Experiment No. 1

Aim: Construction of standard DT signals(Sine,Cosine,Unit Step, Unit Impulse, Ramp) and perform different operations (addition ,subtraction , multiplication & Time scaling > 1 & Time scaling <1) on the signal

Objective: Develop a program to generate standard signals like unit impulse, unit step and unit ramp and perform different manipulation operations on them.

Software:MATLAB

Theory:

Signal Generation Standard Signals

1. Unit Impulse Sequence

$$\delta[n] = \begin{cases} 1, & n = 0 \\ 0, & n \neq 0 \end{cases}$$

2. Unit Step Sequence

$$u[n] = \begin{cases} 0, & n < 0 \\ 1, & n \geq 0 \end{cases}$$

3. Unit Ramp Sequence

$$\text{ramp}[n] = \begin{cases} n, & n \geq 0 \\ 0, & n < 0 \end{cases}$$

Signal Manipulation

1. Time scaling Compression
2. Time scaling Expansion
3. Signal Addition
4. Signal Subtraction
5. Signal Multiplication

Problem Definition:

Plot a graph of all signal generation and manipulation techniques by taking time duration on x-axis and signal amplitude on y-axis.

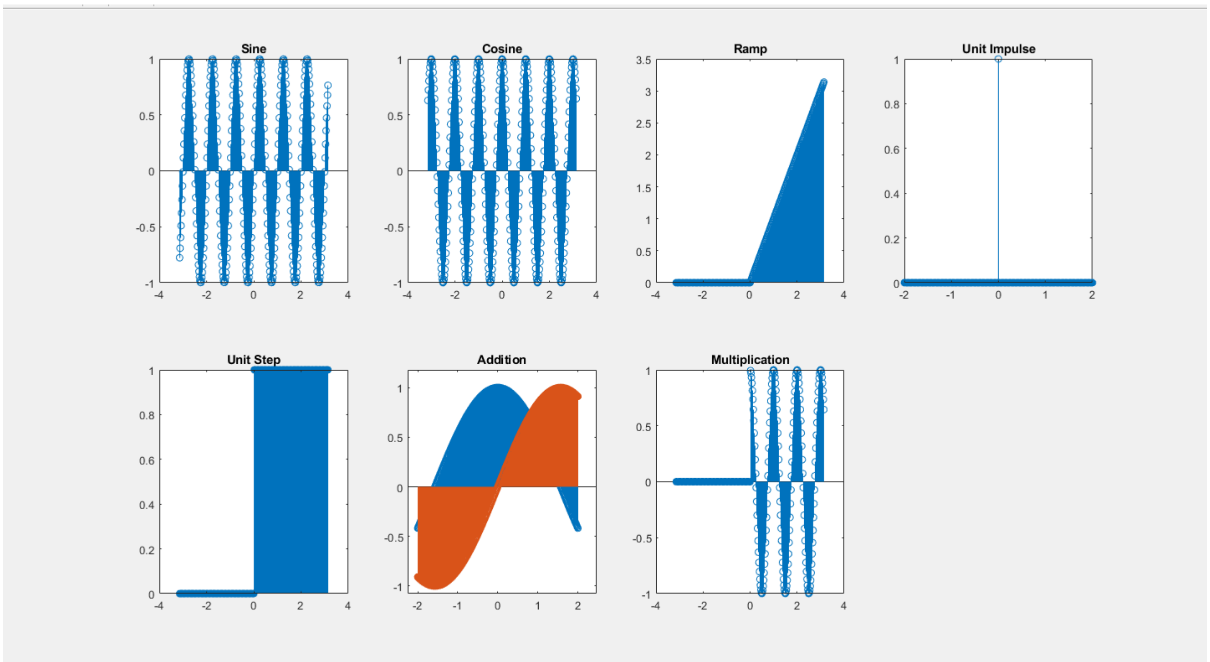
Experiment 1A:

Code:

```
n = [-pi: 0.02: pi];
t=-2:0.01:2;


sine_y = sin(2*n*pi);
cosine_y= cos(2*n*pi);
ramp_y= max(0,n);
unitImpulse_y = t==0;
unitstep_y = n>=0;

subplot(2,4,1),stem(n,sine_y);
title('Sine');

subplot(2,4,2),stem(n,cosine_y);
title('Cosine');

subplot(2,4,3),stem(n,ramp_y);
title('Ramp');

subplot(2,4,4),stem(t,unitImpulse_y);
title('Unit Impulse');

subplot(2,4,5),stem(n,unitstep_y);
title('Unit Step');

addition_t = -pi:0.01:(2*pi);
x = cos(t);
```

```
y = sin(t);
subplot(2,4,6),stem(t,x); hold on;
stem(t,y);
axis([0 2*pi -1.5 1.5])
title('Addition');


multiplication=cosine_y.*unitstep_y;
subplot(2,4,7),stem(n,multiplication);
title('Multiplication');
```

Output:



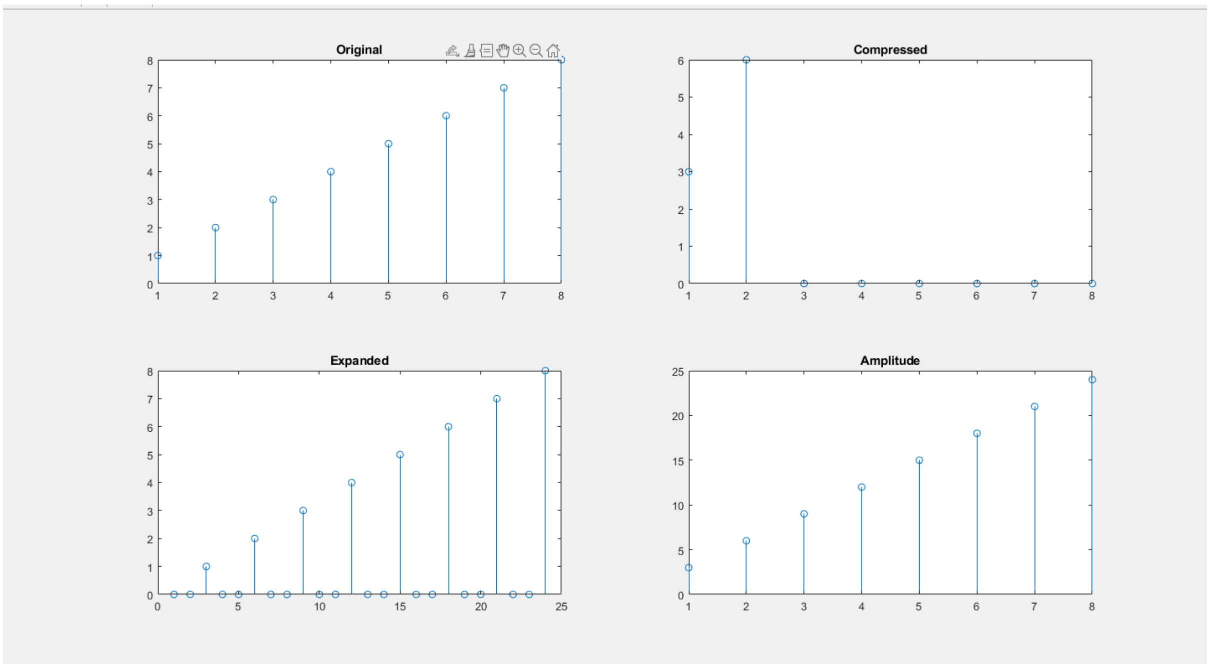
Experiment 1B:

Code:

```
signal = input("Enter Signal : ");
%signal = [1,2,3,4,5,6,7,8];x
signal_length = length(signal);
samples = 1:1:signal_length;
```

```
subplot(2,2,1),stem(samples,signal);
title('Original');

factor = input('Enter Factor : ');
%Create an array of zeros
compressed = zeros(1,signal_length);
for i = 1:signal_length
    %If new array ietm is out of bounds
    if i*factor > signal_length
        compressed(1,i) = 0;
    else
        compressed(1,i) = signal(1,factor*i);
    end
end
subplot(2,2,2),stem(samples,compressed);
title('Compressed');

samples = 1:1:factor*signal_length;
expanded = zeros(1,factor*signal_length);
for i=1:1:factor*signal_length
    if floor(i/factor) == i/factor
        expanded(1,i) = signal(1,i/factor);
    end
end
subplot(2,2,3),stem(samples,expanded);
title('Expanded');

samples = 1:1:signal_length;
subplot(2,2,4),stem(samples,factor*signal);
title('Amplitude');
```

Output:



Conclusion:

In this experiment, we learnt about various signal operations and plotted them using MATLAB's array operations. MATLAB treats signals as an array of numbers with the index linking them together and number of dimensions equal to the number of arrays.

Batch: BE Comps A2