Digital Signal Processing Lab

Laboratory report submitted for the partial fulfillment
of the requirements for the degree of

Bachelor of Technology
in
Electronics and Communication Engineering

by

Student-Ojasav
Roll No.-21UCC071

Course Coordinator
Dr. Joyeeta Singha

Department of Electronics and Communication Engineering
The LNM Institute of Information Technology, Jaipur

September 2022

Chapter 1

Experiment - 4

1.1 Objective of the Experiment

a) Circular convolution and DFT Multiplication for two sequences. b) Simulink based circular convolution.

1.2 Theory

a) Circular Convolution and DFT Multiplication for Two Sequences:

a.1) Circular Convolution: Circular convolution is a mathematical operation used in signal processing to combine two sequences using circular boundary conditions. It's often denoted as , and for two sequences x[n] and h[n], their circular convolution y[n] is defined as:

$$y[n] = \sum_{k=0}^{N-1} x[k] \cdot h[(n-k) \; mod \; N]$$

Where: - N is the length of the sequences. - $x[n]$ and $h[n]$ are the input sequences. - $y[n]$ is the circularly convolved output sequence.

Circular convolution is useful in scenarios where signal periodicity is involved, and it's often implemented using techniques like the Discrete Fourier Transform (DFT).

a.2) DFT Multiplication: The Discrete Fourier Transform (DFT) is a mathematical transformation used to analyze and manipulate signals in the frequency domain. When you multiply two sequences $X[k]$ and $H[k]$ in the frequency domain, you are effectively performing a convolution operation in the time domain. This is known as DFT multiplication, and it's a key concept in fast convolution algorithms like the Fast Fourier Transform (FFT).

The DFT multiplication can be expressed as follows:

$$Y[k] = X[k] \cdot H[k]$$

Where: - $X[k]$ and $H[k]$ are the DFTs of the input sequences $x[n]$ and $h[n]$. - $Y[k]$ is the DFT of the circularly convolved output $y[n]$.

The circular convolution and DFT multiplication are related through the Convolution Theorem, which states that convolution in the time domain is equivalent to pointwise multiplication in the frequency domain.

b) Simulink-Based Circular Convolution:

Simulink is a graphical programming environment commonly used for modeling, simulating, and analyzing dynamic systems and processes, including signal processing operations. To implement circular convolution in Simulink, you can use blocks that represent key components of the operation. Here's a high-level outline of how you might set up a Simulink-based circular convolution:

1. Input Blocks: Add blocks to represent the input sequences $x[n]$ and $h[n]$. These could be, for example, step input blocks or signal sources.

2. Circular Shifting: Implement circular shifting of one of the input sequences. You can use Simulink's built-in blocks for this purpose. Circular shifting ensures that the sequences align properly for circular convolution.

3. Multiplier Block: Multiply the shifted sequence with the other input sequence using a multiplier block. This represents the pointwise multiplication in the time domain.

4. Summation Block: Sum the results from the multiplier block to obtain the circular convolution output.

5. Output Block: Add an output block to observe or analyze the circularly convolved signal.

Software uesd is MATLAB

1.3 Functions in Matlab

```
function [H] = myCirConvMat(h,lmax)
      H=zeros(lmax,lmax);
      j=1;
      k=0;
      for i=1:lmax
            temp=circshift(h,k);
            H(:,i)=temp;
            k=k+1;
      end
```

```
Not enough input arguments.

Error in myCirConvMat (line 2)
      H=zeros(lmax,lmax);
```

Published with MATLAB® R2023b

```
function [D] = myDFT(N)

D = zeros(N,N);

for i = 1:N
    for k = 1:N
        D(i,k) = exp(-1j*2*pi*(i-1)*(k-1)/N);
    end
end

end
```

```
Not enough input arguments.

Error in myDFT (line 3)
D = zeros(N,N);
```

Published with MATLAB® R2023b

```
function [D] = myIDFT(N)

D = zeros(N,N);

for i = 1:N
    for k = 1:N
        D(i,k) = exp(1j*2*pi*(i-1)*(k-1)/N);
    end
end

end
```

```
Not enough input arguments.

Error in myIDFT (line 3)
D = zeros(N,N);
```

Published with MATLAB® R2023b

1.4 Task1

```
clc;
clear all;
close all;
h = [1,1,1,1,2,2,2,2];
x = [1 2 3 4 5 6 7 8];
lh = length(h);
lx = length(x);
l  = max(lh,lx);
N = length(h);
D = myDFT(N);
H =myCirConvMat(h,l);
Y = H*x';
D8 = myDFT(N);
h_k = D8*h';
x_k = D8*x';
y_k = x_k.*h_k;
D08 = myIDFT(N);
Y1 = (1/N)*(D08*y_k);
D_inv = inv(D);
Hf = D*H*D_inv;
display(Hf);
Xf = D*x';
Yf = Hf*Xf;
yf = D_inv*Yf;
display(yf);
display(abs(yf));
```

```
Hf =

  Columns 1 through 4

  12.0000 - 0.0000i   0.0000 - 0.0000i  -0.0000 + 0.0000i   0.0000 - 0.0000i
  -0.0000 + 0.0000i  -1.0000 + 2.4142i   0.0000 + 0.0000i  -0.0000 - 0.0000i
  -0.0000 - 0.0000i   0.0000 - 0.0000i   0.0000 - 0.0000i  -0.0000 + 0.0000i
  -0.0000 + 0.0000i   0.0000 + 0.0000i  -0.0000 - 0.0000i  -1.0000 + 0.4142i
  -0.0000 - 0.0000i   0.0000 - 0.0000i   0.0000 + 0.0000i   0.0000 - 0.0000i
  -0.0000 - 0.0000i  -0.0000 - 0.0000i   0.0000 - 0.0000i   0.0000 - 0.0000i
  -0.0000 - 0.0000i  -0.0000 - 0.0000i  -0.0000 + 0.0000i  -0.0000 - 0.0000i
   0.0000 + 0.0000i  -0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i

  Columns 5 through 8

   0.0000 - 0.0000i   0.0000 - 0.0000i  -0.0000 + 0.0000i  -0.0000 + 0.0000i
  -0.0000 + 0.0000i  -0.0000 - 0.0000i   0.0000 - 0.0000i  -0.0000 + 0.0000i
   0.0000 + 0.0000i  -0.0000 + 0.0000i  -0.0000 - 0.0000i  -0.0000 + 0.0000i
   0.0000 - 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i  -0.0000 + 0.0000i
  -0.0000 - 0.0000i  -0.0000 + 0.0000i  -0.0000 + 0.0000i  -0.0000 + 0.0000i
  -0.0000 - 0.0000i  -1.0000 - 0.4142i   0.0000 - 0.0000i  -0.0000 - 0.0000i
   0.0000 - 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i  -0.0000 - 0.0000i
   0.0000 - 0.0000i  -0.0000 - 0.0000i  -0.0000 - 0.0000i  -1.0000 - 2.4142i
```

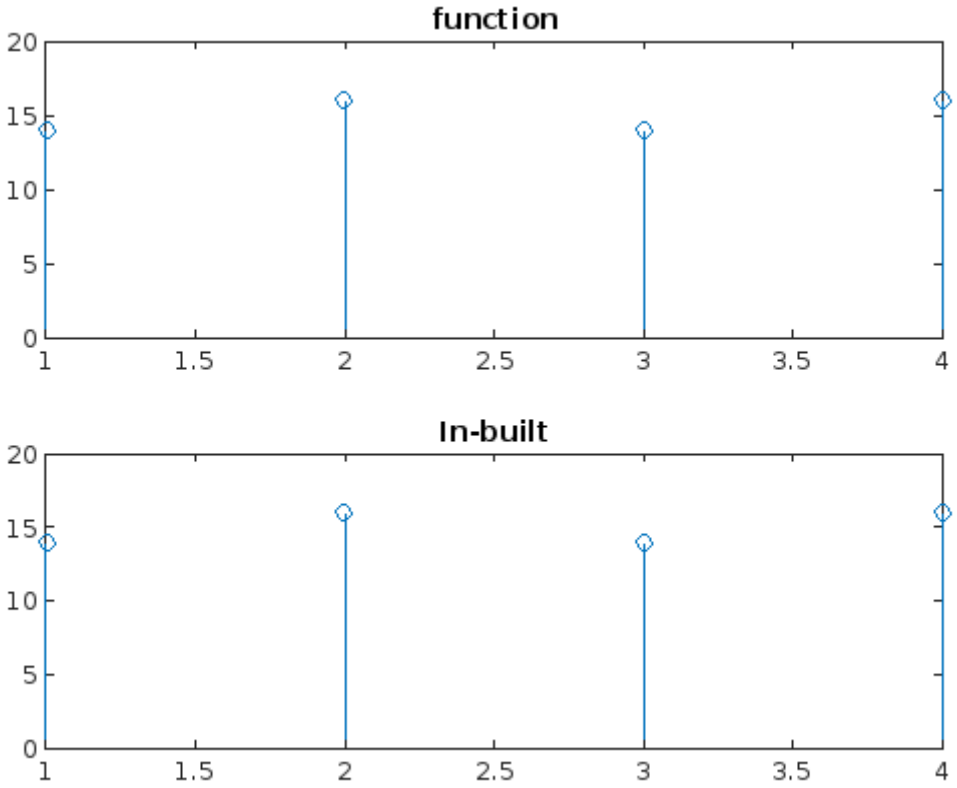
```
yf =

  50.0000 - 0.0000i
  54.0000 + 0.0000i
  58.0000 + 0.0000i
  62.0000 - 0.0000i
  58.0000 + 0.0000i
  54.0000 - 0.0000i
  50.0000 - 0.0000i
  46.0000 - 0.0000i

   50.0000
   54.0000
   58.0000
   62.0000
   58.0000
   54.0000
   50.0000
   46.0000
```

Published with MATLAB® R2023b

1.5 Task 2

```
clc;
clear all;
close all;
h=[2,1,2,1];
x=[1,2,3,4];
lh=length(h);
lx=length(x);
lmax=max(lh,lx);
h=[h,zeros(1,lmax-lh)];
x=[x,zeros(1,lmax-lx)];
H=myCirConvMat(h,lmax);
display(H);

y=H*x';
display(y');
subplot(2,1,1);
stem(y);
title("function");
subplot(2,1,2);
stem(cconv(h,x,lmax));
title("In-built");
```

```
H =

     2     1     2     1
     1     2     1     2
     2     1     2     1
     1     2     1     2

    14    16    14    16
```

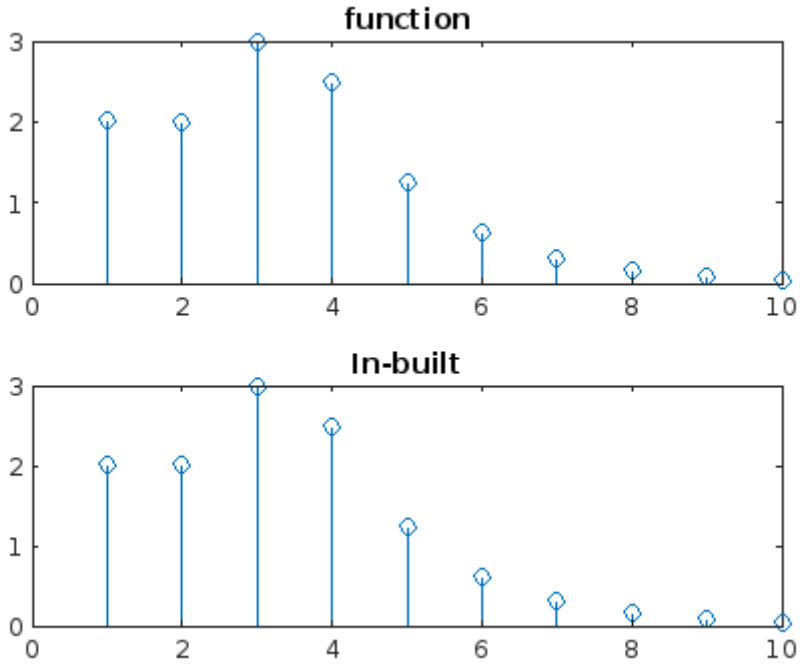


Published with MATLAB® R2023b

1.6 Task 3

```
clc;
clear all;
close all;
h=[2,1,2,1];
n=0:9;
x=0.5.^n;
lh=length(h);
lx=length(x);
lmax=max(lh,lx);
h=[h,zeros(1,lmax-lh)];
x=[x,zeros(1,lmax-lx)];
H=myCirConvMat(h,lmax);
display(H);
y=H*x';
display(y');
subplot(2,1,1);
stem(y);
title("function");
subplot(2,1,2);
stem(cconv(h,x,lmax));
title("In-built");
```

```
H =

     2     0     0     0     0     0     0     1     2     1
     1     2     0     0     0     0     0     0     1     2
     2     1     2     0     0     0     0     0     0     1
     1     2     1     2     0     0     0     0     0     0
     0     1     2     1     2     0     0     0     0     0
     0     0     1     2     1     2     0     0     0     0
     0     0     0     1     2     1     2     0     0     0
     0     0     0     0     1     2     1     2     0     0
     0     0     0     0     0     1     2     1     2     0
     0     0     0     0     0     0     1     2     1     2

  Columns 1 through 7

    2.0176    2.0078    3.0020    2.5000    1.2500    0.6250    0.3125

  Columns 8 through 10

    0.1562    0.0781    0.0391
```

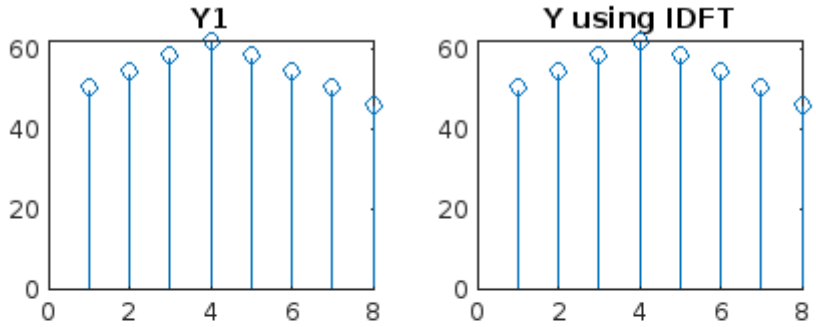


Published with MATLAB® R2023b

1.7 Task 4

```
clc;
clear all;
close all;
h = [1,1,1,1,2,2,2,2];
x = [1 2 3 4 5 6 7 8];
lh = length(h);
lx = length(x);
l  = max(lh,lx);
N = length(h);
D = myDFT(N);
H =myCirConvMat(h,l);
Y = H*x';
D8 = myDFT(N);
h_k = D8*h';
x_k = D8*x';
y_k = x_k.*h_k;
D08 = myIDFT(N);
Y1 = (1/N)*(D08*y_k);
subplot(2,2,1)
stem(Y);
title("Y1");
subplot(2,2,2)
stem(Y1);
title("Y using IDFT");
```

Warning: Using only the real component of complex data.



Published with MATLAB® R2023b

1.8 Conclusion

The output of all the comparison is same and we obtain similar graphs for pre-defined function and inbuilt function. Hence, we can say that relation between circular convolution and DFT matrix multiplication method is valid.