Aim ↬ Using MATLAB write a program for various MATLAB operations like folding, scaling, shifting, addition, subtraction, multiplication, and division.

Software Required ↬ MATLAB

Theory ↬

Operations on signals, including addition, multiplication, scaling, shifting, and folding, allow us to modify and analyze these signals to extract meaningful insights or achieve specific objectives. These fundamental operations are essential for manipulating signals in signal processing, enabling tasks such as filtering, signal enhancement, and system analysis.

1. Addition of Signals ↪

Signal addition combines two or more signals by adding their corresponding values at each point in time. The resulting signal at any point is the sum of the values of the individual signals at that point. If x(t) and y(t) are two continuous-time signals, their addition z(t) is given by:

$$z(t) = x(t) + y(t)$$

For discrete-time signals x[n] and y[n]:

$$z[n] = x[n] + y[n]$$

2. Subtraction of Signals ↪

Signal subtraction reduces two or more signals by subtracting their corresponding values at each point in time. The resulting signal at any point is the subtraction of the values of the individual signals at that point. If x(t) and y(t) are two continuous-time signals, their addition z(t) is given by:

$$z(t) = x(t) - y(t)$$

For discrete-time signals x[n] and y[n]:

$$z[n] = x[n] - y[n]$$

3. Multiplication of Signals ↪

Signal multiplication generates a new signal by multiplying the corresponding values of two signals at each point in time. The product signal at any point is the

product of the values of the individual signals at that point. For continuous-time signals x(t) and y(t), the product z(t) is:

$$z(t) = x(t) \cdot y(t)$$

For discrete-time signals x[n] and y[n]:

$$z[n] = x[n] \cdot y[n]$$

4. Division of Signals ↪

Signal division generates a new signal by dividing the corresponding values of two signals at each point in time. The product signal at any point is the division of the values of the individual signals at that point. For continuous-time signals x(t) and y(t), the product z(t) is:

$$z(t) = x(t)/y(t)$$

For discrete-time signals x[n] and y[n]:

$$z[n] = x[n]/y[n]$$

5. Time Scaling of Signals ↪

Time scaling involves compressing or expanding a signal in time. This operation changes the time duration of the signal without affecting its amplitude. For a continuous-time signal x(t) and a scalar a:

$$y(t) = x(\propto t)$$

For a discrete-time signal x[n] and a scalar a:

$$y[n] = x[\propto n]$$

If $|\propto| > 1$, the signal is compressed in time (appears faster).

If $|\propto| < 1$, the signal is expanded in time (appears slower).

6. Amplitude Scaling of Signals ↪

Amplitude Scaling a signal involves multiplying every value of the signal by a constant factor. This operation changes the amplitude of the signal without altering its shape. For a continuous-time signal x(t) and a scalar a:

$$y(t) = a \cdot x(t)$$

For a discrete-time signal x[n] and a scalar a:

$$y[n] = a \cdot x[n]$$

7. Shifting of Signals ↪

Shifting a signal involves moving it forward or backward in time. A right shift (or delay) means the signal occurs later in time, while a left shift (or advance) means the signal occurs earlier. For a continuous-time signal x(t):

- Right shift [delay] by T: $y(t) = x(t - T)$
- Left shift [advance] by T: $y(t) = x(t + T)$

For a discrete-time signal x[n]:

- Right shift [delay] by k: $y[n] = x[n - k]$
- Left shift [advance] by k: $y[n] = x[n + k]$

8. Time Reversal [Folding] of Signals ↪

Folding, or time reversal, of a signal is the operation of flipping the signal around the vertical axis, effectively reversing the direction of time for the signal. For a continuous-time signal x(t):

$$y(t) = x(-t)$$

For a discrete-time signal x[n]:

$$y[n] = x[-n]$$

Code ↬

```
%Operation on Continuous Time Signals
clc;
clear;

t = -5:0.01:5;

x_t = cos(2*pi*t);
y_t = (1-(abs(t)./7)).*(t>=-0.5 & t<=4.5) + 0.5.*(-4<t&t<-1.5);
```

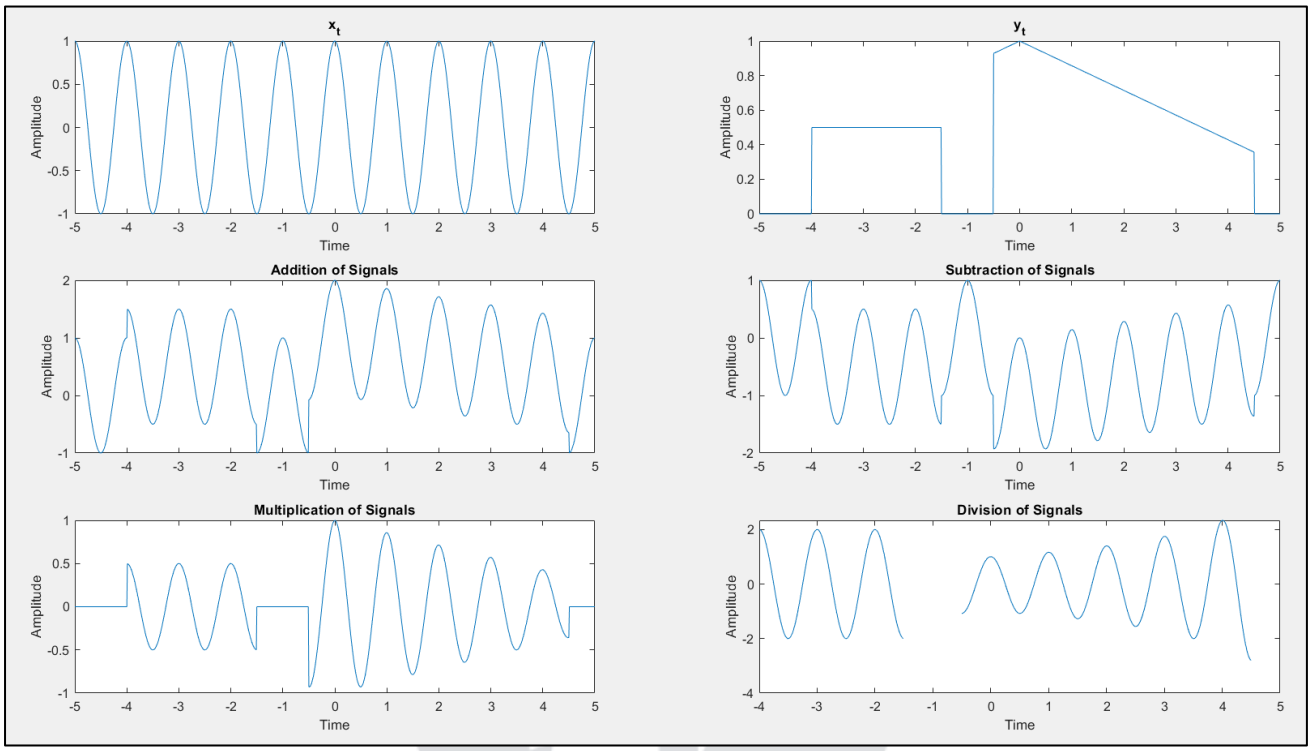
```
figure;
subplot(3,2,1); plot(t,x_t); xlabel("Time"); ylabel("Amplitude"); title("x_t");
subplot(3,2,2); plot(t,y_t); xlabel("Time"); ylabel("Amplitude"); title("y_t");

a_t = x_t + y_t;    %Addition
s_t = x_t - y_t;    %Subtraction
m_t = x_t .* y_t;   %Multiplication
d_t = x_t ./ y_t;   %Division

subplot(3,2,3); plot(t,a_t); xlabel("Time"); ylabel("Amplitude"); title("Addition of
Signals");
subplot(3,2,4); plot(t,s_t); xlabel("Time"); ylabel("Amplitude"); title("Subtraction of
Signals");
subplot(3,2,5); plot(t,m_t); xlabel("Time"); ylabel("Amplitude"); title("Multiplication of
Signals");
subplot(3,2,6); plot(t,d_t); xlabel("Time"); ylabel("Amplitude"); title("Division of Signals");

%Time Scaling
ts = input("Enter Time Scaling parameter: ");

x_t_compress = cos(2*pi*ts*t);
x_t_expand = cos(2*pi*(1/ts)*t);

figure;
subplot(2,2,1); plot(t,x_t_expand); xlabel("Time"); ylabel("Amplitude"); title("Time
Scaling - Expansion")
subplot(2,2,2); plot(t,x_t_compress); xlabel("Time"); ylabel("Amplitude"); title("Time
Scaling - Compression")

%Amplitude Scaling
as = input("Enter Amplitude Scaling parameter: ");

x_t_amplify = as.*cos(2*pi*t);
x_t_attenuate = (1/as).*cos(2*pi*t);

subplot(2,2,3); plot(t,x_t_amplify); xlabel("Time"); ylabel("Amplitude"); title("Amplitude
Scaling - Amplify")
subplot(2,2,4); plot(t,x_t_attenuate); xlabel("Time"); ylabel("Amplitude");
title("Amplitude Scaling - Attenuation")
```

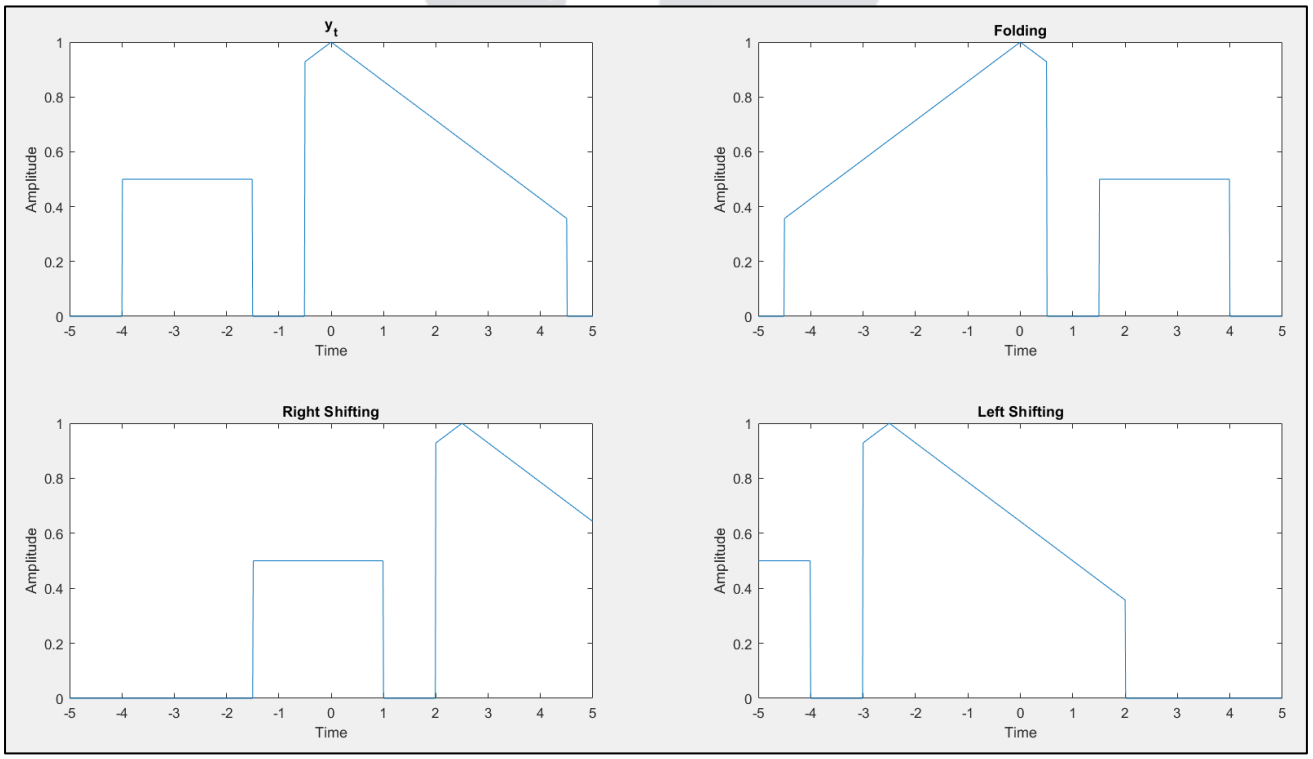
```
%Folding
figure;
subplot(2,2,1); plot(t,y_t); xlabel("Time"); ylabel("Amplitude"); title("y_t");

f_t = fliplr(y_t);
subplot(2,2,2); plot(t,f_t); xlabel("Time"); ylabel("Amplitude"); title("Folding");

%Time Shifting
s = input("Enter value to shift the signal: ");
rs = t-s;
ls = t+s;

y_t_rs = (1-(abs(rs)./7)).*(rs>=-0.5 & rs<=4.5) + 0.5.*(-4<rs & rs<-1.5);
y_t_ls = (1-(abs(ls)./7)).*(ls>=-0.5 & ls<=4.5) + 0.5.*(-4<ls & ls<-1.5);

subplot(2,2,3); plot(t,y_t_rs); xlabel("Time"); ylabel("Amplitude"); title("Right Shifting");
subplot(2,2,4); plot(t,y_t_ls); xlabel("Time"); ylabel("Amplitude"); title("Left Shifting");

%Operation on Discrete Time Signals
clc;
clear;

n = -2:1:4;

x_n = [-4,2,3,0,5,-3,1];
y_n = [1,-2,2,-3,4,0,3];

figure;
subplot(3,2,1); stem(n,x_n); xlabel("Time"); ylabel("Amplitude"); title("x_n");
subplot(3,2,2); stem(n,y_n); xlabel("Time"); ylabel("Amplitude"); title("y_n");

a_n = x_n + y_n;    %Addition
s_n = x_n - y_n;    %Subtraction
m_n = x_n .* y_n;   %Multiplication
d_n = x_n ./ y_n;   %Division

subplot(3,2,3); stem(n,a_n); xlabel("Time"); ylabel("Amplitude"); title("Addition of
Signals");
subplot(3,2,4); stem(n,s_n); xlabel("Time"); ylabel("Amplitude"); title("Subtraction of
Signals");
```

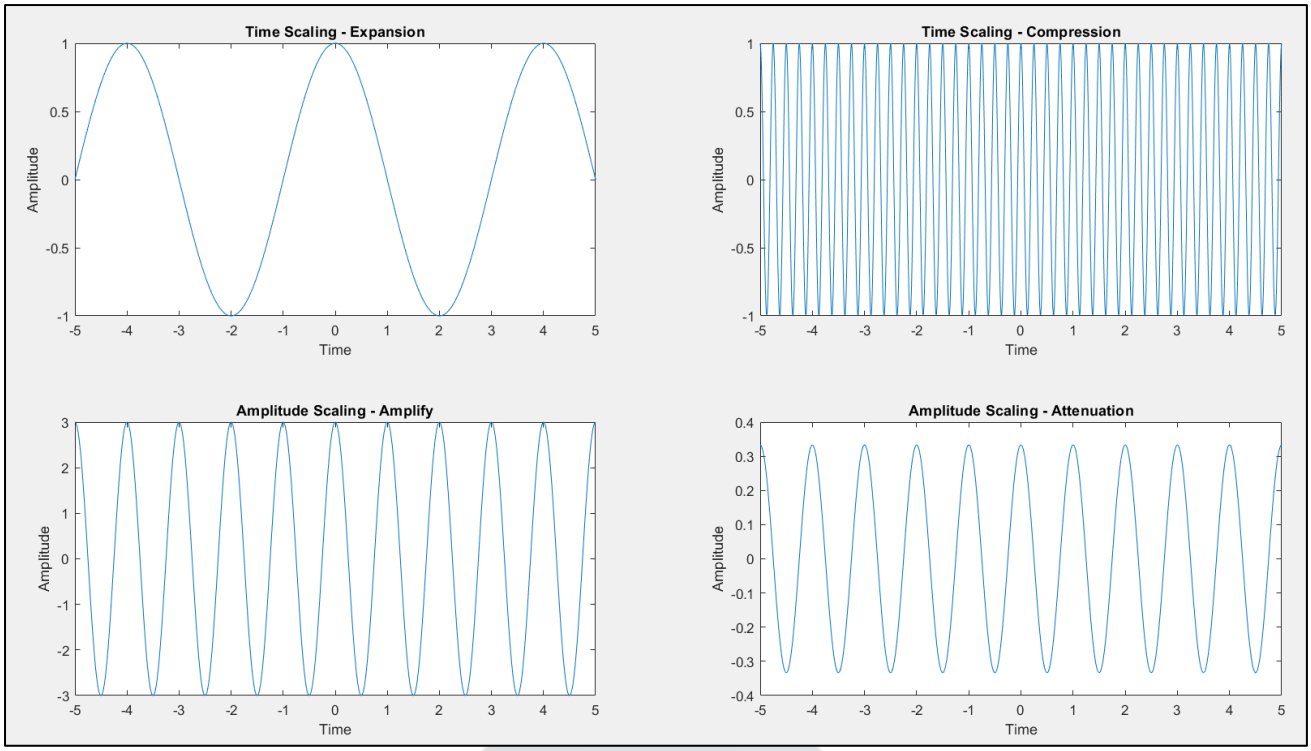
```
subplot(3,2,5); stem(n,m_n); xlabel("Time"); ylabel("Amplitude"); title("Multiplication of Signals");
subplot(3,2,6); stem(n,d_n); xlabel("Time"); ylabel("Amplitude"); title("Division of Signals");

%Time Scaling
ns = input("Enter Time Scaling parameter: ");
ns_amplify = ns*n;
ns_attenuate = (1/ns)*n;

figure;
subplot(2,2,1); stem(ns_amplify,x_n); xlabel("Time"); ylabel("Amplitude"); title("Time Scaling - Expansion")
subplot(2,2,2); stem(ns_attenuate,x_n); xlabel("Time"); ylabel("Amplitude"); title("Time Scaling - Compression")

%Amplitude Scaling
as = input("Enter Amplitude Scaling parameter: ");

x_n_amplify = as.*x_n;
x_n_attenuate = (1/as).*x_n;

subplot(2,2,3); stem(n,x_n_amplify); xlabel("Time"); ylabel("Amplitude");
title("Amplitude Scaling - Amplify")
subplot(2,2,4); stem(n,x_n_attenuate); xlabel("Time"); ylabel("Amplitude");
title("Amplitude Scaling - Attenuation")

%Folding
figure;
subplot(2,2,1); stem(n,y_n); xlabel("Time"); ylabel("Amplitude"); title("y_n");

f_n = fliplr(y_n);
subplot(2,2,2); stem(n,f_n); xlabel("Time"); ylabel("Amplitude"); title("Folding");

%Time Shifting
s = input("Enter value to shift the signal: ");

y_n_rs = [zeros(1,s), y_n(1:end-s)];
y_n_ls = [y_n(s+1:end),zeros(1,s)];
```

```
subplot(2,2,3); stem(n,y_n_rs); xlabel("Time"); ylabel("Amplitude"); title("Right Shifting");
subplot(2,2,4); stem(n,y_n_ls); xlabel("Time"); ylabel("Amplitude"); title("Left Shifting");
```

Output ↬







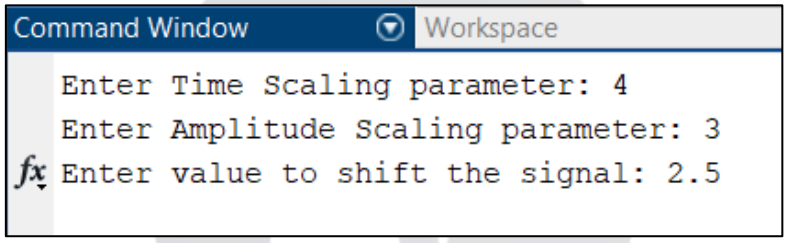
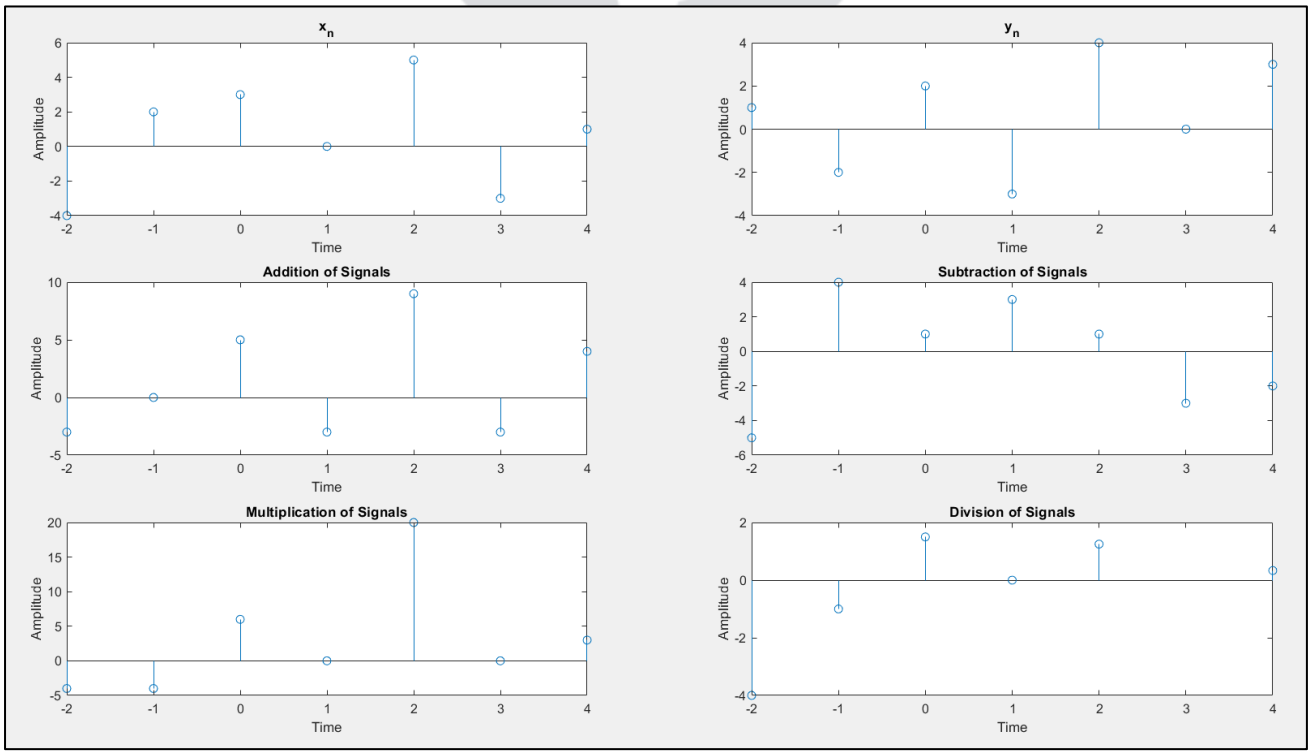
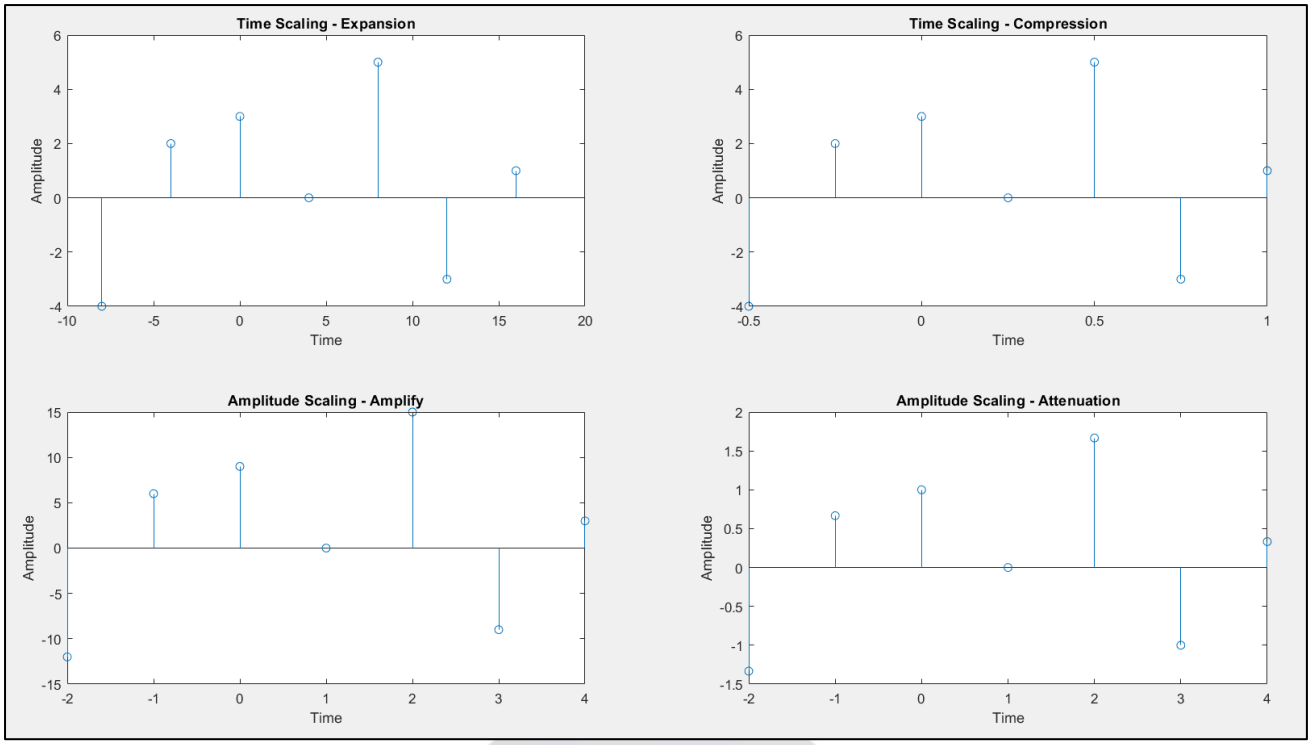
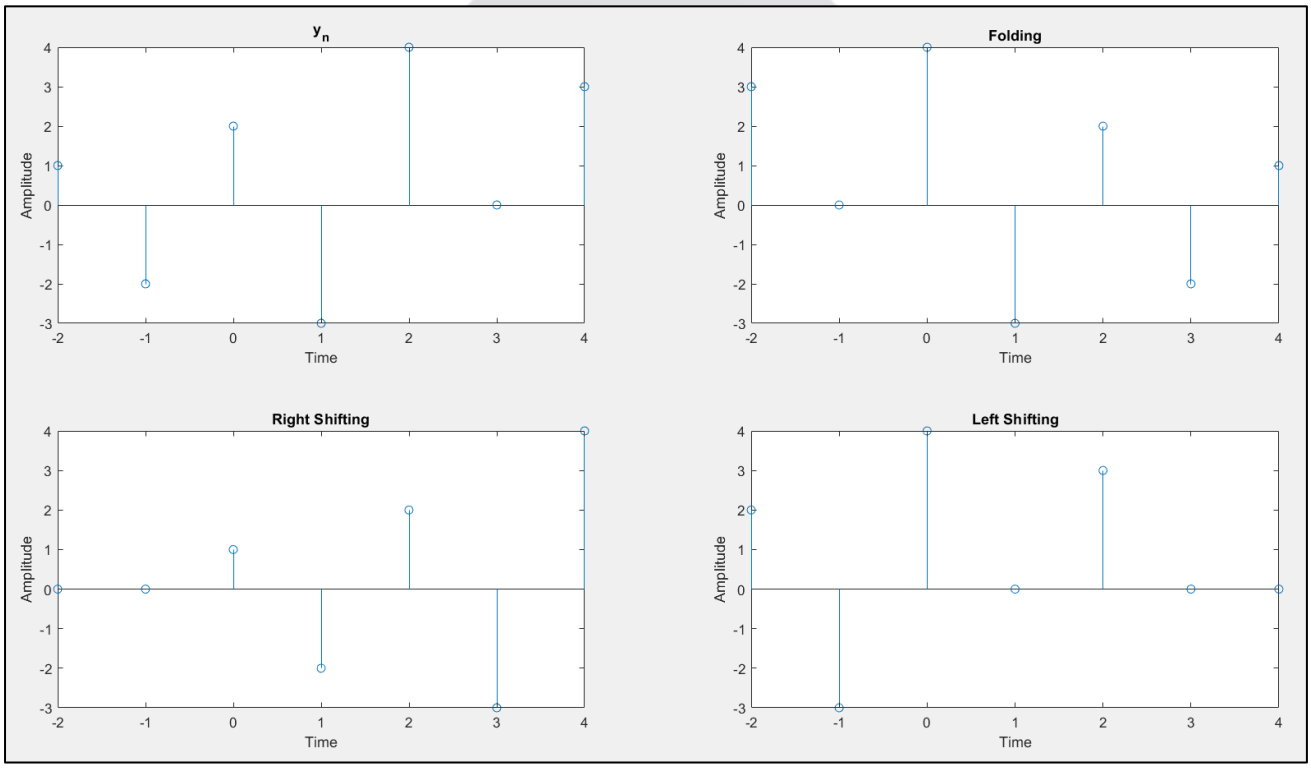


Fig. i) Continuous Time Signal Operations and Parameters







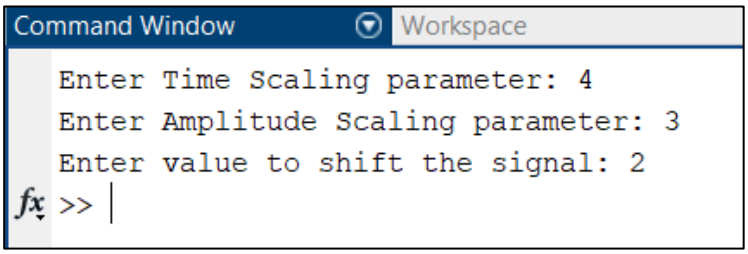


Fig. ii) Discrete Time Signal Operations and Parameters

Result ↬

By using MATLAB, we successfully performed various operations on signals, including addition, subtraction, multiplication, division, time scaling, amplitude scaling, shifting, and folding. The output signals were visualized and their properties were analyzed.

Conclusion ↬

The MATLAB-based analysis demonstrated effective techniques for manipulating and analyzing signals. These operations are fundamental in signal processing, allowing us to understand and modify signal characteristics for various applications.

Precautions ↬

- Ensure correct sampling rates to avoid aliasing.
- Use appropriate time and amplitude scaling factors to prevent distortion.
- Verify signal lengths and indexing when performing operations on discrete-time signals.
- Handle large datasets carefully to manage computational resources efficiently.