"Heaven's light is our guide"

Rajshahi University of Engineering & Technology

Dept. of Electrical & Computer Engineering

Course No: ECE 4124

Course Title: Digital Signal Processing Sessional

Submitted by:

Name: Adnan Sami

Roll: 1810035

Department: ECE

RUET

Submitted to:

Hafsa Binte Kibria

Lecturer

Dept. of ECE, RUET

Index

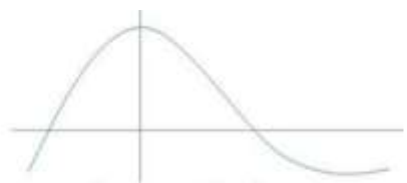
Experiment Date: 20 March, 2023

Experiment No: 01

Name of the Experiment:

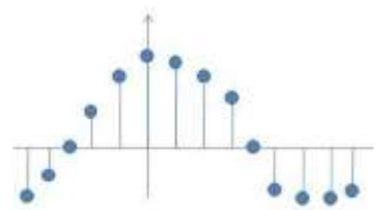
i) Plot unit step, unit ramp, unit impulse signal using MATLAB.
ii) Plot discrete signal using MATLAB.
iii) Plot two different discrete signal and show their addition and subtraction using MATLAB.
iv) Plot two different continuous signal using MATALB.

Theory: A continuous signal or a continuous-time signal is a varying quantity whose domain, which is often time, is a continuum. That is, the function's domain is an uncountable set. The function itself need not to be continuous.



Continuous Signal

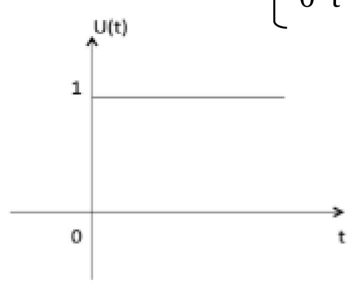
A discrete signal or discrete-time signal is a time series consisting of a sequence of quantities. Unlike a continuous-time signal, a discrete-time signal is not a function of a continuous argument; however, it may have been obtained by sampling from a continuous-time signal.



Discrete Signal

If a step signal has unity magnitude, then it is known as unit step signal or unit step function. It is denoted by u(t). The step signal is equivalent to applying a signal to a system whose magnitude suddenly changes and remains constant forever after application.

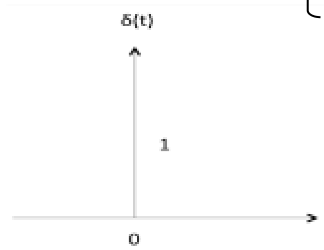
Unit step function is denoted by u(t). It is defined as $u(t) = \begin{cases} 1 & t \geq 0 \\ 0 & t < 0 \end{cases}$



- It is used as best test signal.
- Area under unit step function is unity.

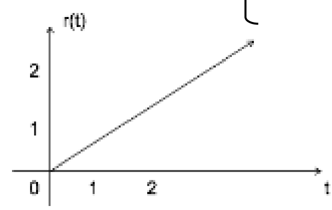
An ideal impulse signal is a signal that is zero everywhere but at the origin (t = 0), it is infinitely high. Although, the area of the impulse is finite. The unit impulse signal is the most widely used standard signal used in the analysis of signals and systems.

Impulse function is denoted by δ(t). and it is defined as $\delta(t) = \begin{cases} 1 & t = 0 \\ 0 & t \neq 0 \end{cases}$



A ramp function or ramp signal is a type of standard signal which starts at $t = 0$ and increases linearly with time. The unit ramp function has unit slop.

Ramp signal is denoted by r(t), and it is defined as $r(t) = \begin{cases} t & t \geq 0 \\ 0 & t < 0 \end{cases}$



<u>Code:</u>

Code for plotting unit step, unit ramp, unit impulse signal:

```
clc
clear all
close all
t = -10:0.01:10
step = t>=0
subplot(3,1,1);
plot(t,step);
xlabel('time');
ylabel('amplitude');
title('unit step');
ramp = t.*step
subplot(3,1,2);
plot(t,ramp);
xlabel('time');
ylabel('amplitude');
title('unit ramp');
impulse = t==0
subplot(3,1,3);
plot(t,impulse);
xlabel('time');
ylabel('amplitude');
title('unit impulse');
```

Code for plotting discrete signal:

```
x = -4:3
y = [1,2,3,3,2,1,4,1]
stem(x,y);
```

Code for plotting two different discrete signal and show their addition and subtraction:

```
clc
clear all
close all
t = -10:2:20
n1 = t>=0 & t<=10
subplot(4,1,1);
stem(t,n1);
xlabel('time');
ylabel('amplitude');
title('1st Signal');
n2 = t>=5 & t<=15
subplot(4,1,2);
stem(t,n2);
xlabel('time');
ylabel('amplitude');
title('2nd Signal');
add = n1+n2
subplot(4,1,3);
stem(t,add);
xlabel('time');
ylabel('amplitude');
title('Addition');
sub = n1-n2
subplot(4,1,4);
stem(t,sub);
xlabel('time');
ylabel('amplitude');
title('Subtraction');
```

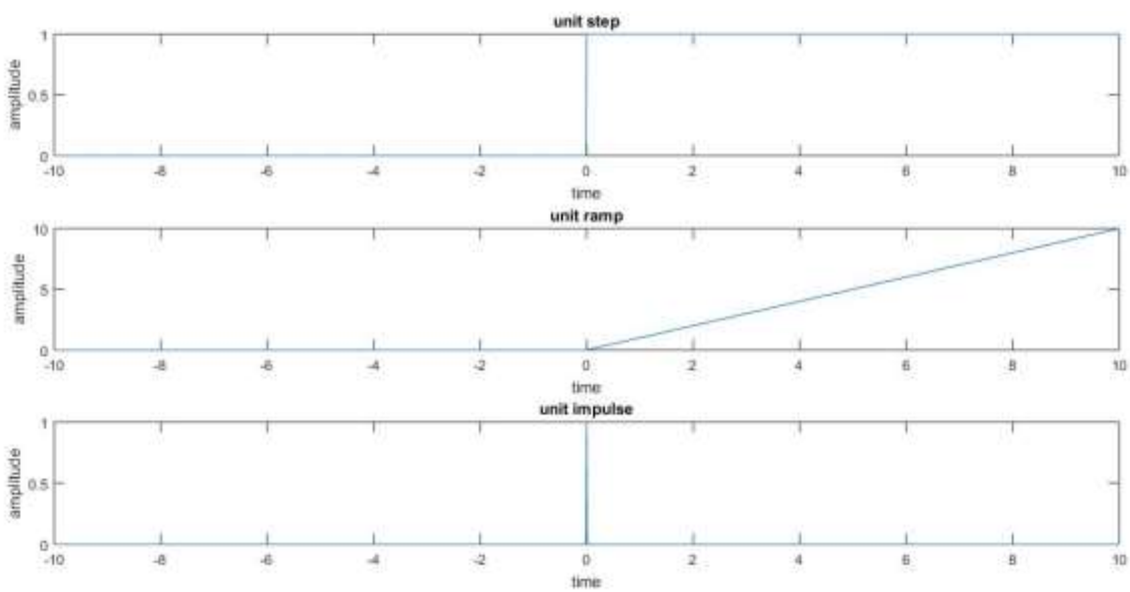
Code for plotting two different continuous signal:

```
clc
clear all
close all
t = -10:0.01:20
n1 = t>=0 & t<=7
n2 = t>=1 & t<=6
n3 = t>=2 & t<=5
s1 = n1+n2+2*n3;
```

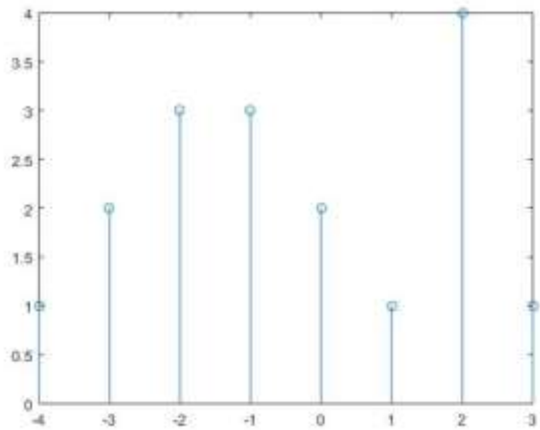
```
subplot(4,1,1);
    plot(t,s1);
    xlabel('time');
    ylabel('amplitude');
    title('1st signal');
    t1 = -3:0.01:3
    impulse = t1>=0
    n11 = t1.*impulse
    n22 = t1>=3 & t1<=5
    s2 = n11+n22;
    subplot(4,1,2);
    plot(t1,s2);
    xlabel('time');
    ylabel('amplitude');
    title('2nd signal');
```

<u>Output:</u>

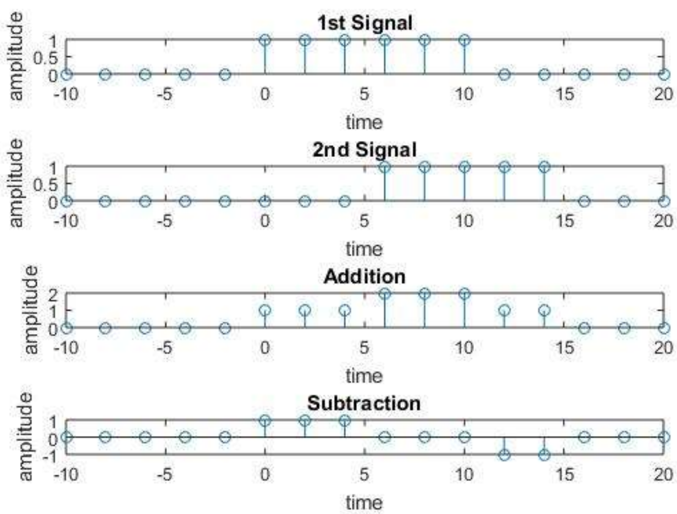
Output for plotting unit step, unit ramp, unit impulse signal:



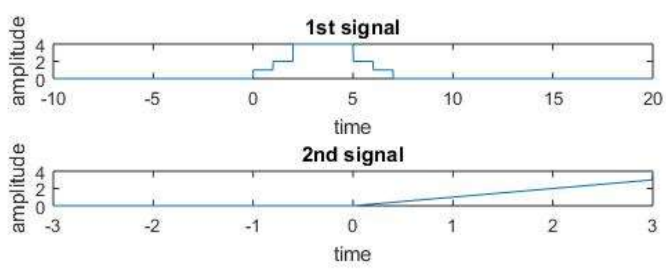
Output for plotting discrete signal:



Output for plotting two different discrete signal and show their addition and subtraction:



Output for plotting two different continuous signal:



Discussion: For plotting different types of signal the code was implemented using MATLAB. For plotting unit step signal a range t was declared. Unit step signal will be t greater than zero which was declared. Then plot and subplot was used to show the unit step signal. For plotting ramp signal, at first unit step signal was declared. Then t.*step was written for ramp signal. Using plot and subplot to show the ramp signal. To plot impulse signal, at first a range t was declared. Then t == 0 was written for impulse signal. This impulse signal was shown using plot and subplot.

For plotting discrete signal, a range was declared in x and value was declared in y. “stem” function was used to show the discrete signal.

For addition and subtraction of two signal, at first a range t was declared. Then the signal was plotted using plot and subplot according to the condition. Similarly, the second signal was plotted. For addition of two signal, the signals were added and for subtraction, the signals were subtracted. The final signal was shown using plot and subplot.

In the last two signal, one signal was looking like a stair and the other signal had three part. One part was looking like a ramp signal, other part was straight line which was parallel to base and another part was decreasing ramp signal. To implement the first code a range t was declared. Then wrote the condition for different part of the signal and added that condition to get the final signal. Here plot and subplot was used to show the signal. In the second code, I plotted only the ramp signal part according to the condition. I tried it but I could not complete and I failed to show the whole signal.

Conclusion: The code was executed successfully and no errors were found. Form this experiment, we had learned about different types of signal and how to plot them using MATALB.

References:

[1]"Continuous Signal – ScienceDirect", ScienceDirect, 2023. [Online].
Available: https://www.sciencedirect.com/topics/engineering/continuous-signal
[Accessed: 20 - April - 2023]

[2]"Discrete Signal – ScienceDirect", ScienceDirect, 2023. [Online].
Available: https://www.sciencedirect.com/topics/engineering/discrete-signal
[Accessed: 20 - April - 2023]

[3]"Unit Step Function – tutorialspoint", tutorialspoint, 2023. [Online].
Available: https://www.tutorialspoint.com/signals_and_systems/signals_basic_types.htm
[Accessed: 20 - April - 2023]

[4]"Unit Impulse Function – tutorialspoint", tutorialspoint, 2023. [Online].
Available: https://www.tutorialspoint.com/signals_and_systems/signals_basic_types.htm
[Accessed: 20 - April - 2023]

[5]"Ramp Signal – tutorialspoint", tutorialspoint, 2023. [Online].
Available: https://www.tutorialspoint.com/signals_and_systems/signals_basic_types.htm
[Accessed: 20 - April - 2023]

Experiment Date: 30 April, 2023

Experiment No: 02

Name of the Experiment: Write a code for linear convolution and plot the signal using MATLAB.

Theory: Convolution is a mathematical way of combining two signals to form a third signal. It is the single most important technique in Digital Signal Processing. Using the strategy of impulse decomposition, systems are described by a signal called the impulse response.

In linear systems, convolution is used to describe the relationship between three signals of interest: the input signal, the impulse response, and the output signal. If, x(n) is a M- point sequence h(n) is a N – point sequence then, y(n) is a (M+N-1) – point sequence.

If the input and impulse response of a system are x[n] and h[n] respectively, the convolution is given by the expression,

$x[n] * h[n] = \varepsilon\, x[k]\, h[n-k]$
Where k ranges between $-\infty$ and ∞

If,
x(n) is a M- point sequence h(n) is a N – point sequence then, y(n) is a (M+N-1) – point sequence.
In this equation, x(k), h(n-k) and y(n) represent the input to and output from the system at time n. Here we could see that one of the inputs is shifted in time by a value every time it is multiplied with the other input signal. Linear Convolution is quite often used as a method of implementing filters of various types.

Code:

Code for linear convolution:

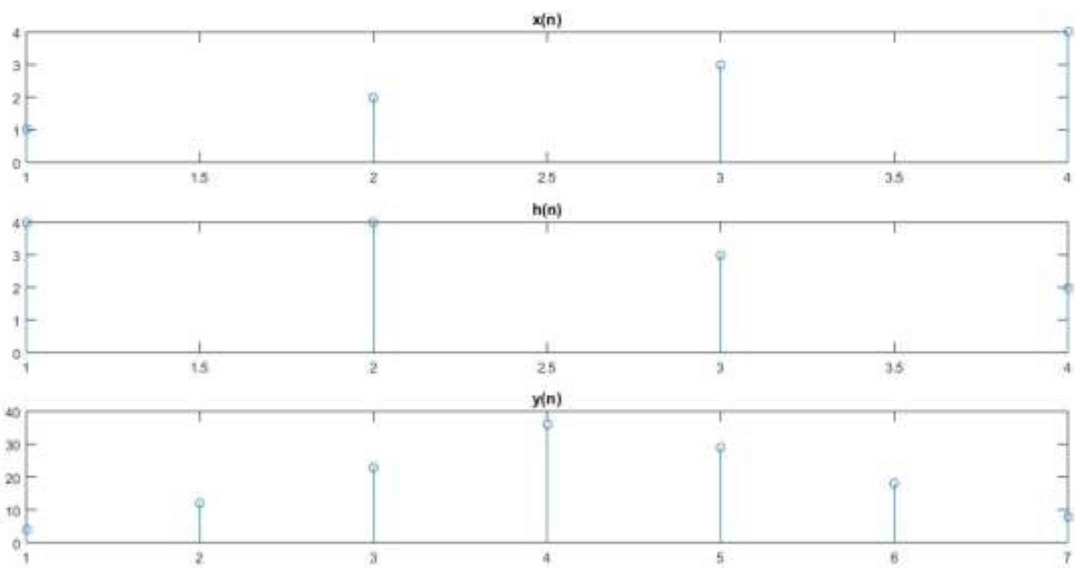
```
clc
clear all
close all
xn = [1 2 3 4];
hn = [4 4 3 2];
L = length(xn);
M = length(hn);
X = [xn, zeros(1,L)];
H = [hn, zeros(1,M)];
for n = 1 : L+M-1
    y(n)=0;
    for i = 1 : L
        if(n-i+1>0)
        y(n) = y(n)+X(i)*H(n-i+1)
        %s (n) = H(n-i+1);
        end
    end
end
subplot(3,1,1)
```

```
stem (xn)
title('x(n)')
subplot(3,1,2)
stem (hn)
title('h(n)')
subplot(3,1,3)
stem (y)
title('y(n)')
```

<u>Output:</u>

Output for linear convolution:

```
y =

     4    12    23    36    29    18     8
```



<u>Discussion:</u> Linear convolution code was implemented using MATLAB. For implementing the code, two 1×4 matrix was declared. Then length function was used for the length of the matrix. After that, a nested for loop was used. Here the first for loop was run from 1 to L+M-1. In the second for loop a condition was applied using if condition. Through the if condition a formula was written for the output. After the for loop, I wrote some code for plotting these signal. Here subplot was used. As the signal was discrete, I used stem function to plot the signal.

<u>Conclusion:</u> The code was executed successfully and no errors were found. Form this experiment, we had learned about linear convolution and how to plot the signal using MATALB.

References:

[1]"Convolution – Wikipedia", Wikipedia, 2023. [Online].
Available: https://en.wikipedia.org/wiki/Convolution
[Accessed: 03 - May - 2023]

[2]"Linear Convolution – PantechSolutions", PantechSolutions, 2023. [Online].
Available: https://www.pantechsolutions.net/linear-convolution-using-tms320c6745
[Accessed: 03 - May - 2023]

Experiment Date: 07 May, 2023

Experiment No: 03

Name of the Experiment: Write a code for auto correlation and cross-correlation and plot the signal using MATLAB.

Theory: The correlation of two functions or signals or waveforms is defined as the measure of similarity between those signals. There are two types of correlations. They are

1. Autocorrelation
2. Cross-correlation

The autocorrelation function is defined as the measure of similarity or coherence between a signal and its time delayed version. Therefore, the autocorrelation is the correlation of a signal with itself. Autocorrelation is useful for finding repeating patterns in a signal, such as determining the presence of a periodic signal which has been buried under noise or identifying the missing fundamental frequency in a signal implied by its harmonic frequencies.

The cross-correlation between two different signals or functions or waveforms is defined as the measure of similarity or coherence between one signal and the time-delayed version of another signal. The cross-correlation between two different signals indicates the degree of relatedness between one signal and the time-delayed version of another signal. In signal processing, cross-correlation is a measure of similarity of two series as a function of the displacement of one relative to the other. This is also known as a sliding dot product or sliding inner-product. It is commonly used for searching a long signal for a shorter, known feature.

Correlation coefficient is a measure of how well two signals are related to each other in terms of their similarity, alignment, and dependence. It is widely used in digital signal processing (DSP) for various purposes, such as signal detection, estimation, compression, enhancement, and classification.

Code:

Code for autocorrelation:

```
clc
clear all
close all
x1 = [1 2 3 4];
h1 = [4 3 2 1];
n = length(x1);
m = length(h1);
k = n+m-1;
x = [x1 zeros(1,k-n)]';
h = [h1 zeros(1,k-m)]';
for i=1:k
    c(:,i) = circshift(x,i-1)
end
y = c*h

```

```
subplot(3,1,1)
stem (x1)
title('x(1)')
subplot(3,1,2)
stem (h1)
title('h(1)')
subplot(3,1,3)
stem (y)
title('y(n)')
```

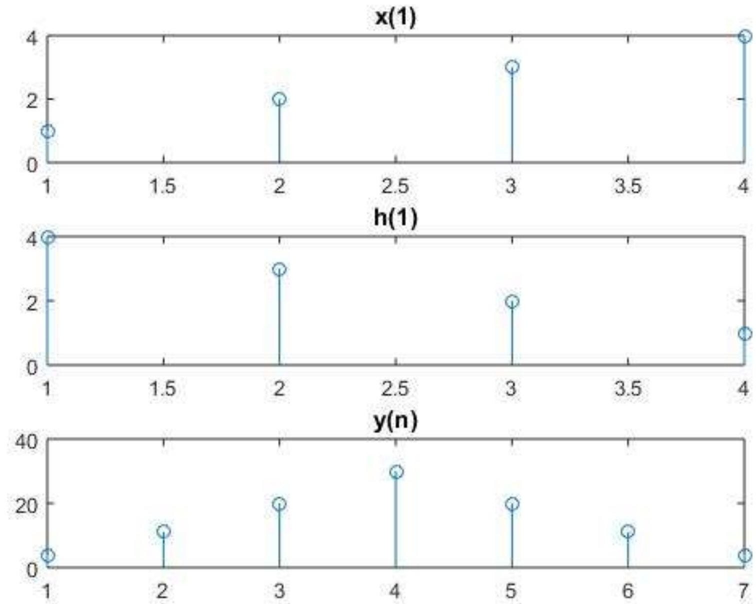
Code for cross-correlation:

```
clc
clear all
close all
x1 = [1 2 3 4];
h1 = [1 2 3 4];
n = length(x1);
m = length(h1);
k = n+m-1;
x = [x1 zeros(1,k-n)]';
h = [h1 zeros(1,k-m)]';
for i=1:k
    c(:,i) = circshift(x,i-1)
end
y = c*x
subplot(3,1,1)
stem (x1)
title('x(1)')
subplot(3,1,2)
stem (h1)
title('h(1)')
subplot(3,1,3)
stem (y)
title('y(n)')
```

<u>Output:</u>

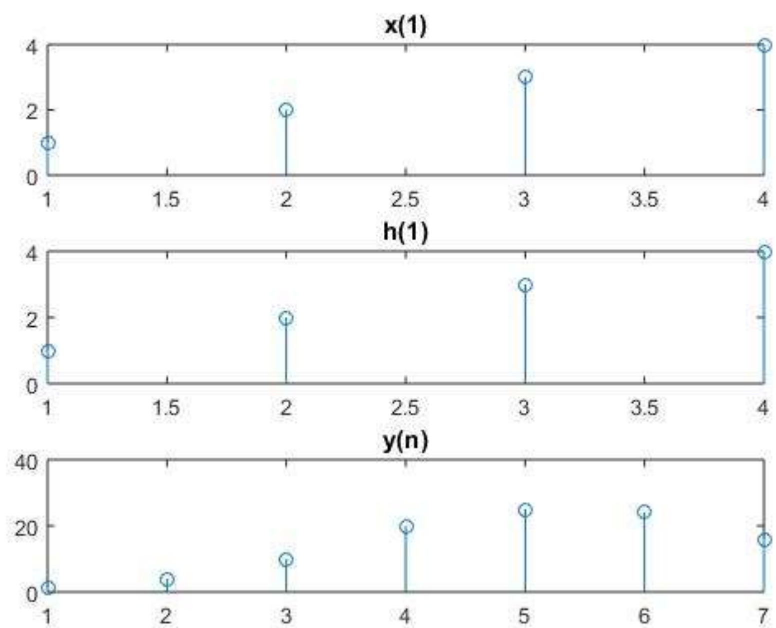
Output for autocorrelation:

```
y =

     4
    11
    20
    30
    20
    11
     4
```



Output for cross-correlation:

```
y =

     1
     4
    10
    20
    25
    24
    16
```



Discussion: Autocorrelation and cross-correlation code was implemented using MATLAB. For implementing the autocorrelation code, two 1×4 matrix was declared. Then length function was used for the length of the matrix. After that, a for loop was used. It was used for shifting the one matrix values. After the for loop a multiplication was written for the output. Then I wrote some code for plotting these signal. Here subplot was used. As the signal was discrete, I used stem function to plot the signal. Implementation of cross-correlation, the code was similar to the autocorrelation. Some changes were created in the multiplication part.

Conclusion: The code was executed successfully and no errors were found. Form this experiment, we had learned about autocorrelation and cross-correlation and how to plot the signal using MATALB.

References:

[1]"Correlation – tutorialspoint", tutorialspoint, 2023. [Online].
Available: https://www.tutorialspoint.com/what-is-correlation-in-signals-and-systems
[Accessed: 11 - May - 2023]

Experiment Date: 14 May, 2023

Experiment No: 04

Name of the Experiment:

i) Take a continuous square wave signal, make a delay of that signal and do the auto correlation of the two signal. Write a code for that signal using MATLAB.

ii) Take a discrete square wave signal, make a delay of that signal and do the auto correlation of the two signal. Write a code for that signal using MATLAB.

iii) Write a code for calculating z-transform of a signal using MATLAB.

Theory: The correlation of two functions or signals or waveforms is defined as the measure of similarity between those signals. There are two types of correlations. They are

1. Autocorrelation
2. Cross-correlation

The autocorrelation function is defined as the measure of similarity or coherence between a signal and its time delayed version. Therefore, the autocorrelation is the correlation of a signal with itself. Autocorrelation is useful for finding repeating patterns in a signal, such as determining the presence of a periodic signal which has been buried under noise or identifying the missing fundamental frequency in a signal implied by its harmonic frequencies.

The cross-correlation between two different signals or functions or waveforms is defined as the measure of similarity or coherence between one signal and the time-delayed version of another signal. The cross-correlation between two different signals indicates the degree of relatedness between one signal and the time-delayed version of another signal. In signal processing, cross-correlation is a measure of similarity of two series as a function of the displacement of one relative to the other. This is also known as a sliding dot product or sliding inner-product. It is commonly used for searching a long signal for a shorter, known feature.

Continuous time signals are defined along a continuum of time and are thus, represented by a continuous independent variable. Continuous-time signals are often referred to as analog signals. This type of signal shows continuity both in amplitude and time. These will have values at each instant of time. Sine and cosine functions are the best example of Continuous time signal.

The signals which are defined at discrete times are known as discrete signals. Therefore, every independent variable has distinct value. Thus, they are represented as sequence of numbers. Although speech and video signals have the privilege to be represented in both continuous and discrete time format; under certain circumstances, they are identical. Amplitudes also show discrete characteristics.

A z-Transform is important for analyzing discrete signals and systems. We know analog signals or signals that are continuous in the time domain. But modern-day communication and system are based on digital processing. This forces us to change our analog signals to the digital domain. The first step in doing this is to sample the analog signal at a rate above a threshold (known as Nyquist sampling rate) as a discrete sequence of points. These points are discretized in time. Each sample occurs at $t=nT_s$, where T_s is the sampling time. After sampling, we need to quantize these samples to one out of M given levels and then encode the quantized samples to binary for further storing, analyzing, or transmitting.

Code:

Code for autocorrelation of continuous signal:

```
clc
clear all
close all
t = 0:0.001:4;
a1 = t>=0 & t<=1;
a2 = t>=1.5 & t<=2.5;
a = a1-a2;
subplot(3, 1, 1);
plot(a);
title('Square wave signal');
d1 = t>=1 & t<=2;
d2 = t>=2.5 & t<=3.5;
d = d1-d2;
subplot(3, 1, 2);
plot(d);
title('Delay signal');
N = -(length(t)-1):(length(t)-1);
corr = xcorr(d,a);
subplot(3, 1, 3);
plot(N,corr);
title('Auto correlation signal');
% max value calculation
maxx = max(corr);
F = find(corr == maxx);
maximum_value = F-length(t)
```

Code for autocorrelation of discrete signal:

```
clc
clear all
close all
t = 0:0.1:4;
a1 = t>=0 & t<=1;
a2 = t>=1.5 & t<=2.5;
a = a1-a2;
subplot(3, 1, 1);
stem(a);
title('Square wave signal');
d1 = t>=1 & t<=2;
d2 = t>=2.5 & t<=3.5;
d = d1-d2;
subplot(3, 1, 2);
stem(d);
title('Delay signal');
```

```
N = -(length(t)-1):(length(t)-1);
corr = xcorr(d,a);
subplot(3, 1, 3);
stem(N,corr);
title('Auto correlation signal');
% max value calculation
maxx = max(corr);
F = find(corr == maxx);
maximum_value = F-length(t)
```

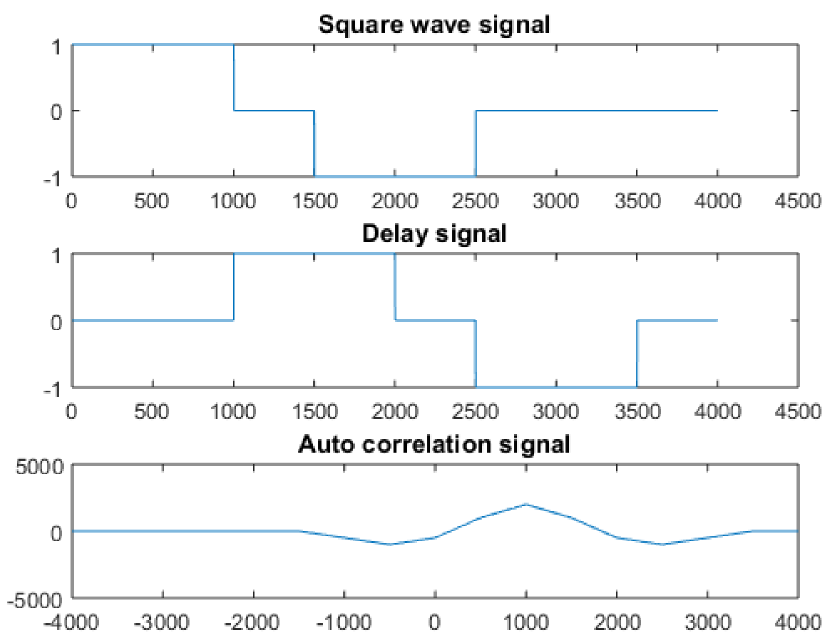
Code for z-transform of a signal:

```
clc
clear all
close all
syms n z
x = 2^n
z_transform = ztrans(x)
```

<u>Output:</u>

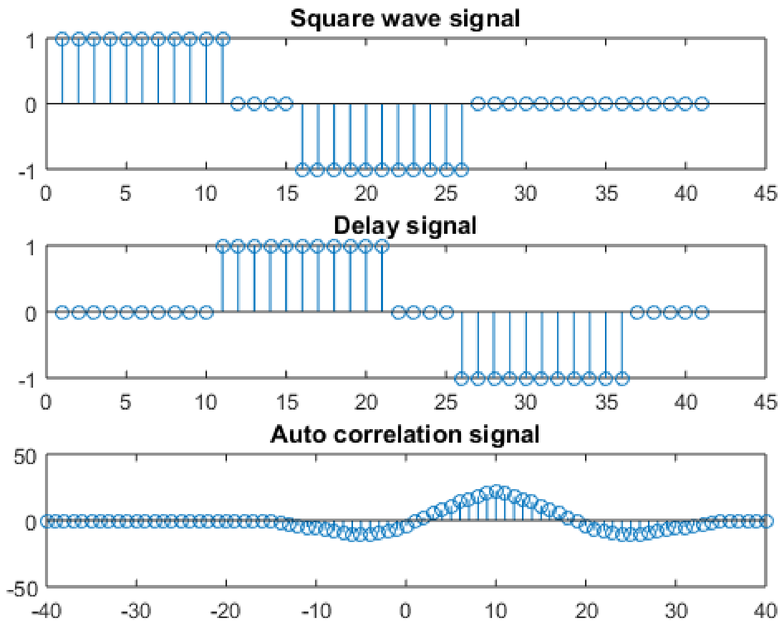
Output for autocorrelation of continuous signal:

```
maximum_value =

        1000
```



Output for autocorrelation of discrete signal:

```
maximum_value =

    10
```



Output for z-transform of a signal:

```
z_transform =

z/(z - 2)
```

Discussion: Autocorrelation and z-transform code was implemented using MATLAB. For implementing the autocorrelation of continuous signal code, first time t was declared. Then wrote some condition for creating a signal and plotted the continuous square wave signal. Next wrote some condition for creating the delay signal and plotted the signal. Then auto correlated the two signal and plotted the signal. For auto correlation, xcorr function was used. In auto correlation signal, length was declared. Then the maximum value of delay was calculated. Implementation of auto correlation of discrete signal code was similar to the continuous signal code. Here stem function was used instead of plot function.

For implementing the z transform code, a function syms n z was declared. Then a signal was declared. Finally, the z transform of the signal was calculated and displayed. Here ztrans function was used.

Conclusion: The code was executed successfully and no errors were found. Form this experiment, we had learned about autocorrelation and z-transform and how to plot the signal using MATALB.

References:

[1]"Correlation – tutorialspoint", tutorialspoint, 2023. [Online].
Available: https://www.tutorialspoint.com/what-is-correlation-in-signals-and-systems
[Accessed: 17 - May - 2023]

[2]"Continuous signal – tutorialspoint", tutorialspoint, 2023. [Online].
Available: https://www.tutorialspoint.com/digital_signal_processing/dsp_signals_definition.htm#
[Accessed: 17 - May - 2023]

[3]"Discrete signal – tutorialspoint", tutorialspoint, 2023. [Online].
Available: https://www.tutorialspoint.com/digital_signal_processing/dsp_signals_definition.htm#
[Accessed: 17 - May - 2023]

[4]"Z-transform – GeeksforGeek ", GeeksforGeek, 2023. [Online].
Available: https://www.geeksforgeeks.org/what-is-z-transform/
[Accessed: 17 - May - 2023]

Experiment Date: 21 May, 2023

Experiment No: 05

Name of the Experiment:

i) Write a code for calculating z-transform of a signal considering left and right sided by using MATLAB.

ii) Write a code for calculating z-transform for a non-causal signal using MATLAB.

Theory: The Z-transform (ZT) is a mathematical tool which is used to convert the difference equations in time domain into the algebraic equations in z-domain. Z transform only exists when the infinite series is convergent. Convergent series is a sum of series which results is fine finite value.

The Z-transform is a very useful tool in the analysis of a linear shift invariant (LSI) system. An LSI discrete time system is represented by difference equations. To solve these difference equations which are in time domain, they are converted first into algebraic equations in z-domain using the Z-transform, then the algebraic equations are manipulated in z-domain and the result obtained is converted back into time domain using the inverse Z-transform.

A system is said to be causal if input does not depend on future values of that input but depends only on the present or past values of the input. y(n) = x(n) is the example of causal system.

A system is said to be non-causal if input depends on future values of that input. y(n-1) = x(n) is the example of non-causal system.

Code:

Code for z-transform of a signal considering left and right sided:

```
clc
clear all
close all
x = [1 2 3 4 5 6 7];
l = length(x);
y = sym('z');
zt_l = 0;
zt_r = 0;
for i=1:l
        zt_l = zt_l+x(i)*y^(-i);
    end
    for i=1:l
        zt_r = zt_r+x(i)*y^(i);
    end
    disp('Right = ')
    disp(zt_l)
    disp('Left = ')
    disp(zt_r)
```

Code for z-transform for non-causal signal:

```
clc
clear all
close all
x = [1 2 3 4 5 6 7];
inp = input('Enter the index: ');
l = length(x);
y = sym('z');
zt_l = 0;
zt_r = 0;
      for i=1:l
          if i>=inp
               zt_r = zt_r+x(i)*y^(inp-i);
          else
              zt_l = zt_l+x(i)*y^((-1)*(i-inp));
          end
      end
      output = zt_l + zt_r;
      disp('Output = ')
      disp(output)
```

<u>Output:</u>

Output for z-transform of a signal considering left and right sided:

```
Right =
1/z + 2/z^2 + 3/z^3 + 4/z^4 + 5/z^5 + 6/z^6 + 7/z^7

Left =
7*z^7 + 6*z^6 + 5*z^5 + 4*z^4 + 3*z^3 + 2*z^2 + z
```

Output for z-transform for non-causal signal:

```
Enter the index: 3
Output =
2*z + 4/z + 5/z^2 + z^2 + 6/z^3 + 7/z^4 + 3
```

Discussion: Z-transform code was implemented using MATLAB. For implementing the z transform code considering left and right sided, a signal was taken. The signal length was calculated using length function. Then a for loop was used. In the for loop a condition was written for left sided. Then another for loop was used for right sided condition. After the for loop output was displayed.

For implementing the z transform of non-causal signal code, a signal was taken. Index no. was taken by the user. The signal length was calculated using length function. Then a for loop was used. In the for loop if else condition was used. In the if else, a condition was written for left and right sided. Then the summation of left and right sided output was written. After that the output of the z transform was displayed.

Conclusion: The code was executed successfully and no errors were found. Form this experiment, we had learned about z-transform, causal and non-causal signal.