# VALIDATION REPORT

**Implementation of Standard Backward Forward Sweep(B/FS) method and Branch Current Based B/FS by MATLAB programming language.**

**Course: Electric Power Distribution Systems**
**Professor: Dr. Karimi**
**Student: Hamed Najafi**
**University of Kashan**

JANUARY 9, 2023

دانشگاه کاشان

# Results

**By performing a power flow calculation on IEEE-33 bus radial distribution system by MATPOWER7.1 we can see the same results as we had for BFS MATLAB code in initial report.**

**MATPOWER Output:**

MATPOWER Version 7.1, 08-Oct-2020 -- AC Power Flow (Newton)

Newton's method power flow (power balance, polar) converged in 4 iterations.

**Converged in 0.39 seconds**

```
================================================================
|    System Summary                                           |
================================================================

How many?            How much?           P (MW)         Q (MVAr)
---------------------   -------------------  -------------  -----------------
Buses            33     Total Gen Capacity      10.0        -10.0 to 10.0
Generators        1     On-line Capacity        10.0        -10.0 to 10.0
Committed Gens    1     Generation (actual)      3.9             2.4
Loads            32     Load                     3.7             2.3
  Fixed          32       Fixed                  3.7             2.3
  Dispatchable    0       Dispatchable        -0.0 of -0.0      -0.0
Shunts            0     Shunt (inj)             -0.0             0.0
Branches         37     Losses (I^2 * Z)         0.20            0.14
Transformers      0     Branch Charging (inj)     -              0.0
Inter-ties        0     Total Inter-tie Flow     0.0             0.0
Areas             1

                     Minimum                      Maximum
            ------------------------  --------------------------------
Voltage Magnitude   0.913 p.u. @ bus 18        1.000 p.u. @ bus 1
Voltage Angle      -0.50 deg   @ bus 18         0.50 deg   @ bus 30
P Losses (I^2*R)             -                0.05 MW    @ line 2-3
Q Losses (I^2*X)             -                0.03 MVAr  @ line 5-6

================================================================
```

| Bus Data |

| Bus # | Voltage Mag(pu) | Voltage Ang(deg) | Generation P (MW) | Generation Q (MVAr) | Load P (MW) | Load Q (MVAr) |
|---|---|---|---|---|---|---|
| 1 | 1.000 | 0.000* | 3.92 | 2.44 | - | - |
| 2 | 0.997 | 0.014 | - | - | 0.10 | 0.06 |
| 3 | 0.983 | 0.096 | - | - | 0.09 | 0.04 |
| 4 | 0.975 | 0.162 | - | - | 0.12 | 0.08 |
| 5 | 0.968 | 0.228 | - | - | 0.06 | 0.03 |
| 6 | 0.950 | 0.134 | - | - | 0.06 | 0.02 |
| 7 | 0.946 | -0.096 | - | - | 0.20 | 0.10 |
| 8 | 0.941 | -0.060 | - | - | 0.20 | 0.10 |
| 9 | 0.935 | -0.133 | - | - | 0.06 | 0.02 |
| 10 | 0.929 | -0.196 | - | - | 0.06 | 0.02 |
| 11 | 0.928 | -0.189 | - | - | 0.04 | 0.03 |
| 12 | 0.927 | -0.177 | - | - | 0.06 | 0.04 |
| 13 | 0.921 | -0.269 | - | - | 0.06 | 0.04 |
| 14 | 0.919 | -0.347 | - | - | 0.12 | 0.08 |
| 15 | 0.917 | -0.385 | - | - | 0.06 | 0.01 |
| 16 | 0.916 | -0.408 | - | - | 0.06 | 0.02 |
| 17 | 0.914 | -0.485 | - | - | 0.06 | 0.02 |
| 18 | 0.913 | -0.495 | - | - | 0.09 | 0.04 |
| 19 | 0.997 | 0.004 | - | - | 0.09 | 0.04 |
| 20 | 0.993 | -0.063 | - | - | 0.09 | 0.04 |
| 21 | 0.992 | -0.083 | - | - | 0.09 | 0.04 |
| 22 | 0.992 | -0.103 | - | - | 0.09 | 0.04 |
| 23 | 0.979 | 0.065 | - | - | 0.09 | 0.05 |
| 24 | 0.973 | -0.024 | - | - | 0.42 | 0.20 |
| 25 | 0.969 | -0.067 | - | - | 0.42 | 0.20 |
| 26 | 0.948 | 0.173 | - | - | 0.06 | 0.03 |
| 27 | 0.945 | 0.229 | - | - | 0.06 | 0.03 |
| 28 | 0.934 | 0.312 | - | - | 0.06 | 0.02 |
| 29 | 0.926 | 0.390 | - | - | 0.12 | 0.07 |
| 30 | 0.922 | 0.496 | - | - | 0.20 | 0.60 |
| 31 | 0.918 | 0.411 | - | - | 0.15 | 0.07 |
| 32 | 0.917 | 0.388 | - | - | 0.21 | 0.10 |
| 33 | 0.917 | 0.380 | - | - | 0.06 | 0.04 |

| Total: 0.203(MW) | 0.14(MVAr) |
|---|---|

**BFS code Output (rounded to 3 decimal places):**

| Bus NO | \|U\|(pu) | θ(°) |
|---|---|---|
| 1 | 1 | 0 |
| 2 | 0.997 | 0.014 |
| 3 | 0.983 | 0.096 |
| 4 | 0.975 | 0.162 |
| 5 | 0.968 | 0.228 |
| 6 | 0.95 | 0.134 |
| 7 | 0.946 | -0.096 |
| 8 | 0.941 | -0.06 |
| 9 | 0.935 | -0.133 |
| 10 | 0.929 | -0.196 |
| 11 | 0.928 | -0.189 |
| 12 | 0.927 | -0.177 |
| 13 | 0.921 | -0.268 |
| 14 | 0.919 | -0.347 |
| 15 | 0.917 | -0.385 |
| 16 | 0.916 | -0.408 |
| 17 | 0.914 | -0.485 |
| 18 | 0.913 | -0.495 |
| 19 | 0.997 | 0.004 |
| 20 | 0.993 | -0.063 |
| 21 | 0.992 | -0.083 |
| 22 | 0.992 | -0.103 |
| 23 | 0.979 | 0.065 |
| 24 | 0.973 | -0.024 |
| 25 | 0.969 | -0.067 |
| 26 | 0.948 | 0.173 |
| 27 | 0.945 | 0.229 |
| 28 | 0.934 | 0.312 |
| 29 | 0.926 | 0.39 |
| 30 | 0.922 | 0.495 |
| 31 | 0.918 | 0.411 |
| 32 | 0.917 | 0.388 |
| 33 | 0.917 | 0.38 |

Number of Iterations=3

---

Elapsed time is 0.025467 seconds.

Total Power Loss in lines (kW) =202.5193

Total ReactivePower Consumed by Lines (kVAr) =135.0338

**So the written BFS code is working correctly.**

**By the same way I found a minor mistake in my previous code that is mentioned in the next section ("Correction"). After correcting two lines we can see that our BCBBFS code has no problem as well. The corrected code output is shown after MATPOWER output.**

**MATPOWER Output:**

MATPOWER Version 7.1, 08-Oct-2020 -- AC Power Flow (Newton)
Newton's method power flow (power balance, polar) converged in 4 iterations.

Converged in 0.11 seconds

```
================================================================================
|     System Summary                                                          |
================================================================================

How many?                How much?              P (MW)            Q (MVAr)
---------------------    --------------------  -------------  -----------------
Buses            33     Total Gen Capacity      30.0          -10.2 to 10.3
Generators        3     On-line Capacity        30.0          -10.2 to 10.3
Committed Gens    3     Generation (actual)      3.8                2.3
Loads            32     Load                     3.7                2.3
  Fixed          32       Fixed                  3.7                2.3
  Dispatchable    0       Dispatchable          -0.0 of -0.0       -0.0
Shunts            0     Shunt (inj)             -0.0                0.0
Branches         37     Losses (I^2 * Z)         0.13               0.09
Transformers      0     Branch Charging (inj)    -                  0.0
Inter-ties        0     Total Inter-tie Flow     0.0                0.0
Areas             1

                         Minimum                      Maximum
                ------------------------  -------------------------------
Voltage Magnitude   0.935 p.u. @ bus 32       1.000 p.u. @ bus 1
Voltage Angle      -0.23 deg   @ bus 16        0.57 deg   @ bus 30
P Losses (I^2*R)             -                0.04 MW    @ line 2-3
Q Losses (I^2*X)             -                0.02 MVAr  @ line 5-6

|     Bus Data                                                               |
================================================================================
```

| Bus # | Voltage Mag(pu) | Voltage Ang(deg) | Generation P (MW) | Generation Q (MVAr) | Load P (MW) | Load Q (MVAr) |
|---|---|---|---|---|---|---|
| 1 | 1.000 | 0.000* | 3.34 | 2.10 | - | - |
| 2 | 0.997 | 0.013 | - | - | 0.10 | 0.06 |
| 3 | 0.985 | 0.088 | - | - | 0.09 | 0.04 |
| 4 | 0.979 | 0.149 | - | - | 0.12 | 0.08 |
| 5 | 0.974 | 0.210 | - | - | 0.06 | 0.03 |
| 6 | 0.959 | 0.161 | - | - | 0.06 | 0.02 |
| 7 | 0.957 | -0.013 | - | - | 0.20 | 0.10 |
| 8 | 0.953 | 0.003 | - | - | 0.20 | 0.10 |
| 9 | 0.950 | -0.056 | - | - | 0.06 | 0.02 |
| 10 | 0.947 | -0.106 | - | - | 0.06 | 0.02 |
| 11 | 0.946 | -0.104 | - | - | 0.04 | 0.03 |
| 12 | 0.945 | -0.103 | - | - | 0.06 | 0.04 |
| 13 | 0.943 | -0.164 | - | - | 0.06 | 0.04 |
| 14 | 0.942 | -0.199 | - | - | 0.12 | 0.08 |
| 15 | 0.943 | -0.218 | - | - | 0.06 | 0.01 |
| 16 | 0.943 | -0.230 | - | - | 0.06 | 0.02 |
| 17 | 0.945 | -0.208 | - | - | 0.06 | 0.02 |
| 18 | 0.946 | -0.203 | 0.25 | 0.15 | 0.09 | 0.04 |
| 19 | 0.997 | 0.003 | - | - | 0.09 | 0.04 |
| 20 | 0.994 | -0.056 | - | - | 0.03 | 0.00 |
| 21 | 0.994 | -0.075 | - | - | 0.09 | 0.04 |
| 22 | 0.993 | -0.096 | - | - | 0.09 | 0.04 |
| 23 | 0.982 | 0.058 | - | - | 0.09 | 0.05 |
| 24 | 0.975 | -0.031 | - | - | 0.42 | 0.20 |
| 25 | 0.972 | -0.074 | - | - | 0.42 | 0.20 |
| 26 | 0.958 | 0.200 | - | - | 0.06 | 0.03 |
| 27 | 0.956 | 0.255 | - | - | 0.06 | 0.03 |
| 28 | 0.946 | 0.368 | - | - | 0.06 | 0.02 |
| 29 | 0.940 | 0.467 | - | - | 0.12 | 0.07 |
| 30 | 0.937 | 0.570 | - | - | 0.20 | 0.60 |
| 31 | 0.935 | 0.527 | - | - | 0.15 | 0.07 |
| 32 | 0.935 | 0.522 | - | - | 0.21 | 0.10 |
| 33 | 0.935 | 0.544 | 0.20 | 0.10 | 0.06 | 0.04 |

| Total: | 0.132 (MW) | 0.09 (MVAr) |
|---|---|---|

**BCBBFS code Output (rounded to 3 decimal places):**

| Bus NO | \|U\|(pu) | θ(°) |
|---|---|---|
| 1 | 1 | 0 |
| 2 | 0.997 | 0.013 |
| 3 | 0.985 | 0.088 |
| 4 | 0.979 | 0.149 |
| 5 | 0.974 | 0.21 |
| 6 | 0.959 | 0.161 |
| 7 | 0.957 | -0.013 |
| 8 | 0.953 | 0.003 |
| 9 | 0.95 | -0.056 |
| 10 | 0.947 | -0.106 |
| 11 | 0.946 | -0.104 |
| 12 | 0.945 | -0.103 |
| 13 | 0.943 | -0.164 |
| 14 | 0.942 | -0.199 |
| 15 | 0.943 | -0.218 |
| 16 | 0.943 | -0.23 |
| 17 | 0.945 | -0.208 |
| 18 | 0.946 | -0.203 |
| 19 | 0.997 | 0.003 |
| 20 | 0.994 | -0.056 |
| 21 | 0.994 | -0.075 |
| 22 | 0.993 | -0.096 |
| 23 | 0.982 | 0.058 |
| 24 | 0.975 | -0.031 |
| 25 | 0.972 | -0.074 |
| 26 | 0.958 | 0.2 |
| 27 | 0.956 | 0.255 |
| 28 | 0.946 | 0.368 |
| 29 | 0.94 | 0.467 |
| 30 | 0.937 | 0.57 |
| 31 | 0.935 | 0.527 |
| 32 | 0.935 | 0.522 |
| 33 | 0.935 | 0.544 |

Number of Iterations (outer loop) = 3

---

Elapsed time is 0.045709 seconds.

Total Power Loss in lines (kW) =131.6967
Total ReactivePower Consumed by Lines (kVAr) =86.3908

# Correction

**A correction has been made in BCBBFS MATLAB code and consequently in the results as follows:**

**In previous report in Appendix: MATLAB code for BCBB/FS:" %% Updating Q Values" we had:**

```
 1. %% Updating Q Values
 2. DV=abs(pv_bus_sp(:,2))-abs(v(pv_bus_sp(:,1)));    %Calculating DeltaV Matrix
 3. DQ=X_s\DV;                                        %Calculating DeltaV Matrix
 4. Q(pv_bus_no,1)=Q(pv_bus_no,1)-sign(DV).*DQ;      %Updating Q values (pu)
 5. for qqi=1:length(pv_bus_no)                       %Checking Q limits (pu)
 6.     qq=pv_bus_no(qqi);
 7.     if(Q(qq,1)<-abs(pv_qm_gen(qq)))
 8.         Q(qq,1)=-abs(pv_qm_gen(qq));
 9.     elseif(Q(qq,1)>abs(pv_qm_con(qq)))
10.         Q(qq,1)=abs(pv_qm_con(qq));
11.     end
12. end
```

**In Lines 8 and 10 the generation value of reactive power at bus(bus number:'qq') is set on DG reactive power generation(/consumption) limit values and it's wrong so it must be the limit value plus $Q_{Load}$ amount (Qcons._net=Qcons.+$Q_{Load}$) therefore this part changed to:**

```
 1. %% Updating Q Values
 2. DV=abs(pv_bus_sp(:,2))-abs(v(pv_bus_sp(:,1)));    %Calculating DeltaV Matrix
 3. DQ=X_s\DV;                                        %Calculating DeltaV Matrix
 4. Q(pv_bus_no,1)=Q(pv_bus_no,1)-sign(DV).*DQ;      %Updating Q values (pu)
 5. for qqi=1:length(pv_bus_no)                       %Checking Q limits (pu)
 6.     qq=pv_bus_no(qqi);
 7.     if(Q(qq,1)<QLoad(qq)-abs(pv_qm_gen(qq)))
 8.         Q(qq,1)=QLoad(qq)-abs(pv_qm_gen(qq));
 9.     elseif(Q(qq,1)>(abs(pv_qm_con(qq)+QLoad(qq))))
10.         Q(qq,1)=abs(pv_qm_con(qq))+(QLoad(qq));
11.     end
12. end
```

***Notice: In MATPOWER default data for 33-bus system (case33) in branch data for branch number 7 (7to8) there is a difference in some references that mentioned as (in "case33mg.m": in some references 0.71 and 0.23 (see case33bw.m)) and our choice was 0.7114 and 0.2351 for r and x respectively.**

# Appendix A

## Validated MATLAB Code For B/FS:

```
clc
clear
close all
%% Reading Loads & Lines Data
Load_Data = readmatrix('bus33.xls');          %Reading Load Data (P & Q in kW & kVAr)
Line_Data = readmatrix('branch33.xls');       %Reading Line Data  (R & X in Ohms)
%% Setting Initial Parameters
N=10;                          % N: Maximum Number of Iterations
Sb=1;                          %  S_base (MVA)
Ub=12.66;                  % U_base (kV)   (line2line)
e=0.01;                    % Epsilon -> Convergence criteria : max(|vnew-vold|)<e
%% Evaluating Zbase
Zb=(Ub^2)/Sb;          % Z_base
%%
br_no=length(Line_Data);            % Number of Branches
bus_no=length(Load_Data);           % Number of Buses
%% Per unit Values
R = Line_Data(:,4)./Zb;
X = Line_Data(:,5)./Zb;
P = ((Load_Data(:,2))./(1000*Sb));     % P in kw & Sb in MVA
Q = ((Load_Data(:,3))./(1000*Sb));     % Q in kVAr & Sb in MVA
%% Forming Connection Matrix (C(branches,buses))
s_buses=Line_Data(:,2);                   %Sending buses
r_buses=Line_Data(:,3);                   %Recieving buses
C=zeros(br_no,bus_no);
for branch=1:br_no
    C(branch,s_buses(branch))=-1;    %Sending bus : -1
    C(branch,r_buses(branch))=+1;    %Recieving bus : +1
end
%% Print Values
% (1:no)
% [(1:br)',C]
%% Determining End Nodes
endnode=(find(sum(C)==1))';          %End Buses indexes
%% Print Values
%endnode
%% Determining the Path of each Radius(:Routes from first Bus to each Endnode)
h=length(endnode);                                                  % h= Number of Radiuses
g=[0];                                                              % EndBuses Path to the first Bus
for route_no=1:h
    rnode=endnode(route_no);                            %Recieving node
    snode=s_buses(r_buses==rnode);                      %Sending Node
    g(route_no,1)=rnode;
    g(route_no,2)=snode;
    j=2;
    while(snode~=1)
        rnode=snode;
        g(route_no,j)=rnode;
        snode=s_buses(r_buses==rnode);
        g(route_no,j+1)=snode;
        j=j+1;
    end
end
%% Print Values
%% Sorting Radius Matrix Elements
gs=g;  %gs is sorted form of g
for i=1:length(endnode)
    rout=sort(nonzeros(g(i,:)));
    gs(i,1:length(rout))=rout;
end
%% print Values
%gs;
%% Forming Route Matrices for applications
gb=g;
mr=1;                                               %MainRoute_Row_index in g
for i=1:size(gb,1)
    if length(nonzeros(gb(i,:)))>length(nonzeros(gb(mr,:)))
        mr=i;
```

```
    end
end
temp=gb(1,:);           %Main Route placed in at the 1st row
gb(1,:)=gb(mr,:);
gb(mr,:)=temp;
for i=1:size(gb,1)
    for j=1:size(gb,2)
        n=gb(i,j);
        for ii=((i+1):size(gb,1))
            for jj=1:(size(gb,2)-1)
                if gb(ii,jj)==n
                    gb(ii,jj+1)=0;
                end
            end
        end
    end
end
gv=gb;                  %gv matrix will be used for KVL in Forward steps
for i=1:length(endnode)
    rout=sort(nonzeros(gb(i,:)));
    gv(i,1:length(rout))=rout;
end
sc=zeros(size(gb,1),1);
for j=1:size(gb,1)
    for i=1:size(gb,1)-1
        a=length(nonzeros(gb(i,:)));
        b=length(nonzeros(gb(i+1,:)));
        if a>b
            t=gb(i,:);
            gb(i,:)=gb(i+1,:);
            gb(i+1,:)=t;
        end
    end
end
g=gb;
%% initial guess
v = ones(bus_no,1);           %Bus Voltages vector Initialization (complex value) (flat initial guess)
I = zeros(br_no,1);           %Branches' Current vector Initialization %C:(br,bus) '
%% Iteration Loop
for ni=1:N
    %% Backward Step
    vold=v;
    LC = conj(complex(P,Q)./v);     %Bus Load Currents  vector
    for r=1:route_no
        for i=1:size(g,2)-1
            b=g(r,i);
            if b==0
                break;
            end
            if sum(C(:,b))==1
                I(C(:,b)==1)=LC(b);
                LC(g(r,i+1))=LC(g(r,i+1))+LC(b);
            else
                if g(r,i+1)==1
                    I(C(:,b)==1)=LC(b);
                    break;
                else
                    I(C(:,b)==1)=LC(b);
                    if g(r,i+1)~=0
                        LC(g(r,i+1))=LC(g(r,i+1))+LC(b);
                    end
                end
            end
        end
    end
    %% Forward Step
    for r=1:route_no
        for i=1:size(gv,2)-1
            if gv(r,i+1)==0
                continue;
            end
            b= find(C(:,gv(r,i+1))==1);
            v(gv(r,i+1))=v(gv(r,i))-complex(R(b),X(b))*I(b);
            %fprintf("V("+num2str(gv(r,i+1))+")=V("+num2str(gv(r,i))+")-zI("+num2str(b)+")\n");
        end
```

```
    end
    vnew=v;
    if max(abs(vnew-vold))<e
        fprintf("Algorithm Converged!\nNumber of Iterations="+num2str(ni)+"\n-------------------------\n")
        break;
    end
end
```

```
%% Print Calculated Values
vbp=[abs(v),angle(v).*(180/pi)];
vbp2=[((1:bus_no)'),abs(v),angle(v).*(180/pi)];  %'
h={'Bus NO','|U|(pu)','?(°)'};
T = array2table(vbp2,'VariableNames',h);
T2 = array2table(round(vbp2,2),'VariableNames',h);
%% Print
fprintf("Final Voltages:\n")
disp(T2)
f=figure;
t=uitable(f,'data',vbp2,'columnname',h);
%% Plot Voltage Profile
f2=figure;
p=plot(vbp2(:,1),vbp2(:,2),'-b','LineWidth',2);
hold on
plot([0 33],[0.95 0.95],'-r','LineWidth',1);
plot([0 33],[0.9 0.9],'-r','LineWidth',1);
hold off
xlim([0 33])
ylim([0.85 1.02])
yaxes=[[0.86:0.02:0.95],[0.95:0.01:1.2]];
yticks(yaxes)
xticks(0:33)
xlabel("BUS Number")
ylabel("V_{Line} pu")
grid on
%%
Ibrpu=[abs(I) angle(I)*180/pi];     % Branches' Currents in pu magnitude and angle '
%% Line Consumed Power Calculation
PL  = R.*(abs(I).^2);       % Active Power (pu) Consumed by each Line
QL = X.*(abs(I).^2);        %Reactive Power (pu) Consumed by each Line
PLkW=(PL)*Sb*1000;
QLkVAr=(QL)*Sb*1000;
PLt=sum(PL)*Sb*1000;    %Total kW consumed by lines (total power loss)
QLt=sum(QL)*Sb*1000;    %Total kVAr consumed by lines
%% Print
fprintf('-----------------------------------------------------------------------\n')
fprintf("Total Power Loss in lines (kW) ="+num2str(PLt)+'\n');
fprintf("Total ReactivePower Consumed by Lines (kVAr) ="+num2str(QLt)+'\n');
```

# Appendix B

## Validated MATLAB Code for BCBB/FS:

```
clc
clear
close all
%% Reading Loads & Lines Data
Load_Data = readmatrix('bus33.xls');            %Reading Load Data (P & Q in kW & kVAr)
Line_Data = readmatrix('branch33.xls');       %Reading Line Data  (R & X in Ohms)
DG_Data= readmatrix('DG_BUS.xlsx');
%% Setting Initial Parameters
N1=10;                          % N: Maximum Number of Iterations of Inner loop
N2=10;                          % N2: Maximum Number of Iterations
Sb=1;                            %  S_base (MVA)
Ub=12.66;                    % U_base (kV)   (line2line)
e=0.001;                         % Epsilon -> Convergence criteria : max(||vsp_pu|-|vcal_pu||)<e
e1=0.001;                      % Inner loop Convergence criteria: max(|vsp_old-vcal_new|)<e1
%% Processing const.P-const.Q DG Data in Load_Data Matrix
pq_dg=find(DG_Data(:,4)==0);
Load_Data([DG_Data(pq_dg,1)],[2 3])=Load_Data([DG_Data(pq_dg,1)],[2 3])-DG_Data(pq_dg,[2 3]);
%% Processing const.P-const.V DG Data in Load_Data Matrix and Extracting data
pv_dg=find(DG_Data(:,4)==1); % pv buses indices in DG_Data
Load_Data([DG_Data(pv_dg,1)],[2])=Load_Data([DG_Data(pv_dg,1)],[2])-DG_Data(pv_dg,2);
pv_bus_sp=DG_Data(pv_dg,[1 5]); %DG Voltage set points (pu)
pv_bus_no=DG_Data(pv_dg,1);
pv_bus_maxgen_q=DG_Data(pv_dg,[1 6]);   % Maximum Generated Q (pu)
pv_bus_maxcon_q=DG_Data(pv_dg,[1 7]);   % Maximum Consumed Q (pu)
pv_bus_maxgen_q(:,2)=pv_bus_maxgen_q(:,2)./(1000*Sb);
pv_bus_maxcon_q(:,2)=pv_bus_maxcon_q(:,2)./(1000*Sb);
pv_qm_gen=containers.Map(pv_bus_maxgen_q(:,1),pv_bus_maxgen_q(:,2));
pv_qm_con=containers.Map(pv_bus_maxcon_q(:,1),pv_bus_maxcon_q(:,2));
%% Evaluating Zbase
Zb=(Ub^2)/Sb;          % Z_base
%%
br_no=length(Line_Data);              % Number of Branches
bus_no=length(Load_Data);           % Number of Buses
%% Per unit Values
R = Line_Data(:,4)./Zb;
X = Line_Data(:,5)./Zb;
P = ((Load_Data(:,2))./(1000*Sb));      % P in kw & Sb in MVA
Q = ((Load_Data(:,3))./(1000*Sb));     % Q in kVAr & Sb in MVA
QLoad=Q;
%% Forming Connection Matrix (C(branches,buses))
s_buses=Line_Data(:,2);                   %Sending buses
r_buses=Line_Data(:,3);                    %Recieving buses
C=zeros(br_no,bus_no);
for branch=1:br_no
    C(branch,s_buses(branch))=-1;     %Sending bus : -1
    C(branch,r_buses(branch))=+1;    %Recieving bus : +1
end
%% Formin Reactance Sesetivity Matrix
s=Line_Data(:,2);       % Sending Buses
r=Line_Data(:,3);       % Receiving Buses
wx=[Line_Data(:,1),X]; % line number   line X(pu)
NG=graph(s,r,wx(:,2));
figure

gp=plot(NG,'EdgeLabel',wx(:,1),'Layout','layered','Direction','right','LineWidth',4,'EdgeColor','c',
'MarkerSize',8,'Marker','o','NodeFontSize',14,'EdgeFontSize',10);
highlight(gp,pv_bus_no,'Marker','s','MarkerSize',12)
```

```
X_s=zeros(length(pv_bus_no));
pv_path_bus=zeros(length(pv_dg),1);
pv_d_s=zeros(length(pv_dg),1);
pv_path_line=zeros(length(pv_dg),1);
for i=1:length(pv_dg)
    [a,b,c] = shortestpath(NG,1,pv_bus_no(i),'Method','unweighted');
    pv_path_bus(i,1:length(a))=a;
    pv_d_s(i)=b;
    pv_path_line(i,1:length(c))=c;
    singlepath=nonzeros(pv_path_line(i,:));
    X_s(i,i)=sum(wx(singlepath,2));
    colora=['c','y','r','b','g','m'];
    highlight(gp,a,'EdgeColor',colora(mode(i,4)+1))
end
for i=1:length(pv_bus_no)
    for j=i+1:length(pv_bus_no)
        %fprintf("i="+num2str(i)+','+'j='+num2str(j)+'\n');
        temp=pv_path_line(i,:)-pv_path_line(j,:);
        for ii=1:length(temp)
            if temp(ii)~=0
                break;
            end
        end
        ii=ii-1;
        commonpath=pv_path_line(i,1:ii);
        X_s(i,j)=sum(wx(commonpath,2));
        X_s(j,i)=X_s(i,j);
    end
end
%% Determining the Path of each Radius(:Routes from first Bus to each Endnode)
h=length(endnode);                                                  % h= Number of Radiuses
g=[0];                                                                  % EndBuses Path
to the first Bus
for route_no=1:h
    rnode=endnode(route_no);                                % Recieving node
    snode=s_buses(r_buses==rnode);                      %Sending Node
    g(route_no,1)=rnode;
    g(route_no,2)=snode;
    j=2;
    while(snode~=1)
        rnode=snode;
        g(route_no,j)=rnode;
        snode=s_buses(r_buses==rnode);
        g(route_no,j+1)=snode;
        j=j+1;
    end
end
%% Sorting Radius Matrix Elements
gs=g;  %gs is sorted form of g
for i=1:length(endnode)
    rout=sort(nonzeros(g(i,:)));
    gs(i,1:length(rout))=rout;
end
%% Forming Route Matrices for applications
gb=g;
mr=1;                                                   %MainRoute_Row_index in g
for i=1:size(gb,1)
    if length(nonzeros(gb(i,:)))>length(nonzeros(gb(mr,:)))
        mr=i;
    end
end
temp=gb(1,:);           %Main Route placed in at the 1st row
gb(1,:)=gb(mr,:);
```

```
gb(mr,:)=temp;
for i=1:size(gb,1)
    for j=1:size(gb,2)
        n=gb(i,j);
        for ii=((i+1):size(gb,1))
            for jj=1:(size(gb,2)-1)
                if gb(ii,jj)==n
                    gb(ii,jj+1)=0;
                end
            end
        end
    end
end
gv=gb;                  %gv matrix will be used for KVL in Forward steps
for i=1:length(endnode)
    rout=sort(nonzeros(gb(i,:)));
    gv(i,1:length(rout))=rout;
end
sc=zeros(size(gb,1),1);
for j=1:size(gb,1)
    for i=1:size(gb,1)-1
        a=length(nonzeros(gb(i,:)));
        b=length(nonzeros(gb(i+1,:)));
        if a>b
            t=gb(i,:);
            gb(i,:)=gb(i+1,:);
            gb(i+1,:)=t;
        end
    end
end
g=gb;
%% initial guess
v = ones(bus_no,1); %Bus Voltages vector Initialization (complex value) (flat initial guess)
v(pv_bus_sp(:,1))=pv_bus_sp(:,2);  % PV Buses initial guess according to their set points
I = zeros(br_no,1);     %Branches' Current vector Initialization % %C(br,bus)'
```

```
%% outer Iteration Loop
for oc=1:N2
    v_pv_old_abs=abs(v(pv_bus_no));
    %% Inner Iteration Loop (BFS)
    for ni=1:N1
        %% Backward Step
        vold=v;
        LC = conj(complex(P,Q)./v);     %Bus Load Currents  vector
        for r=1:route_no
            for i=1:size(g,2)-1
                b=g(r,i);
                if b==0
                    break;
                end
                if sum(C(:,b))==1
                    I(C(:,b)==1)=LC(b);
                    LC(g(r,i+1))=LC(g(r,i+1))+LC(b);
                else
                    if g(r,i+1)==1
                        I(C(:,b)==1)=LC(b);
                        break;
                    else
                        I(C(:,b)==1)=LC(b);
                        if g(r,i+1)~=0
                            LC(g(r,i+1))=LC(g(r,i+1))+LC(b);
                        end
                    end
```

```
                end
            end
        end
        %% Forward Step
        for r=1:route_no
            for i=1:size(gv,2)-1
                if gv(r,i+1)==0
                    continue;
                end
                b= find(C(:,gv(r,i+1))==1);
                v(gv(r,i+1))=v(gv(r,i))-complex(R(b),X(b))*I(b);
                %fprintf("V("+num2str(gv(r,i+1))+")=V("+num2str(gv(r,i))+")-
zI("+num2str(b)+")\n");
            end
        end
        vnew=v;
        if max(abs(vnew-vold))<e1
            %fprintf("Algorithm Converged!\nNumber of Iterations="+num2str(ni)+"\n----\n")
            break;
        end
    end
    v_pv_new_abs=abs(v(pv_bus_no));
    if max(abs(v_pv_new_abs-v_pv_old_abs))<e
        fprintf("Algorithm Converged!\nNumber of Iterations(outer loop)="+num2str(oc)+"\n--\n")
        break;
    end
    %% Updating Q Values
    DV=abs(pv_bus_sp(:,2))-abs(v(pv_bus_sp(:,1)));    %Calculating DeltaV Matrix
    DQ=X_s\DV;                                        %Calculating DeltaV Matrix
    Q(pv_bus_no,1)=Q(pv_bus_no,1)-sign(DV).*DQ;       %Updating Q values (pu)
    for qqi=1:length(pv_bus_no)                       %Checking Q limits (pu)
        qq=pv_bus_no(qqi);
        if(Q(qq,1)<QLoad(qq)-abs(pv_qm_gen(qq)))
            Q(qq,1)=QLoad(qq)-abs(pv_qm_gen(qq));
        elseif(Q(qq,1)>(abs(pv_qm_con(qq)+QLoad(qq))))
            Q(qq,1)=abs(pv_qm_con(qq))+(QLoad(qq));
        end
    end
end
%% Print Calculated Values
```

```
vbp=[abs(v),angle(v).*(180/pi)];220. vbp2=[((1:bus_no)'),abs(v),angle(v).*(180/pi)]; %'
h={'Bus NO','|U|(pu)','?(°)'};
T = array2table(vbp2,'VariableNames',h);
T2 = array2table(round(vbp2,2),'VariableNames',h);
%% Print
fprintf("Final Voltages:\n")
disp(T2)
f=figure;
t=uitable(f,'data',vbp2,'columnname',h);
%% Plot Voltage Profile
f2=figure;
p=plot(vbp2(:,1),vbp2(:,2),'-b','LineWidth',2);
hold on
plot([0 bus_no],[0.95 0.95],'-r','LineWidth',1);
plot([0 bus_no],[0.9 0.9],'-r','LineWidth',1);
hold off
xlim([0 bus_no])
ylim([0.85 1.02])
yaxes=[[0.86:0.02:0.95],[0.95:0.01:1.2]];
yticks(yaxes)
xticks(0:bus_no)
xlabel("BUS Number")
```

```
ylabel("V_{Line} pu")
grid on
%%
```

```
Ibrpu=[abs(I) angle(I)*180/pi];     % Branches' Currents in pu magnitude and angle '
%% Line Consumed Power Calculation
PL  = R.*(abs(I).^2);       % Active Power (pu) Consumed by each Line
QL = X.*(abs(I).^2);        %Reactive Power (pu) Consumed by each Line
PLkW=(PL)*Sb*1000;
QLkVAr=(QL)*Sb*1000;
PLt=sum(PL)*Sb*1000;    %Total kW consumed by lines (total power loss)
QLt=sum(QL)*Sb*1000;    %Total kVAr consumed by lines
%% Print
fprintf('-----------------------------------------------------------------------\n')
fprintf("Total Power Loss in lines (kW) ="+num2str(PLt)+'\n');
fprintf("Total ReactivePower Consumed by Lines (kVAr) ="+num2str(QLt)+'\n');
```

**Thank you for your time.**