Exp - 09

# Load Flow Solution Using Fast De-Coupled

Aim:

To determine the load flow solution of the given problem with Fast De-coupled method using MATLAB coding.

Apparatus Required:

| SI.No | Apparatus | Specification |
|---|---|---|
| 1 | PC | Dual core, RAM 512 MB 1.2 GHz speed, 80 GB |
| 2 | MATLAB | 7.5 |

Theory:

Due to the weak coupling between PV and q-δ half of the elements of jacobian matrix are neglected further the assumptions made are

$\cos \delta ij = 1$

$\sin \delta ij = 0$

$Qi << Bij |V|^2$

The simplified FDLF equations are

$$(\Delta P / |V|) = [B'][\Delta \delta]$$

$$(\Delta Q / |V|) = [B''][\Delta |V|]$$

One iteration implies one solution for [Δδ] to update [δ] and one solution for [ΔP/|V|] to update [|V|] and is termed as 1-δ and 1-V iteration. The convergence for the real and reactive power is achieved when $\max[\Delta P] < \sum P$; $\max[\Delta Q] <= \sum Q$.

The main advantage of the decoupled load flow as compared to Newton Raphson method is its reduced memory is storing Jacobian.

Algorithm:

Step1:Formulate Y bus matrix, then compute the bus susceptance matrices B'&B''

Step2: Assume flat start for status V solution

$\delta_1^0 = 0$ for i=1,2,…..N (all buses except slack buses)

$|V_i| = 1 - 0$ for i = M+1…….N (for all PQ buses)

$|V_i| = |V_i|_{spec}$ for all PV buses & slack bus

Step3: For load buses calculate $P_{i\,cal}$ & $Q_{i\,cal}$ using

$$P_{i\,cal} = \sum_{j=1}^{N} |V_i||Y_{ij}||V_j| \cos(Q_{ij} + \delta_j - \delta_i)$$

$$Q_{i\,cal} = \sum_{i=1}^{N} |V_i||Y_{ij}||V_j| \sin(Q_{ij} + \delta_j - \delta_i)$$

Step4: For PV buses check for Q limit violation if Qimin < $Q_i$< $Q_{i\,max}$ calculate $P_{i\,cal}$. $Q_{i\,cal} > Q_{i\,max,}$, $Q_{i\,spec}=Q_{i\,max}$ then PV bus will act as PQ bus

Step5: Calculate mismatch vector using

$\Delta P_i = P_{i(spec)} - P_{i\,cal}$

$\Delta Q_i = Q_{i(spec)} - Q_{i\,cal}$

Step6: compute

$$\Delta P_{i(\max)} = \max|\Delta P_i|, i = 1,2,.....N \text{ \& slack bus}$$

$$\Delta Q_{i(\max)} = \max|\Delta Q_i|, M+1.....N$$

Step7: calculate

$$[\Delta\delta_i] = [-B]^{-1} \frac{\Delta P_i}{|\Delta V_i|}$$

$$[\Delta V_i] = [-B^{"}]^{-1} \frac{\Delta Q_i}{|\Delta V_i|}$$

Step8: This procedure is continued until

$|\Delta P_i| \langle \varepsilon + Q_i \langle \varepsilon$ otherwise go to step 3)

Program:

```
clear all;
clc;
n=input('enter the number of buses:');
for i=1:n
   for j=1:n
      y(i,j)=input('Enter the admittance value:');
   end
end
yb(n,n)=0;
for i=1:n
   for j=1:n
      if i==j
         for k=1:n
            yb(i,j)=yb(i,j)+y(i,k);
         end
      else
         yb(i,j)=-y(i,j);
      end
   end
end
th(n)=0;
for i=1:n
   Bus=i
   mag(i)=input('Enter the voltage value:');
   acp(i)=input('Enter real power value:');
   acq(i)=input('Enter reactive power');
end
j=1;
for i=1:n
   if(mag(i)==1)
      k(j)=i;
      j=j+1;
   end
end
my=abs(yb);an=angle(yb);g=real(yb);b=imag(yb);
yb
mag
th
acp
acq
Chmag(n)=0;Chth(n)=0;Pp(n)=0;Qq(n)=0;
for i=1:n
   for j=1:n
      Pp(i)=Pp(i)+mag(i)*my(i,j)*mag(j)*cos(an(i,j)-th(i)+th(j));
       Qq(i)=Qq(i)-mag(i)*my(i,j)*mag(j)*sin(an(i,j)-th(i)+th(j));
   end
   Pp
   Qq
end
for i=2:n
   for j=2:n
      if i~=j
         j1(i,j)=mag(i)*mag(j)*(g(i,j)*sin(th(i)-th(j))-b(i,j)*cos(th(i)-th(j)));
```

```
            j3(i,j)=0;
            j2(i,j)=0;
            j4(i,j)=j1(i,j);
        else
            j1(i,j)=-Qq(i)-b(i,j)*(mag(i)^2);
            j2(i,j)=0;
            j3(i,j)=0;
            j4(i,j)=Qq(i)-b(i,j)*(mag(i)^2);
        end
    end
end
ja1(1:n-1,1:n-1)=j1(2:n,2:n);
ja2(1:n-1,1:n-1)=j2(2:n,2:n);
ja3(1:n-1,1:n-1)=j3(2:n,2:n);
ja4(1:n-1,1:n-1)=j4(2:n,2:n);
jacob=[ja1 ja2;ja3 ja4]
bb(1:n-1, 1:n-1)=b(2:n,2:n)
for i=2:n
    j=1;
    if(i==k(j))
        bbb(j,:)=b(i,i);
        j=j+1;
    end
end
bbb
delp(2:n)=acp(2:n)-Pp(2:n)
delq(2:n)=acq(2:n)-Qq(2:n)
for i=2:n
    x(i)=delp(i)/abs(mag(i));
     y(i)=delq(i)/abs(mag(i));
end
Chth(2:n)=-inv(bb)*x(2:n)';
for i=2:n
    j=1;
    if(i==k(j))
        Chmag(i)=-inv(bbb)*y(i)';
    end
end
mag=mag+Chmag;
th=th+Chth;
fprintf('the voltage value for buses');
mag
fprintf('the angle values for buses');
th
```

Result: