# Scilab Textbook Companion for Power System: Analysis & Design by Thomas Overbye, J. Duncan Glover, Mulkutla .S. Sarma[1]

Created by
Jain B Marshel
ME
Electrical Engineering
St.Xavier's Catholic College of Engineering
College Teacher
None
Cross-Checked by
None

July 31, 2019

---

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

# Book Description

**Title:** Power System: Analysis & Design

**Author:** Thomas Overbye, J. Duncan Glover, Mulkutla .S. Sarma

**Publisher:** Cengage Learning India Private Limited

**Edition:** 5

**Year:** 2011

**ISBN:** 978-8131516355

Scilab numbering policy used in this document and the relation to the above book.

**Exa** Example (Solved example)

**Eqn** Equation (Particular equation of the above book)

**AP** Appendix to Example(Scilab Code that is an Appednix to a particular Example of the above book)

For example, Exa 3.51 means solved example 3.51 of this book. Sec 2.3 means a scilab code whose theory is explained in Section 2.3 of the book.

# Contents

# List of Scilab Codes

# List of Figures

# Chapter 2

# FUNDAMENTALS

**Scilab code Exa 2.1** `Instantaneous real and reactive power and power factor`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 2 ; Example 2.1
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;
Vmax=141.4;                           //Peak source voltage in Volts
R=10;                                 //Load resistance in Ohms
Xl=3.77;                              //Inductive reactance in Ohms
Vrms=Vmax/sqrt(2);                    //RMS value of source voltage in Volts
Ir=Vrms/(R);                          //Current through the resistor in Amperes
Il=Vrms/(%i*Xl);                      //Current through the inductor in Amperes
Iload=Ir+Il;                          //Current through
```

```
the load in Amperes
wt=0:0.1:2*%pi;
v=Vmax*cos(wt);                       //Instantaneous voltage in Volts
ir=Vmax*cos(wt)/R;                    //Instantaneous current through the resistor in Amperes
il=Vmax*cos(wt+90*%pi/180);           //Instantaneous current through the inductor in Amperes
Pr=Vrms*Ir*(1+cos(2*wt));             //Instantaneous Power absorbed by Resistor in Watts
Pl=Vrms*abs(Il)*sin(2*wt);            //Instantaneous Power absorbed by Inductor in Watts
del=0;
bet=atan(imag(Iload),real(Iload));
P=Vrms*abs(Iload)*cos(del-bet);       //Real power absorbed by the load in Watts
Q=Vrms*abs(Iload)*sin(del-bet);       //Reactive power absorbed by the load in VAR
pf=cos(del-bet);                      //Power factor
clf;                                  //To clear figures from previous programs
subplot(231);
plot(wt,v);
xtitle('Input Voltage','Angular displacement','Voltage(Volts)');
subplot(232);
plot(wt,ir);
xtitle('Current through resistor','Angular displacement','Current(Amp.)');
subplot(233);
plot(wt,Pr);
xtitle('Power dissipated in resistor','Angular displacement','Power(Watts)');
subplot(236);
xtitle('Power throgh the inductor','Angular displacement','Power(VAR)');
plot(wt,Pl);
subplot(234);
```

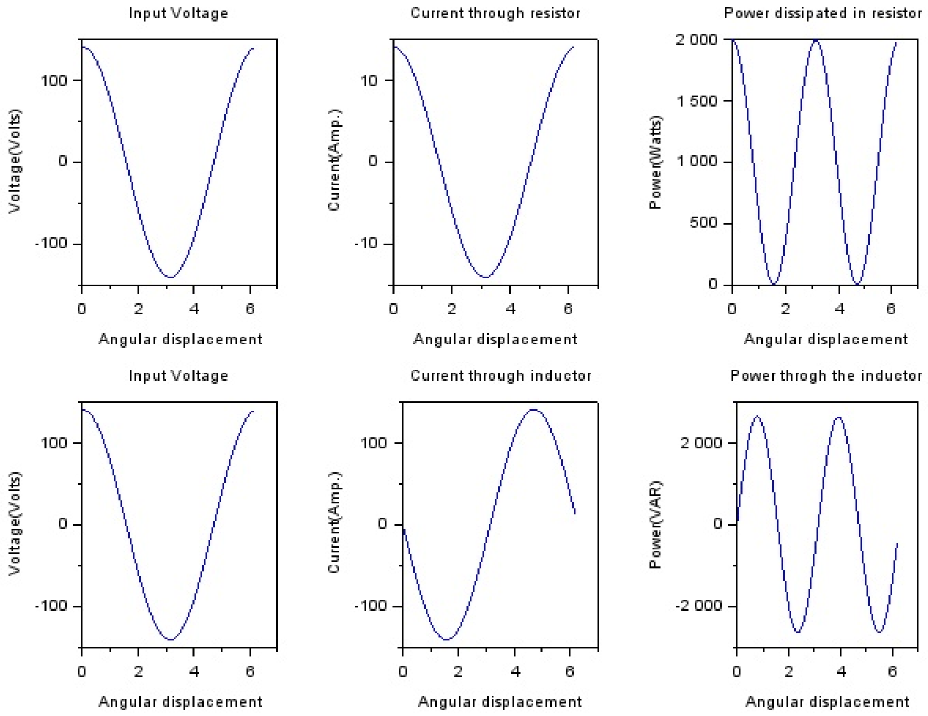
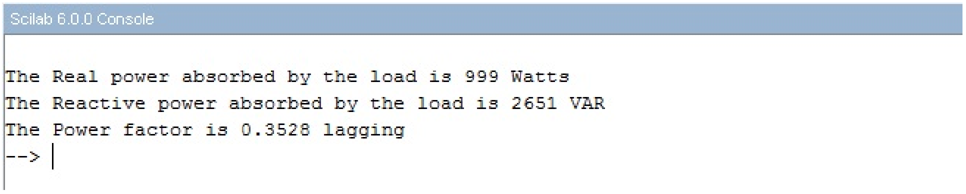


Figure 2.1: Instantaneous real and reactive power and power factor

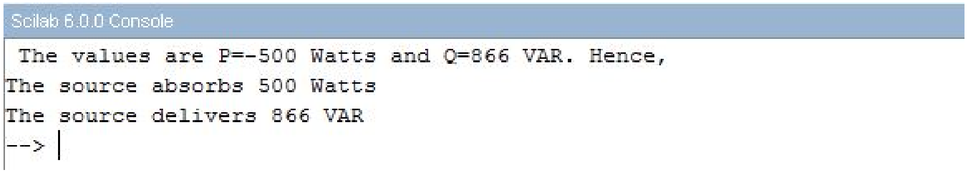
```
plot(wt,v);
xtitle('Input Voltage','Angular displacement','Voltage(Volts)');
subplot(235);
plot(wt,il);
xtitle('Current through inductor','Angular displacement','Current(Amp.)');
printf('\nThe Real power absorbed by the load is %d Watts\n',P);
printf('The Reactive power absorbed by the load is %d VAR\n',Q);
printf('The Power factor is %.4f lagging',pf);
```

---



```
The Real power absorbed by the load is 999 Watts
The Reactive power absorbed by the load is 2651 VAR
The Power factor is 0.3528 lagging
-->
```

Figure 2.2: Instantaneous real and reactive power and power factor



```
 The values are P=-500 Watts and Q=866 VAR. Hence,
The source absorbs 500 Watts
The source delivers 866 VAR
-->
```

Figure 2.3: Real and reactive power delivered or absorbed

**Scilab code Exa 2.2** `Real and reactive power delivered or absorbed`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 2 ; Example 2.2
//Scilab Version - 6.0.0 ; OS - Windows


clc;
clear;
V=100*exp(%i*130*%pi/180);                    //Source Voltage in Volts
I=10*exp(%i*10*%pi/180);                      //Source current in Amperes
```

```
The capacitor delivers 42.131589 kVAR
-->
```

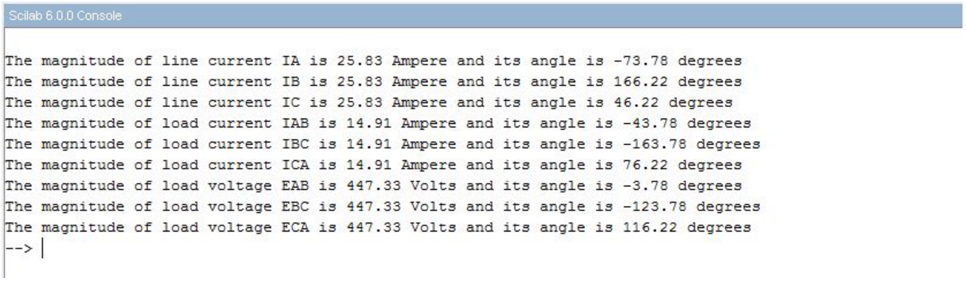
Figure 2.4: Power factor correction

```
S=V*conj(I);    //Apparent power in VA
P=real(S);    //Real power in Watts
Q=imag(S);    //Reactive power in VAR
printf('The values are P=%d Watts and Q=%d VAR. Hence,',P,Q);
if P<0 then
    P=-P
    printf('\nThe source absorbs %d Watts',P);
else
    printf('\nThe source delivers %d Watts',P);
end
if Q<0 then
    Q=-Q;
    printf('\nThe source absorbs %d VAR',Q);
else
    printf('\nThe source delivers %d VAR',Q);
end
```

---

**Scilab code Exa 2.3** `Power factor correction`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 2 ; Example 2.3
```



```
Scilab 6.0.0 Console

The magnitude of line current IA is 25.83 Ampere and its angle is -73.78 degrees
The magnitude of line current IB is 25.83 Ampere and its angle is 166.22 degrees
The magnitude of line current IC is 25.83 Ampere and its angle is 46.22 degrees
The magnitude of load current IAB is 14.91 Ampere and its angle is -43.78 degrees
The magnitude of load current IBC is 14.91 Ampere and its angle is -163.78 degrees
The magnitude of load current ICA is 14.91 Ampere and its angle is 76.22 degrees
The magnitude of load voltage EAB is 447.33 Volts and its angle is -3.78 degrees
The magnitude of load voltage EBC is 447.33 Volts and its angle is -123.78 degrees
The magnitude of load voltage ECA is 447.33 Volts and its angle is 116.22 degrees
-->|
```

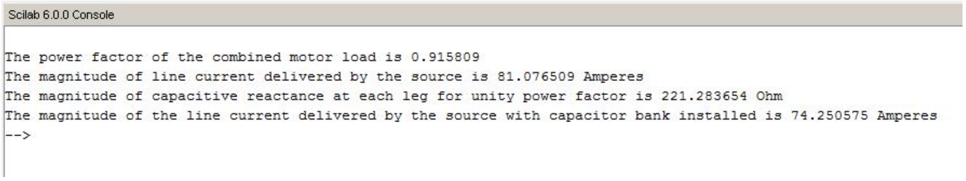
Figure 2.5: Balanced Delta and Wye loads

```
//Scilab Version - 6.0.0 ; OS - Windows


clear;
clc;
P=100              //Real power in kW
pf=0.8;            //Power factor
pfc=0.95           //Corrected power factor with capacitor
Ol=acos(pf);       //Power factor angle without capacitor
Oc=acos(pfc);      //Power factor angle with capacitor
Ql=P*tan(Ol);      //Reactive power delivered by the source without capacitor in kVAR
Sl=P/cos(Ol);      //Apparent power delivered by the source without capacitor in kVA
Qs=P*tan(Oc);      //Reactive power delivered by the source with capacitor in kVAR
Ss=P/cos(Oc);      //Apparent power delivered by the source with capacitor in kVA
Qc=Ql-Qs;          //Reactive power delivered by the capacitor in kVAR
printf('\nThe capacitor delivers %f kVAR',Qc);
```

**Scilab code Exa 2.4** Balanced Delta and Wye loads

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 2 ; Example 2.4
//Scilab Version - 6.0.0 ; OS - Windows


clc;
clear;
Eab=480*(cos(0*%pi/180)+%i*sin(0*%pi/180));                         //Line Voltage of the source in Volts
Zdel=30*(cos(40*%pi/180)+%i*sin(40*%pi/180));                       //Impedance of the delta load in Ohm
Zlineperphase=1*(cos(85*%pi/180)+%i*sin(85*%pi/180));               //Line Impedance in Ohm
Zstar=Zdel/3;                                                        //Impedance of delta load converted to star load in Ohm
[r theta]=polar(Eab);
Ebc=r*(cos(theta-120*%pi/180)+%i*sin(theta-120*%pi/180));
Eca=r*(cos(theta+120*%pi/180)+%i*sin(theta+120*%pi/180));
Ean=r*(cos(theta-30*%pi/180)+%i*sin(theta-30*%pi/180))/sqrt(3); //Phase voltage of the source in Volts
[r theta]=polar(Ean);
Ebn=r*(cos(theta-120*%pi/180)+%i*sin(theta-120*%pi/180));
```

```
Ecn=r*(cos(theta+120*%pi/180)+%i*sin(theta+120*%pi/180));
Ia=Ean/(Zlineperphase+Zstar);                         //Line current in Amperes
Ib=Ebn/(Zlineperphase+Zstar);
Ic=Ecn/(Zlineperphase+Zstar);
[r theta]=polar(Ia);
Iab=r*(cos(theta+30*%pi/180)+%i*sin(theta+30*%pi/180))/sqrt(3); //Phase current in Amperes
[r theta]=polar(Ib);
Ibc=r*(cos(theta+30*%pi/180)+%i*sin(theta+30*%pi/180))/sqrt(3);
[r theta]=polar(Ic);
Ica=r*(cos(theta+30*%pi/180)+%i*sin(theta+30*%pi/180))/sqrt(3);
EAB=Zdel*Iab;                                         //Line voltage across the load in Volts
EBC=Zdel*Ibc;
ECA=Zdel*Ica;
printf('\nThe magnitude of line current IA is %.2f Ampere and its angle is %.2f degrees',abs(Ia),atand(imag(Ia),real(Ia)));
printf('\nThe magnitude of line current IB is %.2f Ampere and its angle is %.2f degrees',abs(Ib),atand(imag(Ib),real(Ib)));
printf('\nThe magnitude of line current IC is %.2f Ampere and its angle is %.2f degrees',abs(Ic),atand(imag(Ic),real(Ic)));
printf('\nThe magnitude of load current IAB is %.2f Ampere and its angle is %.2f degrees',abs(Iab),atand(imag(Iab),real(Iab)));
printf('\nThe magnitude of load current IBC is %.2f Ampere and its angle is %.2f degrees',abs(Ibc),atand(imag(Ibc),real(Ibc)));
printf('\nThe magnitude of load current ICA is %.2f Ampere and its angle is %.2f degrees',abs(Ica),
```



```
The power factor of the combined motor load is 0.915809
The magnitude of line current delivered by the source is 81.076509 Amperes
The magnitude of capacitive reactance at each leg for unity power factor is 221.283654 Ohm
The magnitude of the line current delivered by the source with capacitor bank installed is 74.250575 Amperes
-->
```

Figure 2.6: Power in a balanced three phase system

```
atand(imag(Ica),real(Ica)));
printf('\nThe magnitude of load voltage EAB is %.2f Volts and its angle is %.2f degrees',abs(EAB),atand(imag(EAB),real(EAB)));
printf('\nThe magnitude of load voltage EBC is %.2f Volts and its angle is %.2f degrees',abs(EBC),atand(imag(EBC),real(EBC)));
printf('\nThe magnitude of load voltage ECA is %.2f Volts and its angle is %.2f degrees',abs(ECA),atand(imag(ECA),real(ECA)));
```

**Scilab code Exa 2.5** `Power in a balanced three phase system`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 2 ; Example 2.5
//Scilab Version - 6.0.0 ; OS - Windows


clc;
clear;
Pim=400;                                        //Real Power of induction motor in kW
pfim=0.8;                                       //Power factor of the induction motor
```

```
Ssm=150;                                    //Apparent power of the synchronous motor in kVA
pfsm=0.9;                                   //Power factor of the synchronous motor
Vline=4160;                                 //RMS line voltage of AC supply in Volts
Sim=Pim/pfim;                               //Apparent power of the induction motor in kVA
Qim=sqrt(Sim*Sim-Pim*Pim);                  //Reactive power absorbed by the induction motor in kVAR
Psm=Ssm*pfsm;                               //Real power absorbed by the synchronous motor in kW
Qsm=sqrt(Ssm*Ssm-Psm*Psm);                  //Reactive power delivered by the synchronous motor in kVAR
P=Pim+Psm;                                  //Total real power of the combined load in kW
Q=Qim-Qsm;                                  //Total reactive power absorbed by the combined load in kVAR
S=sqrt(P*P+Q*Q);                            //Total apparent power absorbed by the combined load in kVA
pf=P/S;                                     //Power factor of the combined load
Iline=S*1000/(sqrt(3)*Vline);               //Line current of the combined load in Amperes
XCdel=3*Vline*Vline/(Q*1000);               //Capacitive reactance at each leg for unity power factor in Ohm
Iupf=P*1000/(sqrt(3)*Vline);                //Line current at unity poiwer factor
printf('\nThe power factor of the combined motor load is %f',pf);
printf('\nThe magnitude of line current delivered by the source is %f Amperes',Iline);
printf('\nThe magnitude of capacitive reactance at
```

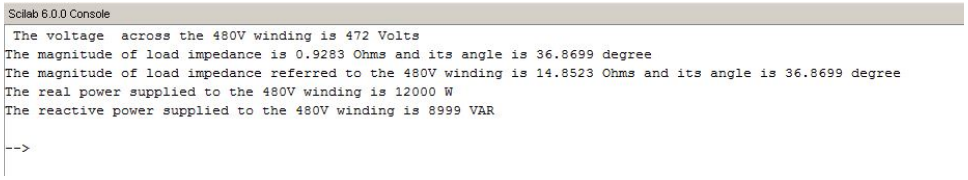
```
   each leg for unity power factor is %f Ohm',XCdel)
   ;
printf('\nThe magnitude of the line current
   delivered by the source with capacitor bank
   installed is %f Amperes',Iupf);
```

# Chapter 3

# POWER TRANSFORMERS

**Scilab code Exa 3.1** `Ideal single phase two winding transformer`

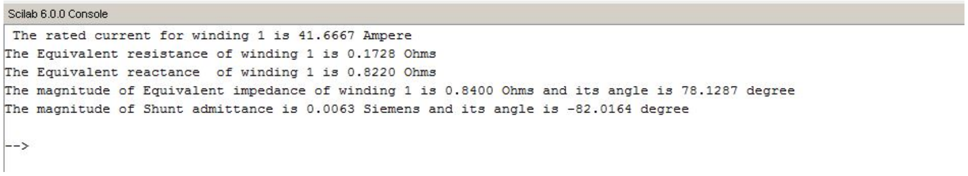
```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-3 ;Example 3.1
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;

Sr=20                                    //rated input power
```



```
 The voltage  across the 480V winding is 472 Volts
The magnitude of load impedance is 0.9283 Ohms and its angle is 36.8699 degree
The magnitude of load impedance referred to the 480V winding is 14.8523 Ohms and its angle is 36.8699 degree
The real power supplied to the 480V winding is 12000 W
The reactive power supplied to the 480V winding is 8999 VAR

-->
```

Figure 3.1: Ideal single phase two winding transformer

```
in kVA
E1rated=480                      //Rated voltage across winding 1 in Volts
E2rated=120                      //Rated voltage across winding 2 in Volts
F=60                             //frequency in Hertz
S1=15                            //Load power in kVA
pf = 0.8                         // power factor lagging
E2=118                           //Load voltage in Volts

at=E1rated/E2rated               // Calculation of turns ratio
E1=at*E2                         // voltage across winding 1 in Volts
theta=acos(pf)
S2=S1*exp(%i*theta)*1000         //complex load power in VA
I2=conj(S2)/conj(E2)             // Load current in Ampere
Z2=E2/I2                         // Load impedance in Ohms
Z2r=at^2*Z2                      //Load impedance referred to the 480V in Ohms
S1=S2                            //since complex power entering winding 1 is equal to the complex power leaving winding 2
P1=real(S1)
Q1=imag(S1)

printf('The voltage  across the 480V winding is %d Volts\n',E1);
printf('The magnitude of load impedance is %.4f Ohms and its angle is %.4f degree\n',abs(Z2),atand(imag(Z2),real(Z2)));
printf('The magnitude of load impedance referred to the 480V winding is %.4f Ohms and its angle is %
```



```
 The rated current for winding 1 is 41.6667 Ampere
The Equivalent resistance of winding 1 is 0.1728 Ohms
The Equivalent reactance  of winding 1 is 0.8220 Ohms
The magnitude of Equivalent impedance of winding 1 is 0.8400 Ohms and its angle is 78.1287 degree
The magnitude of Shunt admittance is 0.0063 Siemens and its angle is -82.0164 degree

-->
```

Figure 3.2: Transformer short circuit and open circuit tests

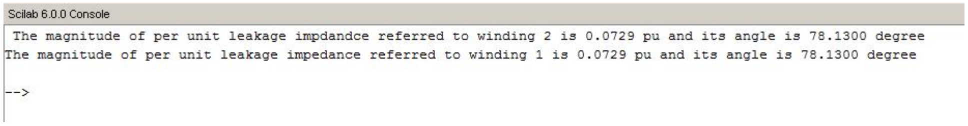
```
.4f degree\n',abs(Z2r),atand(imag(Z2r),real(Z2r)));
printf('The real power supplied to the 480V winding is %d W\n',P1);
printf('The reactive power supplied to the 480V winding is %d VAR\n',Q1);
```

---

**Scilab code Exa 3.2** `Transformer short circuit and open circuit tests`

```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-3 ;Example 3.2
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;

Srated=20                               //rated input power in kVA
E1rated=480                             //Rated voltage across winding 1 in Volts
E2rated=120                             //Rated voltage across winding 2 in Volts
```

```
F=60                                //frequency in
   Hertz
Sl=15                               //Load power in
   kVA
pf = 0.8                            //power factor
   lagging
E2=118                              //Load voltage
   in Volts
V1s=35                              //Short circuit
   voltage in Volts
P1=300                              //Short circuit
   power in Watts
I2=12                               //Open circuit
   Winding 2 current in Amps
P2=200                              //Open circuit
   power in Watts

I1rated=(Srated*1000)/E1rated       //Rated current
   for winding 1
Req1=P1/(I1rated)^2                 //Equivalent
   resistance of winding 1 in Ohms
Zeqm=abs(V1s/I1rated)               //Magnitude of
   equivalent impedance of winding 1 in Ohms
Xeq1=sqrt(Zeqm^2-Req1^2)            //Equivalent
   reactance  of winding 1 in Ohms
Zeq1=Req1+%i*Xeq1                   //Equivalent
   impedance of winding 1 in Ohms
V1o=E1rated                         //Since winding
   1 open circuit voltage is equal to winding 1
   rated volgage
Gc=P2/V1o^2
Ymm=abs((E2rated/E1rated)*I2/V1o)
Bm=sqrt(Ymm^2-Gc^2)
Ym=Gc-%i*Bm                         //Shunt
   admittance in Siemens

printf('The rated current for winding 1 is %.4f
   Ampere\n',I1rated);
```



```
 The magnitude of per unit leakage impdandce referred to winding 2 is 0.0729 pu and its angle is 78.1300 degree
The magnitude of per unit leakage impedance referred to winding 1 is 0.0729 pu and its angle is 78.1300 degree

-->
```

Figure 3.3: Per unit impedance single phase transformer

```
printf('The Equivalent resistance of winding 1 is %.4f Ohms\n',Req1);
printf('The Equivalent reactance  of winding 1 is %.4f Ohms\n',Xeq1);
printf('The magnitude of Equivalent impedance of winding 1 is %.4f Ohms and its angle is %.4f degree\n',abs(Zeq1),atand(imag(Zeq1),real(Zeq1)));
printf('The magnitude of Shunt admittance is %.4f Siemens and its angle is %.4f degree \n',abs(Ym),atand(imag(Ym),real(Ym)));
```

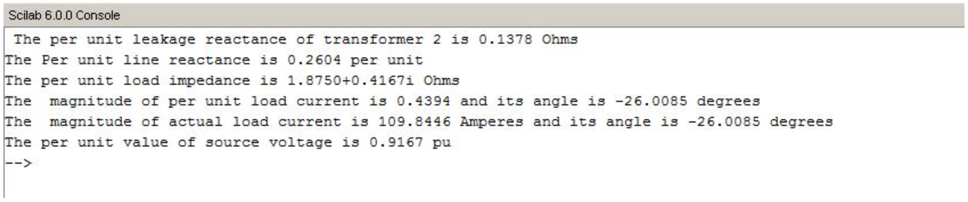
---

**Scilab code Exa 3.3** `Per unit impedance single phase transformer`

```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-3 ;Example 3.3
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;

Sb=20                                    //Base input power in kVA
```

```
Vb1=480                                //Base voltage across winding 1 in Volts
Vb2=120                                //Base voltage across winding 2 in Volts
f=60                                   //frequency in Hertz
Zeq2=0.0525*exp(%i*78.13*%pi/180)      //Equivalent impedance of the transformer referred to 120 Volt winding


Zb2=((Vb2^2)/(Sb*1000))                //Base impedance on the 120 Volts side of the transformer
Zeq2pu=Zeq2/Zb2                        //Per unit leakage impdeandce referred to winding 2
Zeq1=((Vb1/Vb2)^2)*Zeq2                //leakage impdeandce referred to winding 1
Zb1=((Vb1^2)/(Sb*1000))                //Base impedance on the 480 Volts side of the transformer
Zeq1pu=Zeq1/Zb1                        //Per unit leakage impdeandce referred to winding 1

printf('The magnitude of per unit leakage impdandce referred to winding 2 is %.4f pu and its angle is %.4f degree\n',abs(Zeq2pu),atand(imag(Zeq2pu),real(Zeq2pu)));
printf('The magnitude of per unit leakage impedance referred to winding 1 is %.4f pu and its angle is %.4f degree\n',abs(Zeq1pu),atand(imag(Zeq1pu),real(Zeq1pu)));
```

**Scilab code Exa 3.4** `Per unit circuit three zone single phase network`

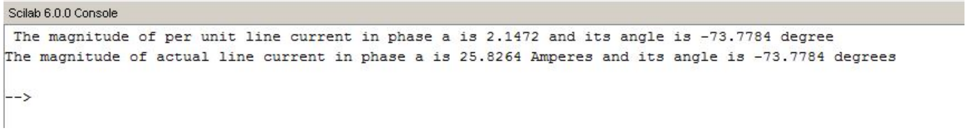


```
 The per unit leakage reactance of transformer 2 is 0.1378 Ohms
The Per unit line reactance is 0.2604 per unit
The per unit load impedance is 1.8750+0.4167i Ohms
The  magnitude of per unit load current is 0.4394 and its angle is -26.0085 degrees
The  magnitude of actual load current is 109.8446 Amperes and its angle is -26.0085 degrees
The per unit value of source voltage is 0.9167 pu
-->
```

Figure 3.4: Per unit circuit three zone single phase network

```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-3 ;Example 3.4
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;

Sb=30                                   //Base input power in kVA
Vg=220                                  //Actual value of source voltage
Vb1=240                                 //Base voltage across primary of transformer 1 in Volts
VT1p=240                                //Rated voltage of primary of transformer 1 in Volts
VT1s=480                                //Rated voltage of secondary of transformer 1 in Volts
VT2p=460                                //Rated voltage of primary of transformer 2 in Volts
VT2s=115                                //Rated voltage of secondary of transformer 2 in Volts
```

```
Xline=2                                                  // Line reactance in Ohms
Zload=.9+%i*.2                                           // Load impedance in Ohms
XT1=0.1                                                  // reactance of transformer 1 in per unit
XT2=0.1                                                  // reactance of transformer 2 in per unit
Sb1=30                                                   // MVA rating of transformer 1
Sb2=20                                                   // MVA rating of transformer 2
Vspu=Vg/Vb1;                                             // per unit source voltage

Vb2=(VT1s/VT1p)*Vb1                                      // Base voltage across the secondary of transformer 1 in Volts
Vb3=(VT2s/VT2p)*Vb2                                      // Base voltage across the secondary of transformer 2 in Volts
Zb2=(Vb2^2)/(Sb*1000)                                    // Base impedance of zone 2 in Ohms
Zb3=(Vb3^2)/(Sb*1000)                                    // Base impedance of zone 3 in Ohms
Ib3=(Sb*1000)/Vb3                                        // Base current in zone 3 in Amperes
XT1pu=0.1                                                // MVA rating of system is equal to kVA rating of transformer 1
XT2pu=(XT2)*((VT2p/Vb2)^2)*(Sb/Sb2 )                     // per unit leakage reactance of transformer 2
Xlinepu=Xline/Zb2                                        // Per unit line reactance
Zloadpu=Zload/Zb3                                        // per unit load impedance
Iloadpu=Vspu/(%i*(XT1+Xlinepu+XT2pu)+Zloadpu)            // per unit load current
```



```
 The magnitude of per unit line current in phase a is 2.1472 and its angle is -73.7784 degree
The magnitude of actual line current in phase a is 25.8264 Amperes and its angle is -73.7784 degrees

-->
```

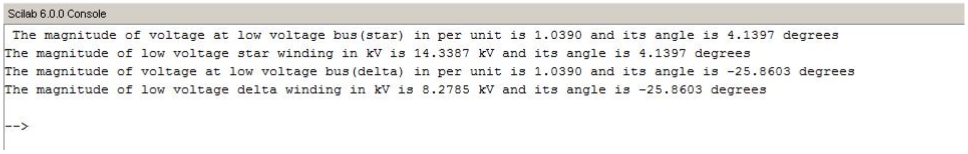
Figure 3.5: Per unit and actual currents in balanced three phase networks

```
Iload=Iloadpu*Ib3                         //Actual load current in Amperes


printf('The per unit leakage reactance of transformer 2 is %.4f Ohms\n',XT2pu);
printf('The Per unit line reactance is %.4f per unit\n',Xlinepu);
printf('The per unit load impedance is %.4f+%.4fi Ohms\n',real(Zloadpu),imag(Zloadpu));
printf('The  magnitude of per unit load current is %.4f and its angle is %.4f degrees\n',abs(Iloadpu),(180/%pi)*atan(imag(Iloadpu),real(Iloadpu)));
printf('The  magnitude of actual load current is %.4f Amperes and its angle is %.4f degrees\n',abs(Iload),(180/%pi)*atan(imag(Iload),real(Iload)));
printf('The per unit value of source voltage is %.4f pu',Vspu)
```

**Scilab code Exa 3.5** `Per unit and actual currents in balanced three phase networks`

```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-3 ;Example 3.5
//Scilab Version - 6.0.0; OS - Windows
```

```

clc;
clear;

Eab=480                                           //Line voltage of star connected voltage source in Volts
ZL=10*exp(%i*40*%pi/180)                          // Load impedance in Ohms
Zl=1*exp(%i*85*%pi/180)                           // Line impedance between source and load in Ohms
Sb=10                                             //Base power in kVA
VbLL=480                                          //line to line base voltage in Volts

Zb =((VbLL)^2/(Sb*1000))                          //Base impedance in Ohms
Zlpu=Zl/Zb                                        //per unit line impedance
ZLpu=ZL/Zb                                        //per unit load impedance
VbLN=VbLL/(sqrt(3))                               //line to neutral base voltage in Volts
Eanpu=(277*exp(%i*(-30)*%pi/180))/277             //source voltage in per unit
Iapu=Eanpu/(Zlpu+ZLpu)                            //per unit line current in phase a
Ib=(Sb*1000)/(sqrt(3)*VbLL)                       //base current in Amperes
Ia=Iapu*Ib                                        //actual phase a line current in Amperes

printf('The magnitude of per unit line current in phase a is %.4f and its angle is %.4f degree\n',abs(Iapu),atand(imag(Iapu),real(Iapu)));
printf('The magnitude of actual line current in phase a is %.4f Amperes and its angle is %.4f degrees\n',abs(Ia),atand(imag(Ia),real(Ia)));
```



```
 The magnitude of voltage at low voltage bus(star) in per unit is 1.0390 and its angle is 4.1397 degrees
The magnitude of low voltage star winding in kV is 14.3387 kV and its angle is 4.1397 degrees
The magnitude of voltage at low voltage bus(delta) in per unit is 1.0390 and its angle is -25.8603 degrees
The magnitude of low voltage delta winding in kV is 8.2785 kV and its angle is -25.8603 degrees

-->
```

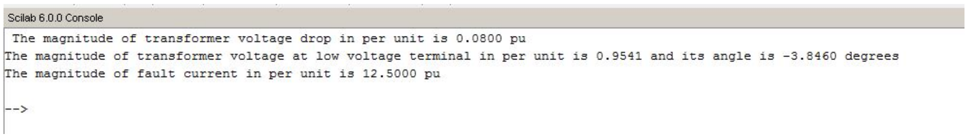
Figure 3.6: Voltage calculations balanced star star and delta star transformers

---

**Scilab code Exa 3.7** `Voltage calculations balanced star star and delta star transfo`

```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-3 ;Example 3.7
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;

Sr=400                                          //rated power of transformer in MVA
VT1p=13.8                                       //rated voltage of transformer primary side in kV
VT1s=199.2                                      //rated voltage of transformer secondary side in kV
Xeq=0.10                                        //leakage reactance of transformer in Ohms
```

```
Sa=1000 //High voltage side absorbs power in MVA
pf=0.90 // lagging power factor
VANH=199.2
Sb=1200 //base power in MVA
VbHLL=345 //Hihg volgage side lini to line base voltag in kV
IbH=1200/(345*sqrt(3)) //high voltage side base current in Amperes

VAN=1.0 //per unit load voltage
Theta=acos(0.9)
IA=((1000/(345*(sqrt(3))))/2.008)*(exp(%i*(-Theta))) //Per unit load current
Van=VAN+(%i*Xeq)*IA // voltage at low voltage bus
VbXLN1=13.8
Van1L=Van*VbXLN1 //low voltage wye winding in kV
Ean=(exp(%i*(-30)*(%pi/180)))*VAN //source voltage in per unit
Ia=(exp(%i*(-30)*(%pi/180)))*IA //source current in per unit
Van2=Ean+(%i*Xeq)*Ia
VbXLN2=13.8/(sqrt(3))
```



```
 The magnitude of transformer voltage drop in per unit is 0.0800 pu
The magnitude of transformer voltage at low voltage terminal in per unit is 0.9541 and its angle is -3.8460 degrees
The magnitude of fault current in per unit is 12.5000 pu

-->
```

Figure 3.7: Per unit voltage drop and per unit fault current balanced three phase transformer

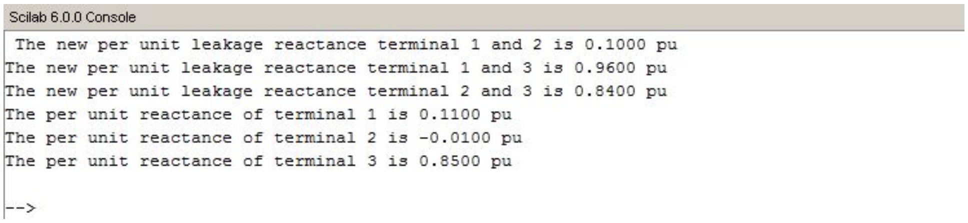
```
Van2L=Van2*VbXLN2                                           //low voltage delta winding in kV

printf('The magnitude of voltage at low voltage bus(star) in per unit is %.4f and its angle is %.4f degrees\n',abs(Van),atand(imag(Van),real(Van)));
printf('The magnitude of low voltage star winding in kV is %.4f kV and its angle is %.4f degrees\n',abs(Van1L),atand(imag(Van1L),real(Van1L)));
printf('The magnitude of voltage at low voltage bus(delta) in per unit is %.4f and its angle is %.4f degrees\n',abs(Van2),atand(imag(Van2),real(Van2)));
printf('The magnitude of low voltage delta winding in kV is %.4f kV and its angle is %.4f degrees\n',abs(Van2L),atand(imag(Van2L),real(Van2L)));
```

---

**Scilab code Exa 3.8** `Per unit voltage drop and per unit fault current balanced thre`

```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-3 ;Example 3.8
//Scilab Version - 6.0.0; OS - Windows
```

```

clc;
clear;

Sr=200                                          //rated power of transformer in MVA
VT1p=345                                        // rated voltage of transformer primary side in kV
VT1s=34.5                                       // rated voltage of transformer secondary side in kV
Xeq=0.08                                        //leakage reactance of transformer in ohms
pf=0.8                                          //lagging power factor
Irated=1.0                                      //rated current in Amperes
Irated1=1.0*exp(%i*(-36.87)*(%pi/180))          //consider real and imaginary value of rated current
VAN=1.0                                         //source voltage in Volts
Vdrop=Irated*Xeq                                //per unit magnitudes of transformer voltage drop
Van=VAN-(%i*Xeq)*Irated1                        //per unit magnitudes of transformer voltage at low voltage terminals
Isc=VAN/Xeq                                     //per unit magnitudes of transformer fault current


printf('The magnitude of transformer voltage drop in per unit is %.4f pu \n',Vdrop);
printf('The magnitude of transformer voltage at low voltage terminal in per unit is %.4f and its angle is %.4f degrees\n',abs(Van),atand(imag(Van),real(Van)));
printf('The magnitude of fault current in per unit is %.4f pu\n',Isc);
```



```
 The new per unit leakage reactance terminal 1 and 2 is 0.1000 pu
The new per unit leakage reactance terminal 1 and 3 is 0.9600 pu
The new per unit leakage reactance terminal 2 and 3 is 0.8400 pu
The per unit reactance of terminal 1 is 0.1100 pu
The per unit reactance of terminal 2 is -0.0100 pu
The per unit reactance of terminal 3 is 0.8500 pu

-->
```

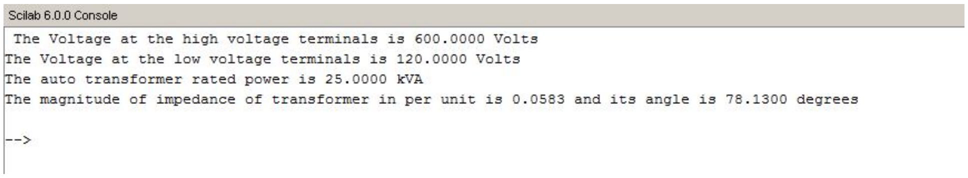
Figure 3.8: Three winding single phase transformer per unit impedances

---

**Scilab code Exa 3.9** `Three winding single phase transformer per unit impedances`

```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-3 ;Example 3.9
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;

Sb=300                          //rated power of transformer in MVA
Vb1=13.8                        // Terminal 1 base voltage in kV
Vb2=199.2                       // Terminal 2 base voltage in kV
Vb3=19.92                       // Terminal 3 base voltage in kV
X12old=0.10                     //given per unit leakage reactance terminal 1 and 2
X13old=0.16                     //given per unit
```

```
leakage reactance terminal 1 and 3
X23old=0.14                            //given per unit leakage reactance terminal 2 and 3
Sb12=300                               //rated power corresponding to leakage reactance X12 in MVA
Sb13=50                                //rated power corresponding to leakage reactance X13 in MVA
Sb23=50                                //rated power corresponding to leakage reactance X23 in MVA

X12new=X12old*(Sb/Sb12)                //new per unit leakage reactance terminal 1 and 2
X13new=X13old*(Sb/Sb13)                //new per unit leakage reactance terminal 1 and 3
X23new=X23old*(Sb/Sb23)                //new per unit leakage reactance terminal 2 and 3
X1=(1/2)*(X12new+X13new-X23new)
X2=(1/2)*(X12new+X23new-X13new)
X3=(1/2)*(X13new+X23new-X12new)

printf('The new per unit leakage reactance terminal 1 and 2 is %.4f pu\n',X12new);
printf('The new per unit leakage reactance terminal 1 and 3 is %.4f pu\n',X13new);
printf('The new per unit leakage reactance terminal 2 and 3 is %.4f pu\n',X23new);
printf('The per unit reactance of terminal 1 is %.4f pu\n',X1);
printf('The per unit reactance of terminal 2 is %.4f pu\n',X2);
printf('The per unit reactance of terminal 3 is %.4f pu\n',X3);
```



```
 The Voltage at the high voltage terminals is 600.0000 Volts
The Voltage at the low voltage terminals is 120.0000 Volts
The auto transformer rated power is 25.0000 kVA
The magnitude of impedance of transformer in per unit is 0.0583 and its angle is 78.1300 degrees

-->
```

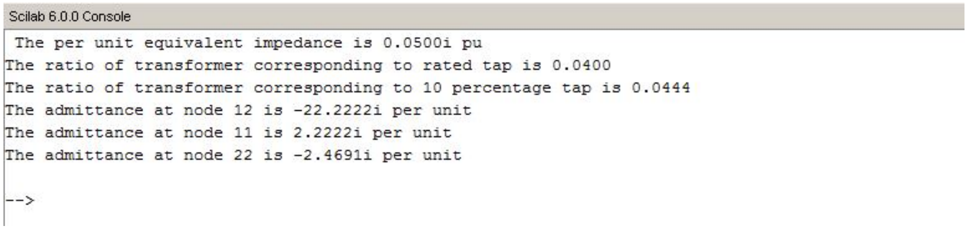
Figure 3.9: Autotransformer single phase

**Scilab code Exa 3.11** `Autotransformer single phase`

```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-3 ;Example 3.11
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;

Sr=20                                   //rated power of transformer in kVA
E1=120                                  //voltage at 120 Volt winding
E2=480                                  //voltage induced across the 480 Volt winding
Zleak=0.0729*exp(%i*78.13*(%pi/180))    //per unit leakage impedance of two winding transformer

EH=E1+E2                                //Voltage at the high Voltage terminals
I2=((Sr*1000)/E2)                       //
```

```
   rated current of 480 Volt winding in Ampere
SH=EH*I2
   //kVA rating of 480 Volt winding
I1=(E2/E1)*I2
   //Current induced in the 120 Volt winding
Ix=I1+I2
Sx=E1*Ix
   //auto transformer rated power
ZbaseHold=((E2)^2)/(Sr*1000)                                          //base impedance at high voltage terminal of normal transformer
ZbaseHnew=((EH)^2)/(Sx)                                                      //base impedance at high voltage terminal of autotransformer
Zpunew=(0.0729*exp(%i*78.13*(%pi/180)))*(ZbaseHold/ZbaseHnew)    //per unit impedance of transformer

printf('The Voltage at the high voltage terminals is %.4f Volts\n',EH);
printf('The Voltage at the low voltage terminals is %.4f Volts\n',E1);
printf('The auto transformer rated power is %.4f kVA\n',Sx/1000);
printf('The magnitude of impedance of transformer in per unit is %.4f and its angle is %.4f degrees\n',abs(Zpunew),atand(imag(Zpunew),real(Zpunew)));
```

**Scilab code Exa 3.12** Tap changing three phase transformer per unit positive sequen



```
Scilab 6.0.0 Console
 The per unit equivalent impedance is 0.0500i pu
The ratio of transformer corresponding to rated tap is 0.0400
The ratio of transformer corresponding to 10 percentage tap is 0.0444
The admittance at node 12 is -22.2222i per unit
The admittance at node 11 is 2.2222i per unit
The admittance at node 22 is -2.4691i per unit

-->
```

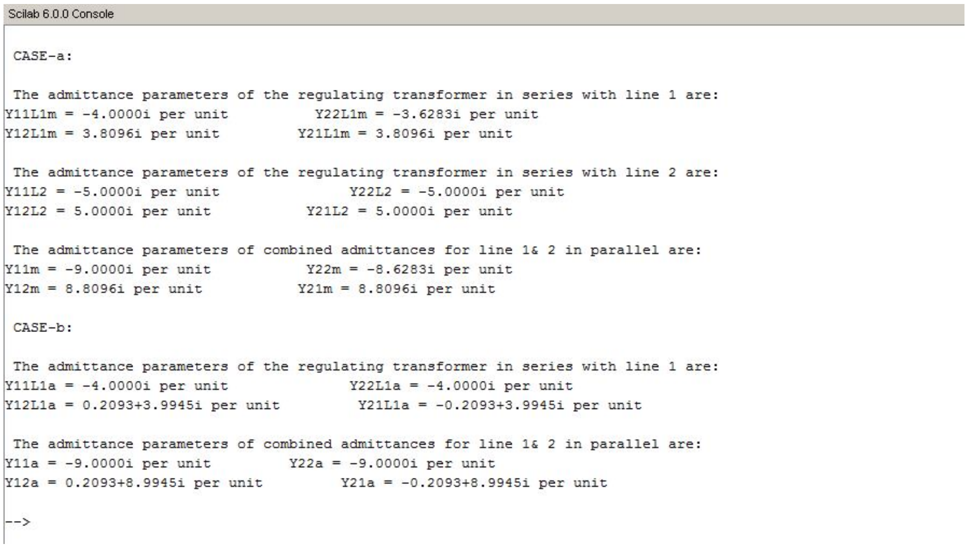
Figure 3.10: Tap changing three phase transformer per unit positive sequence network

```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-3 ;Example 3.12
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;

Sr=1000                               //rated power of transformer in kVA
V1rated=13.8                          //rated voltage of delta winding of transformer in kV
V2rated=345                           //rated voltage of wye winding of transformer in kV
Zeq=%i*0.10                           //per unit equivalent impedance
Sb=500                                //rated power of transformer in MVA
VbXLL=13.8                            //line to line X terminal base voltage in kV
VbHLL=345                             //line to line H terminal base voltage in kV


at=(V1rated/V2rated)                  //ratio of transformer corresponding to rated tap
```

```
b=(VbXLL/VbHLL)
c=at/b
Zpunew=Zeq*(Sb/Sr)                    //per unit equivalent impedance
at10=(V1rated/(V2rated*0.9))          //ratio of transformer corresponding to 10 percentage tap
b10=(V1rated/V2rated)
c10=(at10/b10)
Yeq=(1/Zpunew)
Y12=c10*Yeq                           //admittance at node 12 in per unit
Y11=(1-c10)*Yeq                       //admittance at node 11 in per unit
Y22=(((abs(c10))^2)-c10)*Yeq            //admittance at node 22 in per unit


printf('The per unit equivalent impedance is %.4fi pu\n',imag(Zpunew));
printf('The ratio of transformer corresponding to rated tap is %.4f\n',at);
printf('The ratio of transformer corresponding to 10 percentage tap is %.4f\n',at10);
printf('The admittance at node 12 is %.4fi per unit\n',imag(Y12));
printf('The admittance at node 11 is %.4fi per unit\n',imag(Y11));
printf('The admittance at node 22 is %.4fi per unit\n',imag(Y22));
```

---

**Scilab code Exa 3.13** `Voltage regulating and phase shifting three phase transformer`



```
 CASE-a:

 The admittance parameters of the regulating transformer in series with line 1 are:
Y11L1m = -4.0000i per unit          Y22L1m = -3.6283i per unit
Y12L1m = 3.8096i per unit         Y21L1m = 3.8096i per unit

 The admittance parameters of the regulating transformer in series with line 2 are:
Y11L2 = -5.0000i per unit               Y22L2 = -5.0000i per unit
Y12L2 = 5.0000i per unit           Y21L2 = 5.0000i per unit

 The admittance parameters of combined admittances for line 1& 2 in parallel are:
Y11m = -9.0000i per unit           Y22m = -8.6283i per unit
Y12m = 8.8096i per unit           Y21m = 8.8096i per unit

 CASE-b:

 The admittance parameters of the regulating transformer in series with line 1 are:
Y11L1a = -4.0000i per unit              Y22L1a = -4.0000i per unit
Y12L1a = 0.2093+3.9945i per unit         Y21L1a = -0.2093+3.9945i per unit

 The admittance parameters of combined admittances for line 1& 2 in parallel are:
Y11a = -9.0000i per unit         Y22a = -9.0000i per unit
Y12a = 0.2093+8.9945i per unit         Y21a = -0.2093+8.9945i per unit

-->
```

Figure 3.11: Voltage regulating and phase shifting three phase transformers

```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-3 ;Example 3.13
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;

XL1=0.25                                //Positive sequence series reactance at parallel line 1 in per unit
XL2=0.20                                //Positive sequence series reactance at parallel line 2 in per unit
Cm=0.9524

Y11L1m=(1/(%i*.25))                     //The voltage magnitude regulating transformer admittance Y11L1m
Y22L1m=(Cm^2)*Y11L1m                    //The voltage
```

```
magnitude regulating transformer admittance Y22L1m
Y12L1m=(-Cm)*Y11L1m                 //The voltage magnitude regulating transformer admittance Y12L1m
Y21L1m=(-Cm)*Y11L1m                 //The voltage magnitude regulating transformer admittance Y21L1m
Y11L2=(1/(%i*.20))
Y22L2=(1/(%i*.20))
Y12L2=-Y11L2
Y21L2=-Y11L2
Y11m=Y11L1m+Y11L2                   //parallel admittance Y11m
Y22m=Y22L1m+Y22L2                   //parallel admittance Y11m
Y12m=Y12L1m+Y12L2                   //parallel admittance Y11m
Y21m=Y12L1m+Y12L2                   //parallel admittance Y11m
Y11L1a=(1/(%i*.25))                 //The phase angle regulating transformer admittance Y11L1a
Ca=1.0*(exp(%i*(-3)*(%pi/180)))
Y22L1a=((abs(Ca))^2)*(-%i*4.0)      //The phase angle regulating transformer admittance Y22L1a
Y12L1a= (-Ca)*(-%i*4.0)             //The phase angle regulating transformer admittance Y12L1a
Y21L1a= (-conj(Ca))*(-%i*4.0)       //The phase angle regulating transformer admittance Y21L1a
Y11a=Y11L1a+Y11L2                   //parallel admittance Y11a
Y22a=Y22L1a+Y22L2                   //parallel admittance Y22a
Y12a=Y12L1a+Y12L2                   //parallel admittance Y12a
Y21a=Y21L1a+Y21L2                   //parallel admittance Y21a

```

```
disp('CASE-a:')
disp('The admittance parameters of the regulating transformer in series with line 1 are:')
printf('Y11L1m = %.4fi per unit          Y22L1m = %.4fi per unit\n',imag(Y11L1m),imag(Y22L1m));
printf('Y12L1m = %.4fi per unit\          Y21L1m = %.4fi per unit\n',imag(Y12L1m),imag(Y21L1m));

disp('The admittance parameters of the regulating transformer in series with line 2 are:')
printf('Y11L2 = %.4fi per unit                    Y22L2 = %.4fi per unit\n',imag(Y11L2),imag(Y22L2));
printf('Y12L2 = %.4fi per unit             Y21L2 = %.4fi per unit\n',imag(Y12L2),imag(Y21L2));

disp('The admittance parameters of combined admittances for line 1& 2 in parallel are:')
printf('Y11m = %.4fi per unit              Y22m = %.4fi per unit\n',imag(Y11m),imag(Y22m));
printf('Y12m = %.4fi per unit              Y21m = %.4fi per unit\n',imag(Y12m),imag(Y21m));

disp('CASE-b:')
disp('The admittance parameters of the regulating transformer in series with line 1 are:')
printf('Y11L1a = %.4fi per unit                  Y22L1a = %.4fi per unit\n',imag(Y11L1a),imag(Y22L1a));
printf('Y12L1a = %.4f+%.4fi per unit          ',real(Y12L1a),imag(Y12L1a));
printf('Y21L1a = %.4f+%.4fi per unit\n',real(Y21L1a),imag(Y21L1a));

disp('The admittance parameters of combined admittances for line 1& 2 in parallel are:')
printf('Y11a = %.4fi per unit          Y22a = %.4fi per unit\n',imag(Y11a),imag(Y22a));
printf('Y12a = %.4f+%.4fi per unit          ',real(Y12a),imag(Y12a));
```

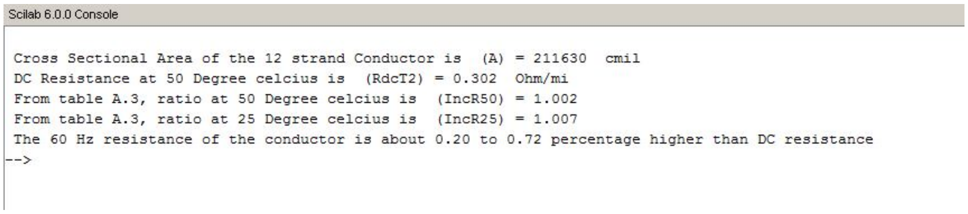
```
printf('Y21a = %.4f+%.4fi per unit\n',real(Y21a),
   imag(Y21a));
```

# Chapter 4

# TRANSMISSION LINE PARAMETERS

**Scilab code Exa 4.1** `Stranded conductor dc and ac resistance`

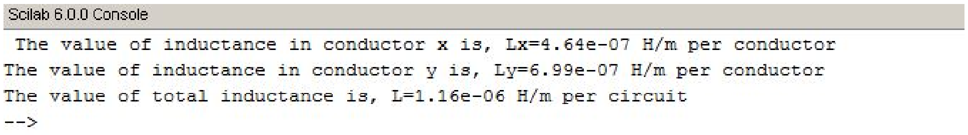
```
// Book - Power System: Analysis & Design 5th Edition
// Authors - J. Duncan Glover, Mulukutla S. Sharma, Thomas J. Overbye
// Chapter - 4 : Example 4.1
// Scilab Version 6.0.0 : OS - Windows

clc;
```



```
Cross Sectional Area of the 12 strand Conductor is  (A) = 211630  cmil
DC Resistance at 50 Degree celcius is  (RdcT2) = 0.302  Ohm/mi
From table A.3, ratio at 50 Degree celcius is  (IncR50) = 1.002
From table A.3, ratio at 25 Degree celcius is  (IncR25) = 1.007
The 60 Hz resistance of the conductor is about 0.20 to 0.72 percentage higher than DC resistance
-->
```

Figure 4.1: Stranded conductor dc and ac resistance

```
clear;

S = 12;                          // Number of strands
Sd = 0.1328;                     // Diameter of the Strand
R = 0.302;                       // Resistance at 50 Deg Celcius in Ohm/miles
f = 60;                          // Frequency
T = 241.5;                       // Temperature Constant of Hard Drawn Copper
T1 = 20;                         // Temperature in Degree Celcius
T2 = 50;                         // Temperature in Degree Celcius
T3 = 25;                         // Temperature in Degree Celcius
R60T2 = 0.303;                   // Resistance at 60 Hz with 50 degree celcius From the Table A.3
R60T3 = 0.278;                   // Resistance at 60 Hz with 25 degree celcius From the Table A.3
RdcT3 = 0.276;                   // DC Resistace at 25 Degree Celcius

Sd = (0.1328*1000);              // Coverting Strand Diameter from inch to mil/inch
A = 12*Sd^2 ;                    // Cross Sectional Area of the 12 strand Conductors in cmil
pT1 = 10.66;                     // Resistivity at Temperature T1
pT2 = pT1*((T2+T)/(T1+T));       // Resistivity at 50 deg Celcius in Ohm-cmil/ft
L = (5280*1.02);                 // Length of the Conductor in ft
RdcT2 = (pT2*L)/A;               // DC Resistance at 50 Degree celcius in Ohm/miles
IncR50 = (R60T2)/(RdcT2);        // Percentage Increase in Resistace for 50 degree celcius at 60 Hz Versus dc
```



```
 The value of inductance in conductor x is, Lx=4.64e-07 H/m per conductor
The value of inductance in conductor y is, Ly=6.99e-07 H/m per conductor
The value of total inductance is, L=1.16e-06 H/m per circuit
-->
```

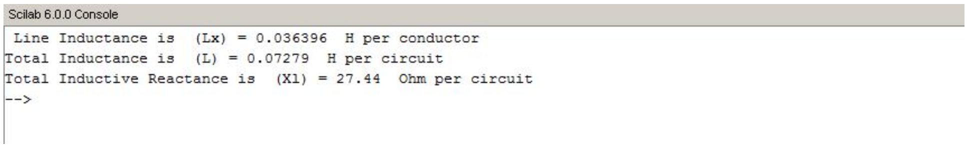
Figure 4.2: GMR GMD and inductance single phase two conductor line

```
IncR25 = (R60T3)/(RdcT3);        // Percentage Increase in Resistace for 25 degree celcius at 60 Hz Versus dc


printf('\n Cross Sectional Area of the 12 strand Conductor is  (A) = %0.0f  cmil',A);
printf('\n DC Resistance at 50 Degree celcius is  (RdcT2) = %0.3f  Ohm/mi',RdcT2);
printf('\n From table A.3, ratio at 50 Degree celcius is  (IncR50) = %0.3f ',IncR50);
printf('\n From table A.3, ratio at 25 Degree celcius is  (IncR25) = %0.3f ',IncR25);
printf('\n The 60 Hz resistance of the conductor is about %.2f to %.2f percentage higher than DC resistance',(IncR50-1)*100,(IncR25-1)*100);

//There is a small variation in the result since the value of cross sectional area which is actually 211630 is rounded off to 211600 in the book.
```

**Scilab code Exa 4.2** `GMR GMD and inductance single phase two conductor line`

```
// Book - Power System: Analysis & Design 5th Edition
// Authors - J. Duncan Glover, Mulukutla S. Sharma,
```

```
Thomas J. Overbye
// Chapter - 4 : Example 4.2
// Scilab Version 6.0.0 : OS - Windows

clc;
clear;

rx=0.03;                    //Radius of conductor x in meter
ry=0.04;                    //Radius of conductor y in meter
N=3;
M=2;

Ddash=[4 4.3;3.5 3.8;2 2.3];                //Equivalent distances in meter to find Dxy
Dx=[exp(-1/4)*rx 0.5 2;0.5 exp(-1/4)*rx 1.5;2 1.5 exp(-1/4)*rx];       //Equivalent distances in meter to find Dxx
Dy=[exp(-1/4)*ry 0.3;0.3 exp(-1/4)*ry];                    //Equivalent distances in meter to find Dyy

Dxyr=1;
for i=1:N
    for j=1:M
        Dxyr=Dxyr*Ddash(i,j)
    end
end
Dxy=nthroot(Dxyr,M*N);

Dxxr=1
for i=1:N
    for j=1:N
        Dxxr=Dxxr*Dx(i,j)
    end
end
Dxx=nthroot(Dxxr,N*N);
```



```
 Line Inductance is  (Lx) = 0.036396  H per conductor
Total Inductance is  (L) = 0.07279  H per circuit
Total Inductive Reactance is  (Xl) = 27.44  Ohm per circuit
-->
```

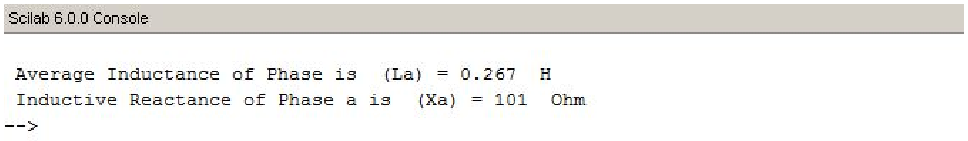
Figure 4.3: Inductance and inductive reactance single phase line

```

Dyyr=1
for i=1:M
    for j=1:M
        Dyyr=Dyyr*Dy(i,j)
    end
end
Dyy=nthroot(Dyyr,M*M);

Lx=2e-7*log(Dxy/Dxx);
Ly=2e-7*log(Dxy/Dyy);
L=Lx+Ly;

printf('The value of inductance in conductor x is, Lx=%3.2e H/m per conductor\n',Lx)
printf('The value of inductance in conductor y is, Ly=%3.2e H/m per conductor\n',Ly)
printf('The value of total inductance is, L=%3.2e H/m per circuit',L)
```

### Scilab code Exa 4.3 Inductance and inductive reactance single phase line

```
// Book - Power System: Analysis & Design 5th Edition
// Authors - J. Duncan Glover, Mulukutla S. Sharma, Thomas J. Overbye
```

```
// Chapter - 4 : Example 4.3
// Scilab Version 6.0.0 : OS - Windows


clc;
clear;

f = 60;                                          // Single Phase line operating fruquency in Hz
S = 12;                                          // Strand Copper conductors
Dxy = 5;                                         // Geometrical Mean Distance between conductor centers in ft
Dxx =0.01750;                                    // Geometrical Mean Radiance of Copper Conductor in feet from Table A.3
Dyy = Dxx;
l = 20;                                          // Line length in miles

Lx = (2*10^-7)*log(Dxy/Dxx)*1609*l;              // Line Inductance in Henry per conductor
Ly = Lx;
L = Lx+Ly;                                       // Total Inductance in Henry per Circuit
Xl = (2*%pi*f*L);                                // Total Inductive Reactance in Ohm per circuit

printf('Line Inductance is  (Lx) = %f  H per conductor',Lx);
printf('\nTotal Inductance is  (L) = %0.5f  H per circuit',L);
printf('\nTotal Inductive Reactance is  (Xl) = %0.2f Ohm per circuit',Xl);
```

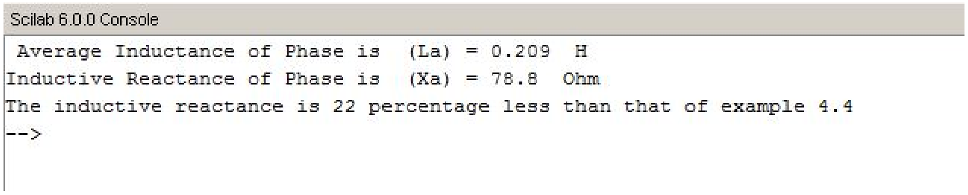


```
 Average Inductance of Phase is  (La) = 0.267  H
 Inductive Reactance of Phase a is  (Xa) = 101  Ohm
-->
```

Figure 4.4: Inductance and inductive reactance three phase line

**Scilab code Exa 4.4** `Inductance and inductive reactance three phase line`

```
// Book - Power System: Analysis & Design 5th Edition
// Authors - J. Duncan Glover, Mulukutla S. Sharma, Thomas J. Overbye
// Chapter - 4 : Example 4.4
// Scilab Version 6.0.0 : OS - Windows


clc;
clear;

f = 60;                                    // Frequency of the Three Phase Line in Hz
Q = 10;                                    // Spacing between Adjacent Conductors in metres
T = 1590000;                               // Size of the Conductor in cmil
l = 200;                                   // Line Length in Kilometres
```



```
 Average Inductance of Phase is  (La) = 0.209  H
Inductive Reactance of Phase is  (Xa) = 78.8  Ohm
The inductive reactance is 22 percentage less than that of example 4.4
-->
```

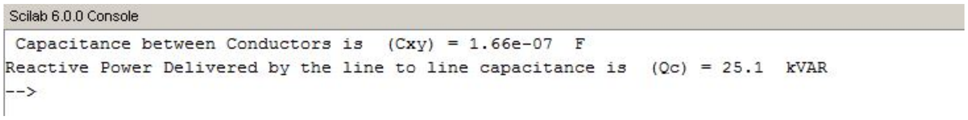
Figure 4.5: Inductive reactance three phase line with bundled conductors

```

Ds = (0.0520)*(1/3.28);                             // From Table A.4, the GMR of a 15,90,000 cmil 54/3 ACSR condutor in metres
Deq = nthroot([10*10*20],3);                        // Eqivalent GMR of a Conductor in metres
La = (2*10^-7)*(log(Deq/Ds))*(1000/1)*(200);        // Average Inductance of Phase a in Henry
Xa = (2*%pi*f*La);                                  // Inductive Reactance of Phase a in Ohm

printf('\n Average Inductance of Phase is  (La) = %0.3f  H', La);
printf('\n Inductive Reactance of Phase a is  (Xa) = %0.0f  Ohm', Xa);
```

**Scilab code Exa 4.5** Inductive reactance three phase line with bundled conductors

```
// Book - Power System: Analysis & Design 5th Edition
// Authors - J. Duncan Glover, Mulukutla S. Sharma,
```

```
Thomas J. Overbye
// Chapter - 4 : Example 4.5
// Scilab Version 6.0.0 : OS - Windows


clc;
clear;

T = 795000; // Size of the Conductor in Circular mils (cmil)
S = 0.40; // Spacing between Conductors in metre
l = 10; // Spacing between the Adjacent Conductors in metres
f = 60;
Xa4=101; // Inductive reactance of Example 4.4 in Ohms

r = 0.0375; // Geometric Mean Radius at 60Hz in feet, from the table A.4
Ds = r*(1/3.28); // Solid Cylinderical Conductor value in metres
Dsl = sqrt(Ds*S); // For the two conductor bundle GMR in metres
Deq = nthroot([10*10*20], 3); // Eqivalent GMR of a Conductor in metres from Ex 4.4
La = ((2*10^-7)*(log(Deq/Dsl))*(1000*200)); // Average Inductance of Phase a in Henry
Xa = (2*%pi*f*La); // Inductive Reactance of Phase a in Ohms

dif=100-(Xa/Xa4)*100;

printf('Average Inductance of Phase is  (La) = %0.3f H', La);
```



```
 Capacitance between Conductors is  (Cxy) = 1.66e-07  F
Reactive Power Delivered by the line to line capacitance is  (Qc) = 25.1  kVAR
-->
```

Figure 4.6: Capacitance admittance and reactive power supplied single phase line

```
printf('\nInductive Reactance of Phase is  (Xa) = %0.1f  Ohm', Xa);
printf('\nThe inductive reactance is %0.0f percentage less than that of example 4.4',dif );
```

---

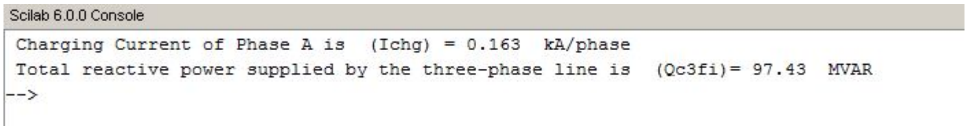
**Scilab code Exa 4.6** `Capacitance admittance and reactive power supplied single phas`

```
// Book - Power System: Analysis & Design 5th Edition
// Authors - J. Duncan Glover, Mulukutla S. Sharma, Thomas J. Overbye
// Chapter - 4 : Example 4.6
// Scilab Version 6.0.0 : OS - Windows


clc;
clear;

V = 20;                                           // Line Voltage in kV
D = 0.552;                                        // Diameter of a 4/0 12 strand copper conductor From Table A.3
f = 60;
```

```
// Frequency Hz


r = (D/2)*(1/12);                                            // The radius of a 4/0 12 strand copper conductor From Table A.3
e = 8.854*10^-12;
C = (%pi*e)/log(5/r);
Cxy = C*1609*20;                                             // Capacitance between Conductors in F/m
w = (2*%pi*f);                                               // Angular Velocity in rad/sec
Yxy = (%i)*(w)*(Cxy);                                        // Shunt Admitance Siemens
Qc = abs(Yxy)*(20*10^3)^2*(1/1000);                          // Reactive Power Delivered by the linw to line capacitance in kVAR


printf('Capacitance between Conductors is  (Cxy) = %0.2e  F', Cxy);
printf('\nReactive Power Delivered by the line to line capacitance is  (Qc) = %0.1f  kVAR', Qc);
```

**Scilab code Exa 4.7** Capacitance and shunt admittance charging current and reactive

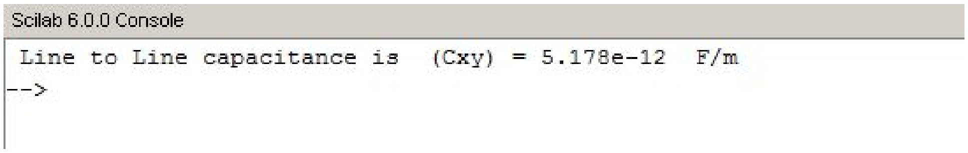
```
// Book - Power System: Analysis & Design 5th Edition
```



```
Scilab 6.0.0 Console
 Charging Current of Phase A is  (Ichg) = 0.163  kA/phase
 Total reactive power supplied by the three-phase line is  (Qc3fi)= 97.43  MVAR
-->
```

Figure 4.7: Capacitance and shunt admittance charging current and reactive power supplied three phase line

```
// Authors - J. Duncan Glover, Mulukutla S. Sharma, Thomas J. Overbye
// Chapter - 4 : Example 4.7
// Scilab Version 6.0.0 : OS - Windows


clc;
clear;

V = 345;                                                            // Line Voltage in kV
T = 795000;                                                     // Size of the Conductor in cmil
D = 1.108;                                                        // Diameter of the conductor in inch
f= 60;                                                  // Frequency in Hz
e = 8.854*10^-12;

r = (D/2)*0.0254;                                          // Radius of the copper conductor in metre
Dsc = sqrt((r)*(0.40));                               // Equivalent radius of the two onductor bundle
Deq = nthroot([10*10*20], 3);
```



```
Line to Line capacitance is  (Cxy) = 5.178e-12  F/m
-->
```

Figure 4.8: Effect of earth on capacitance single phase line

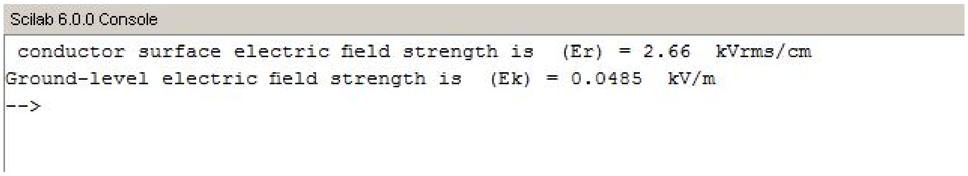
```
                                            // Eqivalent GMR of a onductor in metres from Ex 4.5
Can = (2*%pi)*(e)/(log(12.6/0.0750))*(1000)*(200);          // Deviation of the capacitance in Farad
w = (2*%pi*f);                                                           // Angular Velocity in rad/sec
Yan = (%i*w*Can);                                                  // Shunt admitance-to-neutral in Siemens
e = (V/sqrt(3));
Ichg = (abs(Yan)*e);                                          // Charging Current of Phase A
Qc3fi = (abs(Yan)*(345)^2);                          // Total reactive power supplied by the three-phase line in MVAR

printf('Charging Current of Phase A is  (Ichg) = %0.3f  kA/phase', Ichg);
printf('\n Total reactive power supplied by the three-phase line is  (Qc3fi)= %0.2f  MVAR', Qc3fi);
```

**Scilab code Exa 4.8** `Effect of earth on capacitance single phase line`

```
// Book - Power System: Analysis & Design 5th Edition
// Authors - J. Duncan Glover, Mulukutla S. Sharma, Thomas J. Overbye
// Chapter - 4 : Example 4.8
// Scilab Version 6.0.0 : OS - Windows

clc;
clear;

H = 18;                                        // Average line heightin ft
e = 8.854*10^-12;
D = 5;                                         // Diameter of the conductor in ft
r = 0.023;                                     // Radius of the copper conductor ft

Hxx = 2*(H);                                   // Geometric mean radius in ft
Hxy = sqrt((Hxx)^2 + (5)^2);                   // Geometric mean distance in ft
Cxy = ((%pi)*(e))/((log(D/r))-(log(Hxy/Hxx))); // Line to Line capacitance in F/m

printf('Line to Line capacitance is (Cxy) = %0.3e F/m', Cxy);
```

**Scilab code Exa 4.9** Conductor surface and ground level electric field strengths si

```
// Book - Power System: Analysis & Design 5th Edition
// Authors - J. Duncan Glover, Mulukutla S. Sharma,
```



```
 conductor surface electric field strength is  (Er) = 2.66  kVrms/cm
Ground-level electric field strength is  (Ek) = 0.0485  kV/m
-->
```

Figure 4.9: Conductor surface and ground level electric field strengths single phase line

```
Thomas J. Overbye
// Chapter - 4 : Example 4.9
// Scilab Version 6.0.0 : OS - Windows


clc;
clear;

Vxy = 20;  // Line voltage in kV
e = 8.854*10^-12;
r = (0.023*0.3048);  // Radius of the copper conductor in metre

Cxy = 5.178*10^-12;  // Line to Line capacitance in F/m
qx = ((Cxy)*(Vxy)*(10^3));  // Charge in Columb/metre
qy = -qx;  // Charge in Columb/metre
Er = (qx/(2*%pi*e*r))*(1/1000)*(1/100);  // conductor surface electric field strength in kVrms/cm
Xx = -0.762;
```

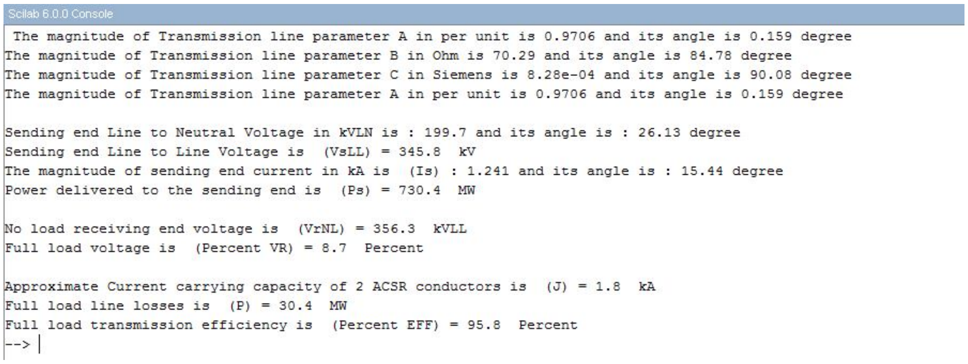
```
                                                                  // Coordinate for conuctor x with the reference point R
Yx = 0.762;                                                       // Coordinate for conuctor Y with the reference point R
w = 5.49;                                                         // Distance of the conductor from the reference point along Y axis
z = (2*%pi*e);
g = ((2*w)/(w^2));
n =  (2*w)/((5.49)^2+(Yx+Yx)^2);
Ek = (qx/z)*(g-n)*10^-3;                                          // Ground-level electric field strength in kV/m

printf('conductor surface electric field strength is (Er) = %0.2f kVrms/cm', Er);
printf('\nGround-level electric field strength is (Ek) = %0.4f kV/m', Ek);
```

# Chapter 5

# TRANSMISSION LINES STEADY STATE OPERATION

**Scilab code Exa 5.1** `ABCD parameters and the nominal pi circuit medium length line`

```
// Book - Power System: Analysis & Design 5th Edition
```



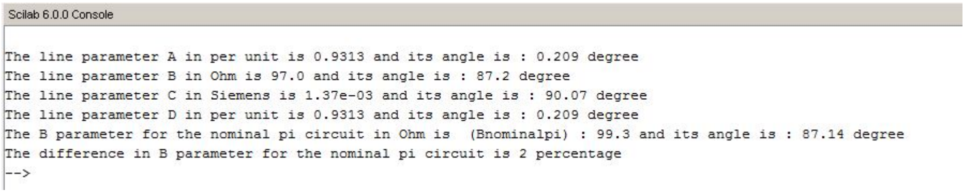
```
 The magnitude of Transmission line parameter A in per unit is 0.9706 and its angle is 0.159 degree
The magnitude of Transmission line parameter B in Ohm is 70.29 and its angle is 84.78 degree
The magnitude of Transmission line parameter C in Siemens is 8.28e-04 and its angle is 90.08 degree
The magnitude of Transmission line parameter A in per unit is 0.9706 and its angle is 0.159 degree

Sending end Line to Neutral Voltage in kVLN is : 199.7 and its angle is : 26.13 degree
Sending end Line to Line Voltage is  (VsLL) = 345.8  kV
The magnitude of sending end current in kA is  (Is) : 1.241 and its angle is : 15.44 degree
Power delivered to the sending end is  (Ps) = 730.4  MW

No load receiving end voltage is  (VrNL) = 356.3  kVLL
Full load voltage is  (Percent VR) = 8.7  Percent

Approximate Current carrying capacity of 2 ACSR conductors is  (J) = 1.8  kA
Full load line losses is  (P) = 30.4  MW
Full load transmission efficiency is  (Percent EFF) = 95.8  Percent
-->
```

Figure 5.1: ABCD parameters and the nominal pi circuit medium length line

```
// Authors - J. Duncan Glover, Mulukutla S. Sharma, Thomas J. Overbye
// Chapter - 5 : Example 5.1
// Scilab Version 6.0.0 : OS - Windows

clc;
clear;

f = 60;                                   // Frequency in Hz
N = 2;                                    // Number of Conductors
V = 345;                                  // Voltage in kV
L = 200;                                  // Line length
S = 795000;                               // Size of the conductor
z = 0.032+(%i*0.35);                      // Impedance in Ohm/km
y = (%i)*(4.2*10^-6);                     // Admitance in S/km
Pr = 700;                                 // Full load Power in MW
pf = 0.99;                                // Power factor
v = 95/100;                               // rated voltage
```

```

Z = z*L;
    // Total series impedance
Y = y*L;
    // Total shunt Admitance
A = 1 + ((Y*Z)/2);
    // Line Paramater A in per unit
D = A;
    // Line Paramater D in per unit
B = Z;
    // Line Paramater B in Ohm
C = Y*(1+(Y*Z)/4);
    // Line Paramater C in Siemens

VrLL = V*v;
    // Receiving end Line to Line Voltage in kVLL
VrLN = VrLL/sqrt(3);
    // Receiving end Line to Neutral Voltage in kVLN
theta = acos(pf);
Ir = (((Pr)*exp(%i*theta))/(sqrt(3)*(v*V)*(pf)));
    // Receiving end current in kA
VsLN = ((A*VrLN)+(B*Ir));
    // Sending end Line to Neutral Voltage in kVLN
VsLL = abs(VsLN)*sqrt(3);
    // sending end Line to Line Voltage in kVLL
Is = ((C*VrLN)+(D*Ir));
    // sending end current in kA
```

```
[r theta1] = polar(VsLN);
[r theta2] = polar(Is);
Ps = sqrt(3)*abs(VsLL)*abs(Is)*cos(theta1-theta2);           // Power delivered to the sending end in MW

VrNL = abs(VsLL)/abs(A);                                      // No load receiving end voltage in kVLL
PercentVR = ((abs(VrNL)-abs(VrLL))/abs(VrLL))*100;           // Full load voltage in percent

J = N*0.9;                                                    // Approximate Current carrying capacity of 2 ACSR conductors in kA taken From table A.4
P = Ps-Pr;                                                    // Full load line losses in MW
PercentEFF = (Pr/Ps)*100;                                     // Full load transmission efficiency in percent

printf('The magnitude of Transmission line parameter A in per unit is %0.4f and its angle is %0.3f degree', abs(A), atand(imag(A), real(A)));
printf('\nThe magnitude of Transmission line parameter B in Ohm is %0.2f and its angle is %0.2f degree', abs(B), atand(imag(B), real(B)));
printf('\nThe magnitude of Transmission line parameter C in Siemens is %0.2e and its angle is %0.2f degree', abs(C), atand(imag(C), real(C)));
printf('\nThe magnitude of Transmission line parameter D in per unit is %0.4f and its angle is %0.3f degree', abs(D), atand(imag(D), real(D)));

printf('\n\nSending end Line to Neutral Voltage in kVLN is : %0.1f and its angle is : %0.2f degree',
```



```
Scilab 6.0.0 Console

The line parameter A in per unit is 0.9313 and its angle is : 0.209 degree
The line parameter B in Ohm is 97.0 and its angle is : 87.2 degree
The line parameter C in Siemens is 1.37e-03 and its angle is : 90.07 degree
The line parameter D in per unit is 0.9313 and its angle is : 0.209 degree
The B parameter for the nominal pi circuit in Ohm is  (Bnominalpi) : 99.3 and its angle is : 87.14 degree
The difference in B parameter for the nominal pi circuit is 2 percentage
-->
```

Figure 5.2: Exact ABCD parameters long line

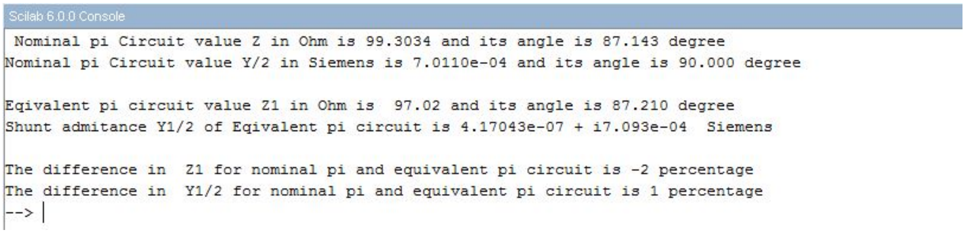
```
    abs(VsLN), atand(imag(VsLN), real(VsLN)));
printf('\nSending end Line to Line Voltage is  (VsLL) = %0.1f  kV', abs(VsLL));
printf('\nThe magnitude of sending end current in kA is  (Is) : %0.3f and its angle is : %0.2f degree', abs(Is), atand(imag(Is), real(Is)));
printf('\nPower delivered to the sending end is  (Ps) = %0.1f  MW', Ps);

printf('\n\nNo load receiving end voltage is  (VrNL) = %0.1f  kVLL', VrNL);
printf('\nFull load voltage is  (Percent VR) = %0.1f  Percent', PercentVR);

printf('\n\nApproximate Current carrying capacity of 2 ACSR conductors is  (J) = %0.1f  kA', J);
printf('\nFull load line losses is  (P) = %0.1f  MW', P);
printf('\nFull load transmission efficiency is  (Percent EFF) = %0.1f  Percent', PercentEFF);
```

---

**Scilab code Exa 5.2** `Exact ABCD parameters long line`

```
// Book - Power System: Analysis & Design 5th Edition
```

```
// Authors - J. Duncan Glover, Mulukutla S. Sharma, Thomas J. Overbye
// Chapter - 5 : Example 5.2
// Scilab Version 6.0.0 : OS - Windows

clc;
clear;

V = 765;                                        // Line voltage in kV
f = 60;                                         // frequency in Hz
L = 300;                                        // line length in km
z = 0.0165+(%i*0.3306);                         // Positive sequence impedance in Ohm/km
y = %i*4.674e-6;                                // Positive sequence admitance in S/km
Zc = sqrt(z/y);                                 // Characteristic impedance in Ohm
GammaL = sqrt(z*y)*L;                           // Propagation constant in per unit
eGammaL = exp(0.00930)*exp(%i*0.3730);
eNegGammaL = exp(-0.00930)*exp(-%i*0.3730);
coshGammaL = (eGammaL+eNegGammaL)/2;           // Hyperbolic function
sinhGammaL = (eGammaL-eNegGammaL)/2;           // Hyperbolic function
A = cosh(GammaL);                              // line parameter in per unit
D = A;                                         // line parameter in per unit
B = Zc*sinh(GammaL);                           // line parameter in Ohm
C = (1/Zc)*sinh(GammaL);                       // Line parameter in S
Bnominalpi = z*L;                              // The B parameter for the nominal pi circuit in Ohm
```



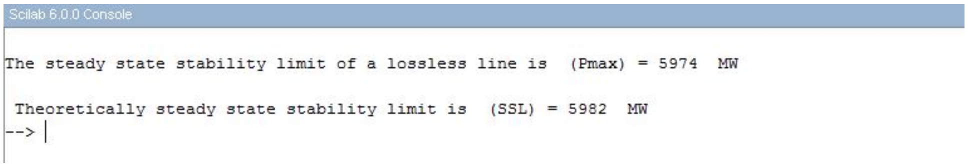
```
 Nominal pi Circuit value Z in Ohm is 99.3034 and its angle is 87.143 degree
Nominal pi Circuit value Y/2 in Siemens is 7.0110e-04 and its angle is 90.000 degree

Eqivalent pi circuit value Z1 in Ohm is  97.02 and its angle is 87.210 degree
Shunt admitance Y1/2 of Eqivalent pi circuit is 4.17043e-07 + i7.093e-04  Siemens

The difference in  Z1 for nominal pi and equivalent pi circuit is -2 percentage
The difference in  Y1/2 for nominal pi and equivalent pi circuit is 1 percentage
-->
```

Figure 5.3: Equivalent pi circuit long line

```
Bdiff=100-(abs(B)/abs(Bnominalpi))*100;       //The difference in B parameter in percentage

printf('\n\The line parameter A in per unit is %0.4f and its angle is : %0.3f degree', abs(A), atand(imag(A), real(A)));
printf('\nThe line parameter B in Ohm is %0.1f and its angle is : %0.1f degree', abs(B), atand(imag(B), real(B)));
printf('\nThe line parameter C in Siemens is %0.2e and its angle is : %0.2f degree', abs(C), atand(imag(C), real(C)));
printf('\n\The line parameter D in per unit is %0.4f and its angle is : %0.3f degree', abs(A), atand(imag(A), real(A)));
printf('\nThe B parameter for the nominal pi circuit in Ohm is  (Bnominalpi) : %0.1f and its angle is : %0.2f degree', abs(Bnominalpi), atand(imag(Bnominalpi), real(Bnominalpi)));
printf('\nThe difference in B parameter for the nominal pi circuit is %d percentage',Bdiff)
```

**Scilab code Exa 5.3** `Equivalent pi circuit long line`

```
// Book - Power System: Analysis & Design 5th Edition
// Authors - J. Duncan Glover, Mulukutla S. Sharma, Thomas J. Overbye
// Chapter - 5 : Example 5.3
// Scilab Version 6.0.0 : OS - Windows

clc;
clear;

z = 0.0165+(%i*0.3306);          // Positive sequence impedance in Ohm/km
y = %i*4.674*10^-6;              // Positive sequence admitance in S/km
L = 300;                         // line length in km

Z = z*L;                         // Circuit Impedance in Ohm
Y = (y/2)*L;                     // Circuit admitance in Siemens
GammaL = sqrt(z*y)*L;            // Propagation constant in per unit
F1 = sinh(GammaL)/(GammaL);      // Correction factor in per unit
F2 = tanh(GammaL/2)/(GammaL/2);  // Correction factor in per unit
Z1 = Z*F1;                       // Eqivalent pi circuit value in Ohm
Y1 = (Y)*(F2);                   // Shunt admitance of a Eqivalent pi circuit in Siemens
Zc=100-(abs(Z)*100/abs(Z1))      //Difference in Z for nominal and equivalent pi circuits
Yc=100-(abs(Y)*100/abs(Y1))      //Difference in Y/2 for nominal and equivalent pi circuits

printf('Nominal pi Circuit value Z in Ohm is %0.4f and its angle is %0.3f degree', abs(Z), atand(
```



```
The steady state stability limit of a lossless line is  (Pmax) = 5974  MW

 Theoretically steady state stability limit is  (SSL) = 5982  MW
-->
```

Figure 5.4: Theoretical steady state stability limit long line

```
imag(Z), real(Z)));
printf('\nNominal pi Circuit value Y/2 in Siemens is %0.4e and its angle is %0.3f degree', abs(Y), atand(imag(Y), real(Y)));
printf('\n\nEqivalent pi circuit value Z1 in Ohm is %0.2f and its angle is %0.3f degree', abs(Z1), atand(imag(Z1), real(Z1)));
printf('\nShunt admitance Y1/2 of Eqivalent pi circuit is %0.5e + i%0.3e  Siemens', real(Y1), imag(Y1));
printf('\n\nThe difference in  Z1 for nominal pi and equivalent pi circuit is %d percentage',Zc)
printf('\nThe difference in  Y1/2 for nominal pi and equivalent pi circuit is %d percentage',Yc)
```

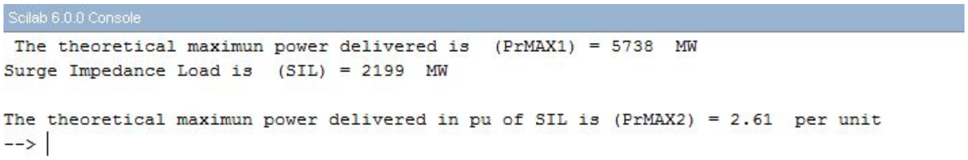
**Scilab code Exa 5.4** `Theoretical steady state stability limit long line`

```
// Book - Power System: Analysis & Design 5th Edition
// Authors - J. Duncan Glover, Mulukutla S. Sharma, Thomas J. Overbye
// Chapter - 5 : Example 5.4
// Scilab Version 6.0.0 : OS - Windows

clc;
clear;
```

```

L = 300;                                  // Line length in km
SI = 266.1;                               // Surge impedance in Ohm
lambda = 5000;                            // Wavelength in km
Vs = 765;                                 // Sending end voltage in kV
Vr = Vs;                                  // Receiving end voltage in kV

SIL = (Vs)^2/SI;                          // Surge impedance load in MW
Vspu = (765/765);                         // Sending end voltage in per unit
Vrpu = (765/765);                         // Receiving end voltage in per unit
Pmax = Vspu*Vrpu*SIL/sin(2*%pi*L/lambda);  // The theoretical steady state stability limit of a lossless line in MW
SSL = 2.72*SIL;                           // Theoretically steady state stability limit in MW; taken from Figure 5.12

printf('\nThe steady state stability limit of a lossless line is  (Pmax) = %0.0f  MW', Pmax);
printf('\n\n Theoretically steady state stability limit is  (SSL) = %0.0f  MW', SSL);
```

---

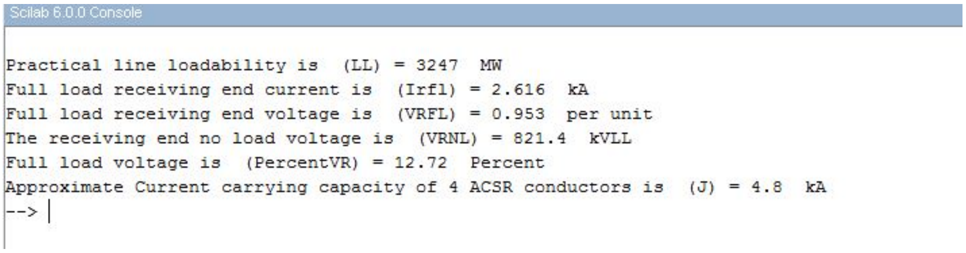
**Scilab code Exa 5.5** `Theoretical maximum power delivered long line`



```
 The theoretical maximun power delivered is  (PrMAX1) = 5738  MW
Surge Impedance Load is  (SIL) = 2199  MW

The theoretical maximun power delivered in pu of SIL is (PrMAX2) = 2.61  per unit
-->
```

Figure 5.5: Theoretical maximum power delivered long line

```
// Book - Power System: Analysis & Design 5th Edition
// Authors - J. Duncan Glover, Mulukutla S. Sharma, Thomas J. Overbye
// Chapter - 5 : Example 5.5
// Scilab Version 6.0.0 : OS - Windows

clc;
clear;

Vs = 765;                                                   // Sending end voltage in kV
Vr= Vs;                                                     // receiving end voltage is equal to sending end voltage

A = 0.9313;                                                 // Absolute line paramter A value in per unit from Ex 5.2
thetaA = 0.209*(%pi/180);                                   // angle value of Parameter A in degree from Ex 5.2
B = 97;                                                     // Absolute line paramter B value in Ohm from Ex 5.2
thetaZ = 87.2*(%pi/180);
```



```
Practical line loadability is  (LL) = 3247  MW
Full load receiving end current is  (Irfl) = 2.616  kA
Full load receiving end voltage is  (VRFL) = 0.953  per unit
The receiving end no load voltage is  (VRNL) = 821.4  kVLL
Full load voltage is  (PercentVR) = 12.72  Percent
Approximate Current carrying capacity of 4 ACSR conductors is  (J) = 4.8  kA
-->
```

Figure 5.6: Practical line loadability and percent voltage regulation long line

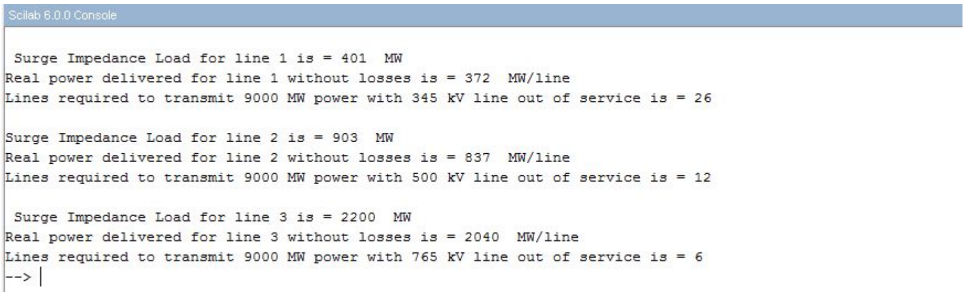
```
                                                    // angle value of Parameter B in degree from Ex 5.2
Z1 = B;
Zc = 266.1;
                                                    // The magnitude of Characteristic impedance in ohm from Ex 5.2
PrMAX1 = ((Vr*Vs)/Z1)-(((A*Vr^2)/Z1)*(cos(thetaZ-thetaA)));    // The theoretical maximum real power delivered in MW
SIL = (Vr)^2/Zc;
                                                    // Surge Impedance Load in MW
PrMAX2 = PrMAX1/SIL;
                                                    // The theoretical maximum real power delivered in per unit

printf('The theoretical maximun power delivered is (PrMAX1) = %d  MW', PrMAX1);
printf('\nSurge Impedance Load is  (SIL) = %d  MW', SIL);
printf('\n\nThe theoretical maximun power delivered in pu of SIL is (PrMAX2) = %0.2f  per unit', PrMAX2);
```

**Scilab code Exa 5.6** `Practical line loadability and percent voltage regulation long`

```
// Book - Power System: Analysis & Design 5th
   Edition
// Authors - J. Duncan Glover, Mulukutla S. Sharma,
   Thomas J. Overbye
// Chapter - 5 : Example 5.6
// Scilab Version 6.0.0 : OS - Windows

clc;
clear;

N = 4;
   // Number of Conductors
l = 300;
   // Line lenght in km
s = 1272000;
   // Size of the conductor in cmil
Vs1 = 765;
   // Sending end voltage in kV
V = 765;
   // Base Voltage
Vr1 = 0.95*765;
   // Receiving end voltage in kV

delta = 35;
   // Phase angle in degree
```

```
Z1 = 97;                         // Absolute line paramter value in Ohm from Ex 5.5
thetaZ = 87.2;                   // Angle value of Parameter B in radians from Ex 5.5
A = 0.9313;                      // Absolute line paramter value in per unit from Ex 5.5
thetaA = 0.209;                  // Angle value of Parameter A in degree from Ex 5.5
Pr = (((Vr1*Vs1)/Z1)*cosd(thetaZ-delta))-(((A*(Vr1)^2)/Z1)*cosd(thetaZ-thetaA));     // The real power delivered to the receiving end in MW
SIL = 2199; // Surge Impedance Load in MW taken from Ex 5.5
L = 1.49;                        // Loadability in per unit of SIL  taken from fig 5.12
LL = L*SIL;                      // Practical Line loadability in MW using fig. 5.12

pf = 0.986;                      // Power factor
IRFL = Pr/(sqrt(3)*Vr1*pf);      // Full load receiving end current in kA

A = 0.9313*exp(%i*0.209)*(%pi/180);
```

```
    // Line parameter value in per unit taken from Ex 5.2
B = 97.0*exp(%i*87.20)*(%pi/180);

    // Line parameter value in Ohm taken from Ex 5
theta = acos(pf);
Irfl = 2.616*exp(%i*theta);
Vs2 = Vs1/sqrt(3);

    // line Voltage in kV
a = 0.8673;

    // coefficient of second order Vrfl from the equation in part c
b = -54.24;

    // coefficient Vrfl from the equation in part c
c = -130707.89;

    // coefficient constant from the equation in part c
Vrfl = (-b+sqrt((b^2)-(4*a*c)))/(2*a);
                                                        //
    Vrfl value from the 2nd order Quadratic equation
Vrfl2 = Vrfl*sqrt(3);

    // Full load receiving end voltage in kVLL
VRFL = Vrfl2/V;

    // Full load receiving end in per unit

absA = 0.9313;

    // Absolute value of A taken from Ex 5.2
VRNL = V/absA;

    // The receiving end no load voltage in kVLL
```



```
 Surge Impedance Load for line 1 is = 401  MW
Real power delivered for line 1 without losses is = 372  MW/line
Lines required to transmit 9000 MW power with 345 kV line out of service is = 26

Surge Impedance Load for line 2 is = 903  MW
Real power delivered for line 2 without losses is = 837  MW/line
Lines required to transmit 9000 MW power with 500 kV line out of service is = 12

 Surge Impedance Load for line 3 is = 2200  MW
Real power delivered for line 3 without losses is = 2040  MW/line
Lines required to transmit 9000 MW power with 765 kV line out of service is = 6
-->
```

Figure 5.7: Selection of transmission line voltage and number of lines for power transfer

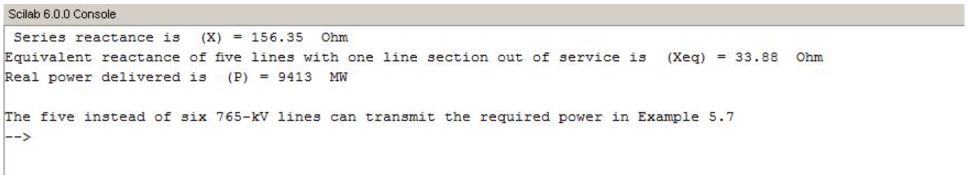
```
    taken from 5.1.19
PercentVR = ((VRNL-Vrfl2)/Vrfl2)*100;                                    //
    Full load voltage in percent

J = N*1.2;
    // Approximate Current carrying capacity of 4
    ACSR conductors in kA taken From table A.4

printf('\nPractical line loadability is  (LL) = %0.0
    f  MW', Pr);
printf('\nFull load receiving end current is  (Irfl)
     = %0.3f  kA', IRFL);
printf('\nFull load receiving end voltage is  (VRFL)
     = %0.3f  per unit', VRFL);
printf('\nThe receiving end no load voltage is  (
    VRNL) = %0.1f  kVLL', VRNL);
printf('\nFull load voltage is  (PercentVR) = %0.2f
     Percent', PercentVR);
printf('\nApproximate Current carrying capacity of 4
     ACSR conductors is  (J) = %0.1f  kA', J);
```

**Scilab code Exa 5.7** `Selection of transmission line voltage and number of lines for`

```
// Book - Power System: Analysis & Design 5th Edition
// Authors - J. Duncan Glover, Mulukutla S. Sharma, Thomas J. Overbye
// Chapter - 5 : Example 5.7
// Scilab Version 6.0.0 : OS - Windows

clc;
clear;

p = 9000;                                                   // Power in MW
l = 500;                                                    // Load center distance from the plant in km
f = 60;                                                     // Frequency in Hz
Vs = 1.0;                                                   // Sending end voltage in per unit
Vr = 0.95;                                                  // Receiving end voltage in per unit
delta = 35*(%pi/180);                                       // Phase angle in degree
lamdba = 5000;                                              // Wavelength in km

```

```
v1 = 345;                                   // 1st line voltage in kV
Zc1 = 297;                                  // Characteristic impedance of 1st line in Ohm
SIL1 = (v1^2)/Zc1;                          // Surge Impedance Load for line 1 in MW
P1 = (Vs*Vr*SIL1*sin(delta))/(sin((2*%pi*l)/lamdba));      // Real power delivered for line 1 without losses in MW/line
line1 = ceil((p/P1)+1);                     // Lines required to transmit 9000 MW power with 345 kV line out of service

v2 = 500;                                   // 2nd line voltage in kV
Zc2 = 277;                                  // Characteristic impedance of 2nd line in Ohm
SIL2 = (v2^2)/Zc2;                          // Surge Impedance Load for line 2 in MW
P2 = (Vs*Vr*SIL2*sin(delta))/(sin((2*%pi*l)/lamdba));      // Real power delivered for line 2 without losses in MW/line
line2 = ceil((p/P2)+1);                     // Lines required to transmit 9000 MW power with 500 kV line out of service

v3 = 765;                                   // 3rd line voltage in kV
Zc3 = 266;
```

```
// Characteristic impedance of 3rd line in Ohm
SIL3 = (v3^2)/Zc3;                    // Surge Impedance Load for line 3 in MW
P3 = (Vs*Vr*SIL3*sin(delta))/(sin((2*%pi*l)/lamdba));       // Real power delivered for line 3 without losses in MW/line
line3 = ceil((p/P3)+1);                    // Lines required to transmit 9000 MW power with 765 kV line out of service

printf('\n Surge Impedance Load for line 1 is = %0.0f  MW', SIL1);
printf('\nReal power delivered for line 1 without losses is = %0.0f  MW/line', P1);
printf('\nLines required to transmit 9000 MW power with 345 kV line out of service is = %0.0f', line1);

printf('\n\nSurge Impedance Load for line 2 is = %0.0f  MW', SIL2);
printf('\nReal power delivered for line 2 without losses is = %0.0f  MW/line', P2);
printf('\nLines required to transmit 9000 MW power with 500 kV line out of service is = %0.0f', line2);

printf('\n\n Surge Impedance Load for line 3 is = %0.0f  MW', SIL3);
printf('\nReal power delivered for line 3 without losses is = %0.0f  MW/line', P3);
printf('\nLines required to transmit 9000 MW power with 765 kV line out of service is = %0.0f', line3);
```



```
 Series reactance is  (X) = 156.35  Ohm
Equivalent reactance of five lines with one line section out of service is  (Xeq) = 33.88  Ohm
Real power delivered is  (P) = 9413  MW

The five instead of six 765-kV lines can transmit the required power in Example 5.7
-->
```

Figure 5.8: Effect of intermediate substations on number of lines required for power transfer

**Scilab code Exa 5.8** `Effect of intermediate substations on number of lines required`

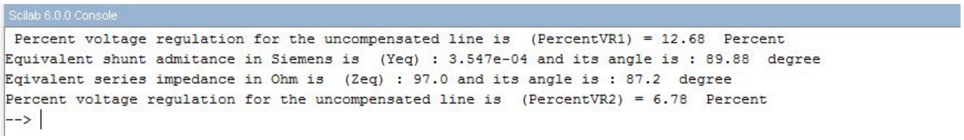
```
// Book - Power System: Analysis & Design 5th Edition
// Authors - J. Duncan Glover, Mulukutla S. Sharma, Thomas J. Overbye
// Chapter - 5 : Example 5.8
// Scilab Version 6.0.0 : OS - Windows

clc;
clear;

N = 6;                                          // Number of transmission lines
Vs = 765;                                       // Transmission voltage in kV
l = 167;                                        // Intermediate substations distance in km
Pl=9000;                                        //Load power , value taken from Example 5.7
lambda = 5000;                                  // Wavelength in km
Beta = (2*%pi)/lambda;                          // Taken from Eq 5.4.15
```

```
L = 500;                                   // Eqivalent pi circuit lenght in km
Zc = 266;                                  // Characteristic impedance of the line in Ohm
X = Zc*sin(Beta*L);                        // Series reactance in Ohm ; taken from Eq 5.4.10
Xeq = ((1/5)*((2/3)*X))+((1/4)*(X/3));     // Equivalent reactance of five lines with one line section out of service in Ohm
Vr = 0.95*765;                             // Receiving end voltage in kV
delta = 35;                                // Phase angle in degree
P = ((Vs*Vr)/Xeq)*sind(delta);             // Real power delivered in MW ; taken from Eq 5.4.26

printf('Series reactance is (X) = %0.2f Ohm', X);
printf('\nEquivalent reactance of five lines with one line section out of service is (Xeq) = %0.2f Ohm', Xeq);
printf('\nReal power delivered is (P) = %0.0f MW', P);

if 0.97*P>Pl            //Assuming 3% as losses
    printf('\n\nThe five instead of six 765-kV lines can transmit the required power in Example 5.7')
end
```

---

**Scilab code Exa 5.9** `Shunt reactive compensation to improve transmission line volta`

```
// Book - Power System: Analysis & Design 5th Edition
```

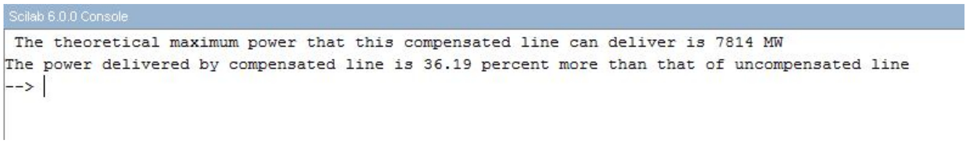


```
 Percent voltage regulation for the uncompensated line is  (PercentVR1) = 12.68  Percent
Equivalent shunt admitance in Siemens is  (Yeq) : 3.547e-04 and its angle is : 89.88  degree
Eqivalent series impedance in Ohm is  (Zeq) : 97.0 and its angle is : 87.2  degree
Percent voltage regulation for the uncompensated line is  (PercentVR2) = 6.78  Percent
--> |
```

Figure 5.9: Shunt reactive compensation to improve transmission line voltage regulation

```
// Authors - J. Duncan Glover, Mulukutla S. Sharma, Thomas J. Overbye
// Chapter - 5 : Example 5.9
// Scilab Version 6.0.0 : OS - Windows

clc;
clear;

l = 300;                                        // line lenght in km
If = 1.90;                                      // Full load current in kA
pf = 1;                                         // Power Factor
VF = 730;                                       // Voltage in kV
V = 730/sqrt(3);                                // Line voltage in kV

Irfl = 1.9*exp(%i*0*%pi/180);                   // Full load receiving end current in kA
Vrfl = V*exp(%i*0*%pi/180);                     // Full load receiving end voltage in kV
A = 0.9313*exp(%i*0.209*%pi/180);               // Line parameter value in per unit ; taken from Ex 5.2
B = 97.0*exp(%i*87.20*%pi/180);                 // Line parameter value in Ohm ; taken from Ex 5.2
VsLN = (A*Vrfl)+(B*Irfl);
VsLL = abs(VsLN)*sqrt(3);                       // Sending end voltage in kVLN
```

```
Vrnl = VsLL/abs(A);                          // No load Receiving end Voltage in kVLL
PercentVR1 = ((Vrnl - VF)/VF)*100;           // Percent voltage regulation for the uncompensated line in Percent

Y = 2*(3.7*10^-7+%i*7.094*10^-4);            // Shunt admitance of a Eqivalent pi circuit in Siemens ; taken from Ex 5.3
Yeq = real(Y)+%i*imag(Y)*(1-(75/100));       // Equivalent shunt admitance in Siemens
Zeq = B;                                     // Eqivalent series impedance in Ohm

Aeq = 1+((Yeq*Zeq)/2);                       // The eqivalent A parameter for the compensated line in per unit
VRNL = VsLL/abs(Aeq);                        // No load Receiving end Voltage in kVLL
PercentVR2 = ((VRNL - VF)/VF)*100;           // Percent voltage regulation for the uncompensated line in Percent


printf('Percent voltage regulation for the uncompensated line is  (PercentVR1) = %0.2f Percent', PercentVR1)
printf('\nEquivalent shunt admitance in Siemens is (Yeq) : %0.3e and its angle is : %0.2f  degree', abs(Yeq), atand(imag(Yeq), real(Yeq)));
printf('\nEqivalent series impedance in Ohm is  (Zeq) : %0.1f and its angle is : %0.1f  degree', abs(Zeq), atand(imag(Zeq), real(Zeq)));
printf('\nPercent voltage regulation for the uncompensated line is  (PercentVR2) = %0.2f Percent', PercentVR2)
```



```
 The theoretical maximum power that this compensated line can deliver is 7814 MW
The power delivered by compensated line is 36.19 percent more than that of uncompensated line
--> |
```

Figure 5.10: Series capacitive compensation to increase transmission line loadability

**Scilab code Exa 5.10** `Series capacitive compensation to increase transmission line`

```
// Book - Power System: Analysis & Design 5th Edition
// Authors - J. Duncan Glover, Mulukutla S. Sharma, Thomas J. Overbye
// Chapter - 5 : Example 5.10
// Scilab Version 6.0.0 : OS - Windows

clc;
clear;

Comp = 30/100;                          // Compensation in percent
Vs = 765;                               // Sending end voltage in kV
Vr = Vs;                                // Receiving end voltage in kV
Z = 97.02;                              // Absolute eqivalent pi circuit value ; Taken from Ex 5.3
PRmaxun=5738                            // Maximum power that can be delivered by uncompensated line(From example 5.5)
F1 = sind(87.210);                      // Eqivalent pi circuit angle ; Taken from Ex 5.3
```

```
X1 = Z*F1;                              // Eqivalent series reactance without compensation in Ohm ; taken from Ex 5.3
Zcap = -(%i)*(1/2)*Comp*X1;             // Impedance of series capacitor in Ohm
ABCD = [1 Zcap; 0 1];                   // From figure 5.4 for series impedance the ABCD matrix
ABCD2 = [ 0.9313*exp(%i*0.209*%pi/180) 97.0*exp(%i*87.2*%pi/180);
         1.37*10^(-3)*exp(%i*90.06*%pi/180) 0.9313*exp(%i*0.209*%pi/180) ]; // The ABCD parameters taken from Ex 5.2
ABCDeq = ABCD*ABCD2*ABCD;                                                   // The eqivalent ABCD matrix of the compensated line
Aeq = abs(ABCDeq(1,1));                                                     // Absolute value of the line parameter A
thetaAeq = atand(imag(ABCDeq(1,1))/real(ABCDeq(1,1)));              // Angle value of the line parameter A
Beq = abs(ABCDeq(1,2));                                                     // Absolute value of the line parameter B
thetaBeq = atand(imag(ABCDeq(1,2))/real(ABCDeq(1,2)));              // Angle value of the line parameter B
PRmax=(Vs^2/Beq)-(Aeq*Vs^2/Beq)*cosd(thetaBeq-thetaAeq);            // maximum power that can be delivered
dif=(PRmax/PRmaxun)*100-100;                                                                    // Percentage difference in power delivered between compensated and uncompensated line
printf('The theoretical maximum power that this compensated line can deliver is %d MW',PRmax)
printf('\nThe power delivered by compensated line is
```

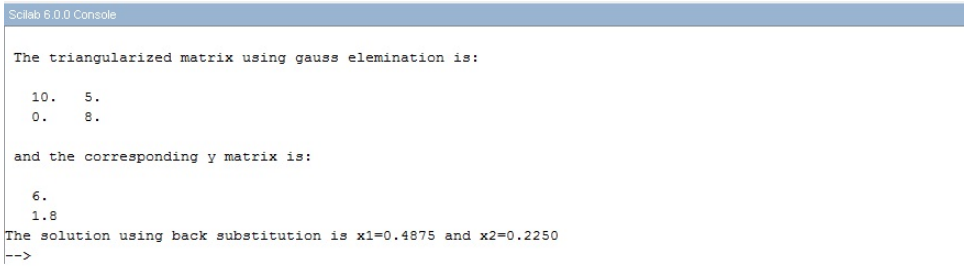
```
 %.2f percent more than that of uncompensated
line',dif)
```

# Chapter 6

# POWER FLOWS

**Scilab code Exa 6.1** `Gauss elimination and back substitution direct solution to lin`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 6 ; Example 6.1
//Scilab Version - 6.0.0 ; OS - Windows

clc;
```



```
 The triangularized matrix using gauss elemination is:

   10.   5.
   0.    8.

 and the corresponding y matrix is:

   6.
   1.8
The solution using back substitution is x1=0.4875 and x2=0.2250
-->
```

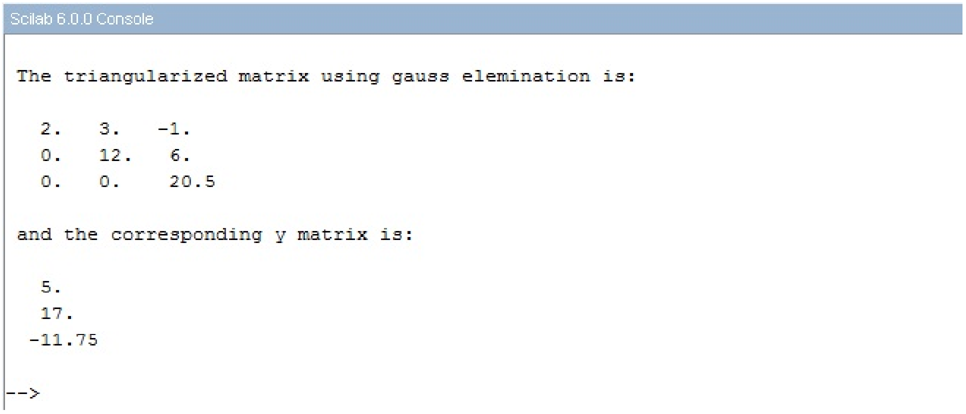
Figure 6.1: Gauss elimination and back substitution direct solution to linear algebraic equations

```
clear;

A=[10 5;2 9];
y=[6;3];
N=length(y);                  //Number of variables
st=N-1;                       //Number of Gauss elimination steps

//Gauss Elimination step:
B=A;
for i=1:st
        for j=i+1:N
            m=(B(j,i)/B(i,i));
            A(j,i+1:N)=A(j,i+1:N)-m*(A(i,i+1:N));
            A(i+1:N,i)=0;
            y(j)=y(j)-m*y(i);
        end
    B=A;
end

//Back Substitution step
x2=y(2)/A(2,2)
x1=(y(1)-A(1,2)*x2)/A(1,1);
disp(A,'The triangularized matrix using gauss elemination is:')
disp(y,'and the corresponding y matrix is:')
printf('The solution using back substitution is x1=%.4f and x2=%.4f',x1,x2)
```

---

**Scilab code Exa 6.2** `Gauss elimination triangularizing a matrix`

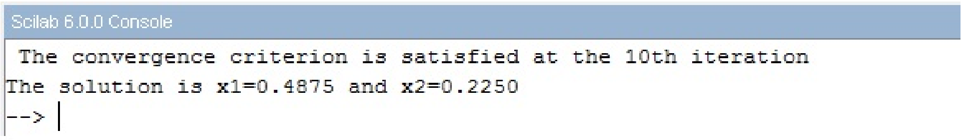
```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma,
```



```
The triangularized matrix using gauss elemination is:

  2.   3.   -1.
  0.   12.   6.
  0.   0.    20.5

and the corresponding y matrix is:

  5.
  17.
 -11.75

-->
```

Figure 6.2: Gauss elimination triangularizing a matrix

```
and Thomas J. Overbye
//Chapter - 6 ; Example 6.2
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;

A=[2 3 -1;-4 6 8;10 12 14];
y=[5;7;9];
N=length(y);                    //Number of variables
st=N-1;                         //Number of Gauss elimination steps

//Gauss Elimination step:
B=A;
for i=1:st
        for j=i+1:N
            m=(B(j,i)/B(i,i));
            A(j,i+1:N)=A(j,i+1:N)-m*(A(i,i+1:N));
            A(i+1:N,i)=0;
            y(j)=y(j)-m*y(i);
        end
    B=A;
end
```



```
 The convergence criterion is satisfied at the 10th iteration
The solution is x1=0.4875 and x2=0.2250
-->
```

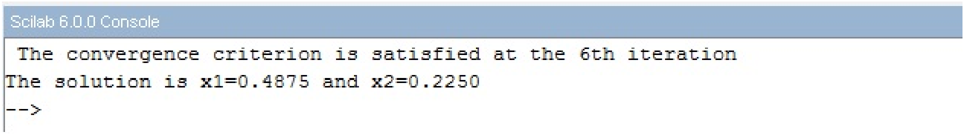
Figure 6.3: Jacobi method iterative solution to linear algebraic equations

```
disp(A,'The triangularized matrix using gauss elemination is:')
disp(y,'and the corresponding y matrix is:')
```

---

**Scilab code Exa 6.3** `Jacobi method iterative solution to linear algebraic equations`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 6 ; Example 6.3
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;

A=[10 5;2 9];              //Coefficients of variables in matrix form
y=[6;3];                   //Constant coefficients in matrix form
tol=1e-4;                  //Tolerance value
x=[0;0]

D=[A(1,1) 0;0 A(2,2)];        //Matrix containing the diagonal elements of A
M=inv(D)*(D-A);

```



```
 The convergence criterion is satisfied at the 6th iteration
The solution is x1=0.4875 and x2=0.2250
-->
```

Figure 6.4: Gauss Seidel method iterative solution to linear algebraic equations

```
err=1;
iter=0;

while err>tol
    temp=x;
    x=M*x+inv(D)*y;
    if temp(1) ~= 0 | temp(2) ~= 0
    err=max(abs((x(1)-temp(1))/temp(1)),abs((x(2)-temp(2))/temp(2)));
    end
    iter=iter+1;
end

printf('The convergence criterion is satisfied at the %dth iteration\n',iter)
printf('The solution is x1=%.4f and x2=%.4f',x(1),x(2))
```

---

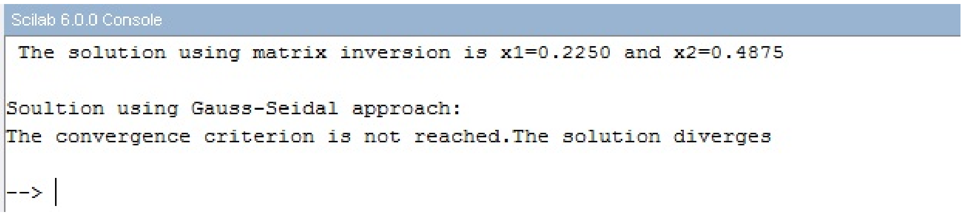
**Scilab code Exa 6.4** `Gauss Seidel method iterative solution to linear algebraic equ`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 6 ; Example 6.4
//Scilab Version - 6.0.0 ; OS - Windows

```

```
clc;
clear;

A=[10 5;2 9];           //Coefficients of variables in matrix form
y=[6;3];                //Constant coefficients in matrix form
tol=1e-4;               //Tolerance value
x=[0;0]

D=[A(1,1) 0;A(2,1) A(2,2)];     //Matrix containing the lower triangular elements of A
M=inv(D)*(D-A);

err=1;
iter=0;

while err>tol
    temp=x;
    x=M*x+inv(D)*y;
    if temp(1) ~=0 |temp(2)~=0
    err=max(abs((x(1)-temp(1))/temp(1)),abs((x(2)-temp(2))/temp(2)));
    end
    iter=iter+1;
end

printf('The convergence criterion is satisfied at the %dth iteration\n',iter)
printf('The solution is x1=%.4f and x2=%.4f',x(1),x(2))
```

---

**Scilab code Exa 6.5** `Divergence of Gauss Seidel method`



```
Scilab 6.0.0 Console
 The solution using matrix inversion is x1=0.2250 and x2=0.4875

Soultion using Gauss-Seidal approach:
The convergence criterion is not reached.The solution diverges

--> |
```

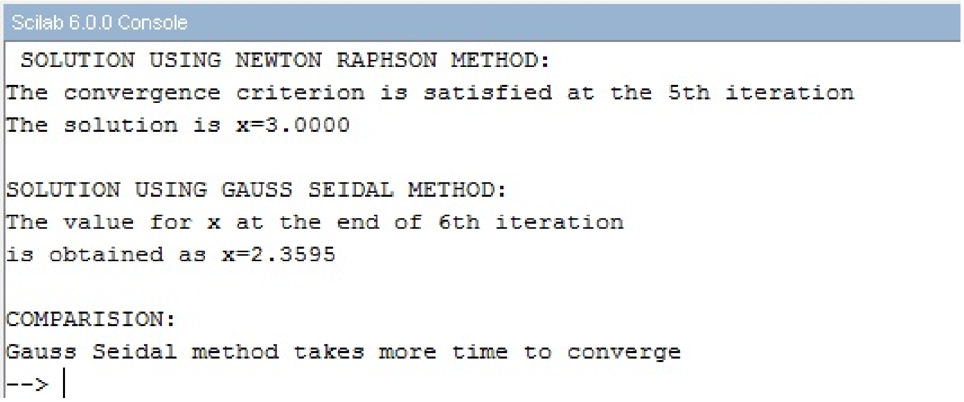
Figure 6.5: Divergence of Gauss Seidel method

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 6 ; Example 6.5
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;

A=[5 10;9 2];              //Coefficients of variables in matrix form
y=[6;3];                   //Constant coefficients in matrix form
tol=1e-4;                  //Tolerance value
x=[0;0]

//Solution by matrix inversion

xm=inv(A)*y;

//Solution using Gauss Seidel method

D=[A(1,1) 0;A(2,1) A(2,2)];       //Matrix containing the lower triangular elements of A
M=inv(D)*(D-A);

err=1;
iter=0;

```

```
while err>tol
    temp=x;
    x=M*x+inv(D)*y;
    if temp(1) ~=0 | temp(2) ~= 0
    err=max(abs((x(1)-temp(1))/temp(1)),abs((x(2)-temp(2))/temp(2)));
    end
    iter=iter+1;
end

printf('The solution using matrix inversion is x1=%.4f and x2=%.4f\n\n',xm(1),xm(2))
printf('Soultion using Gauss-Seidal approach:\n')
if isnan(err)
    printf('The convergence criterion is not reached.The solution diverges\n')
else
    printf('The convergence criterion is satisfied at the %dth iteration\n',iter)
    printf('The solution is x1=%.4f and x2=%.4f',x(1),x(2))
end
```

**Scilab code Exa 6.6** `Newton Raphson method solution to polynomial equations`

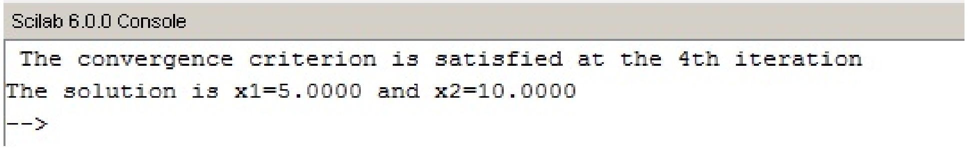
```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 6 ; Example 6.6
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;
```



```
 SOLUTION USING NEWTON RAPHSON METHOD:
The convergence criterion is satisfied at the 5th iteration
The solution is x=3.0000

SOLUTION USING GAUSS SEIDAL METHOD:
The value for x at the end of 6th iteration
is obtained as x=2.3595

COMPARISION:
Gauss Seidal method takes more time to converge
--> 
```

Figure 6.6: Newton Raphson method solution to polynomial equations

```

//Solution for x^2=9 using Newton Raphson method:
err=1;
iternr=0;                            //Initial iteration value for Newton Raphson method
tol=1e-4;                            //Tolerance value for Newton Raphson method
xn=1;                                //Initial value for x for Newton Raphson method

while err>tol
     temp=xn;
     J=2*xn;                          // Jacobian Matrix
     xn=xn+inv(J)*(9-xn^2);
     err=abs((xn-temp)/temp)
     iternr=iternr+1;
end

//Solution for x^2=9 using Gauss Seidel method
err=1;
D=3;
itergs=0;                           //Initial iteration value for Gauss Seidal method
xg=1;                               //Initial value for x for
```



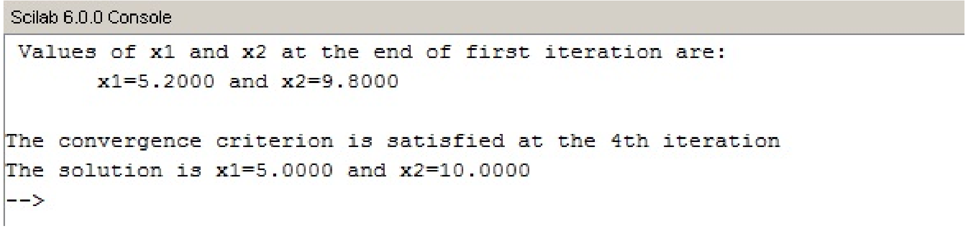
```
 The convergence criterion is satisfied at the 4th iteration
The solution is x1=5.0000 and x2=10.0000
-->
```

Figure 6.7: Newton Raphson method solution to nonlinear algebraic equations

```
    Gauss Seidal method

while err>tol & itergs<iternr+1
    temp=xg;
    xg=xg+inv(D)*(9-xg^2)
    err=abs((xg-temp)/temp)
    itergs=itergs+1
end
printf('SOLUTION USING NEWTON RAPHSON METHOD:\n')
printf('The convergence criterion is satisfied at the %dth iteration\n',iternr)
printf('The solution is x=%.4f\n\n',xn)

printf('SOLUTION USING GAUSS SEIDEL METHOD:\n')
printf('The value for x at the end of %dth iteration\n',itergs)
printf('is obtained as x=%.4f\n\n',xg)

printf('COMPARISON:\n')
if itergs>iternr
    printf('Gauss Seidel method takes more time to converge')
else
    printf('Newton Raphson method takes more time to converge')
end
```

**Scilab code Exa 6.7** `Newton Raphson method solution to nonlinear algebraic equation`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 6 ; Example 6.7
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;

c=[15;50];                          //Constant coefficients in the equations
x=[4;9];                            //Initial values for x1 and x2

err=1;                              //Initialization of error value
tol=1e-4;                           //Tolerance value for Newton Raphson method
iter=0;                             //Initialization of iteration value

while err>tol
    temp=x;
    f=[x(1)+x(2);x(1)*x(2)]                 //Function Value
    J=[1 1;x(2) x(1)];                  //Jacobian Matrix
    x=x+inv(J)*(c-f)
    err=max(abs((x(1)-temp(1))/temp(1)),abs((x(2)-temp(2))/temp(2)));
    iter=iter+1;
end
```



```
 Values of x1 and x2 at the end of first iteration are:
       x1=5.2000 and x2=9.8000

The convergence criterion is satisfied at the 4th iteration
The solution is x1=5.0000 and x2=10.0000
-->
```

Figure 6.8: Newton Raphson method in four steps

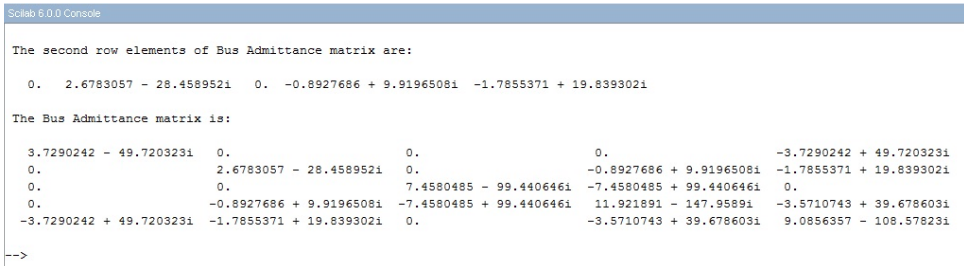
```
printf('The convergence criterion is satisfied at
    the %dth iteration\n',iter)
printf('The solution is x1=%.4f and x2=%.4f',x(1),x
    (2))
```

---

**Scilab code Exa 6.8** `Newton Raphson method in four steps`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma,
    and Thomas J. Overbye
//Chapter - 6 ; Example 6.8
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;

y=[15;50];                          //Constant
    coefficients in the equations
x=[4;9];                            //Initial values
    for x1 and x2

err=1;                              //Initialization of
     error value
tol=1e-4;                           //Tolerance value
```

```
for Newton Raphson method
iter=0;                               //Initialization of iteration value

while err>tol
    temp=x;
    f=[x(1)+x(2);x(1)*x(2)]                          //Function Value
    dely=y-f;
    J=[1 1;x(2) x(1)];                               //Jacobian Matrix
    //Reduction of Jacobian using Gauss elimination
    Jg=[J(1,1) J(1,2);0 J(2,2)-J(2,1)/J(1,1)]
    delyg=[dely(1);dely(2)-dely(1)*J(2,1)/J(1,1)]
    //Solution using back substitution
    delx2=delyg(2)/Jg(2,2);
    delx1=(delyg(1)-Jg(1,2)*delx2)/Jg(1,1)
    delx=[delx1;delx2]
    x=x+delx
    err=max(abs((x(1)-temp(1))/temp(1)),abs((x(2)-temp(2))/temp(2)));
    iter=iter+1;
    //Displaying first iteration results
    if iter==1
        printf('Values of x1 and x2 at the end of first iteration are:\n')
        printf('       x1=%.4f and x2=%.4f\n\n',x(1),x(2))
    end
end
printf('The convergence criterion is satisfied at the %dth iteration\n',iter)
printf('The solution is x1=%.4f and x2=%.4f',x(1),x(2))
```

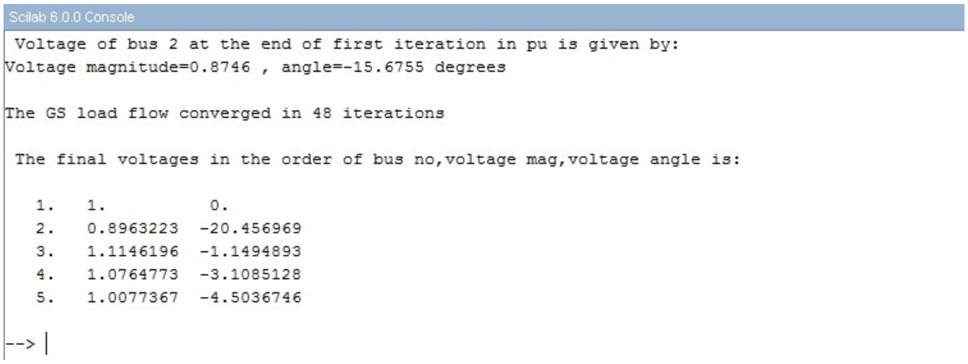


```
The second row elements of Bus Admittance matrix are:

  0.   2.6783057 - 28.458952i   0.  -0.8927686 + 9.9196508i  -1.7855371 + 19.839302i

The Bus Admittance matrix is:

  3.7290242 - 49.720323i   0.                        0.                        0.                       -3.7290242 + 49.720323i
  0.                       2.6783057 - 28.458952i    0.                       -0.8927686 + 9.9196508i  -1.7855371 + 19.839302i
  0.                       0.                        7.4580485 - 99.440646i  -7.4580485 + 99.440646i    0.
  0.                      -0.8927686 + 9.9196508i  -7.4580485 + 99.440646i   11.921891 - 147.9589i    -3.5710743 + 39.678603i
 -3.7290242 + 49.720323i  -1.7855371 + 19.839302i   0.                       -3.5710743 + 39.678603i   9.0856357 - 108.57823i

-->
```

Figure 6.9: Power flow input data and Ybus

**Scilab code Exa 6.9** `Power flow input data and Ybus`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 6 ; Example 6.8
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;

linedata=[2 4 0.0090   0.10   1.72       //Entering line data from table 6.2 & 6.3
          2 5 0.0045   0.05   0.88
          4 5 0.00225  0.025  0.44
          1 5 0.00150  0.02   0.00
          3 4 0.00075  0.01   0.00];

sb= linedata(:,1);
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
```



```
 Voltage of bus 2 at the end of first iteration in pu is given by:
Voltage magnitude=0.8746 , angle=-15.6755 degrees

The GS load flow converged in 48 iterations

 The final voltages in the order of bus no,voltage mag,voltage angle is:

   1.   1.          0.
   2.   0.8963223  -20.456969
   3.   1.1146196  -1.1494893
   4.   1.0764773  -3.1085128
   5.   1.0077367  -4.5036746

-->
```

Figure 6.10: Power flow solution by Gauss Seidel

```
    =R+jX
sa=linedata(:,5)*%i;                    //shunt admittance=jB since conductsnce G=0 for all lines
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)/2;
    ybus(n,n)=ybus(n,n)+1/lz(i)+sa(i)/2;
    ybus(m,n)=-1/lz(i);
    ybus(n,m)=ybus(m,n);
end
disp(ybus(2,:),'The second row elements of Bus Admittance matrix are:')
disp(ybus,'The Bus Admittance matrix is:')
```

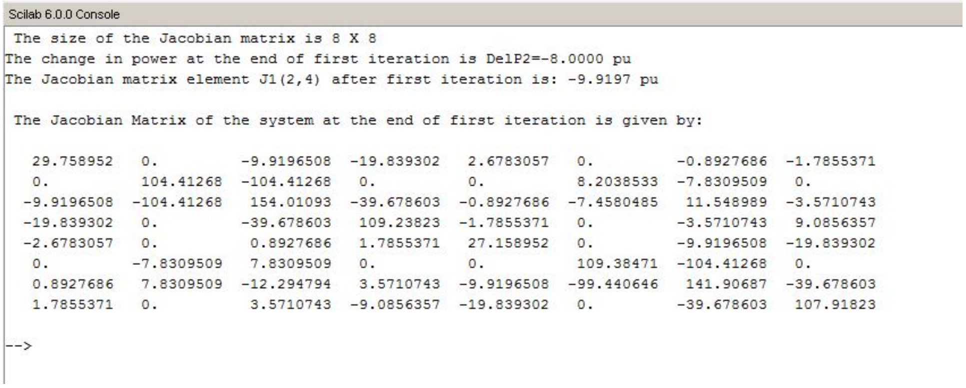
**Scilab code Exa 6.10** Power flow solution by Gauss Seidel

```
//Book - Power System: Analysis & Design 5th Edition
```

```
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 6 ; Example 6.10
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;

linedata=[2 4 0.0090   0.10  1.72        //Entering line data from table 6.2 & 6.3
          2 5 0.0045   0.05  0.88
          4 5 0.00225  0.025 0.44
          1 5 0.00150  0.02  0.00
          3 4 0.00075  0.01  0.00];

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 since conductsnce G=0 for all lines
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)/2;
    ybus(n,n)=ybus(n,n)+1/lz(i)+sa(i)/2;
    ybus(m,n)=-1/lz(i);
    ybus(n,m)=ybus(m,n);
end
y=ybus;
//enter busdata in the order type (1.slack,2.pv,3.pq),PG,QG,PL,QL,vmag,del,Qmin and Qmax.
//Data is taken from table 6.1
busdata=[1  0    0  0    0    1     0  0  0
```

```
            3  0    0  8    2.8  1     0  0  0
            2  5.2  0  0.8  0.4  1.05  0  4  -2.8
            3  0    0  0    0    1     0  0  0
            3  0    0  0    0    1     0  0  0]

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=1;          //Acceleration factor is assumed as 1 since it is not given in the question
tol=1e-4;         //Tolerance value for Gauss Seidal Load flow
iter=0;
err=1;
vn(1)=v(1);
vold=v(1);
while abs(err)>tol
    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)));
            vn(i)=(1/y(i,i))*(((p(i)-%i*q(i))/(conj(vn(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>tol &typ(i)==3
    v(i)=vold(i)+alpha*(v(i)-vold(i));
    end
end
if iter==1
    printf('Voltage of bus 2 at the end of first iteration in pu is given by:\n')
    printf('Voltage magnitude=%.4f , angle=%.4f
```



```
 The size of the Jacobian matrix is 8 X 8
The change in power at the end of first iteration is DelP2=-8.0000 pu
The Jacobian matrix element J1(2,4) after first iteration is: -9.9197 pu

 The Jacobian Matrix of the system at the end of first iteration is given by:

   29.758952   0.          -9.9196508  -19.839302    2.6783057   0.          -0.8927686  -1.7855371
   0.          104.41268  -104.41268    0.           0.          8.2038533  -7.8309509   0.
  -9.9196508  -104.41268   154.01093  -39.678603   -0.8927686  -7.4580485   11.548989  -3.5710743
  -19.839302   0.          -39.678603   109.23823   -1.7855371   0.          -3.5710743   9.0856357
  -2.6783057   0.           0.8927686   1.7855371   27.158952   0.          -9.9196508  -19.839302
   0.         -7.8309509    7.8309509   0.           0.          109.38471  -104.41268    0.
   0.8927686   7.8309509  -12.294794    3.5710743  -9.9196508  -99.440646   141.90687  -39.678603
   1.7855371   0.           3.5710743  -9.0856357  -19.839302   0.          -39.678603   107.91823

-->
```

Figure 6.11: Jacobian matrix and power flow solution by Newton Raphson

```
        degrees\n\n',abs(v(2)),atand(imag(v(2)),real(
        v(2))))
end
end
printf('The GS load flow converged in %d iterations
    \n',iter);
nn=1:nb;
res=[nn' abs(v) (atan(imag(v),real(v)))*(180/%pi)]
disp(res,'The final voltages in the order of bus no,
    voltage mag,voltage angle is:');
```

---

**Scilab code Exa 6.11** Jacobian matrix and power flow solution by Newton Raphson

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma,
    and Thomas J. Overbye
//Chapter - 6 ; Example 6.11
//Scilab Version - 6.0.0 ; OS - Windows

clear;
```

```
clc;
linedata=[2 4 0.0090   0.10  1.72       //Entering line data from table 6.2 & 6.3
          2 5 0.0045   0.05  0.88
          4 5 0.00225  0.025 0.44
          1 5 0.00150  0.02  0.00
          3 4 0.00075  0.01  0.00];
//enter busdata in the order type (1.slack,2.pv,3.pq),Pi,Qi,PL,QL,vmag,del,Qmin and Qmax.
//Data is taken from table 6.1
Busdata=[1  0    0  0    0    1     0  0  0
         3  0    0  8    2.8  1     0  0  0
         2  5.2  0  0.8  0.4  1.05  0  4  -2.8
         3  0    0  0    0    1     0  0  0
         3  0    0  0    0    1     0  0  0]
npv=1;               //Number of generator or PV buses in the system

rem=Busdata(:,1);
Psp=Busdata(:,2)-Busdata(:,4);
Qsp=Busdata(:,3)-Busdata(:,5);
vsp=Busdata(:,6);

//Determination of bus admittance matrix:
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 since conductsnce G=0 for all lines
nb=max(max(sb,eb));   //Number of buses in the system
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)/2;
    ybus(n,n)=ybus(n,n)+1/lz(i)+sa(i)/2;
    ybus(m,n)=-1/lz(i);
    ybus(n,m)=ybus(m,n);
end
Y=ybus;

absY=abs(Y);
thetaY=atan(imag(Y),real(Y));
v=vsp';
iteration=0;                  //Initialization of iteration count
ang=zeros(1,nb);
mismatch=ones(2*nb-2-npv,1);
tol=1e-4;                     //Tolerance value for Newton Raphson Load Flow

while max(abs(mismatch))>tol & iteration<100 //Maximum iteration count is limited to 100
    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)~=2
                  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))
```

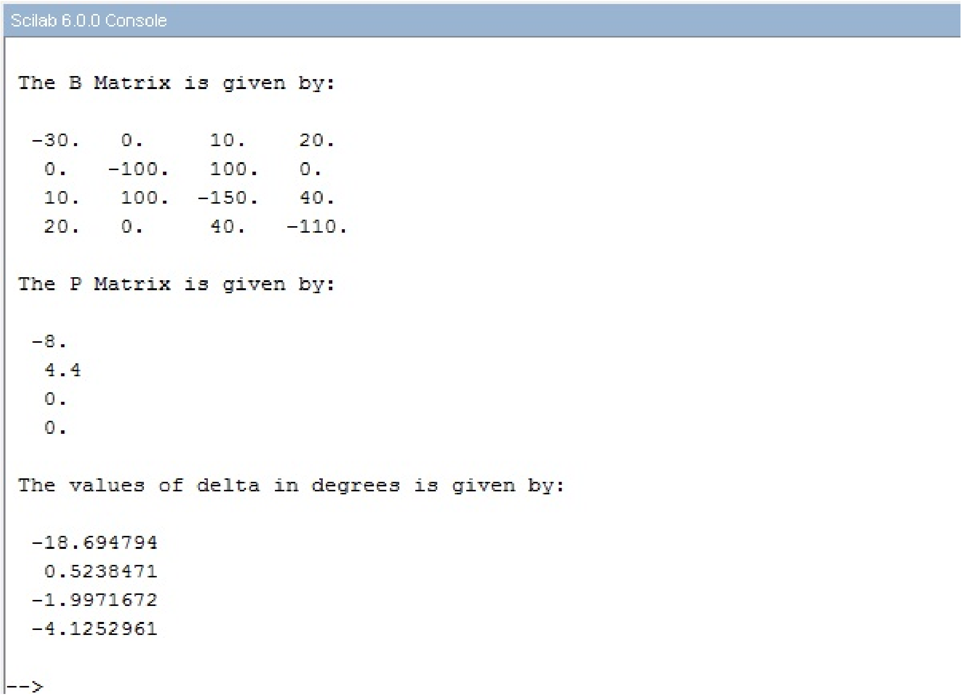
```
        -J4(i-1,i-1);
    end
J=[J1 J2;J3 J4]                              //Entire Jacobian matrix of the system
lenJ=length(J1);
i=2;
j=1;
while j<=lenJ
    if rem(i)==2
        j=j+1;
    else
        J(:,length(J1)+j)=[];
        lenJ=lenJ-1;
    end
end
i=i+1;
lenJ=length(J1);
i=1;
j=2;
while i<=lenJ
    if rem(j)==3
        i=i+1;
    else
        J(length(J1)+i,:)=[];
        lenJ=lenJ-1;
        Q(i+1)=[]
        del_Q(i+1,:)=[]
    end
    end
P(1,:)=[]                         //Removing slack bus entries
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';            //Updating voltage angle for PV and PQ buses
j=1;
for i=2:nb                         //Step to update voltage magnitude for all PQ buses
    if rem(i)==3
        v(i)=v(i)+del_v(j);
        j=j+1;
    end
end
mag=abs(v);
ang=[0 ang];
nbr=1:nb;
iteration=iteration+1;
if iteration==1
    [r c]=size(J);
    printf('The size of the Jacobian matrix is %d X %d\n',r,c)
    printf('The change in power at the end of first iteration is DelP2=%.4f pu\n',del_P(1))
    printf('The Jacobian matrix element J1(2,4) after first iteration is: %.4f pu\n',J(1,3))
    disp(J,'The Jacobian Matrix of the system at the end of first iteration is given by:')
end
end
```

---

**Scilab code Exa 6.17** dc power flow solution for the five bus system

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 6 ; Example 6.17
```



```
The B Matrix is given by:

 -30.    0.      10.     20.
  0.   -100.    100.     0.
  10.   100.   -150.     40.
  20.    0.      40.   -110.

The P Matrix is given by:

 -8.
  4.4
  0.
  0.

The values of delta in degrees is given by:

 -18.694794
  0.5238471
 -1.9971672
 -4.1252961

-->
```

Figure 6.12: dc power flow solution for the five bus system

```
//Scilab Version - 6.0.0 ; OS - Windows

clear;
clc;

linedata=[2 4 0.0090   0.10  1.72        //Entering line data from table 6.2 & 6.3
          2 5 0.0045   0.05  0.88
          4 5 0.00225  0.025 0.44
          1 5 0.00150  0.02  0.00
          3 4 0.00075  0.01  0.00];
linedata(:,3)=0                          //Neglecting Line resistance
linedata(:,5)=0                          //Neglecting shunt suceptance
//enter busdata in the order type (1.slack,2.pv,3.pq),PG,QG,PL,QL,vmag,del,Qmin and Qmax.
//Data is taken from table 6.1
Busdata=[1  0    0  0    0    1     0  0  0
         3  0    0  8    2.8  1     0  0  0
         2  5.2  0  0.8  0.4  1.05  0  4  -2.8
         3  0    0  0    0    1     0  0  0
         3  0    0  0    0    1     0  0  0]

sb= linedata(:,1);
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 since conductsnce G=0 for all lines
nb=max(max(sb,eb));
ybus=zeros(nb,nb);
for i=1:length(sb)
    m=sb(i);
    n=eb(i);
```

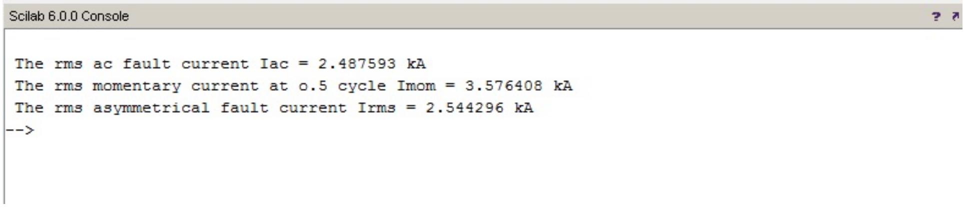
```
    ybus(m,m)=ybus(m,m)+1/lz(i)+sa(i)/2;
    ybus(n,n)=ybus(n,n)+1/lz(i)+sa(i)/2;
    ybus(m,n)=-1/lz(i);
    ybus(n,m)=ybus(m,n);
end

B=imag(ybus(2:nb,2:nb))                //B matrix is the imaginary part of bus admittance matrix neglecting slack bus
P=Busdata(2:nb,2)-Busdata(2:nb,4)      //Net power at each PV and PQ bus
delta=-inv(B)*P
deltad=delta*180/(%pi)                 //Converting delta from radian to degree
disp(B, 'The B Matrix is given by:')
disp(P,'The P Matrix is given by:')
disp(deltad,'The values of delta in degrees is given by:')
```

# Chapter 7

# SYMMETRICAL FAULTS

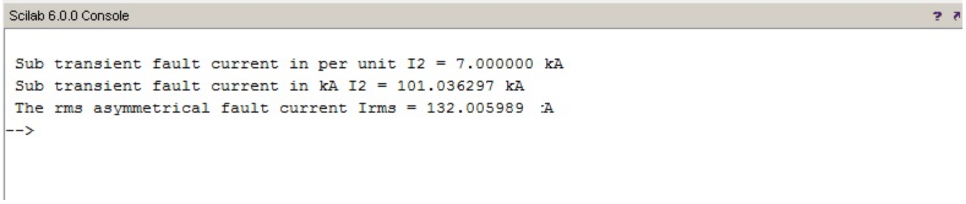
**Scilab code Exa 7.1** `Fault currents RL circuit with ac source`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 7 ; Example 7.1
//Scilab Version - 6.0.0 ; OS - Windows
clc;
clear;
V = 20;                                        //short ciruit voltage in kV
X = 8;                                        //short
```



```
The rms ac fault current Iac = 2.487593 kA
The rms momentary current at o.5 cycle Imom = 3.576408 kA
The rms asymmetrical fault current Irms = 2.544296 kA
-->
```

Figure 7.1: Fault currents RL circuit with ac source



```
Sub transient fault current in per unit I2 = 7.000000 kA
Sub transient fault current in kA I2 = 101.036297 kA
The rms asymmetrical fault current Irms = 132.005989 :A
-->
```

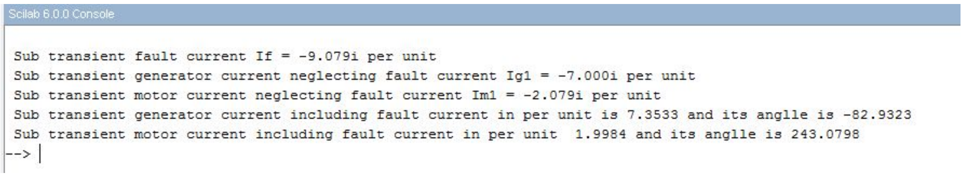
Figure 7.2: Three phase short circuit currents unloaded synchronous generator

```
    circuit inductance in ohm
R = 0.8;                                     //short
    ciruit resistance in ohm
t = 3;                                       //no. of
     cycles after fault inception
Iac = V/(sqrt((X^2)+(R^2)))*1                //rms ac
     fault current in kA
K = sqrt(1+ (2*%e^(-4*%pi*(0.5)/10)));      //
    asymmetry factor for 0.5 cycles
Imom = K*Iac;                               //rms
    momentart current at t=0.5 cycle in kA
K = sqrt(1+ (2*%e^(-4*%pi*(3)/10)));        //
    asymmetry factor for 3 cycles
Irms = K*Iac;                               //rms
    asymmetrical fault current in kA
printf('\n The rms ac fault current Iac = %f kA',Iac
    );
printf('\n The rms momentary current at o.5 cycle
    Imom = %f kA',Imom);
printf('\n The rms asymmetrical fault current Irms =
     %f kA',Irms);
```

---

**Scilab code Exa 7.2** Three phase short circuit currents unloaded synchronous genera

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 7 ; Example 7.2
//Scilab Version - 6.0.0 ; OS - Windows
clc;
clear;
Srated = 500;                    //apparent power in MVA
Vrated = 20;                     //rated voltage in kV
frated = 60;                     //fated frequency in Hz
Xd2 = 0.15;                      //synchoronous reactances per unit
Xd1 = 0.24;                      //synchoronous reactances per unit
Xd = 1.1;                        //synchoronous reactances per unit
Td2 = 0.035;                     //time constants in seconds
Td1 = 2.0;                       //time constants in seconds
Td = 0.20;                       //time constants in seconds
t = 3;                           //no. of cycles
Eg = 1.05;
```



```
 Sub transient fault current If = -9.079i per unit
 Sub transient generator current neglecting fault current Ig1 = -7.000i per unit
 Sub transient motor current neglecting fault current Im1 = -2.079i per unit
 Sub transient generator current including fault current in per unit is 7.3533 and its anglle is -82.9323
 Sub transient motor current including fault current in per unit  1.9984 and its anglle is 243.0798
--> |
```

Figure 7.3: Three phase short circuit currents power system

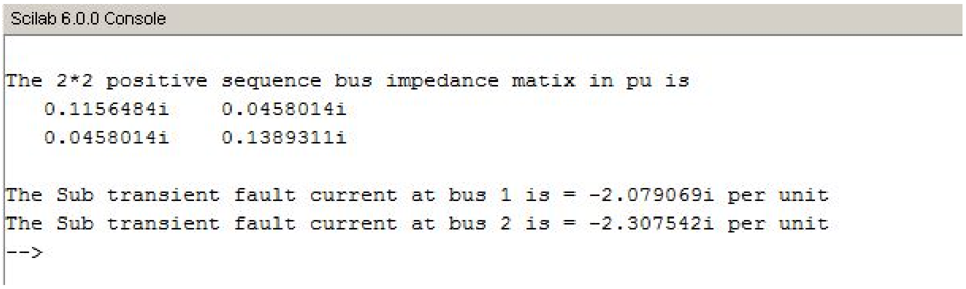
```
    //no load voltage  in per unit
I2u = Eg/Xd2;
    //sub transient fault current in per unit
Ibase = Srated/(sqrt(3)*20);
    //base current in kA
I2 = I2u*Ibase;
    //rms subtransient fault current in kA
Iac = Eg*((((1/Xd2)-(1/Xd1))*exp(-0.05/Td2))+(((1/
    Xd1)-(1/Xd))*exp(-0.05/Td1))+(1/Xd));            //
    rms ac fault current in per unit
Iac=Iac*Ibase;
    //rms ac fault current in kA
Irms = sqrt((Iac^2)+((sqrt(2)*I2*exp(-0.05/Td))^2));
                                                  //rms
     asymmetrical fault current in kA
printf('\n Sub transient fault current in per unit
    I2 = %f kA',I2u);
printf('\n Sub transient fault current in kA I2 = %f
     kA',I2);
printf('\n The rms asymmetrical fault current Irms =
     %f kA',Irms);
```

**Scilab code Exa 7.3** `Three phase short circuit currents power system`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma,
   and Thomas J. Overbye
//Chapter - 7 ; Example 7.3
//Scilab Version - 6.0.0 ; OS - Windows
clc;
clear;
Srated = 100;

   //rated power in MVA
V1 = 13.8;

   //generator supply voltage in kV
Xg = 0.15;

   //generator input reactance in ohm
Vline = 138;

   //transmission line voltage in kV
Xline = 20;

   //transmission line reactance in ohm
Vprtr1 = 13.8;

   //primary side voltage of transformer 1 in kV
Vsectr1 = 138;

   //secondary side voltage of transformer 1 in kV
Xt1 = 0.10;

   //reactance of transformer 1 in ohm
Vprtr2 = 138;
```

```
   //primary side voltage of transformer 2 in kV
Vsectr2 = 13.8;

   //secondary side voltage of transformer 2 in kV
Xt2 = 0.10;

   //reactance of transformer 2 in ohm
V2 = 13.8;

   //motor supply voltage in kV
Xm = 0.20;

   //motor reactance in ohm
pf =0.95;

   //lagging power factor
Rth1 = 0.15;

   //thevenins resistance in ohm
Rth2 = 0.505;

   //thevenins resistance in ohm
Vf=1.05;

   //prefaault voltage at the generator terminals
Zbl = (Vsectr1^2);

   //base impedance of the transmission line in ohm
Xlinepu = Xline/Zbl;

   //transmission line reactance in per unit
Zth = %i*((Rth1*Rth2)/(Rth1+Rth2));
                                                    //Thevenin
   's impedance per unit
If = Vf/Zth;

   //sub transient fault current in per unit
```

```
Ig1 = ((Rth2/(Rth2+Rth1))*If);                                   //sub tranisent generator current in per unit
Im1 = ((Rth1/(Rth2+Rth1))*If);                                   //sub transient motor current in per unit
Ibase = (Srated/((sqrt(3))*(V1)));                               //generator base current in kA
Il = ((Srated/((sqrt(3))*V1*Vf))*(cos(-acos(pf))+%i*sin(-acos(pf))));       //prefault generator current  in kA
Il = Il/Ibase;                                                   //prefault generator current in per unit
Ig = Ig1 + Il;                                                   //sub transient generator current including pre fault current in per unit
Im = Im1 - Il;                                                   //sub transient motor current including pre fault current in per unit

printf('\n Sub transient fault current If = %0.3fi per unit',imag(If));
printf('\n Sub transient generator current neglecting fault current Ig1 = %0.3fi per unit ',imag(Ig1));
printf('\n Sub transient motor current neglecting fault current Im1 = %0.3fi per unit ',imag(Im1));
printf('\n Sub transient generator current including fault current in per unit is %0.4f and its anglle is %0.4f', abs(Ig), atand(imag(Ig), real(Ig)));
printf('\n Sub transient motor current including fault current in per unit  %0.4f and its anglle is %0.4f',abs(Im), atand(imag(Im), real(Im))+360)
```



```
The 2*2 positive sequence bus impedance matix in pu is
   0.1156484i    0.0458014i
   0.0458014i    0.1389311i

The Sub transient fault current at bus 1 is = -2.079069i per unit
The Sub transient fault current at bus 2 is = -2.307542i per unit
-->
```

Figure 7.4: Using Zbus to compute three phase short circuit currents in a power system

```
    ;
//360 is added to get positive angle. There will not be any change in angle because 360 degree and 0 degree are same.
```

---

**Scilab code Exa 7.4** Using Zbus to compute three phase short circuit currents in a

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 7 ; Example 7.4
//Scilab Version - 6.0.0 ; OS - Windows
clc;
clear;
Srated = 100;                                                    //rated power in MVA
V1 = 13.8;                                                       //generator supply voltage in kV
Xg = 0.15;
```

```
   //generator input reactance in ohm
Vline = 138;

   //transmission line voltage in kV
Xline = 20;

   //transmission line reactance in ohm
Vprtr1 = 13.8;                                          //
   primary side voltage of transformer 1 in kV
Vsectr1 = 138;                                          //
   secondary side voltage of transformer 1 in kV
Xt1 = 0.10;

   //reactance of transformer 1 in ohm
Vprtr2 = 138;                                            //
   primary side voltage of transformer 2 in kV
Vsectr2 = 13.8;                                         //
   secondary side voltage of transformer 2 in kV
Xt2 = 0.10;

   //reactance of transformer 2 in ohm
V2 = 13.8;

   //motor supply voltage in kV
Xm = 0.20;

   //motor reactance in ohm
Vf = 1.05;

   //pre fault voltage in per unit
Ybus = -%i*[9.9454 -3.2787; -3.2787 8.2787];
                    //bus admittance matrix in per
   unit using direct inspection from fig 7.5
Zbus = inv(Ybus);
```

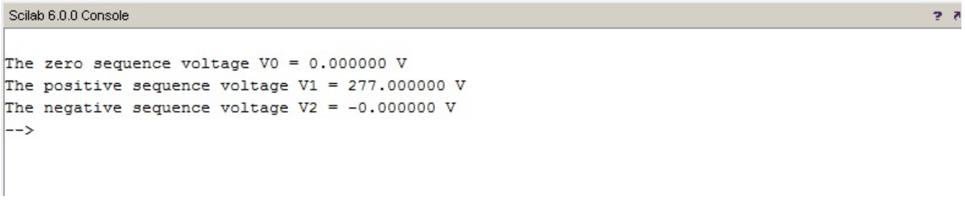
```
                                                        //bus impedance matrix in per unit
If1 = Vf/Zbus(1,1);                                                       //sub transient fault current at bus 1 in per unit
E1 = (1- (Zbus(1,1)/Zbus(1,1)))*Vf;                                  //voltage at bus 1 in V
E2 = (1-((Zbus(2,1)/Zbus(1,1))))*Vf;                                //voltage at bus 2 in V
Xline =Xline*Srated/(Vline^2);                                            //line impedance in ohm
I21 = ((E2-E1)/(%i*(Xline+Xt1+Xt2)));                              //fault current from transmission line in per unit
If2 = Vf/Zbus(2,2);                                                       //sub transient fault current at bus 2 in per unit
E3 = (1-(Zbus(1,2)/Zbus(2,2)))*Vf;                                    //voltage at bus 3 in V
E4 = (1-(Zbus(2,2)/Zbus(2,2)))*Vf;                                    //voltage at bus 4 in V
I12 = ((E3-E4)/(%i*(Xline+Xt1+Xt2)));                              //current to fault from transmission line in per unit

printf('\nThe 2*2 positive sequence bus impedance matix in pu is ');
disp (Zbus);
printf('\nThe Sub transient fault current at bus 1 is = %fi per unit',imag(I21));
printf('\nThe Sub transient fault current at bus 2 is = %fi per unit',imag(I12));
```

# Chapter 8

# SYMMETRICAL COMPONENTS

**Scilab code Exa 8.1** `Sequence components balanced line to neutral voltages`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 8 ; Example 8.1
//Scilab Version - 6.0.0 ; OS - Windows
clc;
clear;
Vp = [277; 277*(cos(-120*%pi/180)+%i*sin(-120*%pi
```



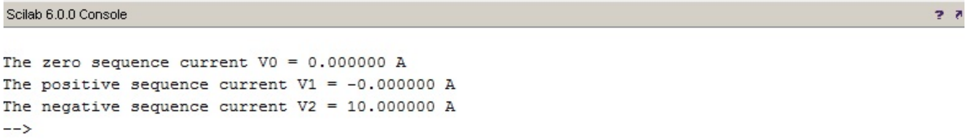
```
The zero sequence voltage V0 = 0.000000 V
The positive sequence voltage V1 = 277.000000 V
The negative sequence voltage V2 = -0.000000 V
-->
```

Figure 8.1: Sequence components balanced line to neutral voltages

```
/180)); 277*(cos(120*%pi/180)+%i*sin(120*%pi/180))];   //given column vector of phase voltage in volts
function [Vp1]=phaseshift(x1,x2)                                //Function for shifting the phase
  [r theta]=polar(x1);
  Vp1=r*(cos(theta+x2*%pi/180)+%i*sin(theta+x2*%pi/180));
endfunction
V0 = 1*(Vp(1,1)+Vp(2,1)+Vp(3,1))/3;                                //zero sequence voltage in V
V1 = 1*(Vp(1,1)+phaseshift(Vp(2,1),120)+phaseshift(Vp(3,1),240))/3;                                //positive sequence voltage in V
V2 = 1*(Vp(1,1)+phaseshift(Vp(2,1),240)+phaseshift(Vp(3,1),120))/3;                                //negative sequence voltage in V
printf('\nThe zero sequence voltage V0 = %f V',V0);
printf('\nThe positive sequence voltage V1 = %f V',V1);
printf('\nThe negative sequence voltage V2 = %f V',V2);
```

**Scilab code Exa 8.2** `Sequence components balanced acb currents`

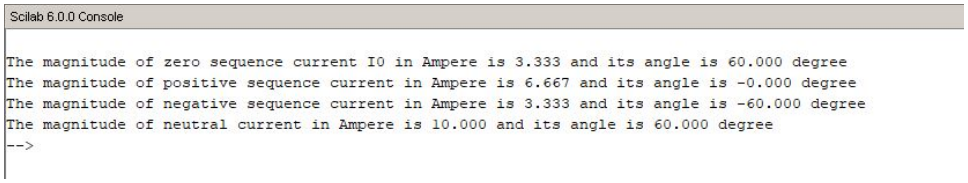
```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 8 ; Example 8.2
```



```
The zero sequence current V0 = 0.000000 A
The positive sequence current V1 = -0.000000 A
The negative sequence current V2 = 10.000000 A
-->
```

Figure 8.2: Sequence components balanced acb currents

```
//Scilab Version - 6.0.0 ; OS - Windows
clc;
clear;
Ip = [10; 10*(cos(120*%pi/180)+%i*sin(120*%pi/180)); 10*(cos(-120*%pi/180)+%i*sin(-120*%pi/180))]; //given column vector of phase current in A
function [Ip1]=phaseshift(x1,x2)                    //Function for shifting the phase
  [r theta]=polar(x1);
  Ip1=r*(cos(theta+x2*%pi/180)+%i*sin(theta+x2*%pi/180));
endfunction
I0 = 1*(Ip(1,1)+Ip(2,1)+Ip(3,1))/3;                  //zero sequence current in A
I1 = 1*(Ip(1,1)+phaseshift(Ip(2,1),120)+phaseshift(Ip(3,1),240))/3;          //positive sequence current in A
I2 = (Ip(1,1)+phaseshift(Ip(2,1),240)+phaseshift(Ip(3,1),120))/3;            //negative sequence current in A
printf('\nThe zero sequence current V0 = %f A',I0);
printf('\nThe positive sequence current V1 = %f A',I1);
printf('\nThe negative sequence current V2 = %f A',
```



```
The magnitude of zero sequence current I0 in Ampere is 3.333 and its angle is 60.000 degree
The magnitude of positive sequence current in Ampere is 6.667 and its angle is -0.000 degree
The magnitude of negative sequence current in Ampere is 3.333 and its angle is -60.000 degree
The magnitude of neutral current in Ampere is 10.000 and its angle is 60.000 degree
-->
```

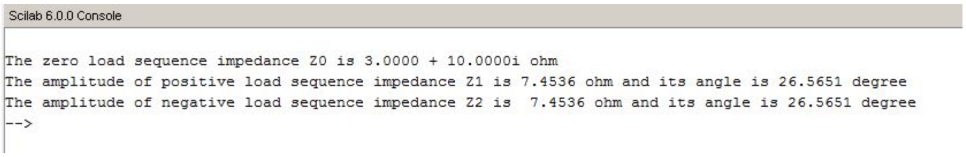
Figure 8.3: Sequence components unbalanced currents

```
  I2);
```

---

**Scilab code Exa 8.3** `Sequence components unbalanced currents`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 8 ; Example 8.3
//Scilab Version - 6.0.0 ; OS - Windows
clc;
clear;
Ip = [10; 0; 10*(cos(120*%pi/180)+%i*sin(120*%pi/180))];;            //given column vector of phase current in A
function [Ip1]=phaseshift(x1,x2)                          //Function for shifting the phase
    [r theta]=polar(x1);
    Ip1=r*(cos(theta+x2*%pi/180)+%i*sin(theta+x2*%pi/180));
endfunction

I0 = (Ip(1,1)+Ip(2,1)+Ip(3,1))/3;                                  //zero sequence current in A
```



```
The zero load sequence impedance Z0 is 3.0000 + 10.0000i ohm
The amplitude of positive load sequence impedance Z1 is 7.4536 ohm and its angle is 26.5651 degree
The amplitude of negative load sequence impedance Z2 is  7.4536 ohm and its angle is 26.5651 degree
-->
```

Figure 8.4: Sequence networks balanced star and balanced delta loads

```
I1 = 1*(Ip(1,1)+(Ip(2,1)+phaseshift(Ip(3,1),240)))/3;            //positive sequence current in A
I2 = (Ip(1,1)+Ip(2,1)+phaseshift(Ip(3,1),120))/3;                //negative sequence current in A
In = (Ip(1,1)+Ip(2,1)+Ip(3,1));                                             //neutral current in A
printf('\nThe magnitude of zero sequence current I0 in Ampere is %0.3f and its angle is %0.3f degree',abs(I0), atand(imag(I0), real(I0)));
printf('\nThe magnitude of positive sequence current in Ampere is %0.3f and its angle is %0.3f degree',abs(I1), atand(imag(I1), real(I1)));
printf('\nThe magnitude of negative sequence current in Ampere is %0.3f and its angle is %0.3f degree',abs(I2), atand(imag(I2), real(I2)));
printf('\nThe magnitude of neutral current in Ampere is %0.3f and its angle is %0.3f degree',abs(In), atand(imag(In), real(In)));
```

---

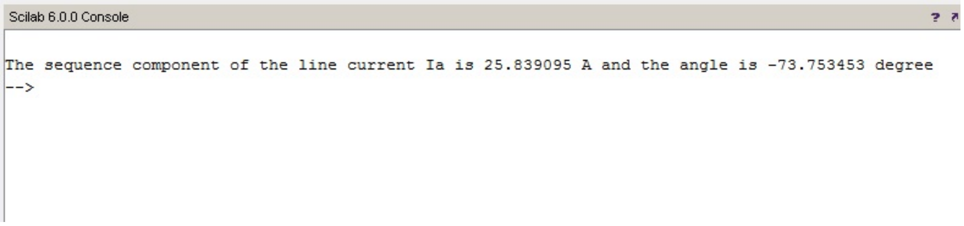
**Scilab code Exa 8.4** `Sequence networks balanced star and balanced delta loads`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 8 ; Example 8.4
```

```
//Scilab Version - 6.0.0 ; OS - Windows
clc;
clear;
Zy = (3+(%i*4));                                  //Y load impedance per phase
Xn = 2;                                           //inductive reactance in ohm per phase
Xc = -%i*30;                                      //capacitor bank reactance in ohm per phase
Zn = %i*2                                         //neutral impedance in ohm per phase
Zdel = Xc/3;

Z0 = Zy+(3*Zn);                                  //zero load sequence impedane in ohm
Z1 = 1/(1/Zy+1/Zdel);                            //positive load sequence impedane in ohm
Z2 =Z1;                                          //negativa load sequence impedane in ohm
printf('\nThe zero load sequence impedance Z0 is %0.4f + %0.4fi ohm',real(Z0), imag(Z0));
printf('\nThe amplitude of positive load sequence impedance Z1 is %.4f ohm and its angle is %.4f degree ',abs(Z1), atand(imag(Z1), real(Z1)));
printf('\nThe amplitude of negative load sequence impedance Z2 is  %.4f ohm and its angle is %.4f degree ',abs(Z2), atand(imag(Z2), real(Z2)));
```

**Scilab code Exa 8.5** Currents in sequence networks

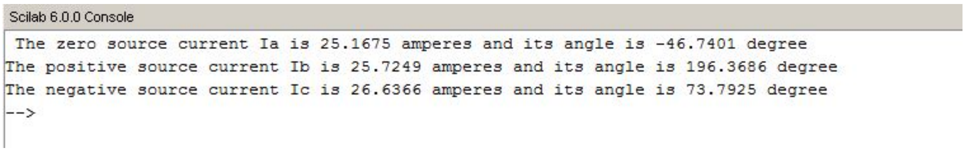
```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
```



```
The sequence component of the line current Ia is 25.839095 A and the angle is -73.753453 degree
-->
```

Figure 8.5: Currents in sequence networks

```
//Chapter - 8 ; Example 8.5
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;

Zn = %i*10;                                       //generator
   neutral impedance in ohm
Zgo = %i*1;                                       //generator
   zero sequence impedance in ohm
Zg1 = %i*15;                                      //generator
   positive sequence impedance in ohm
Zg2 = %i*3;                                       //generator
   negative sequence impedance in ohm
Zl1 = 0.087+(%i*0.99);                            //line
   impedace in ohm
Zdel = 22.98+%i*(19.281);                         //impedance
   of the delta load in ohm
V1=(415.69-(%i*240))/sqrt(3);                     //RMS line to
    neutral phase voltage of AC supply in Volts
I1 = V1/(Zl1+((1/3)*Zdel));                       //sequence
   component of line current in A

printf('\nThe sequence component of the line current
    Ia is %.4f amperes and its angle is %.4f degree
   ',abs(I1), atand(imag(I1), real(I1)));
```



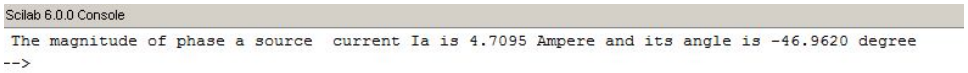
```
 The zero source current Ia is 25.1675 amperes and its angle is -46.7401 degree
The positive source current Ib is 25.7249 amperes and its angle is 196.3686 degree
The negative source current Ic is 26.6366 amperes and its angle is 73.7925 degree
-->
```

Figure 8.6: Solving unbalanced three phase networks using sequence components

**Scilab code Exa 8.6** `Solving unbalanced three phase networks using sequence compone`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 8 ; Example 8.6
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;

Vp = [277; 260*(cos(-120*%pi/180)+%i*sin(-120*%pi/180)); 295*(cos(115*%pi/180)+%i*sin(115*%pi/180))];    //given column vector of phase voltage in volts
Zl1 = 0.087+%i*(0.99);                                  //impedace of line 1 in ohm
Zdel = 22.98+%i*(19.281);                               //impedance of the delta load in ohm
Zl2 = 0.087+%i*(0.99);                                  //impedance of line 2 in ohm
function [Vp1]=phaseshift(x1,x2)                           //Function for shifting the phase
```

```
    [r theta]=polar(x1);
    Vp1=r*(cos(theta+x2*%pi/180)+%i*sin(theta+x2*%pi/180));
endfunction

V0 = (Vp(1,1)+Vp(2,1)+Vp(3,1))/3;                                  //zero sequence voltage in V
V1 = (Vp(1,1)+phaseshift(Vp(2,1),120)+phaseshift(Vp(3,1),240))/3;  //positive sequence voltage in V
V2 = (Vp(1,1)+phaseshift(Vp(2,1),240)+phaseshift(Vp(3,1),120))/3;  //negative sequence voltage in V
I0 = 0;                                                            //zero sequence current in A
I1 = V1/(Zl1+(Zdel/3));                                            //positive sequence current in A
I2 = V2/(Zl2+(Zdel/3));                                            //negative sequence current in A
Ia = I0+I1+I2;                                                     //zero source current in A
Ib = I0+phaseshift(I1,240)+phaseshift(I2,120);                     //positive source current in A
Ic = I0+phaseshift(I1,120)+phaseshift(I2,240);                     //negative source current in A
printf('The zero source current Ia is %.4f amperes and its angle is %.4f degree ',abs(Ia), atand(imag(Ia), real(Ia)));
printf('\nThe positive source current Ib is %.4f
```



```
 The magnitude of phase a source  current Ia is 4.7095 Ampere and its angle is -46.9620 degree
-->
```

Figure 8.7: Solving unbalanced three phase networks with transformers using per unit sequence components

```
   amperes and its angle is %.4f degree ',abs(Ib),
   atand(imag(Ib), real(Ib))+360);
printf('\nThe negative source current Ic is %.4f
   amperes and its angle is %.4f degree ',abs(Ic),
   atand(imag(Ic), real(Ic)));
```

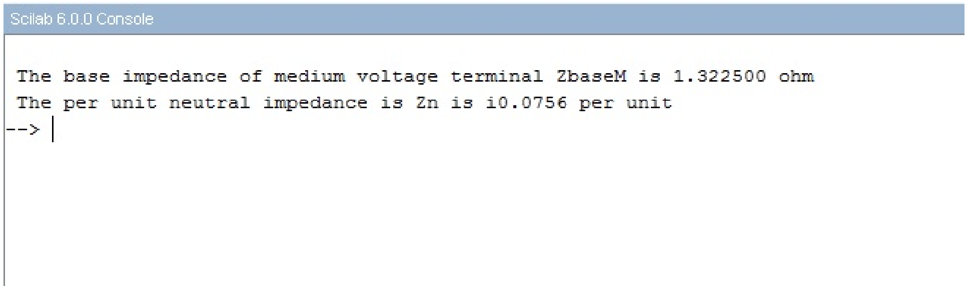
---

**Scilab code Exa 8.7** `Solving unbalanced three phase networks with transformers usin`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma,
   and Thomas J. Overbye
//Chapter - 8 ; Example 8.7
//Scilab Version - 6.0.0 ; OS - Windows
clc;
clear;
Q = 75;

   //rated power in kVA
Vprtr = 480;

   //primary side voltage of transformer in volts
Vsectr = 208;

   //secondary side voltage of transformer in volts
Xeq = 0.10;

   //leakage reactance in per unit
```

```
Sbase = Q/3;                                                    //base quantity of rated power in single phase in kVA
VbaseHLN = Vprtr/(sqrt(3));                                     //base quantity of primary side voltage of transformer in volts
VbaseXLN = Vsectr/(sqrt(3));                                    //base quantity of secondary side voltage of transformer in volts
ZbaseX = 0.5770;                                                //base quantity of impedance in ohm
Vp = [277; 260*(cos(-120*%pi/180)+%i*sin(-120*%pi/180)); 295*(cos(115*%pi/180)+%i*sin(115*%pi/180))];   //given column vector of phase voltage in volts
function [Vp1]=phaseshift(x1,x2)
    [r theta]=polar(x1);
    Vp1=r*(cos(theta+x2*%pi/180)+%i*sin(theta+x2*%pi/180));
endfunction

V0 = (Vp(1,1)+Vp(2,1)+Vp(3,1))/3;                               //zero sequence voltage in V
V1 = (Vp(1,1)+phaseshift(Vp(2,1),120)+phaseshift(Vp(3,1),240))/3;                                    //positive sequence voltage in V
V2 = (Vp(1,1)+phaseshift(Vp(2,1),240)+phaseshift(Vp(3,1),120))/3;                                    //negative sequene voltage in v
V0 = V0/VbaseHLN;
```

```
    //zero sequence voltage in per unit
V1 = V1/VbaseHLN;

    //positive sequence voltage in per unit
V2 = V2/VbaseHLN;

    //negative sequene voltage in per unit
Zline0 = 0.087+%i*(0.99);

    //line impedance in ohm
Zload1 = 22.98+%i*(19.281);

    //load impedance in ohm
Zline0 = Zline0/ZbaseX;

    //line impedance in per unit
Zload1 = Zload1/(3*ZbaseX);

    //line impedance in per unit
I0 = 0;

    //zero sequence component of source current in per unit
I1 = V1/((%i*Xeq)+Zline0+Zload1);

    //positive sequence component of source current in per unit
I2 = V2/((%i*Xeq)+Zline0+Zload1);

    //negative sequence component of source current in per unit
Ia = I0+I1+I2;

    //phase 'a' source current in per unit
IbaseH=(Q*10^3)/(Vprtr*sqrt(3));

    //base current in A
Ia =  Ia*IbaseH;
```



```
 The base impedance of medium voltage terminal ZbaseM is 1.322500 ohm
 The per unit neutral impedance is Zn is i0.0756 per unit
-->
```

Figure 8.8: Three winding three phase transformer per unit sequence networks

```
    //phase 'a' source current in A
printf('The magnitude of phase a source  current Ia is %.4f Ampere and its angle is %.4f degree',abs(Ia),atand(imag(Ia),real(Ia)));
```

---

**Scilab code Exa 8.8** `Three winding three phase transformer per unit sequence networ`

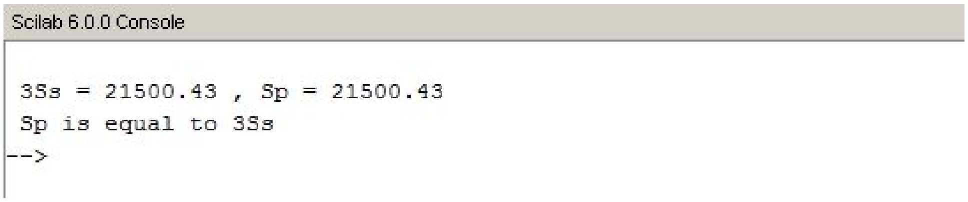
```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 8 ; Example 8.8
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;

Q = 900;                                                //rated power in MVA
Vg = 13.8;                                              //
```

```
generator voltage in kV
Vt = 345;                                                  //transmission line voltage in kV
Vd = 34.5;                                                 //distribution line voltage in kV
V1 = 13.8;                                                 //voltage at the winding X in kV
V2 = 199.2;                                                //voltage at the winding H in kV
V3 = 19.92;                                                //voltage at the winding M in kV
Zn = %i*0.10;                                              //neutral impedance in ohm
VbaseX = 13.8;                                             //rated line to line voltage of terminal X in kV
VbaseM = sqrt(3)*V3;                                       //rated line to line voltage of terminal M in kV
ZbaseM = (Vd^2)/Q;                                         //base impedance of medium line voltage in ohm
Zn = Zn/ZbaseM;                                            //neutral impedance in per unit

printf('\n The base impedance of medium voltage terminal ZbaseM is %f ohm',ZbaseM);
printf('\n The per unit neutral impedance is Zn is i%0.4f per unit', imag(Zn));
```

**Scilab code Exa 8.9** `Power in sequence networks`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 8 ; Example 8.8
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;

Vp = [277; 260*(cos(-120*%pi/180)+%i*sin(-120*%pi/180)); 295*(cos(115*%pi/180)+%i*sin(115*%pi/180))];   //given column vector of phase voltage in volts
Zl1 = 0.087+%i*(0.99);                    //impedace of line 1 in ohm
Zdel = 22.98+%i*(19.281);                  //impedance of the delta load in ohm
Zl2 = 0.087+%i*(0.99);                    //impedance of line 2 in ohm
function [Vp1]=phaseshift(x1,x2)
    [r theta]=polar(x1);
    Vp1=r*(cos(theta+x2*%pi/180)+%i*sin(theta+x2*%pi/180));
endfunction

V0 = (Vp(1,1)+Vp(2,1)+Vp(3,1))/3;                  //zero sequence voltage in V
V1 = (Vp(1,1)+phaseshift(Vp(2,1),120)+phaseshift(Vp
```

```
(3,1),240))/3;                                                    //positive sequence voltage in V
V2 = (Vp(1,1)+phaseshift(Vp(2,1),240)+phaseshift(Vp(3,1),120))/3;                                                    //negative sequence voltage in V
I0 = 0;                                                    //zero sequence current in A
I1 = V1/(Zl1+(Zdel/3));                                                    //positive sequence current in A
I2 = V2/(Zl2+(Zdel/3));                                                    //negative sequence current in A
Ia = I0+I1+I2;                                                    //zero source current in A
Ib = I0+phaseshift(I1,240)+phaseshift(I2,120);                                                    //positive source current in A
Ic = I0+phaseshift(I1,120)+phaseshift(I2,240);                                                    //negative source current in A
Sp = (Vp(1,1)*(conj(Ia)))+(Vp(2,1)*(conj(Ib)))+(Vp(3,1)*(conj(Ic)));                                                    //total complex power delivered to load in VA
Ss = (V0*conj(I0))+(V1*conj(I1))+(V2*conj(I2));                                                    //total complex power delivered to the sequence networks in VA
SS = 3*Ss;
printf('\n 3Ss = %0.2f , Sp = %0.2f ',abs(SS), abs(Sp));
if (ceil(real(SS))==ceil(real(Sp))) then
  printf('\n Sp is equal to 3Ss');
```



```
 3Ss = 21500.43 , Sp = 21500.43
 Sp is equal to 3Ss
-->
```

Figure 8.9: Power in sequence networks

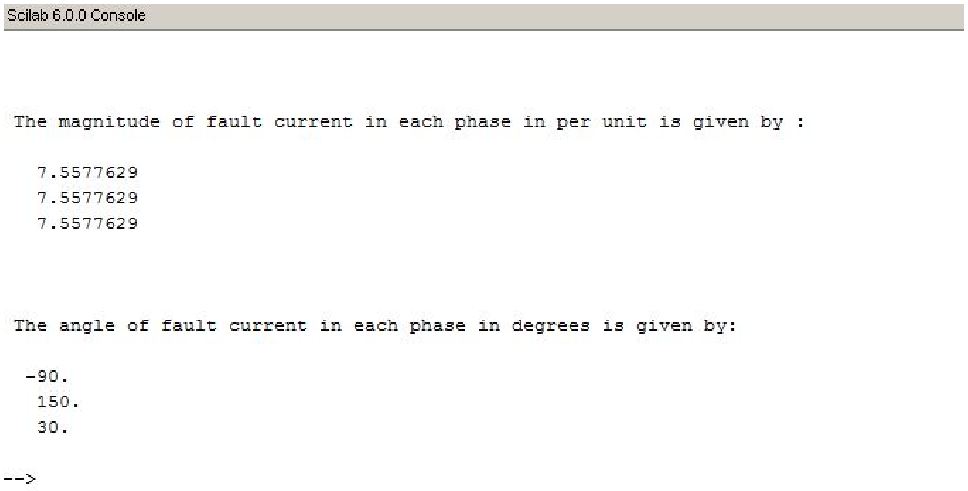
```
else
    printf('\n Sp is not equal to 3Ss');
end
```

# Chapter 9

# UNSYMMETRICAL FAULTS

**Scilab code Exa 9.2** `Three phase short circuit calculations using sequence networks`

```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-9 ;Example 9.2
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;

Xn=0.05                             //motor neutral is grounded through reactance in per unit
Sb=100                              //Base value of system in MVA
Vb=13.8                             //Base voltage of system in kV
Vf=1.05                             //Prefault voltage in per unit
Z1=%i*0.13893                       //Positive sequence
```

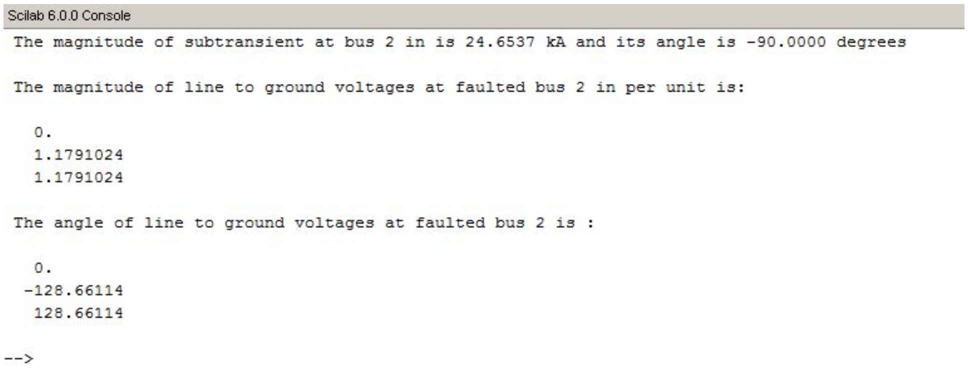


```
Scilab 6.0.0 Console


 The magnitude of fault current in each phase in per unit is given by :

   7.5577629
   7.5577629
   7.5577629



 The angle of fault current in each phase in degrees is given by:

  -90.
   150.
   30.

-->
```

Figure 9.1: Three phase short circuit calculations using sequence networks

```
    impedance in per unit


If=Vf/Z1                                        //positive sequence fault current in per unit
a=exp(%i*(120)*(%pi/180))         //operator a
Isf=[1 1 1;1 (a^2) a;1 a (a^2)]*[0;If;0]   //subtransient fault current in each phase in per unit

disp(abs(Isf),'The magnitude of fault current in each phase in per unit is given by :',);
disp(atand(imag(Isf),real(Isf)),'The angle of fault current in each phase in degrees is given by:',);
```

**Scilab code Exa 9.3** Single line to ground short circuit calculations using sequenc

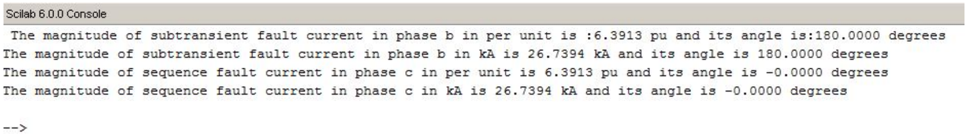


```
 The magnitude of subtransient at bus 2 in is 24.6537 kA and its angle is -90.0000 degrees

 The magnitude of line to ground voltages at faulted bus 2 in per unit is:

   0.
   1.1791024
   1.1791024

 The angle of line to ground voltages at faulted bus 2 is :

   0.
  -128.66114
   128.66114

-->
```

Figure 9.2: Single line to ground short circuit calculations using sequence networks

```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-9 ;Example 9.3
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;

Xn=0.05                           //motor neutral is grounded through reactance in per unit
Sb=100                            //Base value of system in MVA
Vb=13.8                           //Base voltage of system in kV
Vf=1.05                           //Prefault voltage in per unit
Z0=%i*0.250                       //Zero sequence impedance in per unit
Z1=%i*0.13893                     //Positive sequence impedance in per unit
Z2=%i*0.14562                     //Negative sequence
```

```
impedance in per unit
Zf=0                                //Fault through impedance in per unit

If0=Vf/(Z0+Z1+Z2+(3*Zf))            //sequence line to ground fault current in per unit
If1=If0;If2=If0;                    // Since If0=If1=If2
If=[If0;If1;If2]
Isf=3*If0                           //subtransient fault current in per unit
Ib2=Sb/(Vb*sqrt(3))                 //base current at bus 2 in kA
Ib22=Isf*Ib2
Vsf=[0;Vf;0]-([Z0 0 0;0 Z1 0;0 0 Z2]*If)        //sequence componenets of the voltages at the fault in per unit
a=exp(%i*(120)*(%pi/180));                      //operator a
Vlg2=[1 1 1;1 (a^2) a;1 a (a^2)]*Vsf            //line to ground voltages at faulted bus 2 in per unit
for i=1:3                               //This loop is included to avoid discrepancies in angle values when the voltage value is near to zero or zero
   if abs(Vlg2(i))<1e-6                 //For example, atand(0,0) gives 0 degree and atand(0,-0) gives 180 degree
      Vlg2(i)=0;
   end
end

printf('The magnitude of subtransient at bus 2 in is %.4f kA and its angle is %.4f degrees\n',abs(Ib22),atand(imag(Ib22),real(Ib22)));
disp(abs(Vlg2),'The magnitude of line to ground voltages at faulted bus 2 in per unit is:');
disp(atand(imag(Vlg2),real(Vlg2)),'The angle of line to ground voltages at faulted bus 2 is :');
```



```
 The magnitude of subtransient fault current in phase b in per unit is :6.3913 pu and its angle is:180.0000 degrees
The magnitude of subtransient fault current in phase b in kA is 26.7394 kA and its angle is 180.0000 degrees
The magnitude of sequence fault current in phase c in per unit is 6.3913 pu and its angle is -0.0000 degrees
The magnitude of sequence fault current in phase c in kA is 26.7394 kA and its angle is -0.0000 degrees

-->
```

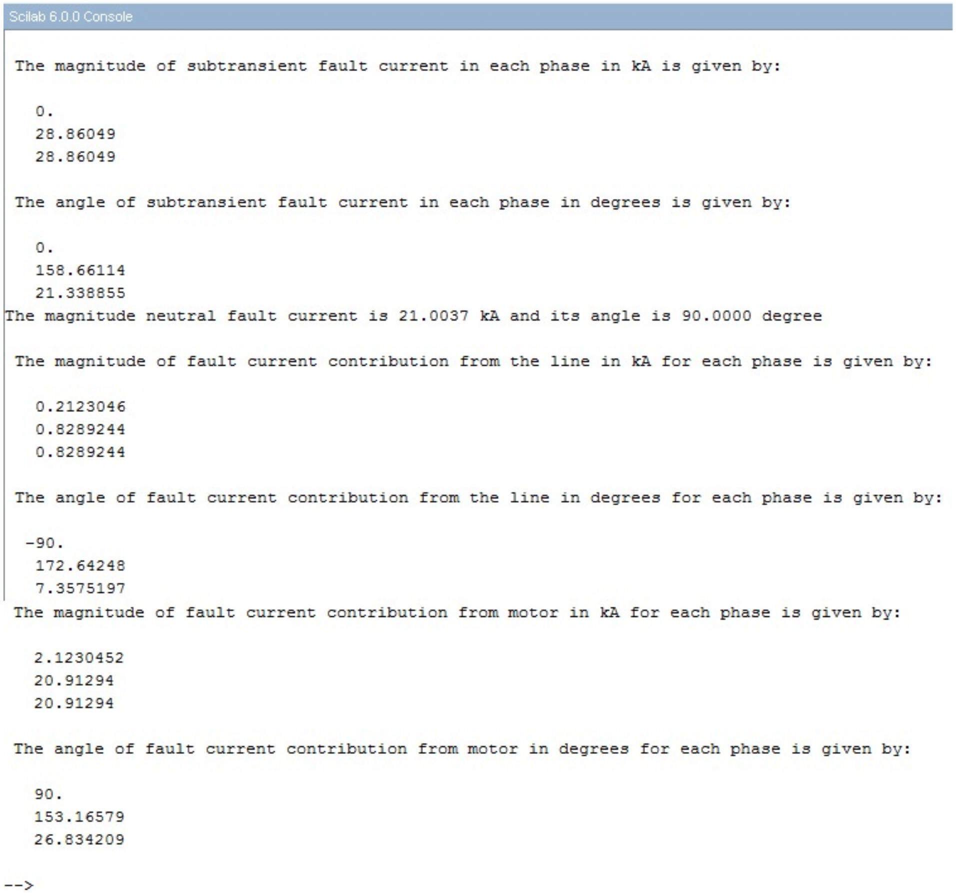
Figure 9.3: Line to line short circuit calculations using sequence networks

**Scilab code Exa 9.4** `Line to line short circuit calculations using sequence network`

```
//Book - Power system: Analysisi & Design 5th
   Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma,
   and Thomas J.Overbye
//Chapter-9 ;Example 9.4
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;

Sb=100                                   //Base value of
   system in MVA
Vb=13.8                                  //Base voltage of
   system in kV
Vf=1.05                                  //Prefault voltage
   in per unit
Z1=%i*0.13893                            //Positive sequence
   impedance in per unit
Z2=%i*0.14562                            //Negative sequence
   impedance in per unit
Zf=0                                     //Fault through
   impedance in per unit
I0=0                                     //Zero sequence
```

```
current in per unit

I1=Vf/(Z1+Z2+Zf)                    //sequence fault current in per unit
Isfb=-%i*sqrt(3)*I1                 //subtransient fault current in phase b in per unit
Ib2=(Sb/(Vb*sqrt(3)))*Isfb          //subtransient fault current at phase b in kA
Isfc=-Isfb;                         //subtransient fault current at phase c in pu
Ic=-Ib2                             //subtransient fault current at phase c in kA

printf('The magnitude of subtransient fault current in phase b in per unit is :%.4f pu and its angle is:%.4f degrees\n',abs(Isfb),(180/%pi)*atan(imag(Isfb),real(Isfb)));
printf('The magnitude of subtransient fault current in phase b in kA is %.4f kA and its angle is %.4f degrees\n',abs(Ib2),(180/%pi)*atan(imag(Ib2),real(Ib2)));


printf('The magnitude of sequence fault current in phase c in per unit is %.4f pu and its angle is %.4f degrees\n',abs(Isfc),(180/%pi)*atan(imag(Isfc),real(Isfc)));
printf('The magnitude of sequence fault current in phase c in kA is %.4f kA and its angle is %.4f degrees\n',abs(Ic),(180/%pi)*atan(imag(Ic),real(Ic)));
```

**Scilab code Exa 9.5** `Double line to ground short circuit calculations using sequenc`



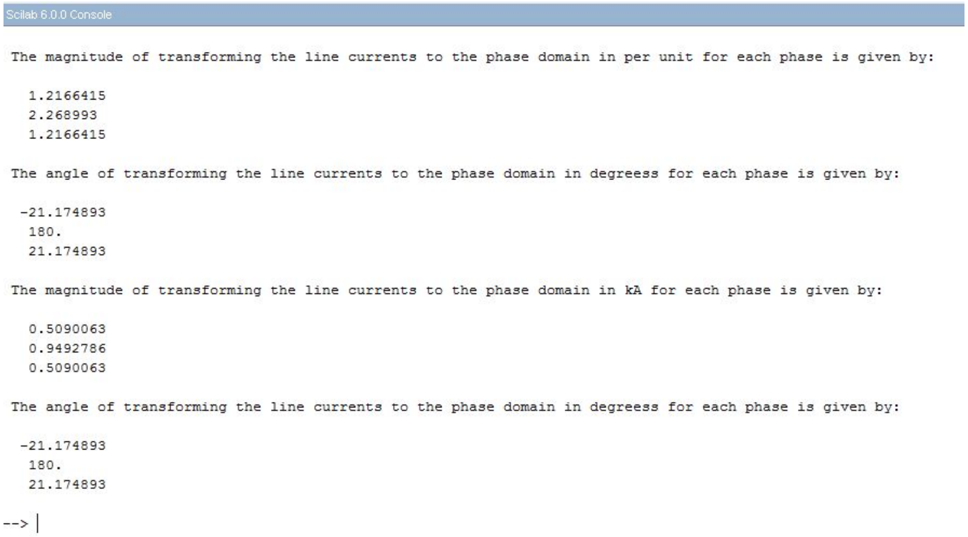
```
 The magnitude of subtransient fault current in each phase in kA is given by:

   0.
   28.86049
   28.86049

 The angle of subtransient fault current in each phase in degrees is given by:

   0.
   158.66114
   21.338855
The magnitude neutral fault current is 21.0037 kA and its angle is 90.0000 degree

 The magnitude of fault current contribution from the line in kA for each phase is given by:

   0.2123046
   0.8289244
   0.8289244

 The angle of fault current contribution from the line in degrees for each phase is given by:

  -90.
   172.64248
   7.3575197
 The magnitude of fault current contribution from motor in kA for each phase is given by:

   2.1230452
   20.91294
   20.91294

 The angle of fault current contribution from motor in degrees for each phase is given by:

   90.
   153.16579
   26.834209

-->
```

Figure 9.4: Double line to ground short circuit calculations using sequence networks

```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-9 ;Example 9.5
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;

Sb=100                                                      //Base value of system in MVA
Vb=13.8                                                     //Base voltage of system in kV
Vf=1.05                                                     //Prefault voltage in per unit
Z0=%i*0.250                                                 //Zero sequence impedance in per unit
Z1=%i*0.13893                                               //Positive sequence impedance in per unit
Z2=%i*0.14562                                               //Negative sequence impedance in per unit
Zf=0                                                        //Fault through impedance in per unit
Zpr=0.20                                                    //The positive sequence thevenin motor impedance at bus 2
Zpl=0.455                                                   //The positive sequence thevenin line impedance at bus 2
Znr=0.21                                                    //The negative sequence thevenin motor impedance at bus 2
Znl=0.475                                                   //The negative sequence thevenin line impedance at bus 2


```

```
I1=Vf/(Z1+((Z0*Z2)/(Z0+Z2)))                       //Positive sequence fault current in per unit
I2=-I1*(Z0/(Z0+Z2))                                //Negative sequence fault current in per unit
I0=-I1*(Z2/(Z0+Z2))                                //Zero sequence fault current in per unit
a=exp(%i*(120)*(%pi/180))                          //operator a
Isf=[1 1 1;1 (a^2) a;1 a (a^2)]*[I0;I1;I2]         //Subtransient fault current in each phase in per unit
Ib2=Isf*((Sb)/(Vb*sqrt(3)))                        //Using the base current 4.1837kA at bus 2 in kA
In=3*I0                                            //Neutral fault current in per unit
In2=In*((Sb)/(Vb*sqrt(3)))                         //Neutral fault current in kA
Iline0=0                                           //Zero sequence fault current from the line in per unit
Imotor0=I0                                         //Zero sequence motor current from the motor in per unit
Iline1=(Zpr/(Zpr+Zpl))*I1                          //Positive sequence fault current from the line in per unit
Imotor1=(Zpl/(Zpr+Zpl))*I1                         //Positive sequence motor current from the motor in per unit
Iline2=(Znr/(Znr+Znl))*I2                          //Negative sequence fault current from the line in per unit
Imotor2=(Znl/(Znr+Znl))*I2                         //Negative sequence motor current from the motor in per unit
Iline=[1 1 1;1 (a^2) a;1 a (a^2)]*[Iline0;Iline1;Iline2]  //transforming to the phase domain for the line
```

```
Ilineb=Iline*(0.41837)                        //Transforming to the phase domain with base currents of 0.41837 kA for the line in kA
Imotor=[1 1 1;1 (a^2) a;1 a (a^2)]*[Imotor0;Imotor1;Imotor2]  //transforming to the phase domain for the motor
Imotorb=Imotor*((Sb)/(Vb*sqrt(3)))            //Transforming to the phase domain with base currents of 4.1837 kA for the motor in kA

disp(abs(clean(Ib2,1e-10)),'The magnitude of subtransient fault current in each phase in kA is given by:');
disp(atand(clean(imag(Ib2),1e-10),clean(real(Ib2),1e-10)),'The angle of subtransient fault current in each phase in degrees is given by:');
printf('The magnitude neutral fault current is %.4f kA and its angle is %.4f degree\n',abs(In2),atand(imag(In2),real(In2)));
disp(abs(clean(Ilineb,1e-10)),'The magnitude of fault current contribution from the line in kA for each phase is given by: ');
disp(atand(clean(imag(Ilineb),1e-10),clean(real(Ilineb),1e-10)),'The angle of fault current contribution from the line in degrees for each phase is given by:');
disp(abs(clean(Imotorb,1e-10)),'The magnitude of fault current contribution from motor in kA for each phase is given by:');
disp(atand(clean(imag(Imotorb),1e-10),clean(real(Imotorb),1e-10)),'The angle of fault current contribution from motor in degrees for each phase is given by:');
```



```
The magnitude of transforming the line currents to the phase domain in per unit for each phase is given by:

  1.2166415
  2.268993
  1.2166415

The angle of transforming the line currents to the phase domain in degreess for each phase is given by:

 -21.174893
  180.
  21.174893

The magnitude of transforming the line currents to the phase domain in kA for each phase is given by:

  0.5090063
  0.9492786
  0.5090063

The angle of transforming the line currents to the phase domain in degreess for each phase is given by:

 -21.174893
  180.
  21.174893

-->|
```

Figure 9.5: Effect of star to delta transformer phase shift on fault currents

**Scilab code Exa 9.6** `Effect of star to delta transformer phase shift on fault curre`

```
//Book - Power system: Analysisi & Design 5th
   Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma,
   and Thomas J.Overbye
//Chapter-9 ;Example 9.6
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;


Vf=1.05

   //Prefault voltage in per unit
Z0=%i*0.250

   //Zero sequence impedance in per unit
Z1=%i*0.13893
```

```
                                         //Positive sequence impedance in per unit
Z2=%i*0.14562                            //Negative sequence impedance in per unit
Zf=0                                     //Fault through impedance in per unit
Zpr=0.20                                 //The positive sequence thevenin motor impedance at bus 2
Zpl=0.455                                //The positive sequence thevenin line impedance at bus 2
Znr=0.21                                 //The negative sequence thevenin motor impedance at bus 2
Znl=0.475                                //The negative sequence thevenin line impedance at bus 2


I1=Vf/(Z1+((Z0*Z2)/(Z0+Z2)))             //Positive sequence fault current in per unit
I2=-I1*(Z0/(Z0+Z2))                      //Negative sequence fault current in per unit
I0=-I1*(Z2/(Z0+Z2))                      //Zero sequence fault current in per unit
Iline0=0                                 //Zero sequence fault current from the line in
```

```
per unit
Imotor0=I0                                                   //Zero sequence motor current from the motor in per unit
Iline1=(Zpr/(Zpr+Zpl))*I1                                    //Positive sequence fault current from the line in per unit
Ilead1=Iline1*exp(%i*(30)*(%pi/180))                         //Positive sequence fault current from the line leads by 30 degree in per unit
Imotor1=(Zpl/(Zpr+Zpl))*I1                                   //Positive sequence motor current from the motor in per unit
Iline2=(Znr/(Znr+Znl))*I2                                    //Negative sequence fault current from the line in per unit
Ilag2=Iline2*exp(%i*(-30)*(%pi/180))                         //Negative sequence fault current from the line lags by 30 degree in per unit
Imotor2=(Znl/(Znr+Znl))*I2                                   //Negative sequence motor current from the motor in per unit
a=exp(%i*(120)*(%pi/180))                                    //operator a
Iline=[1 1 1;1 (a^2) a;1 a (a^2)]*[0;Ilead1;Ilag2]           //transforming the line currents to the phase domain
Ilineb=Iline*0.41837                                         //transforming the line currents to the phase domain with base currents of 0.41837 kA

disp(abs(clean(Iline,1e-10)),'The magnitude of transforming the line currents to the phase domain in per unit for each phase is given by:');
```

```
disp(atand(imag(Iline),real(Iline)),'The angle of transforming the line currents to the phase domain in degreess for each phase is given by:');
disp(abs(clean(Ilineb,1e-10)),'The magnitude of transforming the line currents to the phase domain in kA for each phase is given by:');
disp(atand(imag(Ilineb),real(Ilineb)),'The angle of transforming the line currents to the phase domain in degreess for each phase is given by:');
```

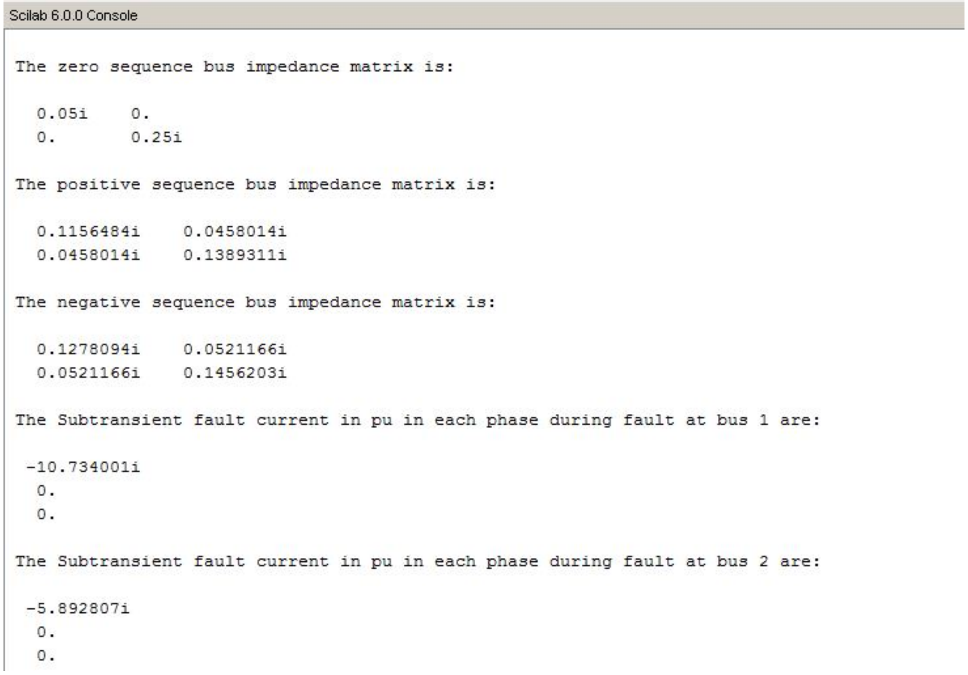
**Scilab code Exa 9.7** `Single line to ground short circuit calculations`

```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-9 ;Example 9.7
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;

Vf=1.05                                              //Prefault voltage in per unit
Zf=0                                                 //Fault through impedance in per unit


Ybus0=-%i*[20 0;0 4]                                 //zero sequence bus admittance matrix in per unit
Zbus0=inv(Ybus0)                                     //zero
```

```
    sequence bus impedance matrix in per unit
Ybus1=-%i*[9.9454 -3.2787;-3.2787 8.2787]                    //Positive sequence bus admittance matrix in per unit
Zbus1=inv(Ybus1)                                              //Positive sequence bus admittance matrix in per unit
Ybus2=-%i*[9.1611 -3.2787;-3.2787 8.0406]                    //Negative bus admittance matrix in per unit
Zbus2=inv(Ybus2)                                              //Negative sequence bus admittance matrix in per unit
Z110=%i*0.05                                                  //zero sequence impedance Z110 find out from the Zbus0
Z111=%i*0.11565                                               //positive sequence impedance Z111 find out from the Zbus1
Z112=%i*0.12781                                               //negative sequence impedance Z112 find out from the Zbus2
I10=Vf/(Z110+Z111+Z112)                                       //zeor sequence fault current at bus 1 in per unit
I11=I10                                                       //positive sequence fault current at bus 1 in per unit
I12=I11                                                       //Negative sequence fault current at bus 1 in per unit
```

```
a=exp(%i*120*%pi/180)                                   //operator a
Isf1=[1 1 1;1 (a^2) a;1 a (a^2)]*[I10;I11;I12]          //Subtransient fault current in each phase at bus 1 in per unit
Z220=%i*0.25                                            //zero sequence impedance Z220 find out from the Zbus0
Z221=%i*0.13893                                         //positive sequence impedance Z221 find out from the Zbus1
Z222=%i*0.14562                                         //negative sequence impedance Z222 find out from the Zbus2
I20=Vf/(Z220+Z221+Z222)                                 //zeor sequence fault current at bus 1 in per unit
I21=I20                                                 //positive sequence fault current at bus 1 in per unit
I22=I21                                                 //Negative sequence fault current at bus 1 in per unit
Isf2=[1 1 1;1 (a^2) a;1 a (a^2)]*[I20;I21;I22]          //Subtransient fault current in each phase at bus 2 in per unit
V1=[0;Vf;0]-[Z110 0 0;0 Z111 0;0 0 Z112]*[I10;I11;I12]       //The sequence components of the line to ground voltages at bus 1 during tha fault at bus 1 with k=1 and n=1 in per unit
V1lg=[1 1 1;1 (a^2) a;1 a (a^2)]*[V1]                   //The line to ground
```

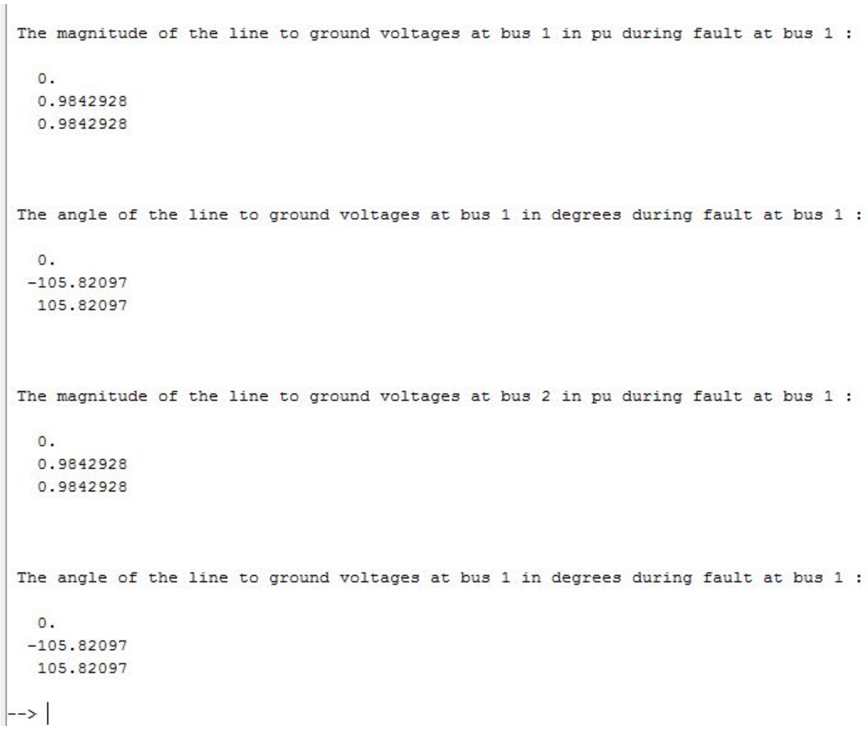
```
voltages at bus 1 during tha fault at bus 1 in per unit
Z210=%i*0.05                                         //zero sequence impedance Z210 find out from the Zbus0
Z211=%i*0.11565                                      //positive sequence impedance Z211 find out from the Zbus1
Z212=%i*0.12781                                      //negative sequence impedance Z212 find out from the Zbus2
V2=[0;Vf;0]-[Z210 0 0;0 Z211 0;0 0 Z212]*[I10;I11;I12]       //The sequence components of the line to ground voltages at bus 1 during tha fault at bus 2 with k=2 and n=1in per unit
V2lg=[1 1 1;1 (a^2) a;1 a (a^2)]*[V2]                //The line to ground voltages at bus 1 during tha fault at bus 1 in per unit



disp(clean(Zbus0,1e-10),'The zero sequence bus impedance matrix is:');
disp(clean(Zbus1,1e-10),'The positive sequence bus impedance matrix is:');
disp(clean(Zbus2,1e-10),'The negative sequence bus impedance matrix is:');
disp(clean(Isf1,1e-10),'The Subtransient fault current in pu in each phase during fault at bus 1 are:');
disp(clean(Isf2,1e-10),'The Subtransient fault current in pu in each phase during fault at bus 2 are:');
disp(abs(clean(V1lg,1e-10)),'The magnitude of the
```



```
The zero sequence bus impedance matrix is:

  0.05i    0.
  0.       0.25i

The positive sequence bus impedance matrix is:

  0.1156484i    0.0458014i
  0.0458014i    0.1389311i

The negative sequence bus impedance matrix is:

  0.1278094i    0.0521166i
  0.0521166i    0.1456203i

The Subtransient fault current in pu in each phase during fault at bus 1 are:

 -10.734001i
  0.
  0.

The Subtransient fault current in pu in each phase during fault at bus 2 are:

 -5.892807i
  0.
  0.
```

Figure 9.6: Single line to ground short circuit calculations

```
line to ground voltages at bus 1 in pu during fault at bus 1 :',);
disp(atand(clean(imag(V1lg),1e-10),clean(real(V1lg),1e-10)),'The angle of the line to ground voltages at bus 1 in degrees during fault at bus 1 :',);
disp(abs(clean(V2lg,1e-10)),'The magnitude of the line to ground voltages at bus 2 in pu during fault at bus 1 :',);
disp(atand(clean(imag(V2lg),1e-10),clean(real(V2lg),1e-10)),'The angle of the line to ground voltages at bus 1 in degrees during fault at bus 1 :',);
```



```
The magnitude of the line to ground voltages at bus 1 in pu during fault at bus 1 :

  0.
  0.9842928
  0.9842928



The angle of the line to ground voltages at bus 1 in degrees during fault at bus 1 :

  0.
 -105.82097
  105.82097



The magnitude of the line to ground voltages at bus 2 in pu during fault at bus 1 :

  0.
  0.9842928
  0.9842928



The angle of the line to ground voltages at bus 1 in degrees during fault at bus 1 :

  0.
 -105.82097
  105.82097

-->
```

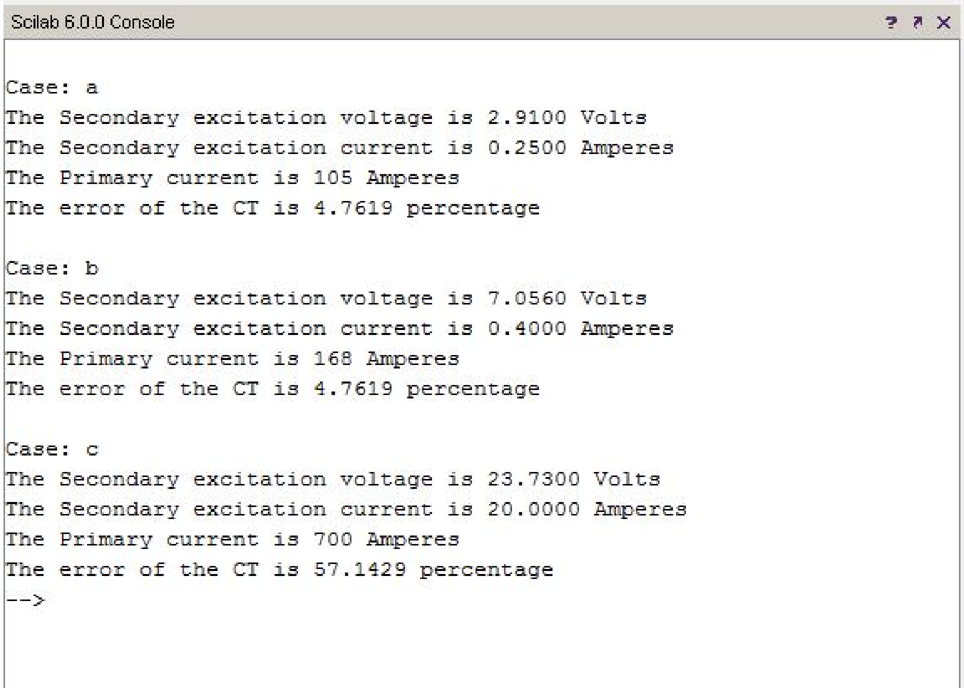
Figure 9.7: Single line to ground short circuit calculations

# Chapter 10

# SYSTEM PROTECTION

**Scilab code Exa 10.1** Current transformer performance

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 10 ; Example 10.1
//Scilab Version - 6.0.0 ; OS - Windows
clc;
clear;
CTratio=100/5;                                             //CT ratio
Zs=0.082;                                                  //Secondary resistance of a 100:5 CT in Ohm
IZB=[5 0.5; 8 0.8; 15 1.5];                                //Secondary output current in Amperes and burden resistance in Ohm
E=(Zs+IZB(1,2))*IZB(1,1);                                  //
```



```
Case: a
The Secondary excitation voltage is 2.9100 Volts
The Secondary excitation current is 0.2500 Amperes
The Primary current is 105 Amperes
The error of the CT is 4.7619 percentage

Case: b
The Secondary excitation voltage is 7.0560 Volts
The Secondary excitation current is 0.4000 Amperes
The Primary current is 168 Amperes
The error of the CT is 4.7619 percentage

Case: c
The Secondary excitation voltage is 23.7300 Volts
The Secondary excitation current is 20.0000 Amperes
The Primary current is 700 Amperes
The error of the CT is 57.1429 percentage
-->
```

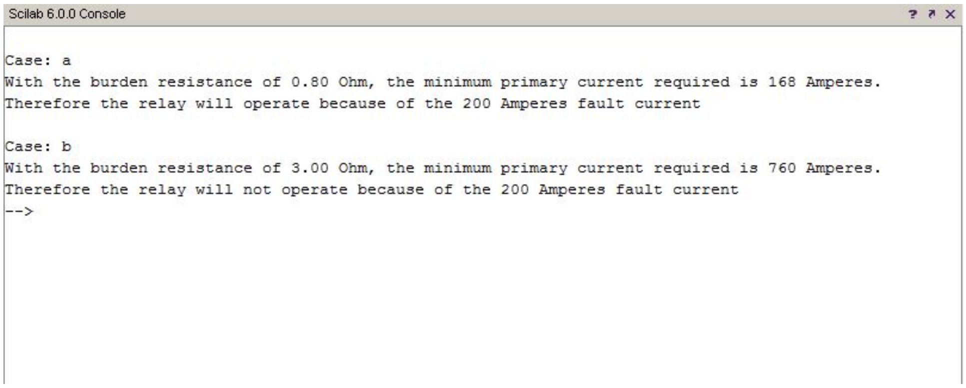
Figure 10.1: Current transformer performance

```
Secondary Excitation voltage in Volts
printf('\nCase: a');
printf('\nThe Secondary excitation voltage is %0.4f Volts',E);
Ie=0.25                                          //Secondary Excitation current for the secondary voltage from figure !0.8 in Amperes
printf('\nThe Secondary excitation current is %0.4f Amperes',Ie);
I=CTratio*(IZB(1,1)+Ie);                         //Primary current of the CT in Amperes
printf('\nThe Primary current is %d Amperes',I);
CTerr=Ie*100/(IZB(1,1)+Ie)';                     //Error in CT
printf('\nThe error of the CT is %0.4f percentage',CTerr);
E=(Zs+IZB(2,2))*IZB(2,1);                        //Secondary Excitation voltage in Volts
printf('\n\nCase: b');
printf('\nThe Secondary excitation voltage is %0.4f Volts',E);
Ie=0.4                                           //Secondary Excitation current for the secondary voltage from figure !0.8 in Amperes
printf('\nThe Secondary excitation current is %0.4f Amperes',Ie);
I=CTratio*(IZB(2,1)+Ie);                         //Primary current of the CT in Amperes
printf('\nThe Primary current is %d Amperes',I);
CTerr=Ie*100/(IZB(2,1)+Ie)';                     //Error in CT
```

```
printf('\nThe error of the CT is %0.4f percentage',CTerr);
E=(Zs+IZB(3,2))*IZB(3,1);                                         //Secondary Excitation voltage in Volts
printf('\n\nCase: c');
printf('\nThe Secondary excitation voltage is %0.4f Volts',E);
Ie=20                                                              //Secondary Excitation current for the secondary voltage from figure !0.8 in Amperes
printf('\nThe Secondary excitation current is %0.4f Amperes',Ie);
I=CTratio*(IZB(3,1)+Ie);                                          //Primary current of the CT in Amperes
printf('\nThe Primary current is %d Amperes',I);
CTerr=Ie*100/(IZB(3,1)+Ie)';                                      //Error in CT
printf('\nThe error of the CT is %0.4f percentage',CTerr);
```

**Scilab code Exa 10.2** `Relay operation versus fault current and CT burden`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 10 ; Example 10.2
//Scilab Version - 6.0.0 ; OS - Windows
clc;
clear;
```



```
Case: a
With the burden resistance of 0.80 Ohm, the minimum primary current required is 168 Amperes.
Therefore the relay will operate because of the 200 Amperes fault current

Case: b
With the burden resistance of 3.00 Ohm, the minimum primary current required is 760 Amperes.
Therefore the relay will not operate because of the 200 Amperes fault current
-->
```

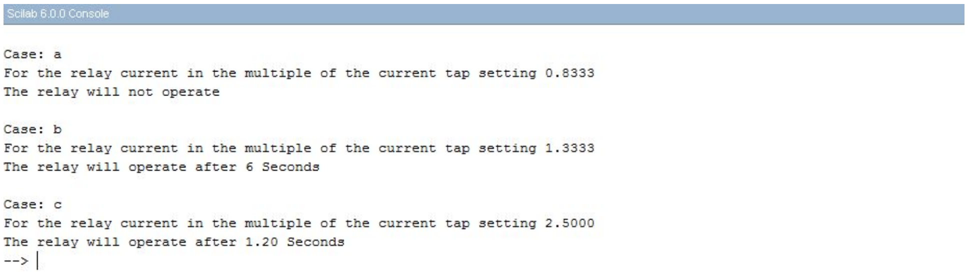
Figure 10.2: Relay operation versus fault current and CT burden

```
Irelay=200                        //Current through the relay in Amperes
CTratio=100/5;                    //CT ratio
Zs=0.082;                         //Secondary resistance of a 100:5 CT in Ohm
IZB=[8 0.8; 8 3];                 //Secondary output current in Amperes and burden resistance in Ohm
E=(Zs+IZB(1,2))*IZB(1,1);         //Secondary Excitation voltage in Volts
Ie=0.40                           //Secondary Excitation current for the secondary voltage from figure !0.8 in Amperes
I=CTratio*(IZB(1,1)+Ie);          //Primary current of the CT in Amperes
printf('\nCase: a');
if (Irelay>I) then
    printf('\nWith the burden resistance of %0.2f Ohm, the minimum primary current required is %d Amperes.\nTherefore the relay will operate because of the 200 Amperes fault current', IZB(1,2),I)
else
    printf('\nWith the burden resistance of %0.2f Ohm, the minimum primary current required is
```

```
    %d Amperes.\nTherefore the relay will not operate because of the 200 Amperes fault current',IZB(1,2),I);
end
E=(Zs+IZB(2,2))*IZB(2,1);    //Secondary Excitation voltage in Volts
Ie=30                        //Secondary Excitation current for the secondary voltage from figure !0.8 in Amperes
I=CTratio*(IZB(2,1)+Ie);     //Primary current of the CT in Amperes
printf('\n\nCase: b');
if (Irelay>I) then
    printf('\nWith the burden resistance of %0.2f Ohm, the minimum primary current required is %d Amperes.\nTherefore the relay will operate because of the 200 Amperes fault current',IZB(2,2),I)
else
    printf('\nWith the burden resistance of %0.2f Ohm, the minimum primary current required is %d Amperes.\nTherefore the relay will not operate because of the 200 Amperes fault current',IZB(2,2),I);
end
```

**Scilab code Exa 10.3** Operating time for a CO 8 time delay over current relay

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 10 ; Example 10.3
//Scilab Version - 6.0.0 ; OS - Windows
```

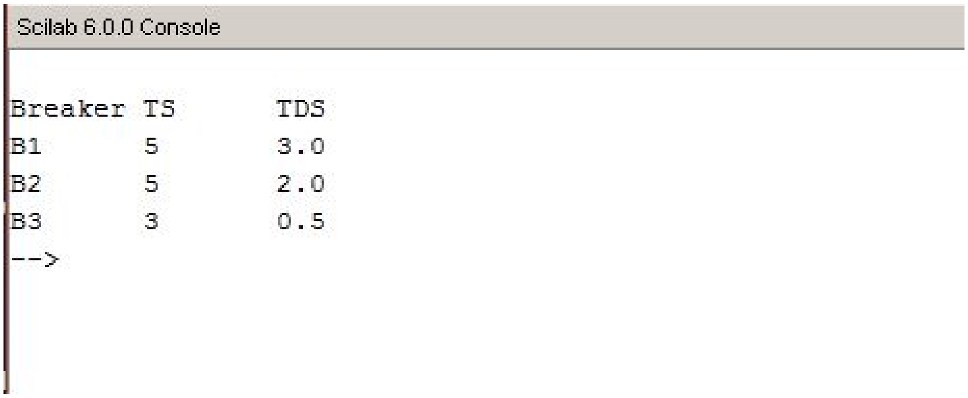


```
Case: a
For the relay current in the multiple of the current tap setting 0.8333
The relay will not operate

Case: b
For the relay current in the multiple of the current tap setting 1.3333
The relay will operate after 6 Seconds

Case: c
For the relay current in the multiple of the current tap setting 2.5000
The relay will operate after 1.20 Seconds
--> |
```

Figure 10.3: Operating time for a CO 8 time delay over current relay

```
clc;
clear;
Crnttap=6;  //Current tap setting in Amperes
TDsetting=1;  //Time dial setting
CTratio=100/5;  //CT ratio
IZB=[5 0.5; 8 0.8; 15 1.5];  //Secondary output current in Amperes and burden resistance in Ohm
RC_multiple_Crntap=IZB(1,1)/Crnttap;    //Relay current in the multiple of the current tap setting
printf('\nCase: a');
if (RC_multiple_Crntap<1) then
    printf('\nFor the relay current in the multiple of the current tap setting %0.4f \nThe relay will not operate',RC_multiple_Crntap);
else
    printf('\nFor the relay current in the multiple of the current tap setting %0.4f \nThe relay will operate after %0.2f Seconds',RC_multiple_Crntap,time);
end
```

```
RC_multiple_Crntap=IZB(2,1)/Crnttap;  //Relay current in the multiple of the current tap setting
time=6                                    //Relay operating time from figure 10.12 in Seconds
printf('\n\nCase: b');
if (RC_multiple_Crntap<1) then
    printf('\nFor the relay current in the multiple of the current tap setting %0.4f \nThe relay will not operate',RC_multiple_Crntap);
else
    printf('\nFor the relay current in the multiple of the current tap setting %0.4f \nThe relay will operate after %d Seconds',RC_multiple_Crntap,time);
end
RC_multiple_Crntap=IZB(3,1)/Crnttap;  //Relay current in the multiple of the current tap setting
time=1.2                                  //Relay operating time from figure 10.12 in Seconds
printf('\n\nCase: c');
if (RC_multiple_Crntap<1) then
    printf('\nFor the relay current in the multiple of the current tap setting %0.4f \nThe relay will not operate',RC_multiple_Crntap);
else
    printf('\nFor the relay current in the multiple of the current tap setting %0.4f \nThe relay will operate after %0.2f Seconds',RC_multiple_Crntap,time);
end
```



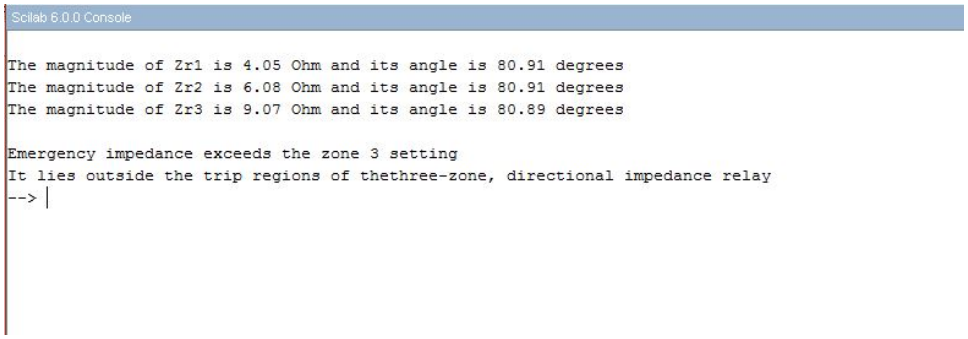
```
Breaker TS      TDS
B1       5      3.0
B2       5      2.0
B3       3      0.5
-->
```

Figure 10.4: Coordinating time delay over current relays in a radial system

**Scilab code Exa 10.4** `Coordinating time delay over current relays in a radial syste`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 10 ; Example 10.4
//Scilab Version - 6.0.0 ; OS - Windows
clc;
clear;
S_Ifmax_CTratio=[11 3000 400/5;4 2000 200/5;6 100 200/5];                    //Apparent power in MVA, maximum fault current in Amperes and CT ratio
V=34.5;                                                                       //RMS line to line voltage in kVolts
Tbreaker=0.083;                                                               //Operating time of breaker for 5 cycles in Second
Tcoordination=0.3;                                                            //Co-ordination time of the breaker in Seconds
I13=S_Ifmax_CTratio(3,1)*10^(3)/(V*sqrt(3)*S_Ifmax_CTratio(3,3));            //Maximum
```

```
   secondary current of breaker 3 in Ampere
Ts3=3;
   //From figure 10.12 the Tap Setting
Il2=(S_Ifmax_CTratio(2,1)+S_Ifmax_CTratio(3,1))
   *10^(3)/(V*sqrt(3)*S_Ifmax_CTratio(2,3));//
   Maximum secondary current of breaker 2 in Ampere
Ts2=5;
   //From figure 10.12 the Tap Setting
Il1=(S_Ifmax_CTratio(1,1)+S_Ifmax_CTratio(2,1)+
   S_Ifmax_CTratio(3,1))*10^(3)/(V*sqrt(3)*
   S_Ifmax_CTratio(1,3));//Maximum secondary current
    of breaker 1 in Ampere
Ts1=5;
   //From figure 10.12 the Tap Setting
Fault_pickupcrnt3=S_Ifmax_CTratio(2,2)/(Ts3*
   S_Ifmax_CTratio(3,3));          //The fault-to-
   pickup current ratio at Breaker 3
t3=0.05;
   //Relay operating time from figure 10.12 in
   Seconds
tds3=0.5;
   //Time-dial settings from figure 10.12
Fault_pickupcrnt2=S_Ifmax_CTratio(2,2)/(Ts2*
   S_Ifmax_CTratio(2,3));          //The fault-to-
   pickup current ratio at Breaker 2
t2=t3+Tbreaker+Tcoordination;
tds2=2;
   //Time-dial settings from figure 10.12
Fault_pickupcrnt2=S_Ifmax_CTratio(1,2)/(Ts2*
   S_Ifmax_CTratio(2,3));          //The fault-to-
   pickup current ratio at Breaker 1
t2=0.38;
```



```
The magnitude of Zr1 is 4.05 Ohm and its angle is 80.91 degrees
The magnitude of Zr2 is 6.08 Ohm and its angle is 80.91 degrees
The magnitude of Zr3 is 9.07 Ohm and its angle is 80.89 degrees

Emergency impedance exceeds the zone 3 setting
It lies outside the trip regions of thethree-zone, directional impedance relay
--> |
```

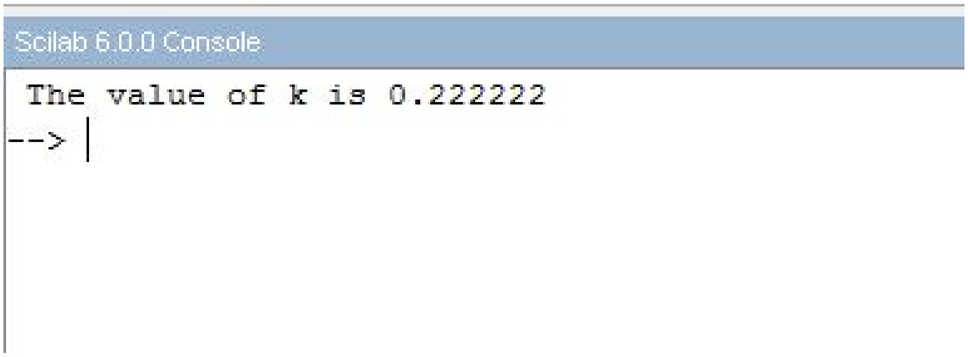
Figure 10.5: Three zone impedance relay settings

```
    //Relay operating time from figure 10.12 in Seconds
tds1=3;
    //Time-dial settings from figure 10.12
Fault_pickupcrnt1=S_Ifmax_CTratio(1,2)/(Ts2*S_Ifmax_CTratio(1,3));
t1=t2+Tbreaker+Tcoordination;
printf('\nBreaker\tTS\tTDS');
printf('\nB1\t%d\t%0.1f',Ts1,tds1);
printf('\nB2\t%d\t%0.1f',Ts2,tds2);
printf('\nB3\t%d\t%0.1f',Ts3,tds3);
```

**Scilab code Exa 10.8** `Three zone impedance relay settings`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 10 ; Example 10.8
//Scilab Version - 6.0.0 ; OS - Windows
```

```
clc;
clear;
Vln=345;
                                        //Source voltage in kVolts
CTratio=1500/5;
                                        //CT ratio
VTratio=3000/1;
                                        //VT ratio
Imax=1500;
                                        //Maximum current during emergency loading in
    Amperes
pf=0.95;
                                        //Power factor
positivesequence=[8+%i*50;8+%i*50;5.3+%i*33;4.3+%i
    *27];          //Positive sequence impedance in
    Ohms
Zsec=CTratio/VTratio;
                                                            //
    Secondary impedance with respect to primary
    impedance in Ohms
Zr1=0.8*positivesequence(1)*Zsec;
                                          //B12 zone 1 relay
    for 80% reach in Ohms
Zr2=1.2*positivesequence(2)*Zsec;
                                          //B12 zone 2 relay
    for 120% reach in Ohms
Zr3=(positivesequence(3)*1.2+positivesequence(2))*
    Zsec          //B12 zone 3 relay for 100% reach
    of line 1  2  and 120% reach of line 2  4  in
    Ohms
Z=(Vln*10^(3)*Zsec/sqrt(3))/(Imax*exp(-%i*acos(pf)))
    ;
printf('\nThe magnitude of Zr1 is %0.2f Ohm and its
```



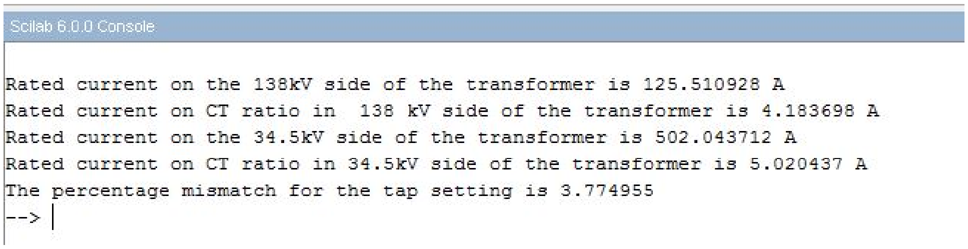
```
 The value of k is 0.222222
-->
```

Figure 10.6: Differential relay protection for a single phase transformer

```
    angle is %0.2f degrees',abs(Zr1),atand(imag(Zr1),real(Zr1)));
printf('\nThe magnitude of Zr2 is %0.2f Ohm and its angle is %0.2f degrees',abs(Zr2),atand(imag(Zr2),real(Zr2)));
printf('\nThe magnitude of Zr3 is %0.2f Ohm and its angle is %0.2f degrees\n',abs(Zr3),atand(imag(Zr3),real(Zr3)));
if abs(Z)>abs(Zr3) then
    printf('\nEmergency impedance exceeds the zone 3 setting\nIt lies outside the trip regions of thethree-zone, directional impedance relay');
else
    printf('\nEmergency impedance does not exceed the zone 3 setting\nIt lies inside the trip regions of thethree-zone, directional impedance relay');
end
```

**Scilab code Exa 10.9** `Differential relay protection for a single phase transformer`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 10 ; Example 10.9
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;

Srated = 10;                                    //power rating in MVA
Vprtr = 80;                                     //primary side of transformer voltage in kV
Vsectr = 20;                                    //secondary side of transformer voltage in kV
CTratiopr = 150/5;                              //primary CT ratio
CTratiosec = 600/5;                             //secondary CT ratio
I1rated = (Srated*10^6)/(Vprtr*10^3);           //rated current 1 in Amperes
I2rated = (Srated*10^6)/(Vsectr*10^3);          //rated current 2 in Amperes
I1 = I1rated/CTratiopr;                         //differential current 1 in Amperes
I2 = I2rated/CTratiosec;                        //differential current 2 in Amperes
I = I1-I2;                                      //differential current at rated conditions in Amperes
```



```
Rated current on the 138kV side of the transformer is 125.510928 A
Rated current on CT ratio in  138 kV side of the transformer is 4.183698 A
Rated current on the 34.5kV side of the transformer is 502.043712 A
Rated current on CT ratio in 34.5kV side of the transformer is 5.020437 A
The percentage mismatch for the tap setting is 3.774955
--> |
```

Figure 10.7: Differential relay protection for a three phase transformer

```
k = 0.5/2.25;                                   //from figure 10.34
printf('The value of k is %f',k);
```

---

**Scilab code Exa 10.10** `Differential relay protection for a three phase transformer`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 10 ; Example 10.10
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;

Srated = 30;                                    //power rating in MVA
Vprtr = 34.5;                                   //primary side of transformer voltage in kV
Vsectr = 138;                                   //
```

```
secondary side of transformer voltage in kV
IArated = (Srated*10^6)/(sqrt(3)*Vsectr*10^3);            //Rated current on the 138-kV side of the transformer in Amperes
CTratiosec = 150/5;                                                   //CT ratio on the 138-kV side
IA = IArated/CTratiosec;                                         //differential current in 138kV side in Amperes
Iarated = (Srated*10^6)/(sqrt(3)*Vprtr*10^3);              //Rated current on the 34.5-kV side of the transformer in Amperes
CTratiopr = 500/5;                                                    //CT ratio on the 34.5-kV side
Ia = Iarated/CTratiopr;                                          //differential current in 138kV side in Amperes
Iab = Ia*sqrt(3);                                                          //diffrential current in lefthand re-straining winding of figure 10.37 in Amperes
crtratio = Iab/IA;                                                       //ratio of the currents in the left- to righthand restraining winding
TA = 5;
Tab = 10;
tapratio = Tab/TA;                                                       //closest relay tap ratio
%mismatch = (((Iab/Tab)-(IA/TA))/(Iab/Tab))*100;        //percentage mismatch for tap setting
printf('\nRated current on the 138kV side of the transformer is %f A',IArated);
printf('\nRated current on CT ratio in  138 kV side of the transformer is %f A',IA);
```

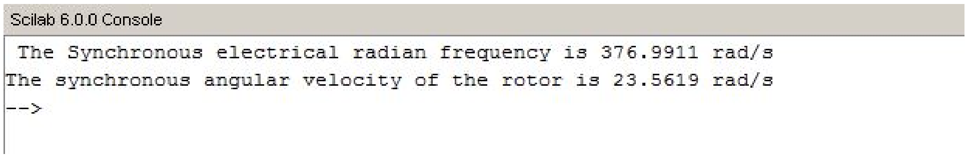
```
printf('\nRated current on the 34.5kV side of the
   transformer is %f A',Iarated);
printf('\nRated current on CT ratio in 34.5kV side
   of the transformer is %f A',Ia);
printf('\nThe percentage mismatch for the tap
   setting is %f',%mismatch);
```

# Chapter 11

# TRANSIENT STABILITY

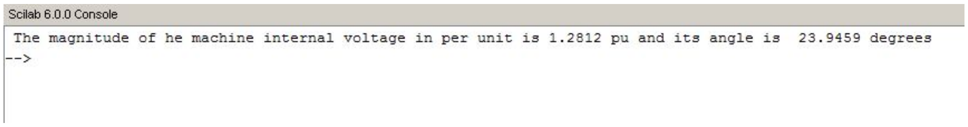
**Scilab code Exa 11.1** `Generator per unit swing equation and power angle during a sh`

```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-11 ;Example 11.1
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;

f=60                         //frequency of hydroelectric
```



```
 The Synchronous electrical radian frequency is 376.9911 rad/s
The synchronous angular velocity of the rotor is 23.5619 rad/s
-->
```

Figure 11.1: Generator per unit swing equation and power angle during a short circuit



```
The magnitude of he machine internal voltage in per unit is 1.2812 pu and its angle is  23.9459 degrees
-->
```

Figure 11.2: Generator internal voltage and real power output versus power angle

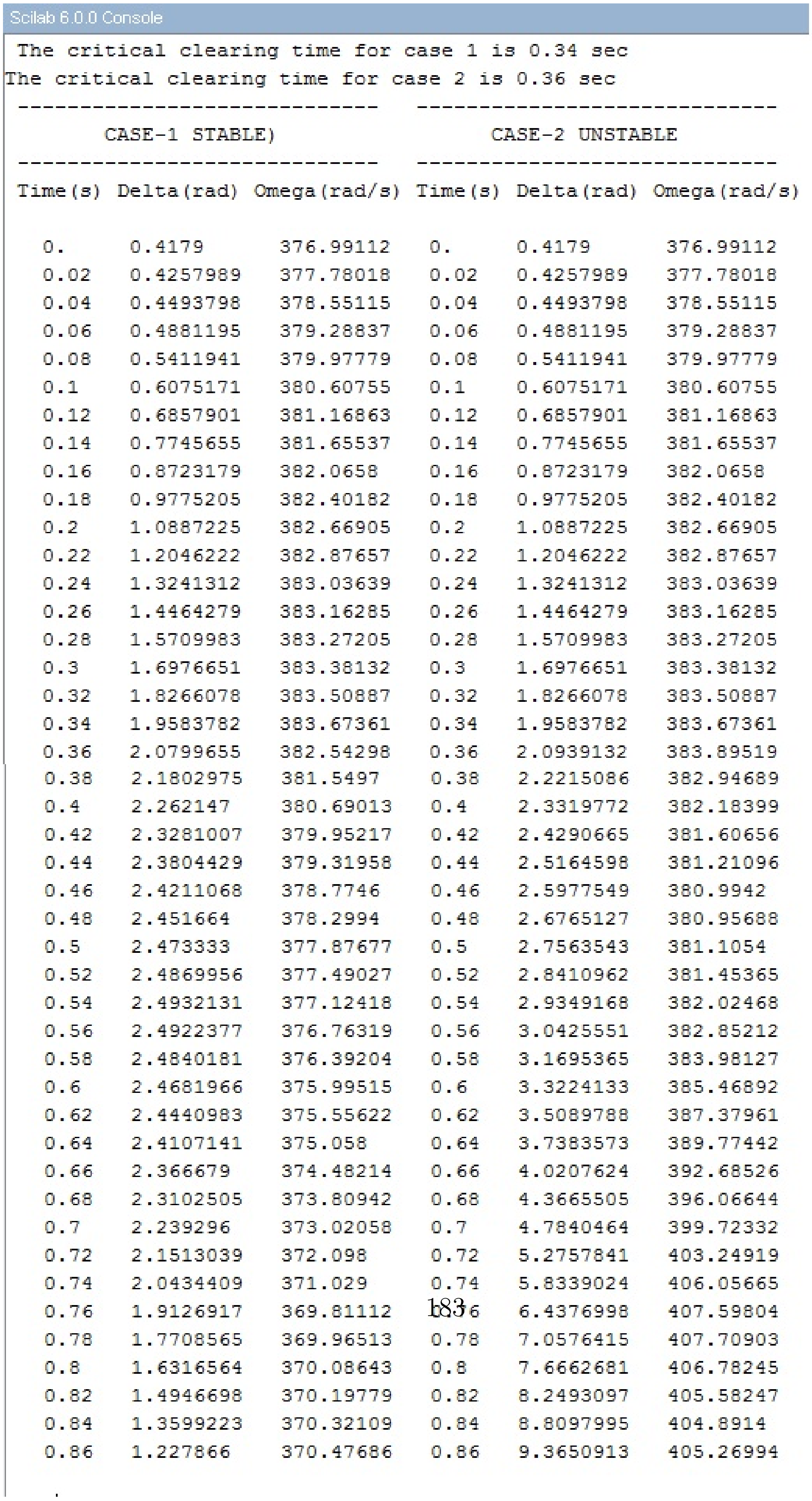
```
generating unit
Pr=500                 //rated power of hydroelectric generator
V=5                    //rated voltage of hrdroelectric generator
p=32                   //pole of hydroelectric generating unit
H=2.0                  //Inertia constant in per unit-seconds

Wsyn=2*%pi*f           //Synchronous electrical radian frequency in rad/s
Wmsyn=(2/p)*Wsyn       //synchronous angular velocity of the rotor in rad/s

printf('The Synchronous electrical radian frequency is %.4f rad/s\n',Wsyn);
printf('The synchronous angular velocity of the rotor is %.4f rad/s',Wmsyn);
```

### Scilab code Exa 11.3 `Generator internal voltage and real power output versus power`

```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
```

```
//Chapter-11 ;Example 11.3
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;

P=1.0                                      //Infinite bus received real power in per unit
Vbus=1.0                                   //Infinite bus voltage in per unit
pf=0.95                                    //Lagging power factor
Xdt=0.30
XTR=0.10
X12=0.20
X13=0.10
X23=0.20

Xeq=Xdt+XTR+(X12*(X13+X23))/(X12+(X13+X23));    //The equialent reactance between the machine internal voltage and infinite bus in per unit
theta=acos(pf);
I=(P/(Vbus*pf))*exp(-%i*theta);                 //Current into the infinite bus in per unit
Ei=Vbus+(%i*Xeq)*I;                             //The machine internal voltage in per unit

printf('The magnitude of he machine internal voltage in per unit is %.4f pu and its angle is  %.4f degrees',abs(Ei),atand(imag(Ei),real(Ei)));
```

**Scilab code Exa 11.7** Eulers method computer solution to swing equation and critica



```
 The critical clearing time for case 1 is 0.34 sec
The critical clearing time for case 2 is 0.36 sec
 -----------------------------   -----------------------------
       CASE-1 STABLE)                  CASE-2 UNSTABLE
 -----------------------------   -----------------------------
 Time(s) Delta(rad) Omega(rad/s) Time(s) Delta(rad) Omega(rad/s)


   0.     0.4179      376.99112  0.     0.4179      376.99112
   0.02   0.4257989   377.78018  0.02   0.4257989   377.78018
   0.04   0.4493798   378.55115  0.04   0.4493798   378.55115
   0.06   0.4881195   379.28837  0.06   0.4881195   379.28837
   0.08   0.5411941   379.97779  0.08   0.5411941   379.97779
   0.1    0.6075171   380.60755  0.1    0.6075171   380.60755
   0.12   0.6857901   381.16863  0.12   0.6857901   381.16863
   0.14   0.7745655   381.65537  0.14   0.7745655   381.65537
   0.16   0.8723179   382.0658   0.16   0.8723179   382.0658
   0.18   0.9775205   382.40182  0.18   0.9775205   382.40182
   0.2    1.0887225   382.66905  0.2    1.0887225   382.66905
   0.22   1.2046222   382.87657  0.22   1.2046222   382.87657
   0.24   1.3241312   383.03639  0.24   1.3241312   383.03639
   0.26   1.4464279   383.16285  0.26   1.4464279   383.16285
   0.28   1.5709983   383.27205  0.28   1.5709983   383.27205
   0.3    1.6976651   383.38132  0.3    1.6976651   383.38132
   0.32   1.8266078   383.50887  0.32   1.8266078   383.50887
   0.34   1.9583782   383.67361  0.34   1.9583782   383.67361
   0.36   2.0799655   382.54298  0.36   2.0939132   383.89519
   0.38   2.1802975   381.5497   0.38   2.2215086   382.94689
   0.4    2.262147    380.69013  0.4    2.3319772   382.18399
   0.42   2.3281007   379.95217  0.42   2.4290665   381.60656
   0.44   2.3804429   379.31958  0.44   2.5164598   381.21096
   0.46   2.4211068   378.7746   0.46   2.5977549   380.9942
   0.48   2.451664    378.2994   0.48   2.6765127   380.95688
   0.5    2.473333    377.87677  0.5    2.7563543   381.1054
   0.52   2.4869956   377.49027  0.52   2.8410962   381.45365
   0.54   2.4932131   377.12418  0.54   2.9349168   382.02468
   0.56   2.4922377   376.76319  0.56   3.0425551   382.85212
   0.58   2.4840181   376.39204  0.58   3.1695365   383.98127
   0.6    2.4681966   375.99515  0.6    3.3224133   385.46892
   0.62   2.4440983   375.55622  0.62   3.5089788   387.37961
   0.64   2.4107141   375.058    0.64   3.7383573   389.77442
   0.66   2.366679    374.48214  0.66   4.0207624   392.68526
   0.68   2.3102505   373.80942  0.68   4.3665505   396.06644
   0.7    2.239296    373.02058  0.7    4.7840464   399.72332
   0.72   2.1513039   372.098    0.72   5.2757841   403.24919
   0.74   2.0434409   371.029    0.74   5.8339024   406.05665
   0.76   1.9126917   369.81112  0.76   6.4376998   407.59804
   0.78   1.7708565   369.96513  0.78   7.0576415   407.70903
   0.8    1.6316564   370.08643  0.8    7.6662681   406.78245
   0.82   1.4946698   370.19779  0.82   8.2493097   405.58247
   0.84   1.3599223   370.32109  0.84   8.8097995   404.8914
   0.86   1.227866    370.47686  0.86   9.3650913   405.26994
```

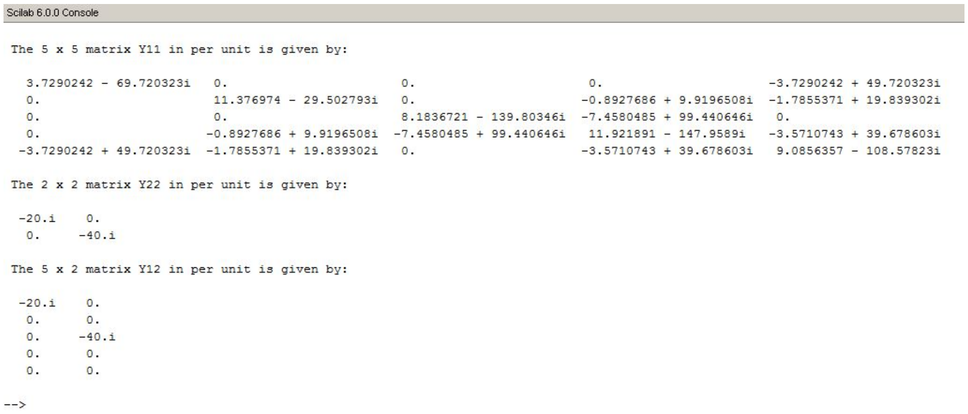
```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-11 ;Example 11.7
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;

function result=table(delcr)           //Function to get result in table format using Eulers method for diferent critical clearing angles
delta=0.4179                           //Initial value of delta in rad taken from example 7.6
omega=2*%pi*60                         //Initial value of omega in rad/s
H=3                                    //Value of H constant in pu-s
omegasyn=omega
t=0;
delt=0.02                              //Step size
result=[];                             //Initialization of result table
tc=0;                                  //Initialization of critical clearing time
while t<0.861                          //Maximum time for Eler's method is o.86
        result=[result;t delta omega]          //Updating the result table
        ddeltat=omega-omegasyn          //Calculation of ddeltat/dt using equation 11.4.7
        deltab=delta+ddeltat*delt          //Calculation of delta_bar using equation 11.4.9

            if delta<delcr
```

```
                 //Steps to calculate accelerating power for prefault condition
                 papu=1-0.9152*sin(delta)
                 pafb=1-0.9152*sin(deltab)
             else   //Steps to calculate accelerating power for postfault condition
                 tc=tc+1;
                 papu=1-2.1353*sin(delta)
                 pafb=1-2.1353*sin(deltab)
             end
        if tc==1 & delcr==1.95   //Displaying result of case 1(Stable) stable with Critical clearing angle of 1.95
            printf('The critical clearing time for case 1 is %.2f sec\n',t)
        elseif tc==1 & delcr==2.09   //Displaying result of case 2(Unstable) stable with Critical clearing angle of 2.09
            printf('The critical clearing time for case 2 is %.2f sec\n',t)
        end
        domegat=papu*omegasyn*omegasyn/(2*H*omega)          //Calculation of domegat/dt using equation 11.4.8
        omegab=omega+domegat*delt                            //Calculation of omega_bar using equation 11.4.10
        ddeltab=omegab-omegasyn                              //Calculation of ddelta_bar/dt using equation 11.4.11
        domegab=pafb*omegasyn*omegasyn/(2*H*omegab)         //Calculation of domega_bar/dt using equation 11.4.12
        delta=delta+(ddeltat+ddeltab)*delt/2                 //Calculation of delta for
```

```
change in time using equation 11.4.13
        omega=omega+(domegat+domegab)*delt/2                //Calculation of omega for change in time using equation 11.4.14
        t=t+delt;
end
endfunction

case1=table(1.95)                                //case1 - critical clearing angle is 1.95 rad
case2=table(2.09)                                //case2 - critical clearing angle is 2.09 rad
printf('------------------------------------------------------------------------\n')
printf('          CASE-1 STABLE                              CASE-2 UNSTABLE          \n')
printf('------------------------------------------------------------------------')
disp([case1 case2],'Time(s) Delta(rad) Omega(rad/s) Time(s) Delta(rad) Omega(rad/s)')
```

**Scilab code Exa 11.8** `Modifying power flow Ybus for application to multi machine st`

```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-11 ;Example 11.8
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;
linedata=[2 4 0.0090   0.10   1.72         //Entering
```



```
 The 5 x 5 matrix Y11 in per unit is given by:

   3.7290242 - 69.720323i   0.                       0.                       0.                      -3.7290242 + 49.720323i
   0.                       11.376974 - 29.502793i   0.                      -0.8927686 + 9.9196508i  -1.7855371 + 19.839302i
   0.                       0.                       8.1836721 - 139.80346i  -7.4580485 + 99.440646i   0.
   0.                      -0.8927686 + 9.9196508i  -7.4580485 + 99.440646i   11.921891 - 147.9589i   -3.5710743 + 39.678603i
  -3.7290242 + 49.720323i  -1.7855371 + 19.839302i   0.                      -3.5710743 + 39.678603i   9.0856357 - 108.57823i

 The 2 x 2 matrix Y22 in per unit is given by:

  -20.i    0.
   0.     -40.i

 The 5 x 2 matrix Y12 in per unit is given by:

  -20.i    0.
   0.      0.
   0.     -40.i
   0.      0.
   0.      0.

-->
```

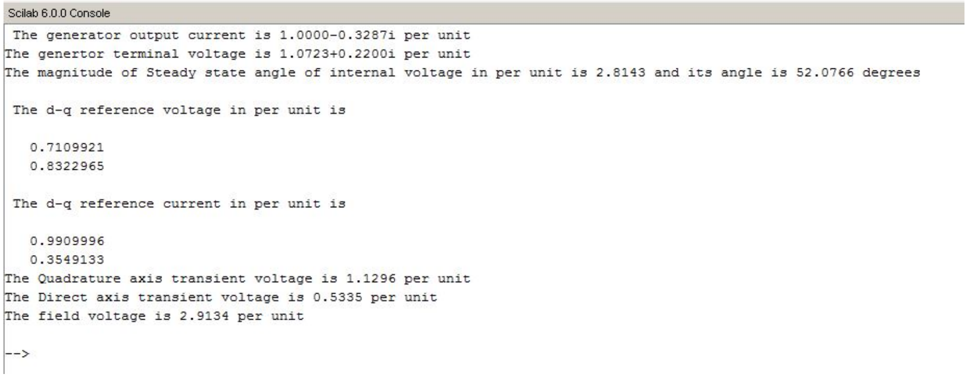
Figure 11.4: Modifying power flow Ybus for application to multi machine stability

```
   line data from table 6.2 & 6.3 of example 6.9
          2 5 0.0045   0.05  0.88
          4 5 0.00225  0.025 0.44
          1 5 0.00150  0.02  0.00
          3 4 0.00075  0.01  0.00];
sb= linedata(:,1);
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 since conductsnce G=0 for all lines
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)/2;
    ybus(n,n)=ybus(n,n)+1/lz(i)+sa(i)/2;
    ybus(m,n)=-1/lz(i);
```

```
    ybus(n,m)=ybus(m,n);
end
P13=0.8;Q13=0.4; P12=8; Q12=2.8;      //Data taken from table 6.1
V3=1.05;V2=0.959;
Qc=184;                               //Capacity of shunt capacitor in kVAR.
xd1dash=0.05;
xd2dash=0.025;
Y13=(P13-%i*Q13)/V3^2;
Y12=(P12-%i*(Q12-Qc/100))/V2^2;
Yd1=1/(%i*xd1dash);           //The inverted generator impedances for machine 1 connected to bus 1
Yd2=1/(%i*xd2dash);           //The inverted generator impedances for machine 2 connected to bus 3

//Updation of bus admittance matrix
Y11=ybus;
Y11(1,1)=Y11(1,1)+Yd1;
Y11(2,2)=Y11(2,2)+Y12;
Y11(3,3)=Y11(3,3)+Y13+Yd2;
disp(Y11,'The 5 x 5 matrix Y11 in per unit is given by:')
Y22=[Yd1 0;0 Yd2];
disp(Y22,'The 2 x 2 matrix Y22 in per unit is given by:')
Y12=[Yd1 0;0 0;0 Yd2;0 0;0 0];
disp(Y12,'The 5 x 2 matrix Y12 in per unit is given by:')
```

---

**Scilab code Exa 11.10** `Two Axis Model Example`



```
 The generator output current is 1.0000-0.3287i per unit
The genertor terminal voltage is 1.0723+0.2200i per unit
The magnitude of Steady state angle of internal voltage in per unit is 2.8143 and its angle is 52.0766 degrees

 The d-q reference voltage in per unit is

   0.7109921
   0.8322965

 The d-q reference current in per unit is

   0.9909996
   0.3549133
The Quadrature axis transient voltage is 1.1296 per unit
The Direct axis transient voltage is 0.5335 per unit
The field voltage is 2.9134 per unit

-->
```

Figure 11.5: Two Axis Model Example

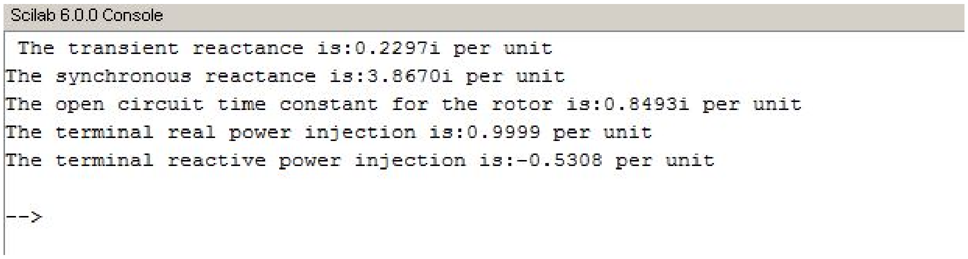
```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-11 ;Example 11.10
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;

P=1.0        //Infinite bus received real power in per unit
Vbus=1.0     //Infinite bus voltage in per unit
Vr=1.0       //system voltage in per unit
pf=0.95      //Lagging power factor
Ra=0         //Machine resistance in per unit
```

```
Xd=2.1

   //direct axis reactance in per unit
Xq=2.0

   //qadrature axis reactance in per unit
Xdt=0.3

   //direct axis transient reactance in per unit
Xqt=0.5

   //qadrature axis transient reactance in per unit
X=%i*0.22

theta=acos(pf);
I=(P/(Vbus*pf))*exp(-%i*theta);
                                 //generator
   output current in per unit
VT=Vr+X*I

   //genertor terminal voltage in per unit
Ireal=1

   //generator real output current in per unit
Iimag=-0.3287

   //Generator imaginary output voltage in per unit
Vreal=1.0723

   //generator real terminal voltage in per unit
Vimag=0.220

   //Generator imaginary terminal voltage
Ei=VT+(%i*Xq)*I

   //Steady state angle of internal voltage in per
   unitge
del=52.1*%pi/180
```

```
Vdq=[sin(del) -cos(del);cos(del) sin(del)]*[Vreal;Vimag];        //d-q reference voltage
Idq=[sin(del) -cos(del);cos(del) sin(del)]*[Ireal;Iimag];        //d-q reference current
Eqs=Vdq(2)+Xdt*Idq(1)                    //Quadrature axis transient voltage
Eds=Vdq(1)-Xqt*Idq(2)                    //Direct axis transient voltage
Efd=Eqs+(Xd-Xdt)*Idq(1)                                                  //field voltage

printf('The generator output current is %.4f%.4fi per unit\n',real(I),imag(I));
printf('The genertor terminal voltage is %.4f+%.4fi per unit\n',real(VT),imag(VT));
printf('The magnitude of Steady state angle of internal voltage in per unit is %.4f and its angle is %.4f degrees\n',abs(Ei),atand(imag(Ei),real(Ei)));
disp(Vdq,'The d-q reference voltage in per unit is');
disp(Idq,'The d-q reference current in per unit is');
printf('The Quadrature axis transient voltage is %.4f per unit\n',Eqs);
printf('The Direct axis transient voltage is %.4f per unit\n',Eds);
printf('The field voltage is %.4f per unit\n',Efd);
```

---

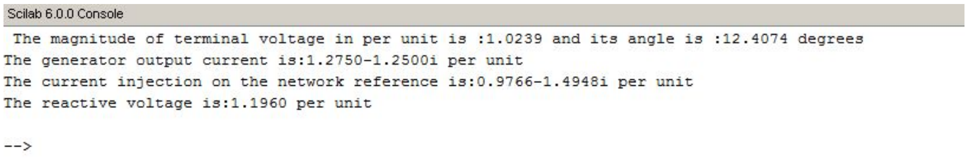
**Scilab code Exa 11.11** `Induction Generator Example`



```
 The transient reactance is:0.2297i per unit
The synchronous reactance is:3.8670i per unit
The open circuit time constant for the rotor is:0.8493i per unit
The terminal real power injection is:0.9999 per unit
The terminal reactive power injection is:-0.5308 per unit

-->
```

Figure 11.6: Induction Generator Example

```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-11 ;Example 11.11
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;

f=60                                  //genertor frequency
H=0.9                                 //Inertia constant in per unit-seconds
Ra=0.013
Xa=0.067                              //leakage reactance
Xm=3.8
R1=0.0124
X1=0.17
S=-0.0111                             //slip
Ert=0.9314
Eit=0.4117
Ir=0.7974
Ii=0.6586

Xt=Xa+((X1*Xm)/(X1+Xm));              //
```



```
 The magnitude of terminal voltage in per unit is :1.0239 and its angle is :12.4074 degrees
The generator output current is:1.2750-1.2500i per unit
The current injection on the network reference is:0.9766-1.4948i per unit
The reactive voltage is:1.1960 per unit

-->
```

Figure 11.7: Doubly Fed Asynchronous Generator Example

```
transient reactance
X=Xa+Xm;                                  //synchronous reactance
omega=2*%pi*f;
Tot=((X1+Xm)/(omega*R1));                 //open circuit time constant for the rotor

Vr=Ert-(Ra*Ir)+(Xt*Ii);
Vi=Eit-(Ra*Ii)-(Xt*Ir);
dErt=(omega*S*Eit)-((1/Tot)*(Ert-(X-Xt)*Ii));
dEit=(-(omega)*S*Ert)-((1/Tot)*(Eit+(X-Xt)*Ir));
Pe=(Vr*Ir+Vi*Ii);                         //The terminal real power injection
Qe=(-Vr*Ii+Vi*Ir);                         //The reactive power produced by the machine

printf('The transient reactance is:%.4fi per unit\n',Xt);
printf('The synchronous reactance is:%.4fi per unit\n',X);
printf('The open circuit time constant for the rotor is:%.4fi per unit\n',Tot);
printf('The terminal real power injection is:%.4f per unit\n',Pe);
printf('The terminal reactive power injection is:%.4f per unit\n',Qe);
```

**Scilab code Exa 11.12** `Doubly Fed Asynchronous Generator Example`

```
//Book - Power system: Analysisi & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J.Overbye
//Chapter-11 ;Example 11.12
//Scilab Version - 6.0.0; OS - Windows

clc;
clear;


Vr=1.0                                      //system voltage in per unit
I=1.0                                       //terminal current
pf=1                                        //Lagging power factor
X=%i*0.22
Xeq=0.8                                     //DFAG reactance in per unit



VT=Vr+I*X                                   //Terminal voltage
Isorc=I+(VT/(%i*Xeq))                       //current injection on the network reference in per unit
Isorcpq=Isorc*(1*exp(%i*-12.41*%pi/180))    //The value of Ip and Iq are then calculated by shifting these values backwards by the angle of the terminal voltage
```

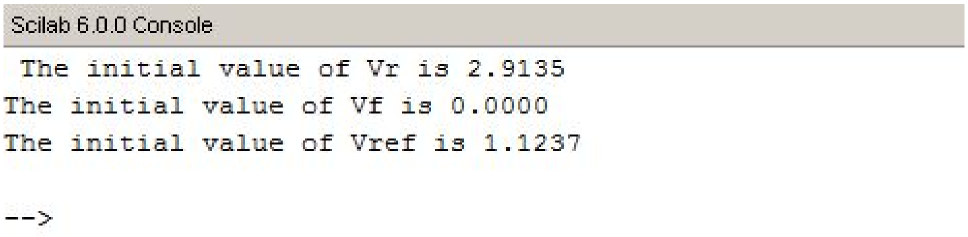
```
Iq=-1.495                                       //reactive power current current
Eq=-Iq*Xeq                                      //The reactive voltage


printf('The magnitude of terminal voltage in per unit is :%.4f and its angle is :%.4f degrees\n',abs(VT),atand(imag(VT),real(VT)));
printf('The generator output current is:%.4f%.4fi per unit\n',real(Isorc),imag(Isorc));
printf('The current injection on the network reference is:%.4f%.4fi per unit\n',real(Isorcpq),imag(Isorcpq));
printf('The reactive voltage is:%.4f per unit\n',Eq);
```

# Chapter 12

# POWER SYSTEM CONTROLS

**Scilab code Exa 12.1** `Synchronous Generator Exciter Response`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 12 ; Example 12.1
//Scilab Version - 6.0.0 ; OS - Windows

```



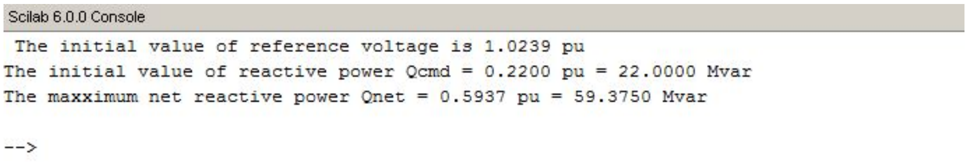
```
 The initial value of Vr is 2.9135
The initial value of Vf is 0.0000
The initial value of Vref is 1.1237

-->
```

Figure 12.1: Synchronous Generator Exciter Response

```
clc;
clear;

Tr=0;
Ka=100;
Ta=0.05;
Vrmax=5;
Vrmin=-5;
Ke=1;
Te=0.26;
Kf=0.01;
Tf=1;

Efd=2.9135;             //Value taken from Example 11.10
Vt=1.0946;              //Value taken from Example 11.10

Vr=Ke*Efd;              //Initial value of Vr
Vf=0;                   //Initial value of vf
Vref=(Vr/Ka)+Vt+Vf;     //Initial value of Vref

printf('The initial value of Vr is %.4f\n',Vr)
printf('The initial value of Vf is %.4f\n',Vf)
printf('The initial value of Vref is %.4f\n',Vref)

//Section 'b' of this problem cannot be simulated using current version of Scilab
```

**Scilab code Exa 12.2** `Type 3 Wind Turbine Reactive Power Control`

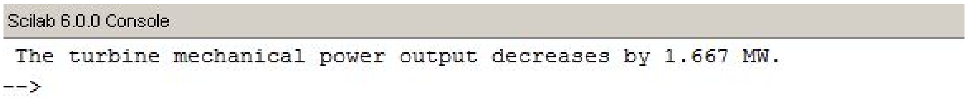
```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma,
```



```
 The initial value of reference voltage is 1.0239 pu
The initial value of reactive power Qcmd = 0.2200 pu = 22.0000 Mvar
The maxximum net reactive power Qnet = 0.5937 pu = 59.3750 Mvar

-->
```

Figure 12.2: Type 3 Wind Turbine Reactive Power Control

```
    and Thomas J. Overbye
//Chapter - 12 ; Example 12.2
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;

KQi=0.4;
KVi=40;
XIqmax=1.45;
XIqmin=0.5;
Vmax=1.1;
Vmin=0.9;

Tr=0;
Ka=100;
Ta=0.05;
Vrmax=5;
Vrmin=-5;
Ke=1;
Te=0.26;
Kf=0.01;
Tf=1;
vt=0.5;
Vf=0;                    //Initial value of vf
Vref=1.0239;             //Initial value of Vref from
    Example 11.12
Isorq=-XIqmax/0.8;       // Reactive component of
    Isorc
Qcmd=0.22;               //Obtained from Example
```



```
 The turbine mechanical power output decreases by 1.667 MW.
-->
```

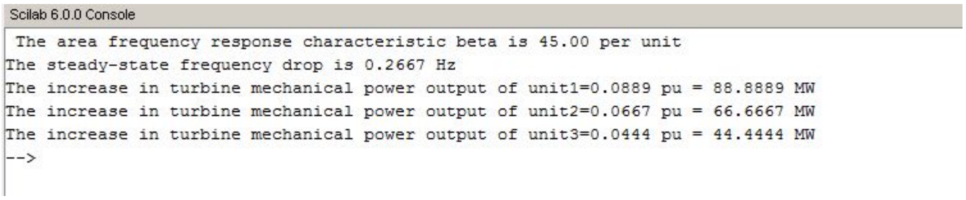
Figure 12.3: Turbine governor response to frequency change at a generating unit

```
    11.12
Qnet=(vt)*abs(Isorq)-(vt)^2/0.8;    // Net reactive power injection in pu

printf('The initial value of reference voltage is %.4f pu\n',Vref)
printf('The initial value of reactive power Qcmd = %.4f pu = %.4f Mvar\n',Qcmd,Qcmd*100)
printf('The maxximum net reactive power Qnet = %.4f pu = %.4f Mvar\n',Qnet,Qnet*100)
```

---

**Scilab code Exa 12.3** `Turbine governor response to frequency change at a generating`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 12 ; Example 12.3
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;

M=500;                  //MVA rating of the generator
f=60;                   //requency in Hertz
R=0.05;                 //Regulation constant in pu
```



```
 The area frequency response characteristic beta is 45.00 per unit
The steady-state frequency drop is 0.2667 Hz
The increase in turbine mechanical power output of unit1=0.0889 pu = 88.8889 MW
The increase in turbine mechanical power output of unit2=0.0667 pu = 66.6667 MW
The increase in turbine mechanical power output of unit3=0.0444 pu = 44.4444 MW
-->
```

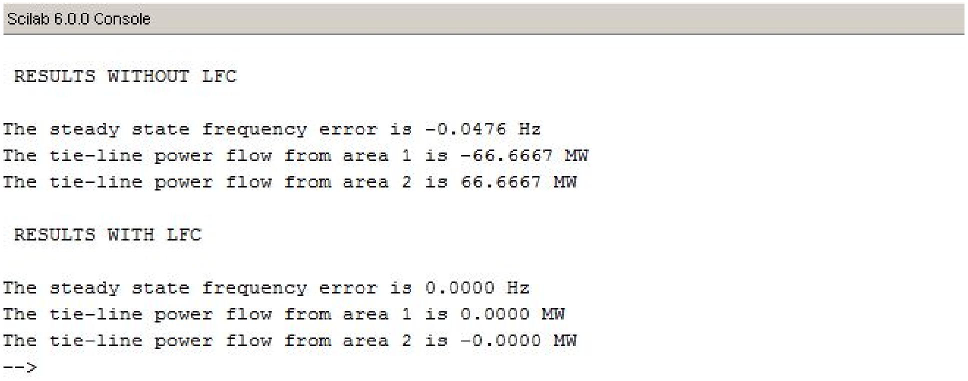
Figure 12.4: Response of turbine governors to a load change in an interconnected power system

```
delF=0.01;              //Increase in frequency in Hertz

delFpu=delF/f;          //Frequency increase in pu
delPref=0;              //Since fixed reference power setting is assumed
delPmpu=delPref-(1/R)*delFpu     //Change in mechanical power in pu
delPm=delPmpu*M;                 //Actual value of mechanical power in MW

printf('The turbine mechanical power output decreases by %.3f MW.',abs(delPm))
```

**Scilab code Exa 12.4** `Response of turbine governors to a load change in an intercon`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 12 ; Example 12.4
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;
```

```
funcprot(0);
f=60;                          //Frequency in Hertz
G=[1000 750 500];              //Rating of unit 1,2 &3 respectively in MVA
R=0.05;                        //Regulation constant of each unit in pu
delP=200;                      //Load increment in MW
SBnew=1000;                    //New MVA base of the entire system

Rnew=R*(SBnew./G);             //Regulation of each generators with common base
beta=sum(1 ./Rnew);             //area frequency response characteristic , beta

printf('The area frequency response characteristic beta is %.2f per unit\n',beta)

delPpu=delP/SBnew;             //Load increment in per unit
delFpu=(-1/beta)*delPpu        //Frequency drop in per unit
delF=delFpu*f;                 //Frequency drop in Hertz

printf('The steady-state frequency drop is %.4f Hz\n',abs(delF))

delPm=delFpu*(-1 ./Rnew);
delPmact=SBnew*delPm;

printf('The increase in turbine mechanical power output of unit1=%.4f pu = %.4f MW\n',delPm(1),delPmact(1))
printf('The increase in turbine mechanical power output of unit2=%.4f pu = %.4f MW\n',delPm(2),delPmact(2))
```



```
 RESULTS WITHOUT LFC

The steady state frequency error is -0.0476 Hz
The tie-line power flow from area 1 is -66.6667 MW
The tie-line power flow from area 2 is 66.6667 MW

 RESULTS WITH LFC

The steady state frequency error is 0.0000 Hz
The tie-line power flow from area 1 is 0.0000 MW
The tie-line power flow from area 2 is -0.0000 MW
-->
```

Figure 12.5: Response of LFC to a load change in an interconnected power system

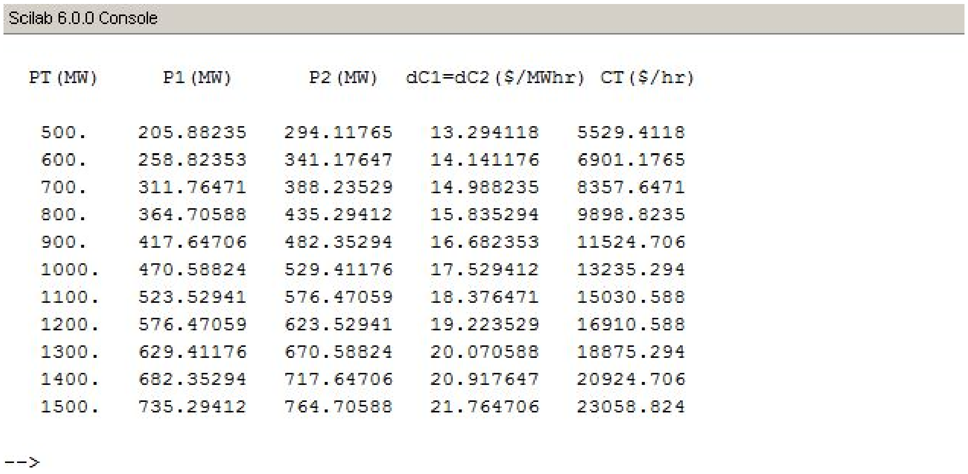
```
printf('The increase in turbine mechanical power output of unit3=%.4f pu = %.4f MW',delPm(3),delPmact(3))
```

---

**Scilab code Exa 12.5** `Response of LFC to a load change in an interconnected power s`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 12 ; Example 12.5
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;

f=60;                          //Frequency of the system in Hertz
G1=2000;                       //Total generation of area 1 in MW
```

```
G2=4000;                    //Total generation of area 2 in MW
beta1=700;                  //Area frequency response characteristic of area 1 in MW/Hz
beta2=1400;                 //Area frequency response characteristic of area 2 in MW/Hz
delPD1=100;                 //Load increment of area 1 in MW
delPD2=0;

//WITHOUT LFC
delF=(delPD1+delPD2)/(-(beta1+beta2));     //Frequency Change in Hertz
delPm1=-beta1*delF;                        //Change in power of area 1
delPm2=-beta2*delF;                        //Change in power of area 2
delPtie1=-delPm2                           //Tie line power flow from area 1 to 2
delPtie2=delPm2                           //Tie line power flow from area 2 to 1

disp('RESULTS WITHOUT LFC')
printf('\nThe steady state frequency error is %.4f Hz',delF)
printf('\nThe tie-line power flow from area 1 is %.4f MW',delPtie1)
printf('\nThe tie-line power flow from area 2 is %.4f MW\n',delPtie2)

//WITH LFC

delFl=0/(beta1+beta2);                //Frequency Change in Hertz (as ACE1+ACE2=0)
delPm1=-beta1*delFl;                  //Change in power of area 1
delPm2=-beta2*delFl;                  //Change in power of area 2
```



```
 PT(MW)      P1(MW)       P2(MW)   dC1=dC2($/MWhr)  CT($/hr)

  500.     205.88235   294.11765   13.294118   5529.4118
  600.     258.82353   341.17647   14.141176   6901.1765
  700.     311.76471   388.23529   14.988235   8357.6471
  800.     364.70588   435.29412   15.835294   9898.8235
  900.     417.64706   482.35294   16.682353   11524.706
  1000.    470.58824   529.41176   17.529412   13235.294
  1100.    523.52941   576.47059   18.376471   15030.588
  1200.    576.47059   623.52941   19.223529   16910.588
  1300.    629.41176   670.58824   20.070588   18875.294
  1400.    682.35294   717.64706   20.917647   20924.706
  1500.    735.29412   764.70588   21.764706   23058.824

-->
```

Figure 12.6: Economic dispatch solution neglecting generator limits and line losses

```
delPtie1=-delPm2                        //Tie line power flow from area 1 to 2
delPtie2=delPm2                         //Tie line power flow from area 2 to 1

disp('RESULTS WITH LFC')
printf('\nThe steady state frequency error is %.4f Hz',delF1)
printf('\nThe tie-line power flow from area 1 is %.4f MW',delPtie1)
printf('\nThe tie-line power flow from area 2 is %.4f MW',delPtie2)
```

---

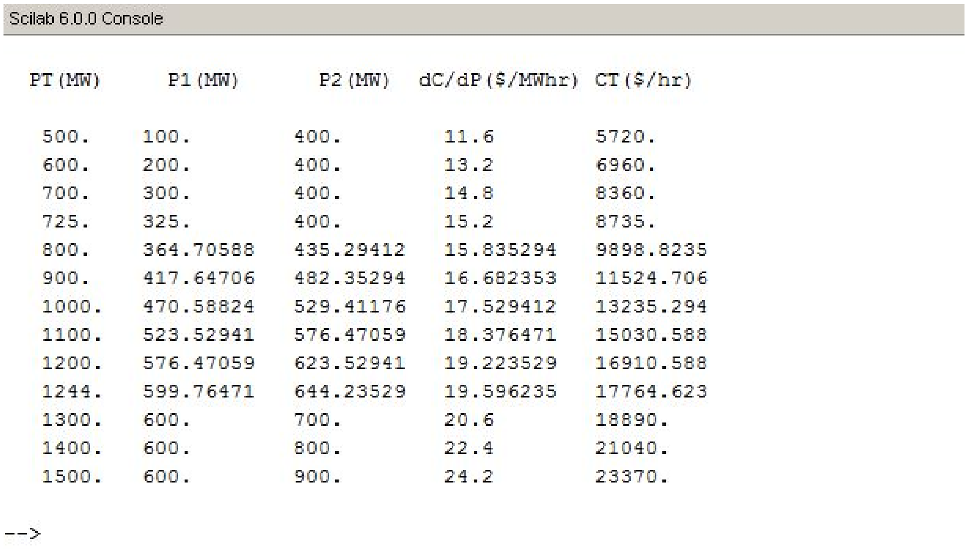
**Scilab code Exa 12.6** `Economic dispatch solution neglecting generator limits and li`

```
//Book - Power System: Analysis & Design 5th Edition
```

```
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 12 ; Example 12.6
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;

C1=[8e-3 10 0]                    //Coefficients of cost equation for unit 1
C2=[9e-3 8  0]                    //Coefficients of cost equation for unit 2

dC1=[2*C1(1) C1(2)]          //Coefficients of incremental cost equation for unit 1
dC2=[2*C2(1) C2(2)]          //Coefficients of incremental cost equation for unit 2
result=[];
for PT=500:100:1500
    P1=(dC2(1)*PT+(dC2(2)-dC1(2)))/(dC2(1)+dC1(1));
    P2=PT-P1;
    dC1value=dC1(1)*P1+dC1(2);
    dC2value=dC2(1)*P2+dC2(2);
    CT=C1(1)*P1^2+C1(2)*P1+C1(3)+C2(1)*P2^2+C2(2)*P2+C2(3);
    result=[result;PT P1 P2 dC1value CT]
end

disp(result,' PT(MW)       P1(MW)         P2(MW)    dC1=dC2($/MWhr)  CT($/hr)');
```

---

**Scilab code Exa 12.7** `Economic dispatch solution including generator limits`

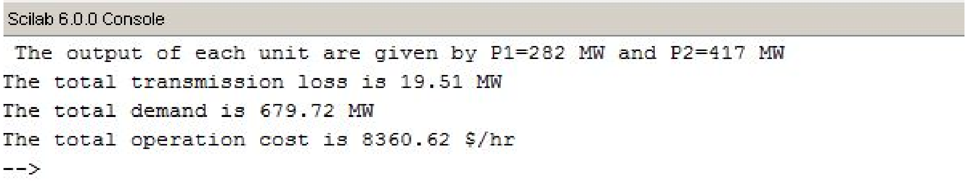


```
 PT(MW)      P1(MW)       P2(MW)   dC/dP($/MWhr)  CT($/hr)

  500.    100.        400.        11.6         5720.
  600.    200.        400.        13.2         6960.
  700.    300.        400.        14.8         8360.
  725.    325.        400.        15.2         8735.
  800.    364.70588   435.29412   15.835294    9898.8235
  900.    417.64706   482.35294   16.682353    11524.706
  1000.   470.58824   529.41176   17.529412    13235.294
  1100.   523.52941   576.47059   18.376471    15030.588
  1200.   576.47059   623.52941   19.223529    16910.588
  1244.   599.76471   644.23529   19.596235    17764.623
  1300.   600.        700.        20.6         18890.
  1400.   600.        800.        22.4         21040.
  1500.   600.        900.        24.2         23370.

-->
```

Figure 12.7: Economic dispatch solution including generator limits

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 12 ; Example 12.7
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;

C1=[8e-3 10 0]                  //Coefficients of cost equation for unit 1
C2=[9e-3 8  0]                  //Coefficients of cost equation for unit 2

dC1=[2*C1(1) C1(2)]          //Coefficients of incremental cost equation for unit 1
dC2=[2*C2(1) C2(2)]          //Coefficients of incremental cost equation for unit 2

P1lim=[100 600];             //Lower and upper
```

```
generation limit for unit 1
P2lim=[400 1000];            //Lower and upper generation limit for unit 2

result=[];
for PT=[500 600 700 725 800 900 1000 1100 1200 1244 1300 1400 1500]
    P1=(dC2(1)*PT+(dC2(2)-dC1(2)))/(dC2(1)+dC1(1));
    P2=PT-P1;
    dC1value=dC1(1)*P1+dC1(2);
    dC2value=dC2(1)*P2+dC2(2);

    if P1<P1lim(1) | P1>P1lim(2)       //Checking for limits of P1
        if P1<P1lim(1)
            P1=P1lim(1)
        else
            P1=P1lim(2)
        end
        P2=PT-P1;
        dC1value=dC2(1)*P2+dC2(2);
    elseif P2<P2lim(1) | P2>P2lim(2)   //Checking for limits of P2
        if P2<P2lim(1)
            P2=P2lim(1)
        else
            P2=P2lim(2)
        end
        P1=PT-P2;
        dC1value=dC1(1)*P1+dC1(2);
    end

    CT=C1(1)*P1^2+C1(2)*P1+C1(3)+C2(1)*P2^2+C2(2)*P2+C2(3);   //Total cost in $/hr
    result=[result;PT P1 P2 dC1value CT]
end
disp(result,' PT(MW)      P1(MW)        P2(MW)   dC/dP($/MWhr)  CT($/hr)');
```



```
 The output of each unit are given by P1=282 MW and P2=417 MW
The total transmission loss is 19.51 MW
The total demand is 679.72 MW
The total operation cost is 8360.62 $/hr
-->
```

Figure 12.8: Economic dispatch solution including generator limits and line losses

---

**Scilab code Exa 12.9** `Economic dispatch solution including generator limits and lin`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 12 ; Example 12.9
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;

B11=1.5e-4; B12=2e-5; B22=3e-5;                    //Loss coefficients
lamda=16;                                          //Area incremental cost in $/MWhr

e1=[20.8e-3 32e-5 6];         //Coefficients of incremental operating cost equation1
e2=[32e-5 18.96e-3 8];        //Coefficients of incremental operating cost equation2

P1=(e2(2)*e1(3)-e1(2)*e2(3))/(e2(2)*e1(1)-e1(2)*e2(1));    //Solution of P1 from incremental cost
```

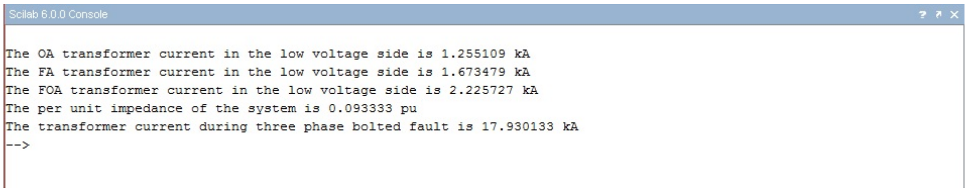
```
equations
P2=(e2(1)*e1(3)-e1(1)*e2(3))/(e1(2)*e2(1)-e2(2)*e1(1));    //Solution of P2 from incremental cost equations

Pl=B11*P1^2+B12*P1*P2+B22*P2^2;              //Total losses

Pt=P1+P2-Pl;                                 //Total demand

CT=10*P1+8e-3*P1^2+8*P2+9e-3*P2^2;          // Cost equation taken from example 12.6

printf('The output of each unit are given by P1=%d MW and P2=%d MW\n',P1,P2)
printf('The total transmission loss is %.2f MW\n',Pl)
printf('The total demand is %.2f MW\n',Pt)
printf('The total operation cost is %.2f $/hr',CT)
```

# Chapter 14

# POWER DISTRIBUTION

**Scilab code Exa 14.1** `Distribution Substation Transformer Rated Current and Short C`

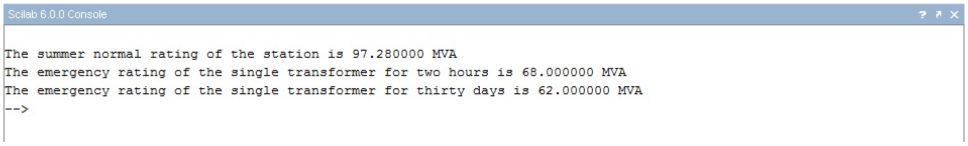
```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 14 ; Example 14.1
//Scilab Version - 6.0.0 ; OS - Windows


clc;
clear;
Vdelpri=230;                                  //The
```



```
The OA transformer current in the low voltage side is 1.255109 kA
The FA transformer current in the low voltage side is 1.673479 kA
The FOA transformer current in the low voltage side is 2.225727 kA
The per unit impedance of the system is 0.093333 pu
The transformer current during three phase bolted fault is 17.930133 kA
-->
```

Figure 14.1: Distribution Substation Transformer Rated Current and Short Circuit Current

```
rated RMS line voltage of the primary winding in kV
Vwyesec=34.5;                                   //The rated RMS line voltage of the secondary winding in kV
MVAoa=75;                                       //The MVA OA rating of the transformer
MVAfa=100;                                      //The MVA FA rating of the transformer
MVAfoa=133;                                     //The MVA FOA rating of the transformer
Zpu=0.07;                                       //The percentage impedance of the transformer in terms of MVA OA ratings
MVAbase=100;                                    //The MVA base in MVA
kVbase=34.5;                                    //The KV base in kV
Ioa=(MVAoa/3)/(Vwyesec/sqrt(3));                //The OA transformer current in the low voltage side in kA
Ifa=(MVAfa/3)/(Vwyesec/sqrt(3));                //The FA transformer current in the low voltage side in kA
Ifoa=(MVAfoa/3)/(Vwyesec/sqrt(3));              //The FOA transformer current in the low voltage side in kA
Zbasepu=Zpu*MVAbase/MVAoa;                      //The per unit impedance of the system in ohm pu
Isc3ph=(1/Zpu)*Ioa;                             //The transformer current during three phase bolted faultin kA
printf('\nThe OA transformer current in the low voltage side is %f kA',Ioa);
printf('\nThe FA transformer current in the low voltage side is %f kA\n',Ifa);
printf('The FOA transformer current in the low voltage side is %f kA\n',Ifoa);
```



```
The summer normal rating of the station is 97.280000 MVA
The emergency rating of the single transformer for two hours is 68.000000 MVA
The emergency rating of the single transformer for thirty days is 62.000000 MVA
-->
```

Figure 14.2: Distribution Substation Normal Emergency and Allowable Ratings

```
printf('The per unit impedance of the system is %f pu\n',Zbasepu);
printf('The transformer current during three phase bolted fault is %f kA',Isc3ph);

//The answer in the book is not correct. Eg.75/(sqrt(3)*34.5)---Actual result is 1.255, but it is given as 7.372 in the book.
```

---

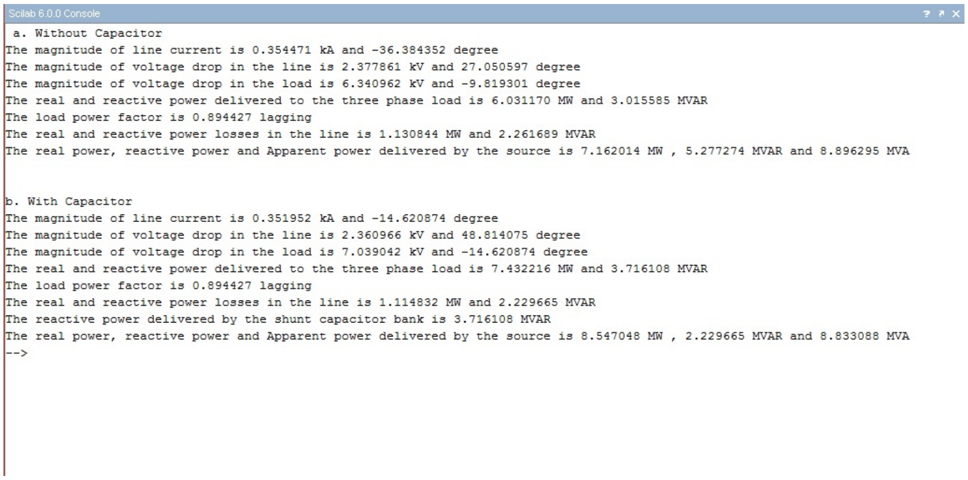
**Scilab code Exa 14.2** `Distribution Substation Normal Emergency and Allowable Rating`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 14 ; Example 14.2
//Scilab Version - 6.0.0 ; OS - Windows


clc;
clear;
MVAtr1=40;                              //MVA FOA rating of transformer 1
MVAtr2=40;                              //MVA FOA rating of transformer 2
```

```
normal=1.28;                                        //Factor for normal summer operation
emergency2hr=1.70;                                  //Factor for two hour emergency operation
emergency30day=1.55;                                //Factor for thirty days emergency operation
unequalloadingfactor=0.95;                          //Factor to account for unequal transformer loading
MVAstation=normal*(MVAtr1+MVAtr2)*unequalloadingfactor;     //MVA rating of thr station
MVAstationemergency2hr=emergency2hr*MVAtr1;                 //MVA rating of a single transformer for two hour emergency
MVAstationemergency30day=emergency30day*MVAtr1;             //MVA rating of a single transformer for thirty days emergency
printf('\nThe summer normal rating of the station is %f MVA',MVAstation);
printf('\nThe emergency rating of the single transformer for two hours is %f MVA',MVAstationemergency2hr);
printf('\nThe emergency rating of the single transformer for thirty days is %f MVA',MVAstationemergency30day)
```

**Scilab code Exa 14.3** Shunt Capacitor Bank at End of Primary Feeder

```
//Book - Power System: Analysis & Design 5th Edition
```



```
 a. Without Capacitor
The magnitude of line current is 0.354471 kA and -36.384352 degree
The magnitude of voltage drop in the line is 2.377861 kV and 27.050597 degree
The magnitude of voltage drop in the load is 6.340962 kV and -9.819301 degree
The real and reactive power delivered to the three phase load is 6.031170 MW and 3.015585 MVAR
The load power factor is 0.894427 lagging
The real and reactive power losses in the line is 1.130844 MW and 2.261689 MVAR
The real power, reactive power and Apparent power delivered by the source is 7.162014 MW , 5.277274 MVAR and 8.896295 MVA


b. With Capacitor
The magnitude of line current is 0.351952 kA and -14.620874 degree
The magnitude of voltage drop in the line is 2.360966 kV and 48.814075 degree
The magnitude of voltage drop in the load is 7.039042 kV and -14.620874 degree
The real and reactive power delivered to the three phase load is 7.432216 MW and 3.716108 MVAR
The load power factor is 0.894427 lagging
The real and reactive power losses in the line is 1.114832 MW and 2.229665 MVAR
The reactive power delivered by the shunt capacitor bank is 3.716108 MVAR
The real power, reactive power and Apparent power delivered by the source is 8.547048 MW , 2.229665 MVAR and 8.833088 MVA
-->
```

Figure 14.3: Shunt Capacitor Bank at End of Primary Feeder

```
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 14 ; Example 14.3
//Scilab Version - 6.0.0 ; OS - Windows


clc;
clear;
kV=13.8;                    //The sending end line voltage in kVolts
Vsln=1.05*kV/sqrt(3);       //The sending end voltage with 5% above rated in kVolts
Rload=20;                   //The Wye connected load resistance in Ohm
Xload=40*%i;                //The Wye connected load inductive reactance in Ohm
Xc=-40*%i;                  //The Wye connected capacitive reactance in Ohm
Rline=3;                    //The line resistance in Ohm
Xline=6*%i;                 //
```

```
   The line inductive reactance in Ohm
Ztot1=Rline+Xline+(Rload*Xload/(Rload+Xload));     //
   The total impedance seen by source without
   capacitance in Ohm
Iline1=Vsln/Ztot1;                                  //
   The line current without shunt capacitor in kA
Vdrop1=(Rline+Xline)*Iline1;                        //
   The voltage drop across the line without shunt
   capacitor in KVolts
Vload1=Vsln-Vdrop1;                                 //
   The voltage drop across the load without shunt
   capacitor in KVolts
Pload1=3*abs(Vload1)^2/Rload;                       //
   The real power delivered to the load without
   shunt capacitor in MW
Qload1=3*abs(Vload1)^2/abs(Xload);                  //
   The reactive power delivered to the load without
   shunt capacitor in MVAR
pf1=cos((atan(Qload1/Pload1)));                     //
   The power factor of the load without shunt
   capacitor
Pline1=3*abs(Iline1)^2*Rline;                       //
   The real power loss in the line without shunt
   capacitor in MW
Qline1=3*abs(Iline1)^2*abs(Xline);                  //
   The reactive power loss in the line without shunt
    capacitor in MVAR
Psource1=Pload1+Pline1;                             //
   The real power delivered by the source without
   shunt capacitor in MW
Qsource1=Qload1+Qline1;                             //
   The reactive power delivered by the source
   without shunt capacitor in MVAR
Ssource1=sqrt(Psource1^2+Qsource1^2);               //
   The apparent power delivered by the source
   without shunt capacitor in MVA
Ztot2=Rline+Xline+(1/(1/Rload+1/Xload+1/Xc));       //
   The total impedance seen by source with
```

```
   capacitance in Ohm
Iline2=Vsln/Ztot2;                                  //
   The line current with shunt capacitor in kA
Vdrop2=(Rline+Xline)*Iline2;                        //
   The voltage drop across the line with shunt
   capacitor in KVolts
Vload2=Vsln-Vdrop2;                                 //
   The voltage drop across the load with shunt
   capacitor in KVolts
Pload2=3*abs(Vload2)^2/Rload;                       //
   The real power delivered to the load with shunt
   capacitor in MW
Qload2=3*abs(Vload2)^2/abs(Xload);                  //
   The reactive power delivered to the load with
   shunt capacitor in MVAR
pf2=cos((atan(Qload2/Pload2)));                     //
   The power factor of the load with shunt capacitor
Pline2=3*abs(Iline2)^2*Rline;                       //
   The real power loss in the line with shunt
   capacitor in MW
Qline2=3*abs(Iline2)^2*abs(Xline);                  //
   The reactive power loss in the line with shunt
   capacitor in MVAR
Qc=3*abs(Vload2)^2/abs(Xc);                         //
   The reactive power delivered by the shunt
   capacitor inb MVAR
Psource2=Pload2+Pline2;                             //
   The real power delivered by the source with shunt
    capacitor in MW
Qsource2=Qload2+Qline2-Qc;                          //
   The reactive power delivered by the source with
   shunt capacitor in MVAR
Ssource2=sqrt(Psource2^2+Qsource2^2);               //
   The apparent power delivered by the source with
   shunt capacitor in MVA
printf('a. Without Capacitor');
printf('\nThe magnitude of line current is %f kA and
    %f degree',abs(Iline1),atand(imag(Iline1)/real(
```

```
Iline1)));
printf('\nThe magnitude of voltage drop in the line is %f kV and %f degree',abs(Vdrop1),atand(imag(Vdrop1)/real(Vdrop1)));
printf('\nThe magnitude of voltage drop in the load is %f kV and %f degree',abs(Vload1),atand(imag(Vload1)/real(Vload1)));
printf('\nThe real and reactive power delivered to the three phase load is %f MW and %f MVAR',Pload1,Qload1);
printf('\nThe load power factor is %f lagging',pf1);
printf('\nThe real and reactive power losses in the line is %f MW and %f MVAR',Pline1,Qline1);
printf('\nThe real power, reactive power and Apparent power delivered by the source is %f MW , %f MVAR and %f MVA',Psource1,Qsource1,Ssource1);
printf('\n\n\nb. With Capacitor');
printf('\nThe magnitude of line current is %f kA and %f degree',abs(Iline2),atand(imag(Iline2)/real(Iline2)));
printf('\nThe magnitude of voltage drop in the line is %f kV and %f degree',abs(Vdrop2),atand(imag(Vdrop2)/real(Vdrop2)));
printf('\nThe magnitude of voltage drop in the load is %f kV and %f degree',abs(Vload2),atand(imag(Vload2)/real(Vload2)));
printf('\nThe real and reactive power delivered to the three phase load is %f MW and %f MVAR',Pload2,Qload2);
printf('\nThe load power factor is %f lagging',pf2);
printf('\nThe real and reactive power losses in the line is %f MW and %f MVAR',Pline2,Qline2);
printf('\nThe reactive power delivered by the shunt capacitor bank is %f MVAR',Qc);
printf('\nThe real power, reactive power and Apparent power delivered by the source is %f MW , %f MVAR and %f MVA',Psource2,Qsource2,Ssource2);
                                                        //
```

The third part of this question cannot be executed in SCILAB because of its theoritical nature

```
SAIFI = 1.607500 interruptions/year
SAIDI = 86.109750 minutes/year
CAIDI = 53.567496 minutes/year
ASAI = 99.983617 percentage
-->
```

Figure 14.4: Distribution Reliability Indices

**Scilab code Exa 14.4** `Distribution Reliability Indices`

```
//Book - Power System: Analysis & Design 5th Edition
//Authors - J. Duncan Glover, Mulukutla S. Sarma, and Thomas J. Overbye
//Chapter - 14 ; Example 14.1
//Scilab Version - 6.0.0 ; OS - Windows

clc;
clear;
time_interruptions=[8.17 200;71.3 600; 30.3 25; 267.2 90; 120 700; 10 1500; 40 100];           //Number of custumers interrupted for a time duration in minutes
total_customers=2000;                          //Total number of customers
SAIFI=(time_interruptions(1,2)+time_interruptions(2,2)+time_interruptions(3,2)+time_interruptions(4,2)+time_interruptions(5,2)+time_interruptions(6,2)+time_interruptions(7,2))/total_customers;                          //System average interruption frequency index
SAIDI=(time_interruptions(1,2)*time_interruptions(1,1)+time_interruptions(2,2)*time_interruptions(2,1)+time_interruptions(3,2)*time_interruptions(3,1)+time_interruptions(4,2)*time_interruptions(4,1)+time_interruptions(5,2)*time_interruptions(5,1)+time_interruptions(6,2)*time_interruptions
```

```
    (6,1)+time_interruptions(7,2)*time_interruptions
    (7,1))/total_customers;

    //System average interruption duration index
CAIDI=SAIDI/SAIFI;                          //Customer
    average interruption duration index
ASAI=(365*24*total_customers-(time_interruptions
    (1,2)*time_interruptions(1,1)+time_interruptions
    (2,2)*time_interruptions(2,1)+time_interruptions
    (3,2)*time_interruptions(3,1)+time_interruptions
    (4,2)*time_interruptions(4,1)+time_interruptions
    (5,2)*time_interruptions(5,1)+time_interruptions
    (6,2)*time_interruptions(6,1)+time_interruptions
    (7,2)*time_interruptions(7,1))/60)*100/(365*24*
    total_customers);                    //Average
    service availability index
printf('\nSAIFI = %f interruptions/year',SAIFI);
printf('\nSAIDI = %f minutes/year',SAIDI);
printf('\nCAIDI = %f minutes/year',CAIDI);
printf('\nASAI = %f percentage',ASAI);
```