,1)
stem(x)
xlabel('n')
ylabel('x(n)')
title('input x')
subplot(3,1,2)
stem(y)
xlabel('n')
ylabel('y(n)')
title('input y')
disp('cross correlated sequence')
disp(c)
subplot(3,1,3)
stem(c)
xlabel('n')
ylabel('c(n)')
title('cross correlated sequence')
```

RESULT:

Thus, the program to find the cross correlation of the given sequences using MATLAB is executed and the output is verified.

Expt. No. 3a. DFT & IDFT

AIM:

To write a program to find the Discrete Fourier Transform of the given sequence using MATLAB and plot the magnitude and phase response.

SOFTWARE REQUIRED:

MATLAB Software

ALGORITHM:

1. Clear the command window.
2. Get the input sequence x(n).

3. Get the N-point value.
4. If N>length of input sequence, pad zeros to input sequence.
5. For each value of X(k), compute temp=temp+x(n)*exp(-j*(2*pi/N)*(n-1)*(k-1))
6. Display DFT of the input sequence
7. Compute the magnitude of X(k) using the command
   mag_dft=abs(x_dft)
8. Compute the phase of X(k) using the command
   phase_dft=angle(x_dft)
9. Plot the input sequence, magnitude of X(k), and phase of X(k) in a single window.

FLOWCHART: close all;

Start

Get the input sequence x(n)

Get the N-point value

If N>length of input sequence,
pad zeros to input sequence.

Compute DFT using the formula
temp=temp+x(n)*exp(-j*(2*pi/N)*(
n-1)*(k 1))

PROGRAM: clc;
clear all;

Compute the magnitude of X(k) using the **'abs'** command

Compute the phase of X(k) using the **'angle'** command

Plot x(n), magnitude of X(k) and

phase of X(k) Stop

Display DFT

```
x=input('enter the
sequence'); N=input('enter
the length') if(N>length(x))
 x=[x zeros(1,(N-length(x)))]
end
for k=1:1:N
 X(k)=0;
for n=1:1:length(x)
 X(k)=X(k)+x(n)*exp(-j*(2*pi/N)*(n-1)*(k-1));
end
 disp(X(k))
end

subplot(3,1,1)
stem(x,'k')
xlabel('n->')
ylabel('amp->')
title('input')
mag_X=abs(X)
subplot(3,1,2)
stem(mag_X,'k')
xlabel('n->')
ylabel('amp->')
title('magnitude response')
phase_X=angle(X)
subplot(3,1,3)
stem(phase_X,'k')
xlabel('n->')
ylabel('amp->')
```

```
title('phase response')
for n=1:1:N
 y(n)=0;
 for k=1:1:length(X)
 y(n)=y(n)+(1/N)*X(k)*exp(j*(2*pi/N)*(n-1)*(k-1));
 end
 disp(y(n))
end
```

RESULT:

Thus, the program to find the Discrete Fourier Transform and IDFT of the given sequence using MATLAB is executed and the output is verified.

Expt. No. 3b. FFT and IFFT

AIM:

To write a program to find the FFT of the given sequence using MATLAB and plot the magnitude and phase response.

SOFTWARE REQUIRED:

MATLAB Software

ALGORITHM:

1. Clear command window.
2. Get the input sequence x(n).
3. Get the N-point value.
4. Compute FFT using x_fft=fft(x,n).
5. Display FFT of the input sequence
6. Compute the magnitude of X(k) using the command
   mag_fft=abs(x_fft)
7. Compute the phase of X(k) using the command
   phase_fft=angle(x_fft)
8. Plot the input sequence, magnitude of X(k), and phase of X(k) in a single window.

FLOWCHART:

Get the N-point value

Compute fft using fft command

Display FFT

Compute the magnitude of X(k) using the **'abs'** command

Compute the phase of X(k) using the **'angle'** command

Plot x(n), magnitude of X(k) and

phase of X(k) Stop

PROGRAM:

```
clc
clear all
close all
x=input('enter the sequence')
N=input('enter the length')
```

Start

Get the input sequence x(n)

```
X=fft(x) %FFT function
subplot(3,1,1)
stem(x,'k')
xlabel('time->')
ylabel('amp->')
title('input->')
mag_X=abs(X)
subplot(3,1,2)
stem(mag_X,'k')
xlabel('time->')
ylabel('amp->')
title('phase response->')
phase_X=angle(X)
subplot(3,1,3)
stem(phase_X,'k')
xlabel('time->')
ylabel('amp->')
title('magnitude response->')
y=ifft(X)
```

RESULT:

Thus, the program to find the FFT and IFFT of the given sequence using MATLAB is executed and the output is verified.

Expt. No. 4 Linear Convolution and Circular Convolution

Expt. No. 4a Linear Convolution

AIM:

To compute linear convolution of two sequences using in-built function in MATLAB.

ALGORITHM:

1. Clear command window.
2. Get the input sequence x(n).
3. Get the impulse response h(n).
4. Compute the linear convolution using conv(x,h) command.
5. Display the output.
6. Plot the input sequence, impulse response and output sequence in a single window.

FLOWCHART:

Start

Get the input sequence x(n)

Get the impulse response h(n)

Compute the linear convolution using conv(x,h)

Display the output y(n)

Stop

PROGRAM:

```
clc
clear all
close all
x=input('enter the input sequence')
h=input('eneter the impulse response')
l=length(x)+length(h)-1
y=conv(x,h)
subplot(3,1,1)
stem(x,'k')
xlabel('time->')
ylabel('amp->')
title('input->')
subplot(3,1,2)
stem(h,'k')
xlabel('time->')
ylabel('amp->')
title('impulse->')
subplot(3,1,3)
stem(y,'k')
xlabel('time->')
ylabel('amp->')
title('linear convolution->')
```

RESULT:

Thus, the program to find the linear convolution of two sequences using in-built function in MATLAB is executed and the output is verified

Expt. No. 4b Circular convolution using FFT

AIM:

To compute circular convolution of two sequences using FFT in MATLAB.

ALGORITHM:

1. Clear command window.
2. Get the two sequences x(n) and h(n) from user.
3. Calculate the length of sequences len_x, len_h and find the maximum value. 4.
Compute X(k) using **fft** command by specifying fft length as maximum length of x and h. 5.
Compute H(k) using **fft** command by specifying fft length as maximum length of x and h. 6.

Multiply the two fft sequences element by element and store in y_fft . 7. Calculate inverse FFT of y_fft using **ifft** command and store it in y.
8. Display the output.
9. Plot the two input sequences and the output sequence.

FLOWCHART:

Get the two sequences
x(n) and h(n)

Calculate the length of sequences
len_x, len_h

Find len_y as the maximum value of
len_x and len_h

Compute X(k) using **fft(x,len_y)**

Compute H(k) using **fft(h,len_y)**

Compute Y(k) = X(k).*H(k)

Compute y(n) using **ifft(Y(k),len_y)**

PROGRAM: clc
clear all
close all

Start

Stop

Plot x(n),h(n) and y(n)

```
x=input('enter then input sequence')
h=input('enter the impulse
response') l1=length(x)
l2=length(h)
l3=max(l1,l2)
X=fft(x)
H=fft(h)
for i=1:1:l3
 Y(i)=X(i)*H(i);
end
y=ifft(Y)
subplot(3,1,1)
```

```
stem(x,'k')
xlabel('n->')
ylabel('amp->')
title('input')
subplot(3,1,2)
stem(h,'k')
xlabel('n->')
ylabel('amp->')
title('impulse response')
subplot(3,1,3)
stem(y,'k')
xlabel('n->')
ylabel('amp->')
title('circular convolution using fft')
```

RESULT:

Thus, the program to find the circular convolution of two sequences using FFT in MATLAB is executed and the output is verified

Expt. No. 4c Circular Convolution using in-built function

AIM:

To compute the circular convolution of two sequences using in-built function in MATLAB.

ALGORITHM:

1. Clear command window.
2. Get the two sequences x(n) and h(n) from user.
3. Calculate the length of sequences len_x, len_h and find the maximum value. 4. Perform the circular convolution using 'cconv' function by specifying output length as maximum length of x and h.
5. Display the output.
6. Plot the two input sequences and the output sequence.

FLOWCHART:

Start

Get the two sequences
x(n) and h(n)

Calculate the length of sequences as len_x, len_h

Find len_y as the maximum value of len_x and
len_h

Compute y(n) using **cconv(x,h,len_y)**

Plot x(n),h(n) and y(n)

Stop

PROGRAM:

```
clc
clear all
close all
x=input('enter the input sequence')
i=input('enter the impulse response')
l1=length(x)
l2=length(i)
l3=max(l1,l2)
y=cconv(x,i,l3)
subplot(3,1,1)
stem(x,'k')
xlabel('n->')
ylabel('amp->')
title('input')
subplot(3,1,2)
stem(i,'k')
xlabel('n->')
ylabel('amp->')
title('impulse response')
subplot(3,1,3)
stem(y,'k')
xlabel('n->')
ylabel('amp->')
title('circular convolution')
```

RESULT:

Thus, the program to find the circular convolution of two sequences using using in-built function in MATLAB is executed and the output is verified