Power Systems Laboratory (EE3P006)

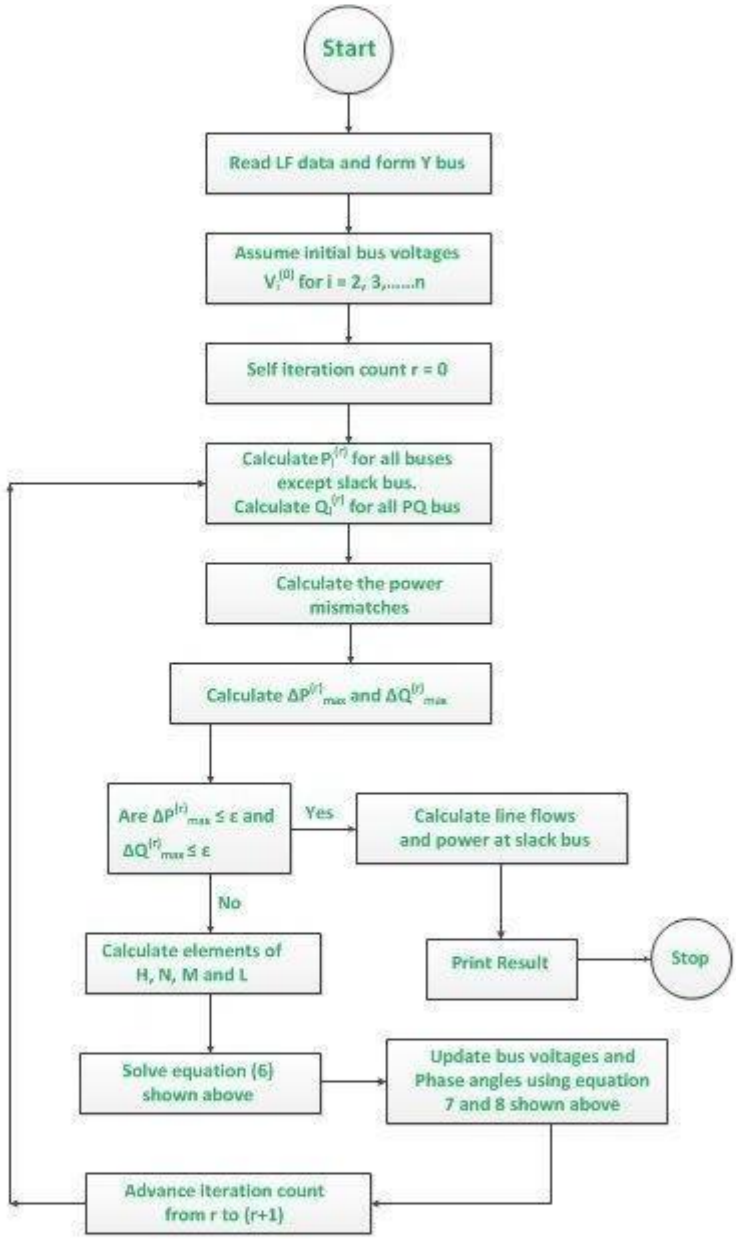
EXPERIMENT-4

LOAD FLOW STUDIES USING NEWTON RAPHSON METHOD

Shorya Sharma
19EE01017

Aim of the Experiment:

Steady State analysis of given Power System using Newton Raphson method ofLoad Flow study. Its Algorithm is given below.



CODE

```
%code for newton rapson

clc
clear
close all

Base_MVA = 100;
% Nbus = 3;

%    From   To        line_impedance            B/2
L = [ 1,     2,        0.0200+1i*0.080,        1i*0.02;
      2,     3,        0.0200+1i*0.080,        1i*0.02;
      1,     3,        0.0200+1i*0.080,        1i*0.02];

% 1 = slackbus; 2 = PQ bus; 3 = PV bus
%    Bus   Type    Vt.Mag   Vt.Ph     Pi       Qi
B = [1,     1,     1.04,      0,        0,      0;
     2,     2,        1,      0,     0.50,      1;
     3,     3,     1.04,      0,     -1.5,   -0.6];
Qmin = 0;
```

```
Qmax = 1.5;

ybus = yBusFunction(L);
theta = angle(ybus);
Y_mag = abs(ybus);

disp("Y bus : ");
disp(ybus);
disp(" ");
disp("Initial Bus Data: ");
disp("    Bus no.  Bus Type Voltage_Mag   Phase     Pi        Qi");
disp(B);

Nbus = length(B(:,2));
Npv = sum(B(:,2)==3);
Npq = Nbus-Npv-1;

Pspe = zeros(1,Npq+Npv);
Qspe = zeros(1,Npq);
del0 = zeros(1,Npq+Npv);
V0 = zeros(1,Npq);
for i = 1:Npq
    Pspe(i) = B(i+1,5);
    Qspe(i) = B(i+1,6);
    del0(i) = B(i+1,4);
    V0(i) = B(i+1,3);
end
for i = (Npq+1):(Npq+Npv)
        Pspe(i) = B(i+1,5);
        del0(i) = B(i+1,4);
end
%adding q matrix below p matrix
PQspe = cat(1,Pspe',Qspe');
delVold = cat(1,del0',V0');

err = 1;
iter = 0;


while err > 0.01
iter = iter + 1;
Vold = B(:,3);

V_mag = B(:,3);
del = B(:,4);

Pcal = zeros(1,Npq+Npv);
Qcal = zeros(1,Npq);
for i = 1:Npq
    for k = 1:Nbus
        Pcal(i) = Pcal(i) + V_mag(i+1)*V_mag(k)*Y_mag(i+1,k)*cos(theta(i+1,k)+del(k)-del(i+1));
        Qcal(i) = Qcal(i) - V_mag(i+1)*V_mag(k)*Y_mag(i+1,k)*sin(theta(i+1,k)+del(k)-del(i+1));
    end
end
for i = (Npq+1):(Npq+Npv)
    for k = 1:Nbus
        Pcal(i) = Pcal(i) + V_mag(i+1)*V_mag(k)*Y_mag(i+1,k)*cos(theta(i+1,k)+del(k)-del(i+1));
    end
end

PQcal = cat(1,Pcal',Qcal');
deltaPQ = PQspe - PQcal;

% Jacobian matrix calculation
J = JacobianFunction(V_mag,Y_mag,theta,del,Nbus,Npq,Npv);
% deltaDelV = inv(J)*deltaPQ;
deltaDelV = J\deltaPQ;


delVnew = delVold + deltaDelV;
delnew = delVnew(1:(Npq+Npv),1);
Vnew = delVnew((Npq+Npv+1):Nbus,1);

for i = 1:Npq
    B(i+1,4) = delnew(i);
    B(i+1,3) = Vnew(i);
end
for i = (Npq+1):(Npq+Npv)
    B(i+1,4) = delnew(i);
end
delVold = delVnew;
err = max(abs(deltaPQ));

end
```

```
disp(" ");
disp("Updated Bus Data: ");
disp("    Bus no.  Bus Type Voltage_Mag   Phase     Pi        Qi");
disp(B);
disp(" ");
disp("Error : "+err);
disp("No. of Itteration : "+iter);

%
V_mag = B(:,3);
del = B(:,4);




% computation of slack bus power
PS = zeros(size(ybus));
QS = zeros(size(ybus));

for i = 1:Nbus
    for k = 1:Nbus
        PS(i,k) = PS(i,k) + (Y_mag(i,k)*V_mag(i)*V_mag(k)*cos(theta(i,k)-del(i)+del(k)));
        QS(i,k) = QS(i,k) - (Y_mag(i,k)*V_mag(i)*V_mag(k)*sin(theta(i,k)-del(i)+del(k)));
    end
end
Ping = Base_MVA*sum(PS,2);
Qing = Base_MVA*sum(QS,2);

disp(" ");
disp("Slack bus power:");
disp("Real power = "+Ping(1));
disp("Reactive power = "+Qing(1));

% Power Flows
P = zeros(Nbus);
Q = zeros(Nbus);
for i = 1:Nbus
    for k = 1:Nbus
        if i ~= k
            P(i,k) = -(V_mag(i)^2*Y_mag(i,k)*cos(theta(i,k))) + (Y_mag(i,k)*V_mag(i)*V_mag(k)*cos(theta(i,k)-
del(i)+del(k)));
            Q(i,k) =  (V_mag(i)^2*Y_mag(i,k)*sin(theta(i,k))) - (Y_mag(i,k)*V_mag(i)*V_mag(k)*sin(theta(i,k)-
del(i)+del(k)));
        end
    end
end

disp(" ");
disp("Power flow matrix : ");
disp("Real power (P_ij = power flow from bus i to bus j) = ");
disp(Base_MVA*P);
disp(" ");
disp("Reactive power (Q_ij = power flow from bus i to bus j) = ");
disp(Base_MVA*Q);

% Computaion of losses
P_loss = zeros(Nbus);
Q_loss = zeros(Nbus);
for i = 1:Nbus
    for k = i:Nbus
        P_loss(i,k) = P(i,k) + P(k,i);
        Q_loss(i,k) = Q(i,k) + Q(k,i);
    end
end

disp(" ");
disp("Line loss is each line = ");
disp("Real power: ");
disp(Base_MVA*sum(P_loss,2));
disp("Reactive power: ");
disp(Base_MVA*sum(Q_loss,2));

Ploss = sum(Base_MVA*sum(P_loss));
Qloss = sum(Base_MVA*sum(Q_loss));

disp(" ");
disp("Total Real power loss : "+Ploss);
disp("Total Reactive power loss : "+Qloss);


%% Functions

% Function to calculate Jacobian matrix
function [J,J1,J2,J3,J4] = JacobianFunction(V_mag,Y_mag,theta,del,Nbus,Npq,Npv)
```

```
J1 = zeros(Npq+Npv,Npq+Npv);
J2 = zeros(Npq+Npv,Npq);
J3 = zeros(Npq,Npq+Npv);
J4 = zeros(Npq,Npq);

ik = size(J1);
for i = 1:ik(1)
    for k = 1:ik(2)
        if i == k
            for l = 1:Nbus
                if l~=i+1
                    J1(i,i) = J1(i,i) + V_mag(i+1)*V_mag(l)*Y_mag(i+1,l)*sin(theta(i+1,l)+del(l)-del(i+1));
                end
            end
        else
            J1(i,k) = -1*V_mag(i+1)*V_mag(k+1)*Y_mag(i+1,k+1)*sin(theta(i+1,k+1)+del(k+1)-del(i+1));
        end
    end
end

ik = size(J2);
for i = 1:ik(1)
    for k = 1:ik(2)
        if i == k
            J2(i,k) = 2*V_mag(i+1)*Y_mag(i+1,i+1)*cos(theta(i+1,i+1));
            for l = 1:Nbus
                if l~=i+1
                    J2(i,i) = J2(i,i) + V_mag(l)*Y_mag(i+1,l)*cos(theta(i+1,l)+del(l)-del(i+1));
                end
            end
        else
            J2(i,k) = V_mag(i+1)*Y_mag(i+1,k+1)*cos(theta(i+1,k+1)+del(k+1)-del(i+1));
        end
    end
end

ik = size(J3);
for i = 1:ik(1)
    for k = 1:ik(2)
        if i == k
            for l = 1:Nbus
                if l~=i+1
                    J3(i,i) = J3(i,i) + V_mag(i+1)*V_mag(l)*Y_mag(i+1,l)*cos(theta(i+1,l)+del(l)-del(i+1));
                end
            end
        else
            J3(i,k) = -1*V_mag(i+1)*V_mag(k+1)*Y_mag(i+1,k+1)*cos(theta(i+1,k+1)+del(k+1)-del(i+1));
        end
    end
end

ik = size(J4);
for i = 1:ik(1)
    for k = 1:ik(2)
        if i == k
            J4(i,k) = -2*V_mag(i+1)*Y_mag(i+1,i+1)*sin(theta(i+1,i+1));
            for l = 1:Nbus
                if l~=i+1
                    J4(i,i) = J4(i,i) - V_mag(l)*Y_mag(i+1,l)*sin(theta(i+1,l)+del(l)-del(i+1));
                end
            end
        else
            J4(i,k) = -1*V_mag(i+1)*Y_mag(i+1,k+1)*sin(theta(i+1,k+1)+del(k+1)-del(i+1));
        end
    end
end

J = cat(1,cat(2,J1,J2),cat(2,J3,J4));
end

% ybus function
function op = yBusFunction(data)

fb = data(:,1);
tb = data(:,2);
n = max(max(fb),max(tb));
ybus = zeros(n);

yAdm = 1./data(:,3);
sAdm = data(:,4);

for i = 1:length(fb)
    ybus (fb(i),tb(i)) = -yAdm(i);
    ybus (tb(i),fb(i)) = -yAdm(i);
end
```

```
for i = 1:n
    for j = 1:length(fb)
        if (fb(j) == i || tb(j) == i)
            ybus (i,i) = ybus(i,i) + yAdm(j) + sAdm(j)./2 ;
        end
    end
end
op = ybus;

end
```

MATLAB OUTPUT

```
Y bus :
   5.8824 -23.5094i  -2.9412 +11.7647i  -2.9412 +11.7647i
  -2.9412 +11.7647i   5.8824 -23.5094i  -2.9412 +11.7647i
  -2.9412 +11.7647i  -2.9412 +11.7647i   5.8824 -23.5094i


Initial Bus Data:
    Bus no.  Bus Type Voltage_Mag   Phase     Pi        Qi
    1.0000    1.0000    1.0400         0         0         0
    2.0000    2.0000    1.0000         0    0.5000    1.0000
    3.0000    3.0000    1.0400         0   -1.5000   -0.6000


Updated Bus Data:
    Bus no.  Bus Type Voltage_Mag   Phase     Pi        Qi
    1.0000    1.0000    1.0400         0         0         0
    2.0000    2.0000    1.0819   -0.0241    0.5000    1.0000
    3.0000    3.0000    1.0400   -0.0655   -1.5000   -0.6000


Error : 0.0013362
No. of Itteration : 3

Slack bus power:
Real power = 103.1563
Reactive power = -79.0656

Power flow matrix :
Real power (P_ij = power flow from bus i to bus j) =
         0   19.1578   83.9985
  -18.4506         0   68.4506
  -82.6332  -67.3668         0


Reactive power (Q_ij = power flow from bus i to bus j) =
         0  -58.8039  -18.0985
   61.6330         0   40.7079
   23.5595  -36.3726         0


Line loss is each line =
Real power:
    2.0725
    1.0838
         0

Reactive power:
    8.2901
    4.3352
         0
```

```
Total Real power loss : 3.1563
Total Reactive power loss : 12.6253
```

DISCUSSION

1. Determine values of P_{calc} and Q_{calc} flowing into the system at every bus for the specified or estimated values of voltage magnitudes and angles for the first iteration or the most recently determined voltages for subsequent iterations.
2. Calculate $\triangle P$ at every bus.
3. Calculate values for the jacobian using estimated or specified values of voltage magnitude and angle in the equations for partial derivatives.
4. Invert the Jacobians and calculate the voltage corrections $\triangle\delta_k$ and $\triangle|V_k|$ on every bus.
5. Calculate new values of δ_k and $|V_k|$ by adding $\triangle\delta_k$ and $\triangle|V_k|$ to previous values.
6. Return to step 1 and repeat the process using the most recently determined values of voltage magnitudes and angles until either all values of $\triangle P$ and $\triangle Q$ or all values of $\triangle\delta$ and $\triangle|V|$ are less than a chosen precision index.

CONCLUSION

- Newton-Raphson method is an iterative method for solving nonlinear problems. It starts with an initial guess and then makes use of the Taylor series expansion and the approximation for the solution by the first order gradient.

- We observe that the number of iterations required by the Newton-Raphson method using bus admittances is practically independent of the number of buses. The time for the Gauss-Seidel method increases almost directly with the number of buses.

- Computing elements of the jacobian is time consuming therefore time per iteration is comparatively larger for this method. The advantage of shorter computation time for a solution of the same accuracy is in favor of Newton-Raphson method for large systems.