A

LABORATORY MANUAL

For

POWER SYSTEMS SIMULATION LABORATORY

Version 2014-2015

For Final Year 2nd Semester EEE Students

DEPARTMENT OF ELECTRICAL & ELECTRONICS ENGINEERING
Sir C.R. Reddy College of Engineering
ELURU - 534 007 (A.P)

CONTENTS

E1 - Introduction to MATLAB and its basic commands

AIM: To learn basic operations and matrix manipulations in MATLAB and write simple scripts for performing given tasks.

1. Introduction to MATLAB:

MATLAB is a widely used numerical computation package. It serves both as a simple calculator and as a sophisticated tool for making long complicated calculations and plot graphs of different functions depending upon requirement. Models of dynamic systems can be built easily using SIMULINK.

Some Benefits of MATLAB are:

- Simple to use
- Fast computations are possible
- Wide working range
- Solution of matrix of any order
- Desired operations are performed in matrices
- Different Programming languages can be used
- Simulation is possible

To start using MATLAB/SIMULINK, open editor to create an m-file or an .mdl Simulink model in Simulink window. Always save using file names without breaks in words.

Some very important functions performed by MATLAB are:

- Matrix computations
- Vector Analysis
- Differential Equations computations
- Integration
- Computer language programming
- Simulation
- 2-D & 3-D Plotting

Further Reading:

1. Modern Power System Analysis, 4th Edition by *Nagrath & Kothari*
2. Introduction to MATLAB by *Rudra Pratap*
3. MATLAB User Manual by *Mathworks*

2. Basic Commands:

Some basic MATLAB commands are given as follows. Type these at the command prompt to verify.

Addition: A+B **Subtraction**: A-B **Multiplication**: A*B
Division: A/B **Power**: A^B **Power of individual element**: A. ^B
Range : A: B **Square-Root**: A=sqrt (B) where A & B are any arbitrary integers

3. Basic Matrix Operations:

This is a demonstration of some aspects of the MATLAB language.
Execute the commands in MATLAB and print out the results.

Creating a Vector:

Let's create a simple vector with 9 elements called a.

```
a = [1 2 3 4 6 4 3 4 5]
a =
1       2       3       4       6       4       3       4       5
```

Now let's add 2 to each element of our vector, a, and store the result in a new vector.Notice how MATLAB requires no special handling of vector or matrix math.

Adding an element to a Vector:

```
b = a + 2
b =
3       4       5       6       8       6       5       6       7
```

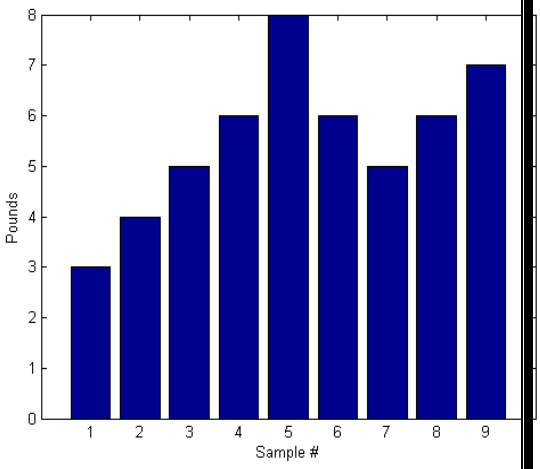
Plots and Graphs:

Creating graphs in MATLAB is as easy as one command. Let's plot the result of our vector addition with grid lines.

```
plot (b)
grid on
```

MATLAB can make other graph types as well, with axis labels.

```
bar(b)
xlabel('Sample #')
ylabel('Pounds')
```



MATLAB can use symbols in plots as well. Here is an example using stars to mark the points. MATLAB offers a variety of other symbols and line types.

```
plot(b,'*')
axis([0 10 0 10])
```

Creating a matrix:

One area in which MATLAB excels is matrix computation. Creating a matrix is as easy as making a vector, using semicolons (;) to separate the rows of a matrix.

```
A = [1 2 0; 2 5 -1; 4 10 -1]
A =
1       2       0
2       5       -1
4       10      -1
```

Adding a new Row:

```
A(4,:)=[7 8 9]
ans=
     1    2    0
     2    5   -1
     4   10   -1
     7    8    9
```

Adding a new Column:

```
A(:,4)=[7 8 9]
ans=
     1    2    0   7
     2    5   -1   8
     4   10    -1   9
```

Transpose:

We can easily find the transpose of the matrix A.

```
A = [1 2 0; 2 5 -1; 4 10 -1]

A'

=
     1    2    4
     2    5   10
     0   -1   -1
```

Matrix Multiplication:

Now let's multiply these two matrices together. Note again that MATLAB doesn't require you to deal with matrices as a collection of numbers. MATLAB knows when you are dealing with matrices and adjusts your calculations accordingly.

```
A = [1 1 1; 2 2 2; 3 3 3]
B = [4 4 4; 5 5 5; 6 6 6]
```

```
C =A*B
C =
     15    15    15
     30    30    30
     45    45    45
```

Instead of doing a matrix multiply, we can multiply the corresponding elements of two matrices or vectors using the'.* 'operator.

```
C =A.*B
C =
      4     4     4
     10    10    10
     18    18    18
```

Inverse:
Let's find the inverse of a matrix
A = [1 2 0; 2 5 -1; 4 10 -1]

```
X=inv(A)
X =
    5     2    -2
   -2    -1     1
    0    -2     1
```

... and then illustrate the fact that a matrix times its inverse is the identity matrix.

```
I = inv (A) * A
I =
    1     0     0
    0     1     0
    0     0     1
```

Commands used:

Result:

****

E2 - MATLAB Program to Simulate Ferranti Effect

AIM: To find Ferranti effect of a 5000 kM transmission line and to plot the locus of voltage for the given problem and verify results in MATLAB.

PROBLEM:

A 3-Phase 50 Hz transmission line is 5000 kM long. The line parameters are R= 0.125Ω/km,X= 0.4 Ω/km and Y= $2.8*10^{-6}$ mho/km. If the line is open circuited with a receiving end voltage of 220KV, find the rms value and phase angle of the following. Use the receiving-end line to neutral voltage as reference.

(a) The incident and reflected voltages to neutral at the receiving-end.

(b) The incident and reflected voltages to neutral at 200 km from the receiving-end.

(c) The resultant voltage at 200 km from the receiving end.

Solve the problem theoretically. Vary the length of the long transmission line in steps of 10 KM from zero (receiving end) to 5000KM (sending end), and plot the sending end voltage phasor using MATLAB.

THEORY:

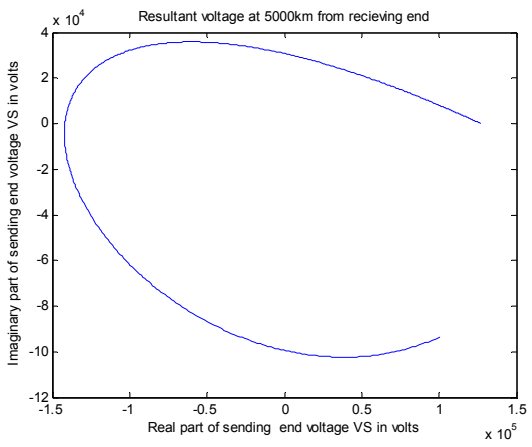
THEORETICAL SOLUTION:

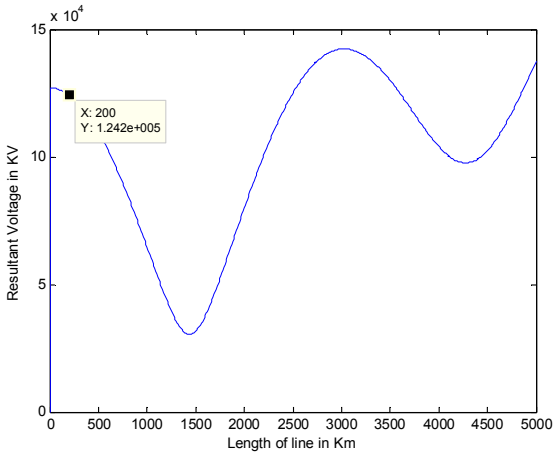
MATLAB PROGRAM:

```
%Program to illustrate Ferranti effect
%it simulates the effect by varying the length of transmission line from
%zero(receiving end) to 5000km in steps of 10km
%and plots the sending end voltage phasor
clc
clear all

VR=220e3/sqrt(3);
alpha=0.163e-3;
beta=1.0683e-3;
L=5000;
k=1;
```

```
for i=0:10:L,
    VS=(VR/2)*exp(alpha*i)*exp(j*beta*i)+(VR/2)*exp(-alpha*i)*exp(-j*beta*i);
    X(k)=real(VS);
    Y(k)=imag(VS);
    k=k+1;
    p(k)=VS;
    q(k)=i;
end

figure(1);
plot(p,q)

figure(2);
plot(X,Y)
```

EXPECTED OUTPUT:





Commands used:

RESULT:

****

E3 - MATLAB Program to Model Transmission Lines

AIM: To find the parameters of the given transmission line using short-line and nominal-pi methods and verify using MATLAB..

PROBLEM:

A 50 Hz transmission line 300km long has a total series impedance of 40+j 125 ohms and a total shunt admittance of 10^{-3} mho. The receiving-end load is 50 MW at 220kV with 0.8 lagging power factor. Find the sending-end voltage, current, power and power factor using (a) short line approximation, and (b) nominal-pi method. Compare the results and comment.

Solve the problem theoretically and verify with MATLAB

THEORY:

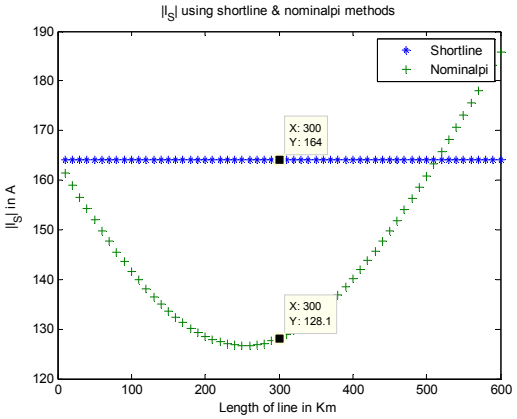
THEORETICAL SOLUTION:

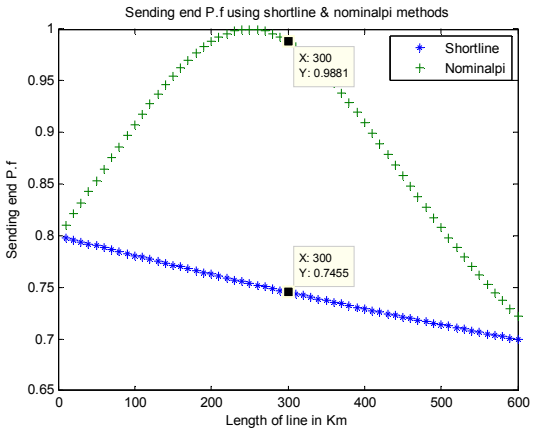
MATLAB PROGRAM:

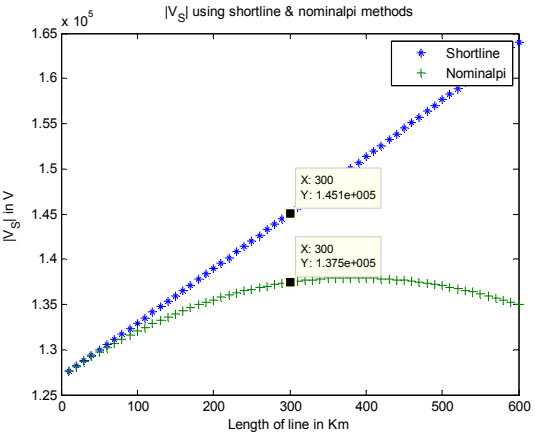
```
clc
clear all
f=50;L=300;z=40+i*125;y=i*1e-3;
PR=50e6/3;VR=220e3/(sqrt(3));Pfload=0.8;IRR=PR/(VR*Pfload);IR=IRR*(0.8-
i*0.6);
z=z/L;y=y/L;k=1;
for i=10:10:600,
   %short line aproximation
   VS_shortline(k)=VR+((z*i*IR));
   IS_shortline(k)=IR;
   spf_shortline(k)=cos(angle(VS_shortline(k))-angle(IS_shortline(k)));

spower_shortline(k)=3*abs(VS_shortline(k))*abs(IS_shortline(k))*spf_shortline(k);
   %nominal pi method
   A=1+(y*i)*(z*i)/2;
   B=z*i;
   C=y*i*(1+(y*i)*(z*i)/4);
   D=A;
   VS_nominalpi(k)=A*VR+B*IR;
   IS_nominalpi(k)=C*VR+D*IR;
   spf_nominalpi(k)=cos(angle(VS_nominalpi(k))-angle(IS_nominalpi(k)));

spower_nominalpi(k)=3*abs(VS_nominalpi(k))*abs(IS_nominalpi(k))*spf_nominalpi(
k);
   point(k)=i;
   k=k+1;
end
```

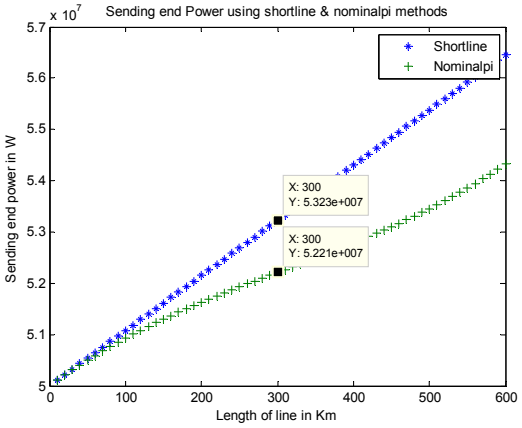
```
%plots of short line in red and nominal pi in red
figure(1);
plot(point,abs(VS_shortline),'r',point,abs(VS_nominalpi),'g')
figure(2);
plot(point,abs(IS_shortline),'r',point,abs(IS_nominalpi),'g')
figure(3);
plot(point,abs(spf_shortline),'r',point,abs(spf_nominalpi),'g')
figure(4);
plot(point,abs(spower_shortline),'r',point,abs(spower_nominalpi),'g')
```

EXPECTED OUTPUT:









Commands used:

RESULT:

* * * * * * * * * * * * * * *

E4 - MATLAB Program to Solve Load Flow Equations By Gauss-Seidel Method

AIM: To find load flow solution of the given power system using Gauss-Seidel method theoretically for one iteration and obtain full solution using MATLAB.

PROBLEM:

For the sample power system shown below, the generators are connected at all the four buses, while loads are at buses 2 and 3. Values of real and reactive powers are listed in the table. All buses other than the slack are PQ type. Assuming a flat voltage start, find the voltages and bus angles at the three buses at the end of first GS iteration.

Input data:

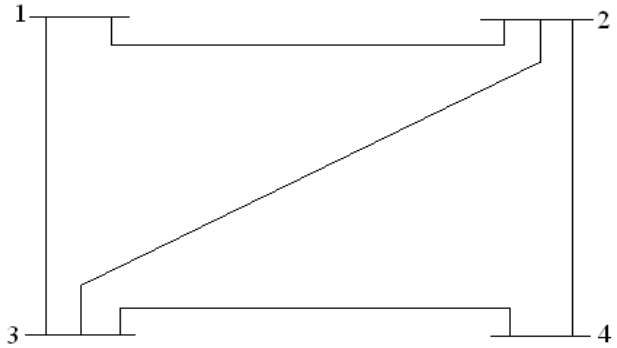


Figure: a simple 4-bus power system

Bus	P_i , pu	Q_i , pu	V_i , pu	Remarks
1	-	-	$1.04\angle 0^0$	Slack bus
2	0.5	-0.2	-	PQ bus
3	-1.0	0.5	-	PQ bus
4	0.3	-0.1	-	PQ bus

For the above system, the bus admittance matrix is

$$Y_{BUS} = \begin{bmatrix} +3-j9 & -2.00+j6 & -1.00+j3 & 0 \\ -2+j6 & +3.666-j11 & -0.666+j2 & -1+j3 \\ -1+j3 & -0.666+j2 & +3.666-j11 & -2+j6 \\ 0 & -1.00+j3 & -2.00+j6 & +3-j9 \end{bmatrix}$$

Write a MATLAB program to solve the load flow equations of the above sample power system by using Gauss-Seidal method.

THEORY:

THEORETICAL SOLUTION:

MATLAB PROGRAM:

```
% Load flow using gauss siedel method
clc
clear
n=4;
V=[1.04 1 1 1];
Y=[3-j*9    -2+j*6      -1+j*3  0
  -2+j*6   3.666-j*11  -0.666+j*2  -1+j*3
  -1+j*3   -0.666+j*2  3.666-j*11  -2+j*6
  0       -1+j*3      -2+j*6      3-j*9];
type=ones(n,1);
typechanged=zeros(n,1);
Qlimitmax=zeros(n,1);
Qlimitmin=zeros(n,1);
Vmagfixed=zeros(n,1);
type(2)=2;
Qlimitmax(2)=1.0;
Qlimitmin(2)=-0.2;
Vmagfixed(2)=1.04;
diff=10;
noofiter=1;
Vprev=V;
while (diff>0.00001 | noofiter==1),
```

```
abs(V);
abs(Vprev);
Vprev=V;
P=[inf 0.5 -1 0.3];
Q=[inf -0.3 0.5 -0.1];
S=[inf 0.5-j*0.2 -1.0+j*0.5 0.3-j*0.1];
for i=2:n,
      if type(i)==2 |typechanged(i)==1,
          if (Q(i)>Qlimitmax(i)| Q(i)<Qlimitmin(i))
                if (Q(i)<Qlimitmin(i))
                   Q(i)=Qlimitmin(i);
                else
                    Q(i)=Qlimitmax(i);
                end
                type(i)=1;
                typechanged(i)=1;
          else
              type(i)=2;
              typechanged(i)=0;
          end
       end
 sumyv=0;
 for k=1:n,
    if(i~=k)
```

```
            sumyv=sumyv+Y(i,k)*V(k);
        end
    end
    V(i)=(1/Y(i,i))*((P(i)-j*Q(i))/conj(V(i))-sumyv)
    if type(i)==2 & typechanged(i)~=1,
        V(i)=PolarTorect(Vmagfixed(i),angle(V(i)*180/pi))
    end
end
diff=max(abs(abs(V(2:n))-abs(Vprev(2:n))));
noofiter=noofiter+1
end
```

EXPECTED OUTPUT:

no of iterations	V1	V2	V3	V4
1	1.04	1.0191 + 0.0464i	1.0280 - 0.0870i	1.0250 - 0.0092i
2	1.04	1.0290 + 0.0269i	1.0352 - 0.0934i	1.0334 - 0.0208i
3	1.04	1.0335 + 0.0223i	1.0401 - 0.0999i	1.0385 - 0.0269i
4	1.04	1.0360 + 0.0193i	1.0427 - 0.1034i	1.0413 - 0.0304i
5	1.04	1.0374 + 0.0176i	1.0442 - 0.1053i	1.0429 - 0.0324i
6	1.04	1.0382 + 0.0167i	1.0450 - 0.1064i	1.0437 - 0.0335i
7	1.04	1.0386 + 0.0161i	1.0455 - 0.1070i	1.0442 - 0.0341i
8	1.04	1.0388 + 0.0158i	1.0457 - 0.1074i	1.0445 - 0.0344i
9	1.04	1.0390 + 0.0157i	1.0459 - 0.1076i	1.0446 - 0.0346i
10	1.04	1.0390 + 0.0156i	1.0460 - 0.1077i	1.0447 - 0.0347i
11	1.04	1.0391 + 0.0155i	1.0460 - 0.1078i	1.0448 - 0.0348i
12	1.04	1.0391 + 0.0155i	1.0460 - 0.1078i	1.0448 - 0.0348i
13	1.04	1.0391 + 0.0155i	1.0460 - 0.1078i	1.0448 - 0.0348i
14	1.04	1.0391 + 0.0155i	1.0461 - 0.1078i	1.0448 - 0.0349i

Commands used:

RESULT:

****

E5 –MATLAB Program to Find Optimum Loading Of Generators Neglecting Transmission Losses

AIM: To find optimum loading of two units for the given load neglecting transmission losses and verify using MATLAB.

PROBLEM:

Incremental fuel costs in rupees per MWh for a plant consisting of two units are:

$\frac{dF_1}{dP_1} = 0.2P_1+40$ and $\frac{dF_2}{dP_2} = 0.25P_2+30$

Assume that both units are operating at all times, and total load varies from 40 MW to 250 MW, and the maximum and minimum loads on each unit are to be 125 MW and 20 MW respectively. How will the load be shared between the two units as the system load varies over the full range? What are the corresponding values of the plant incremental costs?

Solve the problem theoretically and verify using MATLAB.

THEORY:

THEORETICAL SOLUTION:

MATLAB PROGRAM:

```
% the demand is taken as 231.25 MW, n is no of generators, Pd stands for load
%demand; alpha and beta arrays denote alpha beta coefficients for given
%generators
clc
clear all

n=2;Pd=231.25;alpha=[0.20 0.25];beta=[40 30];
% initial guess for lamda
lamda =20;lamdaprev=lamda;
% tolerance is eps and increment in lamda is deltalamda
eps=1;deltalamda=0.25;
% the min. and max. limits of each generating unit are stored in arrays Pgmin and
Pgmax.
Pgmax= [125 125];Pgmin= [20  20];Pg = 100*ones(n,1);
while abs(sum(Pg)-Pd)>eps
for i = 1:n,
   Pg(i)=(lamda-beta(i))/alpha(i);
```

```
    if Pg(i)>Pgmax(i)
        Pg(i)=Pgmax(i);
end
if Pg(i)<Pgmin(i)
    Pg(i)=Pgmin(i);
end
end
if (sum(Pg)-Pd)<0
    lamdaprev=lamda;
    lamda=lamda+deltalamda;
else
    lamdaprev=lamda;
    lamda=lamda-deltalamda;
end
end
disp('The final value of Lamda is')
lamdaprev
disp('The distribution of load shared by two units is')
Pg
```

EXPECTED OUTPUT:

```
The final value of Lamda is
lamdaprev =  61.2500
The distribution of load shared by two units is
Pg =
  106.2500
  125.0000
```

Commands used:

RESULT:

****

E6 –MATLAB Program to Find Optimum Loading Of Generators with Penalty Factors

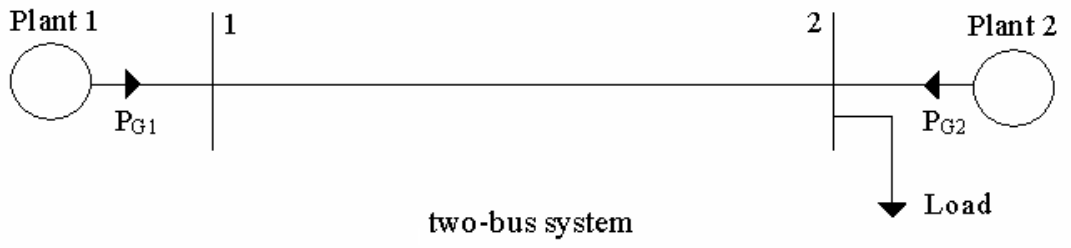
AIM: To find optimum loading of two units for the given load with penalty factors and verify using MATLAB.

PROBLEM:

A two-bus system is shown in figure. If 100 MW is transmitted from plant 1 to the load, a transmission loss of 10 MW is incurred. Find the required generation for each plant and the power received by load when the system λ is Rs 25/MWh. The incremental fuel costs of the two plants are given below:

$$\frac{dC_1}{dP_{G1}} = 0.02\ P_{G1} + 16.0 \text{ Rs/MWh}$$

$$\frac{dC_2}{dP_{G2}} = 0.04\ P_{G2} + 20.0 \text{ Rs/MWh}$$



Solve the problem theoretically. Use the data in the following MATLAB program

THEORY:

THEORETICAL SOLUTION:

MATLAB PROGRAM:

```
% this program finds the optimal loading of generators including penalty factors
% Pd stands for load demand, alpha and beta arrays denote alpha beta coefficients
%for given generators, and n is the no of generators
clc
clear
n=2;Pd=237.04;
alpha=[0.020 0.04];
beta=[16 20];
% initial guess for lamda is 20;tolerance is eps and increment in lamda is deltalamda
```

```
lamda = 20; lamdaprev = lamda ; eps = 1; deltalamda = 0.25;
% the min. and max. limits of each generating unit are stored in arrays Pgmin and
Pgmax
Pgmax=[200 200];Pgmin=[0 0];
B = [0.0010   0
       0    0];
noofiter=0;PL=0;Pg = zeros(n,1);
while abs(sum(Pg)-Pd-PL)>eps
   for i=1:n,
      sigma=B(i,:)*Pg-B(i,i)*Pg(i);
      Pg(i)=(1-beta(i)/(lamda-(2*sigma)))/(alpha(i)/lamda+2*B(i,i));
      %PL=Pg'*B*Pg;
   if Pg(i)>Pgmax (i)
     Pg(i)=Pgmax (i);
   end
   if Pg(i)<Pgmin(i)
     Pg(i)=Pgmin(i);
   end
   end

   PL = Pg'*B*Pg;
   if (sum(Pg)-Pd-PL)<0
      lamdaprev=lamda;
      lamda=lamda+deltalamda;
   else
      lamdaprev=lamda;
      lamda=lamda-deltalamda;
   end
      noofiter=noofiter + 1;
      Pg;
end
```

```
disp ('The no of iterations required are')
noofiter
disp ('The final value of lamda is')
lamdaprev
disp ('The optimal loading of generators including penalty factors is')
Pg
disp('The losses are')
PL
```

EXPECTED OUTPUT:

```
The no of iterations required are
noofiter =   21
The final value of lamda is
lamdaprev =    25
The optimal loading of generators including penalty factors is
Pg =
  128.5714
  125.0000
The losses are
PL =
   16.5306
```

Commands used:

RESULT:

****

E7 - MATLAB Program to Solve Swing Equation using Point-by-Point Method

Aim: To solve the swing equation of the given problem by using point-by-point method and write a MATLAB program to verify the result.

PROBLEM:

A 20 MVA, 50Hz generator delivers 18MW over a double circuit line to an infinite bus. The generator has KE of 2.52MJ/MVA at rated speed. The generator transient reactance is Xd=0.35p.u. Each transmission circuit has R=0 and a reactance of 0.2pu on 20 MVA Base. |E|=1.1 p.u and infinite bus voltage V=1.0. A three phase short circuit occurs at the midpoint of one of the transmission lines. Plot swing curves with fault cleared by simultaneous opening of breakers at both ends of the line at 6.25 cycles after the occurrence of fault. Also plot the swing curve over the period of 0.5 s if the fault sustained. **Solve the swing equation by point-by-point method theoretically and verify using MATLAB Program. Comment on system stability.**

THEORY:

THEORETICAL SOLUTION:

MATLAB PROGRAM:

Program 1: *Save this part in another m-file with name swing.m*

```
%Defining the function swing
function[time ang]=swing(tc)
k=0;v=1;E=1.1;pm=0.9;T=0.5;delT=0.05;ddelta=0;time(1)=0;ang(1)=21.64;xdf=1
.25;xaf=0.55;t=0;
delta=21.64*pi/180;i=2;
m=2.52/(180*50);
while t<T
      if t<tc
         x=xdf;
      else x=xaf;
      end
   pmax=(E*v)/x;
```

```
    pa=pm-pmax*sin(delta);
    ddelta=ddelta+(delT^2*(pa/m));
    delta=(delta*180/pi+ddelta)*(pi/180);
    deltadeg=delta*180/pi;
    t=t+delT;
    time(i)=t;
    ang(i)=deltadeg;
    i=i+1;
end
end
```

Program 2: *Main program that is dependent on swing.m*

```
%solution of Swing equation by point-by-point method
clc
clear all
close all
for i=1:2
    tc=input('enter the value of clearing time:\n');
    [time,ang]=swing(tc)
    t(:,1)=time;
    a(:,i)=ang;
end
plot(t,a(:,1),'*-',t,a(:,2),'d-')
axis([0 0.5 0 inf])
t,a
```

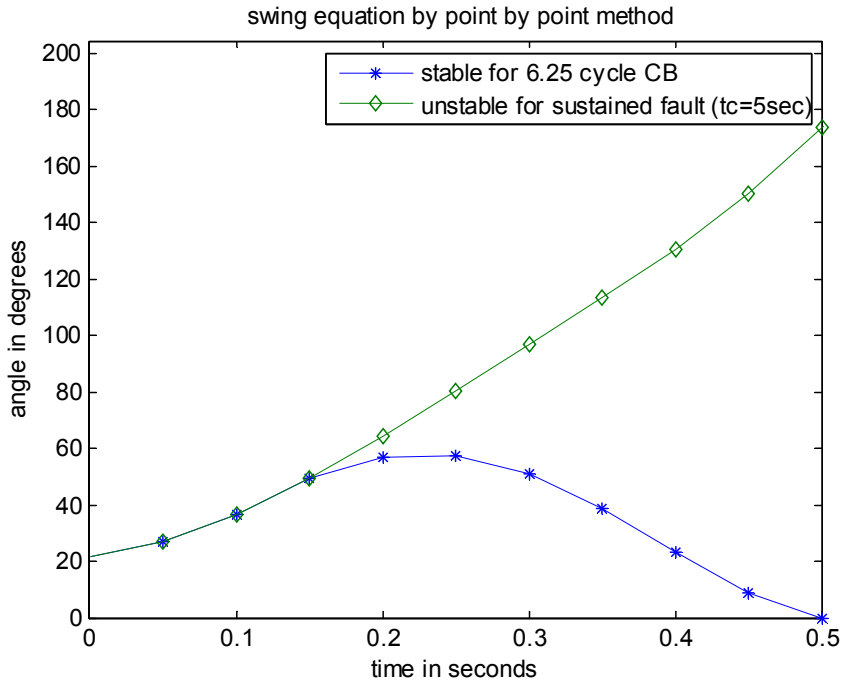
Inputs to main program:

Enter the value of clearing time as

0.25 sec, and

5 sec

EXPECTED OUTPUT:



Commands used:

RESULT:

****

E8-Simulink Model of Single Area Load frequency Control without and with PI Controller

AIM: To find dynamic response of the given single area load frequency control problem theoretically and to plot and verify the results in SIMULINK.

PROBLEM:

The parameters for load frequency control of a single area are:

Speed governor gain	Kg=10
Time constant of speed governor	Tg=0.4
Speed regulation of speed governor	R=3
Gain of turbine	Kt=0.1
Time constant of turbine	Tt=0.5
Gain of power system	Kp=100
Time constant of power system	Tp=20
Changes in the load	ΔPD=0.01 pu

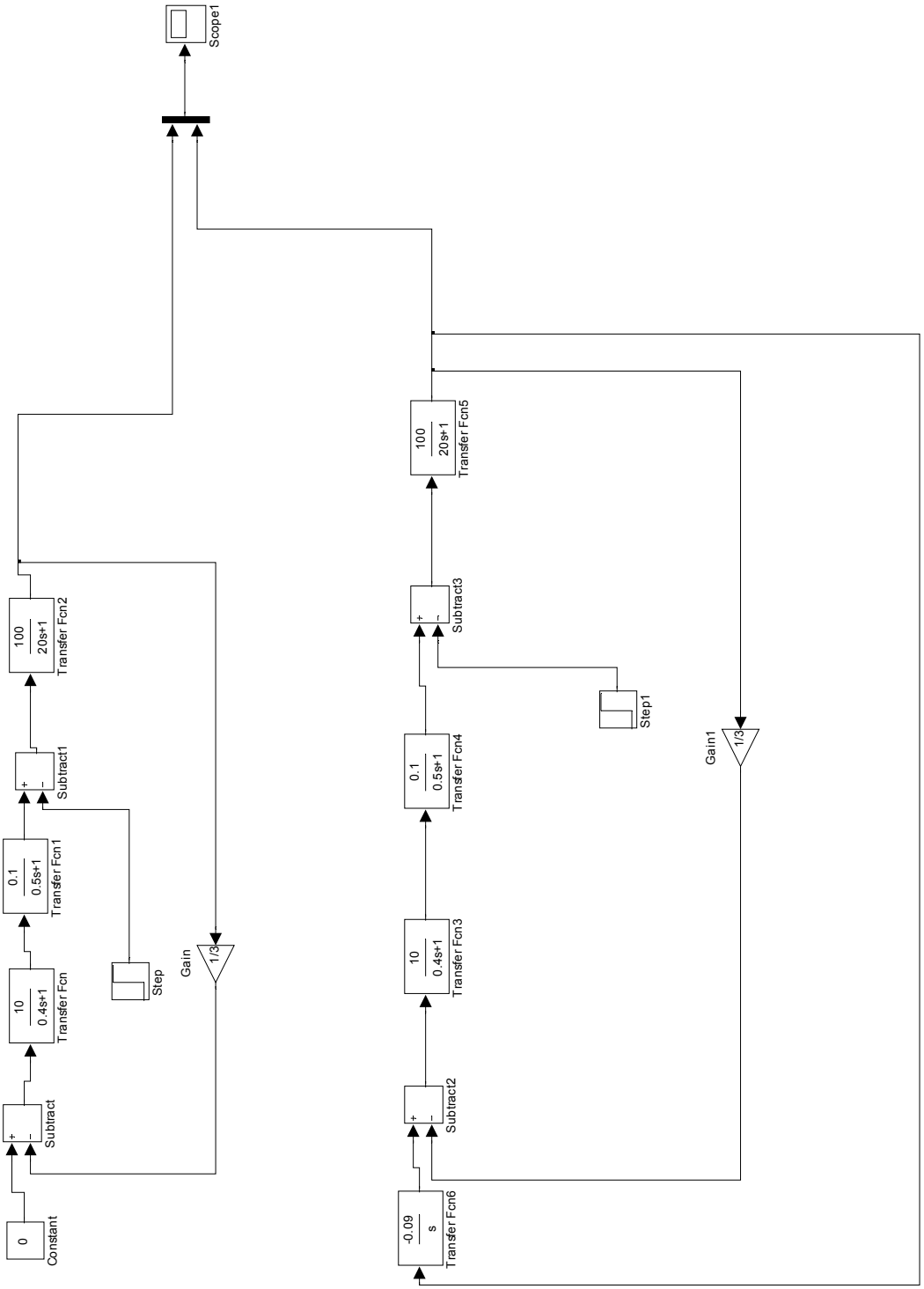
An integral controller with gain Ki=0.09 is now used to reduce steady state error. What is the dynamic response of the system with and without the controller?
Obtain the dynamic response of the system with and without the PI controller by developing a SIMULINK model and verify the responses

THEORY:

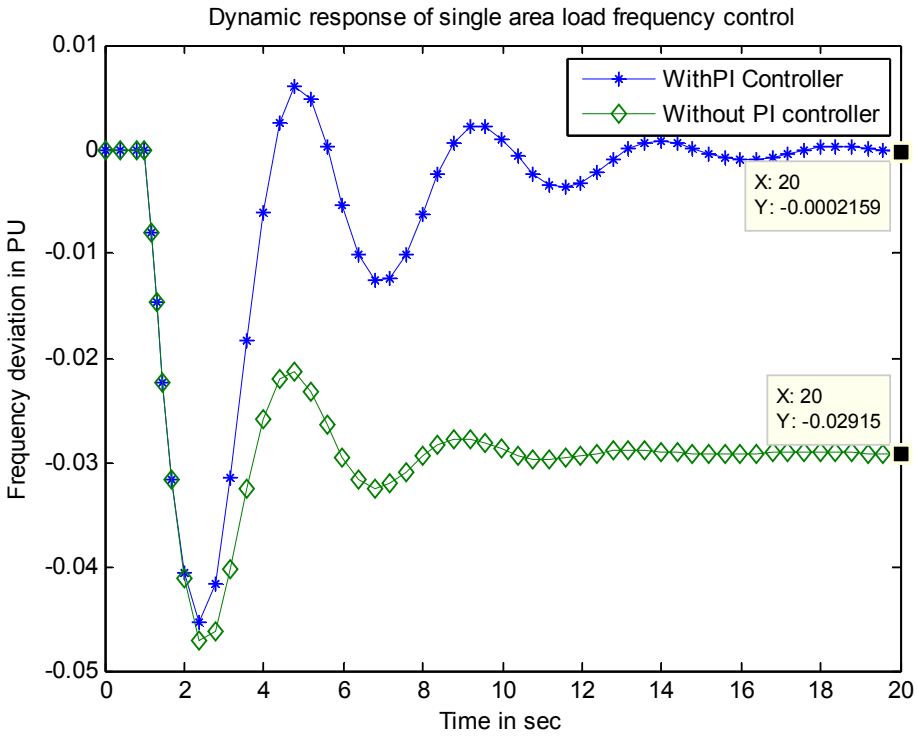
THEORETICAL MODEL:

THEORETICAL SOLUTION:

SIMULINK MODEL WITHOUT & WITH PI CONTROLLER:



EXPECTED OUTPUT:



Blocks used:

RESULT:

****

E9 - SIMULINK MODEL FOR TWO AREA LOAD FREQUENCY CONTROL

AIM: To find dynamic response of the given two - area load frequency control problem theoretically and to plot and verify the results in SIMULINK

PROBLEM:

The parameters for load frequency control of a two area are:

Speed governor gain	Ksg=1
Time constant of speed governor	Tsg=0.4
Speed regulation of speed governor	R=3
Gain of turbine	Kt=1
Time constant of turbine	Tt=0.5
Gain of power system	Kps=100
Time constant of power system	Tps=10
Proportional plus integral gain	Ki=0.07
Synchronizing co-efficient	Tr=0.05
Frequency bias	0.425s

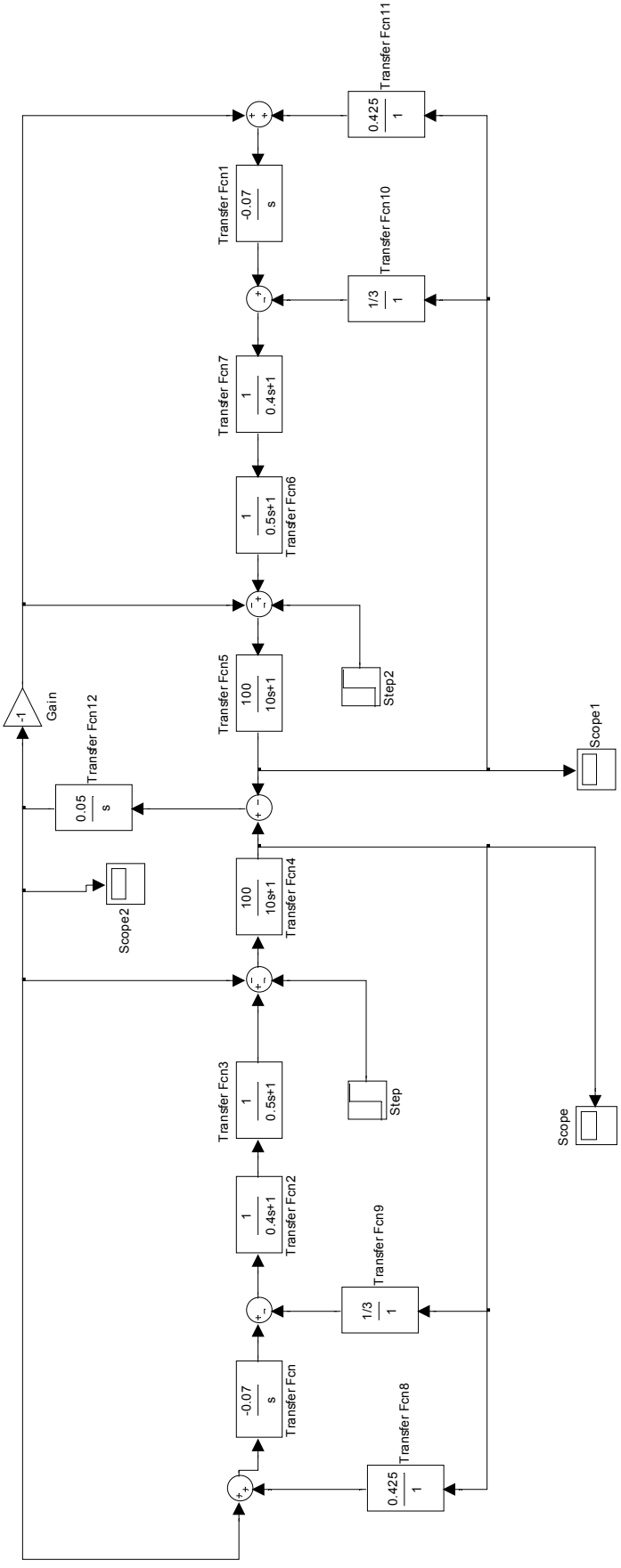
Develop a SIMULINK model for two area load frequency control with PI controller and obtain the frequency deviations in both areas and tie-line power deviations for a load change of 1pu in Area-2

THEORY:

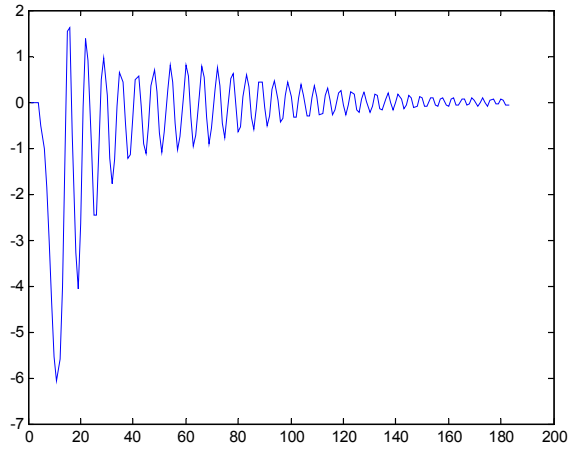
THEORETICAL MODEL:

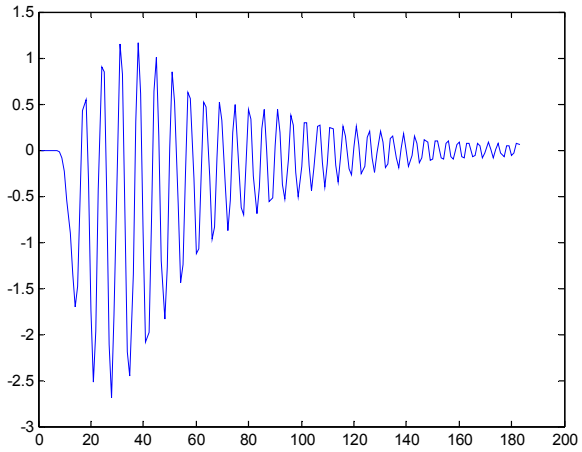
THEORETICAL SOLUTION:

SIMULINK MODEL:



EXPECTED OUTPUT:





Blocks used:

RESULT:

****

E10 - SIMULINK MODEL FOR EVALUATING TRANSIENT STABILITY OF SINGLE MACHINE CONNECTED TO INFINITE BUS

PROBLEM:

A 20 MVA, 50 Hz generator delivers 18 MW over a double circuit line to an infinite bus. The generator has KE of 2.52 MJ/MVA at rated speed. The generator transient reactance is $X'_{d=0.35\ pu.}$

Each transmission circuit has R=0 and a reactance of 0.2 pu on a 20 MVA base .|E′|=1.1 p u and infinite bus voltage V=1.0/_0^0.A three-phase short circuit occurs at the mid point of one of the transmission lines. Plot swing curves with fault cleared by simultaneous opening of breakers at both ends of the line at 2.5 cycles and 6.25 cycles after the occurrence of fault. Also plot the swing curve over the period of 0.5 s if the fault is sustained.

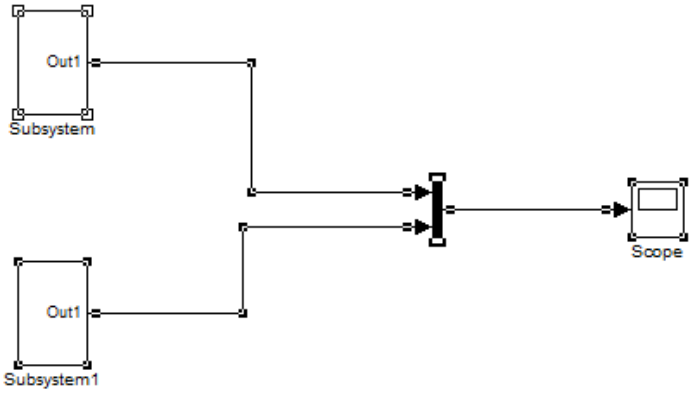
Simulate the problem in SIMULINK and compare with theoretical results.

Note: ***Before running simulation, integrator 1 has to be initialized to pre fault value of δ, i.e., δ_0. This can be done by double-clicking on integrator 1 block and changing the initial value from 0 to δ_0 (in radians). Also double click the switch block and change the threshold value from 0 to the fault clearing time (in sec).***

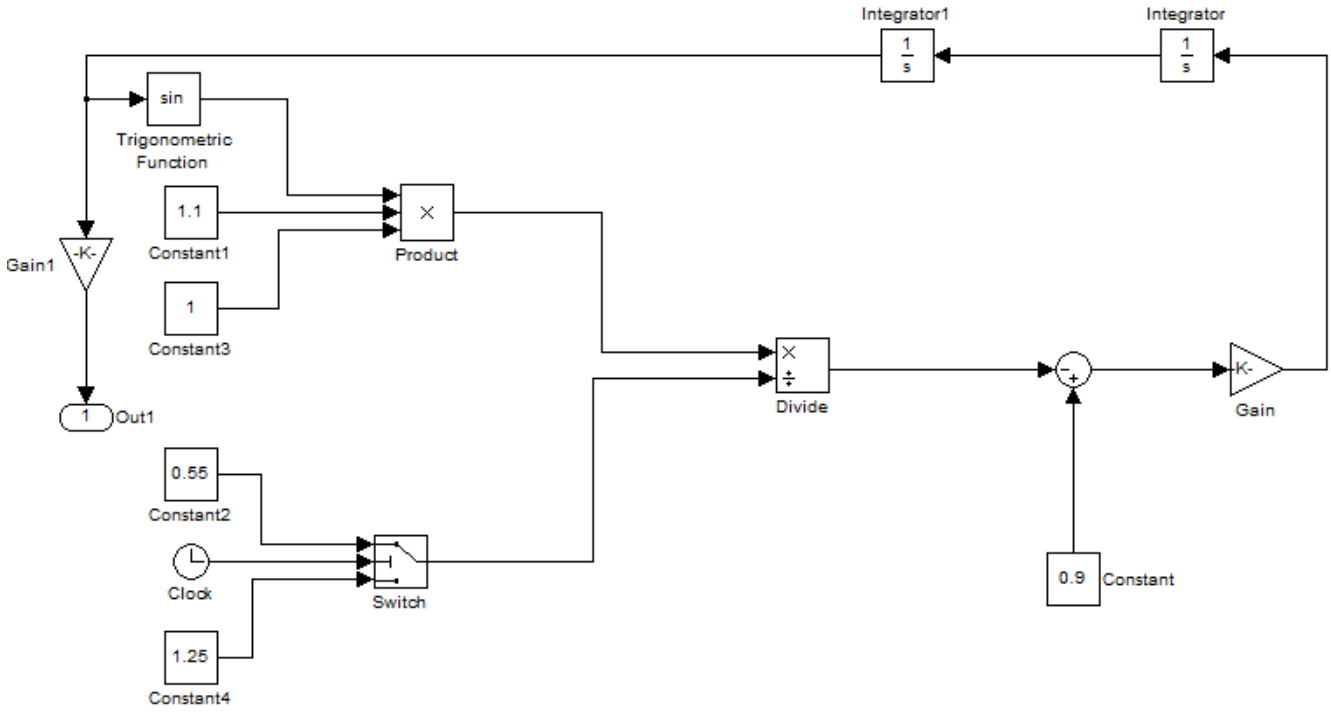
THEORETICAL SOLUTION:

This experiment is solution of swing equation of experiment 7 implemented in SIMULINK.

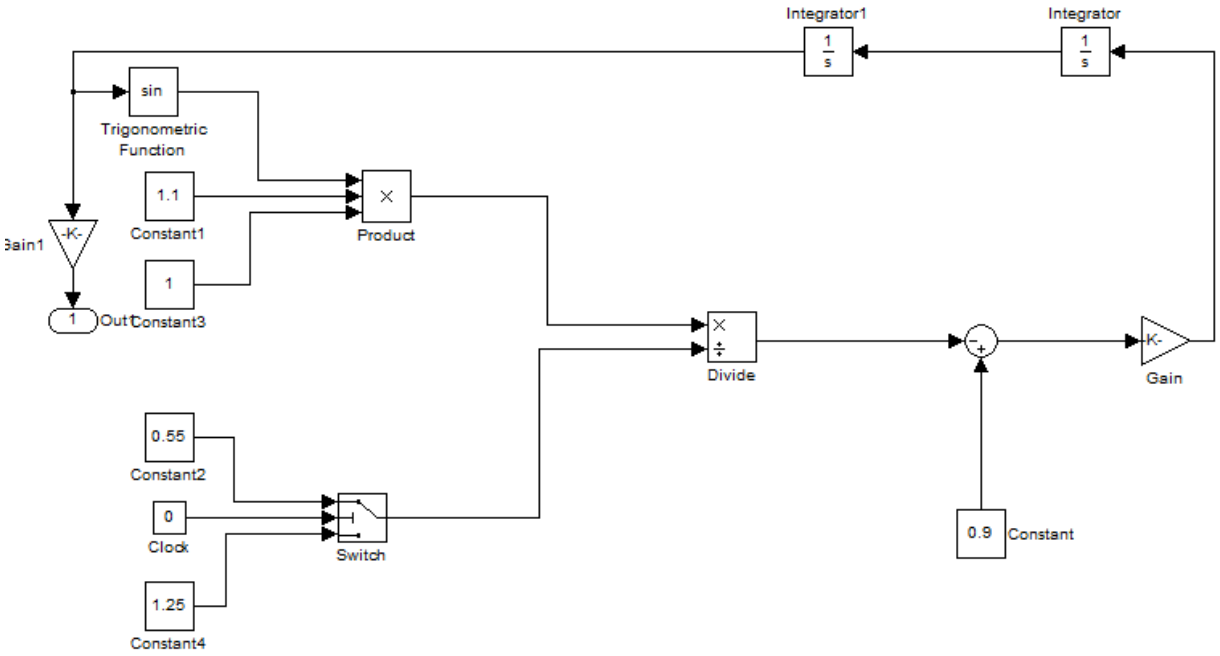
SIMULINK MODEL:



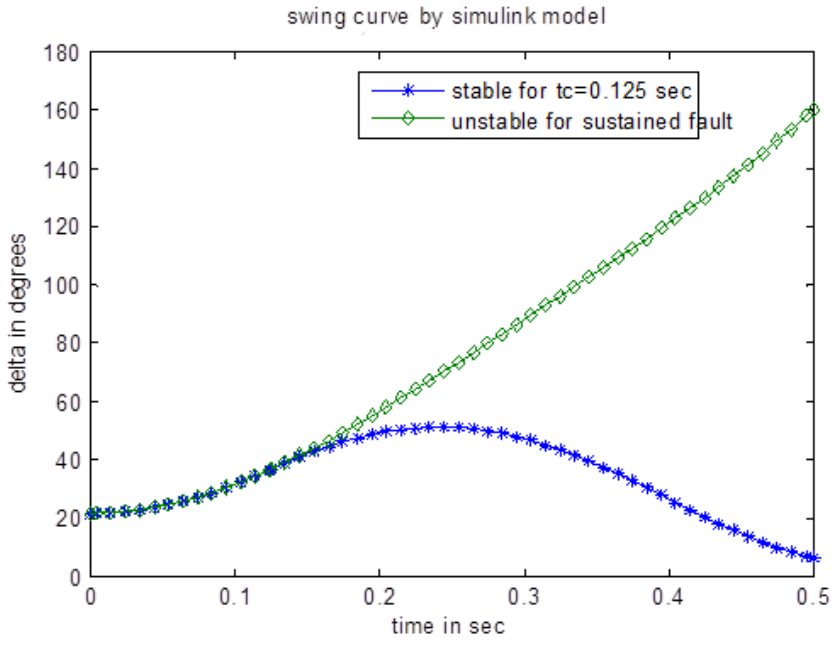
SUBSYSTEM DETAILS:



SUBSYSTEM1 DETAILS:



EXPECTED OUTPUT:



Blocks used:

RESULT:

****