**LAB#01**

**Object:** Introduction To Matlab.

**TASK #01:**

Try all help method to get help of any function.

**RESULTS:**

1. **Help function name:**

help roots

roots Find polynomial roots.

roots(C) computes the roots of the polynomial whose coefficients

are the elements of the vector C.

1. **Lookfor:**

>> lookfor date

calendarDuration - Arrays to represent lengths of time in flexible-length calendar date/time units.

datetime - Arrays to represent dates and times.

candidateset - Constructor for the base CandidateSet class

1. **Help:**

help

pack - Consolidate workspace memory.

javachk - Validate level of Java support.

nargchk - Validate number of input arguments.

narginchk - Validate number of input arguments.

nargoutchk - Validate number of output arguments.

validateattributes - Check validity of array.

validatestring - Check validity of text string.

cholupdate - Rank 1 update to Cholesky factorization.

qrupdate - Rank 1 update to QR factorization.

NaT - Not-a-Datetime.

**TASK #02:**

a) Assign to samples from 0 to 9 of time to the vector t.

b) Assign a vector of samples w ithout assigning it to a variable.

c) Assign 10 samples from 0 to 9 without printing it to screen.

**RESULTS:**

1. Assign to samples from 0 to 9 of time to the vector t.

t= 0:9

t =

0 1 2 3 4 5 6 7 8 9

b) Assign a vector of samples ithout assigning it to a variable.

0:9

ans =

0 1 2 3 4 5 6 7 8 9

c) Assign 10 samples from 0 to 9 without printing it to screen.

t=0:9;

**TASK #03:**

Investigate the difference between multiplication & element wise multiplication of vector/matrices.

**RESULTS:**

>> a=[3,2;2,3]

a =

3 2

2 3

>> b=[1,4;4,1]

b =

1 4

4 1

>> a\*b

ans =

11 14

14 11

>> a.\*b

ans =

3 8

8 3

**TASK #04:**

Generate a complete valued matrix a=ones(1,10)+i\*(1:10) & calculate absolute value .Also calculate real and imaginary values.

**RESULT:**

a=ones(1,10)+i\*(1:10)

a =

Columns 1 through 4

1.0000 + 1.0000i 1.0000 + 2.0000i 1.0000 + 3.0000i 1.0000 + 4.0000i

Columns 5 through 8

1.0000 + 5.0000i 1.0000 + 6.0000i 1.0000 + 7.0000i 1.0000 + 8.0000i

Columns 9 through 10

1.0000 + 9.0000i 1.0000 +10.0000i

>> real(a)

ans =

1 1 1 1 1 1 1 1 1 1

>> imag(a)

ans

1 2 3 4 5 6 7 8 9 10

>> abs(a)

ans =

Columns 1 through 9

1.4142 2.2361 3.1623 4.1231 5.0990 6.0828 7.0711 8.0623 9.0554

Column 10

10.0499

**TASK #05:**

Implement a function [x]=sinewave(t) which produces a sinwave of 1001 samples.

**CODING:**

function [x] = sinewave( t)

f=1

x=sin(2\*pi\*f\*t)

stem(t,x)

end

**RESULT:**

>> sinewave(0:1/1000:1)

**GRAPH:**



**TASK #06:**

Use Matlab help to get familiar with the syntax of FOR, WHILE, If-ELSE statements.

**RESULT:**

**>>Help FOR**

for Repeat statements a specific number of times.

The general form of a for statement is:

for variable = expr, statement, ..., statement END

The columns of the expression are stored one at a time in

the variable and then the following statements, up to the

END, are executed. The expression is often of the form X:Y,

in which case its columns are simply scalars. Some examples

(assume N has already been assigned a value).

for R = 1:N

for C = 1:N

A(R,C) = 1/(R+C-1);

end

end

Step S with increments of -0.1

for S = 1.0: -0.1: 0.0, do\_some\_task(S), end

Set E to the unit N-vectors

for E = eye(N), do\_some\_task(E), end

Long loops are more memory efficient when the colon expression appears

in the for statement since the index vector is never created.

The BREAK statement can be used to terminate the loop prematurely.

See also parfor, if, while, switch, break, continue, end, colon.

Reference page in Help browser

doc for

**>> help WHILE**

while Repeat statements an indefinite number of times.

The general form of a while statement is:

while expression

statements

END

The statements are executed while the real part of the expression

has all non-zero elements. The expression is usually the result of

expr rop expr where rop is ==, <, >, <=, >=, or ~=.

The BREAK statement can be used to terminate the loop prematurely.

For example (assuming A already defined):

E = 0\*A; F = E + eye(size(E)); N = 1;

while norm(E+F-E,1) > 0,

E = E + F;

F = A\*F/N;

N = N + 1;

end

See also for, if, switch, break, continue, end.

Reference page in Help browser

doc while

**>> help else-if**

if Conditionally execute statements.

The general form of the if statement is

if expression

statements

ELSEIF expression

statements

ELSE

statements

END

The statements are executed if the real part of the expression

has all non-zero elements. The ELSE and ELSEIF parts are optional.

Zero or more ELSEIF parts can be used as well as nested if's.

The expression is usually of the form expr rop expr where

rop is ==, <, >, <=, >=, or ~=.

Example

if I == J

A(I,J) = 2;

elseif abs(I-J) == 1

A(I,J) = -1;

else

A(I,J) = 0;

end

See also relop, else, elseif, end, for, while, switch.

Reference page in Help browser

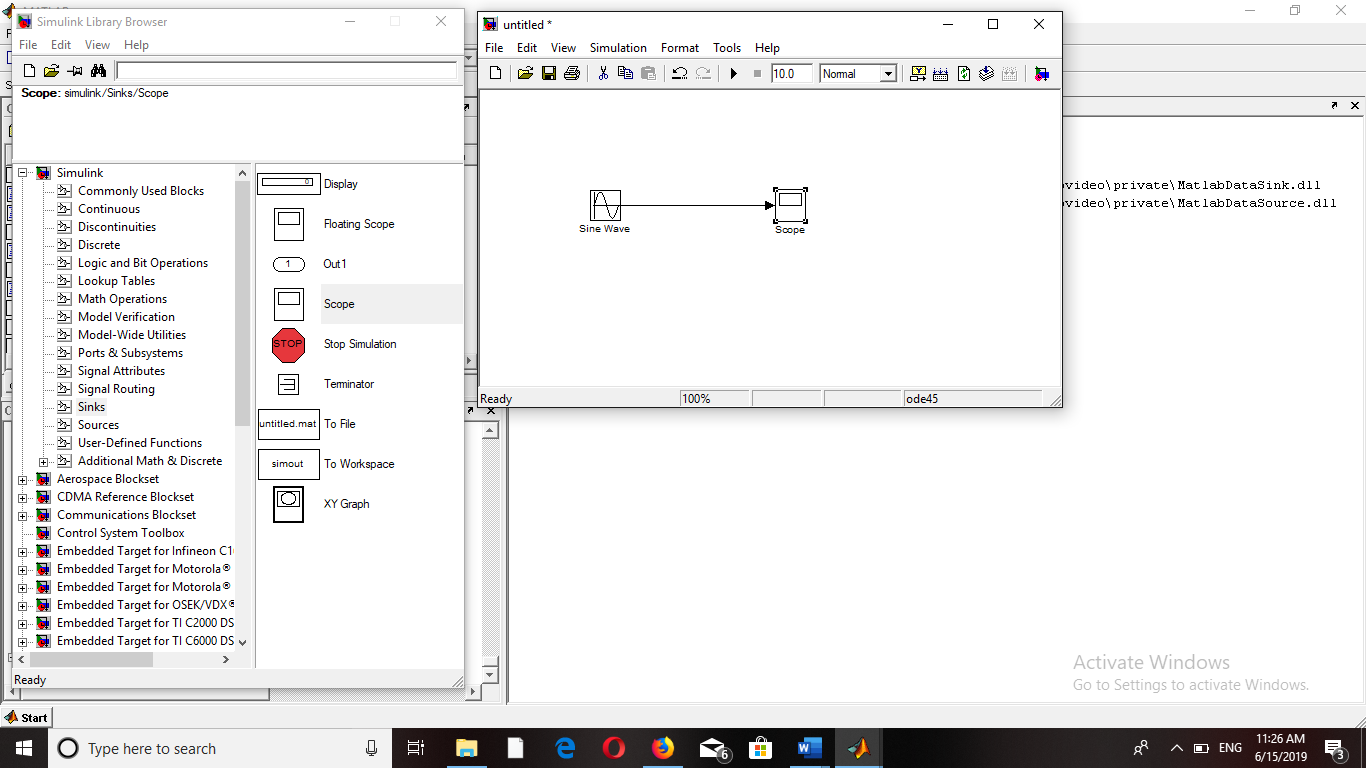
doc if

**TASK #07:**

1. Using simulnk plot a sinewave between t=0S & t=10S.
2. Using simulnk plot a unitstep between t=100S.

**RESULT:**

1. Using simulnk plot a sinewave between t=0S & t=10S.

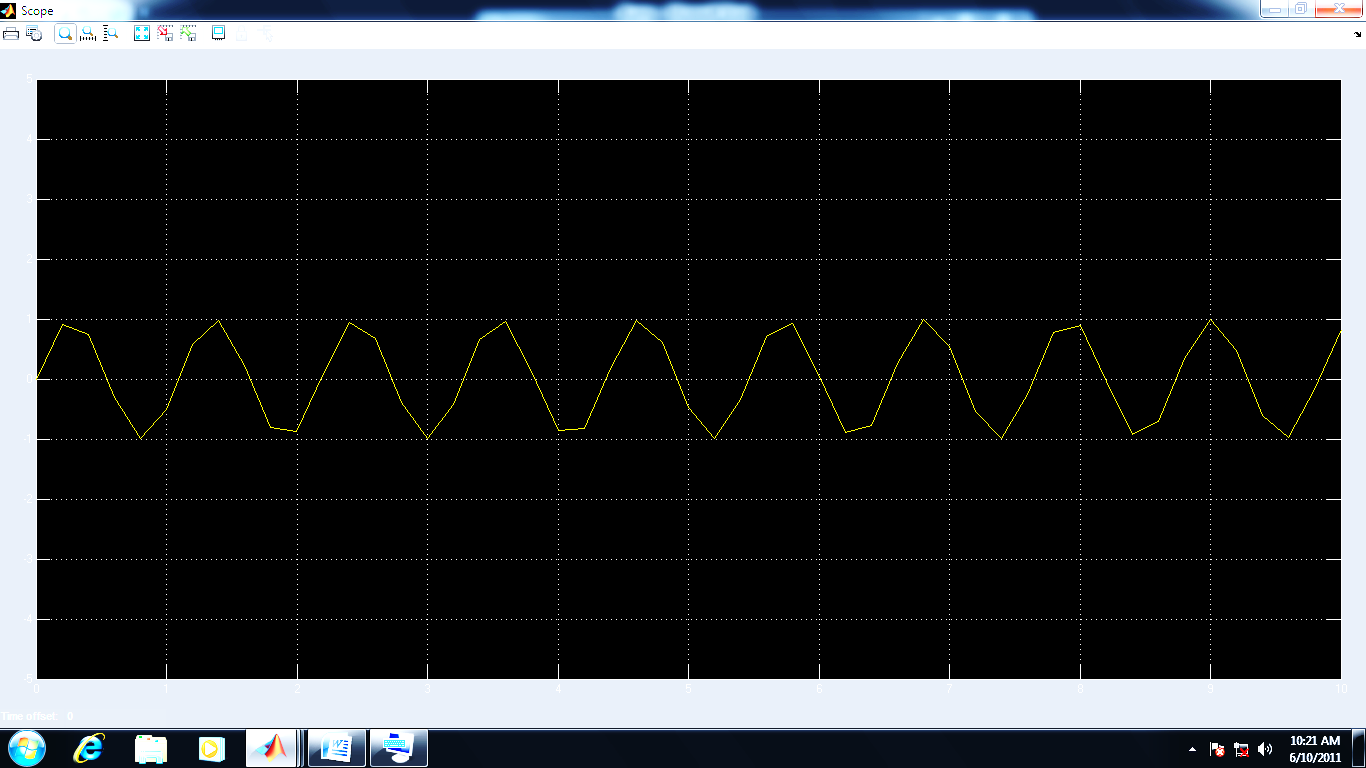
**SIMULINK:**

**FIGURES:**

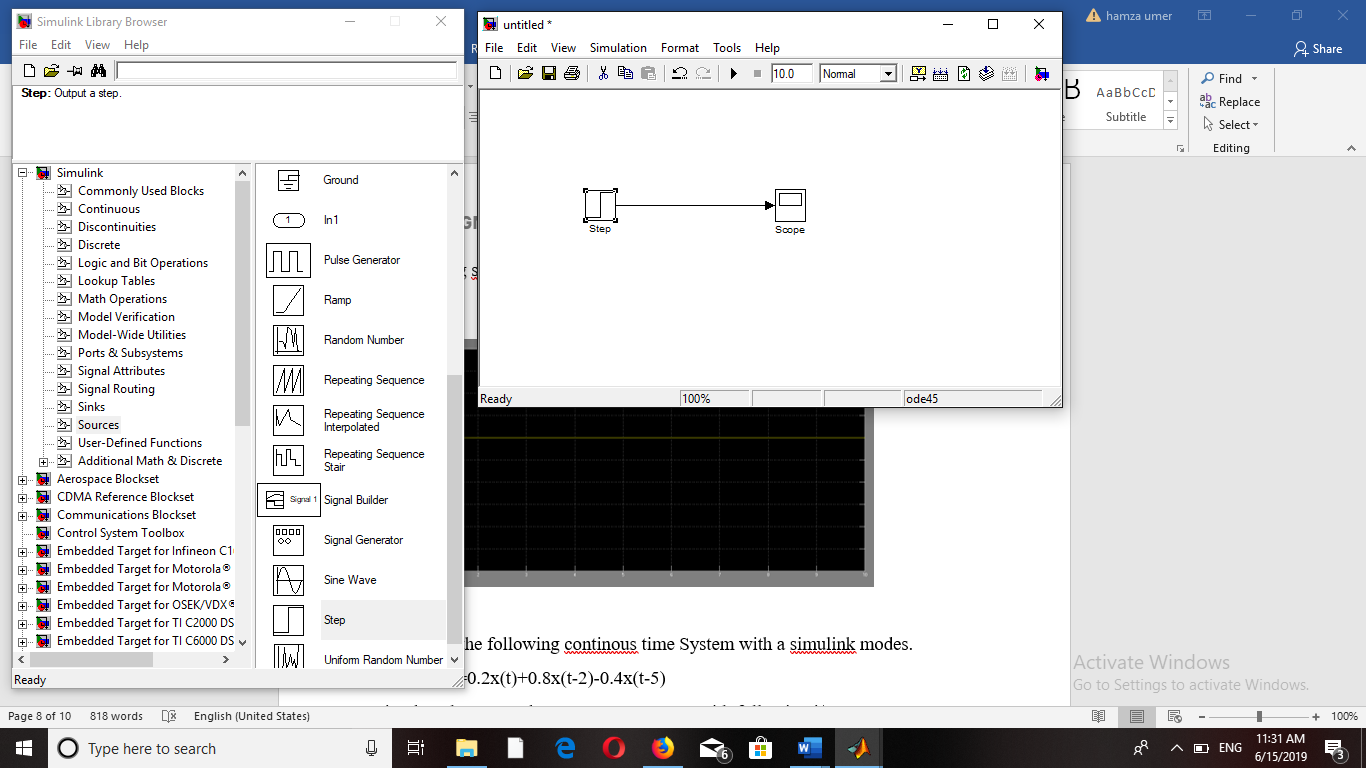
For t=0S:



For t=10S:

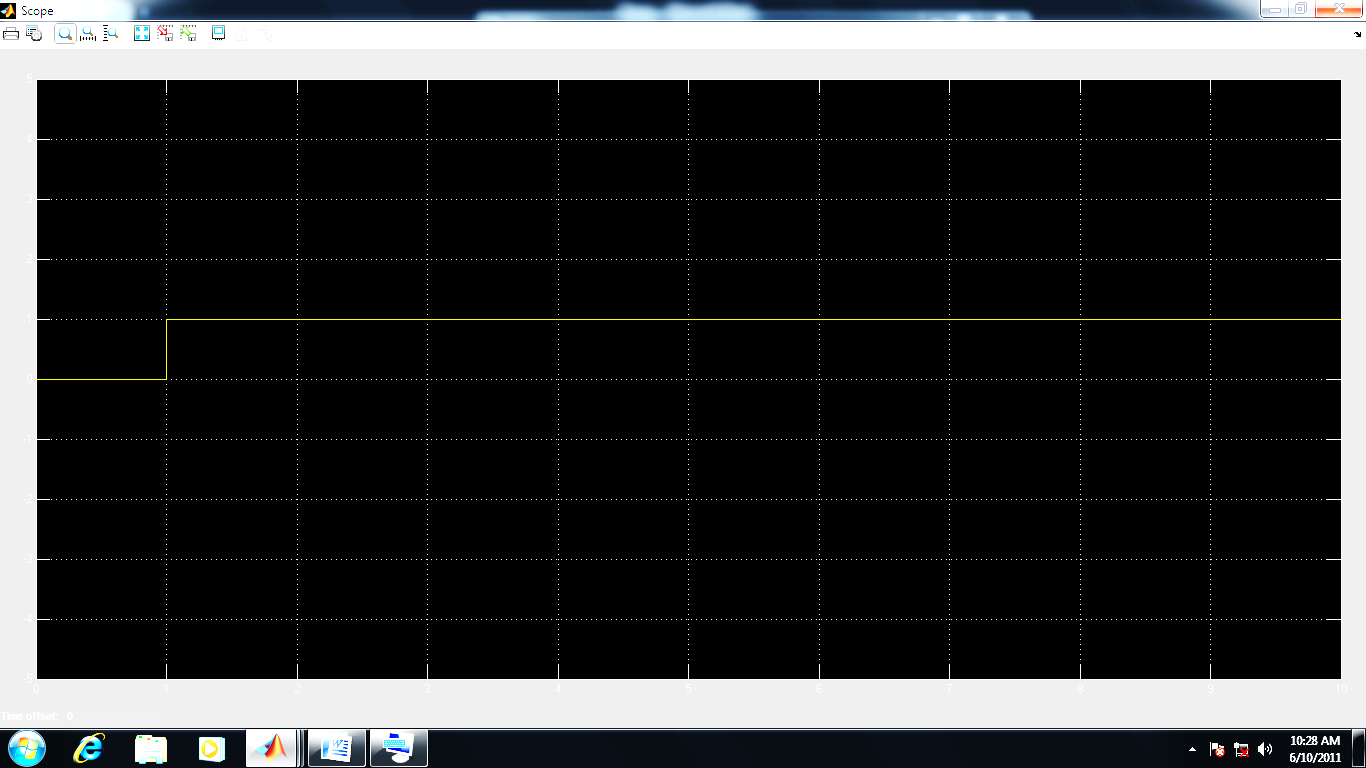


1. Using simulnk plot a unitstep between t=100S.

**SIMULINK:**

**FIGURE:**

For t=100S:



**TASK #08:**

Implement the following continous time System with a simulink modes.

Y(t)=0.2x(t)+0.8x(t-2)-0.4x(t-5)

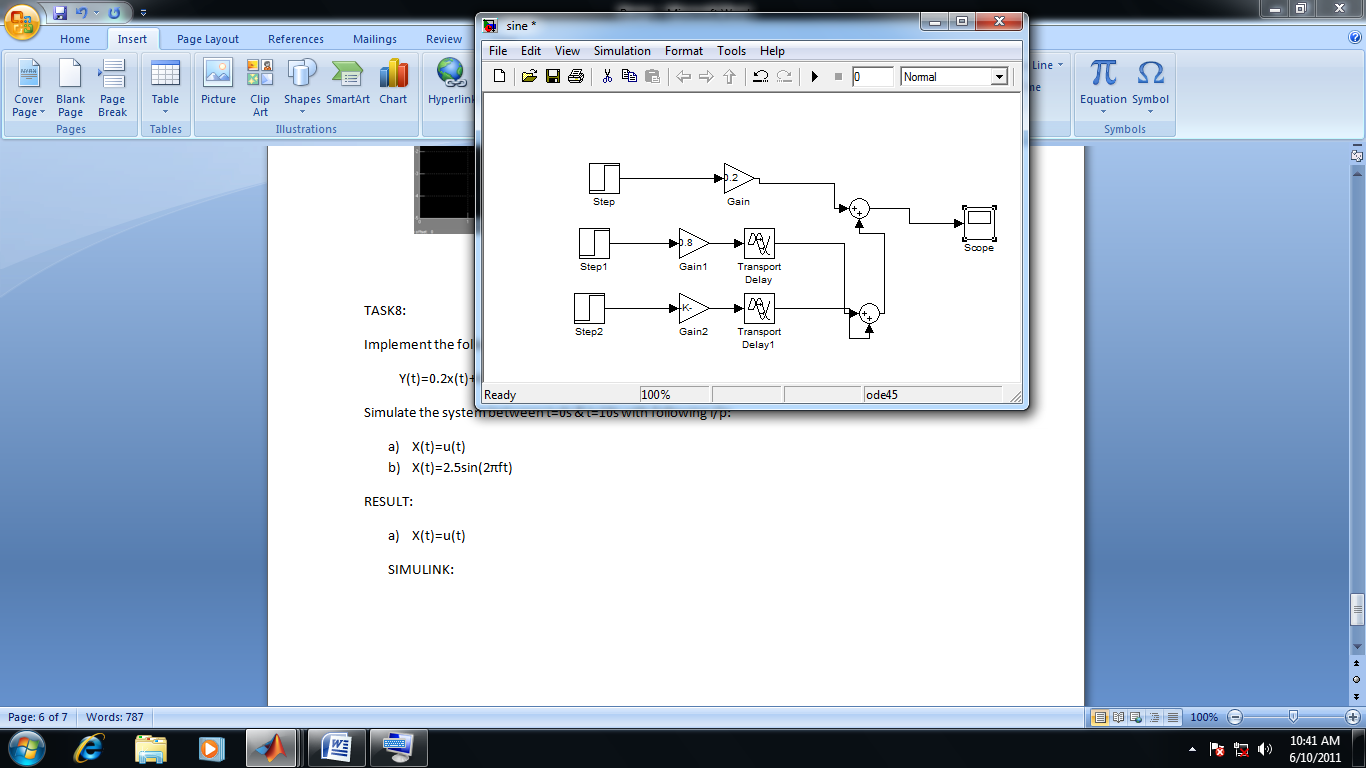
Simulate the system between t=0s & t=10s with following i/p:

1. X(t)=u(t)
2. X(t)=2.5sin(2πft)

**RESULT:**

1. X(t)=u(t)

**SIMULINK:**

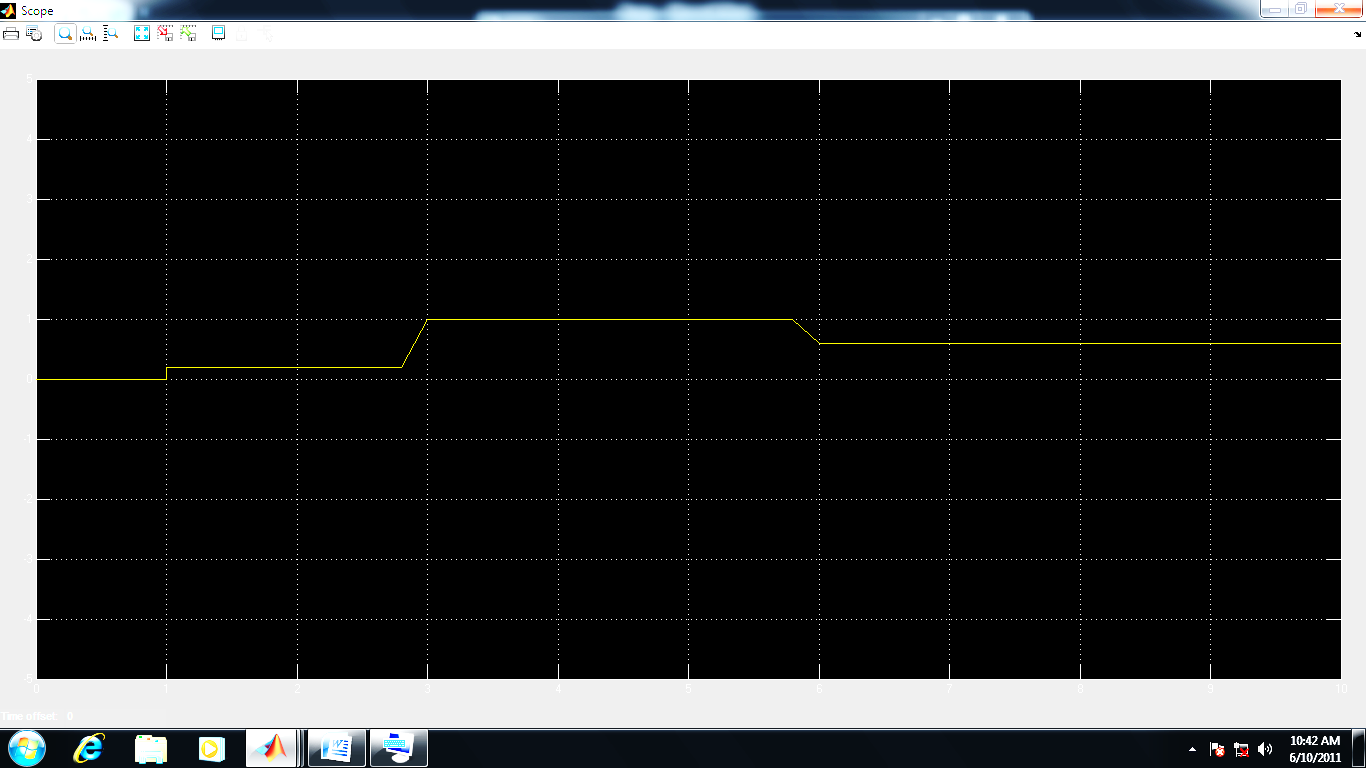


**FIGURE:**

For t=0s:

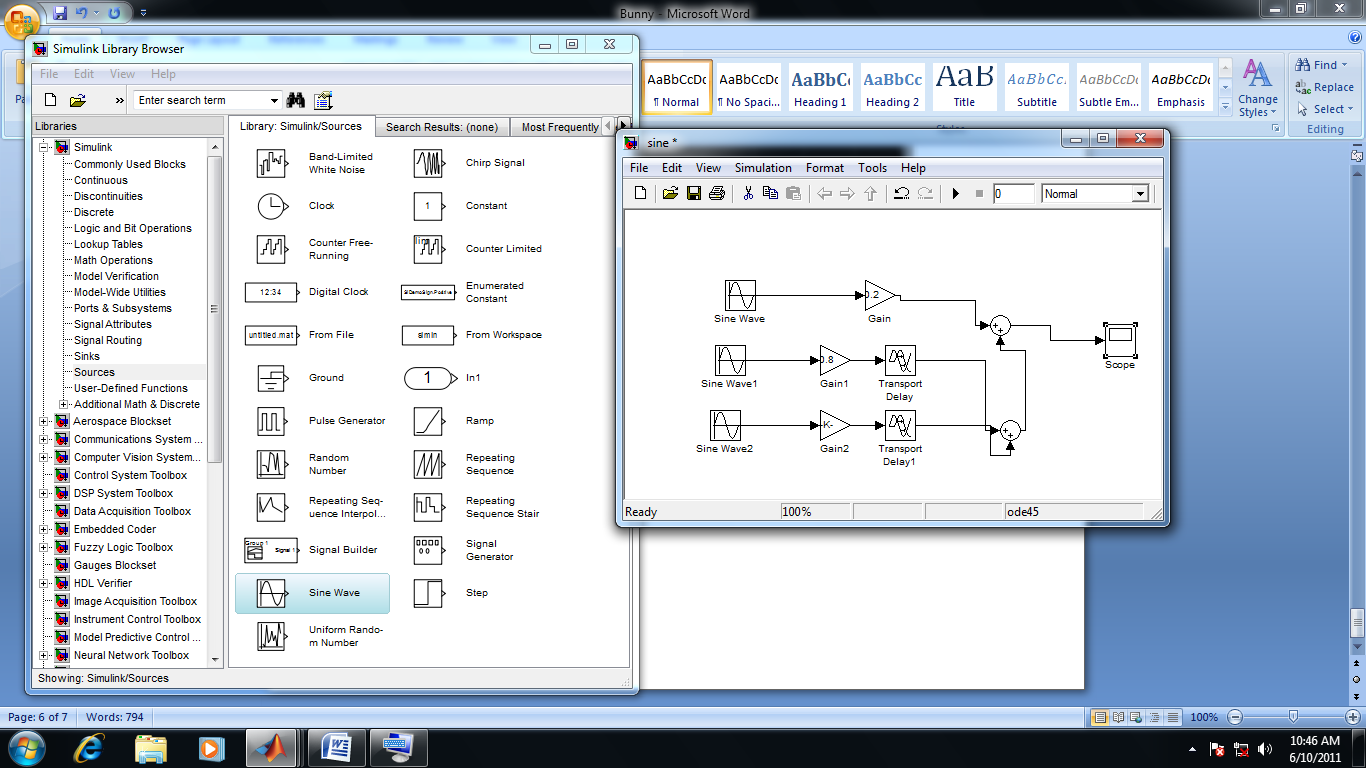


For t=10s:



1. X(t)=2.5sin(2πft)

**SIMULINK:**



For t=0s:



For t=10s:

