PS MATLAB CODES

1] Y BUS

```
clc
clear all
n=input('enter no. of buses:'); % no. of buses excluding reference
nl= input('enter no. of lines:'); % no. of transmission lines
sb= input('enter starting bus of each line:'); % starting bus of a line
eb= input('enter ending bus of each line'); % ending bus of a line
zser= input('enter resistance and reactance of each line:'); % line resistance and reactance
(R, X)
yshty= input('enter shunt admittance of the bus:'); % shunt admittance
i=0;
k=1;
while i<nl
zser1(i+1)=zser(k)+j*zser(k+1); % impedance of a line (R+jX)
i=i+1;
k=k+2;
end
zser2=reshape(zser1,nl,1);
yser=ones(nl,1)./zser2;
ypri=zeros(nl+n,nl+n);
ybus=zeros(n,n);
a=zeros(nl+n,n);
for i=1:n
a(i,i)=1;
end
for i=1:nl
a(n+i,sb(i))=1;
a(n+i,eb(i))=-1;
ypri(n+i,n+i)=yser(i);
ypri(sb(i),sb(i))=ypri(sb(i),sb(i))+yshty(i);
ypri(eb(i),eb(i))=ypri(sb(i),eb(i))+yshty(i);
end
at=transpose(a);
ybus=at*ypri*a
zbus=inv(ybus)
```

---

inputs in command window:

```
enter no. of buses:
3
enter no. of lines:
3
enter starting bus of each line:
[1 1 2]
enter ending bus of each line
[2 3 3]
enter resistance and reactance of each line:
[0.01 0.03 0.08 0.24 0.06 0.18]
enter shunt admittance of the bus:
[0.01 0.025 0.02]
```

2] GUASS SEIDAL (LOAD FLOW ANALYSIS)

```
 clc;
clear all;
n=input ('no of buses');
l=input ('no of lines');
s=input ('impedance 1 or admittance 2');
for i=1:l
a=input ('starting bus');
b=input ('ending bus');
t=input ('admittance or impedance value');
if s==1
y(a,b)=1/t;
else
y(a,b)=t;
end
y(b,a)=y(a,b);
end
ybus=zeros(n,n);
for i=1:n
for j=1:n
if i==j
for k=1:n
ybus(i,j)=ybus(i,j)+y(i,k);
end
else
ybus(i,j)=-y(i,j);
end
ybus(j,i)=ybus(i,j);
end
end
ybus
p=zeros(1,n);
q=zeros(1,n);
v=zeros(1,n);
pv=input ('no of pv buses');
pq=input('no of pq buses');
s=input ('slack bus number');
v(s)=input ('slack bus voltage');
acc=input ('accleration factor');
for i=1:pv
b(i)=input('pv bus number');
p(b(i))=input('real power');
v(b(i))=input ('voltage value');
qmin(b(i))=input ('min value of q');
qmax(b(i))=input ('max value of q');
end
for i=1:pq
c(i)=input('pq bus number');
p(c(i))=input('real power');
p(c(i))=-p(c(i));
q(c(i))=input ('reactive power');
q(c(i))=-q(c(i));
v(c(i))=1+0i;
end
e=v;
enew(s)=v(s);
it=0;
yy=zeros(1,n);
for ii=1:n
```

```
ypq(ii)=0;
if ii~=s
flag=0;
gen=0;
for j=1:pv
if ii==b(j)
flag=1;
end
end
if flag==1
for k=1:n
yy(ii)=yy(ii)+ybus(ii,k)*v(k);
end
qcal(ii)=-imag(conj(v(ii))*yy(ii));
if qcal(ii)<qmin(ii)
qcal(ii)=qmin(ii);
elseif qcal(ii)>qmax(ii)
qcal(ii)=qmax(ii);
else
qcal(ii)=qcal(ii);
gen=1;
end
else
qcal(ii)=q(ii);
end
qcal(ii)=qcal(ii)*sqrt(-1);
for k=1:n
if k~=ii
ypq(ii)=ypq(ii)+ybus(ii,k)*e(k);
end
end
enew(ii)=(((p(ii)-qcal(ii))/conj(e(ii)))-ypq(ii))/ybus(ii,ii);
dele(ii)=enew(ii)-e(ii);
enew(ii)=e(ii)+acc*dele(ii);
if gen==1 ang=angle(enew(ii));
enew(ii)=v(ii)*cos(ang)+v(ii)*sin(ang)*sqrt(-1);
end
e(ii)=enew(ii);
end
end
disp('voltages');
enew
```

INPUTS IN THE COMMAND WINDOW:

```
no of buses
4
no of lines
5
impedance 1 or admittance 2
2
starting bus
1
ending bus
2
admittance or impedance value
2-8i
starting bus
1
ending bus
3
admittance or impedance value
1-4i
```

```
starting bus
2
ending bus
3
admittance or impedance value
0.666-2.664i
starting bus
2
ending bus
4
admittance or impedance value
1-4i
starting bus
3
ending bus
4
admittance or impedance value
2-8i

ybus =

   3.0000 -12.0000i  -2.0000 + 8.0000i  -1.0000 + 4.0000i   0.0000 + 0.0000i
  -2.0000 + 8.0000i   3.6660 -14.6640i  -0.6660 + 2.6640i  -1.0000 + 4.0000i
  -1.0000 + 4.0000i  -0.6660 + 2.6640i   3.6660 -14.6640i  -2.0000 + 8.0000i
   0.0000 + 0.0000i  -1.0000 + 4.0000i  -2.0000 + 8.0000i   3.0000 -12.0000i

no of pv buses
0
no of pq buses
3
slack bus number
1
slack bus voltage
1.06
accleration factor
1
pq bus number
2
real power
0.5
reactive power
0.2
pq bus number
3
real power
0.4
reactive power
0.3
pq bus number
4
real power
0.3
reactive power
0.1
voltages

enew =

   1.0600 + 0.0000i   1.0119 - 0.0289i   0.9929 - 0.0261i   0.9855 - 0.0486i
```

3] GUASS SEIDAL (LOAD FLOW ANALYSIS)

```
%line data – Save as linedat.m
function d = linedat
% |FromBus|ToBus|Impedance|LineCharging|
d = [ 1 2 .02+.06j .030j;
 1 3 .08+.24j .025j;
 2 3 .06+.18j .020j;
 2 4 .06+.18j .020j;
 2 5 .04+.12j .015j;
 3 4 .01+.03j .010j;
 4 5 .08+.24j .025j;];
%bus data – Save as busdat.m
function d = busdat
Slack=1;
PV=2;
PQ=3;
% |Bus | Type | Vsp |del| PGi | QGi | PLi | QLi |
d = [ 1 Slack 1.06 0 0 0 0 0;
 2 PV 1.0 0 .40 .30 .20 .10;
 3 PQ 1.0 0 0 0 .45 .15;
 4 PQ 1.0 0 0 0 .40 .05;
 5 PQ 1.0 0 0 0 .60 .10;];
%Y-bus formation – Save as ybusf.m
function Y = ybusf(l)
fb = l(:,1); tb = l(:,2);
y = 1./l(:,3); b = l(:,4) + y;
nb = max(max(fb,tb)); Y = zeros(nb);
Y(nb*(fb-1)+tb)=-y; Y(nb*(tb-1)+fb)=-y;
for m=1:nb
 Y(m,m)=Y(m,m)+sum(b(fb==m))+sum(b(tb==m));
end
%FDLF Program – Save and run
clc;
clear all;
b=busdat;
l=linedat;
nb=length(b(:,1));
n1=find(b(:,2)>1); % finds index of non slack buses
n2=find(b(:,2)>2); % finds index of pq buses
Y=ybusf(l); % Ybus function
B1 =imag(Y(n1,n1));
B11=imag(Y(n2,n2));
Ps=b(:,5)-b(:,7);
dP=Ps; % schedule powers and
Qs=b(:,6)-b(:,8);
dQ=Qs; % initial values
d=b(:,4);V=b(:,3);
while (max(abs(dP(n1)))>0.0001||max(abs(dQ(n2)))>0.0001)
 S=(V.exp(-1i*d)).(Y*(V.*exp(1i*d))); % S=(V-d)x(Yx(V?d))
 dP=Ps-real(S); dQ=Qs+imag(S); % S=V*I || S=P-jQ
 d(n1)=d(n1)-B1\ dP(n1) ./V(n1);
 V(n2)=V(n2)-B11\dQ(n2)./V(n2);
end
display(V);
D=d*180/pi;
display(D);
```

4] Z BUS

```
clc;
clear all;
linedata = [1 1 2 0.001 0.015 0.001
2 2 3 0.002 0.021 0.0005
3 3 1 0.004 0.046 0.0015 ];
fb = linedata(:,1);
tb =linedata(:,2);
r = linedata(:,3);
x = linedata(:,4);
b = linedata(:,5);
z = r + i*x;
y = 1./z;
b = i*b;
nbus = max(max(fb),max(tb));
nbranch = length(fb);
Y = zeros(nbus,nbus);
for k=1:nbranch;
Y(fb(k),tb(k)) = Y(fb(k),tb(k))-y(k);
Y(tb(k),fb(k)) = Y(fb(k),tb(k));
end
for m =1:nbus;
for n=1:nbranch;
if fb(n) == m
Y(m,m) = Y(m,m) + y(n)+ b(n);
elseif tb(n) == m
Y(m,m) = Y(m,m) + y(n) + b(n);
end
end
end
Y
zbus = inv(Y)
```

5] SECURITY ANALYSIS

```
clc;
clear all;
busdata=[1 0 1 0;
2 0.5 3 0;
3 0.6 3 0;
4 0 2 0.7;
5 0 2 0.7;
6 0 2 0.7];
linedata=[1 2 0.2;
1 4 0.2;
1 5 0.3;
2 3 0.25;
2 4 0.1;
2 5 0.3;
2 6 0.2;
3 5 0.26;
3 6 0.1;
4 5 0.4;
5 6 0.3];
powerlimit=[12 0.3;
14 0.5;
15 0.4;
23 0.2;
24 0.4;
25 0.2;
26 0.3;
35 0.2;
36 0.6;
45 0.2;
56 0.2];
bus=busdata(:,1);
pg=busdata(:,2);
pl=busdata(:,4);
bustype=busdata(:,3);
from=linedata(:,1);
to=linedata(:,2);
x=linedata(:,3);
nb=max(bus);
n1=length(from);
line=powerlimit(:,1);
limit=powerlimit(:,2);
for i=2:nb;
p(i-1)=pg(i)-pl(i);
end
b=zeros(nb-1,nb-1);
for i=2:nb
for j=1:n1
if(from(j)==i | to(j)==i)
b(i-1,i-1)=(1/x(j))+b(i-1,i-1);
end
end
end
for i=1:n1
if (from(i)~=1 & to(i)~=1)
b(from(i)-1,to(i)-1)=-1/x(i);
b(to(i)-1,from(i)-1)=b(from(i)-1,to(i)-1);
end
end
the=inv(b)*p';
t=[0;the];
pn=zeros(nb,nb);
```

```
for i=1:n1
pn(from(i),to(i))=(1/x(i))*(t(from(i))-t(to(i)));
pn(to(i),from(i))=-pn(from(i),to(i));
end
P=pn*100;
pf=zeros(nb,1);
for i=1:nb
for j=1:nb
pf(i)=P(i,j)+pf(i);
end
end
for i=2:nb
if bustype(i)==3
generator(i-1)=i;
end
end
xn=inv(b);
x1=[zeros(1,nb-1);xn];
X=[zeros(nb,1) x1];
k=1;
disp('Contingency Analysis');
disp('Generation Shift Factor and Power flow in line');
for i=2:nb
if bustype(i)==3
disp(generator(i-1));
for j=1:n1
a(from(j),to(j))=(X(from(j),i)-X(to(j),i))/x(j);
f(from(j),to(j))=P(from(j),to(j))+((a(from(j),to(j)))*(-pf(i)));
if(a(from(j),to(j))>limit(j))
flag=1;
display(line(j));
disp('Overloaded');
end
end
display(a);
display(f);
end
end
```

6] LOAD DISPATCH

```
clc;
clear all;
disp('Without Equality Constraints');
F= [0.004 5.3 500
0.006 5.5 400
0.009 5.8 200];
Pd=975;
A=F(:,1);
B=F(:,2);
C=F(:,3);
n=length(A);
I=ones(n,1);
L1=B./A;
L2=I./A;
Lambda=((2*Pd)+sum(L1))/(sum(L2))
P=zeros(n,1);
for i=1:n
P(i)=(Lambda-B(i))/(2*A(i));
```

```
end
P
disp('Fuel Cost:');
FC=sum((A.*P.*P)+(B.*P)+C)
disp('After equality constraints:');
F=[0.006 5.5 400 0.009 5.8 200];
Pd=975-450;
A=F(:,1);
B=F(:,2);
C=F(:,3);
n=length(A);
I=ones(n,1);
L1=B./A;
L2=I./A;
Lambda=((2*Pd)+sum(L1))/(sum(L2))
P=zeros(n,1);
for i=1:n
P(i)=(Lambda-B(i))/(2*A(i));
end
PN=[450;P]
FC1=sum((A.*P.*P)+(B.*P)+C);
FC2=(0.004*450^2)+(5.3*450)+500;
disp('Fuel Cost:');
FC=FC1+FC2
```

7] TRANSIENT STABILITY

NOTE: 2 files to be executed

Trans1:

```
Pm = 0.8;
E = 1.17;
V = 1.0;
X1=0.65;
X2=inf;
X3=0.65;
eacfault(Pm, E, V, X1, X2, X3)

Pm = 0.8;
E = 1.17;
V = 1.0;
X1=0.65;
X2=1.8;
X3=0.8;
eacfault(Pm, E, V,X1,X2,X3)
```

Trans1:

```
function eacfault(Pm, E, V, X1, X2, X3) %save as function witheacfault.m

if exist('Pm')~=1
Pm = input('Generator output power in p.u. Pm = ');else,end
if exist('E')~=1
E = input('Generator e.m.f. in p.u. E = ');else,end
if exist('V')~=1
V = input('Infinite bus-bar voltage in p.u. V = ');else,end
if exist('X1')~=1
X1 = input('Reactance before Fault in p.u. X1 = ');else,end
if exist('X2')~=1
X2 = input('Reactance during Fault in p.u. X2 = ');else,end
if exist('X3')~=1
X3 = input('Reactance after Fault in p.u. X3 = ');else,end
Pe1max = E*V/X1; Pe2max=E*V/X2; Pe3max=E*V/X3;
delta = 0:.01:pi;
Pe1 = Pe1max*sin(delta);
Pe2 = Pe2max*sin(delta);
Pe3 = Pe3max*sin(delta);
d0 =asin(Pm/Pe1max);
dmax = pi-asin(Pm/Pe3max);

cosdc =(Pm*(dmax-d0)+Pe3max*cos(dmax)-Pe2max*cos(d0))/(Pe3max-Pe2max);

if abs(cosdc) > 1
fprintf('No critical clearing angle could be found.\n')
fprintf('system can remain stable during this disturbance.\n\n')
return
else,
end
dc=acos(cosdc);
if dc > dmax
fprintf('No critical clearing angle could be found.\n')
fprintf('System can remain stable during this disturbance.\n\n')
return
else,
end
Pmx=[0 pi-d0]*180/pi; Pmy=[Pm Pm];
x0=[d0 d0]*180/pi; y0=[0 Pm];
xc=[dc dc]*180/pi; yc=[0 Pe3max*sin(dc)];
xm=[dmax dmax]*180/pi;
ym=[0 Pe3max*sin(dmax)];
d0=d0*180/pi;
dmax=dmax*180/pi;
dc=dc*180/pi;
x=(d0:.1:dc);
y=Pe2max*sin(x*pi/180);
y1=Pe2max*sin(d0*pi/180);
y2=Pe2max*sin(dc*pi/180);
x=[d0 x dc];
y=[Pm y Pm];
xx=dc:.1:dmax;
h=Pe3max*sin(xx*pi/180);
xx=[dc xx dmax];
hh=[Pm h Pm];
delta=delta*180/pi;
if X2 == inf
fprintf('\nFor this case tc can be found from analytical formula.\n')
H=input('To find tc enter Inertia Constant H, (or 0 to skip) H = ');
if H ~= 0
d0r=d0*pi/180; dcr=dc*pi/180;
```

```
tc = sqrt(2*H*(dcr-d0r)/(pi*60*Pm));
else, end
else, end
%clc
fprintf('\nInitial power angle = %7.3f \n', d0)
fprintf('Maximum angle swing = %7.3f \n', dmax)
fprintf('Critical clearing angle = %7.3f \n\n', dc)
if X2==inf & H~=0
fprintf('Critical clearing time= %7.3f \n\n', dc)
else, end
h = figure;
figure(h);
fill(x,y,'m')
hold;
fill(xx,hh,'c')
plot(delta, Pe1,'-', delta, Pe2,'r-', delta, Pe3,'g-', Pmx, Pmy,'b-', x0,y0, xc,yc, xm,ym),
grid on
title('Application of equal area criterion to a critically clearedsystem')
xlabel('Power angle, degree'),
ylabel(' Power, per unit')
text(5, 1.07*Pm, 'Pm')
text(50, 1.05*Pe1max,['Critical clearing angle = ',num2str(dc)])
axis([0 180 0 1.1*Pe1max])
hold off;
```

8] DIFFERENT TYPES OF FAULTS

NOTE: 7 FILES TO BE EXECUTED

1] zff1.m

2] zff2.m

3] main.m

4] sysmfault.m

5] llfault.m

6] lgfault.m

7] dlgfault.m

Zff1.m

```
%z-bus formation-For positive symmetrical components
function [Zbus] = zff1(linedata)
linedata =[0 1 0 0.25
0 2 0 0.25
1 2 0 0.125
1 3 0 0.15
2 3 0 0.25]
nl = linedata(:,1);
nr = linedata(:,2);
R = linedata(:,3);
X = linedata(:,4);
nbr=length(linedata(:,1));
nbus = max(max(nl), max(nr));
for k=1:nbr
if R(k) == inf | X(k) ==inf
R(k) = 9999;
X(k) = 9999;
else, end
end
ZB = R + j*X;
Zbus = zeros(nbus, nbus);
tree=0; %%%%new
% Adding a branch from a new bus to reference bus 0
for I = 1:nbr
ntree(I) = 1;
if nl(I) == 0 | nr(I) == 0
if nl(I) == 0 n = nr(I);
elseif nr(I) == 0 n = nl(I);
end
if abs(Zbus(n, n)) == 0 Zbus(n,n) =ZB(I);tree=tree+1; %%new
else Zbus(n,n) = Zbus(n,n)*ZB(I)/(Zbus(n,n) + ZB(I));
end
ntree(I) = 2;
else,end
end
% Adding a branch from new bus to an existing bus
while tree < nbus %%% new
for n = 1:nbus
nadd = 1;
if abs(Zbus(n,n)) == 0
for I = 1:nbr
if nadd == 1;
if nl(I) == n | nr(I) == n
if nl(I) == n k = nr(I);
elseif nr(I) == n k = nl(I);
end
if abs(Zbus(k,k)) ~= 0
for m = 1:nbus
if m ~= n
Zbus(m,n) = Zbus(m,k);
Zbus(n,m) = Zbus(m,k);
else, end
end
Zbus(n,n) = Zbus(k,k) + ZB(I);
tree=tree+1; %%new
nadd = 2;
ntree(I) = 2;
else, end
else, end
else, end
end
else, end
end
```

```
end %%%%%%new
% Adding a link between two old buses
for n = 1:nbus
for I = 1:nbr
if ntree(I) == 1
if nl(I) == n | nr(I) == n
if nl(I) == n k = nr(I);
elseif nr(I) == n k = nl(I);
end
DM = Zbus(n,n) + Zbus(k,k) + ZB(I) - 2*Zbus(n,k);
for jj = 1:nbus
AP = Zbus(jj,n) - Zbus(jj,k);
for kk = 1:nbus
AT = Zbus(n,kk) - Zbus(k, kk);
DELZ(jj,kk) = AP*AT/DM;
end
end
Zbus = Zbus - DELZ;
ntree(I) = 2;
else,end
else,end
end
end
```

zff2.m

```
function [Zbus] = hzff2(linedata)
linedata =[0 1 0 0.40
0 2 0 0.10
1 2 0 0.30
1 3 0 0.35
2 3 0 0.7125];
nl = linedata(:,1);
nr = linedata(:,2);
R = linedata(:,3);
X = linedata(:,4);
nbr=length(linedata(:,1))
;nbus = max(max(nl), max(nr))
for k=1:nbr
if R(k) == inf | X(k) ==inf
R(k) = 9999;
X(k) = 9999;
else, end
end
ZB = R + j*X;
Zbus = zeros(nbus, nbus);
tree=0; %%%%new
% Adding a branch from a new bus to reference bus 0
for I = 1:nbr
ntree(I) = 1;
if nl(I) == 0 | nr(I) == 0
if nl(I) == 0 n = nr(I);
elseif nr(I) == 0 n = nl(I);
end
if abs(Zbus(n, n)) == 0 Zbus(n,n) =ZB(I);tree=tree+1; %%new
else Zbus(n,n) = Zbus(n,n)*ZB(I)/(Zbus(n,n) + ZB(I));
end
ntree(I) = 2;
else,end
end
% Adding a branch from new bus to an existing bus
```

```
while tree < nbus %%% new
for n = 1:nbus
nadd = 1;
if abs(Zbus(n,n)) == 0
for I = 1:nbr
if nadd == 1;
if nl(I) == n | nr(I) == n
if nl(I) == n k = nr(I);
elseif nr(I) == n k = nl(I);
end
if abs(Zbus(k,k)) ~= 0
for m = 1:nbus
if m ~= n
Zbus(m,n) = Zbus(m,k);
Zbus(n,m) = Zbus(m,k);
else, end
end
Zbus(n,n) = Zbus(k,k) + ZB(I);
tree=tree+1; %%new
nadd = 2;
ntree(I) = 2;
else, end
else, end
else, end
end
else, end
end
end %%%%%%new
% Adding a link between two old buses
for n = 1:nbus
for I = 1:nbr
if ntree(I) == 1
if nl(I) == n | nr(I) == n
if nl(I) == n k = nr(I);
elseif nr(I) == n k = nl(I);
end
DM = Zbus(n,n) + Zbus(k,k) + ZB(I) - 2*Zbus(n,k);
for jj = 1:nbus
AP = Zbus(jj,n) - Zbus(jj,k);
for kk = 1:nbus
AT = Zbus(n,kk) - Zbus(k, kk);
DELZ(jj,kk) = AP*AT/DM;
end
end
Zbus = Zbus - DELZ;
ntree(I) = 2;
else,end
else,end
end
end
```

Main.m

```
zdata1 = [0 1 0 0.25
0 2 0 0.25
1 2 0 0.125
1 3 0 0.15
2 3 0 0.25];
zdata0=[0 1 0 0.40
0 2 0 0.10
1 2 0 0.30
1 3 0 0.35
2 3 0 0.7125];
zdata2 = zdata1;
Zbus1 = zff1(zdata1);
Zbus0 = zff2(zdata0);
Zbus2 = Zbus1;
symfault(zdata1,Zbus1);
lgfault(zdata0, Zbus0, zdata1, Zbus1, zdata2, Zbus2);
llfault(zdata1, Zbus1, zdata2, Zbus2);
dlgfault(zdata0, Zbus0, zdata1, Zbus1, zdata2, Zbus2);
```

sysmfault.m

```
function symfault(zdata, Zbus, V)
nl = zdata(:,1);
nr = zdata(:,2);
R = zdata(:,3);
X = zdata(:,4);
nc = length(zdata(1,:));
if nc > 4
BC = zdata(:,5);
elseif nc ==4, BC = zeros(length(zdata(:,1)), 1);
end
ZB = R + j*X;
nbr=length(zdata(:,1)); nbus = max(max(nl), max(nr)); if exist('V')== 1
if length(V) == nbus
V0 = V;
else, end
else, V0 = ones(nbus, 1) + j*zeros(nbus, 1);
end
fprintf('Three-phase balanced fault analysis \n')
fprintf('SYMMETRICAL FAULTS\n')
ff = 999;
while ff > 0
nf = input('Enter Faulted Bus No. -> ');
while nf <= 0 | nf > nbus
fprintf('Faulted bus No. must be between 1 & %g \n', nbus)
nf = input('Enter Faulted Bus No. -> ');
end
fprintf('\nEnter Fault Impedance Zf = R + j*X in ')
Zf = input('complex form (for bolted fault enter 0). Zf = ');
fprintf(' \n')
fprintf('Balanced three-phase fault at bus No. %g\n', nf)
If = V0(nf)/(Zf + Zbus(nf, nf));
Ifm = abs(If); Ifmang=angle(If)*180/pi;
fprintf('Total fault current = %8.4f per unit\n', Ifm) %fprintf('p.u. \n\n', Ifm)
fprintf('Bus Voltages during fault in per unit \n')
fprintf(' Bus Voltage Angle\n')
```

```
fprintf(' No. Magnitude degrees\n')
for n = 1:nbus
if n==nf
Vf(nf) = V0(nf)*Zf/(Zf + Zbus(nf,nf));
Vfm = abs(Vf(nf));
angv=angle(Vf(nf))*180/pi;
else, Vf(n) = V0(n) - V0(n)*Zbus(n,nf)/(Zf + Zbus(nf,nf));
Vfm = abs(Vf(n)); angv=angle(Vf(n))*180/pi;
end
fprintf(' %4g', n), fprintf('%13.4f', Vfm),fprintf('%13.4f\n', angv)
end
fprintf(' \n')
fprintf('Line currents for fault at bus No. %g\n\n', nf)
fprintf(' From To Current Angle\n')
fprintf(' Bus Bus Magnitude degrees\n')
for n= 1:nbus %Ign=0;
for I = 1:nbr
if nl(I) == n || nr(I) == n
if nl(I) ==n k = nr(I);
elseif nr(I) == n k = nl(I);
end
if k==0
Ink = (V0(n) - Vf(n))/ZB(I);
Inkm = abs(Ink);
th=angle(Ink);
%if th <= 0
if real(Ink) > 0
fprintf(' G'), fprintf(' %7g',n), fprintf('%12.4f', Inkm),
fprintf('%12.4f\n', th*180/pi)
elseif real(Ink) ==0 & imag(Ink) < 0
fprintf(' G'), fprintf(' %7g',n), fprintf('%12.4f', Inkm),
fprintf('%12.4f\n', th*180/pi)
else, end
Ign=Ink;
elseif k ~= 0
Ink = (Vf(n) - Vf(k))/ZB(I)+BC(I)*Vf(n); %Ink = (Vf(n) -Vf(k))/ZB(I);
Inkm = abs(Ink); th=angle(Ink); %Ign=Ign+Ink;
%if th <= 0
if real(Ink) > 0
fprintf('%7g', n), fprintf('%10g', k), fprintf('%12.4f', Inkm),
fprintf('%12.4f\n', th*180/pi)
elseif real(Ink) ==0 & imag(Ink) < 0
fprintf('%7g', n), fprintf('%10g', k), fprintf('%12.4f', Inkm),
fprintf('%12.4f\n', th*180/pi)
else, end
else, end
else, end
end
if n==nf
fprintf('%7g',n), fprintf(' F'), fprintf('%12.4f', Ifm),
fprintf('%12.4f\n', Ifmang)
else, end
end
resp=0;
while strcmp(resp, 'n')~=1 && strcmp(resp, 'N')~=1 & strcmp(resp,'y')~=1 & strcmp(resp, 'Y')~=1
resp = input('Another fault location? Enter ''y'' or ''n'' withinsingle quote -> ');
if strcmp(resp, 'n')~=1 & strcmp(resp, 'N')~=1 & strcmp(resp,'y')~=1 & strcmp(resp, 'Y')~=1
fprintf('\n Incorrect reply, try again \n\n'),
end
end
if resp == 'y' | resp == 'Y'
nf = 999;
else ff = 0;
end
end
```

```
end
```

llfault.m

```
function llfault(zdata1, Zbus1, zdata2, Zbus2, V)
if exist('zdata2') ~= 1
zdata2=zdata1;
else, end
if exist('Zbus2') ~= 1
Zbus2=Zbus1;
else, end
nl = zdata1(:,1);
nr = zdata1(:,2);
R1 = zdata1(:,3);
X1 = zdata1(:,4);
R2 = zdata2(:,3);
X2 = zdata2(:,4);
ZB1=R1+j*X1;
ZB2=R2+j*X2;
nbr=length(zdata1(:,1));
nbus = max(max(nl), max(nr));
if exist('V') == 1
if length(V) == nbus
V0 = V;
else, end
else,
V0 = ones(nbus, 1) + j*zeros(nbus, 1);
end
fprintf('\nLine-to-line fault analysis \n')
ff = 999;
while ff > 0
nf = input('Enter Faulted Bus No. -> ');
while nf <= 0 | nf > nbus
fprintf('Faulted bus No. must be between 1 & %g \n', nbus)
nf = input('Enter Faulted Bus No. -> ');
end
fprintf('\nEnter Fault Impedance Zf = R + j*X in ')
Zf = input('complex form (for bolted fault enter 0). Zf = ');
fprintf(' \n')
fprintf('Line-to-line fault at bus No. %g\n', nf)
a =cos(2*pi/3)+j*sin(2*pi/3);
sctm = [1 1 1; 1 a^2 a; 1 a a^2];
Ia0=0;
Ia1 = V0(nf)/(Zbus1(nf,nf)+Zbus2(nf, nf)+Zf);
Ia2=-Ia1;
I012=[Ia0; Ia1; Ia2];
Ifabc = sctm*I012;
Ifabcm = abs(Ifabc);
fprintf('Total fault current = %9.4f per unit\n\n', Ifabcm(2))
fprintf('Bus Voltages during the fault in per unit \n\n')
fprintf(' Bus ---------Voltage Magnitude-------- \n')
fprintf(' No. Phase a Phase b Phase c \n')
for n = 1:nbus
Vf0(n)= 0;
Vf1(n)= V0(n) - Zbus1(n, nf)*Ia1;
Vf2(n)= 0 - Zbus2(n, nf)*Ia2;
Vabc = sctm*[Vf0(n);
Vf1(n); Vf2(n)];
Va(n)=Vabc(1);
Vb(n)=Vabc(2);
Vc(n)=Vabc(3);
```

```
fprintf('%5g',n)
fprintf(' %11.4f', abs(Va(n))),fprintf('%11.4f', abs(Vb(n)))
fprintf('%11.4f\n', abs(Vc(n)))
end
fprintf(' \n')
fprintf('Line currents for fault at bus No. %g\n\n', nf)
fprintf(' From To -----Line Current Magnitude---- \n')
fprintf(' Bus Bus Phase a Phase b Phase c \n')
for n= 1:nbus
for I = 1:nbr
if nl(I) == n | nr(I) == n
if nl(I) ==n
k = nr(I);
elseif nr(I) == n
k = nl(I);
end
if k ~= 0
Ink0(n, k) = 0;
Ink1(n, k) = (Vf1(n) - Vf1(k))/ZB1(I);
Ink2(n, k) = (Vf2(n) - Vf2(k))/ZB2(I);
Inkabc = sctm*[Ink0(n, k); Ink1(n, k);
Ink2(n, k)];
Inkabcm = abs(Inkabc);
th=angle(Inkabc);
if real(Inkabc(2)) < 0
fprintf('%7g', n), fprintf('%10g',k),
fprintf('%11.4f', abs(Inkabc(1))),fprintf('%11.4f',abs(Inkabc(2)))
fprintf('%11.4f\n', abs(Inkabc(3)))
elseif real(Inkabc(2)) ==0 & imag(Inkabc(2)) > 0
fprintf('%7g', n), fprintf('%10g', k), fprintf('%11.4f',abs(Inkabc(1))),fprintf(' %11.4f',
abs(Inkabc(2)))
fprintf('%11.4f\n',abs(Inkabc(3)))
else, end
else, end
else, end
end
if n==nf
fprintf('%7g ',n), fprintf(' F'),
fprintf(' %11.4f', Ifabcm(1)),fprintf(' %11.4f', Ifabcm(2))
fprintf(' %11.4f\n', Ifabcm(3))
else, end
end
resp=0;
while strcmp(resp, 'n')~=1 & strcmp(resp, 'N')~=1 & strcmp(resp,'y')~=1 & strcmp(resp, 'Y')~=1
resp = input('Another fault location? Enter ''y'' or ''n'' withinsingle quote-> ');
if strcmp(resp, 'n')~=1 & strcmp(resp, 'N')~=1 & strcmp(resp,'y')~=1 & strcmp(resp, 'Y')~=1
fprintf('\n Incorrect reply, try again \n\n'),
end
end
if resp == 'y' | resp == 'Y'
nf = 999;
else ff = 0;
end
end
```

lgfault.m

```
function lgfault(zdata0, Zbus0, zdata1, Zbus1, zdata2, Zbus2, V)
if exist('zdata2') ~= 1
zdata2=zdata1;
else, end
```

```
if exist('Zbus2') ~= 1 Zbus2=Zbus1;
else, end
nl = zdata1(:,1);
nr = zdata1(:,2);
nl0 = zdata0(:,1);
nr0 = zdata0(:,2);
nbr=length(zdata1(:,1));
nbus = max(max(nl), max(nr));
nbr0=length(zdata0(:,1));
R0 = zdata0(:,3);
X0 = zdata0(:,4);
R1 = zdata1(:,3);
X1 = zdata1(:,4);
R2 = zdata1(:,3);
X2 = zdata1(:,4);
for k=1:nbr0
if R0(k)==inf | X0(k) ==inf
R0(k) = 99999999; X0(k) = 99999999;
else, end
end
ZB1=R1+j*X1;
ZB0=R0+j*X0;
ZB2 = R2 + j*X2;
if exist('V') == 1
if length(V) == nbus V0 = V;
else, end
else, V0 = ones(nbus, 1) + j*zeros(nbus, 1);
end
fprintf('\nLINE TO GROUND FAULT\n')
ff = 999;
while ff > 0
nf = input('Enter Faulted Bus No. -> ');
while nf <= 0 | nf > nbus
fprintf('Faulted bus No. must be between 1 & %g \n', nbus)
nf = input('Enter Faulted Bus No. -> ');
end
fprintf('Enter Fault Impedance Zf = R + j*X in ')
Zf = input('complex form (for bolted fault enter 0). Zf = ');
fprintf('Single line-to-ground fault at bus No. %g\n', nf)
a =cos(2*pi/3)+j*sin(2*pi/3);
sctm = [1 1 1; 1 a^2 a; 1 a a^2];
Ia0 = V0(nf)/(Zbus1(nf,nf)+Zbus2(nf, nf)+ Zbus0(nf, nf)+3*Zf);
Ia1=Ia0;
Ia2=Ia0;
I012=[Ia0; Ia1; Ia2];
Ifabc = sctm*I012;
Ifabcm = abs(Ifabc);
fprintf('Total fault current = %9.4f per unit\n', Ifabcm(1))
fprintf('Bus Voltages during the fault in per unit \n')
fprintf(' Bus ---------Voltage Magnitude--------- \n')
fprintf(' No. Phase a Phase b Phase c \n')
for n = 1:nbus
Vf0(n)= 0 - Zbus0(n, nf)*Ia0;
Vf1(n)= V0(n) - Zbus1(n, nf)*Ia1;
Vf2(n)= 0 - Zbus2(n, nf)*Ia2;
Vabc = sctm*[Vf0(n);
Vf1(n);
Vf2(n)];
Va(n)=Vabc(1);
Vb(n)=Vabc(2);
Vc(n)=Vabc(3);
fprintf(' %5g',n)
fprintf(' %11.4f', abs(Va(n))),fprintf(' %11.4f', abs(Vb(n)))
fprintf(' %11.4f\n', abs(Vc(n)))
end
```

```
fprintf(' \n')
fprintf('Line currents for fault at bus No. %g\n', nf)
fprintf(' From To -------Line Current Magnitude ------- \n')
fprintf(' Bus Bus Phase a Phase b Phase c\n')
for n= 1:nbus
for I = 1:nbr
if nl(I) == n | nr(I) == n
if nl(I) ==n k = nr(I);elseif nr(I) == n k = nl(I); end
if k ~= 0
Ink1(n, k) = (Vf1(n) - Vf1(k))/ZB1(I); Ink2(n, k) = (Vf2(n) -Vf2(k))/ZB2(I); else, end
else, end
end
for I = 1:nbr0
if nl0(I) == n | nr0(I) == n
if nl0(I) ==n k = nr0(I);elseif nr0(I) == n k = nl0(I); end
if k ~= 0
Ink0(n, k) = (Vf0(n) - Vf0(k))/ZB0(I);
else, end
else, end
end
for I = 1:nbr
if nl(I) == n | nr(I) == n
if nl(I) ==n k = nr(I);elseif nr(I) == n k = nl(I); end
if k ~= 0
Inkabc = sctm*[Ink0(n, k);
Ink1(n, k);
Ink2(n, k)];
Inkabcm = abs(Inkabc);
th=angle(Inkabc);
if real(Inkabc(1)) > 0
fprintf('%7g', n), fprintf('%10g', k),
fprintf(' %11.4f', abs(Inkabc(1))),fprintf(' %11.4f',abs(Inkabc(2)))
fprintf(' %11.4f\n', abs(Inkabc(3)))
elseif real(Inkabc(1)) ==0 & imag(Inkabc(1)) < 0
fprintf('%7g', n), fprintf('%10g', k),
fprintf(' %11.4f', abs(Inkabc(1))),fprintf(' %11.4f',abs(Inkabc(2)))
fprintf(' %11.4f\n', abs(Inkabc(3)))
else, end
else, end
else, end
end
if n==nf
fprintf('%7g',n), fprintf(' F'),
fprintf(' %11.4f', Ifabcm(1)),fprintf(' %11.4f', Ifabcm(2))
fprintf(' %11.4f\n', Ifabcm(3))
else, end
end
resp=0;
while strcmp(resp, 'n')~=1 && strcmp(resp, 'N')~=1 & strcmp(resp,'y')~=1 & strcmp(resp, 'Y')~=1
resp = input('Another fault location? Enter ''y'' or ''n'' withinsingle quote -> ');
if strcmp(resp, 'n')~=1 & strcmp(resp, 'N')~=1 & strcmp(resp,'y')~=1 & strcmp(resp, 'Y')~=1
fprintf('\n Incorrect reply, try again \n\n'),
end
end
if resp == 'y' | resp == 'Y'
nf = 999;
else ff = 0;
end
end
end
```

dlgfault.m

```
function dlgfault(zdata0, Zbus0, zdata1, Zbus1, zdata2, Zbus2, V)
if exist('zdata2') ~= 1
zdata2=zdata1;
else, end
if exist('Zbus2') ~= 1 Zbus2=Zbus1;
else, end
nl = zdata1(:,1);
nr = zdata1(:,2);
nl0 = zdata0(:,1);
nr0 = zdata0(:,2);
nbr=length(zdata1(:,1));
nbus = max(max(nl), max(nr));
nbr0=length(zdata0(:,1));
R0 = zdata0(:,3);
X0 = zdata0(:,4);
R1 = zdata1(:,3);
X1 = zdata1(:,4);
R2 = zdata2(:,3);
X2 = zdata2(:,4);
for k = 1:nbr0
if R0(k) == inf | X0(k) == inf
R0(k) = 99999999;
X0(k) = 999999999;
else, end
end
ZB1=R1+j*X1;
ZB0=R0+j*X0;
ZB2 = R2 + j*X2;
if exist('V') == 1
if length(V) == nbus V0 = V;
else, end
else, V0 = ones(nbus, 1) + j*zeros(nbus, 1);
end
fprintf('\nDOUBLE LINE-TO-GROUND FAULT\n')
ff = 999;
while ff > 0
nf = input('Enter Faulted Bus No. -> ');
while nf <= 0 | nf > nbus
fprintf('Faulted bus No. must be between 1 & %g \n', nbus)
nf = input('Enter Faulted Bus No. -> ');
end
fprintf('\nEnter Fault Impedance Zf = R + j*X in ')
Zf = input('complex form (for bolted fault enter 0). Zf = ');
fprintf(' \n')
fprintf('DOUBLE LINE-TO-GROUND FAULT at bus No. %g\n', nf)
a =cos(2*pi/3)+j*sin(2*pi/3);
sctm = [1 1 1; 1 a^2 a; 1 a a^2];
Z11 = Zbus2(nf, nf)*(Zbus0(nf, nf)+ 3*Zf)/(Zbus2(nf, nf)+Zbus0(nf,nf)+3*Zf);
Ia1 = V0(nf)/(Zbus1(nf,nf)+Z11);
Ia2 =-(V0(nf) - Zbus1(nf, nf)*Ia1)/Zbus2(nf,nf);
Ia0 =-(V0(nf) - Zbus1(nf, nf)*Ia1)/(Zbus0(nf,nf)+3*Zf); I012=[Ia0;
Ia1; Ia2];
Ifabc = sctm*I012;
Ifabcm=abs(Ifabc);
Ift = Ifabc(2)+Ifabc(3);
Iftm = abs(Ift);
fprintf('Total fault current = %9.4f per unit\n\n', Iftm)
fprintf('Bus Voltages during the fault in per unit \n\n')
fprintf(' Bus --------Voltage Magnitude---------- \n')
fprintf(' No. Phase a Phase b Phase c \n')
for n = 1:nbus
```

```
Vf0(n)= 0 - Zbus0(n, nf)*Ia0;
Vf1(n)= V0(n) - Zbus1(n, nf)*Ia1;
Vf2(n)= 0 - Zbus2(n, nf)*Ia2;
Vabc = sctm*[Vf0(n); Vf1(n); Vf2(n)];
Va(n)=Vabc(1); Vb(n)=Vabc(2); Vc(n)=Vabc(3);
fprintf(' %5g',n)
fprintf(' %11.4f', abs(Va(n))),fprintf(' %11.4f', abs(Vb(n)))
fprintf(' %11.4f\n', abs(Vc(n)))
end
fprintf(' \n')
fprintf('Line currents for fault at bus No. %g\n\n', nf)
fprintf(' From To ------Line Current Magnitude------ \n')
fprintf(' Bus Bus Phase a Phase b Phase c \n')
for n= 1:nbus
for I = 1:nbr
if nl(I) == n | nr(I) == n
if nl(I)==n k = nr(I);
elseif nr(I) == n k = nl(I);
end
if k ~= 0
Ink1(n, k) = (Vf1(n) - Vf1(k))/ZB1(I);
Ink2(n, k) = (Vf2(n) - Vf2(k))/ZB2(I);
else, end
else, end
end
for I = 1:nbr0
if nl0(I) == n | nr0(I) == n
if nl0(I) ==n k = nr0(I);
elseif nr0(I) == n k = nl0(I);
end
if k ~= 0
Ink0(n, k) = (Vf0(n) - Vf0(k))/ZB0(I);
else, end
else, end
end
for I = 1:nbr
if nl(I) == n | nr(I) == n
if nl(I) ==n k = nr(I);
elseif nr(I) == n k = nl(I);
end
if k ~= 0
Inkabc = sctm*[Ink0(n, k); Ink1(n, k); Ink2(n, k)];
Inkabcm = abs(Inkabc);
th=angle(Inkabc);
if real(Inkabc(2)) < 0
fprintf('%7g', n), fprintf('%10g', k),
fprintf(' %11.4f', abs(Inkabc(1))),fprintf(' %11.4f',abs(Inkabc(2)))
fprintf(' %11.4f\n', abs(Inkabc(3)))
elseif real(Inkabc(2)) ==0 & imag(Inkabc(2)) > 0 fprintf('%7g', n),
fprintf('%10g', k),
fprintf(' %11.4f', abs(Inkabc(1))),fprintf (' %11.4f',abs(Inkabc(2)))
fprintf(' %11.4f\n', abs(Inkabc(3)))
else, end
else, end
else, end
end
if n==nf
fprintf('%7g',n), fprintf(' F'),
fprintf(' %11.4f', Ifabcm(1)),fprintf(' %11.4f', Ifabcm(2))
fprintf(' %11.4f\n', Ifabcm(3))
else, end
end
resp=0;
while strcmp(resp, 'n')~=1 && strcmp(resp, 'N')~=1 & strcmp(resp,'y')~=1 & strcmp(resp, 'Y')~=1
resp = input('Another fault location? Enter ''y'' or ''n'' withinsingle quote -> ');
```

```
if strcmp(resp, 'n')~=1 & strcmp(resp, 'N')~=1 & strcmp(resp,'y')~=1 & strcmp(resp, 'Y')~=1
fprintf('\n Incorrect reply, try again \n\n'),
end
end
if resp == 'y' | resp == 'Y'
nf = 999;
else ff = 0;
end
end
end
```

OUTPUTS

Enter Faulted Bus No. ->

3

Enter Fault Impedance Zf = R + j*X in complex form (for bolted fault enter 0). Zf =

j*0.1

Balanced three-phase fault at bus No. 3

Total fault current = 3.1250 per unit

Bus Voltages during fault in per unit

Bus No.	Voltage Magnitude	Angle degrees
1	0.5938	0.0000
2	0.6250	0.0000
3	0.3125	0.0000

Line currents for fault at bus No. 3

From Bus	To Bus	Current Magnitude	Angle degrees
G	1	1.6250	-90.0000
1	3	1.8750	-90.0000
G	2	1.5000	-90.0000
2	1	0.2500	-90.0000
2	3	1.2500	-90.0000
3	F	3.1250	-90.0000

Another fault location? Enter 'y' or 'n' withinsingle quote ->

'n'

LINE TO GROUND FAULT

Enter Faulted Bus No. ->

3

Enter Fault Impedance Zf = R + j*X in complex form (for bolted fault enter 0). Zf =

j*0.1

Single line-to-ground fault at bus No. 3

Total fault current = 2.7523 per unit

Bus Voltages during the fault in per unit

Bus	Voltage Magnitude		
No.	Phase a	Phase b	Phase c
1	0.6330	1.0046	1.0046
2	0.7202	0.9757	0.9757
3	0.2752	1.0647	1.0647

Line currents for fault at bus No. 3

From Bus	To Bus	Line Current Magnitude		
		Phase a	Phase b	Phase c
1	3	1.6514	0.0000	0.0000
2	1	0.3761	0.1560	0.1560
2	3	1.1009	0.0000	0.0000
3	F	2.7523	0.0000	0.0000

Another fault location? Enter 'y' or 'n' withinsingle quote ->

'n'

Line-to-line fault analysis

Enter Faulted Bus No. ->

3

Enter Fault Impedance Zf = R + j*X in complex form (for bolted fault enter 0). Zf =

j*0.1

Line-to-line fault at bus No. 3

Total fault current = 3.2075 per unit

Bus Voltages during the fault in per unit

Bus	Voltage Magnitude		
No.	Phase a	Phase b	Phase c
1	1.0000	0.6720	0.6720

2 1.0000 0.6939 0.6939

3 1.0000 0.5251 0.5251

Line currents for fault at bus No. 3

From To -----Line Current Magnitude----

Bus Bus Phase a Phase b Phase c

1 3 0.0000 1.9245 1.9245

2 1 0.0000 0.2566 0.2566

2 3 0.0000 1.2830 1.2830

3 F 0.0000 3.2075 3.2075

Another fault location? Enter 'y' or 'n' withinsingle quote->

'n'

DOUBLE LINE-TO-GROUND FAULT

Enter Faulted Bus No. ->

3

Enter Fault Impedance Zf = R + j*X in complex form (for bolted fault enter 0). Zf =

j*0.1

DOUBLE LINE-TO-GROUND FAULT at bus No. 3

Total fault current = 1.9737 per unit

Bus Voltages during the fault in per unit

Bus --------Voltage Magnitude----------

No. Phase a Phase b Phase c

1 1.0066 0.5088 0.5088

2 0.9638 0.5740 0.5740

3 1.0855 0.1974 0.1974

Line currents for fault at bus No. 3

From To ------Line Current Magnitude------

Bus Bus Phase a Phase b Phase c

1 3 0.0000 2.4350 2.4350

2 1 0.1118 0.3682 0.3682

2 3 0.0000 1.6233 1.6233

```
     3 F      0.0000      4.0583      4.0583

Another fault location? Enter 'y' or 'n' withinsingle quote ->

'n'

>>
```