Scilab Manual for Power System Simulation by Prof Jain B. Marshel Electrical Engineering St. Xavier's Catholic College of Engineering[1]

Solutions provided by
Prof Jain B.Marshel
Electrical Engineering
St.Xavier's Catholic College of Engineering

November 6, 2026

[1]Funded by a grant from the National Mission on Education through ICT, http://spoken-tutorial.org/NMEICT-Intro. This Scilab Manual and Scilab codes written in it can be downloaded from the "Migrated Labs" section at the website http://scilab.in

Contents

List of Scilab Solutions 4

1 Computation of Transmission line Parameters 7

2 Modelling of Transmission Lines 14

3 Formation of Bus Admittance matrix 20

4 Formation of Bus Impedance matrix 23

5 Load flow solution using Gauss-Seidal method 27

6 Load flow solution using Newton-Raphson method 33

7 Symmetrical Fault Analysis 40

8 Unsymmetrical Fault Analysis 45

9 Small Signal and transient Stability Analysis of Single-machine Infinite bus system 51

10 Small Signal and transient Stability Analysis of Multi machine Power Systems 55

11 Electromagnetic Transients in Power Systems 60

12 Load frequency dynamics of single Area Power Systems 62

13 Load frequency dynamics of two Area Power Systems 64

14 Economic dispatch in power systems neglecting losses **66**

15 Economic dispatch in power systems Including losses **69**

List of Experiments

Solution 1.1	Inductance of Single Phase line	7
Solution 1.2	Inductance of Three Phase line	8
Solution 1.3	Capacitance of Single Phase line	10
Solution 1.4	Capacitance of Three Phase line	11
Solution 2.1	Nominal T method	14
Solution 2.2	Nominal pi Method	16
Solution 3.1	Bus Admittance Matrix	20
Solution 4.1	Bus Impedance Matrix	23
Solution 5.1	Gauss Seidal Load Flow	27
Solution 6.1	Newton Raphson load Flow	33
Solution 7.1	Symmetrical Fault Analysis	40
Solution 8.1	Unsymmetrical Fault Analysis	45
Solution 9.1	SMIB Stability Analysis	51
Solution 10.1	Multimachine Stability Analysis	55
Solution 14.1	Economic Load Dispatch Excluding Losses	66
Solution 15.1	Economic Load Dispatch Including Losses	69

List of Figures

1.1 Inductance of Single Phase line 8
1.2 Inductance of Three Phase line 10
1.3 Capacitance of Single Phase line 12
1.4 Capacitance of Three Phase line 13

2.1 Nominal T method 16
2.2 Nominal pi Method 19

3.1 Bus Admittance Matrix 22

4.1 Bus Impedance Matrix 26

5.1 Gauss Seidal Load Flow 27
5.2 Gauss Seidal Load Flow 28

6.1 Newton Raphson load Flow 34
6.2 Newton Raphson load Flow 34

7.1 Symmetrical Fault Analysis 41
7.2 Symmetrical Fault Analysis 41

8.1 Unsymmetrical Fault Analysis 49
8.2 Unsymmetrical Fault Analysis 50

9.1 SMIB Stability Analysis 52
9.2 SMIB Stability Analysis 53

10.1 Multimachine Stability Analysis 59

11.1 Transient in RLC series circuit with DC source 61

12.1 Single Area Control 63

13.1 Two Area Control 65

14.1 Economic Load Dispatch Excluding Losses 67

15.1 Economic Load Dispatch Including Losses 70

Experiment: 1

Computation of Transmission line Parameters

Scilab code Solution 1.1 Inductance of Single Phase line

```
1 //Program to find loop inductance of a single phase
    transmission line//
2 //This program requires user input. Sample Problem
    and user input with output are available in the
    result file//
3 //Scilab Version 5.5.2 ; OS:Windows
4 clc;
5 clear;
6 d=input('Enter the spacing between conductors in
    meter:')
7 dia=input('Enter the diameter of the conductor in
    meter:')
8 r=dia/2
9 l=input('Enter the length of the line in km:')
10 li=10^(-7)*(1+4*(log(d/r)))*l*1000
11 disp(li,'The loop inductance for given transmission
    line in Henry is:')
12
13
```

PROBLEM :

A single phase line has two parallel conductors 2 meters apart. The diameter of each conductor is 1·2 cm. Calculate the loop inductance per km of the line.

OUTPUT:

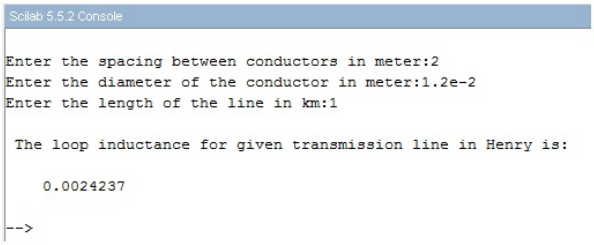


Figure 1.1: Inductance of Single Phase line

```
14 //SAMPLE INPUT:
15
16 //Enter the spacing between conductors in meter: 2
17 //Enter the diameter of the conductor in meter: 1.2e
     -2
18 //Enter the length of the line in km: 1
19
20  //OUTPUT:
21  //The loop inductance for given transmission line
      in Henry is:
22
23    // 0.0024237
```

Scilab code Solution 1.2 Inductance of Three Phase line

```
1 //Program to find loop inductance of a three phase
    transmission line assuming completely transposed
    line//
```

```
2 //This program requires user input. Sample Problem
    and user input with output are available in the
    result file//
3 //Scilab Version 5.5.2 ; OS:Windows
4 clc;
5 clear;
6 d12=input('Enter the distance between the conductor
    1 and 2 in meter:')
7 d23=input('Enter the distance between the conductor
    1 and 3 in meter:')
8 d31=input('Enter the distance between the conductor
    2 and 3 in meter:')
9 deq=(d12*d23*d31)^(1/3)
10 dia=input('Enter the diameter of the conductor in
    meter:')
11 r=dia/2
12 l=input('Enter the length of the line in km:')
13 li=10^(-7)*(0.5+2*(log(deq/r)))*l*1000
14 disp(li,'The loop inductance for given distance in
    Henry is:')
15
16
17 //SAMPLE INPUT:
18 //Enter the distance between the conductor 1 and 2
    in meter:2
19 //Enter the distance between the conductor 1 and 3
    in meter:2.5
20 // the distance between the conductor 2 and 3 in
    meter:4.5
21 //Enter the diameter of the conductor in meter:1.24e
    -2
22 //Enter the length of the line in km:1
23
24  //OUTPUT:
25  //The loop inductance for given distance in Henry
     is:
26
27    // 0.0012742
```

PROBLEM :

The three conductors of a 3-phase line are arranged at the corners of a triangle of sides 2 m, 2·5 m and 4·5 m. Calculate the inductance per km of the line when the conductors are regularly transposed. The diameter of each conductor is 1·24 cm.

OUTPUT:

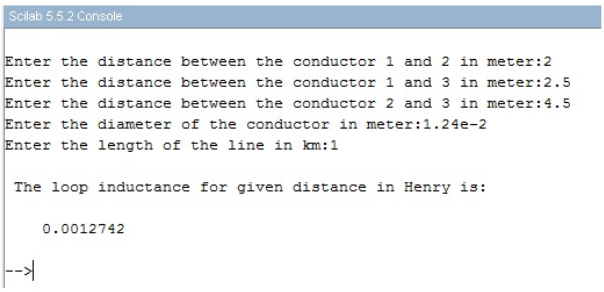


Figure 1.2: Inductance of Three Phase line

Scilab code Solution 1.3 Capacitance of Single Phase line

```
1 //Program to find the capacitance of a single phase
    transmission line//
2 //This program requires user input. Sample Problem
    and user input with output are available in the
    result file//
3 //Scilab Version 5.5.2 ; OS:Windows
4 clc;
5 clear;
6 dia=input('Enter the diameter of the conductor in
    meter:')
7 r=dia/2
8 d=input('Enter the spacing between the conductors in
```

```
    meter:')
l=input('Enter the length of the line in km:')
c=((%pi*8.854*10^(-12)*l*1000)/log(d/r))
disp(c,'the capacitance of the line for given
   distance is:')



//SAMPLE INPUT:
//Enter the diameter of the conductor in meter:2e-2
//Enter the spacing between the conductors in meter
   :3
//Enter the length of the line in km:1

 //OUTPUT:
 //the capacitance of the line for given distance is
    :

  //   4.877D-09
```

Scilab code Solution 1.4 Capacitance of Three Phase line

```
//Program to find line to neutral capacitance of a
   three phase transmission line assuming completely
    transposed line//
//This program requires user input. Sample Problem
   and user input with output are available in the
   result file//
//Scilab Version 5.5.2 ; OS:Windows
clc;
clear;
d12=input('Enter the distance between the conductor
   1 and 2 in meter:')
```

PROBLEM :

A single-phase transmission line has two parallel conductors 3 meters apart, diameter of each conductor being 2 cm. Calculate the capacitance of the line per km.

OUTPUT:

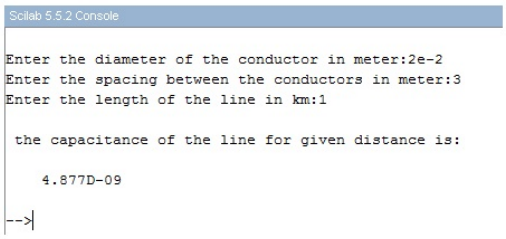


Figure 1.3: Capacitance of Single Phase line

```
7  d23=input('Enter the distance between the conductor
       1 and 3 in meter:')
8  d31=input('Enter the distance between the conductor
       2 and 3 in meter:')
9  deq=(d12*d23*d31)^(1/3)
10 dia=input('Enter the diameter of the conductor in
       meter:')
11 r=dia/2
12 l=input('Enter the length of the line in km:')
13 Cn=((2*%pi*8.85*10^-12)/(log(deq/r)))*l*1000
14 disp(Cn,'The line to neutral capacitance for given
       distance in Farad is:')
15
16
17 //SAMPLE INPUT:
18
19 //Enter the distance between the conductor 1 and 2
       in meter:4
20 //Enter the distance between the conductor 1 and 3
       in meter:4
21 //Enter the distance between the conductor 2 and 3
```

PROBLEM :

A 3-phase, 50 Hz, 132 kV overhead line has conductors placed in a horizontal plane 4 m apart. Conductor diameter is 2 cm. If the line length is 100 km, calculate the capacitance to neutral per phase assuming complete transposition.

OUTPUT:

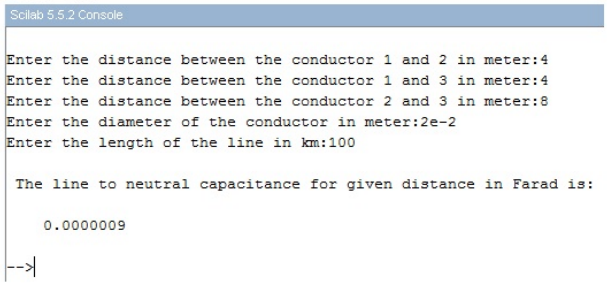


Figure 1.4: Capacitance of Three Phase line

```
   in meter:8
22 //Enter the diameter of the conductor in meter:2e-2
23 //Enter the length of the line in km:100
24
25  //OUTPUT:
26  //The line to neutral capacitance for given
       distance in Farad is:
27
28    // 0.0000009
```

Experiment: 2

Modelling of Transmission Lines

Scilab code Solution 2.1 Nominal T method

```
1 //Calculation of Transmission Line parameters using
    Nominal-T method//
2 //This program requires user input. Sample problem
    with user input and result are available in the
    result file//
3 //Scilab Version 5.5.2 ; OS:Windows
4 clc;
5 clear;
6 pl=input('Enter the power supplied to the load:');
7 vr=input('Enter the receiving end voltage:');
8 pf=input('Enter the power factor:');
9 spf=sin(acos(pf));
10 z=input('Enter the series impedance value of single
    conductor:');
11 y=input('Enter the shunt admittance value:');
12 e=(z*y)/2;
13 a=(1+e);                //calculation of transmission
    line parameters
14 b=z*(1+e/2);
```

```
15 c=y;
16 d=a;
17 disp(d,c,b,a,'The values of ABCD parameters
      respectively are')
18 vrph=vr/sqrt(3);       //receiving end voltage per
      phase
19 ir=pl/(sqrt(3)*vr*pf);          //receiving end current
20 irv=ir*(pf-%i*spf);             // receiving end
      current in vector form
21 vsph=(a*vrph+b*irv);            //sending end voltage
      per phase
22 vsh=abs(vsph);                  // magnitude of sending
       end voltage per phase
23 reg=((abs(vsh/a)-abs(vrph))/vrph)*100;       //
      calculation of percentage regulation
24 disp(reg,'regulation of the line is')
25
26
27 //SAMPLE INPUT:
28 //Enter the power supplied to the load:30e6
29 //Enter the receiving end voltage:132e3
30 //Enter the power factor:0.85
31 //Enter the series impedance value of single
      conductor:20+52*%i
32 //Enter the shunt admittance value:315e-6*%i
33
34
35  //OUTPUT:
36  //The values of ABCD parameters respectively are
37
38     // 0.99181 + 0.00315i
39
40      //19.8362 + 51.81856i
41
42      //0.000315i
43
44 //     0.99181 + 0.00315i
45
```

PROBLEM

A balanced 3-phase load of 30 MW is supplied at 132 KV, 50 Hz and 0.85 pf lagging by needs of a transmission line. The series impedance of a single conductor is (20+j52) ohms and the total phase neutral admittance is $315x10^{-6}$ Siemen. Using nominal T method determine Transmission line ABCD-parameters and the regulation of the line.

OUTPUT

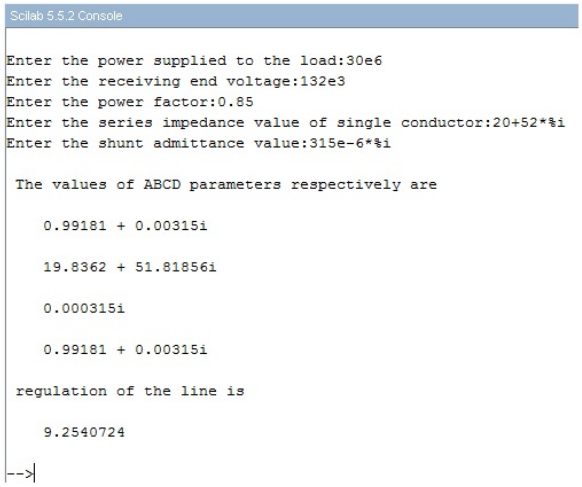


Figure 2.1: Nominal T method

```
46 //regulation of the line is
47
48   // 9.2540724
```

Scilab code Solution 2.2 Nominal pi Method

```
1 //Calculation of Transmission Line parameters using
    Nominal-pi method//
2 //This Program requires user input. Sample Problem
    with user input and result are available in the
```

```
    result file//
3 //Scilab Version 5.5.2 ; OS:Windows
4 clc;
5 clear;
6 d=input('Enter the value of distance:');
7 rkm=input('Enter the value of resistance per km:');
8 xlkm=input('Enter the value of inductive reactance
    per km:');
9 yshkm=input('Enter the value of shunt admittance per
     km:');
10 pl=input('Enter the value of power delivered:');
11 vl=input('Enter the value of line voltage:');
12 pf=input('Enter the value of power factor:');
13 vr=vl/sqrt(3);          //phase voltage
14 r=rkm*d;               //total resistance of the
    transmission line
15 xl=xlkm*d;             //total inductive reactance
    of the transmission line
16 ysh=yshkm*d;           //total shunt admittance of
    the transmission line
17 zs=r+(xl*%i);          //total impedance
18 a=1+(ysh*zs)/2;          //calculation of
    transmission line parameters
19 b=zs;
20 c=ysh*(1+(ysh*zs)/4);
21 d=a;
22 disp(d,c,b,a,'the values of ABCD parameters
    respectively are:')
23 ilo=pl/(sqrt(3)*vl*pf);
24 sp=sin(acos(pf));
25 ir=ilo*(pf-%i*sp);
26 icl=(vr*ysh)/2;
27 il=ir+icl;
28 vs=(vr+(il*(r+(%i*xl))));  //sending end voltage
29 reg=((abs(vs)/abs(a)-abs(vr))/abs(vr))*100;   //
    calculation of percentage regulation
30 disp(reg,'regulation of the line is');
31
```

```
32 //SAMPLE INPUT:
33 //Enter the value of distance:100
34 //Enter the value of resistance per km:0.1
35 //Enter the value of inductive reactance per km:0.2
36 //Enter the value of shunt admittance per km:4e-6*%i
37 //Enter the value of power delivered:10e6
38 //Enter the value of line voltage:66e3
39 //Enter the value of power factor:0.8
40
41  //OUTPUT:
42  //the values of ABCD parameters respectively are:
43
44     // 0.996 + 0.002i
45
46      //10. + 20.i
47
48 //   - 0.0000004 + 0.0003992i
49
50   //   0.996 + 0.002i
51
52  //regulation of the line is
53
54     // 5.8069405
```

PROBLEM

A 3 phase, 50 Hz, 100 Km line has a resistance, inductive reactance and capacitive shunt admittance of 0.1Ω/Km,0.2Ω/Km and $4x10^{-6}$ S/Km per phase. If the line delivers 10 MW at 110 KV and 0.8 pf lagging, determine the transmission line ABCD-parameters and the regulation of the line using nominal-pi method.

OUTPUT

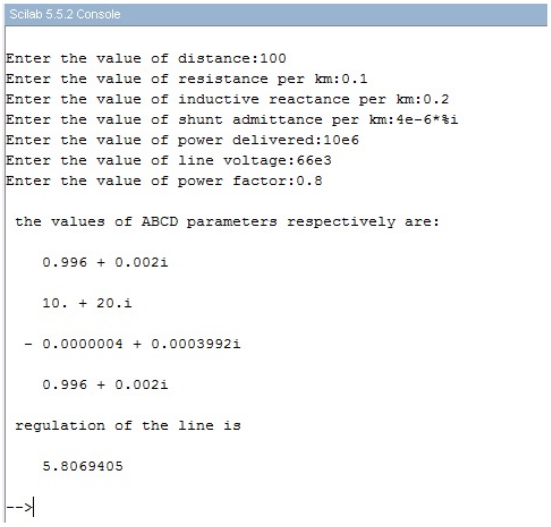


Figure 2.2: Nominal pi Method

Experiment: 3

Formation of Bus Admittance matrix

Scilab code Solution 3.1 Bus Admittance Matrix

```
1
2 //Program to find out bus admittance matrix of a
     power system of any size//
3 //This program requires user input. A sample problem
     with user input and output is available in the
     result file//
4 //Scilab Version 5.5.2 ; OS:Windows
5 clc;
6 clear;
7 linedata=input('Enter line data in order of strt bus
     ,end bus,series resistance,series reactance,shunt
      susceptance:')
8 sb=linedata(:,1)  //Starting bus number of all the
     lines stored in variable sb
9 eb=linedata(:,2)  //Ending bus number of all the
     lines stored in variable eb
10 lz=linedata(:,3)+linedata(:,4)*%i;  //lineimpedance=
     R+jX
11 sa=-linedata(:,5)*%i;    //shunt admittance=-jB
```

```
nb=max(max(sb,eb));
ybus=zeros(nb,nb);
for i=1:length(sb)
    m=sb(i);
    n=eb(i);
    ybus(m,m)=ybus(m,m)+1/lz(i)+sa(i);
    ybus(n,n)=ybus(n,n)+1/lz(i)+sa(i);
    ybus(m,n)=-1/lz(i);
    ybus(n,m)=ybus(m,n);
end
disp(ybus,'The Bus Admittance matrix is:')

//SAMPLE INPUT:

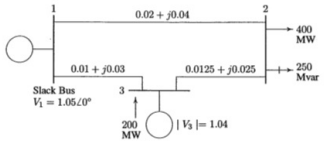
//Enter line data in order of strt bus,end bus,
   series resistance,series reactance,shunt
   susceptance:[1 2 0.02 0.04 0;1 3 0.01 0.03 0;2 3
   0.0125 0.025 0]

 //OUTPUT:
 //The Bus Admittance matrix is:

//    20. - 50.i  - 10. + 20.i  - 10. + 30.i
//  - 10. + 20.i    26. - 52.i  - 16. + 32.i
//  - 10. + 30.i  - 16. + 32.i    26. - 62.i
```

PROBLEM:

For the network shown in figure, determine the bus admittance matrix .



OUTPUT:

```
Scilab 5.5.2 Console

Enter line data in order of strt bus,end bus,series resistance,series reactance,shunt susceptance:[1 2 0.02 0.04 0;1 3 0.01 0.03 0;2 3 0.0125 0.025 0]

    20. - 50.i  - 10. + 20.i  - 10. + 30.i
  - 10. + 20.i    26. - 52.i  - 16. + 32.i
  - 10. + 30.i  - 16. + 32.i    26. - 62.i

-->
```

Figure 3.1: Bus Admittance Matrix

Experiment: 4

Formation of Bus Impedance matrix

Scilab code Solution 4.1 Bus Impedance Matrix

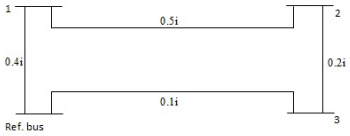
```
1
2 //Program to determine bus impedance matrix of a
     power system of any size using building algorithm
     //
3 //This program needs user input. Sample problem with
     user input and output is available in the result
     file//
4 //Scilab Version 5.5.2 ; OS:Windows
5 clc;
6 clear;
7 linedata=input('enter the line data values in the
     order of starting bus,ending bus,resistance and
     reactance:') //note:enter 0 for reference bus
8 sb=linedata(:,1)
9 eb=linedata(:,2)
10 z=linedata(:,3)+linedata(:,4)*%i    //impedance z=R+
     jX
11 zbus=[];
12 check=[];
```

```
13 for i=1:length(sb)
14     m=sb(i);
15     n=eb(i);
16     mn=min(m,n);
17     nm=max(m,n);
18     ncheck=length(find(check==nm));    //Variable
          used for checking whether bus nm is already
          existing
19     mcheck=length(find(check==mn));    //Variable
          used for checking whether bus mn is already
          existing
20     [rows columns]=size(zbus);
21 //Condition for connection of line between reference
     bus and new bus
22     if mn==0 & ncheck==0
23         zbus=[zbus zeros(rows,1);zeros(1,rows) z(i)
             ];
24         check=[check nm];
25 //Condition for connection of line between existing
    bus and new bus
26          else if mcheck>0 & ncheck==0
27             zbus=[zbus zbus(:,mn);zbus(mn,:) zbus(mn
                ,mn)+z(i)];
28             check=[check nm];
29 //Condition for connection of line between reference
     bus and existing bus
30             elseif mn==0 & ncheck>0
31             zbus=[zbus zbus(:,nm);zbus(nm,:) zbus(nm
                ,nm)+z(i)];
32             //Modifying Z bus size using Kron's
                reduction tehnique
33             zbusn=zeros(rows,rows);
34             for r=1:rows
35                 for t=1:columns
36                     zbusn(r,t)=zbus(r,t)-(zbus(r,
                        rows+1)*zbus(rows+1,t))/(zbus
                        (rows+1,rows+1));
37                 end
```

```
38              end
39              zbus=zbusn
40  //Condition for connection of line between two
       existing buses
41          elseif mcheck>0 & ncheck>0
42              zbus=[zbus zbus(:,nm)-zbus(:,mn);zbus(nm
                   ,:)-zbus(mn,:),z(i)+zbus(mn,nm)+zbus(
                   nm,nm)-2*zbus(nm,mn)];
43              //Modifying Z bus size using Kron's
                   reduction tehnique
44              zbusn=zeros(rows,rows);
45              for r=1:rows
46                  for t=1:columns
47                      zbusn(r,t)=zbus(r,t)-(zbus(r,rows
                           +1)*zbus(rows+1,t))/(zbus(rows
                           +1,rows+1));
48                  end
49              end
50              zbus=zbusn;
51          end
52      end
53  end
54  disp(zbus,'The bus impedance matrix is:');
55
56
57  //SAMPLE INPUT:
58
59  //enter the line data values in the order of
       starting bus,ending bus,resistance and reactance
       :[0 1 0 0.5;1 2 0 0.2;2 3 0 0.1;3 0 0 0.4]
60
61   //OUTPUT:
62   //The bus impedance matrix is:
63
64  //     0.2916667i    0.2083333i    0.1666667i
65  //     0.2083333i    0.2916667i    0.2333333i
66  //     0.1666667i    0.2333333i    0.2666667i
```

PROBLEM:

Determine the bus impedance matrix of the given power system, where the per unit values of line impedances are marked in the diagram.



OUTPUT:

```
Scilab 5.5.2 Console

enter the line data values in the order of starting bus,ending bus,resistance and reactance:[0 1 0 0.5;1 2 0 0.2;2 3 0 0.1;3 0 0 0.4]

    0.2916667i    0.2083333i    0.1666667i
    0.2083333i    0.2916667i    0.2333333i
    0.1666667i    0.2333333i    0.2666667i

-->
```

Figure 4.1: Bus Impedance Matrix

Experiment: 5

Load flow solution using Gauss-Seidal method

Scilab code Solution 5.1 Gauss Seidal Load Flow

```
1 //Program to find out power system voltage at the
    end of the iteration by gauss siedal method//
2 //This program requires user input. A sample problem
```

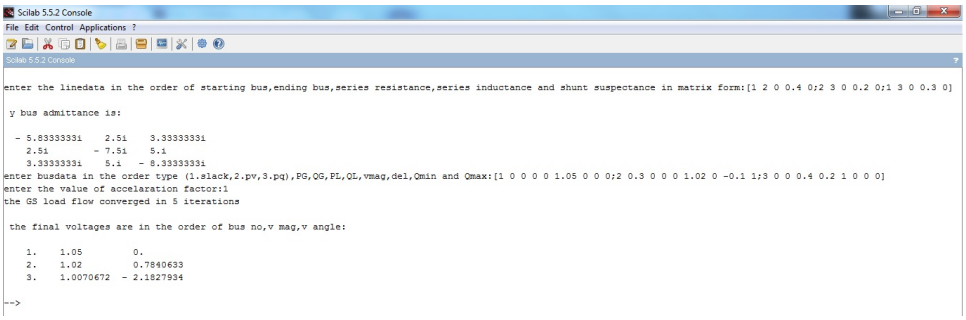
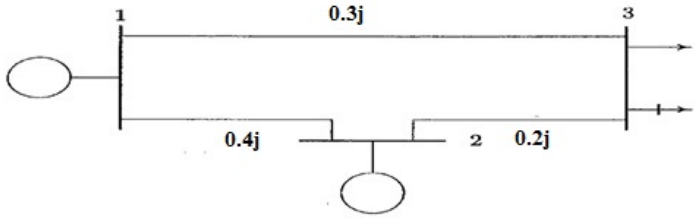


Figure 5.1: Gauss Seidal Load Flow

PROBLEM:

For the system shown in figure determine the voltage at the end of the iteration by gauss siedal method. Assume that base MVA as 100.



Bus no	Voltage	Generator		Load		Q min	Q max
	In p.u	P	Q	P	Q		
1	1.05	-	-	-	-	-	-
2	1.02	0.3	-		-	-10	100
3	-	-	-	0.4	0.2	-	-

Figure 5.2: Gauss Seidal Load Flow

```
      with user input and output is available in the
     result file//
3 //Question of example problem is available in file "
     GausssSeidalQuestionFile.jpg" and result is
     available in the file "GaussSeidalOutputFile.jpg"
4 //Scilab Version 5.5.2 ; OS:Windows
5 clc;
6 clear;
7 linedata=input('enter the linedata in the order of
     starting bus,ending bus,series resistance,series
     inductance and shunt suspectance in matrix form:'
     )
8 sb=linedata(:,1)  //Starting bus number of all the
     lines stored in variable sb //
9 eb=linedata(:,2)  //Ending bus number of all the
     lines stored in variable eb //
10 lz=linedata(:,3)+linedata(:,4)*%i   //lineimpedance=
     R+jX //
11 sa=-linedata(:,5)*%i  //shunt admittance=-jB //
12 nb=max(max(sb,eb));  //number of buses calculation
     //
13 y=zeros(nb,nb);
14 for i=1:length(sb)   // starting of admittance bus
     matrix calculation part //
15     m=sb(i);
16     n=eb(i);
17     y(m,m)=y(m,m)+1/lz(i)+(sa(i)/2);
18     y(n,n)=y(n,n)+1/lz(i)+(sa(i)/2);
19     y(m,n)=-1/lz(i);
20     y(n,m)=y(m,n);
21 end                  // end of admittance bus matrix
     calculation part  //
22 disp(y,'y bus admittance is:');
23 busdata=input('enter busdata in the order type (1.
     slack,2.pv,3.pq),PG,QG,PL,QL,vmag,del,Qmin and
     Qmax:')
24 typ=busdata(:,1) // type of all buses in the power
     system is stored in typ variable //
```

```
25 qmin=busdata(:,8) // minmum limit of Q for all the
      buses is stored in the variable qmin//
26 qmax=busdata(:,9)  // maximum limit of Q for all the
      buses is stored in the variable qmax//
27 p=busdata(:,2)-busdata(:,4) // real power of all the
      buses are calculated and is stored in the
      variable p //
28 q=busdata(:,3)-busdata(:,5)  // reactive power of
      all the buss are calculated and is stored in the
      variable q //
29 v=busdata(:,6).*(cosd(busdata(:,7))+%i*sind(busdata
      (:,7)));
30 alpha=input('enter the value of accelaration factor:
      ');
31 iter=1;
32 err=1;
33 vn(1)=v(1);
34 vold=v(1);
35 while abs(err)>5*10^(-5) //  starting of calculation
      part of bus voltage for first iteration //
36     for i=2:nb
37         sumyv=0;
38         for j=1:nb
39             sumyv=sumyv+y(i,j)*v(j);
40         end
41         if typ(i)==2
42             q(i)=-imag(conj(v(i)*sumyv));
43             if q(i)<qmin(i) |q(n)>qmax(i)
44                 vn(i)=(1/y(i,i))*(((p(i)-%i*q(i))/(
                       conj(v(i))))-(sumyv-y(i,i)*v(i)))
                       ;
45                 vold(i)=v(i);
46                 v(i)=vn(i);
47                 typ(i)=3
48             if q(i)<qmin(i)
49                 q(i)=qmin(i);
50             else
51                 q(i)=qmax(i);
```

```
52              end
53          else
54              vn(i)=(1/y(i,i))*(((p(i)-%i*q(i))/(conj(
                    v(i))))-(sumyv-y(i,i)*v(i)));
55              ang=atan(imag(vn(i)),real(vn(i)));
56              vn(i)=abs(v(i))*(cos(ang)+%i*sin(ang));
57              vold(i)=v(i);
58              v(i)=vn(i);
59          end
60          elseif typ(i)==3
61              vn(i)=(1/y(i,i))*(((p(i)-%i*q(i))/(conj(
                    v(i))))-(sumyv-y(i,i)*v(i)));
62              vold(i)=v(i);
63              v(i)=vn(i);
64          end
65          end
66  err=max(abs(abs(v)-abs(vold)));
67
68  iter=iter+1;
69  for i=2:nb
70      if err>5*10^(-6) &typ(i)==3
71      v(i)=vold(i)+alpha*(v(i)-vold(i));
72      end
73  end
74  end
75  printf('the GS load flow converged in %d iterations
       \n',iter-1);
76  nn=1:nb;
77  res=[nn' abs(v) (atan(imag(v),real(v)))*(180/%pi)]
78  disp(res,'the final voltages are in the order of bus
        no,v mag,v angle:');
79
80  //SAMPLE INPUT and OUTPUT
81  //enter the linedata in the order of starting bus,
       ending bus,series resistance,series inductance
       and shunt suspectance in matrix form:[1 2 0 0.4
       0;2 3 0 0.2 0;1 3 0 0.3 0]
82
```

```
83  //y bus admittance is:
84
85 //   - 5.8333333i     2.5i     3.3333333i
86 //     2.5i         - 7.5i     5.i
87 //     3.3333333i     5.i    - 8.3333333i
88 //enter busdata in the order type (1.slack,2.pv,3.pq
      ),PG,QG,PL,QL,vmag,del,Qmin and Qmax:[1 0 0 0 0
      1.05 0 0 0;2 0.3 0 0 0 1.02 0 -0.1 1;3 0 0 0.4
      0.2 1 0 0 0]
89 //enter the value of accelaration factor:1
90 //the GS load flow converged in 5 iterations
91
92  //the final voltages are in the order of bus no,v
      mag,v angle:
93
94 //     1.     1.05          0.
95 //     2.     1.02          0.7840633
96 //     3.     1.0070672   - 2.1827934
```

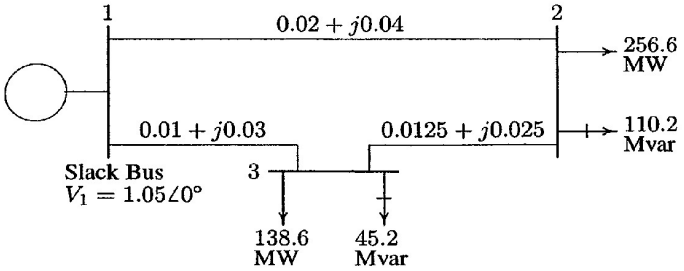
Experiment: 6

Load flow solution using Newton-Raphson method

Scilab code Solution 6.1 Newton Raphson load Flow

```
1
2 //Program to find out load flow solution using
    Newton Raphson method//
3 //This program requires user input. A sample problem
    with user input and output is available in the
    result file//
4 //Example problem is available in the file "
    NRQuestionFile.jpg" and user input and output is
    available in the file "NRResultFile"
5 //Scilab Version 5.5.2 ; OS:Windows
6
7 clear;
8 clc;
9 linedata=input('Enterlinedata in the order line no.,
    Frombus,Tobus,series resistance,series reactance,
```

Figure shows the one-line diagram of a simple three-bus power system with generation at bus 1. The magnitude of voltage at bus 1 is adjusted to 1.05 per unit. The scheduled loads at buses 2 and 3 are as marked on the diagram. Line impedances are marked in per unit on a 100-MVA base and the line charging susceptances are neglected.



Determine the bus voltages and angle of all 3 buses using Newton Raphson method.

Figure 6.1: Newton Raphson load Flow

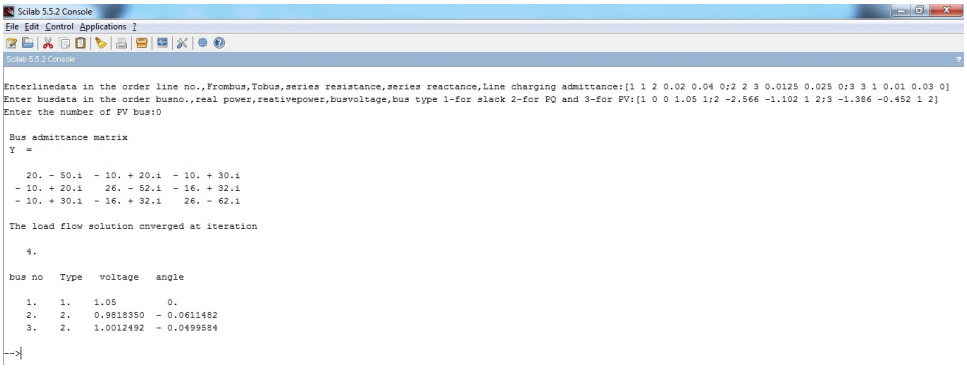


Figure 6.2: Newton Raphson load Flow

```
    Line charging admittance:');
10 Busdata=input('Enter busdata in the order busno.,
    real power,reativepower,busvoltage,bus type 1-for
     slack 2-for PQ and 3-for PV:');
11 npv=input('Enter the number of PV bus:');
12
13 //Determination of bus admittance matrix//
14 nb=max(Busdata(:,1));
15 nl=max(linedata(:,1));
16 Psp=Busdata(:,2);
17 Qsp=Busdata(:,3);
18 vsp=Busdata(:,4);
19 rem=Busdata(:,5);
20 Y=zeros(nb,nb);
21 sb=linedata(:,2);
22 eb=linedata(:,3);
23 z=linedata(:,4)+linedata(:,5)*%i;
24 ly=linedata(:,6);
25 for i=1:nl
26     m=sb(i)
27     n=eb(i);
28     Y(m,m)=Y(m,m)+1/z(i)+ly(i)/2;
29     Y(n,n)=Y(n,n)+1/z(i)+ly(i)/2;
30     Y(m,n)=-1/z(i);
31     Y(n,m)=Y(m,n);
32    end
33 disp('Bus admittance matrix')
34 print(%io(2),Y)
35
36 //NR Load flow//
37 absY=abs(Y);
38 thetaY=atan(imag(Y),real(Y));
39 v=vsp';
40 iteration=1;
41 ang=zeros(1,nb);
42 mismatch=ones(2*nb-2-npv,1);
43 while max(abs(mismatch))>0.0001
44     J1=zeros(nb-1,nb-1);
```

```
45      J2=zeros(nb-1,nb-npv-1);
46      J3=zeros(nb-npv-1,nb-1);
47      J4=zeros(nb-npv-1,nb-npv-1);
48      P=zeros(nb,1);
49      Q=P;
50      del_P=Q;
51      del_Q=Q;
52      del_del=zeros(nb-1,1);
53      del_v=zeros(nb-1-npv,1);
54      ang;
55      mag=abs(v);
56      for i=2:nb
57          for j=1:nb
58              P(i)=P(i)+mag(i)*mag(j)*absY(i,j)*cos(
                    thetaY(i,j)-ang(i)+ang(j));
59               if rem(i)~=3
60                   Q(i)=Q(i)+mag(i)*mag(j)*absY(i,j)*
                        sin(thetaY(i,j)-ang(i)+ang(j));
61                end
62            end
63            end
64  Q=-1*Q;
65  del_P=Psp-P;
66  del_Q=Qsp-Q;
67  for i=2:nb
68      for j=2:nb
69          if j~=i
70              J1(i-1,j-1)=-mag(i)*mag(j)*absY(i,j)*sin
                    (thetaY(i,j)-ang(i)+ang(j));
71              J2(i-1,j-1)=mag(i)*absY(i,j)*cos(thetaY(
                    i,j)-ang(i)+ang(j));
72              J3(i-1,j-1)=-mag(i)*mag(j)*absY(i,j)*cos
                    (thetaY(i,j)-ang(i)+ang(j));
73              J4(i-1,j-1)=-mag(i)*absY(i,j)*sin(thetaY
                    (i,j)-ang(i)+ang(j));
74          end
75      end
76  end
```

```
77  for i=2:nb
78      for j=1:nb
79          if j~=i
80              J1(i-1,i-1)=J1(i-1,i-1)+mag(i)*mag(j)*
                   absY(i,j)*sin(thetaY(i,j)-ang(i)+ang(
                   j));
81              J2(i-1,i-1)=J2(i-1,i-1)+mag(j)*absY(i,j)
                   *cos(thetaY(i,j)-ang(i)+ang(j));
82              J3(i-1,i-1)=J3(i-1,i-1)+mag(i)*mag(j)*
                   absY(i,j)*cos(thetaY(i,j)-ang(i)+ang(
                   j));
83              J4(i-1,i-1)=J4(i-1,i-1)+mag(j)*absY(i,j)
                   *sin(thetaY(i,j)-ang(i)+ang(j));
84          end
85      end
86      J2(i-1,i-1)=2*mag(i)*absY(i,i)*cos(thetaY(i,i))+
           J2(i-1,i-1);
87      J4(i-1,i-1)=-2*mag(i)*absY(i,i)*sin(thetaY(i,i))
           -J4(i-1,i-1);
88      end
89  J=[J1 J2;J3 J4]
90  lenJ=length(J1);
91  i=2;
92  j=1;
93  while j<=lenJ
94      if rem(i)==2
95          j=j+1;
96      else
97          J(:,length(J1)+j)=[];
98          lenJ=lenJ-1;
99      end
100 end
101 i=i+1;
102 lenJ=length(J1);
103 i=1;
104 j=2;
105 while i<=lenJ
106     if rem(j)==2
```

```
107           i=i+1;
108       else
109           J(length(J1)+i,:)=[];
110           lenJ=lenJ-1;
111           Q(i+1)=[]
112           del_Q(i+1,:)=[]
113       end
114     //   j=j+1;
115       end
116 P(1,:)=[]
117 Q(1,:)=[]
118 del_P(1,:)=[];
119 del_Q(1,:)=[];
120 mismatch=[del_P;del_Q];
121 del=J\mismatch;
122 del_del=del(1:nb-1);
123 del_v=del(nb:length(del));
124 ang=ang(2:nb)+del_del';
125 j=1;
126 for i=2:nb
127       if rem(i)==2
128           v(i)=v(i)+del_v(j);
129           j=j+1;
130       end
131 end
132 mag=abs(v);
133 ang=[0 ang];
134 nbr=1:nb;
135 iteration=iteration+1;
136 end
137 disp(iteration-1,'The load flow solution cnverged at
       iteration')
138 disp('bus no   Type   voltage   angle')
139 disp([nbr' rem mag' ang'])
140
141
142 //SAMPLE INPUT and OUTPUT:
143 //Enterlinedata in the order line no.,Frombus,Tobus,
```

```
    series resistance,series reactance,Line charging
    admittance:[1 1 2 0.02 0.04 0;2 2 3 0.0125 0.025
    0;3 3 1 0.01 0.03 0]
144 //Enter busdata in the order busno.,real power,
    reativepower,busvoltage,bus type 1-for slack 2-
    for PQ and 3-for PV:[1 0 0 1.05 1;2 -2.566 -1.102
     1 2;3 -1.386 -0.452 1 2]
145 //Enter the number of PV bus:0
146
147 // Bus admittance matrix
148 // Y =
149
150 //    20. - 50.i  - 10. + 20.i  - 10. + 30.i
151 //  - 10. + 20.i    26. - 52.i  - 16. + 32.i
152 //  - 10. + 30.i  - 16. + 32.i    26. - 62.i
153
154 // The load flow solution cnverged at iteration
155
156 //    4.
157
158 // bus no   Type   voltage   angle
159
160 //    1.    1.    1.05         0.
161 //    2.    2.    0.9818350  - 0.0611482
162 //    3.    2.    1.0012492  - 0.0499584
```

Experiment: 7

Symmetrical Fault Analysis

Scilab code Solution 7.1 Symmetrical Fault Analysis

```
1 //Program to find out fault current,post-fault
    voltages and line flow of a given network//
2 //This program requires user input. A sample problem
     with user input and output is available in the
    result files. Question is available in the file "
    SymmetricalFaultQuestionFile.jpg" and result is
    available in the file "SymmetricalFaultResultFile
    .jpg"//
3 //Scilab Version 5.5.2 ; OS:Windows
4 clc;
5 clear;
6 linedata=input('enter the line data values in the
    order of starting bus,nding bus,resistance and
    reactance:')
7 f=input('enter the bus at wich fault occurs:')
8 bv=input('enter the pre-fault bus voltage:')
9 sb=linedata(:,1)  //Starting bus number of all the
```

PROBLEM:

The generators at buses 1 and 3 of the network has impedances j1.5p.u. If a 3ϕ short circuit fault occurs at bus 2 ,when there is no load(all bus voltages are equal to 1.0 p.u),find initial symmetrical current in fault in the line 1-3 and post fault voltages using bus building algorithm.

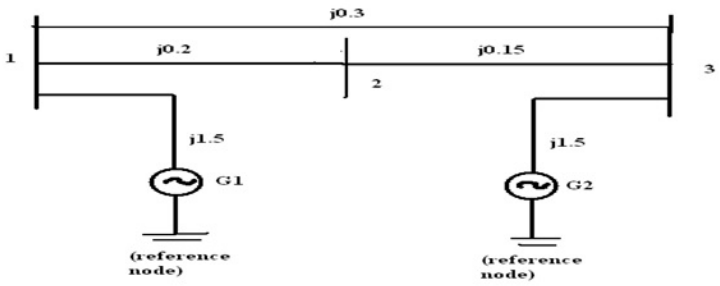


Figure 7.1: Symmetrical Fault Analysis

```
Scilab 5.5.2 Console

enter the line data values in the order of starting bus,nding bus,resistance and reactance:[0 1 0 1.5;1 2 0 0.2;2 3 0 0.15;3 0 0 1.5;1 3 0 0.3]
enter the bus at which fault occurs:2
enter the pre-fault bus voltage:1

 the impedance matrix is:

    0.7745830i    0.7464881i    0.7254170i
    0.7464881i    0.8362160i    0.7535119i
    0.7254170i    0.7535119i    0.7745830i

 the fault current is:

  - 1.1958633i
)
 the post fault voltages v1,v2,v3 respectively are:

    0.1073022

    0

    0.0989028
enter the starting bus,ending bus and the impedance between them to calculate the line flow:1
enter the ending bus to calculate the line flow:3
enter the impedance between the above buses:0.3*%i

 the line flow current is:

  - 0.0279980i
```

Figure 7.2: Symmetrical Fault Analysis

```
    lines stored in variable sb //
eb=linedata(:,2)  //Ending bus number of all the
    lines stored in variable eb //
z=linedata(:,3)+linedata(:,4)*%i  //lineimpedance=R+
    jX //
zbus=[];
check=[];
for i=1:length(sb)  //starting of impedance matrix
    calculation part//
     m=sb(i);
     n=eb(i);
     mn=min(m,n);
     nm=max(m,n);
     ncheck=length(find(check==nm));
     mcheck=length(find(check==mn));
     [rows columns]=size(zbus);
     if mn==0 & ncheck==0
         zbus=[zbus zeros(rows,1);zeros(1,rows) z(i)
            ];
         check=[check nm];
          else if mcheck>0 & ncheck==0
             zbus=[zbus zbus(:,mn);zbus(mn,:) zbus(mn
                ,mn)+z(i)];
             check=[check nm];
             elseif mn==0 & ncheck>0
             zbus=[zbus zbus(:,nm);zbus(nm,:) zbus(nm
                ,nm)+z(i)];
             zbusn=zeros(rows,rows);
             for r=1:rows
                 for t=1:columns
                     zbusn(r,t)=zbus(r,t)-(zbus(r,
                        rows+1)*zbus(rows+1,t))/(zbus
                        (rows+1,rows+1));
                 end
             end
             zbus=zbusn
         elseif mcheck>0 & ncheck>0
             zbus=[zbus zbus(:,nm)-zbus(:,mn);zbus(nm
```

```
                    ,:)-zbus(mn,:),z(i)+zbus(mn,nm)+zbus(
                    nm,nm)-2*zbus(nm,mn)];
39              zbusn=zeros(rows,rows);
40              for r=1:rows
41                  for t=1:columns
42                      zbusn(r,t)=zbus(r,t)-(zbus(r,rows
                            +1)*zbus(rows+1,t))/(zbus(rows
                            +1,rows+1));
43                  end
44              end
45              zbus=zbusn;
46          end
47      end
48  end   //ending of impedance bus matrix calculation
        part//
49  disp(zbus,'the impedance matrix is:');
50  ifa=bv/zbus(f,f)  //calculation of fault current//
51  disp(ifa,'the fault current is:')
52  disp('the post fault voltages v1,v2,v3 respectively
        are:');
53  for i=1:n
54  v(i)=bv-(ifa*zbus(i,f)); //calculation of postfault
        bus voltages//
55  disp(v(i));
56  end
57  a=input('enter the starting bus to calculate the
        line flow:');
58  b=input('enter the ending bus to calculate the line
        flow:');
59  zs=input('enter the impedance between the above
        buses:');
60  i13=(v(a)-v(b))/zs; //calculation oflineflows//
61  disp(i13,'the line flow current is:')
62
63  //SAMPLE INPUT and OUTPUT:
64  //enter the line data values in the order of
        starting bus,nding bus,resistance and reactance
        :[0 1 0 1.5;1 2 0 0.2;2 3 0 0.15;3 0 0 1.5;1 3 0
```

```
    0.3]
65 //enter the bus at wich fault occurs:2
66 //enter the pre-fault bus voltage:1
67
68 // the impedance matrix is:
69
70 //    0.7745830i    0.7464881i    0.7254170i
71 //    0.7464881i    0.8362160i    0.7535119i
72 //    0.7254170i    0.7535119i    0.7745830i
73
74 // the fault current is:
75
76 //  - 1.1958633i
77
78 // the post fault voltages v1,v2,v3 respectively are
    :
79
80 //    0.1073022
81
82 //    0
83
84 //    0.0989028
85 //enter the starting bus to calculate the line flow
    :1
86 //enter the ending bus to calculate the line flow:3
87 //enter the impedance between the above buses:0.3*%i
88
89 // the line flow current is:
90
91 //  - 0.0279980i
```

Experiment: 8

Unsymmetrical Fault Analysis

Scilab code Solution 8.1 Unsymmetrical Fault Analysis

```
1
2 //Program to find out unsymmetrical fault current//
3 //This program requires user input. A sample problem
     with user input and output is available in the
    result file. Question is available in the file"
    UnsymmetricalFaultQuestionFile.jpg" and result is
     available in the file"
    UnsymmetricalFaultResultFile.jpg"//
4 //Scilab Version 5.5.2 ; OS:Windows
5 clc ;
6 clear;
7 a=input('Enter the positve sequence,negative
    sequence and zero sequence of first generator in
    matrix form:')
8 PG1=a(:,1);//positive sequence of generator 1 is
    stored in the variable PG1
9 NG1=a(:,2);//negative sequence of generator 1 is
    stored in the variable NG1
10 ZG1=a(:,3);//negative sequence of generator 1 is
    stored in the variable ZG1
11 b=input('Enter the positve sequence,negative
```

```
      sequence and zero sequence of first transformer
      in matrix form:')
12 PT1=b(:,1);//positive sequence of transformer 1 is
      stored in the variable PT1
13 NT1=b(:,2);//positive sequence of transformer 1 is
      stored in the variable NT1
14 ZT1=b(:,3);//positive sequence of transformer 1 is
      stored in the variable ZT1
15 c=input('Enter the positve sequence,negative
      sequence and zero sequence of first transmission
      line in matrix form:')
16 PTL=c(:,1);//positive sequence of transmission line
      1 is stored in the variable PTL
17 NTL=c(:,2);//positive sequence of transmission line
      1 is stored in the variable NTL
18 ZTL=c(:,3);//positive sequence of transmission line
      1 is stored in the variable ZTL
19 d=input('Enter the positve sequence,negative
      sequence and zero sequence of second transformer
      in matrix form:')
20 PT2=d(:,1);//positive sequence of transformer is
      stored in the variable PT2
21 NT2=d(:,2);//positive sequence of transformer 1 is
      stored in the variable NT2
22 ZT2=d(:,3);//positive sequence of transformer 1 is
      stored in the variable ZT2
23 e=input('Enter the positive sequence,negative
      sequence and zero sequence of second generator in
       matrix form:')
24 PG2=e(:,1);//positive sequence of transformer 1 is
      stored in the variable PG2
25 NG2=e(:,2);//positive sequence of transformer 1 is
      stored in the variable NG2
26 ZG2=e(:,3);//positive sequence of transformer 1 is
      stored in the variable ZG2
27 MVAB=input('Enter the value of base MVA:');
28 KVB=input('Enter the value of base KV:');
29 z1=((PG1*%i+PT1*%i)*(PTL*%i+PT2*%i+PG2*%i))/((PG1*%i
```

```
    +PT1*%i)+(PTL*%i+PT2*%i+PG2*%i));//calculation of
     positive impedence
30  z2=((NG1*%i+NT1*%i)*(NTL*%i+NT2*%i+NG2*%i))/((NG1*%i
    +NT1*%i)+(NTL*%i+NT2*%i+NG2*%i));//calculation of
     negative impedence
31  z0=((ZG1*%i+ZT1*%i)*(ZTL*%i+ZT2*%i+ZG2*%i))/((ZG1*%i
    +ZT1*%i)+(ZTL*%i+ZT2*%i+ZG2*%i));//calculation of
     zero impedence
32  Ib=(MVAB*(10^6))/((1.732*KVB*(10^3)))//calculating
    base current
33  disp(z0,z2,z1,'the values of positive(z1) negative(
    z2),zero(z0) sequence impedance respectively are
    ');
34  disp('OPTION','LG FAULT=1','LL FAULT=2','LLG FAULT=3
    ');
35  MENU=input('Enter the choice of fault:')
36  if MENU==1 //calculating Line to Ground fault
37      If=(3*(1))/(z0+z1+z2)
38      FAULTCURRENT=If*Ib;
39      disp(FAULTCURRENT,'The fault current is :');
40  end
41  if MENU==2//Calculating Line to Line Fault
42      If=((-1.732j)*(1))/(z1+z2)
43      FAULTCURRENT=If*Ib;
44  disp(FAULTCURRENT,'The fault current is :');
45  end
46  if MENU==3//calculating Line-Line-Ground fault
47      z=(z0*z2)/(z0+z2);
48      Ia1=(1)/(z1+z);
49      Ia0=((-1+(Ia1*z1))/z0);
50      If=3*Ia0;
51      FAULTCURRENT=If*Ib;
52   disp(FAULTCURRENT,'The fault current is :');
53  end
54
55  //SAMPLE INPUT and OUTPUT
56
57  //Enter the positve sequence,negative sequence and
```

```
    zero sequence of first generator in matrix form
    :[0.32 0.26 0.09]
58 //Enter the positve sequence,negative sequence and
    zero sequence of first transformer in matrix form
    :[0.23 0.23 0.23]
59 //Enter the positve sequence,negative sequence and
    zero sequence of first transmission line in
    matrix form:[0.56 0.56 0.09]
60 //Enter the positve sequence,negative sequence and
    zero sequence of second transformer in matrix
    form:[0.16 0.16 0.16]
61 //Enter the positive sequence,negative sequence and
    zero sequence of second generator in matrix form
    :[0.38 0.24 0.15]
62 //Enter the value of base MVA:100
63 //Enter the value of base KV:110
64
65 // the values of positive(z1) negative(z2),zero(z0)
    sequence impedance  respectivel
66 //      y are
67
68 //    0.3666667i
69
70 //    0.3244138i
71
72 //    0.1777778i
73
74 // LLG FAULT=3
75
76 // LL FAULT=2
77
78 // LG FAULT=1
79
80 // OPTION
81 //Enter the choice of fault:3
82
83 // The fault current is :
84
```

Unsymmetrical Fault

PROBLEM

Find the positive, negative and zero sequence for a given power system. Also find LG, LL, LLG fault current.
G1: Z1 = 0.32; Z2 = 0.26; Z3=0.09
T1: Z1 = 0.23; Z2 = 0.23; Z3=0.23
T2: Z1 = 0.16; Z2 = 0.16; Z3=0.16
Transmission Line: Z1 = Z2 = 0.56; Z3 = 0.09
G2: Z1 = 0.38; Z2 = 0.24; Z3 = 0.15

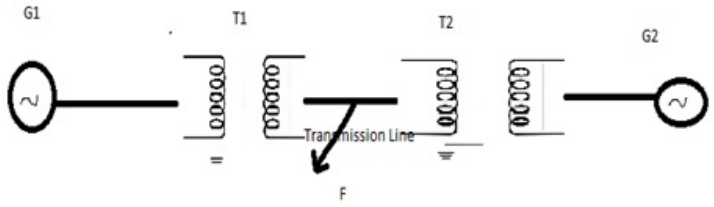


Figure 8.1: Unsymmetrical Fault Analysis

```
85 //      2112.5397 i
```

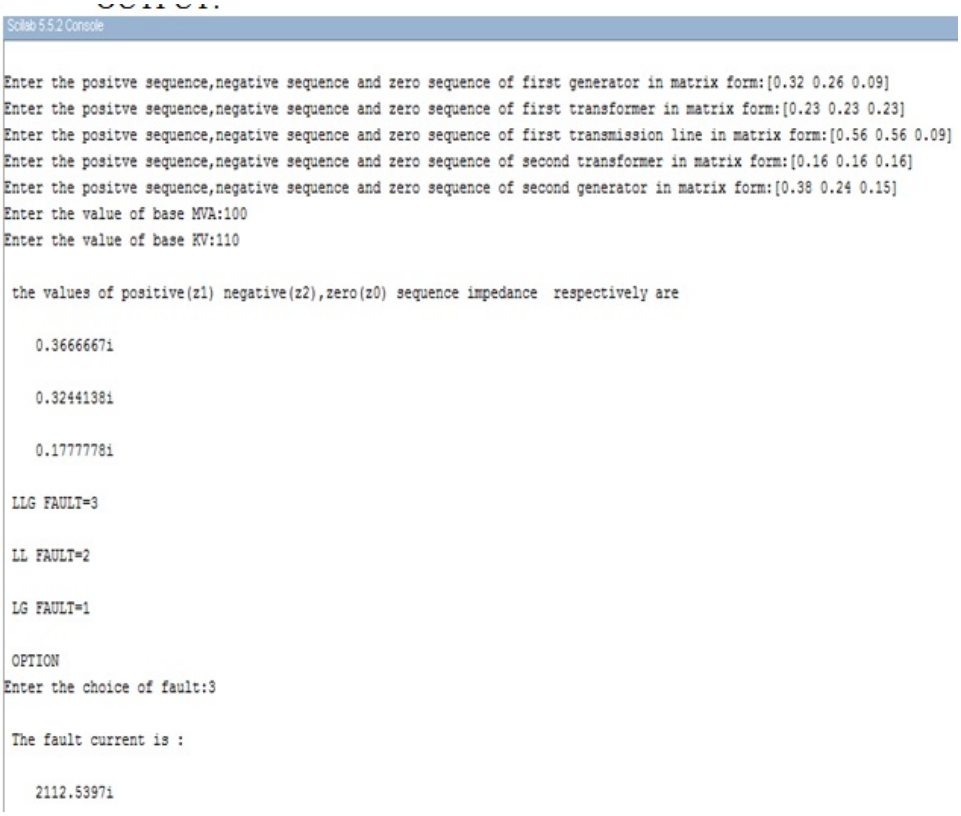


Figure 8.2: Unsymmetrical Fault Analysis

Experiment: 9

Small Signal and transient Stability Analysis of Single-machine Infinite bus system

Scilab code Solution 9.1 SMIB Stability Analysis

```
1 //Program to find out transient stability analysis
     of single machine - infinite bus system //
2 //An example problem and outputs are available in
     files 'result1' and 'result2'
3 //Scilab Version 5.5.2 ; OS:Windows
4
5 clc;
6 clear;
7
8 xd=0.3;
9
```

Consider a synchronous machine characterized by the following parameters:

$$X_d = 1.0 \quad X_q = 0.6 \quad X'_d = 0.3 \quad \text{per unit}$$

and negligible armature resistance. The machine is connected directly to an infinite bus of voltage 1.0 per unit. The generator is delivering a real power of 0.5 per unit at 0.8 power factor lagging. Determine the voltage behind transient reactance neglecting the saliency effect · **Also find the power angle curve.**

Resut

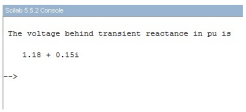


Figure 9.1: SMIB Stability Analysis

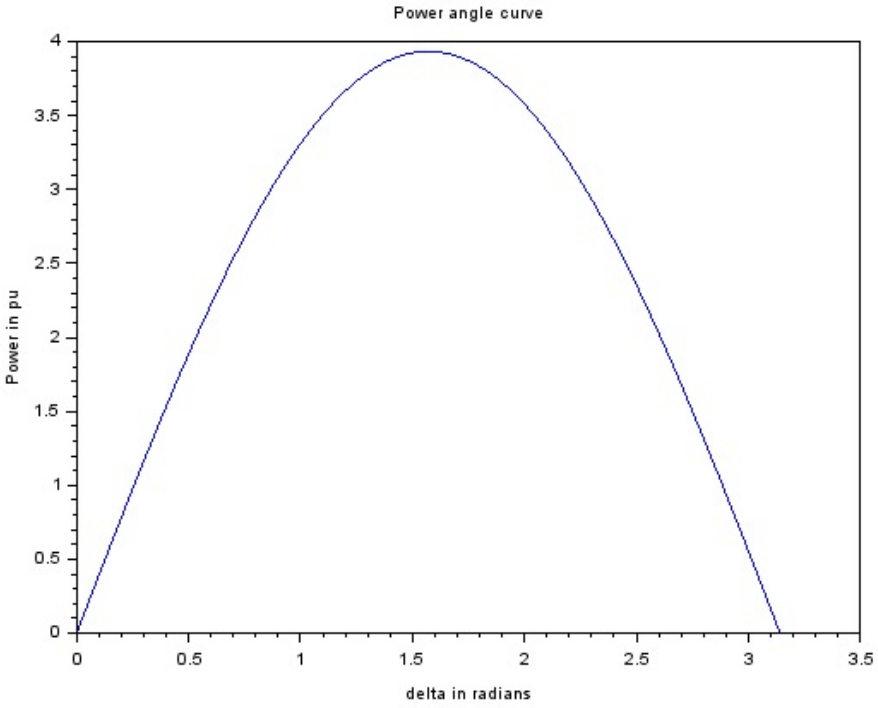


Figure 9.2: SMIB Stability Analysis

```
10 theta=acos(0.8);                                    //
     Power factor angle
11
12 S=(0.5/0.8)*cos(theta)+%i*sin(theta);               //
     Apparant Power
13 V=1;                                                //
     Prefault vltage is assumed to be 1 pu
14 Ia=(conj(S)/V);                                      //
     Pefault steady state current
15
16 E=V+(%i*xd)*(Ia);                                   //
     Voltage behind transient reactance
17
18 disp(E,'The voltage behind transient reactance in pu
     is ')
19
20 //To find the power angle curve
21
22 delta=0:0.001:%pi;
23 P=((E*V)/xd)*(sin(delta));
24
25 plot(delta,P)
26 xlabel('delta in radians')
27 ylabel('Power in pu')
28 title('Power angle curve')
```

Experiment: 10

Small Signal and transient Stability Analysis of Multi machine Power Systems

Scilab code Solution 10.1 Multimachine Stability Analysis

```
1 //Program to find out transient stability analysis
     of multi machine at the end of the iteration //
2 //This program requires user input. A sample problem
     with user input and output is available in the
     result file//
3 //Scilab Version 5.5.2 ; OS:Windows
4 //program for transient stability analysis of multi
     machine//
5 clc;
6 clc;
7 clear;
8 f=input('enter the frequency:');
9 bv=input('enter the base value in MVA:');
10 v=input('enter the value of bus voltage in p.u:');
11 e=input('enter the value of transient reactance
     voltage in pu:');
12 ld=input('enter the total load:');
```

```
13 x1=input('enter the prefault reactance in p.u:');
14 x2=input('enter the post fault reactance value:');
15 x3=input('enter during the fault reactance value:');
16 delt=input('enter the time interval in seconds:');
17 H=input('enter the inertia constant:');
18 pe1=ld/bv;
19
20 pe2=0;
21 delnot=asin((pe1*x1)/(e*v));
22
23 omeganot=2*3.14*f;
24
25
26 ddel=omeganot-(2*3.14*f);
27
28 ddelomega=(((3.14*f)*(pe1-pe2))/H);
29
30 //end of first step at t=0.05sec
31 del1=(delnot+(ddel*delt));//predicted values
32
33 delomega1=ddel+(ddelomega*delt);
34
35 //derivation at the end of t=0.05sec
36 ddel1=ddel+(ddelomega*delt);
37
38 ddelomega=(((3.14*f)*(pe1-pe2))/H);
39
40 delc1=delnot+((delt/2)*(ddel+ddel1));
41
42 delomegac1=ddel+((delt/2)*(ddelomega+ddelomega));
43
44 ddelc1=ddel+((delt/2)*(ddelomega+ddelomega));
45
46 ddelomegac=(((3.14*f)*(pe1-pe2))/H);
47
48 delp2=delc1+ddelc1*delt;
49
50 delomegap2=(delomegac1+(ddelomega*delt));
```

```
51
52 ddelomegap=(((3.14*f)*(pe1-pe2))/H);
53
54
55 delc2=delc1+(delt/2*(ddelc1+delomegap2));
56
57 delomegac=(delomegac1+(ddelomega*delt));
58
59 ddelc2=(delomegac1+(ddelomega*delt));
60
61 ddelomegac2=(((3.14*f)*(pe1-pe2))/H);
62
63 delp3=delc2+delomegac*delt;
64
65 delomega3=delomegac+ddelomegac2*delt;
66
67 //derivation at the end of t=0.15sec
68 disp('The final values at the end of t=0.15 sec are
     displayed below')
69 ddelp3=delomegac+ddelomegac2*delt;
70 disp(ddelp3,'ddelp3=');
71 ddelomega3=(((3.14*f)*(pe1-pe2))/H);
72 disp(ddelomega3,'ddelomega3=');
73 disp('corrected values');
74 delc3=delc2+((delt/2)*(delomegac+delomega3));
75 disp(delc3,'delc3=');
76 delomegac3=delomegac+((delt/2)*(ddelomega3+
     ddelomega3));
77 disp(delomegac3,'delomegac3=');
78
79 //SAMPLE INPUT:
80 //enter the frequency:50
81 //enter the base value in MVA:500
82 //enter the value of bus voltage in p.u:1
83 //enter the value of transient reactance voltage in
     pu:450/400
84 //enter the total load:460
85 //enter the prefault reactance in p.u:0.5
```

```
86 //enter the post fault reactance value:0.75
87 //enter during the fault reactance value:1
88 //enter the time interval in seconds:0.05
89 //enter the inertia constant:2.5
90
91
92  //OUTPUT
93 // The final values at the end of t=0.15 sec are
      displayed below
94
95 // ddelp3=
96
97 //     8.6664
98
99 // ddelomega3=
100
101 //     57.776
102
103 // corrected values
104
105 // delc3=
106
107 //     1.0712162
108
109 // delomegac3=
110
111 //     8.6664
```

Transient Stability of Multi Machine

A 50 HZ, 500 MVA, 400 KV generator (with transformer) is connected to a 400 kV infinite bus bar through an interconnector. The generator has H= 2.5 MJ/MVA, voltage behind transient reactance of 450kV and is loaded 460 MW. The transfer reactance between generator and bus bar under various conditions are:

Prefault: 0.5 p.u
During fault: 1.0 p.u.
Post fault: 0.75 p.u.

Calculate swing equation using intervals of 0.05 sec and assuming that the fault is cleared at 0.15 sec.

OUTPUT:

Scilab 5.5.2 Console

```
enter the frequency:50
enter the base value in MVA:500
enter the value of bus voltage in p.u:1
enter the value of transient reactance voltage in pu:450/400
enter the total load:460
enter the prefault reactance in p.u:0.5
enter the post fault reactance value:0.75
enter during the fault reactance value:1
enter the time interval in seconds:0.05
enter the inertia constant:2.5

 The final values at the end of t=0.15 sec are displayed below

 ddelp3=

    8.6664

 ddelomega3=

    57.776

 corrected values

 delc3=

    1.0712162

 delomegac3=

    8.6664
```

Figure 10.1: Multimachine Stability Analysis

Experiment: 11

Electromagnetic Transients in Power Systems

This code can be downloaded from the website wwww.scilab.in

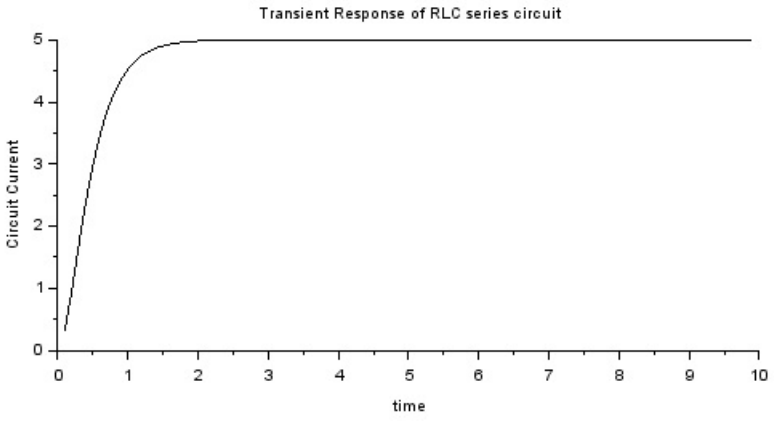


Figure 11.1: Transient in RLC series circuit with DC source

Experiment: 12

Load frequency dynamics of single Area Power Systems

This code can be downloaded from the website wwww.scilab.in

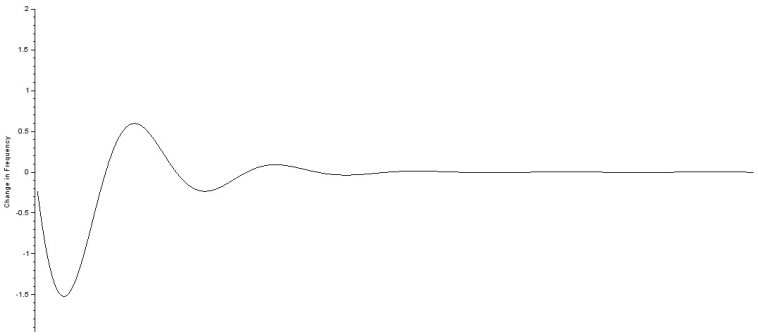


Figure 12.1: Single Area Control

Experiment: 13

Load frequency dynamics of two Area Power Systems

This code can be downloaded from the website wwww.scilab.in

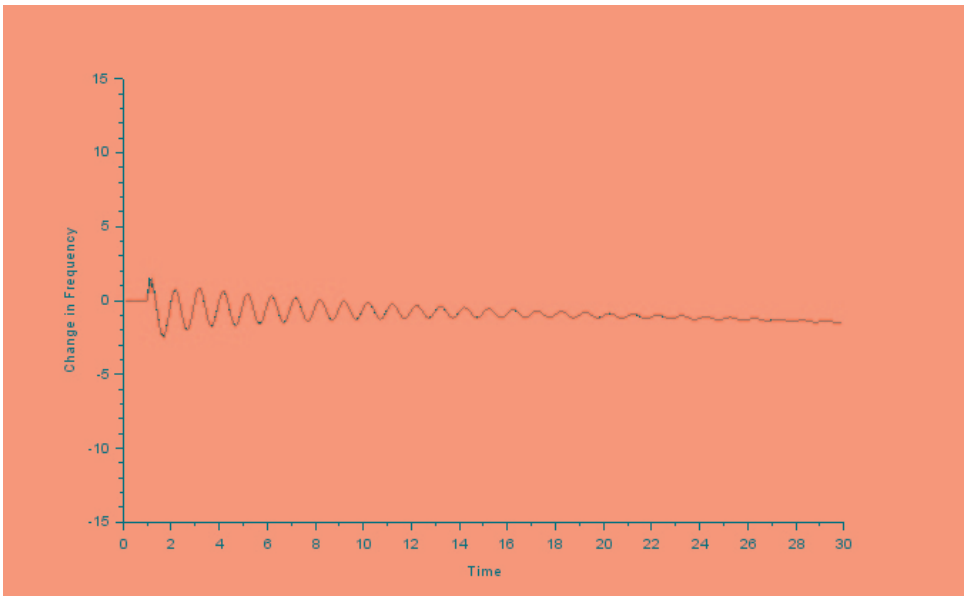


Figure 13.1: Two Area Control

Experiment: 14

Economic dispatch in power systems neglecting losses

Scilab code Solution 14.1 Economic Load Dispatch Excluding Losses

```
1 //Program to find out Economic load dispatch
     neglecting losses//
2 //This program requires user input. A sample problem
     with user input and output is available in the
     result file//
3 //Question and result of example problem is
     available in file "EDwithoutLoss.jpg"
4 //Scilab Version 5.5.2 ; OS:Windows
5
6 clear;
7 clc;
8 n=input('Enter no. of units :');
9 F=input('Enter the cost coefficient in matrix form :
     ');
10 constraint=input('Enter min and max values of P for
     all units:');
11 pd=input('Enter total demand:');
```

Determine the economic generation schedule of three generating unit in a power system to meet a system load of 275 MW. The Cost equations and the operating limits of each unit are given below.

$F_1 = 0.05P_1^2+23.5P_1+700 \quad ; \quad 40 <= P_1 <= 150$
$F_2 = 0.2P_2^2+20P_2+850 \quad ; \quad 40 <= P_2 <= 150$
$F_3 = 0.09P_3^2+18P_3+960 \quad ; \quad 40 <= P_3 <= 150$

OUTPUT

Scilab 5.5.2 Console

```
Enter no. of units :3
Enter the cost coefficient in matrix form :[0.05 23.5 700;0.2 20 850;0.09 18 960]
Enter min and max values of P for all units:[40 150;40 150;40 150]
Enter total demand:275

 The optimum schedule is:
 P  =

    130.53846
    41.384615
    103.07692
```

Figure 14.1: Economic Load Dispatch Excluding Losses

```
a=F(:,1);b=F(:,2);c=F(:,3);
Pmin=constraint(:,1);
Pmax=constraint(:,2);
chk=zeros(n,1);
rem=1;
while rem==1
    sx=0;sy=0;
    for i=1:n
        if i~=chk(i)
            sx=sx+b(i)/(2*a(i));
            sy=sy+1/(2*a(i));
        end
    end
    lamda=(pd+sx)/sy;
    sch=0;
    for i=1:n
       if i~=chk(i)
            P(i)=(lamda-b(i))/(2*a(i));
            if P(i)<Pmin(i)|P(i)>Pmax(i)
                if P(i)< Pmin(i)
                    P(i)=Pmin(i);
```

```
33                  else
34                      P(i)=Pmax(i);
35                  end
36                  pd=pd-P(i);
37                  chk(i)=i;
38                  sch=sch+1;
39              end
40          end
41          if sch==0
42              rem=0;
43          else
44              rem=1;
45          end
46      end
47  end
48  disp('The optimum schedule is:')
49  print(%io(2),P)
50
51  //SAMPLE INPUT
52  //Enter no. of units :3
53  //Enter the cost coefficient in matrix form :[0.05
       23.5 700;0.2 20 850;0.09 18 960]
54  //Enter min and max values of P for all units:[40
       150;40 150;40 150]
55  //Enter total demand:275
56
57   //OUTPUT
58  // The optimum schedule is:
59  // P  =
60
61  //     130.53846
62  //     41.384615
63  //     103.07692
```

Experiment: 15

Economic dispatch in power systems Including losses

Scilab code Solution 15.1 Economic Load Dispatch Including Losses

```
1 //Program for Economic Load Dispatch problem
     including loss coefficients//
2 //This program requires user input. A sample problem
     with user input and output is available in the
     result file named "EDwithLoss.jpg"//
3 //Scilab Version 5.5.2 ; OS:Windows
4 clear;
5 clc;
6 n=input('Enter no. of units :');
7 B=input('Enter the loss coefficient in matrix form :
     ');
8 a=B(:,1);//loss coefficients stored in variable a
9 b=B(:,2);//loss coefficients stored in variable b
10 c=B(:,3);//loss coefficients stored in variable c
11 pg=input('Enter the power of the units in matrix
     form in p.u:');
12 bv=input('Enter the base value');
```

ECONOMIC DISPATCH PROBLEM INCLUDING LOSSES

The transmission loss coefficients are given by
B=

0.01	-0.0003	-0.0002
-0.0003	0.0025	-0.0005
-0.0002	-0.0005	0.0031

Three plants A, B, C supply powers of 50 MW, 100 MW and 200 MW respectively. Calculate the transmission loss in the network in p. u value and the incremental transmission loss of the three plants. Assume base value= 200MVA.

Scilab 5.5.2 Console

```
Enter no. of units :3
Enter the loss coefficient in matrix form :[0.01 -0.0003 -0.0002;-0.0003 0.0025 -0.0005;-0.0002 -0.0005 0.0031]
Enter the power of the units in matrix form in p.u:[50/200 100/200 200/200]
Enter the base value200


 The transmission power loss in pu is

    0.003675

 The incremental losses in pu are

    0.0043
    0.00135
    0.0056
```

Figure 15.1: Economic Load Dispatch Including Losses

```
pl=0;
for i=1:n//calculation of power loss
    for j=1:n
        pl=pl+pg(j)*B(i,j)*pg(i);
    end
end
disp(pl,'The transmission power loss in pu is',);
ITL=zeros(n,1);//Calculation of incremental
   transmission loss
for i=1:n
    for j=1:n
        ITL(i)=ITL(i)+2*B(i,j)*pg(j);

        end
end
disp(ITL,'The incremental losses in pu are');

//SAMPLE INPUT:

//Enter no. of units :3
//Enter the loss coefficient in matrix form :[0.01
   -0.0003 -0.0002;-0.0003 0.0025 -0.0005;-0.0002
   -0.0005 0.0031]
//Enter the power of the units in matrix form in p.u
   :[50/200 100/200 200/200]
//Enter the base value200

 //OUTPUT
 //The transmission power loss in pu is

//    0.003675

// The incremental losses in pu are

//    0.0043
//    0.00135
//    0.0056
```