2019 级《电力系统分析》课程

第一次大作业

学　　院　电气工程及信息工程学院

专　　业　电气工程及其自动化（基地班）

姓　　名　龙茂林

学　　号　1905230342

指导教师　陈 伟

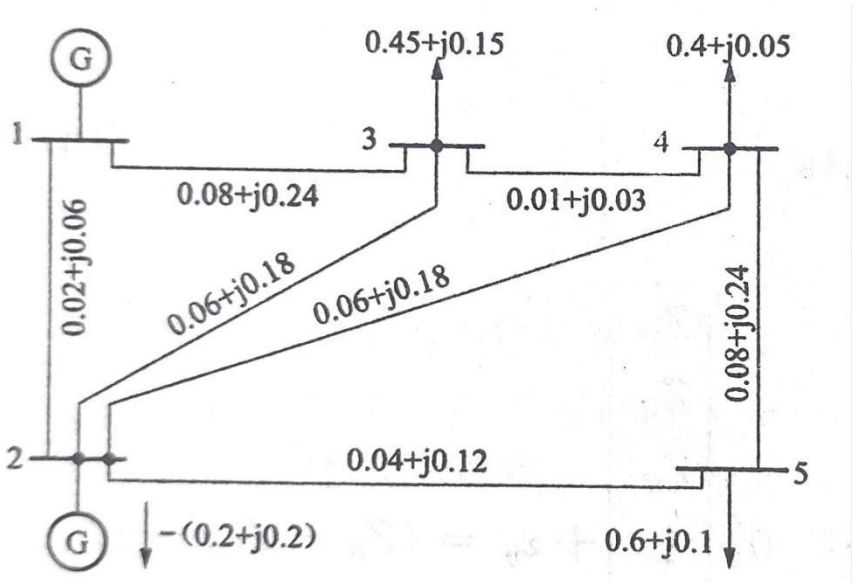
实习时间　2021.10 -2021.11

目　录

一、任务简介

编程实现课本 133 页的例【例 4-3】

【例 4-3】 例【4-2】所示网络中，节点 2 连接的实际是发给发定功率的发电厂，设节点 1 电压保持为 U1=1.06=定值，试运用以直角坐标表示的牛顿-拉夫逊法计算例 4-2 所示系统中的潮流分布。计算精确度要求各节点电压修正量不大于 10-5。



具体要求：

1、C/C++语言实现；

2、现场演示计算过程和结果；

3、提交报告和源程序。

二、计算原理

1、节点导纳矩阵

$$\begin{bmatrix} I_4 \\ I_2 \\ I_3 \\ \vdots \\ I_4 \end{bmatrix} = \begin{bmatrix} Y_{11} & Y_{12} & Y_{13} & \cdots & Y_{1n} \\ Y_{21} & Y_{22} & Y_{23} & \cdots & Y_{2n} \\ Y_{31} & Y_{32} & Y_{33} & \cdots & Y_{3n} \\ \vdots & & & & \\ Y_{n1} & Y_{n2} & Y_{n3} & \cdots & Y_{nn} \end{bmatrix} \begin{bmatrix} U_3 \\ U_2 \\ U_3 \\ \vdots \\ U_n \end{bmatrix}$$

其中：

I 是节点注人电流的列向量。在电力系统计算中，节点注人电流可理解为各节点电源电流与负荷电流之和，并规定电源流向网络的注人电流为正。因此，仅有负荷的负荷节点注人电流就具有负值。

U 是节点电压的列向量。因通常以大地做参考节点，网络中有接地支路时，节点电压通常就指该节点的对地电压。网络中没有接地支路时，各节点电压可指各节点与某一个被选定作参考节点之间的电压差。

Y 是一个 n*n 阶的节点导纳矩阵，其阶数 n 就等于网络中除参考节点外的节点数。

2、电力系统的节点

电力系统的节点基本类型有以下三种：

第一类称 PQ 节点。PQ 节点一般为负荷节点、联络节点和给定功率运行的发电机节点，有功和无功功率为给定值，反映负荷或者负荷电压的调节功能，节点电压幅值和相角未知。

第二类称 PV 节点。PV 节点一般为有无功储备的发电机节点和有一定无功功率电源的变电所母线节点，有功功率和电压幅值为给定值，反映发电机控制系统维持发电机有功功率和电压幅值恒定的功能，无功功率和电压相角未知。

第三类称平衡节点。潮流计算中的参考节点，代表电压恒定的无穷大系统母线，有功和无功容量大小不受限制，节点电压幅值和相角为给定值。

其中 PQ 与 PV 的计算公式如下：

$$\triangle P_i = P_i - \sum_{j=1}^{j=n}[e_i(G_{ij}e_j - B_{ij}f_j) + f(G_{ij}f_j + B_{ij}e_j)]$$

$$\triangle Q_i = Q_i - \sum_{j=1}^{j=n}[f(G_{ij}f_j + B_{ij}e_j) - e_i(G_{ij}e_j - B_{ij}f_j)]$$

$$\triangle U_i^2 = U_i^2 - (e_i^2 + f_i^2)$$

3、牛顿拉夫逊原理

牛顿迭代法是取 x0,在此基础上，求出比 x0 更接近的方程的解，通常情况下，负载功率应用于每个总线是已知的，但每个节点的电压是未知的(平衡节点除外)。根据网络结构可以形成一个节点的电导矩阵，然后显示出该节点的电导矩阵。通过电导矩阵和功率方程，可以将原始的问题转化成对于非线性问题的求解，可以采用牛顿拉夫逊高斯方法，即通过每--节点的斜率作切线与横轴相交的点，所对应的纵坐标为下一个球节点，通过-一次又一次的迭代更新，分别在每一次迭代过程中形成初始值，将初始值代入功率方程进行不断修正，经过三次到四次便可以得到所需要的求解值。

设节点总数为 n,PV 节点数为 r,则潮流应包含 n-1 个有功方程和 n-r-1 个无功方程。

牛顿法的修正方程式为:

$$\begin{bmatrix} \triangle \mathrm{P} \\ \triangle \mathrm{Q} \end{bmatrix} = \begin{bmatrix} \mathrm{H} & \mathrm{N} \\ \mathrm{J} & \mathrm{L} \end{bmatrix} \begin{bmatrix} \triangle \theta \\ \triangle V / V \end{bmatrix}$$

设 n 维非线性方程组为:

$$
\begin{aligned}
&f_1(x_1,x_2,...,x_n)=y_1\\
&f_2(x_1,x_2,...,x_n)=y_2\\
&\vdots\\
&f_n(x_1,x_2,...,x_n)=y_n
\end{aligned}
$$

求解得到修正量，并对各变量修正得：

xi(k+1)=xi(k)+Δxi(k), i=1,2,.....,n(1-8)

很容易看到，修改后的雅可比矩阵方程的系数矩阵具有以下特点:

1)相量矩阵为矢量矩阵;

2)求解矩阵具有不对称性;

3)雅可比矩阵在拓扑学意义上具有稀疏性。

现在把牛顿法推广到多变量非线性方程组的情况。设有变量的非线性联立方程组:

$$
\begin{aligned}
&f_1(x_1,x_2,...,x_n)=0\\
&f_2(x_1,x_2,...,x_n)=0\\
&\vdots\\
&f_n(x_1,x_2,...,x_n)=0
\end{aligned}
$$

取变量初值 x1(0),x2(0),.....,xn(0),假设Δx1(0), Δx2(0),....,△x(0}为其修正量，且满足:

$$
\begin{aligned}
&f_1(x_1-\triangle x_1^{(0)},x_2-\triangle x_2^{(0)},\cdots,x_n-\triangle x_n^{(0)})=0\\
&f_2(x_1-\triangle x_1^{(0)},x_2-\triangle x_2^{(0)},\cdots,x_n-\triangle x_n^{(0)})=0\\
&\vdots\\
&f_n(x_1-\triangle x_1^{(0)},x_2-\triangle x_2^{(0)},\cdots,x_n-\triangle x_n^{(0)})=0
\end{aligned}
$$

从几何意义上说，牛顿法就是切线法，是一种逐渐线性化的方法。

4、修正方程

以矩阵形式表示的修正方程如下：

$$\begin{bmatrix} \Delta P_1 \\ \Delta Q_1 \\ \Delta P_2 \\ \Delta Q_2 \\ \vdots \\ \Delta P_p \\ \Delta U_p^{\ 2} \\ \Delta P_n \\ \Delta U_n^{\ 2} \end{bmatrix} = \begin{bmatrix} H_{11} & N_{11} & H_{12} & N_{12} & \cdots & H_{1p} & N_{1p} & H_{1n} & N_{1n} \\ J_{11} & L_{11} & J_{12} & L_{12} & \cdots & J_{1p} & L_{1p} & J_{1n} & L_{1n} \\ H_{21} & N_{21} & H_{22} & N_{22} & \cdots & H_{2p} & N_{2p} & H_{2n} & N_{2n} \\ J_{21} & L_{21} & J_{22} & L_{22} & \cdots & J_{2p} & L_{2p} & J_{2n} & L_{2n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ H_{p1} & N_{p1} & H_{p2} & N_{p2} & \cdots & H_{pp} & N_{pp} & H_{pn} & N_{pn} \\ R_{p1} & S_{p1} & R_{p2} & S_{p2} & \cdots & R_{pp} & S_{pp} & R_{pn} & S_{pn} \\ H_{n1} & N_{n1} & H_{n2} & N_{n2} & \cdots & H_{np} & N_{np} & H_{nn} & N_{nn} \\ R_{n1} & S_{n1} & R_{n2} & S_{n2} & \cdots & R_{np} & S_{np} & R_{nn} & S_{nn} \end{bmatrix} \begin{bmatrix} \Delta f_1 \\ \Delta e_1 \\ \Delta f_2 \\ \Delta e_2 \\ \vdots \\ \Delta f_p \\ \Delta e_p \\ \Delta f_n \\ \Delta e_n \end{bmatrix}$$

当 j≠i 时，对于特点的 j，只有该特定的 fi 和 ei 是变量，于是雅克布矩阵中各非对角元素表示为：

$$H_{ij} = \frac{\partial P_i}{\partial f_j} = -B_{ij}e_i + G_{ij}f_i ; \qquad N_{ij} = \frac{\partial P_i}{\partial e_j} = G_{ij}e_i + B_{ij}f_i \ ;$$

$$J_{ij} = \frac{\partial Q_i}{\partial f_j} = -N_{ij} ; \qquad L_{ij} = \frac{\partial Q_i}{\partial e_i} = H_{ij} ;$$

$$R_{ij} = \frac{\partial U_1^2}{\partial f_j} = 0; \qquad S_{ij} = \frac{\partial U_1^2}{\partial e_j} = 0$$

当 j＝i 时，雅可比矩阵中各对角元素的表示式为：

$$H_{ii} = \frac{\partial P_i}{\partial f_i} = -B_{ii}e_i + G_{ii}f_i + b_{ii} ; \quad N_{ii} = \frac{\partial P_i}{\partial e_i} = G_{ii}e_i + B_{ii}f + a_{ii} ;$$

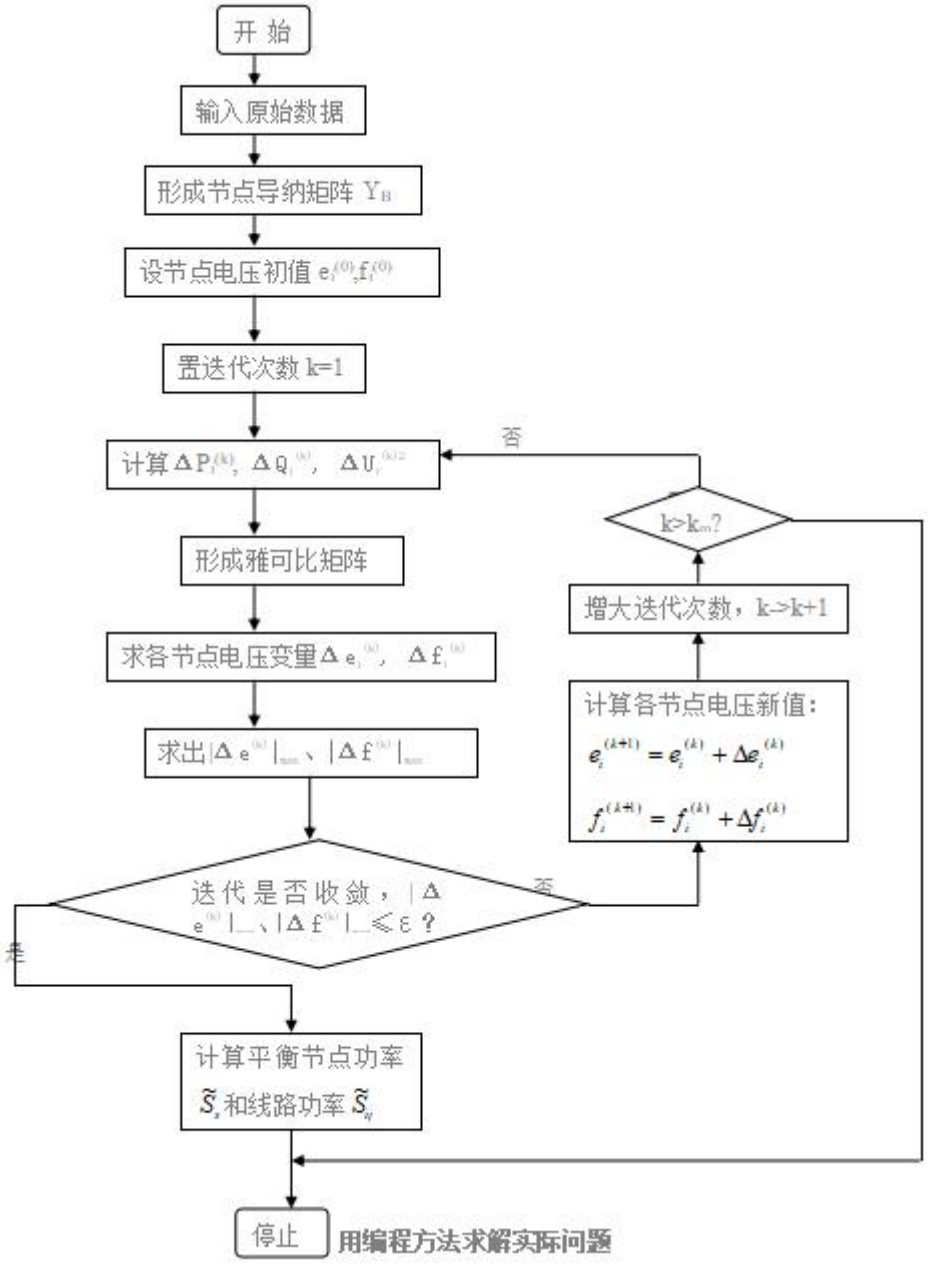
$$N_{ii} = \frac{\partial Q_i}{\partial e_j} = -G_{ii}e_i - B_{ii}f + a_{ii} ; \quad H_{ii} = \frac{\partial P_i}{\partial f_i} = -B_{ii}e_i + G_{ii}f_i - b_{ii} ;$$

$$R_{ii} = \frac{\partial U_i^2}{\partial f_i} = 2f_i ; \qquad S_{ii} = \frac{\partial U_i^2}{\partial e_i} = 2e_i$$

其中：$a_{ii} = (G_{ii}e_i - B_{ii}f_i) + \sum_{\substack{j=1 \\ \neq i}}^{n} (G_{ij}f_i + B_{ij}f_i)$

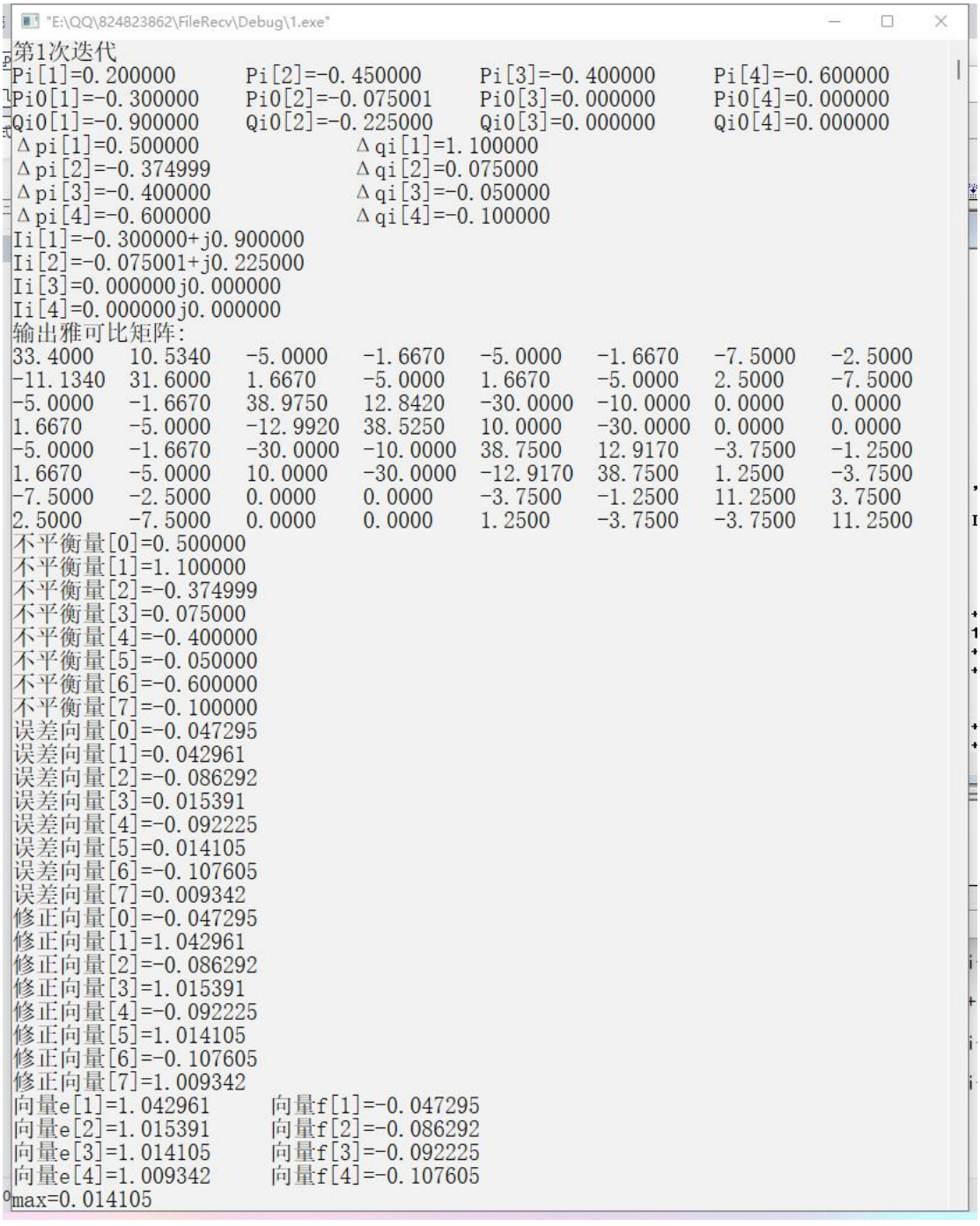
$$b_{ii} = (G_{ii}e_j - B_{ii}f_j) + \sum_{\substack{j=1 \\ \neq i}}^{n}(G_{ij}f_j + B_{ij}e_j)$$

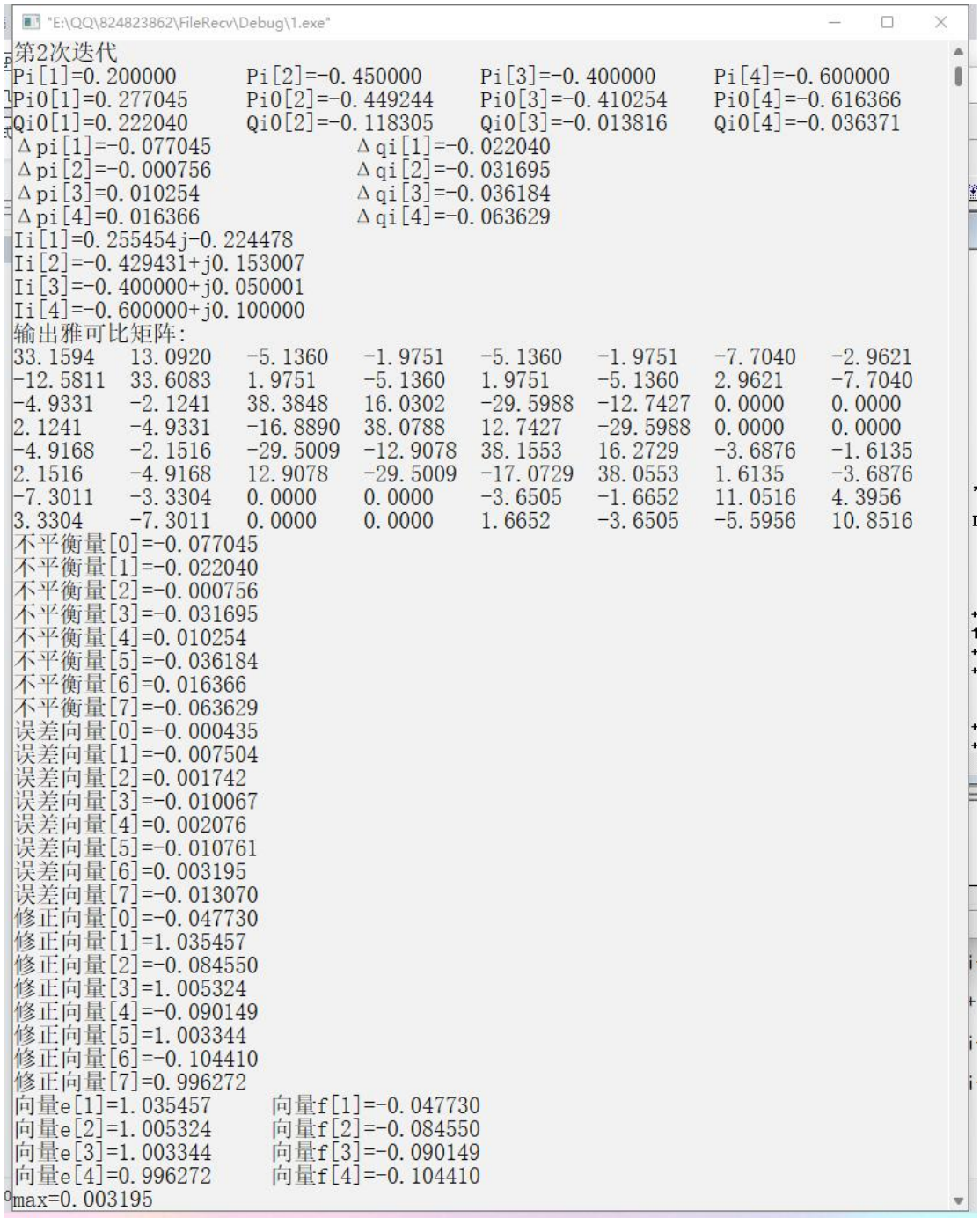
5、计算流程

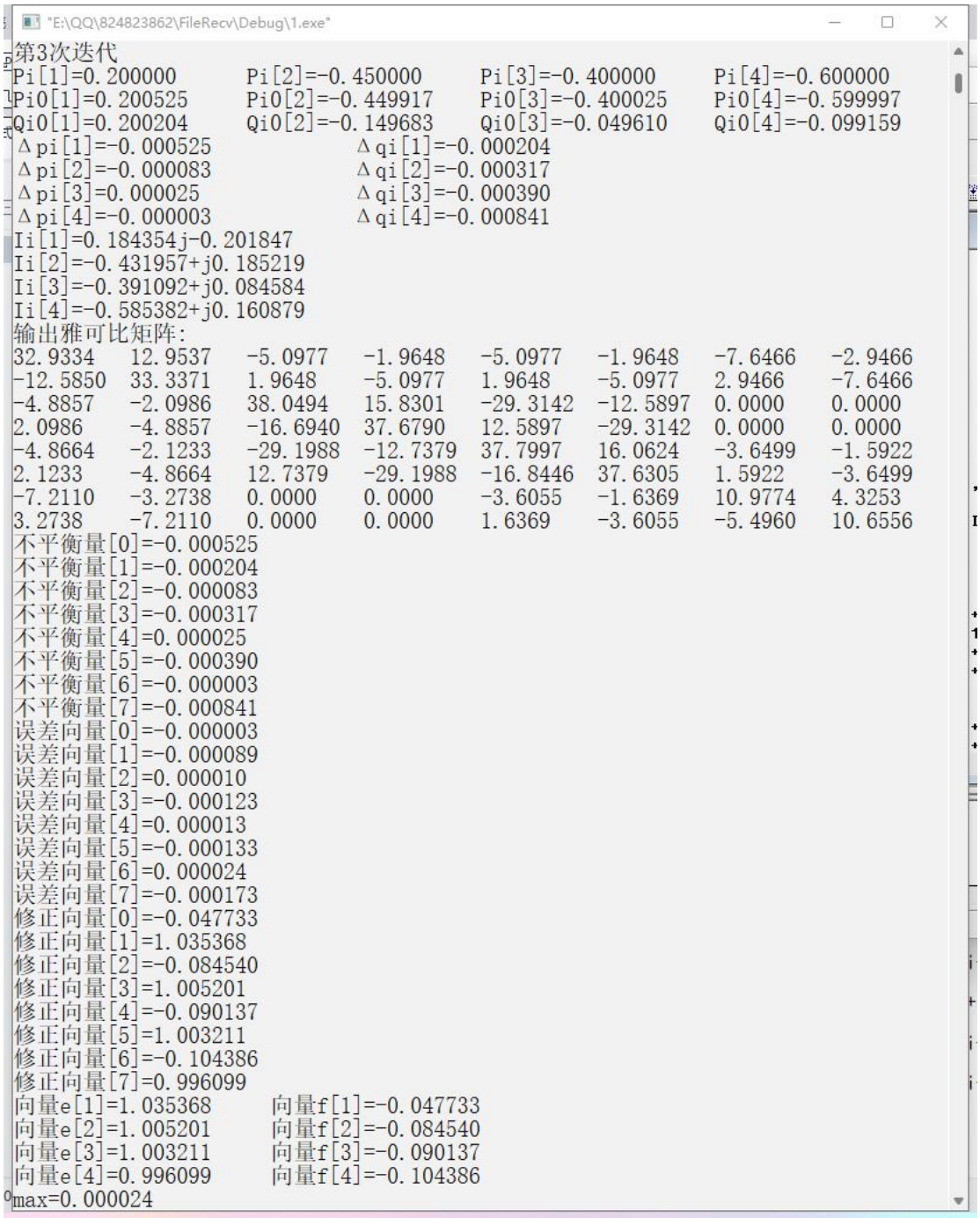


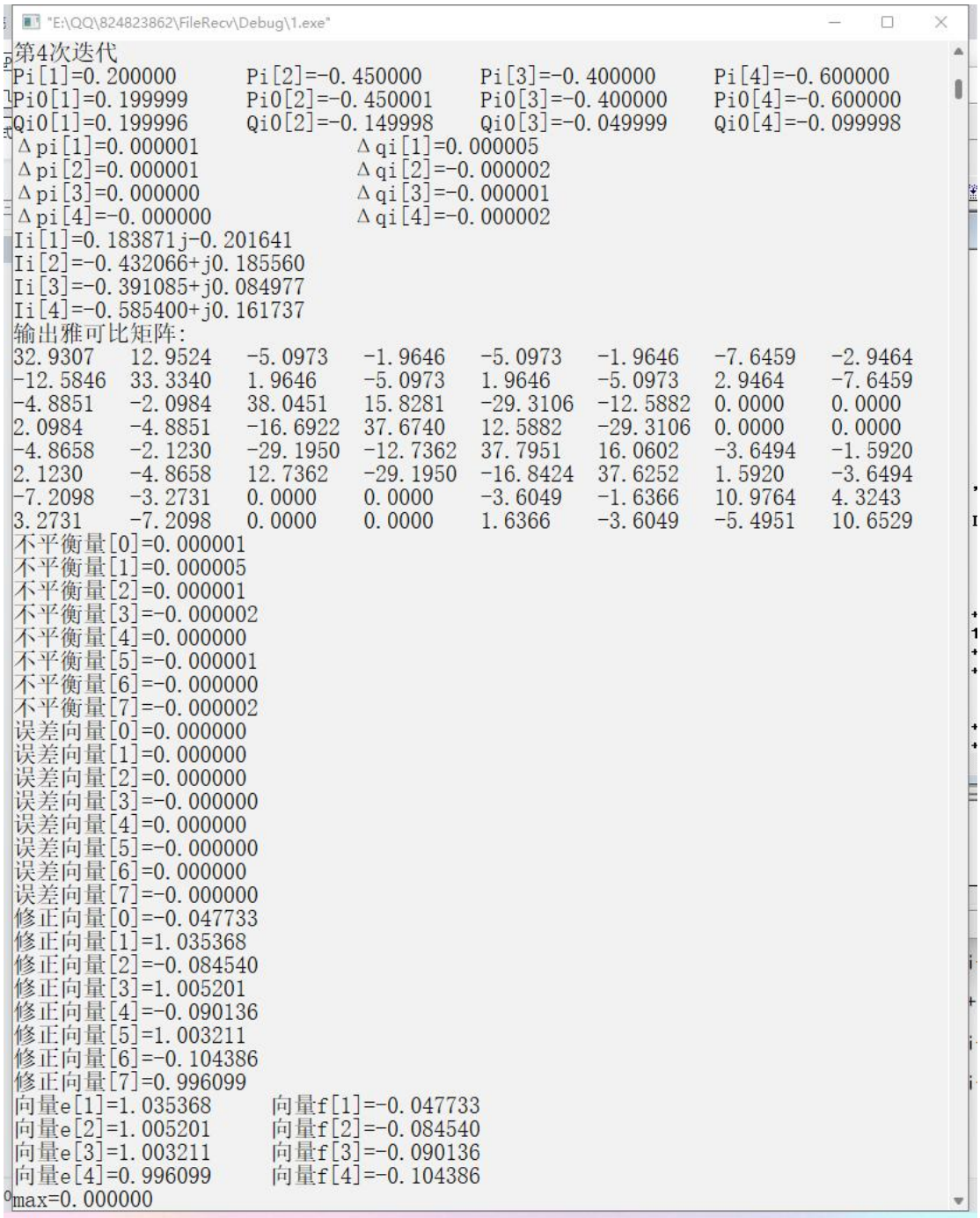
用编程方法求解实际问题

三、结果分析


"E:\QQ\824823862\FileRecv\Debug\1.exe"
第1次迭代
Pi[1]=0.200000 Pi[2]=-0.450000 Pi[3]=-0.400000 Pi[4]=-0.600000
Pi0[1]=-0.300000 Pi0[2]=-0.075001 Pi0[3]=0.000000 Pi0[4]=0.000000
Qi0[1]=-0.900000 Qi0[2]=-0.225000 Qi0[3]=0.000000 Qi0[4]=0.000000
Δpi[1]=0.500000 Δqi[1]=1.100000
Δpi[2]=-0.374999 Δqi[2]=0.075000
Δpi[3]=-0.400000 Δqi[3]=-0.050000
Δpi[4]=-0.600000 Δqi[4]=-0.100000
Ii[1]=-0.300000+j0.900000
Ii[2]=-0.075001+j0.225000
Ii[3]=0.000000j0.000000
Ii[4]=0.000000j0.000000
输出雅可比矩阵:
33.4000 10.5340 -5.0000 -1.6670 -5.0000 -1.6670 -7.5000 -2.5000
-11.1340 31.6000 1.6670 -5.0000 1.6670 -5.0000 2.5000 -7.5000
-5.0000 -1.6670 38.9750 12.8420 -30.0000 -10.0000 0.0000 0.0000
1.6670 -5.0000 -12.9920 38.5250 10.0000 -30.0000 0.0000 0.0000
-5.0000 -1.6670 -30.0000 -10.0000 38.7500 12.9170 -3.7500 -1.2500
1.6670 -5.0000 10.0000 -30.0000 -12.9170 38.7500 1.2500 -3.7500
-7.5000 -2.5000 0.0000 0.0000 -3.7500 -1.2500 11.2500 3.7500
2.5000 -7.5000 0.0000 0.0000 1.2500 -3.7500 -3.7500 11.2500
不平衡量[0]=0.500000
不平衡量[1]=1.100000
不平衡量[2]=-0.374999
不平衡量[3]=0.075000
不平衡量[4]=-0.400000
不平衡量[5]=-0.050000
不平衡量[6]=-0.600000
不平衡量[7]=-0.100000
误差向量[0]=-0.047295
误差向量[1]=0.042961
误差向量[2]=-0.086292
误差向量[3]=0.015391
误差向量[4]=-0.092225
误差向量[5]=0.014105
误差向量[6]=-0.107605
误差向量[7]=0.009342
修正向量[0]=-0.047295
修正向量[1]=1.042961
修正向量[2]=-0.086292
修正向量[3]=1.015391
修正向量[4]=-0.092225
修正向量[5]=1.014105
修正向量[6]=-0.107605
修正向量[7]=1.009342
向量e[1]=1.042961 向量f[1]=-0.047295
向量e[2]=1.015391 向量f[2]=-0.086292
向量e[3]=1.014105 向量f[3]=-0.092225
向量e[4]=1.009342 向量f[4]=-0.107605
max=0.014105


"E:\QQ\824823862\FileRecv\Debug\1.exe"
第2次迭代
Pi[1]=0.200000 Pi[2]=-0.450000 Pi[3]=-0.400000 Pi[4]=-0.600000
Pi0[1]=0.277045 Pi0[2]=-0.449244 Pi0[3]=-0.410254 Pi0[4]=-0.616366
Qi0[1]=0.222040 Qi0[2]=-0.118305 Qi0[3]=-0.013816 Qi0[4]=-0.036371
Δpi[1]=-0.077045 Δqi[1]=-0.022040
Δpi[2]=-0.000756 Δqi[2]=-0.031695
Δpi[3]=0.010254 Δqi[3]=-0.036184
Δpi[4]=0.016366 Δqi[4]=-0.063629
Ii[1]=0.255454j-0.224478
Ii[2]=-0.429431+j0.153007
Ii[3]=-0.400000+j0.050001
Ii[4]=-0.600000+j0.100000
输出雅可比矩阵:
33.1594 13.0920 -5.1360 -1.9751 -5.1360 -1.9751 -7.7040 -2.9621
-12.5811 33.6083 1.9751 -5.1360 1.9751 -5.1360 2.9621 -7.7040
-4.9331 -2.1241 38.3848 16.0302 -29.5988 -12.7427 0.0000 0.0000
2.1241 -4.9331 -16.8890 38.0788 12.7427 -29.5988 0.0000 0.0000
-4.9168 -2.1516 -29.5009 -12.9078 38.1553 16.2729 -3.6876 -1.6135
2.1516 -4.9168 12.9078 -29.5009 -17.0729 38.0553 1.6135 -3.6876
-7.3011 -3.3304 0.0000 0.0000 -3.6505 -1.6652 11.0516 4.3956
3.3304 -7.3011 0.0000 0.0000 1.6652 -3.6505 -5.5956 10.8516
不平衡量[0]=-0.077045
不平衡量[1]=-0.022040
不平衡量[2]=-0.000756
不平衡量[3]=-0.031695
不平衡量[4]=0.010254
不平衡量[5]=-0.036184
不平衡量[6]=0.016366
不平衡量[7]=-0.063629
误差向量[0]=-0.000435
误差向量[1]=-0.007504
误差向量[2]=0.001742
误差向量[3]=-0.010067
误差向量[4]=0.002076
误差向量[5]=-0.010761
误差向量[6]=0.003195
误差向量[7]=-0.013070
修正向量[0]=-0.047730
修正向量[1]=1.035457
修正向量[2]=-0.084550
修正向量[3]=1.005324
修正向量[4]=-0.090149
修正向量[5]=1.003344
修正向量[6]=-0.104410
修正向量[7]=0.996272
向量e[1]=1.035457 向量f[1]=-0.047730
向量e[2]=1.005324 向量f[2]=-0.084550
向量e[3]=1.003344 向量f[3]=-0.090149
向量e[4]=0.996272 向量f[4]=-0.104410
max=0.003195


"E:\QQ\824823862\FileRecv\Debug\1.exe"
第3次迭代
Pi[1]=0.200000 Pi[2]=-0.450000 Pi[3]=-0.400000 Pi[4]=-0.600000
Pi0[1]=0.200525 Pi0[2]=-0.449917 Pi0[3]=-0.400025 Pi0[4]=-0.599997
Qi0[1]=0.200204 Qi0[2]=-0.149683 Qi0[3]=-0.049610 Qi0[4]=-0.099159
Δpi[1]=-0.000525 Δqi[1]=-0.000204
Δpi[2]=-0.000083 Δqi[2]=-0.000317
Δpi[3]=0.000025 Δqi[3]=-0.000390
Δpi[4]=-0.000003 Δqi[4]=-0.000841
Ii[1]=0.184354j-0.201847
Ii[2]=-0.431957+j0.185219
Ii[3]=-0.391092+j0.084584
Ii[4]=-0.585382+j0.160879
输出雅可比矩阵:
32.9334 12.9537 -5.0977 -1.9648 -5.0977 -1.9648 -7.6466 -2.9466
-12.5850 33.3371 1.9648 -5.0977 1.9648 -5.0977 2.9466 -7.6466
-4.8857 -2.0986 38.0494 15.8301 -29.3142 -12.5897 0.0000 0.0000
2.0986 -4.8857 -16.6940 37.6790 12.5897 -29.3142 0.0000 0.0000
-4.8664 -2.1233 -29.1988 -12.7379 37.7997 16.0624 -3.6499 -1.5922
2.1233 -4.8664 12.7379 -29.1988 -16.8446 37.6305 1.5922 -3.6499
-7.2110 -3.2738 0.0000 0.0000 -3.6055 -1.6369 10.9774 4.3253
3.2738 -7.2110 0.0000 0.0000 1.6369 -3.6055 -5.4960 10.6556
不平衡量[0]=-0.000525
不平衡量[1]=-0.000204
不平衡量[2]=-0.000083
不平衡量[3]=-0.000317
不平衡量[4]=0.000025
不平衡量[5]=-0.000390
不平衡量[6]=-0.000003
不平衡量[7]=-0.000841
误差向量[0]=-0.000003
误差向量[1]=-0.000089
误差向量[2]=0.000010
误差向量[3]=-0.000123
误差向量[4]=0.000013
误差向量[5]=-0.000133
误差向量[6]=0.000024
误差向量[7]=-0.000173
修正向量[0]=-0.047733
修正向量[1]=1.035368
修正向量[2]=-0.084540
修正向量[3]=1.005201
修正向量[4]=-0.090137
修正向量[5]=1.003211
修正向量[6]=-0.104386
修正向量[7]=0.996099
向量e[1]=1.035368 向量f[1]=-0.047733
向量e[2]=1.005201 向量f[2]=-0.084540
向量e[3]=1.003211 向量f[3]=-0.090137
向量e[4]=0.996099 向量f[4]=-0.104386
max=0.000024



总计 4 次迭代
平衡节点功率=1.298157+j0.244472
输出电压极坐标形式
u1=1.060000∠0.000000
u2=1.036468∠-2.639599
u3=1.008750∠-4.807429
u4=1.007252∠-5.134129
u5=1.001554∠-5.982488
Press any key to continue

四、总结

电力系统稳态分析是研究电力系统运行和规划方案最重要和最基本的手段，其中潮流计算针对电力系统的各种正常的运行方式进行稳态分析。本次大作业让我了解电力系统的基本原理和分析方法，熟悉了 C 语言编程，体会到了科技对今后学习与生活的帮助，加深了对电力系统潮流计算的理解与应用，为今后从事电力工程设计、运行和维护打下良好的基础。同时也意识到了自己的不足之处，光有理论知识是远远不够的，现在的我们更需要的是理论与实践的结合，从而达到学以致用，也至此明白了今后的学习目标与方向，受益匪浅。

五、源程序清单

```
#include <stdio.h>
#include <math.h>

float divRe(float a,float b,float c,float d);
float divIm(float a,float b,float c,float d);
float mulRe(float a,float b,float c,float d);
float mulIm(float a,float b,float c,float d);
float Max(float a[],int n);                    #引用自定义函数

int main(void)              #主函数
{
int i,j,k=0,n,km=6;         #写入题目所给变量
float
a,b,c,d,e,f,g,h,max,eps=0.00001,
pi0[5],qi0[5],
detpi[5],detqi[5],
```

```
Iir0[5],Iii0[5],J0[8][8],si[8],ui[8],u[8][8],l[8][8],y[8],ui1[8],
H[4][4],N[4][4],J[4][4],L[4][4],
ei1[5],fi1[5];
Static float
R[5][5]={{6.250,-5.000,-1.250,0,0},{-5.000,10.834,-1.667,-1.667,-2.500},{-1.250,-1.667,12.917,-10.000,0},{0,-1.667,-10.000,12.917,-1.250},{0,-2.500,0,-1.250,3.750}};                    #写入导纳矩阵实部
Static float
I[5][5]={{-18.750,15.000,3.750,0,0},{15.000,-32.500,5.000,5.000,7.500},{3.750,5.000,-38.750,30.000,0},{0,5.000,30.000,-38.750,3.750},{0,7.500,0,3.750,-11.250}};                    #写入导纳矩阵虚部

 float ei0[5]={1.06,1.0,1.0,1.0,1.0};          #写入节点电压实部
 float fi0[5]={0,0,0,0,0};                     #写入节点电压虚部
 float pi[5]={0,0.2,-0.45,-0.4,-0.6};          #写入注入有功向量
 float qi[5]={0,0.2,-0.15,-0.05,-0.1};         #写入注入无功向量

 do{                                #进入 do 循环
   k+=1;
   printf("第%d 次迭代\n",k);       #输出第几次迭代
   for(i=1;i<5;i++)
      printf("Pi[%d]=%-14.6f",i,pi[i]);  #输出 pi
  b=0;
  d=0;
  for(i=1;i<5;i++)
     {for(j=0;j<5;j++)
    {

a=(ei0[i]*(ybr[i][j]*ei0[j]-ybi[i][j]*fi0[j])+fi0[i]*(ybr[i][j]*fi0[j]
```

```
+ybi[i][j]*ei0[j]));          #计算 pi0
     b+=a;
    }
       pi0[i]=b;
       printf("Pi0[%d]=%-13.6f",i,pi0[i]);       #输出 pi0
       b=0;
      }
  for(i=1;i<5;i++)
     {for(j=0;j<5;j++)
    {

c=(fi0[i]*(ybr[i][j]*ei0[j]-ybi[i][j]*fi0[j])-ei0[i]*(ybr[i][j]*fi0[j]
+ybi[i][j]*ei0[j]));            #计算 qi0
     d+=c;
    }
      qi0[i]=d;
      printf("Qi0[%d]=%-13.6f",i,qi0[i]);       #输出 pi0
      d=0;
      }
  for(i=1;i<5;i++)
     {
      detpi[i]=pi[i]-pi0[i];           #计算Δpi
      detqi[i]=qi[i]-qi0[i];            #计算Δqi
      printf("Δpi[%d]=%-21.6f",i,detpi[i]);
      printf("Δqi[%d]=%-21.6f\n",i,detqi[i]);
     }
  for(i=1;i<5;i++)
    {Iir0[i]=divRe(pi0[i],-qi0[i],ei0[i],-fi0[i]);
     Iii0[i]=divIm(pi0[i],-qi0[i],ei0[i],-fi0[i]);
```

```
     if(Iii0[i]>0)
        printf("Ii[%d]=%-.6f+j%-22.6f\n",i,Iir0[i],Iii0[i]);
     else
        printf("Ii[%d]=%-.6fj%-22.6f\n",i,Iir0[i],Iii0[i]);
    }                                   #输出各节点注入电流
  for(i=0;i<4;i++)
    {for(j=0;j<4;j++)
    if(i==j)
 {
H[i][j]=-ybi[i+1][j+1]*ei0[i+1]+ybr[i+1][j+1]*fi0[i+1]+Iii0[i+1];
N[i][j]=ybr[i+1][j+1]*ei0[i+1]+ybi[i+1][j+1]*fi0[i+1]+Iir0[i+1];
J[i][j]=-ybr[i+1][j+1]*ei0[i+1]-ybi[i+1][i+1]*fi0[i+1]+Iir0[i+1];
L[i][j]=-ybi[i+1][j+1]*ei0[i+1]+ybr[i+1][j+1]*fi0[i+1]-Iii0[i+1];
       }
    else {
          H[i][j]=ybr[i+1][j+1]*fi0[i+1]-ybi[i+1][j+1]*ei0[i+1];
          N[i][j]=ybr[i+1][j+1]*ei0[i+1]+ybi[i+1][j+1]*fi0[i+1];
          J[i][j]=-ybi[i+1][j+1]*fi0[i+1]-ybr[i+1][j+1]*ei0[i+1];
          L[i][j]=ybr[i+1][j+1]*fi0[i+1]-ybi[i+1][j+1]*ei0[i+1];
         }
}                                       #生成雅可比矩阵

  for(i=0;i<8;i++)
     for(j=0;j<8;j++){
    if(i%2==0 && j%2==0) J0[i][j]=H[i/2][j/2];
    else if(i%2==0&&j%2!=0) J0[i][j]=N[i/2][(j-1)/2];
    else if(i%2!=0&&j%2==0) J0[i][j]=J[(i-1)/2][j/2];
    else J0[i][j]=L[i/2][(j-1)/2];
     }
```

```
   printf("输出雅可比矩阵:\n");
for(i=0;i<8;i++)
    for(j=0;j<8;j++)
       printf("%-10.4f",J0[i][j]);            #逐个输出雅各比矩阵数据
for(i=0;i<8;i++)
  {if(i%2==0) si[i]=detpi[(i+2)/2];
   else si[i]=detqi[(i+1)/2];
   printf("\n 不平衡量[%d]=%-11.6f\r",i,si[i]);      #输出不平衡量
  }
for(i=0;i<8;i++) u[i][i]=1.000;
for(n=0;n<8;n++)
    {for(i=n;i<8;i++)
   {l[i][n]=J0[i][n];
    for(j=0;j<=n-1;j++)
       l[i][n]-=(l[i][j]*u[j][n]);
   }
     for(j=n+1;j<8;j++)
   {u[n][j]=J0[n][j];
    for(i=0;i<=n-1;i++)
       u[n][j]-=(l[n][i]*u[i][j]);
       u[n][j] /= l[n][n];
   }
    }
for(i=0;i<8;i++)
    {y[i]=si[i];
     for(j=0;j<=i-1;j++)
   y[i]-=(l[i][j]*y[j]);
   y[i]/=l[i][i];
    }
```

```
for(i=7;i>=0;i--)
   {ui[i]=y[i];
    for(j=i+1;j<n;j++)
  ui[i]-=(u[i][j]*ui[j]);
   }                                    #计算误差量、修正量、向量 e f
for(i=0;i<8;i++)
    printf("\n 误差向量[%d]=%-11.6f",i,ui[i]);        #输出误差量
for(i=0;i<8;i++)
   {
  {if(i%2==0) ui1[i]=ui[i]+fi0[i/2+1];
   else ui1[i]=ui[i]+ei0[(i+1)/2];
  }
    printf("\n 修正向量[%d]=%-13.6f",i,ui1[i]);       #输出修正量
   }
for(i=1;i<5;i++)
  {ei1[i]=ui1[2*i-1];
   fi1[i]=ui1[2*i-2];
  }
for(i=1;i<5;i++)
  {printf("\n 向量 e[%d]=%-13.6f",i,ei1[i]);          #输出向量 e
   printf("向量 f[%d]=%-13.6f",i,fi1[i]);            #输出向量 f
  }
max=Max(ui,8);
printf("\nmax=%f\n",max);
for(i=1;i<5;i++)
   {ei0[i]=ei1[i];
    fi0[i]=fi1[i];
   }
for(i=1;i<5;i++)
```

```
    {pi[i]=detpi[i]+pi0[i];
     qi[i]=detqi[i]+qi0[i];
    }
}while(max>eps && k<km);            #判定是否继续迭代


printf("总计 %d 次迭代\n",k);          #统计共计几次迭代
e=0;
f=0;
for(i=0;i<k+1;i++)
  {
   g=mulRe(ybr[0][i],-ybi[0][i],ei0[i],-fi0[i]);
   h=mulIm(ybr[0][i],-ybi[0][i],ei0[i],-fi0[i]);
   e+=g;
   f+=h;
  }
pi[0]=mulRe(ei0[0],fi0[0],e,f);
qi[0]=mulIm(ei0[0],fi0[0],e,f);
printf("S1=%f+j%f\n",pi[0],qi[0]);
printf("输出电压极坐标形式\n");
ei1[0]=ei0[0];
fi1[0]=fi0[0];
for(i=0;i<k+1;i++)
  printf("u%d=%f
∠%f\n",i+1,sqrt(ei1[i]*ei1[i]+fi1[i]*fi1[i]),atan(fi1[i]/ei1[i])*180
/3.1415926);                    #输出幅值与相角
}

#自定义函数部分
```

```
float divRe(float a,float b,float c,float d) #定义函数计算注入电压实部
{
    float e;
    e=(a*c+b*d)/(c*c+d*d);
    return(e);
}

float divIm(float a,float b,float c,float d)#定义函数计算注入电压虚部
{
    float e;
    e=(b*c-a*d)/(c*c+d*d);
    return(e);
}

float mulRe(float a,float b,float c,float d)#定义函数计算平衡节点功率实部
{
    float e;
    e=a*c-b*d;
    return(e);
}

float mulIm(float a,float b,float c,float d)#定义函数计算平衡节点功率虚部

{
    float e;
    e=b*c+a*d;
```

```
    return(e);
}

float Max(float a[],int n)      #定义 max 函数，取最大值
{
    int i;
    float max;
    for(i=0;i<n-1;i++)
        if(a[i]>a[i+1])
            max=a[i];a[i]=a[i+1];a[i+1]=max;
    return(max);
}
```