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

May 11, 2025

[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 Experiments

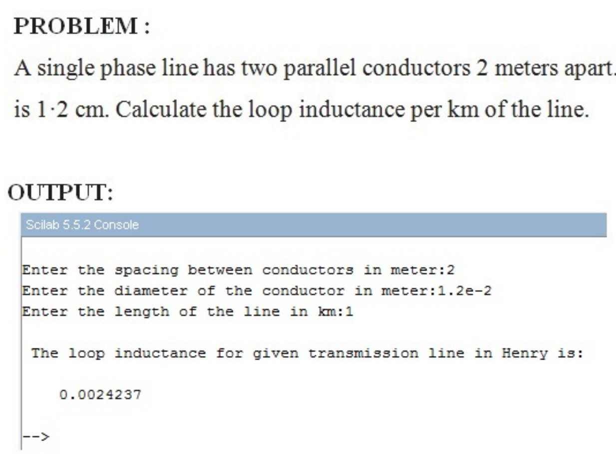
List of Figures

Experiment: 1

Computation of Transmission line Parameters

Scilab code Solution 1.1 Inductance of Single Phase line

```
//Program to find loop inductance of a single phase
   transmission 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;
d=input('Enter the spacing between conductors in
   meter:')
dia=input('Enter the diameter of the conductor in
   meter:')
r=dia/2
l=input('Enter the length of the line in km:')
li=10^(-7)*(1+4*(log(d/r)))*l*1000
disp(li,'The loop inductance for given transmission
   line in Henry is:')


```


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:

```
Scilab 5.5.2 Console

Enter the spacing between conductors in meter:2
Enter the diameter of the conductor in meter:1.2e-2
Enter the length of the line in km:1

 The loop inductance for given transmission line in Henry is:

    0.0024237

-->
```

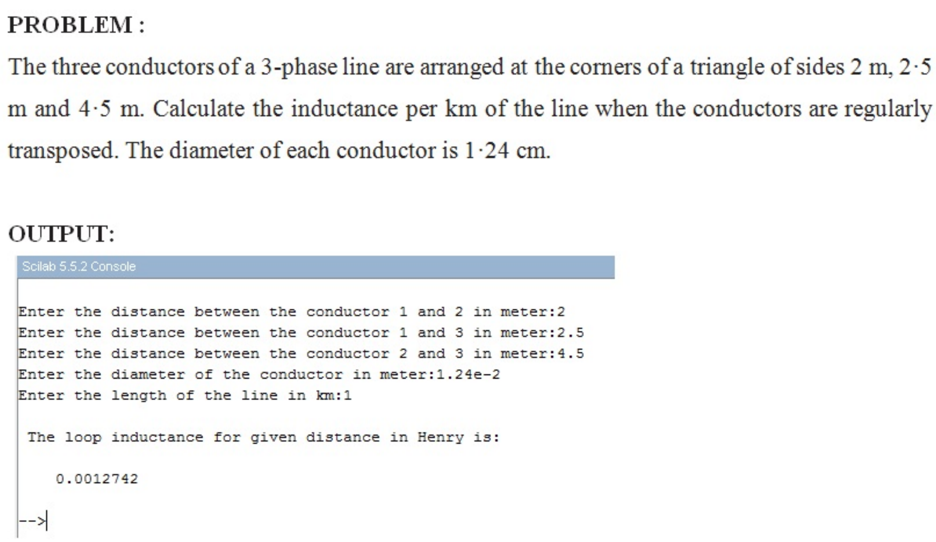
Figure 1.1: Inductance of Single Phase line

```
//SAMPLE INPUT:

//Enter the spacing between conductors in meter: 2
//Enter the diameter of the conductor in meter: 1.2e-2
//Enter the length of the line in km: 1

 //OUTPUT:
 //The loop inductance for given transmission line in Henry is:

   //  0.0024237
```

Scilab code Solution 1.2 Inductance of Three Phase line

```
//Program to find loop inductance 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:')
d23=input('Enter the distance between the conductor
   1 and 3 in meter:')
d31=input('Enter the distance between the conductor
   2 and 3 in meter:')
deq=(d12*d23*d31)^(1/3)
dia=input('Enter the diameter of the conductor in
   meter:')
r=dia/2
l=input('Enter the length of the line in km:')
li=10^(-7)*(0.5+2*(log(deq/r)))*l*1000
disp(li,'The loop inductance for given distance in
   Henry is:')


//SAMPLE INPUT:
//Enter the distance between the conductor 1 and 2
   in meter:2
//Enter the distance between the conductor 1 and 3
   in meter:2.5
// the distance between the conductor 2 and 3 in
   meter:4.5
//Enter the diameter of the conductor in meter:1.24e
   -2
//Enter the length of the line in km:1

 //OUTPUT:
 //The loop inductance for given distance in Henry
    is:

   // 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:

```
Scilab 5.5.2 Console

Enter the distance between the conductor 1 and 2 in meter:2
Enter the distance between the conductor 1 and 3 in meter:2.5
Enter the distance between the conductor 2 and 3 in meter:4.5
Enter the diameter of the conductor in meter:1.24e-2
Enter the length of the line in km:1

 The loop inductance for given distance in Henry is:

    0.0012742

-->
```

Figure 1.2: Inductance of Three Phase line

Scilab code Solution 1.3 Capacitance of Single Phase line

```
//Program to find the capacitance of a single phase
    transmission 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;
dia=input('Enter the diameter of the conductor in
    meter:')
r=dia/2
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:')
```

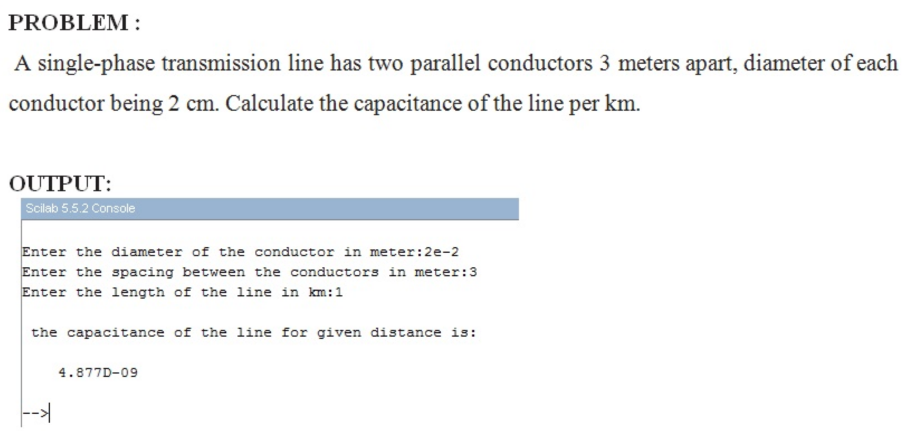
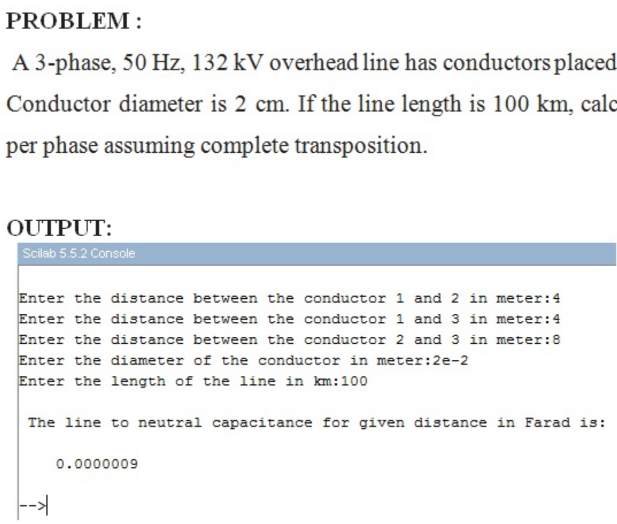


Figure 1.3: Capacitance of Single Phase line

```
d23=input('Enter the distance between the conductor 1 and 3 in meter:')
d31=input('Enter the distance between the conductor 2 and 3 in meter:')
deq=(d12*d23*d31)^(1/3)
dia=input('Enter the diameter of the conductor in meter:')
r=dia/2
l=input('Enter the length of the line in km:')
Cn=((2*%pi*8.85*10^-12)/(log(deq/r))*l*1000
disp(Cn,'The line to neutral capacitance for given distance in Farad is:')


//SAMPLE INPUT:

//Enter the distance between the conductor 1 and 2 in meter:4
//Enter the distance between the conductor 1 and 3 in meter:4
//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:

```
Scilab 5.5.2 Console

Enter the distance between the conductor 1 and 2 in meter:4
Enter the distance between the conductor 1 and 3 in meter:4
Enter the distance between the conductor 2 and 3 in meter:8
Enter the diameter of the conductor in meter:2e-2
Enter the length of the line in km:100

 The line to neutral capacitance for given distance in Farad is:

    0.0000009

-->
```

Figure 1.4: Capacitance of Three Phase line

```
    in meter:8
//Enter the diameter of the conductor in meter:2e-2
//Enter the length of the line in km:100

 //OUTPUT:
 //The line to neutral capacitance for given
    distance in Farad is:

   // 0.0000009
```

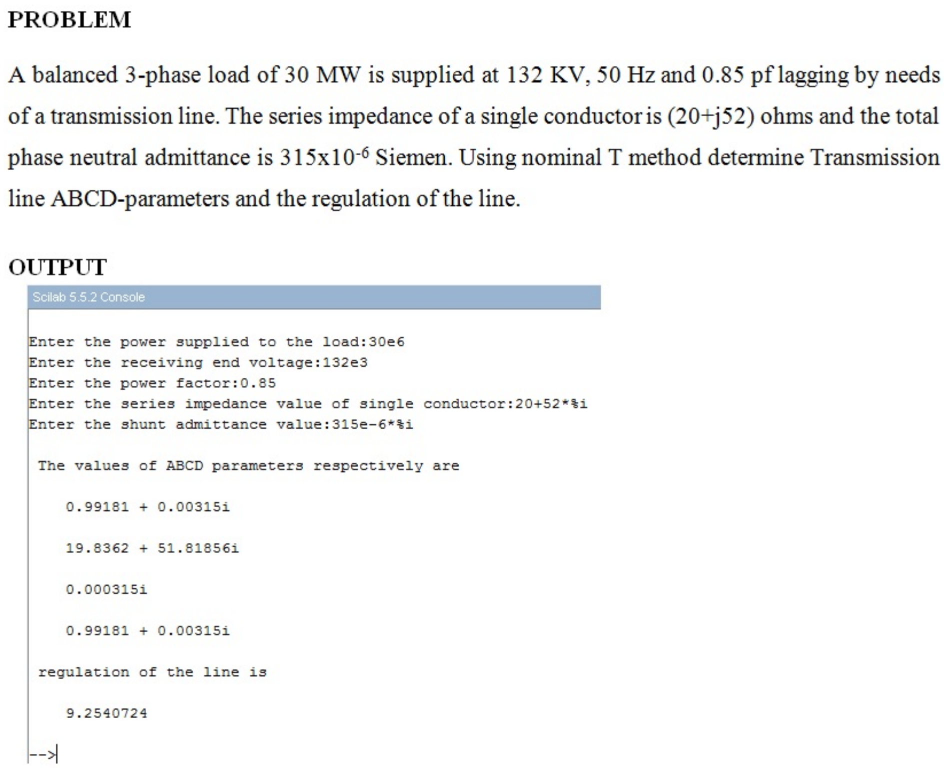
Experiment: 2

Modelling of Transmission Lines

Scilab code Solution 2.1 Nominal T method

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

```
c=y;
d=a;
disp(d,c,b,a,'The values of ABCD parameters
    respectively are')
vrph=vr/sqrt(3);       //receiving end voltage per
    phase
ir=pl/(sqrt(3)*vr*pf);        //receiving end current
irv=ir*(pf-%i*spf);           // receiving end
    current in vector form
vsph=(a*vrph+b*irv);          //sending end voltage
    per phase
vsh=abs(vsph);                // magnitude of sending
     end voltage per phase
reg=((abs(vsh/a)-abs(vrph))/vrph)*100;       //
    calculation of percentage regulation
disp(reg,'regulation of the line is')


//SAMPLE INPUT:
//Enter the power supplied to the load:30e6
//Enter the receiving end voltage:132e3
//Enter the power factor:0.85
//Enter the series impedance value of single
    conductor:20+52*%i
//Enter the shunt admittance value:315e-6*%i


 //OUTPUT:
 //The values of ABCD parameters respectively are

   //  0.99181 + 0.00315i

    //19.8362 + 51.81856i

    //0.000315i

//      0.99181 + 0.00315i

```


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

```
Scilab 5.5.2 Console

Enter the power supplied to the load:30e6
Enter the receiving end voltage:132e3
Enter the power factor:0.85
Enter the series impedance value of single conductor:20+52*%i
Enter the shunt admittance value:315e-6*%i

 The values of ABCD parameters respectively are

    0.99181 + 0.00315i

    19.8362 + 51.81856i

    0.000315i

    0.99181 + 0.00315i

 regulation of the line is

    9.2540724

-->
```

Figure 2.1: Nominal T method

```
//regulation of the line is

   // 9.2540724
```

Scilab code Solution 2.2 Nominal pi Method

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

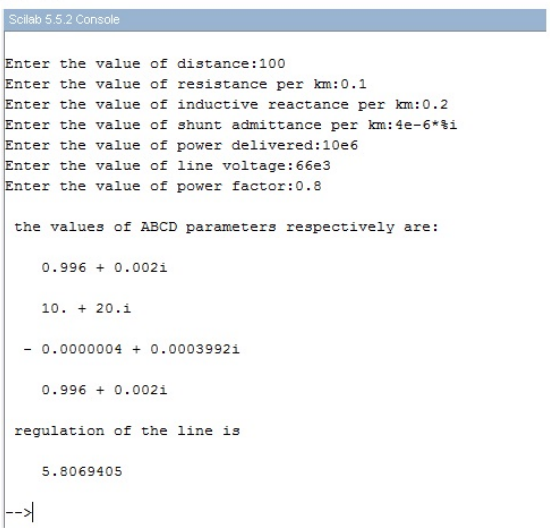
```
result file//
//Scilab Version 5.5.2 ; OS:Windows
clc;
clear;
d=input('Enter the value of distance:');
rkm=input('Enter the value of resistance per km:');
xlkm=input('Enter the value of inductive reactance per km:');
yshkm=input('Enter the value of shunt admittance per km:');
pl=input('Enter the value of power delivered:');
vl=input('Enter the value of line voltage:');
pf=input('Enter the value of power factor:');
vr=vl/sqrt(3);          //phase voltage
r=rkm*d;                //total resistance of the transmission line
xl=xlkm*d;              //total inductive reactance of the transmission line
ysh=yshkm*d;            //total shunt admittance of the transmission line
zs=r+(xl*%i);           //total impedance
a=1+(ysh*zs)/2;           //calculation of transmission line parameters
b=zs;
c=ysh*(1+(ysh*zs)/4);
d=a;
disp(d,c,b,a,'the values of ABCD parameters respectively are:')
ilo=pl/(sqrt(3)*vl*pf);
sp=sin(acos(pf));
ir=ilo*(pf-%i*sp);
icl=(vr*ysh)/2;
il=ir+icl;
vs=(vr+(il*(r+(%i*xl))));  //sending end voltage
reg=((abs(vs)/abs(a)-abs(vr))/abs(vr))*100;   //calculation of percentage regulation
disp(reg,'regulation of the line is');
```

```
//SAMPLE INPUT:
//Enter the value of distance:100
//Enter the value of resistance per km:0.1
//Enter the value of inductive reactance per km:0.2
//Enter the value of shunt admittance per km:4e-6*%i
//Enter the value of power delivered:10e6
//Enter the value of line voltage:66e3
//Enter the value of power factor:0.8

 //OUTPUT:
 //the values of ABCD parameters respectively are:

   // 0.996 + 0.002i

    //10. + 20.i

//  - 0.0000004 + 0.0003992i

  //  0.996 + 0.002i

 //regulation of the line is

   // 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



```
Scilab 5.5.2 Console

Enter the value of distance:100
Enter the value of resistance per km:0.1
Enter the value of inductive reactance per km:0.2
Enter the value of shunt admittance per km:4e-6*%i
Enter the value of power delivered:10e6
Enter the value of line voltage:66e3
Enter the value of power factor:0.8

 the values of ABCD parameters respectively are:

    0.996 + 0.002i

    10. + 20.i

  - 0.0000004 + 0.0003992i

    0.996 + 0.002i

 regulation of the line is

    5.8069405

-->
```

Figure 2.2: Nominal pi Method

Experiment: 3

Formation of Bus Admittance matrix

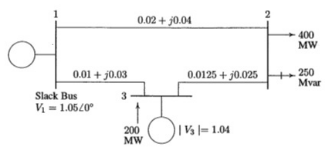
Scilab code Solution 3.1 Bus Admittance Matrix

```

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

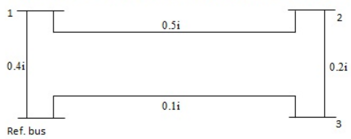
```

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

```
for i=1:length(sb)
    m=sb(i);
    n=eb(i);
    mn=min(m,n);
    nm=max(m,n);
    ncheck=length(find(check==nm));    //Variable used for checking whether bus nm is already existing
    mcheck=length(find(check==mn));    //Variable used for checking whether bus mn is already existing
    [rows columns]=size(zbus);
//Condition for connection of line between reference bus and new bus
    if mn==0 & ncheck==0
        zbus=[zbus zeros(rows,1);zeros(1,rows) z(i)];
        check=[check nm];
//Condition for connection of line between existing bus and new bus
         else if mcheck>0 & ncheck==0
            zbus=[zbus zbus(:,mn);zbus(mn,:) zbus(mn,mn)+z(i)];
            check=[check nm];
//Condition for connection of line between reference bus and existing bus
            elseif mn==0 & ncheck>0
            zbus=[zbus zbus(:,nm);zbus(nm,:) zbus(nm,nm)+z(i)];
            //Modifying Z bus size using Kron's reduction tehnique
            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
//Condition for connection of line between two existing buses
        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)];
            //Modifying Z bus size using Kron's reduction tehnique
            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
            zbusn;
        end
    end
end
disp(zbus,'The bus impedance matrix is:');


//SAMPLE INPUT:

//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]

 //OUTPUT:
 //The bus impedance matrix is:

//    0.2916667i    0.2083333i    0.1666667i
//    0.2083333i    0.2916667i    0.2333333i
//    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

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

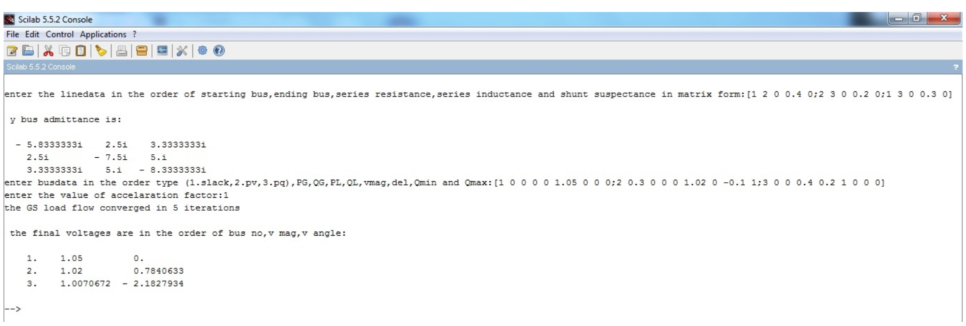
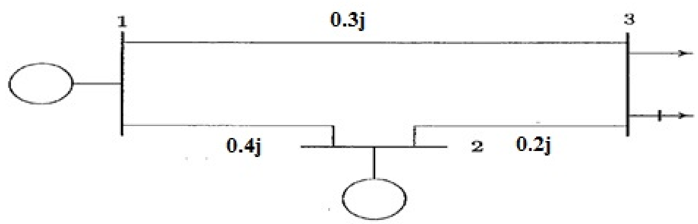


Figure 5.1: Gauss Seidal Load Flow

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

```
qmin=busdata(:,8) // minmum limit of Q for all the buses is stored in the variable qmin//
qmax=busdata(:,9)  // maximum limit of Q for all the buses is stored in the variable qmax//
p=busdata(:,2)-busdata(:,4) // real power of all the buses are calculated and is stored in the variable p //
q=busdata(:,3)-busdata(:,5)  // reactive power of all the buss are calculated and is stored in the variable q //
v=busdata(:,6).*(cosd(busdata(:,7))+%i*sind(busdata(:,7)));
alpha=input('enter the value of accelaration factor: ');
iter=1;
err=1;
vn(1)=v(1);
vold=v(1);
while abs(err)>5*10^(-5) //  starting of calculation part of bus voltage for first iteration //
    for i=2:nb
        sumyv=0;
        for j=1:nb
            sumyv=sumyv+y(i,j)*v(j);
        end
        if typ(i)==2
            q(i)=-imag(conj(v(i)*sumyv));
            if q(i)<qmin(i) |q(n)>qmax(i)
                vn(i)=(1/y(i,i))*(((p(i)-%i*q(i))/(conj(v(i))))-(sumyv-y(i,i)*v(i)));
                vold(i)=v(i);
                v(i)=vn(i);
                typ(i)=3
            if q(i)<qmin(i)
                q(i)=qmin(i);
            else
                q(i)=qmax(i);
```

```
                end
            else
                vn(i)=(1/y(i,i))*(((p(i)-%i*q(i))/(conj(v(i))))-(sumyv-y(i,i)*v(i)));
                ang=atan(imag(vn(i)),real(vn(i)));
                vn(i)=abs(v(i))*(cos(ang)+%i*sin(ang));
                vold(i)=v(i);
                v(i)=vn(i);
            end
            elseif typ(i)==3
                vn(i)=(1/y(i,i))*(((p(i)-%i*q(i))/(conj(v(i))))-(sumyv-y(i,i)*v(i)));
                vold(i)=v(i);
                v(i)=vn(i);
            end
            end
err=max(abs(abs(v)-abs(vold)));

iter=iter+1;
for i=2:nb
    if err>5*10^(-6) &typ(i)==3
    v(i)=vold(i)+alpha*(v(i)-vold(i));
    end
end
end
printf('the GS load flow converged in %d iterations \n',iter-1);
nn=1:nb;
res=[nn' abs(v) (atan(imag(v),real(v)))*(180/%pi)]
disp(res,'the final voltages are in the order of bus no,v mag,v angle:');

//SAMPLE INPUT and OUTPUT
//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]

```

```
 //y bus admittance is:

//  - 5.8333333i    2.5i     3.3333333i
//    2.5i         - 7.5i     5.i
//    3.3333333i    5.i    - 8.3333333i
//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]
//enter the value of accelaration factor:1
//the GS load flow converged in 5 iterations

 //the final voltages are in the order of bus no,v
    mag,v angle:

//    1.    1.05         0.
//    2.    1.02         0.7840633
//    3.    1.0070672  - 2.1827934
```

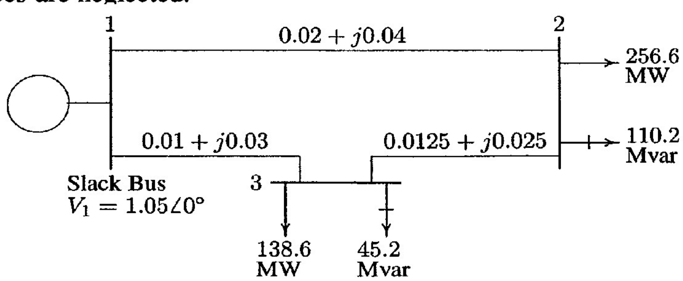
Experiment: 6

Load flow solution using Newton-Raphson method

Scilab code Solution 6.1 Newton Raphson load Flow

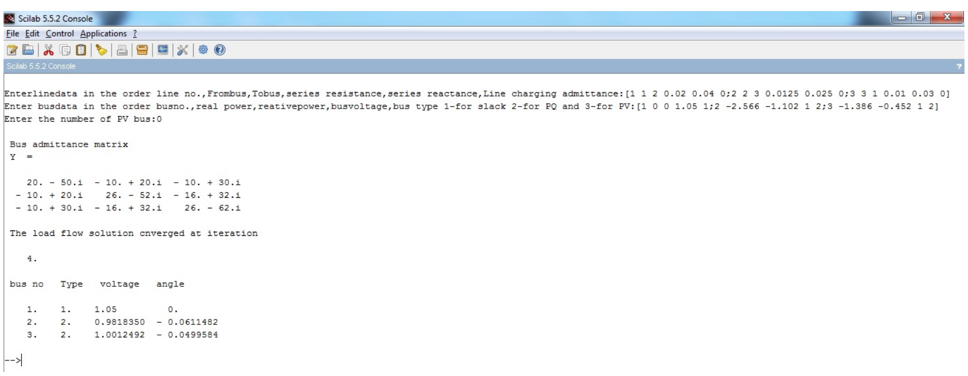
```

//Program to find out load flow solution using
    Newton Raphson method//
//This program requires user input. A sample problem
     with user input and output is available in the
    result file//
//Example problem is available in the file "
    NRQuestionFile.jpg" and user input and output is
    available in the file "NRResultFile"
//Scilab Version 5.5.2 ; OS:Windows

clear;
clc;
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



```
Scilab 5.5.2 Console
File Edit Control Applications ?
Scilab 5.5.2 Console

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]
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]
Enter the number of PV bus:0

 Bus admittance matrix
 Y  =

   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

 The load flow solution cnverged at iteration

   4.

 bus no   Type   voltage   angle

   1.    1.    1.05          0.
   2.    2.    0.9818350  - 0.0611482
   3.    2.    1.0012492  - 0.0499584

-->
```

Figure 6.2: Newton Raphson load Flow

```
    Line charging admittance:');
Busdata=input('Enter busdata in the order busno.,
    real power,reativepower,busvoltage,bus type 1-for
     slack 2-for PQ and 3-for PV:');
npv=input('Enter the number of PV bus:');

//Determination of bus admittance matrix//
nb=max(Busdata(:,1));
nl=max(linedata(:,1));
Psp=Busdata(:,2);
Qsp=Busdata(:,3);
vsp=Busdata(:,4);
rem=Busdata(:,5);
Y=zeros(nb,nb);
sb=linedata(:,2);
eb=linedata(:,3);
z=linedata(:,4)+linedata(:,5)*%i;
ly=linedata(:,6);
for i=1:nl
    m=sb(i)
    n=eb(i);
    Y(m,m)=Y(m,m)+1/z(i)+ly(i)/2;
    Y(n,n)=Y(n,n)+1/z(i)+ly(i)/2;
    Y(m,n)=-1/z(i);
    Y(n,m)=Y(m,n);
   end
disp('Bus admittance matrix')
print(%io(2),Y)

//NR Load flow//
absY=abs(Y);
thetaY=atan(imag(Y),real(Y));
v=vsp';
iteration=1;
ang=zeros(1,nb);
mismatch=ones(2*nb-2-npv,1);
while max(abs(mismatch))>0.0001
    J1=zeros(nb-1,nb-1);
```

```
    J2=zeros(nb-1,nb-npv-1);
    J3=zeros(nb-npv-1,nb-1);
    J4=zeros(nb-npv-1,nb-npv-1);
    P=zeros(nb,1);
    Q=P;
    del_P=Q;
    del_Q=Q;
    del_del=zeros(nb-1,1);
    del_v=zeros(nb-1-npv,1);
    ang;
    mag=abs(v);
    for i=2:nb
        for j=1:nb
            P(i)=P(i)+mag(i)*mag(j)*absY(i,j)*cos(thetaY(i,j)-ang(i)+ang(j));
             if rem(i)~=3
                   Q(i)=Q(i)+mag(i)*mag(j)*absY(i,j)*sin(thetaY(i,j)-ang(i)+ang(j));
              end
          end
          end
Q=-1*Q;
del_P=Psp-P;
del_Q=Qsp-Q;
for i=2:nb
    for j=2:nb
        if j~=i
            J1(i-1,j-1)=-mag(i)*mag(j)*absY(i,j)*sin(thetaY(i,j)-ang(i)+ang(j));
            J2(i-1,j-1)=mag(i)*absY(i,j)*cos(thetaY(i,j)-ang(i)+ang(j));
            J3(i-1,j-1)=-mag(i)*mag(j)*absY(i,j)*cos(thetaY(i,j)-ang(i)+ang(j));
            J4(i-1,j-1)=-mag(i)*absY(i,j)*sin(thetaY(i,j)-ang(i)+ang(j));
        end
    end
end
```

```
for i=2:nb
    for j=1:nb
        if j~=i
            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));
            J2(i-1,i-1)=J2(i-1,i-1)+mag(j)*absY(i,j)*cos(thetaY(i,j)-ang(i)+ang(j));
            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));
            J4(i-1,i-1)=J4(i-1,i-1)+mag(j)*absY(i,j)*sin(thetaY(i,j)-ang(i)+ang(j));
        end
    end
    J2(i-1,i-1)=2*mag(i)*absY(i,i)*cos(thetaY(i,i))+J2(i-1,i-1);
    J4(i-1,i-1)=-2*mag(i)*absY(i,i)*sin(thetaY(i,i))-J4(i-1,i-1);
    end
J=[J1 J2;J3 J4]
lenJ=length(J1);
i=2;
j=1;
while j<=lenJ
    if rem(i)==2
        j=j+1;
    else
        J(:,length(J1)+j)=[];
        lenJ=lenJ-1;
    end
end
i=i+1;
lenJ=length(J1);
i=1;
j=2;
while i<=lenJ
    if rem(j)==2
```

```
            i=i+1;
        else
            J(length(J1)+i,:)=[];
            lenJ=lenJ-1;
            Q(i+1)=[]
            del_Q(i+1,:)=[]
        end
    //  j=j+1;
        end
P(1,:)=[]
Q(1,:)=[]
del_P(1,:)=[];
del_Q(1,:)=[];
mismatch=[del_P;del_Q];
del=J\mismatch;
del_del=del(1:nb-1);
del_v=del(nb:length(del));
ang=ang(2:nb)+del_del';
j=1;
for i=2:nb
    if rem(i)==2
        v(i)=v(i)+del_v(j);
        j=j+1;
    end
end
mag=abs(v);
ang=[0 ang];
nbr=1:nb;
iteration=iteration+1;
end
disp(iteration-1,'The load flow solution cnverged at
    iteration')
disp('bus no   Type   voltage   angle')
disp([nbr' rem mag' ang'])


//SAMPLE INPUT and OUTPUT:
//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]
//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]
//Enter the number of PV bus:0

// Bus admittance matrix
// Y =

//     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

// The load flow solution cnverged at iteration

//     4.

// bus no   Type   voltage    angle

//     1.     1.     1.05           0.
//     2.     2.     0.9818350  - 0.0611482
//     3.     2.     1.0012492  - 0.0499584
```

Experiment: 7

Symmetrical Fault Analysis

Scilab code Solution 7.1 Symmetrical Fault Analysis

```
//Program to find out fault current,post-fault
   voltages and line flow of a given network//
//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"//
//Scilab Version 5.5.2 ; OS:Windows
clc;
clear;
linedata=input('enter the line data values in the
   order of starting bus,nding bus,resistance and
   reactance:')
f=input('enter the bus at wich fault occurs:')
bv=input('enter the pre-fault bus voltage:')
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.

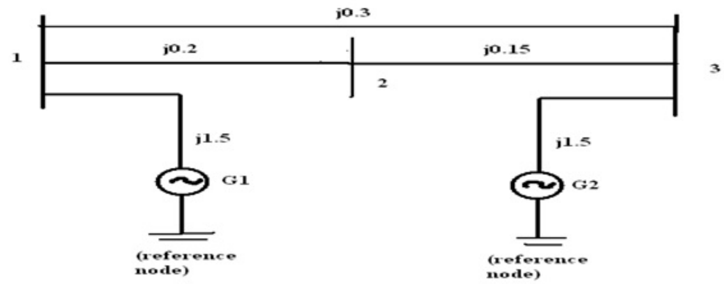
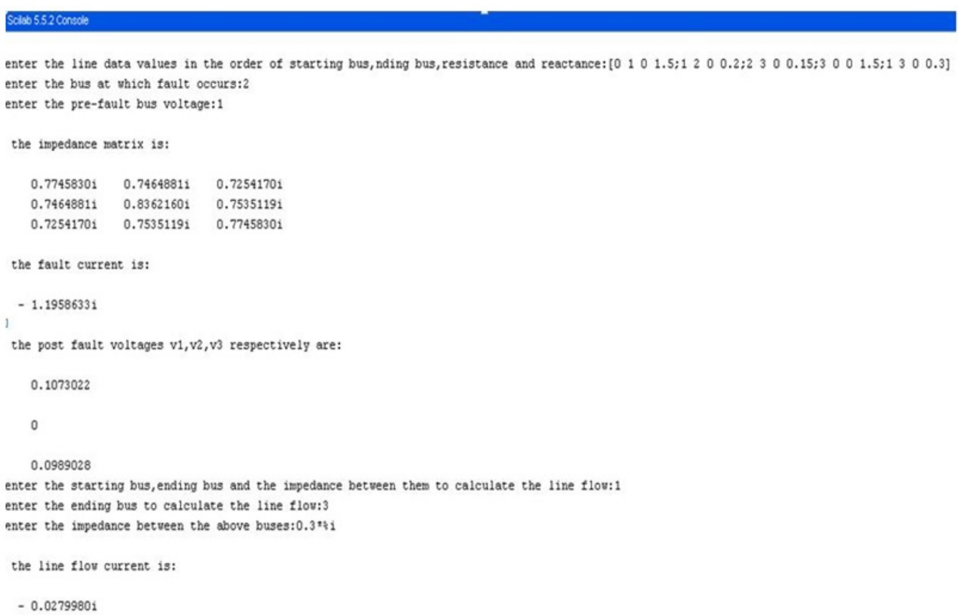


Figure 7.1: Symmetrical Fault Analysis



```
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)];
            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;
        end
    end
end   //ending of impedance bus matrix calculation part//
disp(zbus,'the impedance matrix is:');
ifa=bv/zbus(f,f)  //calculation of fault current//
disp(ifa,'the fault current is:')
disp('the post fault voltages v1,v2,v3 respectively are:');
for i=1:n
v(i)=bv-(ifa*zbus(i,f)); //calculation of postfault bus voltages//
disp(v(i));
end
a=input('enter the starting bus to calculate the line flow:');
b=input('enter the ending bus to calculate the line flow:');
zs=input('enter the impedance between the above buses:');
i13=(v(a)-v(b))/zs; //calculation oflineflows//
disp(i13,'the line flow current is:')

//SAMPLE INPUT and OUTPUT:
//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 wich 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 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
```

Experiment: 8

Unsymmetrical Fault Analysis

Scilab code Solution 8.1 Unsymmetrical Fault Analysis

```

//Program to find out unsymmetrical fault current//
//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"//
//Scilab Version 5.5.2 ; OS:Windows
clc ;
clear;
a=input('Enter the positve sequence,negative
   sequence and zero sequence of first generator in
   matrix form:')
PG1=a(:,1);//positive sequence of generator 1 is
   stored in the variable PG1
NG1=a(:,2);//negative sequence of generator 1 is
   stored in the variable NG1
ZG1=a(:,3);//negative sequence of generator 1 is
   stored in the variable ZG1
b=input('Enter the positve sequence,negative
```

```
sequence and zero sequence of first transformer in matrix form:')
PT1=b(:,1);//positive sequence of transformer 1 is stored in the variable PT1
NT1=b(:,2);//positive sequence of transformer 1 is stored in the variable NT1
ZT1=b(:,3);//positive sequence of transformer 1 is stored in the variable ZT1
c=input('Enter the positve sequence,negative sequence and zero sequence of first transmission line in matrix form:')
PTL=c(:,1);//positive sequence of transmission line 1 is stored in the variable PTL
NTL=c(:,2);//positive sequence of transmission line 1 is stored in the variable NTL
ZTL=c(:,3);//positive sequence of transmission line 1 is stored in the variable ZTL
d=input('Enter the positve sequence,negative sequence and zero sequence of second transformer in matrix form:')
PT2=d(:,1);//positive sequence of transformer is stored in the variable PT2
NT2=d(:,2);//positive sequence of transformer 1 is stored in the variable NT2
ZT2=d(:,3);//positive sequence of transformer 1 is stored in the variable ZT2
e=input('Enter the positive sequence,negative sequence and zero sequence of second generator in matrix form:')
PG2=e(:,1);//positive sequence of transformer 1 is stored in the variable PG2
NG2=e(:,2);//positive sequence of transformer 1 is stored in the variable NG2
ZG2=e(:,3);//positive sequence of transformer 1 is stored in the variable ZG2
MVAB=input('Enter the value of base MVA:');
KVB=input('Enter the value of base KV:');
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
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
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
Ib=(MVAB*(10^6))/((1.732*KVB*(10^3)))//calculating base current
disp(z0,z2,z1,'the values of positive(z1) negative(z2),zero(z0) sequence impedance  respectively are ');
disp('OPTION','LG FAULT=1','LL FAULT=2','LLG FAULT=3');
MENU=input('Enter the choice of fault:')
if MENU==1 //calculating Line to Ground fault
    If=(3*(1))/(z0+z1+z2)
    FAULTCURRENT=If*Ib;
    disp(FAULTCURRENT,'The fault current is :');
end
if MENU==2//Calculating Line to Line Fault
    If=((-1.732j)*(1))/(z1+z2)
    FAULTCURRENT=If*Ib;
disp(FAULTCURRENT,'The fault current is :');
end
if MENU==3//calculating Line-Line-Ground fault
    z=(z0*z2)/(z0+z2);
    Ia1=(1)/(z1+z);
    Ia0=((-1+(Ia1*z1))/z0);
    If=3*Ia0;
    FAULTCURRENT=If*Ib;
 disp(FAULTCURRENT,'The fault current is :');
end

//SAMPLE INPUT and OUTPUT

//Enter the positve sequence,negative sequence and
```

```
    zero sequence of first generator in matrix form
    :[0.32 0.26 0.09]
//Enter the positve sequence,negative sequence and
    zero sequence of first transformer in matrix form
    :[0.23 0.23 0.23]
//Enter the positve sequence,negative sequence and
    zero sequence of first transmission line in
    matrix form:[0.56 0.56 0.09]
//Enter the positve sequence,negative sequence and
    zero sequence of second transformer in matrix
    form:[0.16 0.16 0.16]
//Enter the positive sequence,negative sequence and
    zero sequence of second generator in matrix form
    :[0.38 0.24 0.15]
//Enter the value of base MVA:100
//Enter the value of base KV:110

// the values of positive(z1) negative(z2),zero(z0)
    sequence impedance  respectivel
//       y are

//     0.3666667i

//     0.3244138i

//     0.1777778i

// LLG FAULT=3

// LL FAULT=2

// LG FAULT=1

// OPTION
//Enter the choice of fault:3

// The fault current is :

```

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

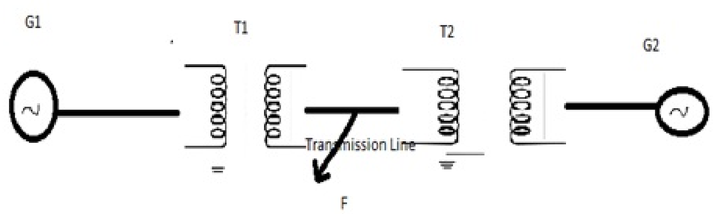


Figure 8.1: Unsymmetrical Fault Analysis

```
85  //      2112.5397 i
```

```
Scilab 5.5.2 Console

Enter the positve sequence,negative sequence and zero sequence of first generator in matrix form:[0.32 0.26 0.09]
Enter the positve sequence,negative sequence and zero sequence of first transformer in matrix form:[0.23 0.23 0.23]
Enter the positve sequence,negative sequence and zero sequence of first transmission line in matrix form:[0.56 0.56 0.09]
Enter the positve sequence,negative sequence and zero sequence of second transformer in matrix form:[0.16 0.16 0.16]
Enter the positve sequence,negative sequence and zero sequence of second generator in matrix form:[0.38 0.24 0.15]
Enter the value of base MVA:100
Enter the value of base KV:110

 the values of positive(z1) negative(z2),zero(z0) sequence impedance  respectively are

    0.3666667i

    0.3244138i

    0.1777778i

 LLG FAULT=3

 LL FAULT=2

 LG FAULT=1

 OPTION
Enter the choice of fault:3

 The fault current is :

    2112.5397i
```

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

```
//Program to find out transient stability analysis
    of single machine - infinite bus system //
//An example problem and outputs are available in
    files 'result1' and 'result2'
//Scilab Version 5.5.2 ; OS:Windows

clc;
clear;

xd=0.3;

```

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

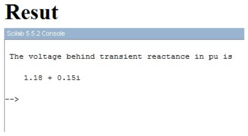


Figure 9.1: SMIB Stability Analysis

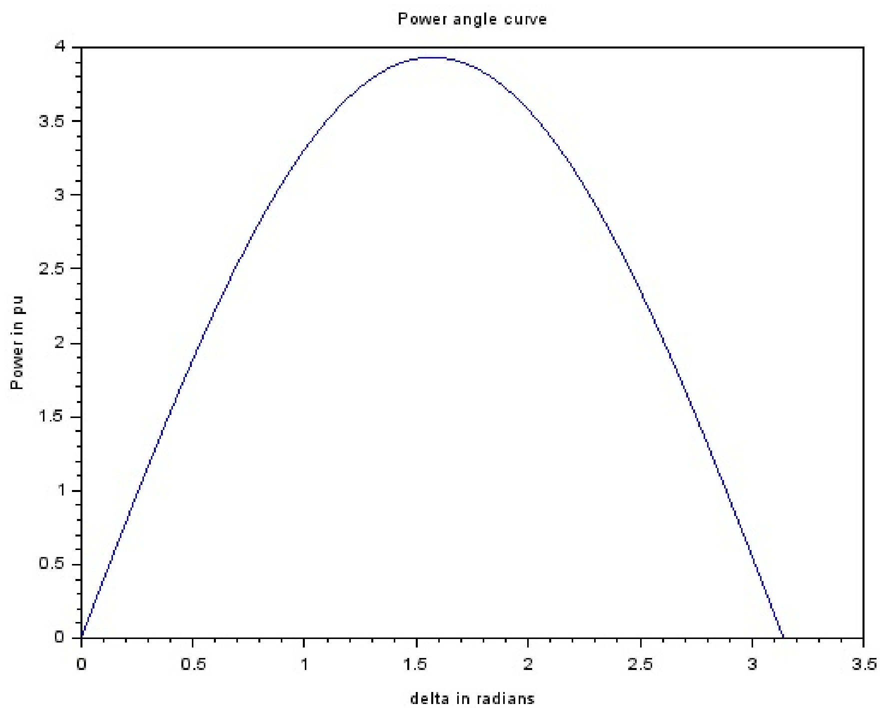


Figure 9.2: SMIB Stability Analysis

```
theta=acos(0.8);                                  //Power factor angle

S=(0.5/0.8)*cos(theta)+%i*sin(theta);             //Apparant Power
V=1;                                              //Prefault vltage is assumed to be 1 pu
Ia=(conj(S)/V);                                    //Pefault steady state current

E=V+(%i*xd)*(Ia);                                 //Voltage behind transient reactance

disp(E,'The voltage behind transient reactance in pu is ')

//To find the power angle curve

delta=0:0.001:%pi;
P=((E*V)/xd)*(sin(delta));

plot(delta,P)
xlabel('delta in radians')
ylabel('Power in pu')
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

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

```
x1=input('enter the prefault reactance in p.u:');
x2=input('enter the post fault reactance value:');
x3=input('enter during the fault reactance value:');
delt=input('enter the time interval in seconds:');
H=input('enter the inertia constant:');
pe1=ld/bv;

pe2=0;
delnot=asin((pe1*x1)/(e*v));

omeganot=2*3.14*f;


ddel=omeganot-(2*3.14*f);

ddelomega=(((3.14*f)*(pe1-pe2))/H);

//end of first step at t=0.05sec
del1=(delnot+(ddel*delt));//predicted values

delomega1=ddel+(ddelomega*delt);

//derivation at the end of t=0.05sec
ddel1=ddel+(ddelomega*delt);

ddelomega=(((3.14*f)*(pe1-pe2))/H);

delc1=delnot+((delt/2)*(ddel+ddel1));

delomegac1=ddel+((delt/2)*(ddelomega+ddelomega));

ddelc1=ddel+((delt/2)*(ddelomega+ddelomega));

ddelomegac=(((3.14*f)*(pe1-pe2))/H);

delp2=delc1+ddelc1*delt;

delomegap2=(delomegac1+(ddelomega*delt));
```

```
ddelomegap=(((3.14*f)*(pe1-pe2))/H);


delc2=delc1+(delt/2*(ddelc1+delomegap2));

delomegac=(delomegac1+(ddelomega*delt));

ddelc2=(delomegac1+(ddelomega*delt));

ddelomegac2=(((3.14*f)*(pe1-pe2))/H);

delp3=delc2+delomegac*delt;

delomega3=delomegac+ddelomegac2*delt;

//derivation at the end of t=0.15sec
disp('The final values at the end of t=0.15 sec are
   displayed below')
ddelp3=delomegac+ddelomegac2*delt;
disp(ddelp3,'ddelp3=');
ddelomega3=(((3.14*f)*(pe1-pe2))/H);
disp(ddelomega3,'ddelomega3=');
disp('corrected values');
delc3=delc2+((delt/2)*(delomegac+delomega3));
disp(delc3,'delc3=');
delomegac3=delomegac+((delt/2)*(ddelomega3+
   ddelomega3));
disp(delomegac3,'delomegac3=');

//SAMPLE INPUT:
//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


 //OUTPUT
// 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
```

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

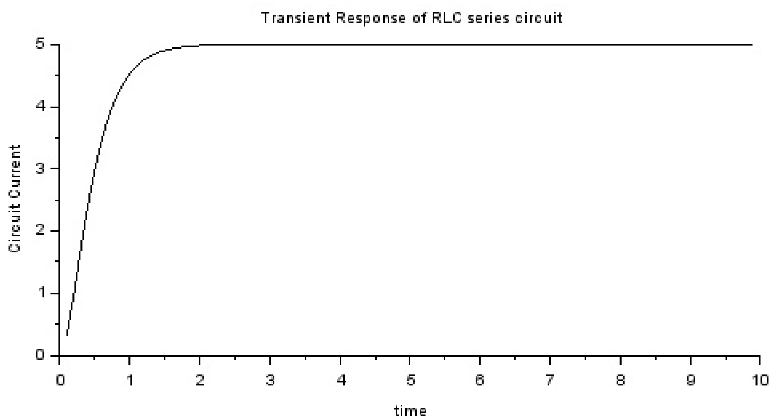


Figure 11.1: Transient in RLC series circuit with DC source

Experiment: 12

Load frequency dynamics of single Area Power Systems

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

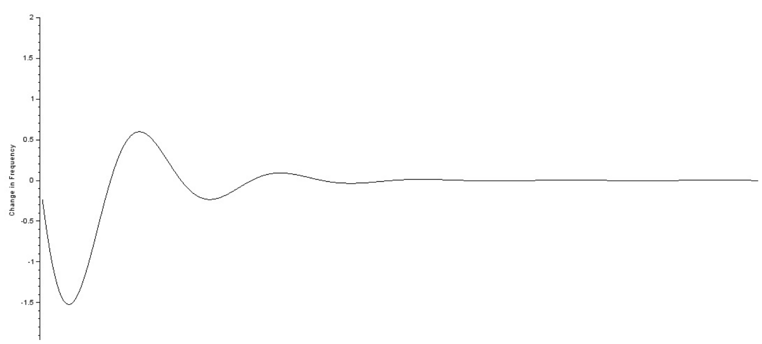


Figure 12.1: Single Area Control

Experiment: 13

Load frequency dynamics of two Area Power Systems

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

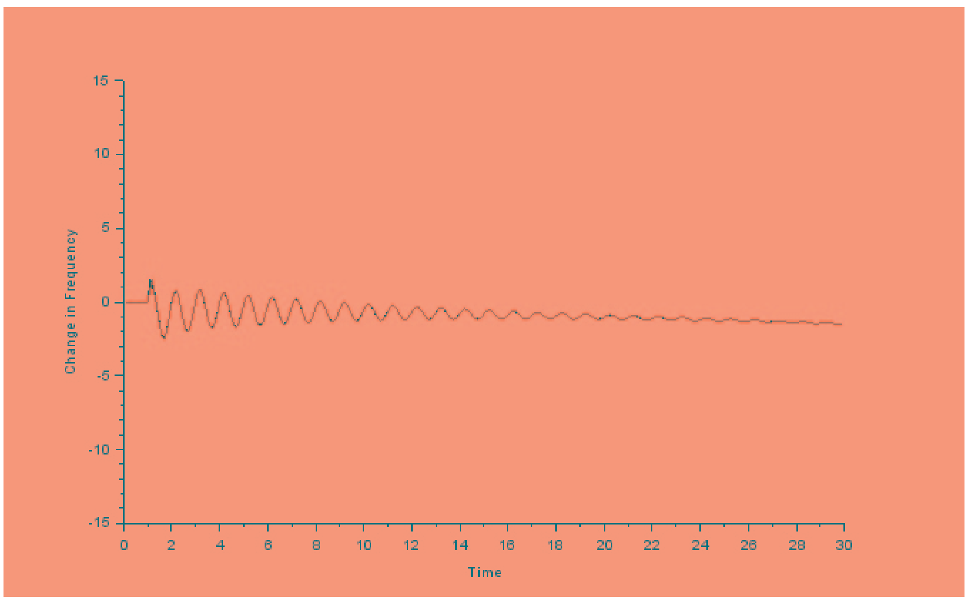


Figure 13.1: Two Area Control

Experiment: 14

Economic dispatch in power systems neglecting losses

Scilab code Solution 14.1 Economic Load Dispatch Excluding Losses

```
//Program to find out Economic load dispatch neglecting losses//
//This program requires user input. A sample problem with user input and output is available in the result file//
//Question and result of example problem is available in file "EDwithoutLoss.jpg"
//Scilab Version 5.5.2 ; OS:Windows

clear;
clc;
n=input('Enter no. of units :');
F=input('Enter the cost coefficient in matrix form : ');
constraint=input('Enter min and max values of P for all units:');
pd=input('Enter total demand:');
```

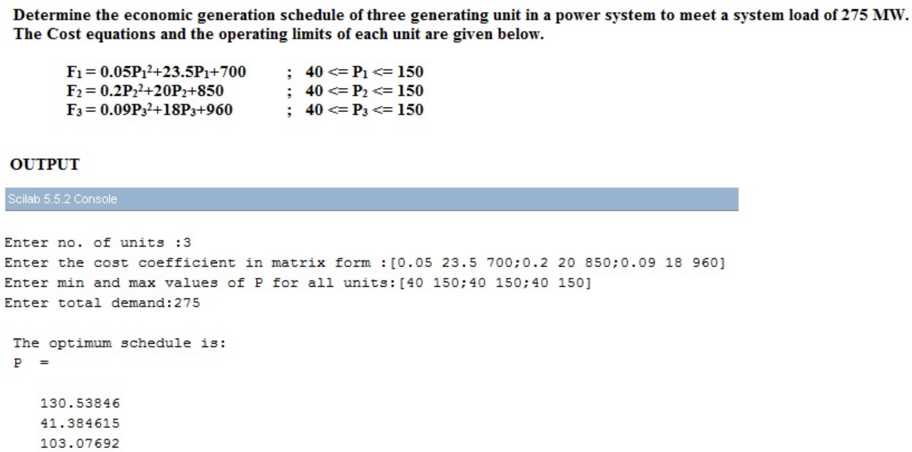


Figure 14.1: Economic Load Dispatch Excluding Losses

```
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);
```

```
                else
                    P(i)=Pmax(i);
                end
                pd=pd-P(i);
                chk(i)=i;
                sch=sch+1;
            end
        end
        if sch==0
            rem=0;
        else
            rem=1;
        end
    end
end
disp('The optimum schedule is:')
print(%io(2),P)

//SAMPLE INPUT
//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

 //OUTPUT
// The optimum schedule is:
// P =

//     130.53846
//     41.384615
//     103.07692
```

Experiment: 15

Economic dispatch in power systems Including losses

Scilab code Solution 15.1 Economic Load Dispatch Including Losses

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

ECONOMIC DISPATCH PROBLEM INCLUDING LOSSES

The transmission loss coefficients are given by
B=

$$\begin{bmatrix} 0.01 & -0.0003 & -0.0002 \\ -0.0003 & 0.0025 & -0.0005 \\ -0.0002 & -0.0005 & 0.0031 \end{bmatrix}$$

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
```