```
//Programa de estimación de estado, autor: Hans Alayo

clear;
clc;
funcprot(0);
//Parametros de la simulación
MVABase=100;    //Potencia Base
Max_iter=10;    //Maximo de iteraciones
error=1e-6;     //Maximo error
pi=3.14159265;    //constante pi

//Datos del problema, colocar archivo de datos
exec wood.sce;

//Dimension de problema
nbus=size(Datos_Barras,1);          //Numero de barras
nlines=size(Datos_lineas,1);        //Numero de ramas
//Parametros lineas
fb=Datos_lineas(:,1);
tb=Datos_lineas(:,2);
z=Datos_lineas(:,3)+%i*Datos_lineas(:,4);
for i=1:nlines
y(i)=1/z(i);
end
gl=real(y);
bl=imag(y);
b=%i*Datos_lineas(:,5);
bsh=imag(b);
a=Datos_lineas(:,6)
//Parametros Barras
bus=Datos_Barras(:,1);
ysh=%i*Datos_Barras(:,7);
swing=1;

//Mediciones disponibles
//-----------------------------------------------------
Pij=Datos_lineas(:,7)
Qij=Datos_lineas(:,8)
Pji=Datos_lineas(:,9)
Qji=Datos_lineas(:,10)

bPij=Datos_lineas(:,11)
bQij=Datos_lineas(:,12)
bPji=Datos_lineas(:,13)
bQji=Datos_lineas(:,14)

Pg=Datos_Barras(:,5)
```

```
Qg=Datos_Barras(:,6)
Pl=Datos_Barras(:,3)
Ql=Datos_Barras(:,4)

bV=Datos_Barras(:,8);
bP=Datos_Barras(:,9);
bQ=Datos_Barras(:,10);

//Mapeos de datos y mediciones
//-----------------------------------------------------
//buses
//-----------------------------------------------------
//Inyecciones de potencia Pi
i=0;
P=[];
RP=[];
for k=1:nbus
   if (bP(k)==1) then
      i=i+1;
      P(i)=(Pg(k)-Pl(k))/MVABase;
      iP(i)=k;
      RP(i)=Datos_Barras(k,12)^2;
   end
end

//Inyecciones de potencia Qi
i=0;
Q=[];
RQ=[];
for k=1:nbus
   if (bQ(k)==1) then
      i=i+1;
      Q(i)=(Qg(k)-Ql(k))/MVABase;
      iQ(i)=k;
      RQ(i)=Datos_Barras(k,13)^2;
   end
end

//Tensiones Vi
i=0;
V=[];
RV=[];
for k=1:nbus
   if (bV(k)==1) then
      i=i+1;
      V(i)=Datos_Barras(k,2);
      iV(i)=k;
```

```
        RV(i)=Datos_Barras(k,11)^2;
    end
end

//lineas
//-----------------------------------------
//Flujos Pij
i=0;
Pij=[];
RPij=[];
for k=1:nlines
    if (bPij(k)==1) then
        i=i+1;
        Pij(i)=Datos_lineas(k,7)/MVABase;
        iPij(i)=k;
        RPij(i)=Datos_lineas(k,15)^2;
    end
end

//Flujos Qij
i=0;
Qij=[];
RQij=[];
for k=1:nlines
    if (bQij(k)==1) then
        i=i+1;
        Qij(i)=Datos_lineas(k,8)/MVABase;
        iQij(i)=k;
        RQij(i)=Datos_lineas(k,16)^2;
    end
end

//Flujos Pji
i=0;
Pji=[];
RPji=[];
for k=1:nlines
    if (bPji(k)==1) then
        i=i+1;
        Pji(i)=Datos_lineas(k,9)/MVABase;
        iPji(i)=k;
        RPji(i)=Datos_lineas(k,17)^2;
    end
end

//Flujos Qji
i=0;
```

```
Qji=[];
RQji=[];
for k=1:nlines
   if (bQji(k)==1) then
      i=i+1;
      Qji(i)=Datos_lineas(k,10)/MVABase;
      iQji(i)=k;
      RQji(i)=Datos_lineas(k,18)^2;
   end
end

//Matriz de admitancias - Modelo de Red
//-------------------------------------------------------------
Y=zeros(nbus,nbus);
for k=1:nlines
    Y(fb(k),tb(k)) = Y(fb(k),tb(k))-y(k)/a(k);
    Y(tb(k),fb(k)) = Y(fb(k),tb(k));
end
for m =1:nbus
    for n =1:nlines
       if fb(n) == m
          Y(m,m) = Y(m,m) + y(n)/(a(n)^2) + b(n);
       elseif tb(n) == m
          Y(m,m) = Y(m,m) + y(n) + b(n);
       end
    end
end

for i=1:nbus
   Y(bus(i),bus(i))=Y(bus(i),bus(i))+ysh(i)
end

G=real(Y);
B=imag(Y);

//Formación de vector de mediciones

Z=[P;Pij;Pji;Q;Qij; Qji; V];

R=diag([RP;RPij;RPji;RQ;RQij;Qji;RV]);

//Estimacion inicial de variables de estado

Ve=ones(nbus,1);
Ae=zeros(nbus,1);

//Comienzo de iteraciones
```

```
//-----------------------------------------------------------
iter=0;
while(iter<Max_iter)

//Formación de la matriz H
//-----------------------------------
dVdA=zeros(length(iV),nbus-1)

//-----------------------------------------------------
dVdV=zeros(length(iV),nbus)

for k=1:length(iV)
   for m=1:nbus
      if bus(iV(k))==m then
         dVdV(k,m)=1;
      end
   end
end

//-----------------------------------------------------------
if P==[] then
dPdA=[]
else
dPdA=zeros(length(iP),nbus-1)
for k=1:length(iP)
   t=0;
   for j=1:nbus
      i=bus(iP(k))
      if j==swing then
      else
      t=t+1
      if i==j then
         sum=0;
         for tp=1:nbus
         sum=sum+Ve(i)*Ve(tp)*(-G(i,tp)*sin(Ae(i)-Ae(tp))+B(i,tp)*cos(Ae(i)-Ae(tp)));
         end
         dPdA(k,t)=sum-B(i,i)*Ve(i)^2;
      else
         dPdA(k,t)=Ve(i)*Ve(j)*(G(i,j)*sin(Ae(i)-Ae(j))-B(i,j)*cos(Ae(i)-Ae(j)));
      end
      end
   end
end
end
//-----------------------------------------------------------
if P==[] then
dPdV=[]
```

```
else
dPdV=zeros(length(iP),nbus);
for k=1:length(iP)
   for j=1:nbus
      i=bus(iP(k));
      if i==j then
         sum=0;
         for tp=1:nbus
         sum=sum+Ve(tp)*(G(i,tp)*cos(Ae(i)-Ae(tp))+B(i,tp)*sin(Ae(i)-Ae(tp)));
         end
         dPdV(k,j)=sum+G(i,i)*Ve(i);
      else
         dPdV(k,j)=Ve(i)*(G(i,j)*cos(Ae(i)-Ae(j))+B(i,j)*sin(Ae(i)-Ae(j)));
      end
   end
end
end
//---------------------------------------------------------------

if Q==[] then
dQdA=[];
else
dQdA=zeros(length(iQ),nbus-1);
for k=1:length(iQ)
   t=0;
   for j=1:nbus
      i=bus(iQ(k));
      if j==swing then
      else
      t=t+1;
      if i==j then
         sum=0;
         for tp=1:nbus
         sum=sum+Ve(i)*Ve(tp)*(G(i,tp)*cos(Ae(i)-Ae(tp))+B(i,tp)*sin(Ae(i)-Ae(tp)));
         end
         dQdA(k,t)=sum-G(i,i)*Ve(i)^2;
      else
         dQdA(k,t)=Ve(i)*Ve(j)*(-G(i,j)*cos(Ae(i)-Ae(j))-B(i,j)*sin(Ae(i)-Ae(j)));
      end
      end
   end
end
end
//-----------------------------------------------------------
if Q==[] then
dQdV=[]
else
```

```
dQdV=zeros(length(iQ),nbus)
for k=1:length(iQ)
   for j=1:nbus
      i=bus(iQ(k))
      if i==j then
         sum=0;
         for tp=1:nbus
         sum=sum+Ve(tp)*(G(i,tp)*sin(Ae(i)-Ae(tp))+B(i,tp)*cos(Ae(i)-Ae(tp)));
         end
         dQdV(k,j)=sum-B(i,i)*Ve(i);
      else
         dQdV(k,j)=Ve(i)*(G(i,j)*sin(Ae(i)-Ae(j))-B(i,j)*cos(Ae(i)-Ae(j)));
      end
   end
end
end

//-------------------------------------------------------------
if Pij==[] then
dPijdV=[]
else
dPijdV=zeros(length(iPij),nbus)
for k=1:length(iPij)
      i=fb(iPij(k))
      j=tb(iPij(k))
      dPijdV(k,i)=-Ve(j)*(gl(iPij(k))*cos(Ae(i)-Ae(j))+bl(iPij(k))*sin(Ae(i)-
Ae(j)))/a(iPij(k))...
      +2*Ve(i)*gl(iPij(k));
      dPijdV(k,j)=-Ve(i)*(gl(iPij(k))*cos(Ae(i)-Ae(j))+bl(iPij(k))*sin(Ae(i)-
Ae(j)))/a(iPij(k));
end
end
//-------------------------------------------------------------
if Pji==[] then
   dPjidV=[]
else
dPjidV=zeros(length(iPji),nbus)
for k=1:length(iPji)
      j=fb(iPji(k))
      i=tb(iPji(k))
      dPjidV(k,i)=-Ve(j)*(gl(iPji(k))*cos(Ae(i)-Ae(j))+bl(iPji(k))*sin(Ae(i)-
Ae(j)))/a(iPji(k))...
      +2*Ve(i)*gl(iPji(k))/a(iPji(k));
      dPjidV(k,j)=-Ve(i)*(gl(iPji(k))*cos(Ae(i)-Ae(j))+bl(iPji(k))*sin(Ae(i)-
Ae(j)))/a(iPji(k));
end
end
```

```
//---------------------------------------------------------------

if Qij==[] then
dQijdV=[]
else
dQijdV=zeros(length(iQij),nbus)
for k=1:length(iQij)
      i=fb(iQij(k))
      j=tb(iQij(k))
      dQijdV(k,i)=-Ve(j)*(gl(iQij(k))*sin(Ae(i)-Ae(j))-bl(iQij(k))*cos(Ae(i)-
Ae(j)))/a(iQij(k))...
      -2*Ve(i)*(bl(iQij(k))+bsh(iQij(k)));
      dQijdV(k,j)=-Ve(i)*(gl(iQij(k))*sin(Ae(i)-Ae(j))-bl(iQij(k))*cos(Ae(i)-
Ae(j)))/a(iQij(k));
end
end
//---------------------------------------------------------------

if Qji==[] then
dQjidV=[]
else
dQjidV=zeros(length(iQji),nbus)
for k=1:length(iQji)
      j=fb(iQji(k))
      i=tb(iQji(k))
      dQjidV(k,i)=-Ve(j)*(gl(iQji(k))*sin(Ae(i)-Ae(j))-bl(iQji(k))*cos(Ae(i)-
Ae(j)))/a(iQji(k))...
      -2*Ve(i)*(bl(iQji(k))/a(iQji(k))^2+bsh(iQji(k)));
      dQjidV(k,j)=-Ve(i)*(gl(iQji(k))*sin(Ae(i)-Ae(j))-bl(iQji(k))*cos(Ae(i)-
Ae(j)))/a(iQji(k));
end
end
//---------------------------------------------------------------

if Pij==[] then
dPijdA=[]
else
dPijdA=zeros(length(iPij),nbus)
for k=1:length(iPij)
   i=fb(iPij(k))
   j=tb(iPij(k))
   dPijdA(k,i)=Ve(i)*Ve(j)*(gl(iPij(k))*sin(Ae(i)-Ae(j))-bl(iPij(k))*cos(Ae(i)-
Ae(j)))/a(iPij(k));
   dPijdA(k,j)=-Ve(i)*Ve(j)*(gl(iPij(k))*sin(Ae(i)-Ae(j))-bl(iPij(k))*cos(Ae(i)-
Ae(j)))/a(iPij(k));
end
dPijdA(:,swing)=[];
```

```
end

if Pji==[] then
dPjidA=[]
else
dPjidA=zeros(length(iPji),nbus)
for k=1:length(iPji)
   j=fb(iPji(k))
   i=tb(iPji(k))
   dPjidA(k,i)=Ve(i)*Ve(j)*(gl(iPji(k))*sin(Ae(i)-Ae(j))-bl(iPji(k))*cos(Ae(i)-
Ae(j)))/a(iPji(k));
   dPjidA(k,j)=-Ve(i)*Ve(j)*(gl(iPji(k))*sin(Ae(i)-Ae(j))-bl(iPji(k))*cos(Ae(i)-
Ae(j)))/a(iPji(k));
end
dPjidA(:,swing)=[];
end

if Qij==[] then
dQijdA=[]
else
dQijdA=zeros(length(iQij),nbus)
for k=1:length(iQij)
   i=fb(iQij(k))
   j=tb(iQij(k))
   dQijdA(k,i)=-Ve(i)*Ve(j)*(gl(iQij(k))*cos(Ae(i)-Ae(j))+bl(iQij(k))*sin(Ae(i)-
Ae(j)))/a(iQij(k));
   dQijdA(k,j)=Ve(i)*Ve(j)*(gl(iQij(k))*cos(Ae(i)-Ae(j))+bl(iQij(k))*sin(Ae(i)-
Ae(j)))/a(iQij(k));
end
dQijdA(:,swing)=[];
end

if Qji==[] then
dQjidA=[]
else
dQjidA=zeros(length(iQji),nbus)
for k=1:length(iQji)
   j=fb(iQji(k))
   i=tb(iQji(k))
   dQjidA(k,i)=-Ve(i)*Ve(j)*(gl(iQji(k))*cos(Ae(i)-Ae(j))+bl(iQji(k))*sin(Ae(i)-
Ae(j)))/a(iQji(k));
   dQjidA(k,j)=Ve(i)*Ve(j)*(gl(iQji(k))*cos(Ae(i)-Ae(j))+bl(iQji(k))*sin(Ae(i)-
Ae(j)))/a(iQji(k));
end
dQjidA(:,swing)=[];
end
//-------------------------------------------------------------------------------------
```

```
// Matriz H
H = [dPdA dPdV; dPijdA dPijdV; dPjidA dPjidV; dQdA dQdV; dQijdA dQijdV; dQjidA
dQjidV; dVdA dVdV ];
//Matriz G
Ge=H'*inv(R)*H;

//Analisis de observabilidad
//---------------------------------------------------

n=size(H,2)
m=rank(H)

if n==m then

// Formación del vector h(x)
//------------------------------------------------------------
//Vector V=h(x)
Vo=[];
for k=1:length(iV)
i=bus(iV(k));
Vo(k)=Ve(i)
end
//---------------------------------------------------------------------
// Vector  P=h(x)
if P==[] then
Pe=[];
else
Pe=[];
for k=1:length(iP)
i=bus(iP(k));
sum=0;
for j=1:nbus
   sum=sum+Ve(j)*(G(i,j)*cos(Ae(i)-Ae(j))+B(i,j)*sin(Ae(i)-Ae(j)))
end
Pe(k)=Ve(i)*sum;
end
end
//---------------------------------------------------------------------
// Vector  Q=h(x)
if Q==[] then
Qe=[];
else
Qe=[];
for k=1:length(iQ)
i=bus(iQ(k));
sum=0;
for j=1:nbus
```

```
    sum=sum+Ve(j)*(G(i,j)*sin(Ae(i)-Ae(j))-B(i,j)*cos(Ae(i)-Ae(j)))
end
Qe(k)=Ve(i)*sum;
end
end
//-----------------------------------------------------------------
//Vector Pij=h(x)
if Pij==[] then
Pije=[]
else
for k=1:length(iPij)
    i=fb(iPij(k))
    j=tb(iPij(k))
    Pije(k)=Ve(i)^2*gl(iPij(k))-Ve(i)*Ve(j)*(gl(iPij(k))*cos(Ae(i)-Ae(j))...
    +bl(iPij(k))*sin(Ae(i)-Ae(j)))/a(iPij(k));
end
end
//-----------------------------------------------------------------
//Vector Pji=h(x)
if Pji==[] then
Pjie=[]
else
for k=1:length(iPji)
    j=fb(iPji(k))
    i=tb(iPji(k))
    Pjie(k)=Ve(i)^2*gl(iPji(k))-Ve(i)*Ve(j)*(gl(iPji(k))*cos(Ae(i)-Ae(j))...
    +bl(iPji(k))*sin(Ae(i)-Ae(j)))/a(iPji(k));
end
end
//-----------------------------------------------------------------
//Vector Qij=h(x)
if Qij==[] then
Qije=[]
else
for k=1:length(iQij)
    i=fb(iQij(k))
    j=tb(iQij(k))
    Qije(k)=-Ve(i)^2*(bl(iQij(k))+bsh(iQij(k)))...
    -Ve(i)*Ve(j)*(gl(iQij(k))*sin(Ae(i)-Ae(j))-bl(iQij(k))*cos(Ae(i)-Ae(j)))/a(iQij(k));
end
end
//-----------------------------------------------------------------
//Vector Qji=h(x)
if Qji==[] then
Qjie=[]
else
for k=1:length(iQji)
```

```
    j=fb(iQji(k))
    i=tb(iQji(k))
    Qjie(k)=-Ve(i)^2*(bl(iQji(k))/a(iQji(k))^2+bsh(iQji(k)))...
    -Ve(i)*Ve(j)*(gl(iQji(k))*sin(Ae(i)-Ae(j))-bl(iQji(k))*cos(Ae(i)-Ae(j)))/a(iQji(k));
end
end
//-------------------------------------------------------------------
//Vector Ze(x)

Ze=[Pe;Pije;Pjie;Qe;Qije; Qjie; Vo];

Anewton=Ge;
bnewton=H'*inv(R)*(Z-Ze);
deltax=Anewton\bnewton;
if max(abs(deltax))<error then
break
end

deltaA=deltax(1:nbus-1);
deltaV=deltax(nbus:2*nbus-1);

Ve=Ve+deltaV;
index=1:nbus;
sigma=~(index==swing)
Ae(sigma)=Ae(sigma)+deltaA

iter=iter+1;
//--------------------------fin
else
    disp('La red es no observable')
    break
end
end
```