**Name: Lê Văn Sỹ**

**ID: 20119156**

## Section:

Laboratory Exercise 1

**DISCRETE-TIME SIGNALS: TIME-DOMAIN REPRESENTATION**

## GENERATION OF SEQUENCES

**Project 1.1 Unit sample and unit step sequences**

A copy of Program P1\_1 is given below.

% Program P1\_1

% Generation of a Unit Sample Sequence

clf;

% Generate a vector from -10 to 20

n = -10:20;

% Generate the unit sample sequence

u = [zeros(1,10) 1 zeros(1,20)];

% Plot the unit sample sequence

stem(n,u);

xlabel('Time index n');ylabel('Amplitude');

title('Unit Sample Sequence');

axis([-10 20 0 1.2]);

## Answers:

**Q1.1** The unit sample sequence u[n] generated by running Program P1\_1 is shown below:



**Q1.2** The purpose of clf command is – clear the current figure

The purpose of axis command is – control axis scaling and appearance

The purpose of title command is – add a title to a graph or an axis and specify text properties

The purpose of xlabel command is – add a label to the x-axis and specify text properties

The purpose of ylabel command is – add a label to the y-axis and specify the text properties

**Q1.3** The modified Program P1\_1 to generate a delayed unit sample sequence ud[n] with a delay of 11 samples is given below along with the sequence generated by running this program.

% Program P1\_1

% Generation of a Unit Sample Sequence

clf;

% Generate a vector from -10 to 20

n = -10:19;

m = 0:11;

% Generate the unit sample sequence

ud = [zeros(1,10) zeros(1,11) 1 zeros(1,20-11-1)];

% Plot the unit sample sequence

stem(n,ud);

xlabel('Time index n');ylabel('Amplitude');

title('Unit Sample Sequence');

axis([-13 20 0 1.2]);



**Q1.4** The modified Program P1\_1 to generate a unit step sequence s[n] is given below along with the sequence generated by running this program.

% Program Q1\_4

% Generation of a Unit Step Sequence

clf;

% Generate a vector from -10 to 20

n = -10:20;

% Generate the unit step sequence

s = [zeros(1,10) ones(1,21)];

% Plot the unit step sequence

stem(n,s);

xlabel('Time index n');ylabel('Amplitude');

title('Unit Step Sequence');

axis([-10 20 0 1.2]);



**Q1.5** The modified Program P1\_1 to generate a unit step sequence sd[n] with an advance of 7 samples is given below along with the sequence generated by running this program.

% Program Q1\_5

% Generation of an ADVANCED Unit Step Sequence

clf;

% Generate a vector from -10 to 20

n = -10:20;

% Generate the ADVANCED unit step sequence

sd = [zeros(1,3) ones(1,28)];

% Plot the ADVANCED unit step sequence

stem(n,sd);

xlabel('Time index n');ylabel('Amplitude');

title('ADVANCED Unit Step Sequence');

axis([-10 20 0 1.2]);



## Project 1.2 Exponential signals

A copy of Programs P1\_2 and P1\_3 are given below.

% Program P1\_2

% Generation of a complex exponential sequence

clf;

c = -(1/12)+(pi/6)\*i;

K = 2;

n = 0:40;

x = K\*exp(c\*n);

subplot(2,1,1);

stem(n,real(x));

xlabel('Time index n');ylabel('Amplitude');

title('Real part');

subplot(2,1,2);

stem(n,imag(x));

xlabel('Time index n');ylabel('Amplitude');

title('Imaginary part');

% Program P1\_3

% Generation of a real exponential sequence

clf;

n = 0:35; a = 1.2; K = 0.2;

x = K\*a.^n;

stem(n,x);

xlabel('Time index n');ylabel('Amplitude');

## Answers:

**Q1.6** The complex-valued exponential sequence generated by running Program P1\_2 is shown below:



**Q1.7** The parameter controlling the rate of growth or decay of this sequence is – the real part of c.

The parameter controlling the amplitude of this sequence is - K

**Q1.8** The result of changing the parameter c to (1/12)+(pi/6)\*i is – since exp(-1/12) is less than one and exp(1/12) is greater than one, this change means that the envelope of the signal will grow with n instead of decay with n.

**Q1.9** The purpose of the operator real is – to extract the real part of a Matlab vector.

The purpose of the operator imag is – to extract the imaginary part of a Matlab vector.

**Q1.10** The purpose of the command subplot is – to plot more than one graph in the same Matlab figure.

**Q1.11** The real-valued exponential sequence generated by running Program P1\_3 is shown below:



**Q1.12** The parameter controlling the rate of growth or decay of this sequence is - a

The parameter controlling the amplitude of this sequence is - K

**Q1.13** The difference between the arithmetic operators ^ and .^ is – “^” raises a square matrix to a power using matrix multiplication. “.^” raises the elements of a matrix or vector to a power; this is a “pointwise” operation.

**Q1.14** The sequence generated by running Program P1\_3 with the parameter a changed to 0.9 and the parameter K changed to 20 is shown below:



**Q1.15** The length of this sequence is - 36

It is controlled by the following MATLAB command line: n = 0:35;

It can be changed to generate sequences with different lengths as follows (give an example command line and the corresponding length): n = 0:99; makes the length 100.

**Q1.16** The energies of the real-valued exponential sequences x[n]generated in Q1.11 and Q1.14 and computed using the command sum are - 4.5673e+004 and 2.1042e+003.

## Project 1.3 Sinusoidal sequences

A copy of Program P1\_4 is given below.

% Program P1\_4

% Generation of a sinusoidal sequence

n = 0:40;

f = 0.1;

phase = 0;

A = 1.5;

arg = 2\*pi\*f\*n - phase;

x = A\*cos(arg);

clf; % Clear old graph

stem(n,x); % Plot the generated sequence

axis([0 40 -2 2]);

grid;

title('Sinusoidal Sequence');

xlabel('Time index n');

ylabel('Amplitude');

axis;

## Answers:

**Q1.17** The sinusoidal sequence generated by running Program P1\_4 is displayed below.



**Q1.18** The frequency of this sequence is - f = 0.1 cycles/sample.

It is controlled by the following MATLAB command line: f = 0.1;

A sequence with new frequency 0.2 can be generated by the following command line:

# f = 0.2;

The parameter controlling the phase of this sequence is - phase The parameter controlling the amplitude of this sequence is - A The period of this sequence is - 2/ = 1/f = 10

**Q1.19** The length of this sequence is - 41

It is controlled by the following MATLAB command line: n = 0:40;

A sequence with new length 81\_ can be generated by the following command line:

# n = 0:80;

**Q1.20** The average power of the generated sinusoidal sequence is –

# sum(x(1:10).\*x(1:10))/10 = 1.1250

**Q1.21** The purpose of axis command is – to set the range of the x-axis to [0,40] and the range of the y-axis to [-2,2].

The purpose of grid command is – to turn on the drawing of grid lines on the graph.

**Q1.22** The modified Program P1\_4 to generate a sinusoidal sequence of frequency 0.9 is given below along with the sequence generated by running it.

% Program Q1\_22A

% Generation of a sinusoidal sequence

n = 0:40;

f = 0.9;

phase = 0;

A = 1.5;

arg = 2\*pi\*f\*n - phase;

x = A\*cos(arg);

clf; % Clear old graph

stem(n,x); % Plot the generated sequence

axis([0 40 -2 2]);

grid;

title('Sinusoidal Sequence');

xlabel('Time index n');

ylabel('Amplitude');

axis;



A comparison of this new sequence with the one generated in Question Q1.17 shows - the two graphs are identical. This is because, in the modified program, we have  = 0.9\*2. This generates the same graph as a cosine with angular frequency  - 2 =

# 0.1\*2. Because cosine is an even function, this is the same as a cosine with angular frequency +0.1\*2, which was the value used in P1\_4.m in Q1.17.

In terms of Hertzian frequency, we have for P1\_4.m in Q1.17 that f = 0.1 Hz/sample. For the modified program in Q1.22, we have f = 0.9 Hz/sample, which generates the same graph as f = 0.9 – 1 = 0.1. Again because cosine is even, this makes a graph that is identical to the one we got in Q1.17 with f = +0.1 Hz/sample.

A sinusoidal sequence of frequency 1.1 generated by modifying Program P1\_4 is shown below.



A comparison of this new sequence with the one generated in Question Q1.17 shows - the graph here is again identical to the one in Q1.17. This is because a cosine of frequency f = 1.1 Hz/sample is identical to one with frequency f = 1.1 – 1 = 0.1 Hz/sample, which was the frequency used in Q1.17.

**Q1.23** The sinusoidal sequence of length 50, frequency 0.08, amplitude 2.5, and phase shift of 90 degrees generated by modifying Program P1\_4 is displayed below.

# **NOTE:** for this program, it is necessary to convert the phase of 90 deg to radians and account for the minus sign that appears in the statement “arg = 2\*pi\*f\*n - phase;” as opposed to the plus sign shown in eq. (1.12) of the lab manual. The correct statement to generate the phase is “phase = -90\*pi/180;”. It is also necessary to modify the axis command to account for the new length and amplitude of the signal. The correct axis statement is “axis([0 50 -3 3]);”.



The period of this sequence is - 2/ = 1/f = 1/0.08 = 1/(8/100) = 100/8 = 25/2. Therefore, the fundamental period is 25 and the graph has the “appearance” of going through 2 cycles of a cosine waveform during each period.

**Q1.24** By replacing the stem command in Program P1\_4 with the plot command, the plot obtained is as shown below:



The difference between the new plot and the one generated in Question Q1.17 is – instead of drawing stems from the x-axis to the points on the curve, the “plot” command connects the points with straight line segments, which approximates the graph of a continuous-time cosine signal.

**Q1.25** By replacing the stem command in Program P1\_4 with the stairs command the plot obtained is as shown below:



The difference between the new plot and those generated in Questions Q1.17 and Q1.24 is – the “stairs” command produces a stairstep plot as opposed to the stem graph that was generated in Q1.17 and the straight-line interpolation plot that was generated in Q1.24.

## Project 1.4 Random signals Answers:

**Q1.26** The MATLAB program to generate and display a random signal of length 100 with elements

uniformly distributed in the interval [–2, 2] is given below along with the plot of the random sequence generated by running the program:

% Program Q1\_26

n = 0:99;

A = 2;

%rand('state',sum(100\*clock)); % Obsolete syntax to "seed" the generator

rng('shuffle'); % new syntax to seed generator

x = 2\*A\*(rand(1,length(n))-0.5);

clf; % Clear old graph

stem(n,x); % Plot the generated sequence

axis([0 length(n) -round(2\*(A+0.5))/2 round(2\*(A+0.5))/2]);

grid;

title('uniform Random Sequence');

xlabel('Time index n');

ylabel('Amplitude');

axis;

****

**Q1.27** The MATLAB program to generate and display a Gaussian random signal of length 75 with elements normally distributed with zero mean and a variance of 3 is given below along with the plot of the random sequence generated by running the program:

% Program Q1\_27

% Generation of a Gaussian random sequence

% NOTE: if X is a random variable with zero mean and

% unity variance, then (aX + b) is a random variable

% with mean b and variance a^2. This follows from

% basic probability theory. n = 0:74;

xmean = 0; % mean of x

xstd = sqrt(3); % standard deviation of x rng(VshuffleV); % new syntax to seed generator

% generate the sequence

x = xstd\*randn(1,length(n)) + xmean;

% setup the graph and plot

clf; % Clear old graph

stem(n,x); % Plot the generated sequence xmax = max(abs(x));

Ylim = round(2\*(xmax+0.5))/2; axis([0 length(n) -Ylim Ylim]); grid;

title(VGaussian Random SequenceV); xlabel(VTime index nV); ylabel(VAmplitudeV);

axis;



**Q1.28** The MATLAB program to generate and display five sample sequences of a random sinusoidal signal of length 31

# {X[n]} = {A·cos(on + )}

where the amplitude A and the phase  are statistically independent random variables with uniform probability distribution in the range 0  A  4 for the amplitude and in the range 0     for the phase is given below. Also shown are five sample sequences generated by running this program five different times.

function Q1\_28

clc;

close all;

clear;

n=0:30;

f=0.5;

for i=1:5

A=4\*rand;

Phi=2\*pi\*rand;

x=A\*cos(2\*pi\*f\*n+Phi);

subplot(5,1,i);

axis([0 30 min(x)-0.2 max(x)+0.2]);

stem(n,x);

end;

end



## SIMPLE OPERATIONS ON SEQUENCES

**Project 1.5 Signal Smoothing**

A copy of Program P1\_5 is given below.

% Program P1\_5

% Signal Smoothing by Averaging clf;

R = 51;

d = 0.8\*(rand(R,1) - 0.5); % Generate random noise m = 0:R-1;

s = 2\*m.\*(0.9.^m); % Generate uncorrupted signal x = s + dV; % Generate noise corrupted signal subplot(2,1,1);

plot(m,dV,Vr-V,m,s,Vg--V,m,x,Vb-.V); xlabel(VTime index nV);ylabel(VAmplitudeV); legend(Vd[n] V,Vs[n] V,Vx[n] V);

x1 = [0 0 x];x2 = [0 x 0];x3 = [x 0 0];

y = (x1 + x2 + x3)/3; subplot(2,1,2); plot(m,y(2:R+1),Vr-V,m,s,Vg--V);

legend( Vy[n] V,Vs[n] V);

xlabel(VTime index nV);ylabel(VAmplitudeV);

## Answers:

**Q1.29** The signals generated by running Program P1\_5 are displayed below:



**Q1.30** The uncorrupted signal s[n]is - the product of a linear growth with a slowly decaying real exponential.

The additive noise d[n]is – a random sequence uniformly distributed between -0.4 and

# +0.4.

**Q1.31** The statement x = s + d CANNOT be used to generate the noise corrupted signal because – d is a column vector, whereas s is a row vector; it is necessary to transpose one of these vectors before adding them.

# **Q1.32** The relations between the signals x1, x2, and x3, and the signal x are – all three signals x1, x2, and x3 are extended versions of x, with one additional sample appended at the left and one additional sample appended to the right. The signal x1 is a delayed version of x, shifted one sample to the right with zero padding on the left. The signal x2 is equal to x, with equal zero padding on both the left and right to account for the extended length. Finally, x3 is a time advanced version of x, shifted one sample to the left with zero padding on the right.

**Q1.33** The purpose of the legend command is – to create a legend for the graphs. In P1\_5, the signals are plotted using different colors and line types; the legend provides information about which color and line type is associated with each signal.

## Project 1.6 Generation of Complex Signals

A copy of Program P1\_6 is given below.

% Program P1\_6

% Generation of amplitude modulated sequence clf;

n = 0:100;

m = 0.4;fH = 0.1; fL = 0.01;

xH = sin(2\*pi\*fH\*n); xL = sin(2\*pi\*fL\*n); y = (1+m\*xL).\*xH; stem(n,y);grid;

xlabel(VTime index nV);ylabel(VAmplitudeV);

## Answers:

**Q1.34** The amplitude modulated signals y[n] generated by running Program P1\_6 for various values of the frequencies of the carrier signal xH[n] and the modulating signal xL[n], and various values of the modulation index m are shown below:

# m=0.4; fH=0.1; fL=0.01:



# m=0.8; fH=0.01; fL=0.005



**Q1.35** The difference between the arithmetic operators \* and .\* is – “\*” multiplies two conformable matrices or vectors using matrix multiplication. “.\*” takes the pointwise products of the elements of two matrices or vectors that have the same dimensions.

A copy of Program P1\_7 is given below.

% Program P1\_7

% Generation of a swept frequency sinusoidal sequence n = 0:100;

a = pi/2/100;

b = 0;

arg = a\*n.\*n + b\*n; x = cos(arg);

clf; stem(n, x);

axis([0,100,-1.5,1.5]);

title(VSwept-Frequency Sinusoidal SignalV); xlabel(VTime index nV); ylabel(VAmplitudeV);

grid; axis;

## Answers:

**Q1.36** The swept-frequency sinusoidal sequence x[n] generated by running Program P1\_7 is displayed below.



**Q1.37** The minimum and maximum frequencies of this signal are - The minimum occurs at n=0, where we have 2an+b = 0 rad/sample = 0 Hz/sample. The maximum occurs at n=100, where we have 2an+b = 200a = 200(0.5)(0.01) =  rad/sample

# = 0.5 Hz/sample.

**Q1.38** The Program 1\_7 modified to generate a swept sinusoidal signal with a minimum frequency of

0.1 and a maximum frequency of 0.3 is given below:

# **Note:** for a minimum frequency of 0.1 Hz/sample = /5 rad/sample at n=0, we must have 2a(0) + b = /5, which implies b=/5. For a maximum frequency of 0.3 Hz/sample = 3/5 rad/sample at n=100, we must have 2a(100) + /5 = 3/5, which implies a=/500.

% Program Q1\_38

% Generation of a swept frequency sinusoidal sequence n = 0:100;

a = pi/500;

b = pi/5;

arg = a\*n.\*n + b\*n; x = cos(arg);

clf; stem(n, x);

axis([0,100,-1.5,1.5]);

title(VSwept-Frequency Sinusoidal SignalV); xlabel(VTime index nV); ylabel(VAmplitudeV);

grid; axis;

## WORKSPACE INFORMATION

**Q1.39** The information displayed in the command window as a result of the who command is – a listing of the names of the variables defined in the current workspace.

**Q1.40** The information displayed in the command window as a result of the whos command is – a long form listing of the variables defined in the current workspace, including the variable names, their dimensions (size), the number of bytes of storage required for each variable, and the datatype of each variable. The total number of bytes of storage for the entire workspace is also displayed.

## OTHER TYPES OF SIGNALS (Optional)

**Project 1.8 Squarewave and Sawtooth Signals Answer:**

**Q1.41** MATLAB programs to generate the square-wave and the sawtooth wave sequences of the type shown in Figures 1.1 and 1.2 are given below along with the sequences generated by running these programs:

% Program Q1\_41A

% Generation of the square wave in Fig. 1.1(a) n = 0:30;

f = 0.1;

phase = 0; duty=60;

A = 2.5;

arg = 2\*pi\*f\*n + phase; x = A\*square(arg,duty);

clf; % Clear old graph

stem(n,x); % Plot the generated sequence axis([0 30 -3 3]);

grid;

title(VSquare Wave Sequence of Fig. 1.1(a)V); xlabel(VTime index nV);

ylabel(VAmplitudeV); axis;

% Program Q1\_41B

% Generation of the square wave in Fig. 1.1(b) n = 0:30;

f = 0.1;

phase = 0; duty=30;

A = 2.5;

arg = 2\*pi\*f\*n + phase; x = A\*square(arg,duty);

clf; % Clear old graph

stem(n,x); % Plot the generated sequence axis([0 30 -3 3]);

grid;

title(VSquare Wave Sequence of Fig. 1.1(b)V); xlabel(VTime index nV);

ylabel(VAmplitudeV); axis;

% Program Q1\_41C

% Generation of the square wave in Fig. 1.2(a) n = 0:50;

f = 0.05;

phase = 0;

peak = 1;

A = 2.0;

arg = 2\*pi\*f\*n + phase;

x = A\*sawtooth(arg,peak);

clf; % Clear old graph

stem(n,x); % Plot the generated sequence axis([0 50 -2 2]);

grid;

title(VSawtooth Wave Sequence of Fig. 1.2(a)V); xlabel(VTime index nV);

ylabel(VAmplitudeV); axis;

% Program Q1\_41D

% Generation of the square wave in Fig. 1.2(b) n = 0:50;

f = 0.05;

phase = 0;

peak = 0.5;

A = 2.0;

arg = 2\*pi\*f\*n + phase;

x = A\*sawtooth(arg,peak);

clf; % Clear old graph

stem(n,x); % Plot the generated sequence axis([0 50 -2 2]);

grid;

title(VSawtooth Wave Sequence of Fig. 1.2(b)V); xlabel(VTime index nV);

ylabel(VAmplitudeV); axis;









**Date: 5 November, 2023 Signature: Sy**