EXPERIMENT 11

Linear and Circular Convolution on TMS320C6713 and MATLAB

Objective

To study the effects of convolution and correlation of two discrete-time signals and observe the difference between the two. Moreover, you will implement the linear and circular convolution using code composer studio on TMS320C6713 and MATLAB.

- **Part 1: Implementation on MATLAB**

Convolution is a mathematical way of combining two signals to form a third signal. The definition of convolution in discrete time domain is given by:

$$\boldsymbol{y[n] = x[n] * h[n] = \sum_{k=-\infty}^{k=+\infty} x[k]\,.\;h[n-k]}$$

Exercise 1:

Write a generic function for the convolution of two sequences which takes the sequences and their indices as input and plots the convolved sequence.

```
function [y,n]=(x1,x2,n1,n2)
```

Exercise 2:

Prove the convolution property stated below:

$$\boldsymbol{y(n) = x(n) * h(n) \rightarrow \ Y\left(e^{j\omega}\right) = X\left(e^{j\omega}\right) . H(e^{j\omega})}$$

Convolution operation can also be implemented using Fourier domain multiplication and the inverse transform afterwards. In fact, this property makes analysis of LTI systems easier in Fourier domain than in time domain.

Show, by plotting the spectra $Y\left(e^{j\omega}\right)$and $H\left(e^{j\omega}\right)$.* $X\left(e^{j\omega}\right)$, that these spectra, obtained from the signals $\boldsymbol{y(n) = conv(x(n), x(n))}$ and $x(n)$=[1 1 1] for n = [0 1 2] are the same. Attach the screen shots of the spectra $Y\left(e^{j\omega}\right)$ and the signal $\boldsymbol{y(n)}$ in the space provided below:

Correlation:

The correlation between the sequences is given by:

$$\boldsymbol{r_{xy}(l) = \sum_{n=-\infty}^{n=+\infty} x(n)y(n-l),\ \ l = 0, \pm 1, \pm 2, \pm 3, \ldots..}$$

$$\boldsymbol{r_{xy}(l) = \sum_{n=-\infty}^{n=+\infty} x(n-l)y(n),\ \ l = 0, \pm 1, \pm 2, \pm 3, \ldots..}$$

Similarly, the auto-correlation is given by

$$\boldsymbol{r_{xx}(l) = \sum_{n=-\infty}^{n=+\infty} x(n)x(n-l) = r_{xx}(-l) \qquad l = 0, \pm 1, \pm 2, \pm 3, \ldots}$$

<u>Exercise 3</u>:

Write a MATLAB code to find the auto-correlation of the sequence $x(n) = [1\ 2\ 3\ 4]$

I. Plot the original and correlated sequence.
II. Prove the property that $|r_{xx}(l)| \leq r_{xx}(0) = E_x$
III. Check if the auto-correlated sequence is $r_{xx}(l)$ an even function or not

Hint: (Use "xcorr()" function in MATLAB for the correlation)

Attach the plot of the original and correlated sequence in the space provided below:

Exercise 4:

Write a MATLAB code to find the cross-correlation of the sequences $x_1(n) = [1\ 2\ 3\ 4]$and $x_2(n) = [4\ 3\ 2\ 1]$

I. Plot the original and correlated sequence.
II. Prove the property that $r_{xy}\ (l) = \ r_{yx}(-l)$ and attach the graphs in the space provided below:

III. Also show that the maximum value in cross-correlated sequence is the $sqrt\ (E_X E_Y)$ that is

$$\left|r_{xy}(l)\right| \leq \sqrt{r_{xx}(0) r_{yy}(0)} = \sqrt{E_x E_y}$$

Attach the screen shot of the maximum value in the space provided below:

- **Part 2: Implementation on TMS320C6713**

Convolution is important because it relates the three signals of interest: the input signal, the output signal, and the impulse response.

Procedure

- Open code composer studio and select C6713 as simulator.
- Make a new project using 'Project→New pull down menu and save it in a separate directory with file name linearconv.pjt.
- Create a new source file named as "linear.c" using File→New→Sourcefile and save it to the project folder.
- Add source file to the project using Project→Add files to the project→linear.c menu.
- Add the linker command file hello.cmd.
    - (path: C:\CCstudio\tutorial\dsk6713\hello\hello.cmd)
- Add the run time support library file rts6700.lib
    - (path: C\CCStudio\cgtools\lib\rts6700.lib)
- Compile the program using project →Compile menu or by Ctrl+F7
- Build the program using project→ Build menu or by F7
- Load the linear.out file (from project folder linearconv\Debug) using File →Load Program
- Run the program using Debug→ Run or F5
- To view the output graphically
- Select View→ Graph→ Time and Frequency
- Repeat the steps 2 to 11 for circular convolution

Program:

Linear Convolution

```
#include<stdio.h>
#define LENGTH1 6
#define LENGTH2 4

int x[LENGTH1+LENGTH2-1]={4,3,2,1,5,6,0,0,0};
int h[LENGTH1+LENGTH2-1]={1,2,1,4,0,0,0,0,0};
int y[LENGTH1+LENGTH2-1];
main()
{
inti=0,j;
for(i=0;i<LENGTH1 + LENGTH2-1;i++)
```

```
{
y[i]=0;
for(j=0;j<=i;j++)
y[i]+=x[j]*h[i-j];
}
for(i=0;i<LENGTH1 + LENGTH2-1;i++)
printf("%d\n",y[i]);
}
```

RESULT

The convoluted sequence is obtained as

4 11 12 24 21 25 21 26 24

Circular convolution

```
#include<stdio.h>
intm,n,x[30],h[30],y[30],i,j,temp[30],k,x2[30],a[30];
void main()
{
printf("Enter the length of 1st sequence\n");
scanf("%d",&m);
printf("Enter the length of 2nd sequence\n");
scanf("%d",&n);
printf("Enter the 1st sequence\n");
for(i=0;i<m;i++)
scanf("%d",&x[i]);
printf("Enter the 2nd sequence\n");
for(j=0;j<n;j++)
scanf("%d",&h[j]);
if(m-n!=0)
{
if(m>n)
{
for(i=n;i<m;i++)
h(i)=0;
n=m;
}
for(i=m;i<n;i++)
x[i]=0;
m=n;
}
y[0]=0;
a[0]=h[0];
for(j=1;j<n;j++)
a[j]=h[n-j];
```

```
for(i=0;i<n;i++)
y[0]+=x[i]*a[i];

for(k=1;k<n;k++)
{
y[k]=0;
for(j=1;j<n;j++)
x2[j]=a[j-1];
x2[0]=a[n-1];
for(i=0;i<n;i++)
{
a[i]=x2[i];
y[k]+=x[i]*x2[i];
}
}
printf("The circular convolution is \n");
for(i=0;i<n;i++)
printf("%d\t",y[i]);
}
```

RESULT

Enter the length of 1st sequence 4
Enter the length of 2nd sequence 4
Enter the 1st sequence 1 2 3 1
Enter the 2nd sequence 4 3 2 2
The circular convolution is 17 19 22 19

Exercise 1:

Modify the code for circular convolution to give correct results even if the length of the two sequences is different such that you need to enter the length of both the sequences.

Write the modified part of the code in the space provided below:

Post Lab:

Write a code in MATLAB for performing the convolution of two signals. You are not allowed to use any built in function for performing the convolution. Attach the screen shot of the convolved sequence below: