RYERSON UNIVERSITY

Faculty of Engineering, Architecture and Science

Department of Electrical and Computer Engineering

Course Number	ELE632
Course Title	Signals and Systems II
Semester/Year	Winter 2022

Instructor	Dr. Dimitri Androutsos

ASSIGNMENT No.	**1**

Assignment Title	Time-Domain Analysis of Discrete-Time Systems - Part 1

Submission Date	February 5, 2022
Due Date	February 6, 2022

Student Name	Reza Aablue
Student ID	500966944
Signature*	R.A.

**By signing above you attest that you have contributed to this written lab report and confirm that all work you have contributed to this lab report is your own work. Any suspicion of copying or plagiarism in this work <u>will result in an investigation of Academic Misconduct and may result in a "0" on the work,</u> an "F" in the course, or possibly more severe penalties, as well as a Disciplinary Notice on your academic record under the Student Code of Academic Conduct, which can be found online at: www.ryerson.ca/senate/current/pol60.pdf.*

Lab Report #1

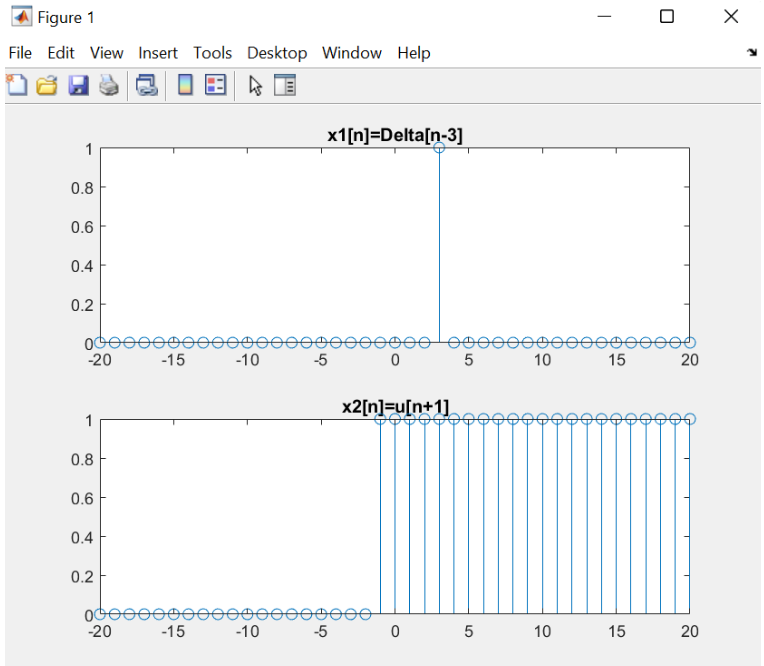
By: Reza Aablue
Date: 05/02/2022

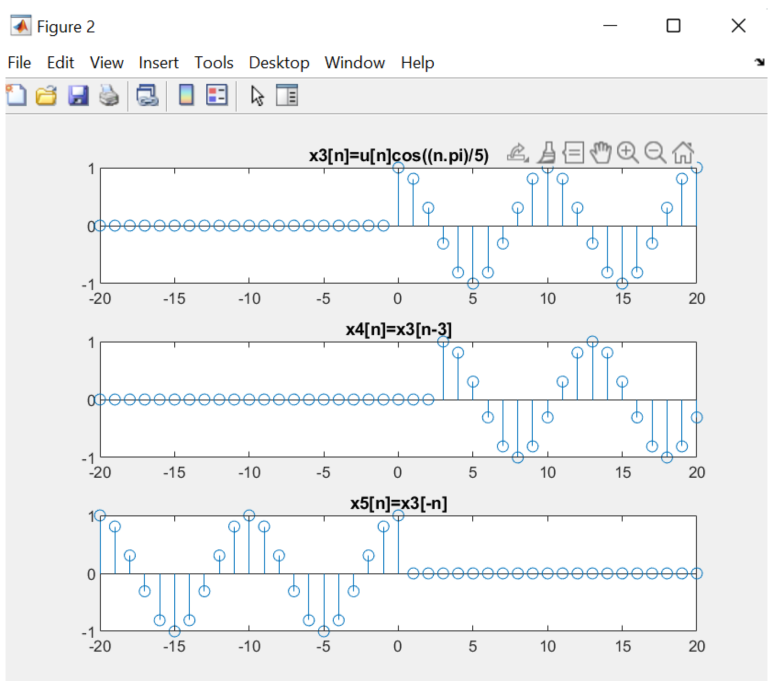
Problem A.1

Lab1ProblemA.m | Lab1ProblemB.m | Lab1ProblemC.m | ProblemD.m

```
% Reza Aablue
% 500966944
% Section 05

%Problem A.1
n = [-20:20]; % Create a discrete range of n values, from -20 up to 20 with increments of 1.

impulseFunction = @(n) (n==0) * 1.0 .* (mod(n,1)==0); % Define impulse function (delta[n]).
x1ofn = impulseFunction (n-3); % x1 [n] = delta [n-3]

unitStepFunction = @(n) (n>=0) * 1.0 .* (mod(n,1)==0); % Define unit step function (u[n]).
x2ofn = unitStepFunction (n+1); % x2 [n] = u [n+1]

xofn = @(n) unitStepFunction(n) .* cos((n.*pi)/5); % Define assignmed x[n] function.
x3ofn = xofn (n); % Labelling of x[n] function.
shift = @(n) xofn (n-3); % Define the fourth function.
x4ofn = shift (n);
reflect = @(n) xofn (-n); % Define fifth function.
x5ofn = reflect (n);

```

```
figure (1); % Contains x1[n] and x2[n] functions.
subplot (2,1,1);
stem(n,x1ofn); % Plot x1[n] function.
title("x1[n]=Delta[n-3]");
subplot (2,1,2);
stem(n,x2ofn); % Plot x1[n] function.
title("x2[n]=u[n+1]");

figure (2); % Contains x3[n], x4[n], and x5[n] functions.
subplot (3,1,1);
stem(n,x3ofn); % Plot x3[n] function.
title("x3[n]=u[n]cos((n.pi)/5)");
subplot (3,1,2);
stem(n,x4ofn); % Plot x4[n] function.
title("x4[n]=x3[n-3]");
subplot (3,1,3);
stem(n,x5ofn); % Plot x5[n] function.
title("x5[n]=x3[-n]");
```

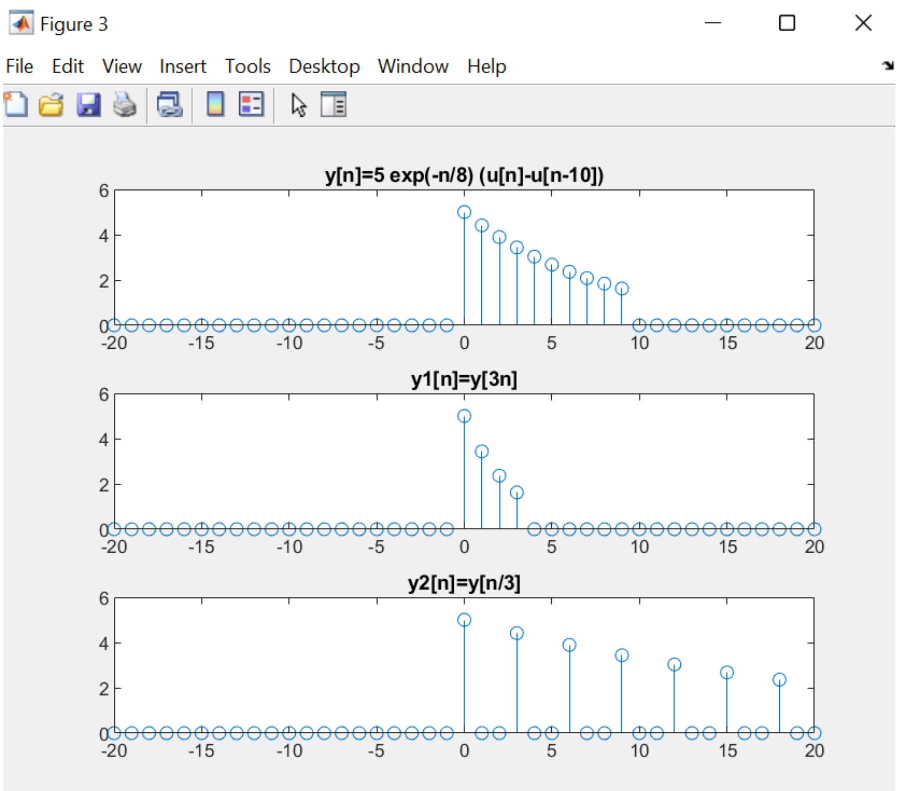




The transformations taking place in $x_1[n]$ and $x_2[n]$ are a time shift of 3 units to the right and a time reversal, respectively.

Problem A.2

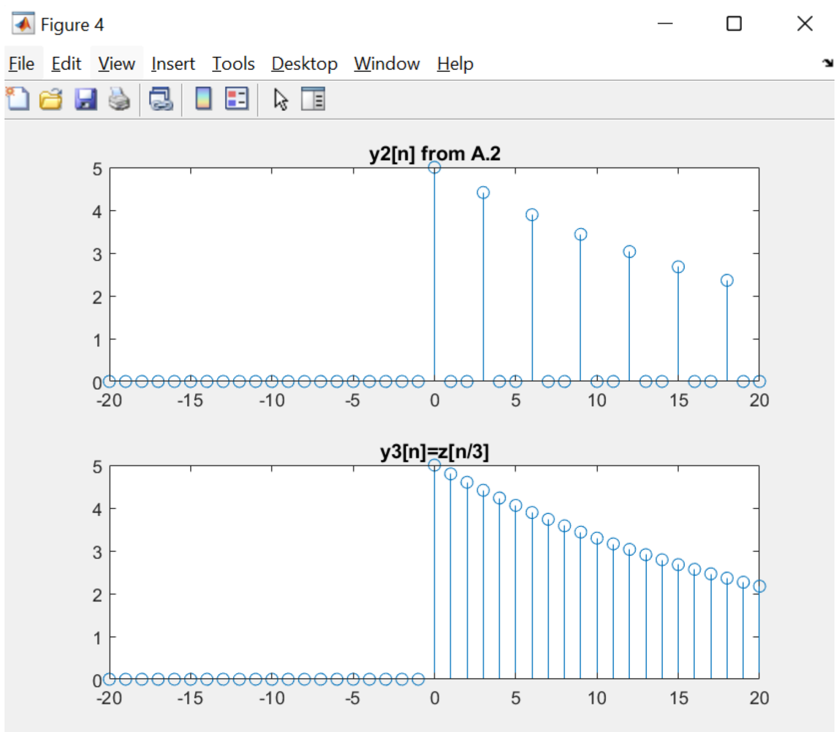
```
% Problem A.2
y = @(n) 5*exp(-n/8).*(unitStepFunction (n) - unitStepFunction (n-10)); % Define y[n] function.
yofn = y(n);

y1 = @(n) y(3.*n);
y1ofn = y1(n);

y2 = @(n) y(n/3);
y2ofn = y2(n);

figure (3); % Contains y[n], y1[n], and y2[n] functions.
subplot (3,1,1);
stem(n,yofn); % Plot y[n] function.
title("y[n]=5 exp(-n/8) (u[n]-u[n-10])");
subplot (3,1,2);
stem(n,y1ofn); % Plot y1[n] function.
title("y1[n]=y[3n]");
subplot (3,1,3);
stem(n,y2ofn); % Plot y2[n] function.
title("y2[n]=y[n/3]");
```



The transformations taking place in $y_1[n]$ and $y_2[n]$ are time scaling by a factor of 3 (downsampling) and time expansion by a factor of ⅓ (upsampling), respectively.

Problem A.3

```
% Problem A.3

% Comparing yofn plot and y3(t) signals.
unitStepFunction2 = @(n) (n>=0) * 1.0;
z = @(n) 5*exp(-n/8).*(unitStepFunction2 (n) - unitStepFunction2 (n-10));
y3 = @(n) z(n/3) .* (mod(n,1)==0);
n=[-20:1:20];

figure(4);
subplot (2,1,1);
stem(n,y2ofn); % Plot y[n] function.
title("y2[n] from A.2");
subplot (2,1,2);
stem(n,y3(n)); % Plot y3[n] function.
title("y3[n]=z[n/3]");
```



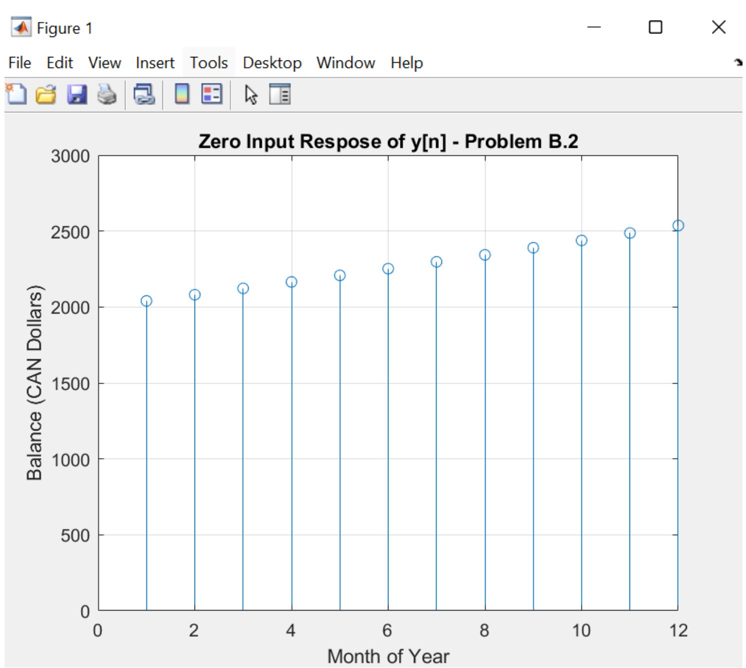
It is observed that $y_3[n]$ has more data points than $y_2[n]$ due to the fact that the transformation was applied to the continuous signal first, which allows the sampling to occur for values that now exist in discrete integer points, which didn't exist before stretching the continuous function.

Problem B.1

```
% Reza Aablue
% 500966944
% Section 05

% Problem B.1

% y[n] = y [n-1] + 0.02 .* y[n-1] + x[n]  (last month's balance + interest
% + deposit)
```

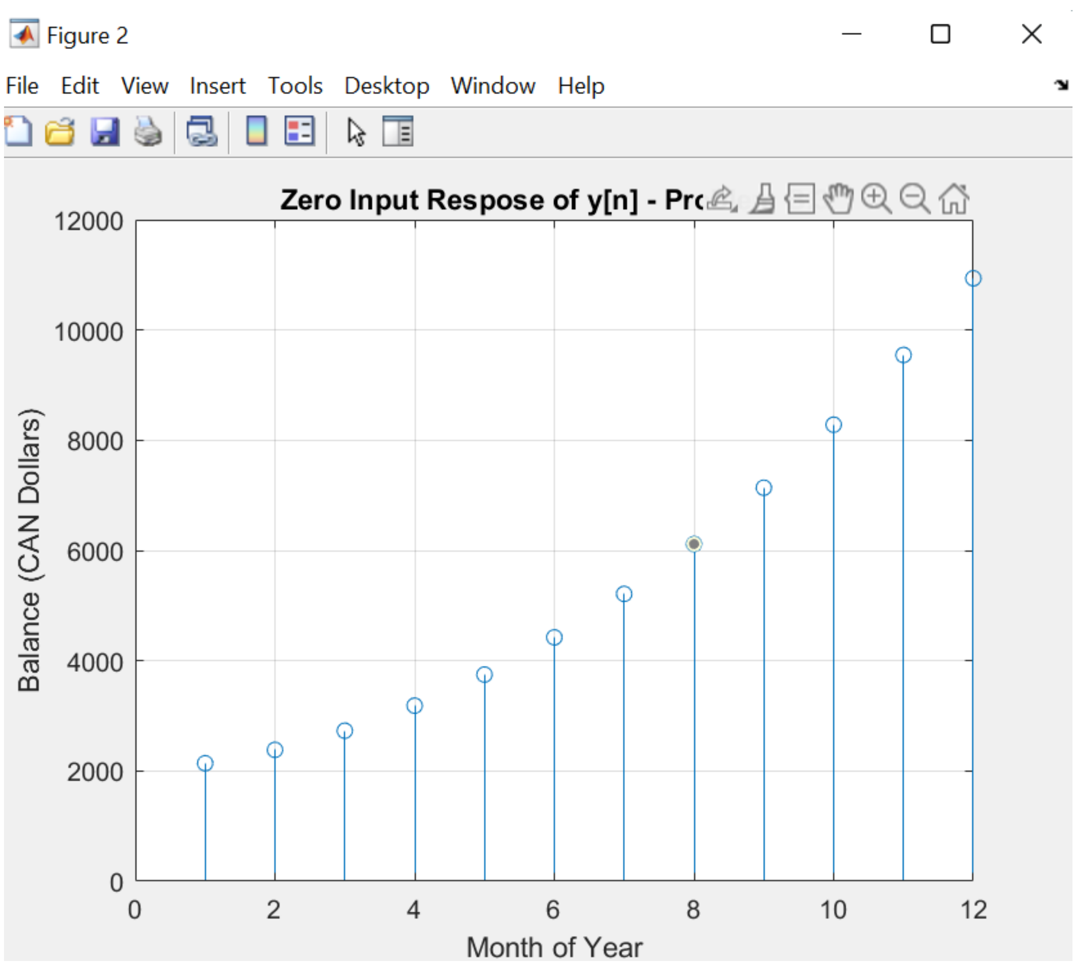
Problem B.2

```
% Problem B.2

y = zeros (1, 12); % Zeros matrix for the balance at each month of the year.
y(1) = 1.02 * 2000; % Balance for January.
% Assuming no deposits in the new year.

for i=2:12
    y(i) = y(i-1) + 0.02 * y(i-1);
end

figure (1);
stem (y); grid;
title ("Zero Input Respose of y[n] - Problem B.2");
xlabel ('Month of Year'); ylabel ("Balance (CAN Dollars)")
```



Problem B.3

```
% Problem B.3
y1 = zeros (1,12);
y1(1) = (1.02 * 2000) + (100 * 1);

for j=2:12
    y1(j) = y1(j-1) + (0.02 * y1(j-1)) + (100 * j);
end

figure (2);
stem (y1); grid;
title ("Zero Input Respose of y[n] - Problem B.3");
xlabel ('Month of Year'); ylabel ("Balance (CAN Dollars)")
```

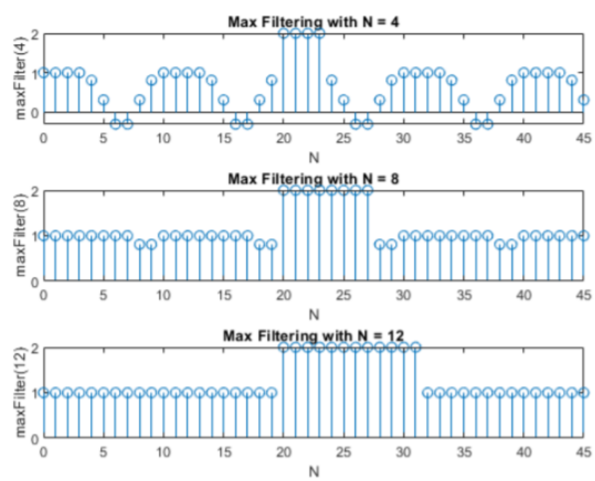


Problem C.1

```
% Problem C.1

function output = Lab1ProblemC (N) % Create max. filter function.
n=[0:45]; % Arbitrary values of n.

impulseFunction = @(n) (n==0) .* 1.0 .* (mod(n,1)==0);

x = @(n) (cos(n.*pi/5) + impulseFunction(n-20) - impulseFunction(n-35)) .* 1.0 .* (mod(n,1)==0);
xofn = x(n); % Create an array of output for the generic function.

f = [zeros(1,(N-1)) xofn]; % Pad input vector with N-1 zeros.
output = [];

for i=1:(length(f)-(N-1))
    temporaryVariable = f(1:1, i:i+(N-1));
    maximumOutput = max (temporaryVariable);
    output = [output maximumOutput];
end
```

Problem C.2

```
% Problem C.2
clear;
n=[0:45];

figure (1);

subplot(3,1,1);
stem(n,Lab1ProblemC(4));
title("Max Filtering with N=4");
xlabel("N")
ylabel("maxFilter(4)");

subplot(3,1,2);
stem(n,Lab1ProblemC(8));
title("Max Filtering with N=8");
xlabel("N");
ylabel("maxFilter(8)");

subplot(3,1,3);
stem(n,Lab1ProblemC(12));
title("Max Filtering with N=12");
xlabel("N");
ylabel("maxFilter(12)");
```



Problem C.3

As the “N” values approach infinity, the signal appears to look like a unit step function multiplied by the maximum value of the signal ($u[n] \;*\; max\ of\ x[n]$).

Problem D.1

```
% Reza Aablue
% 500966944
% Section 05

% Problem D.1

function [power, energy] = ProblemD (x,N)

n = length (x);
a=(2*N)+1;
power = (1/a) .* sum(abs(x.^2));
energy = sum(abs(x.^2));
disp("Power  = "); disp(power);
disp("Energy  = "); disp(energy);
end
```

Problem D.2

This part was completed by using the following command: ProblemD ([-9 -6 -3 0 3 6 9], 3).

```
>> ProblemD ([-9 -6 -3 0 3 6 9],3)
Power  =
    36

Energy  =
   252
```