# DIGITAL SIGNAL PROCESSING EXPERIMENTS

Đinh Hoàng Dũng - 2110096
Dương Gia Minh - 2110352

Ho Chi Minh City, 2024

# Overview

A digital signal processing report includes an introduction, background, methodology, results, discussion, conclusion, and references, providing an overview of DSP theory, methods used, findings and implications. It aims to present and interpret signal processing outcomes, offering insights into its applications and significance.

## 1 Introduction

This report is for Digital Signal Processing Experiment. The report contains 6 labs:
**Lab 1:** Basic command with Scilab, use SciLab to draw the signal after every step in the ADC procedure.
**Lab 2:** Basic operation with signal, use Scilab to draw the graph of signal.
**Lab 3:** Script function in Scilab.
**Lab 4:** Analyze systems using Z-transform. Calculate linear and circular convolution of the discrete signals.
**Lab 5:** Z transform, $Z^+$ transform, and reverse Z transform.
**Lab 6:** Signal and System in Frequency Domain. Use Scilab to draw spectrums.

## 2 Goal

- Classify signals and systems.
- Describe sampling theorem of time-based signal.
- Express signal and system in time domain.
- Apply Z-transform in digital signal processing.
- Produce Discrete Fourier Transform.

## 3 Tools and Methods

### 3.1 Tools: SCILAB

SCILAB is a numerical, programming and graphics environment available for free from the French *Government's "Institut Nationale de Recherche en Informatique et en Automatique - INRIA* (National Institute for Informatics and Automation Research)." It is similar in operation to MATLAB and other existing numerical/graphic environments, and it can be run using a variety of operating systems including UNIX, Windows, Linux, etc. SCILAB is a self-contained package including a large number of intrinsic numeric, programming and graphics functions.

### 3.2 Methods

#### 3.2.1 Z Transform

The Z-transform is a powerful tool in digital signal processing for analyzing discrete-time signals and systems. The Z-transform of a discrete-time signal $x[n]$ is defined as:

$$X(z) = \sum_{n=-\infty}^{\infty} x[n]z^{-n} \tag{1}$$

where $z$ is a complex variable. The Z-transform provides a convenient way to analyze signals and systems in the frequency domain.
**Properties of the Z-Transform**

1. **Linearity:** If $X_1(z)$ and $X_2(z)$ are the Z-transforms of signals $x_1[n]$ and $x_2[n]$, respectively, and $a$ and $b$ are constants, then the Z-transform of the linear combination $ax_1[n] + bx_2[n]$ is given by:

$$aX_1(z) + bX_2(z) \tag{2}$$

2. **Time Shifting:** If $X(z)$ is the Z-transform of signal $x[n]$, then the Z-transform of the delayed signal $x[n-k]$ is given by:
$$z^{-k}X(z) \tag{3}$$

3. **Time Reversal:** If $X(z)$ is the Z-transform of signal $x[n]$, then the Z-transform of the time-reversed signal $x[-n]$ is given by:
$$X\left(\frac{1}{z}\right) \tag{4}$$

4. **Convolution:** If $Y(z)$ is the Z-transform of signal $y[n]$, and $H(z)$ is the Z-transform of the impulse response of a system, then the Z-transform of the convolution $x[n] * h[n]$ is given by:
$$X(z) \cdot H(z) \tag{5}$$

### 3.2.2 Fourier Transform

The Fourier transform is a mathematical tool used to decompose a function into its constituent frequencies. For a continuous-time signal $x(t)$, the Fourier transform $X(\omega)$ is defined as:

$$X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t}\, dt \tag{6}$$

where $j$ is the imaginary unit and $\omega$ represents angular frequency.
**Properties of the Fourier Transform**

1. **Linearity:** If $X_1(\omega)$ and $X_2(\omega)$ are the Fourier transforms of signals $x_1(t)$ and $x_2(t)$, respectively, and $a$ and $b$ are constants, then the Fourier transform of the linear combination $ax_1(t) + bx_2(t)$ is given by:
$$aX_1(\omega) + bX_2(\omega) \tag{7}$$

2. **Time Shifting:** If $X(\omega)$ is the Fourier transform of signal $x(t)$, then the Fourier transform of the time-shifted signal $x(t-\tau)$ is given by:
$$e^{-j\omega\tau}X(\omega) \tag{8}$$

3. **Frequency Shifting:** If $X(\omega)$ is the Fourier transform of signal $x(t)$, then the Fourier transform of the frequency-shifted signal $x(t)e^{j\omega_0 t}$ is given by:
$$X(\omega - \omega_0) \tag{9}$$

4. **Convolution:** If $Y(\omega)$ is the Fourier transform of signal $y(t)$, and $H(\omega)$ is the Fourier transform of the impulse response of a system, then the Fourier transform of the convolution $x(t) * h(t)$ is given by:
$$X(\omega) \cdot H(\omega) \tag{10}$$

# 4 Content

## 4.1 Lab 1

**Exercise 1.1** Using the operators in vector and matrix to do the following tasks

- Create a vector in the form of (x1 + 1, x2 + 1, x3 + 1, x4 + 1) where x1, x2, x3, x4 are components of vector x = 1:4

Code:
```
//Ex1.1
x = 1 : 4;
result = x + 1;
disp('Result 1:', result)
```

Result:
```
"Result 1:"

 2.   3.   4.   5.
```

- Create a vector in the form of (x1y1, x2y2, x3y3, x4y4) where x1...x4 and y1...y4 are components of vector x=1:4 and vector y=5:8, respectively.

Code:
```
//Ex1.2
x = 1 : 4;
y = 5 : 8;
result = x .* y;
disp('Result 2:', result)
```

Result:
```
"Result 2:"

 5.   12.   21.   32.
```

- Create a vector in the form of (sin(x1), sin(x2), . . . , sin(x10)) where x is a vector of 10 values linearly chosen in the interval [0, $\pi$].

Code:

```
//Ex1.3
x = linspace(0, %pi, 10);
result = sin(x);
disp('Result 3:', result)
```

Result:

```
"Result 3:"

      column 1 to 9
 0.   0.3420201   0.6427876   0.8660254   0.9848078   0.9848078   0.8660254   0.6427876   0.3420201
      column 10
 1.225D-16
```

**Exercise 1.2** Consider the following analog signal $x_a(t) = 3sin(100\pi t)$

- Use SciLab to draw $x_a(t)$ in 5 periods.
  First, calculate period of the signal. T = $\frac{2\pi}{\omega} = \frac{2}{100}$
  Code to plot $x_a(t)$:

```
t = linspace(0, 5.*(2/100), 1000);
// Plot the signal
subplot(3, 1, 1);
plot(t, 3 .* sin(100.*%pi.*t), "k");
xlabel('Time (s)');
ylabel('x_a(t)');
title('Analog signal x_a(t) in 5 periods');
```

- Determine the discrete-time signal x(n) of the signal $x_a(t)$ that is sampled with a sampling rate $F_s$ = 300 samples/s.
  $F_s$ = 300 samples/s which means that $F_s$ = 300 Hz. Hence, x(n) = $3sin(\frac{100\pi}{300}n) = 3sin(\frac{\pi n}{3})$

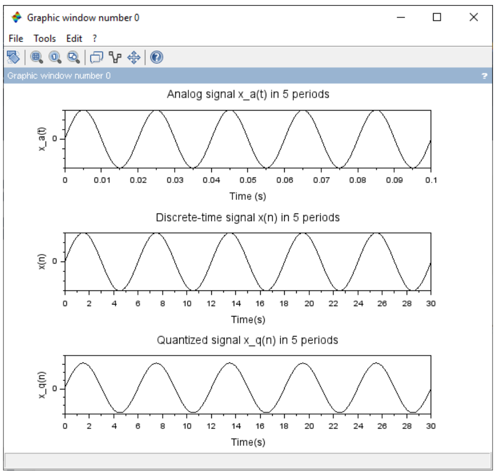
- Determine the discrete-time signal x(n) and determine the periodic property of x(n). If x(n) is a periodic signal, determine the frequency and period of x(n). Then, use SciLab to draw x(n) in 5 periods.
  We have: f = $\frac{\omega}{2\pi} = \frac{\pi/3}{2\pi} = \frac{1}{6}$. Because f is a rational number => x(n) is periodic.
  Code to plot x(n) in 5 periods:

```
n = linspace(0, 5.*6, 1000);
//Plot the signal
subplot(3, 1, 2);
plot(n, 3 .* sin(n.*%pi/3), "k");
xlabel('Time(s)');
ylabel('x(n)');
title('Discrete-time signal x(n) in 5 periods');
```

- Determine the quantized signal $x_q(n)$ if $\Delta = 0.1$ using a truncated method. Then, draw $x_q(n)$ in 5 periods.
  With truncated method, we have: $|e_q(n)| < \Delta$.
  Quantization error: $e_q(n) = x_q(n) - x(n) => x_q(n) = x(n) + e_q(n)$.
  Hence, $x_q(n) = 3sin(\frac{\pi n}{3}) + 0.1$
  Code to plot $x_q(n)$ in 5 periods:

```
n = linspace(0, 5.*6, 1000);
//Plot the signal
subplot(3, 1, 3);
plot(n, 0.1 + 3 .* sin(n.*%pi/3), "k");
xlabel('Time(s)');
ylabel('x_q(n)');
title('Quantized signal x_q(n) in 5 periods');
```

Result:




## 4.2 Lab 2

**Exercise 1** Let's investigate following functions on Scilab and briefly report their functionalities and how to use it.

| Functions | Description |
|---|---|
| plot2d3(..) | Create 2-dimensional plots where data is represented as vertical bars extending down to the x-axis. |
| min(..) | For A, a real vector or matrix, min(A) is the least element of A. |
| max(..) | For A, a real vector or matrix, max(A) is the greatest element of A. |
| subplot(..) | subplot(m,n,p) virtually grids the current graphic window or uicontrol frame into a m-by-n matrix of rectangular sub-areas (cells), and selects the $p^{th}$ cell for receiving the forthcoming drawings. |
| title(..) | Display a title at the top of the current or given axes, or to change properties of the existing title. |
| xlabel(..) | Sets or updates the x-axis label or/and its properties. |
| ylabel(..) | Sets or updates the y-axis label or/and its properties. |
| bool2s(..) | Convert Boolean matrix to a zero one matrix. |
| deff(..) | In-line definition of a (anonymous) function |

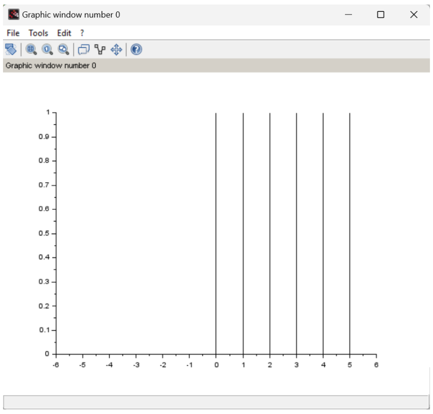
**Exercise 2** Try the following scripts on Scilab and report what your understanding after observing the output.

- First, define a vector n containing integers from -5 to 5.
- Then, creates a new vector msignal by converting the logical expression n $\geq$ 0 to a string vector using bool2s() function. If an element of n is greater than or equal to 0, the condition is true and will be represented as "1", otherwise is "0".
- Finally, plot the data in 2 dimensions n and msignal. Data will be represented as vertical bars.

Code:

```
n = -5:5;
msignal = bool2s (n>=0);
plot2d3(n, msignal);
```

Result:

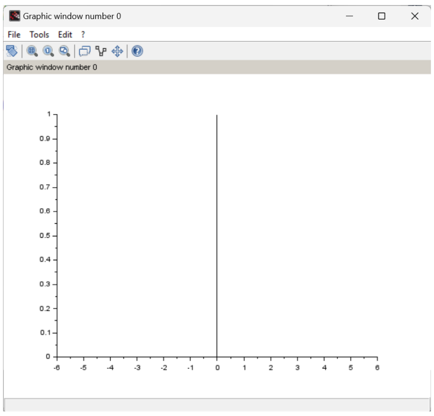



**Exercise 3** Try the following scripts on Scilab and report what your understanding after observing the output. Similar to the previous exercise, but this time the condition is used to check if n is equal to 0 or not.

Code:

```
n = -5:5;
msignal = bool2s(n==0);
plot2d3(n, msignal);
```

Result:

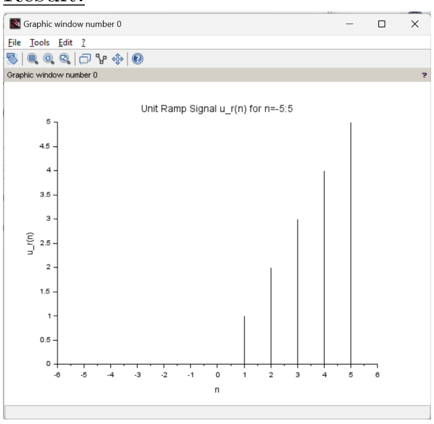



**Exercise 4** Use Scilab to draw the unit ramp signal $u_r(n)$ for n=-5:5

Code:

```
n = -5:5;
ur = max(n, 0);
plot2d3(n, ur);
title('Unit Ramp Signal u_r(n) for n=-5:5');
xlabel('n');
ylabel('u_r(n)');
```

Result:

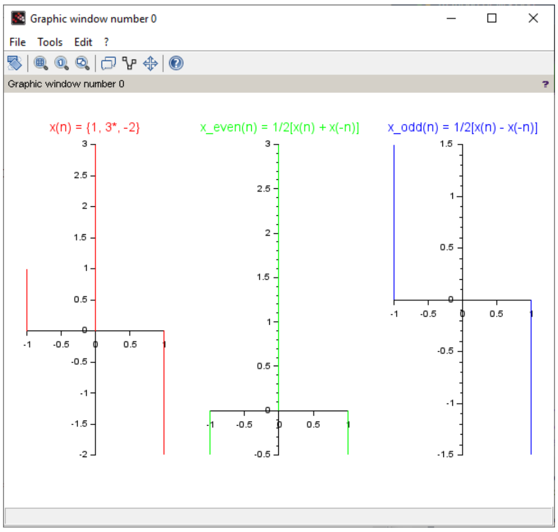


**Exercise 5** Given a discrete-time signal $\boldsymbol{x(n) = \{1, 3\uparrow, -2\}}$. Use Scilab to draw the signal x(n), the odd signal component $x_o(n)$ and the even signal component $x_e(n)$. Each signal will be drawed by a single plot but they are displayed in a single window. Please use title(), xlabel() and ylabel() to represent the name of each plot.

Code:

```
clear; // Clear old memories
function set_origin()
    // Set x-axis and y-axis at y = 0 and x = 0
    a = gca();
    a.x_location = "origin";
    a.y_location = "origin";
endfunction
n = -1:1; // Index with non-zero value of x
x = [1 3 -2];
max_n = max(n);
min_n = min(n);
max_amplitude = max([max_n, -min_n]);
n = -max_amplitude:max_amplitude;
ext_x_len = 1; // Values of signal x from -max_amplitude to max_amplitude
runner_n = -max_amplitude;
while runner_n < min_n
    runner_n = runner_n + 1;
    ext_x(ext_x_len) = 0;
    ext_x_len = ext_x_len + 1
end
x_index = 1;
while runner_n <= max_n
    runner_n = runner_n + 1;
    ext_x(ext_x_len) = x(x_index);
    ext_x_len = ext_x_len + 1;
    x_index = x_index + 1;
end
while runner_n <= max_amplitude
    runner_n = runner_n + 1;
    ext_x(ext_x_len) = 0;
    ext_x_len = ext_x_len + 1;
end
// Calculate x_fold(n) = x(-n)
x_fold = flipdim(ext_x, 1)

// Calculate x_e, x_o
x_even = 1/2 * (ext_x + x_fold);
x_odd = 1/2 * (ext_x - x_fold);
// Draw x(n)
subplot(131);
plot2d3(n, ext_x, 5);
set_origin;
xlabel("color", "red");
ylabel("color", "red");
title("x(n) = {1, 3*, -2}", "color", "red");
// Draw x_even
subplot(132);
plot2d3(n, x_even, 3);
set_origin;
xlabel("color", "green");
ylabel("color", "green");
title("x_even(n) = 1/2[x(n) + x(-n)]", "color", "green");
// Draw x_odd
subplot(133);
plot2d3(n, x_odd, 2);
set_origin;
xlabel("color", "blue");
ylabel("color", "blue");
title("x_odd(n) = 1/2[x(n) - x(-n)]", "color", "blue");
```

Result:




**Exercise 6** Given two discrete-time signals

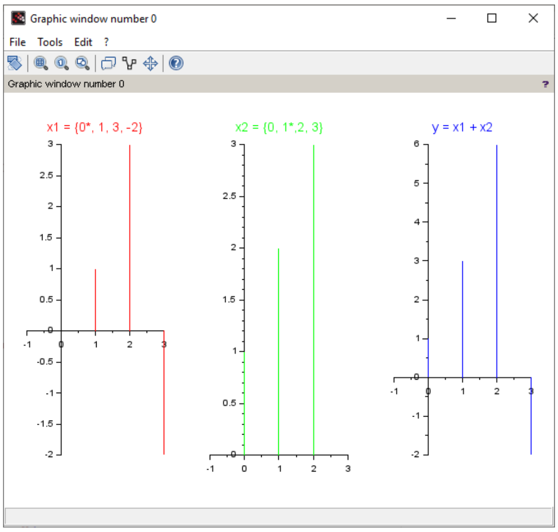
$$\boldsymbol{x_1(n) = \{0\uparrow, 1, 3, -2\}} \text{ and } \boldsymbol{x_2(n) = \{0, 1\uparrow, 2, 3\}}.$$

Determine $\boldsymbol{y(n) = x_1(n) + x_2(n)}$. Then use Scilab to draw $\boldsymbol{x_1(n), x_2(n)}$ and $\boldsymbol{y(n)}$. Each signal will be drawed by a single plot but they are displayed in a single window. Please use title(), xlabel() and ylabel() to represent the name of each plot.

Code:

```
clear
function set_origin()
    // Set x-axis and y-axis at x = 0 and y = 0
    a = gca();
    a.x_location = "origin";
    a.y_location = "origin";
endfunction
n = -1:3
x1 = [0 0 1 3 -2]
x2 = [0 1 2 3 0]
y = x1 + x2
subplot(131)
plot2d3(n, x1, 5)
set_origin
xlabel("color", "red")
ylabel("color", "red")
title("x1 = {0*, 1, 3, -2}", "color", "red")

subplot(132)
plot2d3(n, x2, 3)
set_origin
xlabel("color", "green")
ylabel("color", "green")
title("x2 = {0, 1*,2, 3}", "color", "green")

subplot(133)
plot2d3(n, y, 2)
set_origin
xlabel("color", "blue")
ylabel("color", "blue")
title("y = x1 + x2", "color", "blue")
```

Result:




**Exercise 7** Given two discrete-time signals

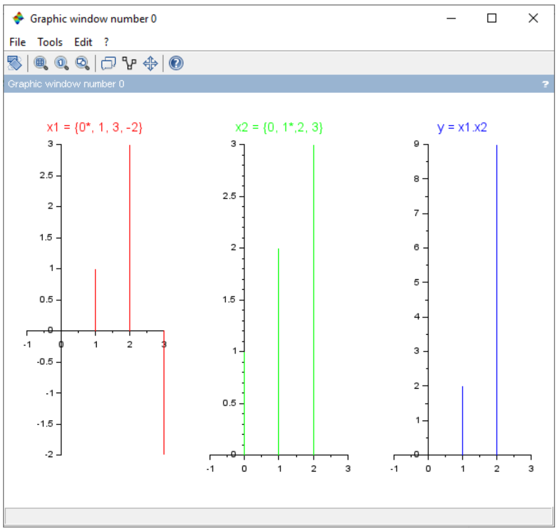
$$x_1(n) = \{0 \uparrow, 1, 3, -2\} \text{ and } x_2(n) = \{0, 1 \uparrow, 2, 3\}.$$

Determine $y(n) = x_1(n).x_2(n)$. Then use Scilab to draw $x_1(n), x_2(n)$ and $y(n)$. Each signal will be drawed by a single plot but they are displayed in a single window. Please use title(), xlabel() and ylabel() to represent the name of each plot.

Code:

```
clear
function set_origin()
    // Set x-axis and y-axis at x = 0 and y = 0
    a = gca();
    a.x_location = "origin";
    a.y_location = "origin";
endfunction
n = -1:3
x1 = [0 0 1 3 -2]
x2 = [0 1 2 3 0]
y = x1 .* x2
subplot(131)
plot2d3(n, x1, 5)
set_origin;
xlabel("color", "red")
ylabel("color", "red")
title("x1 = {0*, 1, 3, -2}", "color", "red")

subplot(132)
plot2d3(n, x2, 3)
set_origin;
xlabel("color", "green")
ylabel("color", "green")
title("x2 = {0, 1*,2, 3}", "color", "green")

subplot(133)
plot2d3(n, y, 2)
set_origin;
xlabel("color", "blue")
ylabel("color", "blue")
title("y = x1.x2", "color", "blue")
```

Result:




**Exercise 8** Given a discrete-time signal $x(n) = \{1, -2, 3 \uparrow, 6\}$. Determine the following signal and then use Scilab to draw the original signal $x(n)$ and the manipulated signal $y_i(n)$. Each pair of plots will be display in a single window. Please use title(), xlabel() and ylabel() to represent the name of each plot.

a. $y_1(n) = x(-n)$

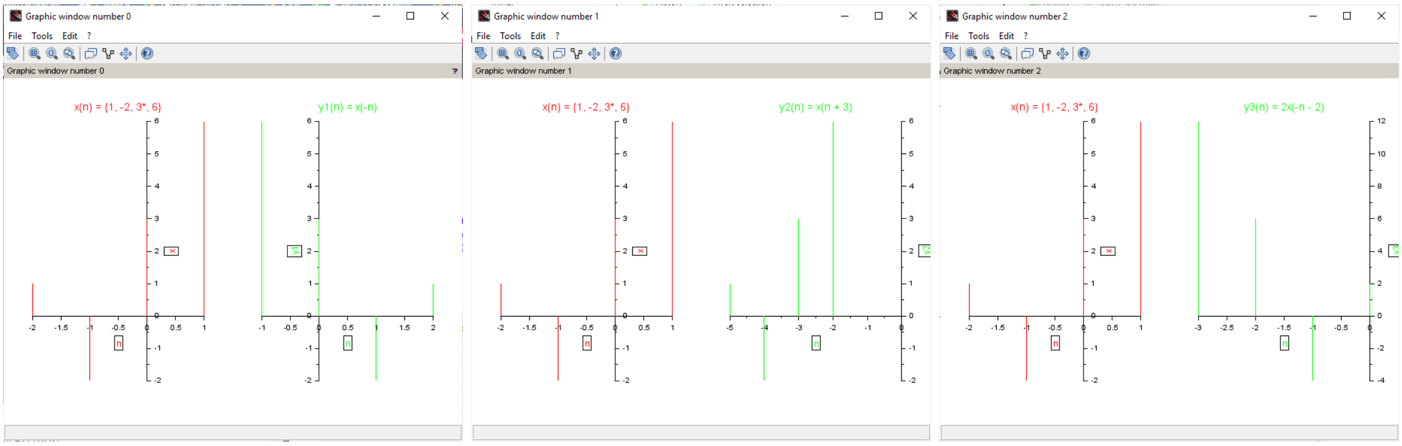
b. $y_2(n) = x(n + 3)$

c. $y_3(n) = 2x(-n - 2)$

Code:

```
clear;
// Set x-axis and y-axis at x = 0 and y = 0
function set_origin()
    a = gca();
    a.x_location = "origin";
    a.y_location = "origin";
endfunction

n = -2:1;
x = [1 -2 3 6];

// Draw original signal
function draw_original()
    subplot(121);
    plot2d3(n, x, 5);
    set_origin;
    xlabel("n", "color", "red", "box", "on");
    ylabel("x", "color", "red", "box", "on");
    title("x(n) = {1, -2, 3*, 6}", "color", "red");
endfunction

draw_original;
// Draw y1(n) = x(-n)
subplot(122);
plot2d3(-n, x, 3);
set_origin;
xlabel("n", "color", "green", "box", "on");
ylabel("y1", "color", "green", "box", "on");
title("y1(n) = x(-n)", "color", "green");

scf;
draw_original;
// Draw y2(n) = x(n + 3)
subplot(122);
plot2d3(n - 3, x, 3);
set_origin;
xlabel("n", "color", "green", "box", "on");
ylabel("y2", "color", "green", "box", "on");
title("y2(n) = x(n + 3)", "color", "green");

scf;
draw_original;
// Draw y3(n) = 2x(-n - 2)
subplot(122);
plot2d3(-n - 2, 2*x, 3);
set_origin;
xlabel("n", "color", "green", "box", "on");
ylabel("y3", "color", "green", "box", "on");
title("y3(n) = 2x(-n - 2)", "color", "green");
```

Result:




(a) (b) (c)

## 4.3 Lab 3

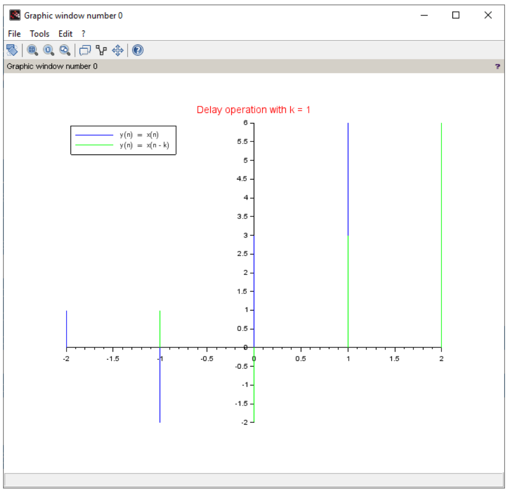
**Exercise 1.** function [yn, yorigin] = delay (xn, xorigin, k) performs delay operation y(n) = x(n - k), where k > 0,

- The discrete-time signal x(n) is presented by vector xn
- xorigin indicates the origin's position of signal x(n).
- yn is vector of the output signal
- yorigin indicates the origin's position of signal y(n).
- x(n) and y(n) are graphically displayed in the same figure.

Code:

```
function set_origin()
    a = gca()
    a.x_location = "origin";
    a.y_location = "origin";
endfunction

function [y_n, y_origin] = delay(x_n, x_origin, k)
    if (k < 0) then error('k must be greater than 0');
    end
    N = length(x_n);
    y_origin = x_origin - k;
    y_n = x_n(1:N);

    //Plot
    clf();
    x_padded = [x_n, zeros(1, k)];
    y_padded = [zeros(1, k), y_n];
    N_padded = length(x_padded);
    n = [1:N_padded] - x_origin;
    set_origin();
    plot2d3(n, x_padded, 2);
    plot2d3(n, y_padded, 3);
    legend(prettyprint(["y(n) = x(n)";"y(n) = x(n - k)"],"latex","",%t),"n", "m", 2);
    title("Delay operation with k = " + string(k), "color", "red");
endfunction

x_n = [1, -2, 3, 6];
x_origin = 3;
k = 1;
[y_n, y_origin] = delay(x_n, x_origin, k);
```

Result:



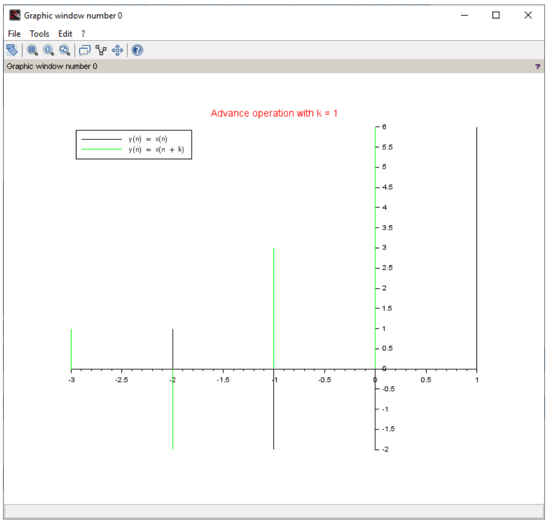

**Exercise 2.** function [yn, yorigin] = advance (xn, xorigin, k) performs advance operation y(n) = x(n + k), where k > 0 and

- The discrete-time signal x(n) is presented by vector xn
- xorigin indicates the origin's position of signal x(n).
- yn is vector of the output signal
- yorigin indicates the origin's position of signal y(n).
- x(n) and y(n) are graphically displayed in the same figure.

Code:

```
function set_origin()
    a = gca()
    a.x_location = "origin";
    a.y_location = "origin";
endfunction

function [y_n, y_origin] = advance(x_n, x_origin, k)
    if (k < 0) then error('k must be greater than 0');
    end
    N = length(x_n);
    y_origin = x_origin + k;
    y_n = x_n(1:N);
    //Plot
    clf();
    x_padded = [zeros(1, k), x_n];
    y_padded = [y_n, zeros(1, k)];
    N_padded = length(x_padded);
    n = [1:N_padded] - x_origin - k;
    set_origin();
    plot2d3(n, x_padded, 2);
    plot2d3(n, y_padded, 3);
    legend(prettyprint(["y(n) = x(n)";"y(n) = x(n + k)"],"latex","",%t), 2);
    title("Advance operation with k = " + string(k), "color", "red");
endfunction

[yn, yorigin] = advance ([1, -2, 3, 6], 3, 1);
```

Result:




**Exercise 3.** function [yn, yorigin] = fold (xn, xorigin, k) performs folding operation y(n) = x(-n), where

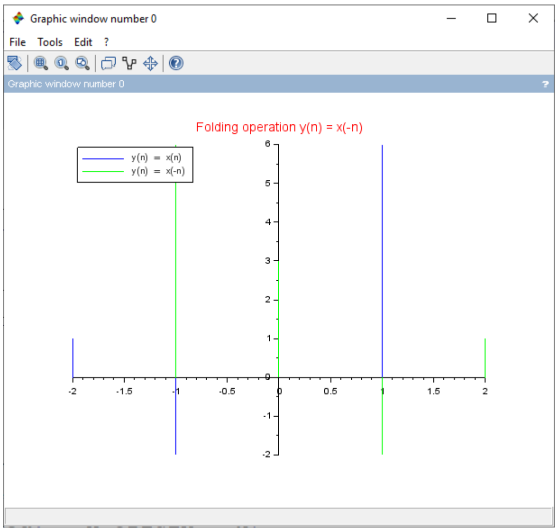
- The discrete-time signal x(n) is presented by vector xn
- xorigin indicates the origin's position of signal x(n).
- yn is vector of the output signal
- yorigin indicates the origin's position of signal y(n).

- x(n) and y(n) are graphically displayed in the same figure.

Code:

```
function set_origin()
    a = gca()
    a.x_location = "origin";
    a.y_location = "origin";
endfunction
function [y_n, y_origin] = fold(x_n, x_origin)
    N = length(x_n);
    y_origin = N - x_origin + 1;
    y_n = flipdim(x_n, 2);
    num_left = x_origin - 1;
    num_right = N - x_origin;
    k = num_left - num_right;
    if (k <= 0) then
        x_padded = [zeros(1, -k), x_n];
        y_padded = [y_n, zeros(1, -k)];
        N = length(x_padded);
        n = [1:N] - x_origin - (-k);
    else
        x_padded = [x_n, zeros(1, k)];
        y_padded = [zeros(1, k), y_n];
        N = length(x_padded);
        n = [1:N] - x_origin;
    end
    // Plot
    clf();
    set_origin();
    plot2d3(n, x_padded, 2);
    plot2d3(n, y_padded, 3);
    legend(prettyprint(["y(n) = x(n)";"y(n) = x(-n)"],"latex","",%t), 2);
    title("Folding operation y(n) = x(-n)", "color", "red");
endfunction

[yn, yorigin] = fold ([1, -2, 3, 6], 3);
```

Result:



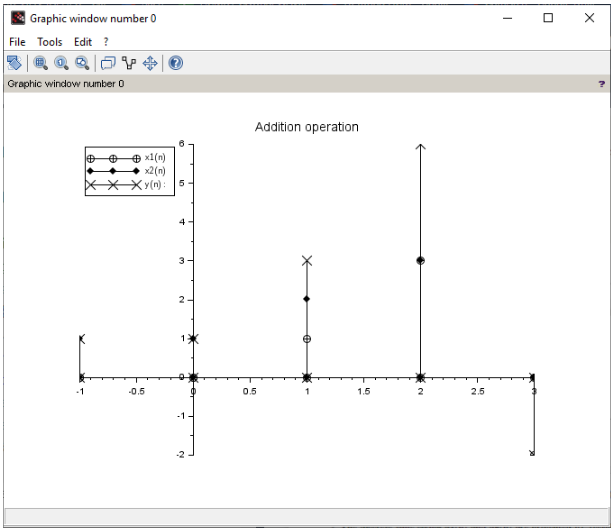

**Exercise 4.** function [yn, yorigin] = add (x1n, x1origin, x2n, x2origin) performs addition operation y(n) = x1(n) + x2(n), where

- The discrete-time signal x1(n) and x2(n) are presented by vectors x1n and x2n, respectively.
- x1origin and x2origin indicates the origin's position of signal x1(n) and x2(n), respectively.
- yn is vector of the output signal
- yorigin indicates the origin's position of signal y(n).
- x1(n), x2(n) and y(n) are graphically displayed in the same figure.

Code:

```
// Set x-axis and y-axis at x = 0 and y = 0
function set_origin()
    a = gca();
    a.x_location = "origin";
    a.y_location = "origin";
endfunction
function [yn, yorigin] = add(x1n, x1origin, x2n, x2origin)
    // Set x1n, x2n to the same size
    diff = length(x1n) - length(x2n);
    if diff > 0 then
        x2n = [x2n, zeros(1, diff)];
    else
        x1n = [x1n, zeros(1, -diff)];
    end
    // Add zeros to conveniently add two vector
    if x1origin > x2origin then
        padded = x1origin - x2origin;
        x2origin = x1origin;
        x1n = [x1n, zeros(1, padded)];
        x2n = [zeros(1, padded), x2n];
    else
        padded = x2origin - x1origin;
        x1origin = x2origin;
        x1n = [zeros(1, padded), x1n];
        x2n = [x2n, zeros(1, padded)];
    end
    yn = x1n + x2n;
    yorigin = x1origin;
    clf;
    set_origin;
    n = [1:length(yn)] - yorigin;
    plot2d3(n, x1n, -3);
    plot2d3(n, x2n, -4);
    plot2d3(n, yn, -2);
    legend(prettyprint(["x1(n)";"x2(n)";"y(n):"],"latex","",%t), 2);
    title("Addition Operation");
endfunction

x1n = [0, 1, 3, -2];
x1origin = 1;
x2n = [1, 1, 2, 3];
x2origin = 2;
[yn, yorigin] = add (x1n, x1origin, x2n, x2origin);
```

Result:



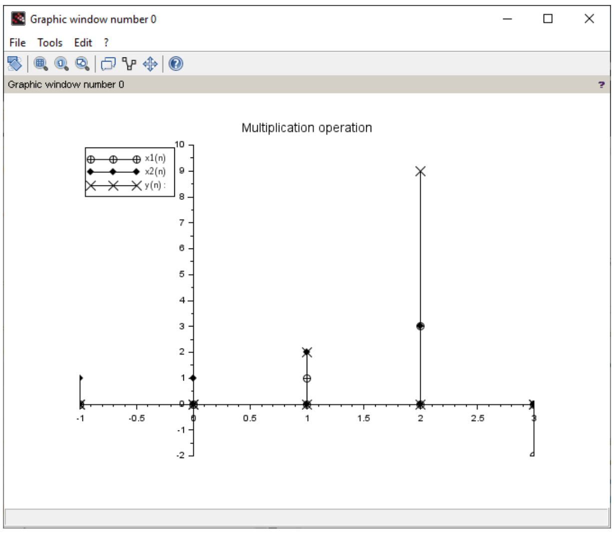

**Exercise 5.** function [yn, yorigin] = multi (x1n, x1origin, x2n, x2origin) performs multiplication operation y(n) = x1(n).x2(n), where

- The discrete-time signal x1(n) and x2(n) are presented by vectors x1n and x2n, respectively.
- x1origin and x2origin indicates the origin's position of signal x1(n) and x2(n), respectively.
- yn is vector of the output signal
- yorigin indicates the origin's position of signal y(n).
- x1(n), x2(n) and y(n) are graphically displayed in the same figure.

Code:

```
// Set x-axis and y-axis at x = 0 and y = 0
function set_origin()
    a = gca();
    a.x_location = "origin";
    a.y_location = "origin";
endfunction
function [yn, yorigin] = multi(x1n, x1origin, x2n, x2origin)
    // Set x1n, x2n to the same size
    diff = length(x1n) - length(x2n);
    if diff > 0 then
        x2n = [x2n, zeros(1, diff)];
    else
        x1n = [x1n, zeros(1, -diff)];
    end
    // Add zeros to conveniently multiply two vector
    if x1origin > x2origin then
        padded = x1origin - x2origin;
        x2origin = x1origin;
        x1n = [x1n, zeros(1, padded)];
        x2n = [zeros(1, padded), x2n];
    else
        padded = x2origin - x1origin;
        x1origin = x2origin;
        x1n = [zeros(1, padded), x1n];
        x2n = [x2n, zeros(1, padded)];
    end
    yn = x1n .* x2n;
    yorigin = x1origin;
    clf;
    set_origin;
    n = [1:length(yn)] - yorigin;
    plot2d3(n, x1n, -3);
    plot2d3(n, x2n, -4);
    plot2d3(n, yn, -2);
    legend(prettyprint(["x1(n)";"x2(n)";"y(n):"],"latex","",%t), 2);
    title("Multiplication operation");
endfunction

x1n = [0, 1, 3, -2];
x1origin = 1;
x2n = [1, 1, 2, 3];
x2origin = 2;
[yn, yorigin] = multi(x1n, x1origin, x2n, x2origin);
```

Result:



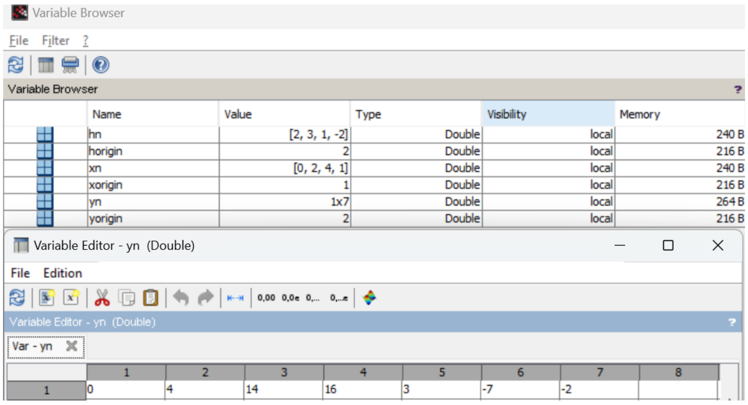

**Exercise 6.** function [yn, yorigin] = convolution (xn, xorigin, hn, horigin) performs convolution y(n) = x(n)*h(n), where

- x(n) is the input signal and h(n) is system characteristic's function.
- xorigin and horigin indicates the origin's position of x(n) and h(n), respectively.
- yn is vector of the output signal
- yorigin indicates the origin's position of signal y(n).

Code:

```
// Set x-axis and y-axis at x = 0 and y = 0
function set_origin()
    a = gca();
    a.x_location = "origin";
    a.y_location = "origin";
endfunction

function [yn, yorigin] = multi(x1n, x1origin, x2n, x2origin)
    // Set x1n, x2n to the same size
    diff = length(x1n) - length(x2n);
    if diff > 0 then
        x2n = [x2n, zeros(1, diff)];
    else
        x1n = [x1n, zeros(1, -diff)];
    end

    // Add zeros to conveniently multiply two vector
    if x1origin > x2origin then
        npad = x1origin - x2origin;
        x2origin = x1origin;
        x1n = [x1n, zeros(1, npad)];
        x2n = [zeros(1, npad), x2n];
    else
        npad = x2origin - x1origin;
        x1origin = x2origin;
        x1n = [zeros(1, npad), x1n];
        x2n = [x2n, zeros(1, npad)];
    end
    yn = x1n .* x2n;
    yorigin = x1origin;
endfunction

function [y_n, y_origin] = delay(x_n, x_origin, k)
    if (k < 0) then error('k must be greater than 0');
    end
    N = length(x_n);
    y_origin = x_origin - k;
    y_n = x_n(1:N);
endfunction

function [y_n, y_origin] = advance(x_n, x_origin, k)
    if (k < 0) then error('k must be greater than 0');
    end
    N = length(x_n);
    y_origin = x_origin + k;
    y_n = x_n(1:N);
endfunction

function [y_n, y_origin] = fold(x_n, x_origin)
    N = length(x_n);
    y_origin = N - x_origin + 1;
    y_n = flipdim(x_n, 2);
    num_left = x_origin - 1;
    num_right = N - x_origin;
    k = num_left - num_right;
    if (k <= 0) then
        x_padded = [zeros(1, -k), x_n];
        y_padded = [y_n, zeros(1, -k)];
        N = length(x_padded);
        n = [1:N] - x_origin - (-k);
    else
        x_padded = [x_n, zeros(1, k)];
        y_padded = [zeros(1, k), y_n];
        N = length(x_padded);
        n = [1:N] - x_origin;
    end
endfunction


function [yn, yorigin] = convolution(xn, xorigin, hn, horigin)
    nmin = -xorigin -horigin + 2;
    nmax = length(xn) - xorigin + length(hn) - horigin;
    for n0 = nmin : nmax
        if n0 < 0 then
            [h_shift, h_shift_org] = delay(hn, horigin, -n0);
        else
            [h_shift, h_shift_org] = advance(hn, horigin, n0);
        end
        [h_shift_fold, h_shift_fold_org] = fold(h_shift, h_shift_org);
        [mul_vec, mul_vec_org] = multi(xn, xorigin, h_shift_fold, h_shift_fold_org);
        yn($ + 1) = sum(mul_vec);
    end
    yn = yn';
    yorigin = -nmin + 1;
endfunction

hn = [2 3 1 -2]
horigin = 2
xn = [0 2 4 1]
xorigin = 1
[yn, yorigin] = convolution(xn, xorigin, hn, horigin)
```

Result:




## 4.4 Lab 4

**Exercise 1** Find the corresponding Z-Transfrom and ROC of this discrete-time signal x(n):

$$x(n) = 2\delta(n+2) - 1\delta(n+1) + 2\delta(n) - 3\delta(n-1) + 4\delta(n-2)$$

We have:

$X(z) = Z\{x(n)\} = \sum\limits_{n=-\infty}^{+\infty} x(n)z^{-n}$

$= \sum\limits_{n=-\infty}^{+\infty} (2\delta(n+2) - 1\delta(n+1) + 2\delta(n) - 3\delta(n-1) + 4\delta(n-2))z^{-n}$

$= 2z^2 - z^1 + 2z - 3z^{-1} + 4z^{-2}$

ROC: All $z$ except 0 or $\infty$

**Exercise 2** Find the corresponding Z-Transfrom and ROC of this discrete-time signal x(n):

$$x(n) = 0.5^n u(n) + 0.4^n u(n)$$

We have:

$X(z) = Z\{x(n)\} = \sum\limits_{n=-\infty}^{+\infty} x(n)z^{-n} = \sum\limits_{n=-\infty}^{+\infty} (0.5^n u(n) + 0.4^n u(n))z^{-n}$

$= \sum\limits_{n=-\infty}^{+\infty} 0.5^n u(n)z^{-n} + \sum\limits_{n=-\infty}^{+\infty} 0.4^n u(n)z^{-n} = \sum\limits_{n=0}^{+\infty} (0.5z^{-1})^n + \sum\limits_{n=0}^{+\infty} (0.4z^{-1})^n$

$= \frac{1}{1-0.5z^{-1}} + \frac{1}{1-0.4z^{-1}}$

ROC1: $|0.5z^{-1}| < 1 \iff |\text{z}| > 0.5$
ROC2: $|0.4z^{-1}| < 1 \iff |\text{z}| > 0.4$
ROC = ROC1 $\cap$ ROC2 $\Rightarrow$ ROC: $|\text{z}| > 0.5$

**Exercise 3** Find the corresponding Z-Transfrom and ROC of this discrete-time signal x(n):

$$x(n) = 0.5^n u(n) + 0.9^n u(-n-1)$$

We have:

$X(z) = Z\{x(n)\} = \sum\limits_{n=-\infty}^{+\infty} x(n)z^{-n} = \sum\limits_{n=-\infty}^{+\infty} (0.5^n u(n) + 0.9^n u(-n-1))z^{-n}$

$= \sum\limits_{n=-\infty}^{+\infty} 0.5^n u(n)z^{-n} + \sum\limits_{n=-\infty}^{+\infty} 0.9^n u(-n-1)z^{-n} = \sum\limits_{n=0}^{+\infty} (0.5z^{-1})^n + \sum\limits_{n=-\infty}^{-1} (0.9z^{-1})^n$

$= \sum\limits_{n=0}^{+\infty} (0.5z^{-1})^n + \sum\limits_{m=1}^{+\infty} (0.9^{-1}z)^m$

$= \frac{1}{1-0.5z^{-1}} + \frac{1}{1-0.9z^{-1}}$

ROC1: $|0.5z^{-1}| < 1 \iff |\text{z}| > 0.5$
ROC2: $|0.9^{-1}z| < 1 \iff |\text{z}| < 0.9$
ROC = ROC1 $\cap$ ROC2 $\Rightarrow$ ROC: $0.9 > |\text{z}| > 0.5$

**Exercise 4** Investigate the existing library or toolbox in Scilab to process audio data.

- **Sound Toolbox:** This toolbox provides various functions for working with sound signals, including reading and writing audio files in various formats (WAV, AU, SND), playing sounds, and basic signal processing operations like filtering, resampling, and spectral analysis.

- **SciCosLab**: While primarily a toolbox for system modeling and simulation, SciCosLab includes some audio processing blocks that can be used for designing and simulating audio processing systems. It might be useful for more advanced users interested in modeling complex audio processing systems.
- **Scilab-Arduino Toolbox:** If you're interested in real-time audio processing or audio processing with embedded systems, you might find the Scilab-Arduino Toolbox useful. This toolbox allows communication between Scilab and Arduino, which can be used to interface with audio sensors or actuators for real-time audio processing applications.

**Exercise 5** Investigate the existing library or or toolbox in Scilab for image processing. Then, you can do an example or a demo to show a simple manipulation (e.g., Histogram display, histogram equalization, blur, and water-masking) on a digital image.

- **SIVP-Scilab Image and Video Processing**: provides a wide range of functions and algorithms for analyzing, manipulating, and enhancing images and videos. With SIVP, users can perform operations such as image filtering, edge detection, morphological operations, image segmentation, feature extraction, and more. The toolbox also supports reading and writing images and videos in various formats.
- **ICPV - Image Processing and Computer Vision**: provides a comprehensive set of tools for analyzing, manipulating, and understanding images within the Scilab environment. It offers a wide range of functions and algorithms for both basic image processing tasks and advanced computer vision applications.

**Demo using ICPV to show histogram display on a digital image.**

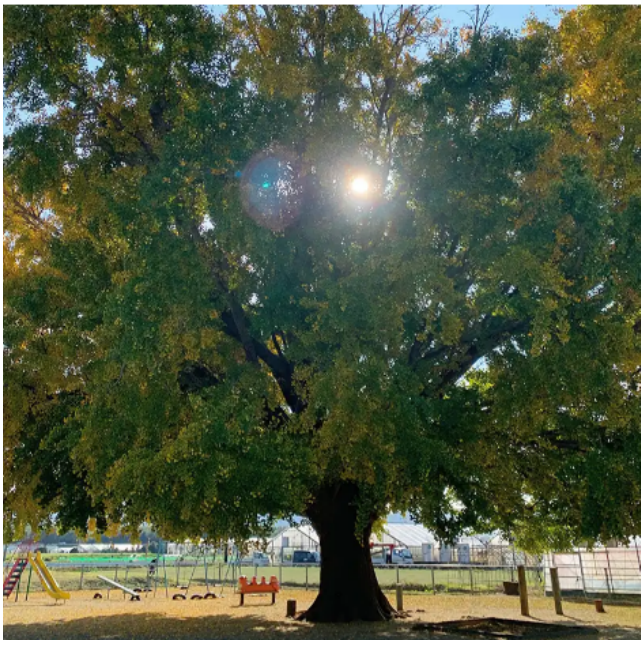
Code:

```
// Load an image
image = imread("image.png");

// Extract RGB Components
im_red = image(:, :, 1);
im_green = image(:, :, 2);
im_blue = image(:, :, 3);

// Convert image to grayscale
im_gray = rgb2gray(image);

// Get histogram of each component
red_hist = imhist(im_red);
green_hist = imhist(im_green);
blue_hist = imhist(im_blue);
gray_hist = imhist(im_gray);
```

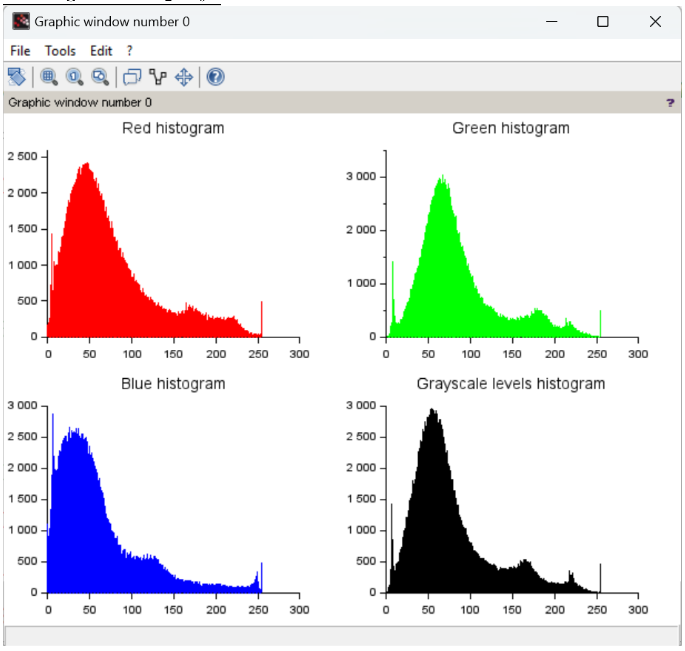
```
// Plot histogram
subplot(2,2,1);
plot2d3([0:255], red_hist, 5);
xtitle("Red histogram");

subplot(2,2,2);
plot2d3([0:255], green_hist, 3);
xtitle("Green histogram");

subplot(2,2,3);
plot2d3([0:255], blue_hist, 2);
xtitle("Blue histogram");

subplot(2,2,4);
plot2d3([0:255], gray_hist, 1);
xtitle("Grayscale levels histogram");
```

Result:

Digital image:



Histogram display:




## 4.5 Lab 5

**Exercise 1** Use Z-transform to find impulse response h(n) of the system represented by the following input-ouput description equation.

$$y(n) - y(n-2) = x(n)$$

Using Z-transform, we have:
$Y(z) - z^{-2}Y(z) = X(z)$
$\iff Y(z)(1 - z^{-2}) = X(z)$
$\iff H(z) = \frac{Y(z)}{X(z)} = \frac{1}{1-z^{-2}}$
$\iff \frac{H(z)}{z} = \frac{z}{z^2-1} = \frac{1}{2}\frac{1}{z-1} + \frac{1}{2}\frac{1}{z+1}$
$\iff H(z) = \frac{1}{2}\frac{1}{1-z^{-1}} + \frac{1}{2}\frac{1}{1+z^{-1}}$
We have:
$Z^{-1}\{\frac{1}{2}\frac{1}{1-z^{-1}}\} = \begin{cases} \frac{1}{2}1^n u(n), |z| > 1 \\ -\frac{1}{2}1^n u(-n-1), |z| < 1 \end{cases}$
$Z^{-1}\{\frac{1}{2}\frac{1}{1+z^{-1}}\} = \begin{cases} \frac{1}{2}(-1)^n u(n), |z| > 1 \\ -\frac{1}{2}(-1)^n u(-n-1), |z| < 1 \end{cases}$
Therefore:
With $|z| > 1 \Rightarrow h(n) = \frac{1}{2}1^n u(n) + \frac{1}{2}(-1)^n u(n)$
With $|z| < 1 \Rightarrow h(n) = -\frac{1}{2}1^n u(-n-1) - \frac{1}{2}(-1)^n u(-n-1)$

**Exercise 2** Use Z and $Z^{-1}$ transform to compute the convolution

a) $x_1(n) = \{1\uparrow, 2, 3, 4, 5\}$ and $x_2(n) = \{1\uparrow, 1, 1\}$
We have:
$x_1(n) = \{1\uparrow, 2, 3, 4, 5\} \Rightarrow X_1(z) = 1 + 2z^{-1} + 3z^{-2} + 4z^{-3} + 5z^{-4}$
$x_2(n) = \{1\uparrow, 1, 1\} \Rightarrow X_2(z) = 1 + z^{-1} + z^{-2}$
And:
$X(z) = X_1(z)X_2(z) = (1 + 2z^{-1} + 3z^{-2} + 4z^{-3} + 5z^{-4})(1 + z^{-1} + z^{-2})$
$\iff X(z) = 1 + 3z^{-1} + 6z^{-2} + 9z^{-3} + 12z^{-4} + 9z^{-5} + 5z^{-6}$
Therefore:
$x(n) = x_1(n) * x_2(n) = Z^{-1}\{X(z)\} = \{1\uparrow, 3, 6, 9, 12, 9, 5\}$

b) $x_1(n) = (\frac{1}{5})^n u(n)$ and $x_2(n) = 2^n u(n)$
We have:
$x_1(n) = (\frac{1}{5})^n u(n) \Rightarrow X_1(z) = \frac{1}{1-\frac{1}{5}z^{-1}}$ $\quad ROC_{X_1(z)} : |z| > \frac{1}{5}$
$x_2(n) = 2^n u(n) \Rightarrow X_2(z) = \frac{1}{1-2z^{-1}}$ $\quad ROC_{X_2(z)} : |z| > 2$

$X(z) = X_1(z)X_2(z) = \frac{1}{1-\frac{1}{5}z^{-1}}\frac{1}{1-2z^{-1}}$ $\quad ROC_{X(z)} = ROC_{X_1(z)} \cap ROC_{X_2(z)} \iff ROC_{X(z)} : |z| > 2$
$\iff X(z) = -\frac{1}{9}\frac{1}{1-\frac{1}{5}z^{-1}} + \frac{10}{9}\frac{1}{1-2z^{-1}}$
We have:
$Z^{-1}\{-\frac{1}{9}\frac{1}{1-\frac{1}{5}z^{-1}}\} = \begin{cases} -\frac{1}{9}(\frac{1}{5})^n u(n), |z| > \frac{1}{5} \\ \frac{1}{9}(\frac{1}{5})^n u(-n-1), |z| < \frac{1}{5} \end{cases}$
$Z^{-1}\{\frac{10}{9}\frac{1}{1-2z^{-1}}\} = \begin{cases} \frac{10}{9}2^n u(n), |z| > 2 \\ -\frac{10}{9}2^n u(-n-1), |z| < 2 \end{cases}$
Therefore: $x(n) = x_1(n) * x_2(n) = Z^{-1}\{X(z)\} = -\frac{1}{9}(\frac{1}{5})^n u(n) + \frac{10}{9}2^n u(n)$ (because $ROC_{X(z)} : |z| > 2$)

c) $x_1(n) = nu(n)$ and $x_2(n) = 2^n u(n-1)$
We have:
$x_1(n) = nu(n) = n1^n u(n) \Rightarrow X_1(z) = \frac{z^{-1}}{(1-z^{-1})^2} = \frac{z}{(z-1)^2}$ $\quad ROC_{X_1(z)} : |z| > 1$
$x_2(n) = 2^n u(n-1) = 2.2^{n-1}u(n-1) \Rightarrow X_2(z) = 2.z^{-1}.\frac{1}{1-2z^{-1}} = \frac{2}{z-2}$ $\quad ROC_{X_2(z)} : |z| > 2$
$X(z) = X_1(z)X_2(z) = \frac{2z}{(z-1)^2(z-2)}$ $\quad ROC_{X(z)} : |z| > 2$
$\iff \frac{X(z)}{z} = \frac{2}{z-2} - \frac{2}{z-1} - \frac{2}{(z-1)^2}$
$\iff X(z) = \frac{2}{1-2z^{-1}} - \frac{2}{1-z^{-1}} - \frac{2z^{-1}}{(1-z^{-1})^2}$
We have:
$Z^{-1}\{\frac{2}{1-2z^{-1}}\} = \begin{cases} 2.2^n.u(n), |z| > 2 \\ 2.(-(2)^n.u(-n-1)), |z| < 2 \end{cases}$
$Z^{-1}\{\frac{-2}{1-z^{-1}}\} = \begin{cases} -2.u(n), |z| > 1 \\ -2.[-u(-n-1)], |z| < 1 \end{cases}$
$Z^{-1}\{\frac{-2z^{-1}}{(1-z^{-1})^2}\} = \begin{cases} -2.n.u(n), |z| > 1 \\ -2.[-n.u(-n-1)], |z| < 1 \end{cases}$
Therefore: x(n) $= 2.2^n.u(n) - 2.u(n) - 2.n.u(n)$ (because $ROC_{X(z)} : |z| > 2$)

**Exercise 3** Find all possible x(n) that has $Z^{-1}$ transform as follows

a) $X_1(z) = \frac{1}{2-3z^{-1}+z^{-2}}$
$X_1(z) = \frac{1}{2-3z^{-1}+z^{-2}} = \frac{1}{(2-z^{-1})(1-z^{-1})} = \frac{1}{1-z^{-1}} - \frac{1}{2-z^{-1}} = \frac{1}{1-z^{-1}} - \frac{1}{2}\frac{1}{1-\frac{1}{2}z^{-1}}$
We have:
$Z^{-1}\{\frac{1}{1-z^{-1}}\} = \begin{cases} 1^n u(n), |z| > 1 \\ -1^n u(-n-1), |z| < 1 \end{cases}$
$Z^{-1}\{-\frac{1}{2}\frac{1}{1-\frac{1}{2}z^{-1}}\} = \begin{cases} -\frac{1}{2}(\frac{1}{2})^n u(n), |z| > \frac{1}{2} \\ \frac{1}{2}(\frac{1}{2})^n u(-n-1), |z| < \frac{1}{2} \end{cases}$
Therefore:
With $|z| < \frac{1}{2} \Rightarrow x_1(n) = -1^n u(-n-1) + \frac{1}{2}(\frac{1}{2})^n u(-n-1)$
With $\frac{1}{2} < |z| < 1 \Rightarrow x_1(n) = -1^n u(-n-1) - \frac{1}{2}(\frac{1}{2})^n u(n)$
With $|z| > 1 \Rightarrow x_1(n) = 1^n u(n) - \frac{1}{2}(\frac{1}{2})^n u(n)$

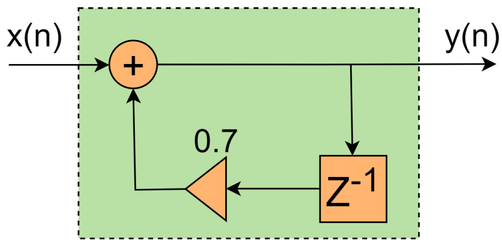
b) $X_2(z) = \frac{1+2z^{-1}+z^{-2}}{1+4z^{-1}+4z^{-2}}$
$X_2(z) = \frac{1+2z^{-1}+z^{-2}}{1+4z^{-1}+4z^{-2}} = \frac{\frac{1}{4}(1+4z^{-1}+4z^{-2})+\frac{3}{4}(1+2z^{-1})-\frac{1}{2}z^{-1}}{1+4z^{-1}+4z^{-2}} = \frac{1}{4} + \frac{3}{4}\frac{1}{1+2z^{-1}} - \frac{1}{2}\frac{z^{-1}}{(1+2z^{-1})^2} = \frac{1}{4} + \frac{3}{4}\frac{1}{1+2z^{-1}} + \frac{1}{4}\frac{(-2)z^{-1}}{(1+2z^{-1})^2}$
We have:
$Z^{-1}\{\frac{1}{4}\} = \frac{1}{4}\delta(n)$, All z
$Z^{-1}\{\frac{3}{4}\frac{1}{1+2z^{-1}}\} = \begin{cases} \frac{3}{4}(-2)^n u(n), |z| > 2 \\ -\frac{3}{4}(-2)^n u(-n-1), |z| < 2 \end{cases}$
$Z^{-1}\{\frac{1}{4}\frac{(-2)z^{-1}}{(1+2z^{-1})^2}\} = \begin{cases} \frac{1}{4}n(-2)^n u(n), |z| > 2 \\ -\frac{1}{4}n(-2)^n u(-n-1), |z| < 2 \end{cases}$
Therefore:
With $|z| < 2 \Rightarrow x_2(n) = \frac{1}{4}\delta(n) - \frac{3}{4}(-2)^n u(-n-1) - \frac{1}{4}n(-2)^n u(-n-1)$
With $|z| > 2 \Rightarrow x_2(n) = \frac{1}{4}\delta(n) + \frac{3}{4}(-2)^n u(n) + \frac{1}{4}n(-2)^n u(n)$

c) $X_3(z) = \frac{1+z^{-1}}{(1-0.5z^{-1})^2(1-0.3z^{-1})} 0.3 < z < 0.5$
$X_3(z) = \frac{1+z^{-1}}{(1-0.5z^{-1})^2(1-0.3z^{-1})} = \frac{z^3+z^2}{(z-0.5)^2(z-0.3)}$
$\iff \frac{X_3(z)}{z} = \frac{z^2+z}{(z-0.5)^2(z-0.3)} = \frac{39}{4}\frac{1}{z-0.3} - \frac{35}{4}\frac{1}{z-0.5} + \frac{15}{4}\frac{1}{(z-0.5)^2}$
$\iff X(z) = \frac{39}{4}\frac{1}{1-0.3z^{-1}} - \frac{35}{4}\frac{1}{1-0.5z^{-1}} + \frac{15}{4}\frac{z^{-1}}{(1-0.5z^{-1})^2} = \frac{39}{4}\frac{1}{1-0.3z^{-1}} - \frac{35}{4}\frac{1}{1-0.5z^{-1}} + \frac{15}{2}\frac{0.5z^{-1}}{(1-0.5z^{-1})^2}$
We have:
$Z^{-1}\{\frac{39}{4}\frac{1}{1-0.3z^{-1}}\} = \begin{cases} \frac{39}{4}(0.3)^n u(n), |z| > 0.3 \\ -\frac{39}{4}(0.3)^n u(-n-1), |z| < 0.3 \end{cases}$
$Z^{-1}\{-\frac{35}{4}\frac{1}{1-0.5z^{-1}}\} = \begin{cases} -\frac{35}{4}(0.5)^n u(n), |z| > 0.5 \\ \frac{35}{4}(0.5)^n u(-n-1), |z| < 0.5 \end{cases}$
$Z^{-1}\{\frac{15}{2}\frac{0.5z^{-1}}{(1-0.5z^{-1})^2}\} = \begin{cases} \frac{15}{2}n(0.5)^n u(n), |z| > 0.5 \\ -\frac{15}{2}n(0.5)^n u(-n-1), |z| < 0.5 \end{cases}$
Therefore, with $0.3 < z < 0.5 : x_3(n) = \frac{39}{4}(0.3)^n u(n) + \frac{35}{4}(0.5)^n u(-n-1) - \frac{15}{2}n(0.5)^n u(-n-1)$

**Exercise 4** Given LTI system by the following input-ouput description equation

$$y(n) = 0.7y(n-1) + x(n)$$

a) Draw the block diagram of the above system




b) Determine $h(n)$
Using Z-transform, we have:
$Y(z) = 0.7z^{-1}Y(z) + X(z)$
$\iff Y(z)(1-0.7z^{-1}) = X(z)$
$\iff H(z) = \frac{Y(z)}{X(z)} = \frac{1}{1-0.7z^{-1}}$
We have:
$Z^{-1}\{\frac{1}{1-0.7z^{-1}}\} = \begin{cases} 0.7^n u(n), |z| > 0.7 \\ -0.7^n u(-n-1), |z| < 0.7 \end{cases}$

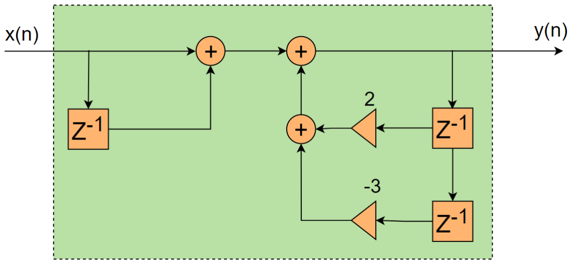
Therefore:
With $|z| > 0.7 \Rightarrow h(n) = 0.7^n u(n)$
With $|z| < 0.7 \Rightarrow h(n) = -0.7^n u(-n-1)$

c) Determine $y(n)$ when $x(n) = u(n)$
When $x(n) = u(n) \iff X(z) = \frac{1}{1-z^{-1}}$, using Z-transform we have:
$Y(z) = 0.7z^{-1}Y(z) + \frac{1}{1-z^{-1}}$
$\iff Y(z) = \frac{1}{(1-z^{-1})(1-0.7z^{-1})}$
$\iff Y(z) = \frac{10}{3}\frac{1}{1-z^{-1}} - \frac{7}{3}\frac{1}{1-0.7z^{-1}}$
We have:
$Z^{-1}\{\frac{10}{3}\frac{1}{1-z^{-1}}\} = \begin{cases} \frac{10}{3}1^n u(n), |z| > 1 \\ -\frac{10}{3}1^n u(-n-1), |z| < 1 \end{cases}$
$Z^{-1}\{-\frac{7}{3}\frac{1}{1-0.7z^{-1}}\} = \begin{cases} -\frac{7}{3}0.7^n u(n), |z| > 0.7 \\ \frac{7}{3}0.7^n u(-n-1), |z| < 0.7 \end{cases}$
Therefore:
With $|z| < 0.7 \Rightarrow y(n) = -\frac{10}{3}1^n u(-n-1) + \frac{7}{3}0.7^n u(-n-1)$
With $0.7 < |z| < 1 \Rightarrow y(n) = -\frac{10}{3}1^n u(-n-1) - \frac{7}{3}0.7^n u(n)$
With $|z| > 1 \Rightarrow y(n) = \frac{10}{3}1^n u(n) - \frac{7}{3}0.7^n u(n)$

**Exercise 5** Given LTI system by the following input-ouput description equation

$$y(n) = 2y(n-1) - 3y(n-2) + x(n) + x(n-1)$$

a) Draw the block diagram of the above system




b) Determine the impulse response $h(n)$
Using Z-transform, we have:
$Y(z) = 2z^{-1}Y(z) - 3z^{-2}Y(z) + X(z) + z^{-1}X(z)$
$\iff Y(z)(1 - 2z^{-1} + 3z^{-2}) = X(z)(1 + z^{-1})$
$\iff H(z) = \frac{Y(z)}{X(z)} = \frac{1+z^{-1}}{1-2z^{-1}+3z^{-2}} = \frac{z^2+z}{z^2-2z+3}$
$\Longrightarrow \frac{H(z)}{z} = \frac{z+1}{z^2-2z+3} = \frac{z+1}{(z-p_1)(z-p_2)}$ with $p_1 = 1 + j\sqrt{2}, p_2 = 1 - j\sqrt{2}$
$\iff \frac{H(z)}{z} = \frac{A_1}{z-p_1} + \frac{A_2}{z-p_2}$ with $A_1 = \frac{1}{2} - j\frac{\sqrt{2}}{2}, A_2 = \frac{1}{2} + j\frac{\sqrt{2}}{2}$
$\iff H(z) = \frac{A_1}{1-p_1z^{-1}} + \frac{A_2}{1-p_2z^{-1}}$
We have:
$Z^{-1}\{\frac{A_1}{1-p_1z^{-1}}\} = \begin{cases} A_1(p_1)^n u(n), |z| > |p_1| = \sqrt{3} \\ -A_1(p_1)^n u(-n-1), |z| < |p_1| = \sqrt{3} \end{cases}$
$Z^{-1}\{\frac{A_2}{1-p_2z^{-1}}\} = \begin{cases} A_2(p_2)^n u(n), |z| > |p_2| = \sqrt{3} \\ -A_2(p_2)^n u(-n-1), |z| < |p_2| = \sqrt{3} \end{cases}$
Therefore:
With $|z| > \sqrt{3} \Rightarrow h(n) = A_1(p_1)^n u(n) + A_2(p_2)^n u(n)$
With $|z| < \sqrt{3} \Rightarrow h(n) = -A_1(p_1)^n u(-n-1) - A_2(p_2)^n u(-n-1)$

c) Determine $y_{zi}(n)$ when $y(-1) = y(-2) = 1$

d) Determine $y_{zs}(n)$ when $x(n) = 2^n u(n)$
Solution for c) and d): Using One-sided Z-transform, we have:
$Y^+(z) = 2z^{-1}[Y^+(z) + y(-1)z] - 3z^{-2}[Y^+(z) + y(-1)z + y(-2)z^2] + X^+(z) + z^{-1}[X^+(z) + x(-1)z]$
$\iff Y^+(z)(1 - 2z^{-1} + 3z^{-2}) = 2y(-1) - 3y(-2) + x(-1) - 3z^{-1}y(-1) + X^+(z)(1 + z^{-1})$
$\iff Y^+(z) = \frac{2y(1)-3y(-2)+x(-1)-3z^{-1}y(-1)}{1-2z^{-1}+3z^{-2}} + \frac{1+z^{-1}}{1-2z^{-1}+3z^{-2}}X^+(z) = \frac{-1-3z^{-1}}{1-2z^{-1}+3z^{-2}} + \frac{1+z^{-1}}{1-2z^{-1}+3z^{-2}}X^+(z) = Y^+_{zi} + Y^+_{zs}$
$*Y^+_{zi} = \frac{-1-3z^{-1}}{1-2z^{-1}+3z^{-2}} = \frac{A'_1}{1-p_1z^{-1}} + \frac{A'_2}{1-p_2z^{-1}}$ with $p_1 = 1 + j\sqrt{2}, p_2 = 1 - j\sqrt{2}, A'_1 = -\frac{1}{2} + j\sqrt{2}, A'_2 = -\frac{1}{2} - j\sqrt{2}$
We have:

$Z^{-1}\{\frac{A_1'}{1-p_1z^{-1}}\} = \begin{cases} A_1'(p_1)^n u(n), |z| > |p_1| = \sqrt{3} \\ -A_1'(p_1)^n u(-n-1), |z| < |p_1| = \sqrt{3} \end{cases}$

$Z^{-1}\{\frac{A_2'}{1-p_2z^{-1}}\} = \begin{cases} A_2'(p_2)^n u(n), |z| > |p_2| = \sqrt{3} \\ -A_2'(p_2)^n u(-n-1), |z| < |p_2| = \sqrt{3} \end{cases}$

Therefore:

With $|z| > \sqrt{3} \Rightarrow y_{zi}(n) = (-\frac{1}{2} + j\sqrt{2})(1 + j\sqrt{2})^n u(n) + (-\frac{1}{2} - j\sqrt{2})(1 - j\sqrt{2})^n u(n)$

With $|z| < \sqrt{3} \Rightarrow y_{zi}(n) = -(-\frac{1}{2} + j\sqrt{2})(1 + j\sqrt{2})^n u(-n-1) - (-\frac{1}{2} - j\sqrt{2})(1 - j\sqrt{2})^n u(-n-1)$

*$Y_{zs}^+(z) = \frac{1+z^{-1}}{1-2z^{-1}+3z^{-2}} X^+(z) = \frac{1+z^{-1}}{1-2z^{-1}+3z^{-2}} \frac{1}{1-2z^{-1}}$(because $x(n) = 2^n u(n) \xrightarrow{Z^+} X^+(z) = \frac{1}{1-2z^{-1}}$ $ROC : |z| > 2$)

$\iff Y_{zs}^+(z) = \frac{A_1''}{1-p_1z^{-1}} + \frac{A_2''}{1-p2z^{-1}} + \frac{2}{1-2z^{-1}}$ with $p_1 = 1 + j\sqrt{2}, p_2 = 1 - j\sqrt{2}, A_1'' = -\frac{1}{2} - j\frac{\sqrt{2}}{2}, A_2'' = -\frac{1}{2} + j\frac{\sqrt{2}}{2}$

We have:

$Z^{-1}\{\frac{A_1''}{1-p_1z^{-1}}\} = \begin{cases} A_1''(p_1)^n u(n), |z| > |p_1| = \sqrt{3} \\ -A_1''(p_1)^n u(-n-1), |z| < |p_1| = \sqrt{3} \end{cases}$

$Z^{-1}\{\frac{A_2''}{1-p_2z^{-1}}\} = \begin{cases} A_2''(p_2)^n u(n), |z| > |p_2| = \sqrt{3} \\ -A_2''(p_2)^n u(-n-1), |z| < |p_2| = \sqrt{3} \end{cases}$

$Z^{-1}\{\frac{2}{1-2z^{-1}}\} = \begin{cases} 2^{n+1} u(n), |z| > 2 \\ -2^{n+1} u(-n-1), |z| < 2 \end{cases}$

With $|z| > 2 \Rightarrow y_{zs} = (-\frac{1}{2} - j\frac{\sqrt{2}}{2})(1 + j\sqrt{2})^n u(n) + (-\frac{1}{2} + j\frac{\sqrt{2}}{2})(1 - j\sqrt{2})^n u(n) + 2^{n+1}u(n)$

**Exercise 6:** If $x(n) \xrightarrow{Z} X(z)$. Then, prove the following statements:

a) $Z[x^*(n)] = X^*(z^*)$

We have: $Z[x^*(n)] = \sum\limits_{n=-\infty}^{+\infty} x^*(n)z^{-n} = \sum\limits_{n=i\infty}^{+\infty} [x(n)(z^*)^{-n}]^* = X^*(z^*)$

b) $Z\{\Re[x(n)]\} = \frac{1}{2}[X(z) + X^*(z^*)]$

We have: $\frac{1}{2}[X(z) + X^*(z^*)] = \frac{1}{2}[Z\{x(n)\} + Z\{x^*(n)\}] = Z[\frac{x(n)+x^*(n)}{2}] = Z\{\Re[x(n)]\}$

c) $Z\{\Im[x(n)]\} = \frac{1}{2j}[X(z) - X^*(z^*)]$

We have: $\frac{1}{2j}[X(z) - X^*(z^*)] = \frac{1}{2j}[Z\{x(n)\} + Z\{x^*(n)\}] = Z[\frac{x(n)-x^*(n)}{2j}] = Z\{\Im[x(n)]\}$

d) $Z\{e^{j\omega_0 n}\} = X\{ze^{-j\omega_0}\}$

We have: $Z\{e^{j\omega_0 n}\} = \sum\limits_{n=-\infty}^{+\infty} e^{j\omega_0 n}x(n)z^{-n} = \sum\limits_{n=-\infty}^{+\infty} x(n)(e^{-j\omega_0}z)^{-n} = X\{ze^{-j\omega_0 n}\}$

## 4.6 Lab 6

**Exercise 1:** Find Fourier transform of the following signals

a) $x_1(t) = \begin{cases} 1 - \frac{|t|}{\tau} & |t| \le \tau \\ 0 & |t| > \tau \end{cases}$

We have: $X_1(F) = \int\limits_{-\infty}^{+\infty} x(t)e^{-j2\pi Ft}dt \Rightarrow X_1(F) = \int\limits_{-\tau}^{+\tau} (1 - \frac{|t|}{\tau})e^{-j2\pi Ft}dt$

$\Rightarrow X_1(F) = \int\limits_{-\tau}^{0} (1 - \frac{|t|}{\tau})e^{-j2\pi Ft}dt + \int\limits_{0}^{+\tau} (1 - \frac{|t|}{\tau})e^{-j2\pi Ft}dt$

$\Rightarrow X_1(F) = \int\limits_{-\tau}^{0} (1 + \frac{t}{\tau})e^{-j2\pi Ft}dt + \int\limits_{0}^{+\tau} (1 - \frac{t}{\tau})e^{-j2\pi Ft}dt$

$\Rightarrow X_1(F) = \frac{1}{(j2\pi F)^2}e^{-j2\pi Ft}(\frac{-j2\pi Ft}{\tau} - j2\pi F - \frac{1}{\tau})\Big|_{-\tau}^{0} + \frac{1}{(j2\pi F)^2}e^{-j2\pi Ft}(\frac{j2\pi Ft}{\tau} - j2\pi F + \frac{1}{\tau})\Big|_{0}^{\tau}$

$\Rightarrow X_1(F) = \frac{1}{(j2\pi F)^2}e^{j2\pi F\tau}(e^{-j2\pi F\tau} - 1)^2$

b) $x_2(t) = e^{j\omega_0 t}$

We have: $X_2(F) = \int\limits_{t=-\infty}^{+\infty} e^{j\omega_0 t}e^{-j\omega t}dt = \int\limits_{t=-\infty}^{+\infty} e^{j(\omega_0-\omega)t}dt = \frac{e^{(j\omega_0-j\omega)t}}{j\omega_0-j\omega}\Big|_{-\infty}^{+\infty}$

**Exercise 2:** Find Fourier transform of the following signals

a) $x_1(n) = u(n) - u(n-6)$

We have: $X_1(\omega) = \sum\limits_{n=-\infty}^{+\infty} x(n)e^{-j\omega n} = \sum\limits_{n=-\infty}^{+\infty} (u(n) - u(n-6))e^{-j\omega n} = \sum\limits_{n=0}^{+\infty} e^{-j\omega n} - \sum\limits_{n=6}^{+\infty} e^{-j\omega n} = \sum\limits_{n=0}^{5} e^{-j\omega n} = \frac{1-e^{-6j\omega}}{1-e^{-j\omega}}$

b) $x_2(n) = 2^n u(-n)$

We have: $X_2(\omega) = \sum_{n=-\infty}^{+\infty} 2^n u(-n) e^{-j\omega n} = \sum_{n=-\infty}^{0} 2^n e^{-j\omega n} = \sum_{k=0}^{+\infty} 2^{-k} e^{j\omega k} = \sum_{k=0}^{+\infty} (2^{-1} e^{j\omega})^k = \frac{1}{1-\frac{1}{2}e^{j\omega}}$

c) $x_3(n) = (\frac{1}{4})^n u(n+4)$

We have: $X_3(\omega) = \sum_{n=-\infty}^{+\infty} (\frac{1}{4})^n u(n+4) e^{-j\omega n} = \sum_{n=-4}^{+\infty} (\frac{1}{4})^n e^{-j\omega n} = (\sum_{m=0}^{+\infty} (\frac{1}{4})^m e^{-j\omega m}) 4^4 e^{4j\omega} = \frac{4^4 e^{4j\omega}}{1-\frac{1}{4}e^{-j\omega}}$

d) $x_4(n) = \begin{cases} 2 - \frac{1}{2}n & |n| \leq 4 \\ 0 & |n| > 4 \end{cases}$

We have: $x_4(n) = \{4, \frac{7}{2}, 3, \frac{5}{2}, 2\uparrow, \frac{3}{2}, 1, \frac{1}{2}, 0\}$

$\Rightarrow X_4(\omega) = 4e^{4j\omega} + \frac{7}{2}e^{3j\omega} + 3e^{2j\omega} + \frac{5}{2}e^{j\omega} + 1 + \frac{3}{2}e^{-j\omega} + e^{-2j\omega} + \frac{1}{2}e^{-3j\omega}$

e) $x_5(n) = |\alpha|^n sin(\omega_0 n) \quad |\alpha| < 1$

We have: $\sum_{n=-\infty}^{+\infty} |x_5(n)| = \sum_{n=-\infty}^{+\infty} |\alpha|^n |sin(\omega_0 n)|$.

Suppose that $\omega_0 = \frac{\pi}{2}$, so that $|sin(\omega_0 n)| = 1$.

$\Rightarrow \sum_{n=-\infty}^{+\infty} |\alpha^n| = \sum_{n=-\infty}^{+\infty} |x_5(n)| \to \infty$

Therefore, the Fourier transform does not exist.

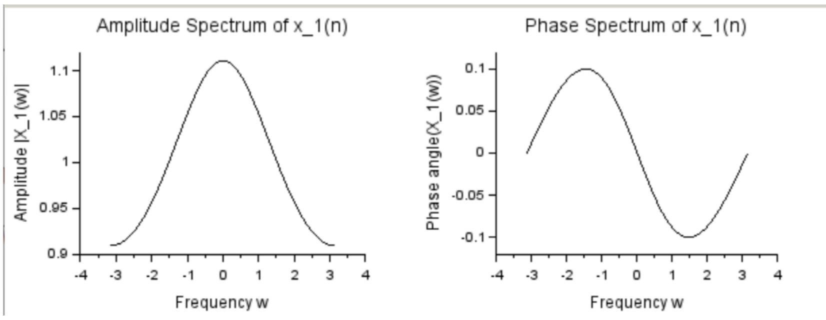
**Exercise 3:** Use Scilab to draw the amplitude spectrum and phase spectrum of the following signals

a) $x_1(n) = 0.1^n u(n)$

We have: $X_1(\omega) = \sum_{n=-\infty}^{+\infty} x(n) e^{-j\omega n} = \frac{1}{1-0.1e^{-j\omega}} = \frac{1-0.1e^{j\omega}}{(1-0.1e^{-j\omega})(1-0.1e^{i\omega})} = \frac{(1-0.1cos\omega)-j(0.1sin\omega)}{1-0.2cos\omega+0.1^2}$

$\iff \begin{cases} X_R(\omega) = \frac{1-0.1cos\omega}{1-0.2cos\omega+0.01} \\ X_I(\omega) = \frac{-0.1sin\omega}{1-0.2cos\omega+0.01} \end{cases} \iff \begin{cases} |X_1(\omega)| = \sqrt{(X_R(\omega))^2 + (X_I(\omega))^2} = \frac{\sqrt{(1-0.1cos\omega)^2+(0.1sin\omega)^2}}{1-0.2cos\omega+0.01} \\ \Theta(\omega) = tan^{-1}(\frac{X_I(\omega)}{X_R(\omega)}) = tan^{-1}(\frac{-0.1sin\omega}{1-0.1cos\omega}) \end{cases}$

Using Scilab to draw the amplitude spectrum and phase spectrum of $x_1(n)$, we have:




b) $x_2(n) = \delta(n) + \delta(n-1) + \delta(n-2) + \delta(n-3)$

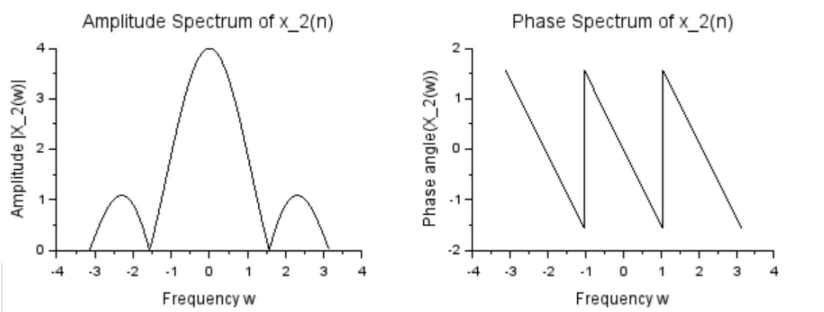
We have: $X_2(\omega) = \sum_{n=-\infty}^{+\infty} x(n) e^{-j\omega n} = 1 + e^{-j\omega} + e^{-2j\omega} + e^{-3j\omega} = 1 + cos\omega - jsin\omega + cos(2\omega) - jsin(2\omega) + cos(3\omega) - jsin(3\omega)$

$\iff X_2(\omega) = (1 + cos\omega + cos(2\omega) + cos(3\omega)) - j(sin\omega + sin(2\omega) + sin(3\omega))$

$\iff \begin{cases} X_R(\omega) = 1 + cos\omega + cos(2\omega) + cos(3\omega) \\ X_I(\omega) = -(sin\omega + sin(2\omega) + sin(3\omega)) \end{cases}$

$\iff \begin{cases} |X_2(\omega)| = \sqrt{(X_R(\omega))^2 + (X_I(\omega))^2} = \sqrt{(1 + cos\omega + cos(2\omega) + cos(3\omega))^2 + (-(sin\omega + sin(2\omega) + sin(3\omega)))^2} \\ \Theta(\omega) = tan^{-1}(\frac{X_I(\omega)}{X_R(\omega)}) = tan^{-1}(\frac{-(sin\omega+sin(2\omega)+sin(3\omega))}{1+cos\omega+cos(2\omega)+cos(3\omega)}) \end{cases}$

Using Scilab to draw the amplitude spectrum and phase spectrum of $x_2(n)$, we have:



**Exercise 4:** Given LTI system by the following input-ouput description equation

$$y(n) + 0.1y(n-1) + 0.2y(n-2) = x(n)$$

Determine the Fourier transform of the impulse response h(n) and then draw the amplitude spectrum and phase spectrum.
We have:

$Y(\omega) + 0.1e^{-j\omega}Y(\omega) + 0.2e^{-2j\omega}Y(\omega) = X(\omega)$

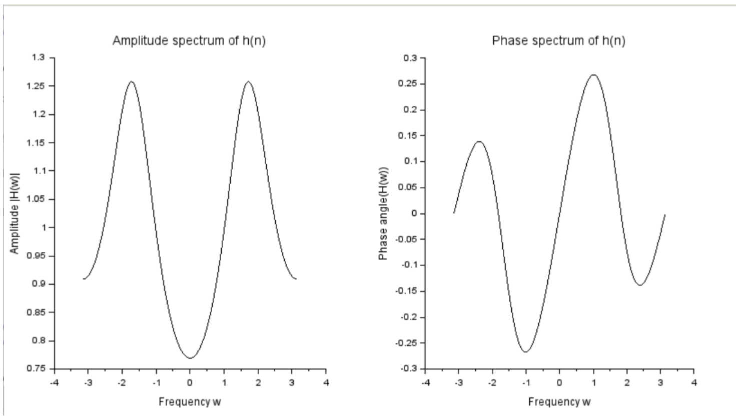
$\iff Y(\omega)(1 + 0.1e^{-j\omega} + 0.2e^{-2j\omega}) = X(\omega)$

$\iff H(\omega) = \frac{Y(\omega)}{X(\omega)} = \frac{1}{1+0.1e^{-j\omega}+0.2e^{-2j\omega}}$

$\iff H(\omega) = \frac{1}{1+0.1(cos\omega-jsin\omega)+0.2[cos(2\omega)-jsin(2\omega)]}$

$\iff \begin{cases} |H(\omega)| = \frac{1}{\sqrt{[1+0.1cos(\omega)+0.2cos(2\omega)]^2+[(-0.1sin(\omega)-0.2.sin(2\omega)]^2}} \\ \Theta(\omega) = -tan^{-1}(\frac{-0.1sin(\omega)-0.2sin(2\omega)}{1+0.1cos(\omega)+0.2cos(2\omega)}) \end{cases}$

Using Scilab to draw the amplitude spectrum and phase spectrum of h(n), we have:




**Exercise 5:** Given LTI system $h(n) = \delta(n) + \delta(n-1)$. Determine output $y(n)$ when input $x(n) = 0.5^n u(n)$ by using Fourier Transform. Then, use SciLab to draw amplitude and phase spectrums.
Using Fourier Transform, we have:

$\begin{cases} H(\omega) = 1 + e^{-j\omega} = \frac{Y(\omega)}{X(\omega)} \\ X(\omega) = \frac{1}{1-0.5e^{-j\omega}} \qquad ROC: e^{j\omega} > 0.5 \end{cases}$

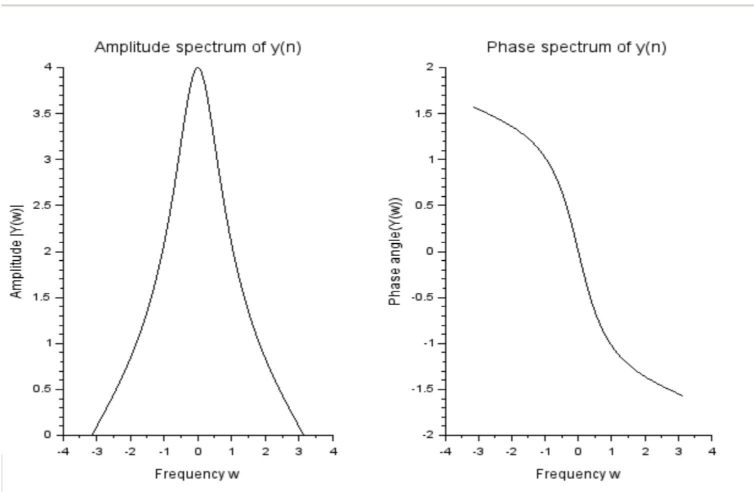
$\iff Y(\omega) = H(\omega)X(\omega) = \frac{1+e^{-j\omega}}{1-0.5e^{-j\omega}} = -2 + \frac{3}{1-0.5e^{-j\omega}}$

$\iff y(n) = -2\delta(n) + 3(0.5)^n u(n) \qquad$ (because $e^{j\omega} > 0.5$)

$Y(\omega) = -2 + \frac{3}{1-0.5e^{-j\omega}} = -2 + \frac{3(1-0.5e^{j\omega})}{(1-0.5e^{-j\omega})(1-0.5e^{j\omega})} = -2 + \frac{3-1.5cos\omega-j(1.5sin\omega)}{1-cos\omega+0.25} = \frac{0.5+0.5cos\omega-j(1.5sin\omega)}{1.25-cos\omega}$

$\iff \begin{cases} |Y(\omega)| = \frac{\sqrt{(0.5+0.5cos\omega)^2+(-1.5sin\omega)^2}}{1.25-cos\omega} \\ \Theta(\omega) = tan^{-1}(\frac{-1.5sin\omega}{0.5+0.5cos\omega}) \end{cases}$

Using Scilab to draw the amplitude spectrum and phase spectrum of $x_2(n)$, we have:




# 5 Discussion

**Accomplishments:**

- Gained a solid understanding of fundamental concepts in Digital Signal Processing such as filtering, spectral analysis and convolution.
- Have chance to work on implementing DSP algorithms in software using Scilab.
- Equipped us with practical skills in processing digital signal.

**Shortcomings**

- The thing we haven't been able to do yet is to use Scilab to perform the Z-transform.
- The absolute accuracy for the exercises has not been ensured yet.

**Future development**

- Applying DSP concepts and techniques to real-world projects or research.
- Further exploring advanced topics in DSP such as adaptive signal processing, filter design, or speech processing.
- Continuing education through advanced courses or specialized training in specific areas of DSP.
- Exploring interdisciplinary applications of DSP in fields such as biomedical engineering, telecommunications, or audio processing.

# 6 References

[1] Scilab Group, INRIA Meta2 Project/ENPC Cergrene,, "Scilab Reference Manual", March 1997.

[2] Scilab Group, INRIA Meta2 Project/ENPC Cergrene, "Signal Processing With Scilab".

[3] John G. Proakis, Dimitris G. Manolakis, Prentice Hall, *"Digital Signal Processing: Principles, Algorithms, and Applications (4th Edition)"*, 2007.