MATPOWER

一种基于 **matlab** 的电力系统仿真组件

版本 **3.1b2**

2006-9-15

手

册

Ray D. Zimmerman ，Carlos E. Murillo-Sánchez，甘德强

中文翻译制作：中国电力市场论坛（www.dlscbbs.net）

目录

译者的话：

在做项目和实验的过程中，偶然使用到 MATPOWER 软件，发现该软件功能强大，但是操作还是比较的复杂，而 MATPOWER 本身的说明文档还没有中文版本，因此，译者产生了翻译用户手册的想法，促使了这个文档的诞生。

由于翻译时间仓促，这个版本几乎没有任何校正，还有大量的错误，随后会对原始版本进行修正，恳请各位网友将错误发送到 wolflove941@163.com 或者直接登陆中国电力市场论坛(www.dlscbbs.net)提出宝贵的意见。

1 绪论

什么是 MATPOWER？

MATPOWER 是一个基于 matlab m 文件的组建包，用来解决电力潮流和优化潮流的问题。它致力于为研究人员和教育从业者提供一种易于使用和可更新的仿真工具。Matpower 的设计理念是用尽可能简单、易懂，可更新的代码来实现最优秀的功能。MATPOWER 的主页为：
http://www.pserc.cornell.edu/matpower/

它从哪里来？

MATPOWER 是由卡奈尔大学电气学院电力系统工程研究中心的 RAY D. ZIMMENRman，CARLOS E.Murillo 和甘德强在 ROBERT THOMAS 的指导下开发出来的。最初的基于 MATLAB 的电力潮流和最优潮流代码是为 POWERWEB 项目的需要而编写的。
谁能够使用它？

- MATPOWER 是完全免费的，任何人都可以使用。
- 我们对 MATPOWER 的代码和作为特殊用途的可行性不作任何保障，授权与暗示。
- 任何使用 MATPOWER 的出版物都必须标注 MATPOWER http://www.pserc.cornell.edu/matpower/。
- 任何出于某种需要而对 MATLAB 进行的修改必须在适当的位置保留初始版权标志。
- MATPOWER 在没有书面许可的情况下不宜私自发布与转让。
- MATPOWER 改进版或源于 MATPOWER 的成果在没有书面许可的情况下不能私自转让或发布。

2 开始

2.1 系统要求

- MATLAB 5.0 或以上版本 [1]
- MATLAB 最优化工具箱（一小部分最优潮流算法需要）

两者都可以从 MathWorks 获得（见 http://www.mathworks.com/）。

2.2 安装

步骤一：到 MATPOWER 主页（http://www.pserc.cornell.edu/mathpower/）上按照下载指导下载。
步骤二：解压下载的文件。
步骤三：将解压后的文件放到 MATLAB 的 PATH 路径下。

2.3 执行电力常规潮流运算

运行一个简单的在文件 case9.m 中有详细的说明 9 节点牛顿潮流，包括默认的运算法则选项，以 matlab 的命令，输入：
>>runpf(‘case9’)

2.4 执行最优潮流程序

计算一个数据在 case30.m 文件中的 30 节点的最优潮流系统，以默认的算法选项，以 maitlab 的命令，输入：
>>runopf(‘case30’)
计算相同的系统，但是以关闭高耗机组处理的方式运行，输入：
>>runuopf(‘case30’)

2.5 获得帮助

当拥有 MATLAB 的内部函数和工具箱代码时，通过输入 help 加上命令或者 M－文件的名称可以获得详细的函数说明，几乎所有的 MATPOWER 的 M－文件都有这样的文档。比如，runopf 的帮助如下：
>> help runopf

```
 RUNOPF    Runs an optimal power flow.

    [baseMVA, bus, gen, gencost, branch, f, success, et] = ...
                runopf(casename, mpopt, fname, solvedcase)

    Runs an optimal power flow and optionally returns the solved values in
    the data matrices, the objective function value, a flag which is true if
    the algorithm was successful in finding a solution, and the elapsed time
    in seconds. All input arguments are optional. If casename is provided it
    specifies the name of the input data file or struct (see also 'help
    caseformat' and 'help loadcase') containing the opf data. The default
    value is 'case9'. If the mpopt is provided it overrides the default
    MATPOWER options vector and can be used to specify the solution
    algorithm and output options among other things (see 'help mpoption' for
    details). If the 3rd argument is given the pretty printed output will be
```

appended to the file whose name is given in fname. If solvedcase is
specified the solved case will be written to a case file in MATPOWER
format with the specified name. If solvedcase ends with '.mat' it saves
the case as a MAT-file otherwise it saves it as an M-file.

MATPOWER 还提供许多想想用来选择算法和输出，输入
>>help mpoption
更多信息详见 3.6 节：MATPOWER 的选项。

3 技术规则

3.1 数据文件格式

MATPOWER 所用的所有数据文件均为 MATLAB 的 M 文件或者 MAT 文件，他们用来定义和返回变量：baseMVA，bus，branch，gen，areas 和 gencost。变量 baseMVA 是标量，其他的都是矩阵。矩阵的每一行都对应于一个单一的母线，线路或者发电机组。列的数据类似于标准的 IEEE 和 PTI 列的数据格式。MATPOWER 案例文件的规范细节可以在 caseformat.m 中看到：

>> help runopf

RUNOPF Runs an optimal power flow.

[baseMVA, bus, gen, gencost, branch, f, success, et] = ...
runopf(casename, mpopt, fname, solvedcase)

Runs an optimal power flow and optionally returns the solved values in
the data matrices, the objective function value, a flag which is true if
the algorithm was successful in finding a solution, and the elapsed time
in seconds. All input arguments are optional. If casename is provided it
specifies the name of the input data file or struct (see also 'help
caseformat' and 'help loadcase') containing the opf data. The default
value is 'case9'. If the mpopt is provided it overrides the default
MATPOWER options vector and can be used to specify the solution
algorithm and output options among other things (see 'help mpoption' for
details). If the 3rd argument is given the pretty printed output will be
appended to the file whose name is given in fname. If solvedcase is
specified the solved case will be written to a case file in MATPOWER
format with the specified name. If solvedcase ends with '.mat' it saves
the case as a MAT-file otherwise it saves it as an M-file.

>> help ceseformat

ceseformat.m not found.

```
>> help caseformat

  CASEFORMAT    Defines the MATPOWER case file format.
    A MATPOWER case file is an M-file or MAT-file which defines the variables
    baseMVA, bus, gen, branch, areas, and gencost. With the exception of
    baseMVA, a scalar, each data variable is a matrix, where a row corresponds
    to a single bus, branch, gen, etc. The format of the data is similar to
    the PTI format described in
        http://www.ee.washington.edu/research/pstca/formats/pti.txt
    except where noted. An item marked with (+) indicates that it is included
    in this data but is not part of the PTI format. An item marked with (-) is
    one that is in the PTI format but is not included here. The columns for
    each data matrix are given below.

    See also IDX_BUS, IDX_BRCH, IDX_GEN, IDX_AREA and IDX_COST regarding
    constants which can be used as named column indices for the data matrices.
    Also described in the first three are additional columns that are added
    to the bus, branch and gen matrices by the power flow and OPF solvers.

    Bus Data Format
        1     bus number (1 to 29997)
        2     bus type
                  PQ bus          = 1
                  PV bus          = 2
                  reference bus   = 3
                  isolated bus    = 4
        3     Pd, real power demand (MW)
        4     Qd, reactive power demand (MVAr)
        5     Gs, shunt conductance (MW (demanded) at V = 1.0 p.u.)
        6     Bs, shunt susceptance (MVAr (injected) at V = 1.0 p.u.)
        7     area number, 1-100
        8     Vm, voltage magnitude (p.u.)
        9     Va, voltage angle (degrees)
    (-)       (bus name)
        10    baseKV, base voltage (kV)
        11    zone, loss zone (1-999)
    (+) 12    maxVm, maximum voltage magnitude (p.u.)
    (+) 13    minVm, minimum voltage magnitude (p.u.)

    Generator Data Format
        1     bus number
    (-)       (machine identifier, 0-9, A-Z)
        2     Pg, real power output (MW)
```

3 Qg, reactive power output (MVAr)
4 Qmax, maximum reactive power output (MVAr)
5 Qmin, minimum reactive power output (MVAr)
6 Vg, voltage magnitude setpoint (p.u.)
(-) (remote controlled bus index)
7 mBase, total MVA base of this machine, defaults to baseMVA
(-) (machine impedance, p.u. on mBase)
(-) (step up transformer impedance, p.u. on mBase)
(-) (step up transformer off nominal turns ratio)
8 status, > 0 - machine in service
<= 0 - machine out of service
(-) (% of total VAr's to come from this gen in order to hold V at
remote bus controlled by several generators)
9 Pmax, maximum real power output (MW)
10 Pmin, minimum real power output (MW)

Branch Data Format
1 f, from bus number
2 t, to bus number
(-) (circuit identifier)
3 r, resistance (p.u.)
4 x, reactance (p.u.)
5 b, total line charging susceptance (p.u.)
6 rateA, MVA rating A (long term rating)
7 rateB, MVA rating B (short term rating)
8 rateC, MVA rating C (emergency rating)
9 ratio, transformer off nominal turns ratio (= 0 for lines)
(taps at 'from' bus, impedance at 'to' bus, i.e. ratio = Vf / Vt)
10 angle, transformer phase shift angle (degrees)
(-) (Gf, shunt conductance at from bus p.u.)
(-) (Bf, shunt susceptance at from bus p.u.)
(-) (Gt, shunt conductance at to bus p.u.)
(-) (Bt, shunt susceptance at to bus p.u.)
11 initial branch status, 1 - in service, 0 - out of service

(+) Area Data Format
1 i, area number
2 price_ref_bus, reference bus for that area

(+) Generator Cost Data Format
NOTE: If gen has n rows, then the first n rows of gencost contain
the cost for active power produced by the corresponding generators.
If gencost has 2*n rows then rows n+1 to 2*n contain the reactive
power costs in the same format.

1 model, 1 - piecewise linear, 2 - polynomial
2 startup, startup cost in US dollars
3 shutdown, shutdown cost in US dollars
4 n, number of cost coefficients to follow for polynomial
cost function, or number of data points for piecewise linear
5 and following, cost data defining total cost function
For polynomial cost:
c2, c1, c0
where the polynomial is c0 + c1*P + c2*P^2
For piecewise linear cost:
x0, y0, x1, y1, x2, y2, ...
where x0 < x1 < x2 < ... and the points (x0,y0), (x1,y1),
(x2,y2), ... are the end- and break-points of the cost function.

某些列被加入到了母线，线路和发电机组矩阵当中，通过查看 idx_bus,idx_brch 和 idx_gen 可以获得更多细节。

3.2 模型

交流模型（AC）

固定负荷被当作恒定有功和无功功率注入，P_d 和 Q_d 分别被指定为 bus 矩阵的第三列和第四列。任何母线的恒阻抗泄漏元件的泄漏导纳都通过 Gsh 和 Bsh 被指定到第五和第六列。

$$Y_{sh} = \frac{G_{sh} + jB_{sh}}{baseMVA}$$

所有的线路，包括输电线路、变压器和调相机，都通过标准的“p”模型建立包括串联电阻 R 和电抗 X 以及所有的线路充电电容 Bc，和理想的变压器串联，对于调相机包括调节比例t 移相角q_{shift}。参数 R，X，B，t 和q_{shift}，在线路矩阵 branch 的第 3，4，5，9 和 10 列。线路首端和末端的电压和电流通过线路导纳矩阵 Ybr 通过以下公式相关联：

$$\begin{bmatrix} I_f \\ I_t \end{bmatrix} = Y_{br} \begin{bmatrix} V_f \\ V_t \end{bmatrix} \qquad (1)$$

其中：
$$Y_{br} = \begin{bmatrix} \left(Y_s + j\frac{B_c}{2}\right)\frac{1}{t^2} & -Y_s\frac{1}{t\,e^{jq_{shift}}} \\ -Ys\frac{1}{t\,e^{-jq_{shift}}} & Y_s + j\frac{B_c}{2} \end{bmatrix}$$
，$Y_s = \frac{1}{R + jX}$ 。

分散的线路导纳矩阵和母线泄漏导纳矩阵被 MATPOWER 联合成为复合母线导纳矩阵 Y_{bus}, 并以它来关联母线电压向量 V_{bus} 和母线电流向量 I_{bus}：

$$I_{bus} = Y_{bus} V_{bus}$$

类似的，通过形成导纳矩阵 Y_f 和 Y_t 来计算线路首末端的电流向量，在给定母线电压 V_{bus} 的情况下：

$$I_f = Y_f V_{bus}$$

$$I_t = Y_t V_{bus}$$

母线功率注入和线路功率注入的复向量可以表达为：

$$S_{bus} = diag(V_{bus}) I_{bus}^*$$

$$S_f = diag(V_f) I_f^*$$

$$S_t = diag(V_t) I_t^*$$

其中 Vf 和 Vt 分别为所有支路首末端电压的复向量，diag（）将一个向量转变为一个以它为对角元素的对角矩阵。

直流公式（DC）

对直流模型来说，仍然是使用原来的参数，并且做了以下的假设：

- 线路阻抗 R 和充电电容 Bc 被忽略（也就是说支路是无损的）
- 所有的母线电压都认为接近与标么值 1
- 电压角相差很小，认为 $\sin(q_{ij}) \approx q_{ij}$

联合这些假设和方程 1 以及考虑 $S = VI^*$，有功潮流和电压相角之间的关系可以写为：

$$\begin{bmatrix} P_f \\ P_t \end{bmatrix} = B_{br} \begin{bmatrix} q_f \\ q_t \end{bmatrix} + \begin{bmatrix} P_{f,shift} \\ P_{t,shift} \end{bmatrix} \qquad (2)$$

其中，

$$B_{br} = \frac{1}{X_t} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \qquad (3)$$

$$\begin{bmatrix} P_{f,shift} \\ P_{t,shift} \end{bmatrix} = \frac{1}{X_t} \begin{bmatrix} 1 \\ -1 \end{bmatrix} \qquad (4)$$

分离的支路的调整注入和 Bbr 矩阵之间的原理在 MATPOWER 中通过生成一个母线 B_{bus} 矩阵和 $P_{bus,\ shift}$ 注入向量来处理，它用来在一直电压相角的情况下计算母线的有功功率：

$$P_{bus} = B_{bus} q_{bus} + P_{bus,shift}$$

类似的，MATPOWER 建立矩阵 Bf 和向量 Pf，shift 用来计算支路功率注入向量 Pf 和 Pt：

$$P_f = B_f q_{bus} + P_{f,shift}$$

$$Pt = -P_f$$

3.3 电力潮流

MATPOWER 拥有 5 种潮流计算方法，他们可以通过 runpf 来调用。除了可以输出到屏幕之外（作为默认方式），runpf 还可以有以下的返回选项可以选择参数来输出解：

```
>> [baseMVA,bus,gen,branch,success,et]=runpf(casename);
```

这些解的值被存储在以下的结果中：

bus(:,VM)	bus voltage magnitudes(母线电压幅值)
bus(:,VA)	bus voltage angles （母线电压相角）
gen(:,PG)	generator real power injections(发电机有功输入)
gen(:,QG)	generator reactive power injections(发电机无功输入)
branch(:,PF)	real power injected into “from” end of branch（支路首端的有功输入）
branch(:,PT)	real power injected into “to” end of branch（支路末端的有功输入）
branch(:,QF)	reactive power injected into “from” end of branch（支路首端的无功输入）
branch(:,QT)	reactive power injected into “to” end of branch（支路末端的无功输入）
success	1=solved successfully,0=unable to solve(1 表示计算成功，0 表示失败)
et	computation time required for solution(计算所用时间)

默认的潮流计算方法是标准的潮流法[12]，采用全雅克比矩阵，迭代求解。这种方法在许多文教科书中都有提到。法则 2 和法则 3 是改进型快速解耦算法[10].MATPOWER 采用 XB 和 BX 变换，参见文献[1].法则 4 是标准的高斯－赛德尔方法[5],基于意大利 Bologna 大学的 Alberto Borhetti 的贡献的代码开发。要使用出默认的牛顿法之外的其他算法，PF_ALG 选项必须正确的设置。比如说，要使用 XB 快速解耦算法：

```
>> mpopt= mpoption('PF_ALG',2);
>> runpf(casename,mpopt);
```

最后一种算法是直流潮流算法[13],它的使用是通过设置 PF_DC 为 1，运行 runpf 而进行的，或者直接使用 rundcpf。直流潮流的计算是通过直接的不迭代的的方法解母线电压相角和指定母线的有功注入获得，基于方程 2，3 和 4。

对于交流潮流解法，如果 ENFORCE_Q_LIMS 选项被设为 true（默认为 false），并且运行过程中有任何发电机组的无功越限，相应的母线被转换为 PQ 母线（节点），将无功出力设定在限制值，并且案例重新计算。该母线的电压幅值为满足无功限制的要求将偏离指定值。如果参考母线（节点）的有功出力达到限制值，该节点将自动转化为 PQ 母线（节点），在下一轮迭代中第一个依然存在的 PV 母线（节点）将被当作松弛母线（节点），这将导致该母线（节点）的机组有功出力稍微偏离指定值。

通常，没有 MATPOWER 的潮流解法中不包含变压器分接头的改变或者操作，或者部分系统从网络中解列等。

潮流计算的解法，除了高斯－赛德尔法之外，都可以很好的解决甚至是大规模网络，因为这些算法和计算充分利用了 MATLAB 的内部稀疏矩阵处理。

3.4 最优潮流

MATPOWER 提供多种解算最优潮流问题（OPF）的方法，可以通过访问函数 runopf 的方法实现。除了提供将计算结果输出到屏幕之外（默认），runopf 函数还可以通过设置以下的参数返回解到其他地方。

```
>>[baseMVA,bus,gen,gencost,branch,f,success,et]=runopf(casename);
```

除了最优潮流解法之外，OPF 的运算还包括一下的值：

bus(:,LAM_P)	母线（节点）的有功失配拉格朗日乘子
bus(: , LAM_Q)	母线（节点）的无功失配拉格朗日乘子
bus(:,MU_VMAX)	母线（节点）的电压上限龙格－库塔乘子
bus(:,MU_VMIN)	母线（节点）的电压下限龙格－库塔乘子
gen(:,MU_PMAX)	发电机组有功出力上限的龙格－库塔乘子
gen(:,MU_QMAX)	发电机组无功出力上限的龙格－库塔乘子
gen(:,MU_PMIN)	发电机组有功出力下限的龙格－库塔乘子
gen(:,MU_QMIN)	发电机组无功出力下限的龙格－库塔乘子
branch（:,MU_SF）	支路首端的潮流限制龙格－库塔乘子
branch(:,MU_ST)	支路末端的潮流限制龙格－库塔乘子
f	最后的目标函数值

MATPOWER 的第一个（按年代顺序排列）OPF 解法是基于包含在 MATLAB 早期版本最优化工具箱中的 constr 函数，它成功的使用了一种二次规划技术，对海森矩阵采用了拟牛顿逼近法。第二种方法基于线性规划。它可以使用最优化工具箱中的 LP（线性规划）解法或者其他可以获得的 MATLAB LP 解法。MATPOWER3.0 版提供一种通用的 OPF 模型，允许一般的线性约束加入到最优化变量中，但是要求能够获得 MATLAB 最优化工具箱 2.0 或以上版本中的 fmincom.m 文件，或者 最优化 MINOPF 包中的基于 MINOS 的 MEX 文件（见 http://www.pserc.cornell.edu.minopf/）。MINOPF 被独立的发布出来是因为它比 MATPOWER 有更加严格的授权要求。

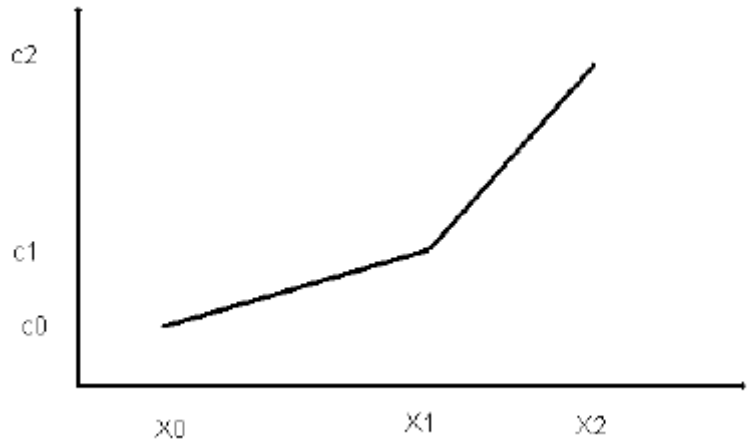
MATPOWER 的性能取决于许多因素。首先，constr 算法使用了一种没有稀疏化的方法，所以它只能针对小型的电力系统。对于需要使用 fmincon 函数的参数组合也一样。对于基于线性规划的算法，从另一个方面来说，采用了稀疏算法，但是低版本最优化工具箱中的线性规划算法并没有采用这种稀疏算法。事实上，基于低版本工具箱中的 LP 算法效率还不如基于 constr 的算法，甚至对于小系统来说。幸运的是还有采用了稀疏算法的第三方解决方法可以得到。通常这些都能大幅度的提高算法的效率。甚至对于基于 constr 或者 fmincon 的方法来说，当采用了满矩阵数据的 BPMPD 而不是以往的 qp.m 文件时，速度也会快得多。

然而，必须指出的是，即使是采用了优秀的线性算法，MATPOWER 的基于 LP 的方法，并不像它的潮流算法一样可以适用非常大的系统。实质性的改进还是需要进行的，他们也许需要更加复杂的代码，甚至于是特定的 LP 解法。然而，当速度是最基本的要求时，当可以获得使用授权时，我们更加推荐基于 MINOS 的 MEX 文件方法。它以 FORTRAN 语言编程并且使用一种优化结构计算需要的雅克比矩阵的值，这种优化结构使用了被 MINOS 采用的压缩列稀疏格式。事实上，MATPOWER3.0 以上的新的 OPF 算法从 MINOS 的数据格式中获得了一些灵感。

MATPOWER 的 OPF 算法目前还不能处理非互连系统或者系统解列的问题。

可变成本约束的分段线性化成本

MATPOWER 的 OPF 算法允许使用凸分段线性成本函数来描述发电机组的有功和无功出力。类似的成本曲线如下例所示：



这个非连续可微的成本函数在建模时对每一个类似的成本曲线都采用了一种额外的附加的成本变量以及对该变量以及 Pg 的额外约束，对每一个分段都一样。这些约束建立起一个凸“盆”等价于成本变量位于成本曲线之上。当成本最小化时，成本变量将远离这个“盆”。如果这个附加成本变量是 y，那么该机组的成本对总成本的贡献也就是 y。在以上的例子中，这两个附加的约束条件是：

1） $y \geq m_1(\mathrm{P}_g - x_0) + c_0$ y 必须位于第一段之上

2） $y \geq m_2(\mathrm{P}_g - x_1) + c_1$ y 必须位于第二段之上

m1 和 m2 分别是两段的斜率。当然，机组出力的上下限约束是必不可少的：

$P_g : P_{\min} \leq P_g \leq P_{\max}$ 。该机组的附加约束就是 y。

可变成本约束（CCV）算法在 MATPOWER 中用来处理所有的分段线性成本函数。

传统的交流 OPF 方程

MATPOWER 中的交流最优潮流解法用来解决没有不连续控制变量和状态变量的“平滑”最优潮流。目标函数为有功和（或）无功出力的总成本。这些成本可能被定义为机组出力的多项式或者分段线性函数。问题建模方程如下：

$$\min_{q,V,P_g,Q_g} \sum_i f_{1i}(P_{gi}) + f_{2i}(Q_{gi})$$

s.t.

$P_i(q,V) - P_{gi} + P_{di} = 0$ 有功平衡约束

$Q_i(q,V) - Q_{gi} + Q_{di} = 0$ 无功平衡约束

$\left|S_{ij}^f(q,V)\right| \leq S_{ij}^{\max}$ 线路首端传输视在功率约束

$\left|S_{ij}^{t}(q,V)\right| \le S_{ij}^{\max}$ 线路末端传输视在功率约束

$V_i^{\min} \le V_i \le V_i^{\max}$ 母线电压约束

$P_i^{\min} \le P_i \le P_i^{\max}$ 机组有功出力约束

$Q_i^{\min} \le Q_i \le Q_i^{\max}$ 机组或者无功设备无功出力约束

f_{1i} 和 f_{2i} 分别为在一个给定的出力上有功和无功机组的成本，可以通过多项式函数或者分段线性函数获得。定义 x 为：

$$x = \begin{bmatrix} q \\ V \\ P_g \\ Q_g \end{bmatrix}$$

这个问题可以被精确的通过以下的模型表示：

$$\min_x f(x)$$

Subject to

$g_1(x) = 0$ 功率平衡约束

$g_2(x) = 0$ 支路潮流约束

$x_{\min} \le x \le x_{\max}$ 变量不等式约束

基于最优化工具箱的 OPF 解法（constr）

MATPOWER 的前两种最优潮流解法都是基于 MATLAB 最优化工具箱中的 constr 非线性约束优化函数的。Constr 函数和它的理论在老版本中都有说明[6]。MATPOWER 在优化时以两个 m 文件的形式提供 constr。一个用来计算在给定的状态变量 x（出力，电压，潮流）下计算目标函数 f 和约束 g，另外一个用来计算他们的偏微分 $\partial f / \partial x$ 和 $\partial g / \partial x$。

MATPOWER 对这些 m 文件提供两个版本。一个用来解决多项式成本函数的系统。在这种算法中，成本函数通过直接的方法引入到目标函数中。另外一个是用来解决分段线性成本半数对应的系统问题。分段线性成本函数是通过在每段的线性函数加入一个成本变量来处理的。目标函数就是简单的各个成本变量的代数和，这些变量必须位于组成分段线性函数的各个段函数的上方。很明显，这个方法只能处理凸函数。在 MATPOWER 的文档中，他被用来处理变动成本约束方程。

算法代码 100 和 200 分别用来标志是基于 constr 的多项式成本函数解法还是分段线性成本函数的解法，成本函数被近似的处理为分段线性函数通过将多项式的系数设定为固定的值，通过检测选项向量来得到。（见 3.6 节选项细节）

需要强调的是，基于 constr 的方法同样利用从超定 QP 解法，比如说 bpmpd。在附录 A 中

有更多关于 LP 和 QP 解法的信息。

基于线性规划的 OPF 解法（LPconstr）

基于线性规划的 OPF 方法在今天已经在工业界获得了广泛的应用。但是，MATPOWER 中的基于 LP 的算法远远比工业应用软件中的算法要简单得多。

MATPOWER 基于 LP 的算法和基于 constr 的算法采用相同的模型，包括 CCV（变动成本约束）的分段线性成本案例。OPF 的简化模型可以写成分离的等式约束和不等于约束 g，以及分离的变量（包括状态变量和控制变量），如下：

$$\min_x f(x_2)$$

Subject to

$g_1(x_1, x_2) = 0$ 等式约束

$g_2(x_1, x_2) \le 0$ 不等式约束

x_1 表示系统的电压幅值和相角，x_2 表示系统的机组哟嘎和无功出力（以及相应的 CCV 模型对应的成本变量）。这是一个典型的非线性规划问题，假定当 x_2 给定时等式约束可以用来解出下 x_1。

基于 LP 的 OPF 解法依托函数 LPconstr 执行，她类似于 constr 函数，且使用相同的 M 文件来计算目标函数，约束条件以及各自的斜率。除此之外，在给定 x_2 后，需要使用另外一个 m 文件（lpeqslvr.m）来计算约束条件的解 x_1。

该算法的计算过程如下所示，上标表示迭代次数：

第 0 步：设定迭代计数器 k＝0，选择一个合适的初值 x_2^0。

第 1 部：解等式约束方程 $g_1(x_1^k, x_2^k) = 0$，得到解 x_1^k。

第 2 步：在 x_1^k 附近将问题线性化，对 Δx 进行线性规划求解。

$$\min_{\Delta x} \left[\frac{\partial f}{\partial x}\Big|_{x=x^k} \right] \mathbf{g} \Delta x$$

subject to

$$\left[\frac{\partial g}{\partial x}\Big|_{x=x^k} \right] \mathbf{g} \Delta x \le -g\left(x^k\right)$$

$$-\Delta \le \Delta x \le \Delta$$

第 3 步 取 k＝k＋1，更新当前解 $x^k = x^k - 1 + \Delta x$ 。

第 4 步 如果 x^k 越界，停止，否则进入第 5 步。

第 5 步 基于置信区间[3]调整步长限制Δ返回第 2 步。

边界条件描述如下：

$$\frac{\partial L}{\partial x} = \frac{\partial f}{\partial x} + l^T \cdot \frac{\partial g}{\partial x} \le tolerance_1$$

$$g(x) \le tolerance_2$$

$$\Delta x \le tolerance_3$$

l 是线性规划问题的拉格朗日乘子向量。第一个条件和斜率的大小有关，第二个和约束条件有关，第三个和步长有关。更加详细的资料可以参见文献【4】。

非常多的情况下，步骤 1 计算出来的 x^k 是不可行的，他会导致线性问题规划无解。在这种情况下，采用对每一个越界约束都加上一个松弛变量。这些松弛变量在最优点处必须为零。

函数 LPconstr 执行一下三种方法：

- 对所有的不等式约束进行稀疏处理
- 用松弛约束进行稀疏处理（ICS，迭代约束搜索）
- 用松弛约束进行密集处理（ICS）【11】

这三种方法对采用多项式成本的系统分别采用算法程序 160，140 和 120，而对于采用分段线性成本的系统分别为 260，240 和 220。因为采用基于 constr 的方法，对于采用 200－300 之间的算法都会导致多项式成本函数转化为分段线性逼近。

对于密集型公式，x_1 中的某些变量和等式约束 g_1 中的某些约束在开始线性规划的子问题时就被排除了。这个处理过程在下面将详细描述，假设线性规划子问题如下：

$$\min c^T \cdot \Delta x$$

Subject to

$$A \cdot \Delta x \le b$$
$$-\Delta \le \Delta x \le \Delta$$

如果他被重新描述为：

$$\min c_1^T \cdot \Delta x_1 + c_2^T \cdot \Delta x_2$$

Subject to

$$A_{11} \cdot \Delta x_1 + A_{12} \cdot \Delta x_2 = b_1$$
$$A_{21} \cdot \Delta x_1 + A_{22} \cdot \Delta x_2 \le b_2$$
$$-\Delta \le \Delta x \le \Delta$$

其中 A_{11} 为方阵， Δx_1 可以通过下式计算：

$$\Delta x_1 = A_{11}^{-1}(b_1 - A_{12}\Delta x_2)$$

带回带原规划中，可以得到新的线性规划问题：

$$\min(-c_1^T \cdot A_{11}^{-1}A_{12} + c_2^T) \cdot \Delta x_2$$

Subject to

$$A_{11} \cdot \Delta x_1 + A_{12} \cdot \Delta x_2 = b_1$$
$$A_{21} \cdot A_{11}^{-1}(b_1 - A_{12}\Delta x_2) + A_{22} \cdot \Delta x_2 \le b_2$$

$$-\Delta_1 \le A_{11}^{-1}(b_1 - A12\Delta x_2) \le \Delta_1$$

$$-\Delta_2 \le \Delta x_2 \le \Delta_2$$

这个新的 LP 问题规模比原来的要小很多，但是却不再稀疏。

因此，要真正实现基于 LP 的最优潮流解法，必须要获得好的线性规划算法，比如 bpmpd。具体细节见附录 A。

3.4.2 广义交流最优潮流解法

和经典的 MATPOWER 传统方法相比，采用 fmincon 和 MINOPF 算法的广义交流最优潮流解法有一些特殊的优点：

- 可以混合多项式和分段线性成本函数
- 负荷调度
- 机组 P－Q 容量曲线
- 支路相角差限制
- 附加用户供应线性约束
- 附加用户供应成本

MATPOWER3.1 提供了新的广义用户供应成本公式，机组容量曲线、支路角差限制和简单的线性约束在 3.0 版本中就已经提供。

该问题从三组优化变量来进行处理，分别是 x,y 和 z。x 变量是 OPF 的变量，包括每条母线（节点）的电压相角q 和幅值V，以及机组的有功和无功注入P_g 和Q_g。y 变量包括分段线性函数 CCV 所使用的附加变量。其他的附加用户自定义变量被规为 z。

该最优化问题可以描述为：

$$\min_{x,y,z} \sum_i (f_{1i}(P_{gi}) + f_{1i}(Q_{gi})) + \sum_i y_i + \frac{1}{2}w^T Hw + C_w^T w$$

Subject to

$$g_p(x) = P(q,V) - P_g + P_d = 0 \qquad \text{有功平衡方程}$$

$$g_Q(x) = Q(q,V) - Q_g + Q_d = 0 \qquad \text{无功平衡方程}$$

$$g_{S_f}(x) = \left|S_f(q,V) - S_{\max}\right| \le 0 \qquad \text{支路首端最大视在功率约束}$$

$$g_{S_t}(x) = \left|S_t(q,V) - S_{\max}\right| \le 0 \qquad \text{支路末端最大视在功率约束}$$

$$l \le \begin{bmatrix} x \\ y \\ z \end{bmatrix} \le u \qquad \text{广义线性约束}$$

$$x_{\min} \le x \le x_{\max} \qquad \text{电压和机组变量限制}$$

对传统的 OPF 模型的最大改进在于广义成本包含 w 以及在通用线性约束中包含了矩阵 A，在一下两节中将会予以描述。这两个结构使得在建模系统时有很大的适应性，从而使 MATPOWR 更加适合于作为研究的工具。

申明：在最优化工具箱 3.0 及之前的版本当中，fmincon 好像不能正确提供约束的影子价格。而这些并不会发生在 constr 中，这也许是这些版本的最优化工具箱的一个 bug。

通用线性约束（一般线性约束）

在标准的节点功率平衡非线性等式约束和支路潮流非线性不等式约束之外，这种模型加入了附加线性约束的框架，包括所有的优化变量。

$$l \le A \begin{bmatrix} x \\ y \\ z \end{bmatrix} \le u \qquad \text{通用线性约束}$$

该线性约束中的部分是由使用者直接提供的，而另外一部分是在案例数据中自动生成的。自动生成的部分包括：

- 机组 P－Q 容量曲线的约束对应的行
- 可调度或者价格敏感负荷的恒定功率因素约束对应的行
- 支路两段相角差限制对应的行
- 机组分段线性成本函数的 CCV 产生的 y 变量以及与他相关的约束对应的行与列

除了这些自动产生的约束之外，用户还可以自己加入矩阵 A_u 以及向量 l_u 和 u_u 来定义更多的线性约束。这些用户子定义的约束可以用来提供一些额外的约束，比如说，特定母线之间的电压相角差的严格约束。矩阵 A_u 必须有至少 n_x 列，n_x 为 x 变量的个数。当 A_u 的列数大于 n_x 时，一个相同的 z 优化变量将会被加入到各个附加的列中。这些 z 变量同样也会加入到通用的成本费用中，描述如下，所以 A_u 和 N 必须有相同的列数。

$$l_u \le A_u \begin{bmatrix} x \\ z \end{bmatrix} \le u_u \qquad \text{用户加入的线性约束}$$

对 MATPOWER3.0 的改进：用户自定义矩阵 A_u 不再包含由机组分段线性成本函数的 y 变量所产生的全零列。这将大大的简化 A_u 矩阵的生成。

通用成本函数

成本函数由 3 部分组成。前两部分分别是多项式和分段线性成本。每个机组的有功出力和无功出力对应的多项式或者分段线性成本被指定到了 gencost 矩阵的特定行上。任何分段线性成本都通过使用 CCV 模型中的 y 辅助变量执行。通用模型允许在同一个系统中混合使用多项式和分段线性成本函数。

第三部分提供了一个通用的框架来在优化变量中加入实现某些功能的附加成本，比如说使用惩罚函数来作为电压的软约束，以及一些约束对应的变量通过拉格朗日松弛方法处理的附加成本，等等。

这些通用的成本费用通过参数 H，Cw，N 和 fparm（描述如下）的集合指定。它由一个 $n_w \times 1$ 向量 w 的变换后的优化变量的通用二次函数组成。

$$\frac{1}{2} w^T H w + C_w^T w$$

H 是一个 $n_w \times n_w$ 的对称稀疏矩阵，作为二次项系数，C_w 是一个 $n_w \times 1$ 的向量作为一次项系数。稀疏矩阵 N 是一个 $n_w \times n_{xz}$ 阶矩阵，它的列数必须和用户自定义矩阵 A_u 匹配。f_{parm} 是一个 $n_w \times 4$ 阶矩阵，它的四列分别为：

$$f_{parm} = \begin{bmatrix} d & \hat{r} & h & m \end{bmatrix}$$

向量 w 由 x 和 z 优化变量在第一次通用线性变换时产生。

$$r = N \begin{bmatrix} x \\ z \end{bmatrix},$$

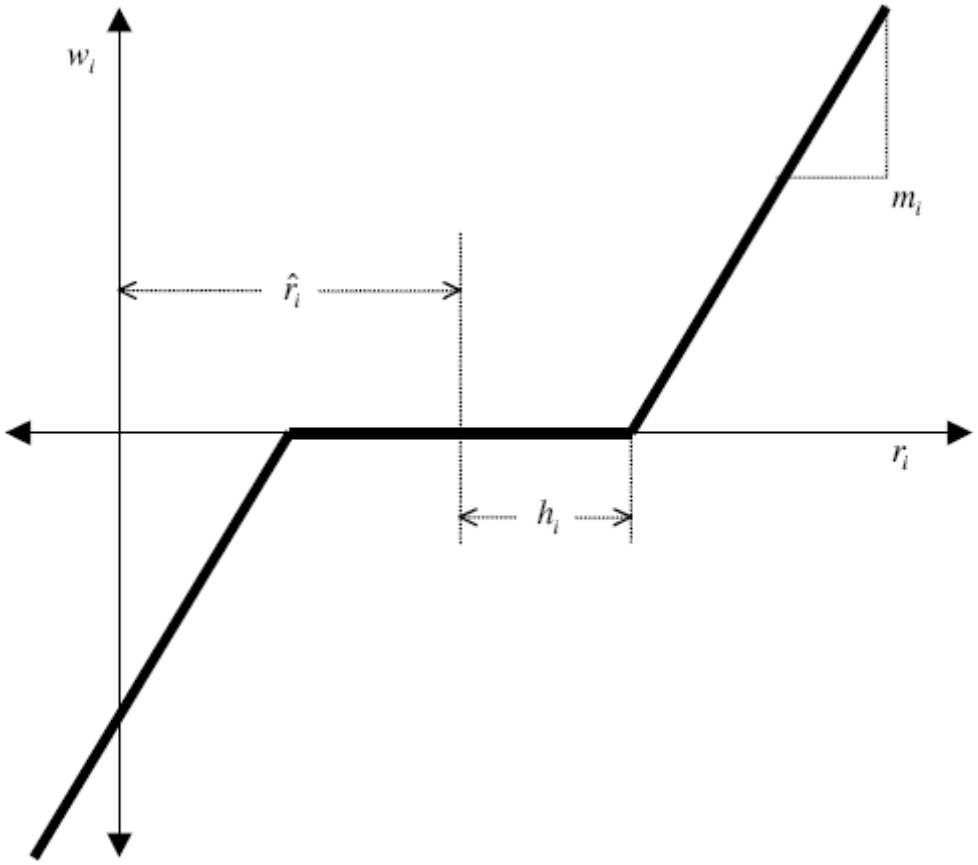
紧随着一个移动“死区”的梯度函数，由剩下的 f_{parm} 参数所定义。r 的每一个元素都通过下式转换为 w 中合适的元素：

$$w_i = \begin{cases} m_i \cdot f_i\left(r_i - \hat{r}_i + h_i\right), & r_i - \hat{r}_i < -h_i \\ 0, & -h_i \le r_i - \hat{r}_i \le h_i \\ m_i \cdot f_i(r_i - \hat{r}_i - h_i), & r_i - \hat{r}_i \ge h_i \end{cases}$$

函数 f_i 是一个由 d_i 中的标志选择的先决函数。当前的式子中包含一次和二次项。

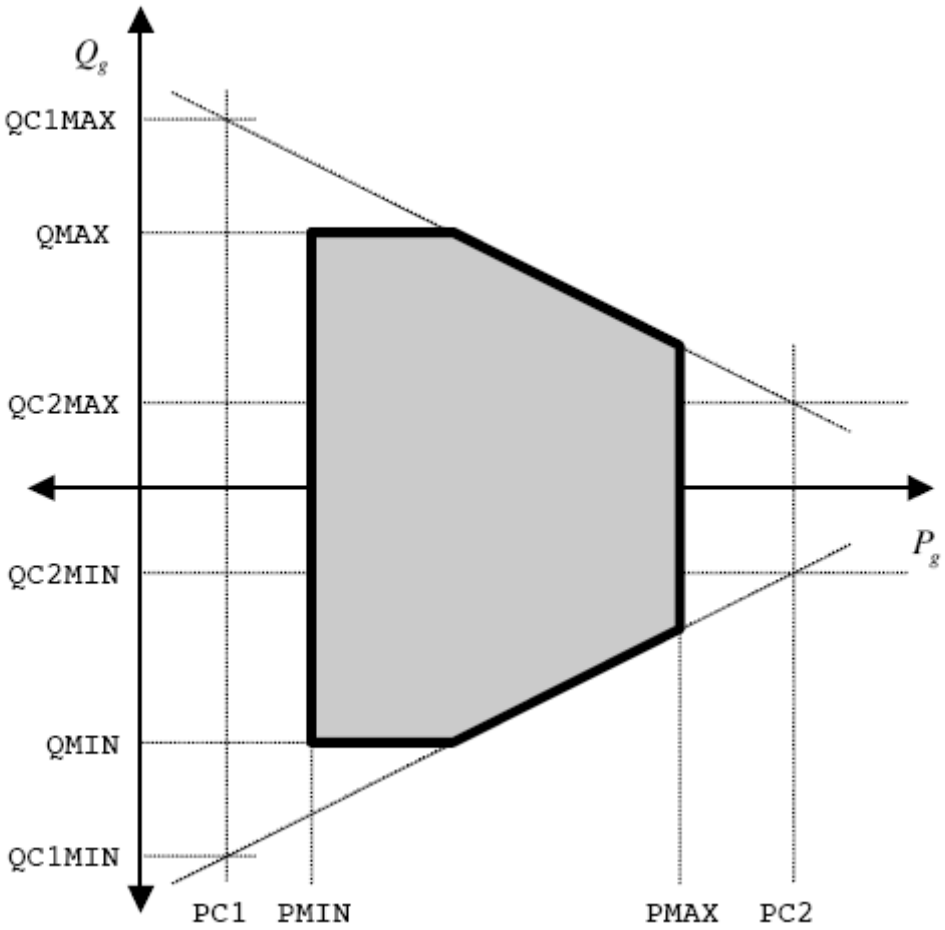
$$f_i(t) = \begin{cases} t, & d_i = 1 \\ t^2, & d_i = 2 \end{cases}$$

当 $d_i = 1$ 时，线性情况如下图所示，其中 w_i 随着 r_i 的增大由 $\hat{r}_i$ 开始，紧接着是由 h_i 决定的“死区”，最后是由 m_i 作为斜率的段。



通用 P－Q 容量曲线

传统的交流 OPF 模型建模时将机组的 P－Q 容量曲线简单的处理为由 gen 矩阵中 PMIN，PMAX，QMIN，QMAX 列定义的类似于方形的约束。在 MATPOWER3.1 中，案例文件格式的版本 2 在 gen 矩阵中提供 6 个新的列，用来指定斜率在容量曲线之上或者之下的部分。这些列分别是：PC1，PC2，QC1MIN，QC1MAX，QC2MIN 和 QC2MAX。这种更加通用的容量曲线的可行区域可以由下图的阴影部分来表示：

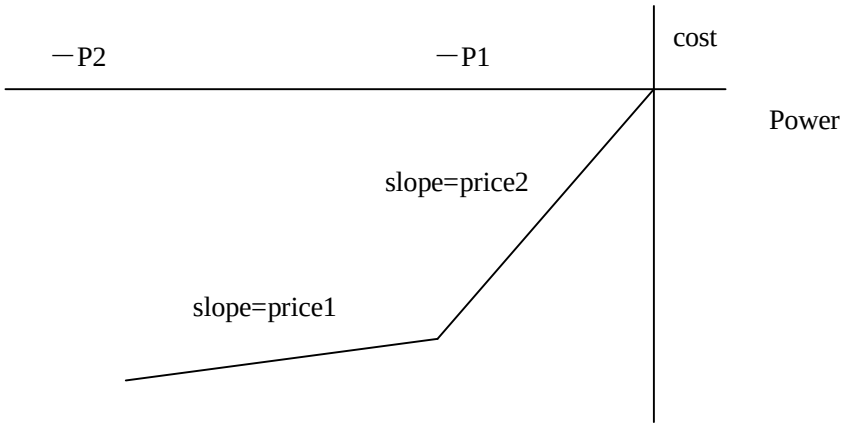


PC1 和 PC2 的值并不太重要，有时候为了方便甚至设定为和 PMIN 及 PMAX 一样。重要是要对 QCnMAX（QCnMIN）取适当的限制值，因为它们的两个交点决定了容量曲线之上（下）的部分。

可调度负荷

通常，可调度或者价格敏感负荷都可以建模成负的带有相关成本的有功注入。当前的想法是：如果一个机组的 $PMIN < PMAX = 0$，那么它确实是一个可以调度的负荷。而如果一个负荷的（价格）需求曲线如下所示

从上图可以看出，当价格高于 *price*2 时，该负荷将不会消费电能。当价格高于 *price*1 但是低于 *price*2 时将消费 *P*1，当价格等于或低于 *price*1 时将消费 *P*2 的电力。作为负的功率注入考虑，调度情况是：当价格高于 *price*2 时，注入为 0。当价格高于 *price*1 但是低于 *price*2 时将注入－ *P*1，当价格等于或低于 *price*1 时将注入－ *P*2 。这种想法可以通过一下的分段线性成本函数来描述：



注意到认为需求可以分开调度或者说是可以被“分裂”的；如果价格刺激能达到需求的一半时，那么负荷必须接受分块的结果。否则，接受或者拒绝整个块将引起一个复杂的混合整数问题，这时目前 MATPOWER 模型中没有解决的问题。

当有可调度负荷存在时，无功调度问题将增加。当“负机组”的 QMIN/QMAX 出力限制不是设置为 0 时，算法将匹配无功出力到最适合的值。既然这不是正常的负荷行为，在通用算法中我们认为可调度负荷维持一个恒定的功率因素。为了保持比例因子 Pg 和 Qg 的恒定，加入附加通用线性约束的机制被用来自动处理这些注入的限制。这些比例因子由表 gen 中 PMIN 和 QMIN（感性负荷）或者 QMAX（容性负荷）推出。因此，将他们正确设置是非常重要的，必须要牢记的是 PG 是一个负数并且对于普通的感性负荷 QG 也应当是负数(一

个正的无功负荷是一个负的无功注入）。gen 矩阵的 PG 和 QG 列的初值必须考虑 PMIN 和合适的 Q 限制所决定的比例因子。

支路相角差限制

支路矩阵的 ANGMIN 和 ANGMAX 列的第 k 行可能被指定在特定的值，对应的为支路始端电压相角q_f和末端电压相角q_t之间的差值。

$$branch(k, ANGMIN) \le q_f - q_t \le branch(k, ANGMAX)$$

这里采用角度表示，并且将 0^o 或者 360^o 认为是 ANGMAX（ANGMIN）没有限制。对应约束的龙格－库塔乘子将返回到 MU_ANGMIN 和 MU_ANGMAX 列。当通过 mpotion 设置 OPF_IGNORE_ANG_LIM 选项为真（TRUE）时，这些支路相角差约束将被忽略。

问题数据转换

为附加线性约束定义一个用户提供矩阵 A 需要掌握 x 向量中优化变量的顺序的知识。这需要理解在问题解决之前对输入数据（bus,gen,branch,areas 和 gencost 表）执行的标准转换。所有的这些数据的转换是在问题解决之后，所以输出数据已经正确地存在于表中。

第一步是将未运行的机组和支路剔除，为数据输出初始化表。

```
comgen = find(gen(:,GEN_STATUS) > 0);%找出在运行机组
onbranch = find(branch(:,BR_STATURS) ~= =);%找出在运行支路
gen = gen(comgen,:);
branch = branch(onbranch,:);
```

第二步是重新编号 bus 表中的母线编号，这样结果表中就能包含从 1 开始连续编号的母线。

```
[ i2e,bus,gen,branch,areas] = ext2int(bus,gen,branch,areas);
```

其中 i2e 是用来存储最后重构的逆。最后，利用母线编号机组被进一步重新排序。

```
ng = size(gen,1);  % 机组或者注入的数量
[tmp,igen] = sort(gen(:,GEN_BUS));
[tmp,inv_gen_old] = sort(igen);% 为最后重构的逆而保存
 if ng == size(gencost,1)
    gencost = gencost(igen,:);
 else
    gencost = gencost([igen;igen+ng],:);
 end
```

做完这些之后，x 向量中的变量就和 bus，gen 表中的向量有一致的顺序

```
x = [ Theta          ;%nb  母线电压相角
      V              ;%nb  母线电压幅值
      Pg             ;%ng  有功出力(标么值) （按升序排列）
      Qg             ];%ng  无功出力(标么值) （按升序排列）
```

非线性约束也有和 bus，branch 表一致的顺序

```
g = [   gp          ;       %nb  有功不平衡量（标么）
```

```
gq          ;       %nb 无功不平衡量（标么）
gsf         ;       %nl 首端视在功率限制 （标么）
gst         ];      %ng 末端视在功率限制（标么）
```

当这些建立好之后，变量的上下限限制应用如下：参考相角被限制在初始 bus 表中指定的值上。x 变量的 V 部分被限制在 bus 表的 VMAX 和 VMIN 的相应的值之间。x 变量的 Pg 和 Qg 部分被限制在 gen 表的 PMAX，PMIN，QMAX 和 QMIN 的相应值之间。非线性约束的建立也是类似的因此 gp 和 gq 都是等数约束（0 RHS）并且 gsf，gst 的约束来自于表 branch 的 RATE_A 列。

附加线性约束的例子

以下的例子用来展示如何将附加的通用线性约束加入到问题模型中去。对 case9.m 的标准解法中，母线 7 的相角滞后于母线 2 的相角 6.09^o，假设我们希望将这个相角差限制在 5 度以内。以下形式的线性约束将达到这个目的：

Theta(2) – Theta(7) <= 5 degrees

我们有 nb＝9 条母线，ng＝3 个发电机组和 nl＝9 条支路。因此，x 变量的前 9 个元素是母线电压相角，10－18 个元素是电压幅值，19－21 个元素是对应于母线 1，2 和 3（按这个顺序）的发电机组的有功注入，22－24 则是按照对应顺序的机组无功注入。注意到这个案例中机组的初始顺序是按照升序排列的，因此不必要再对初始数据重新排列。回到相角限制的问题上，我们可以看到它可以被映射为：

[0 1 0 0 0 0 -1 0 0 zeros(1,nb+ng+ng)]*x <= 5 degrees

我们建立如下的问题：

```
A = sparse([1;1].[2,7],[1,-1],1,24);
l = -Inf;
u = 5 * pi/180;
mpopt = mpoption('OPF_ALG',520);%  使用 fmincon w/通用仿真
opf('case9',A,l,u,mpopt)
```

这样，确实将相角差限制到了 5 度。

3.4.3 直流 OPF 方法

MATPOWER 中的直流潮流问题和以上所述的交流潮流问题类似，只不过是使用了网络的直流模型，这种模型只考虑母线电压相角和有功功率注入和有功潮流。

$$\min_{q,P_g} \sum_i f_i(P_{gi})$$

subject to

$$B_{bus}q = P_g - P_d - P_{bus,shift} - G_{sh} \qquad \text{有功平衡方程}$$

$$B_f q \le P^{\max} - P_{f,shift} \qquad \text{线路首端有功潮流约束}$$

$$-B_f q \le P^{\max} + P_{f,shift} \qquad \text{线路末端有功潮流约束}$$

$$P_{gi}^{\min} \le P_{gi} \le P_{gi}^{\max}$$ 机组有功出力限制

参考母线的相角依然被限在指定值上面。因为所有的约束是线性的，因此根据成本函数的形式该问题只是一个简单的 LP 或者 QP 问题。
当前的直流 OPF 还不能像通用交流 OPF 问题那样加入用户自定义线性约束和成本。

机组组合算法

前面的章节中所涉及到的标准最优潮流方法中我们并没有处理运行昂贵的机组完全停机的问题，取而代之的是在调度时将机组出力限制在最小技术出力之上。MATPOWER 提供了在一段时间内出力机组启停的办法来关停那些运行昂贵的机组以寻求最小成本的组合和调度。作为例子，运行一个 30 节点的系统，输入：
>>runuopf(‘case30’)
MATPOWER 采用了一种类似于动态规划的算法来处理这些机组退出。它通过一组层级来处理，其中层级 N 表示有 N 台机组被关闭，以 N＝0 开始。
算法的执行过程如下：
步骤 1：开始层级 0（N＝0），认为所有的机组都在线并且所有的约束都存在。
步骤 2：解普通的最优潮流。保存解作为当前最佳。
步骤 3：到下一个层级，N＝N＋1。使用上一步所得到的最佳解作为当前层级的基础，形成一个最小机组限制组合的机组候选序列，如果没有候选，跳到步骤 5。
步骤 4：对每一个候选序列中的机组，解当该机组关闭时的最优潮流，如果该机组关闭对应的整个系统成本低于当前最佳值，以该解作为当前最佳解。如果每一个候选值都能进一步降低成本，返回到步骤 3。
步骤 5：返回当前最优解作为最终解。

3.6 MATPOWER 选项

MATPOWER 使用一个选项向量来实现对选项的控制。它类似于 MATLAB 最优化工具箱早期版本中由 foptions 函数提供的现象向量。最主要的差别就在于不用再记住每个选项的索引，只需要根据选项的名称就可以对选型的值作出修改。MATPOWER 的默认选项向量是通过调用无参数 mpoption 来获得的。因此，输入：
>>runopf(‘case30’,mpoption)
就是另外一种执行默认选项的 OPF 算法的方法。
MATPOWER 选项向量实现对以下的控制：
- 潮流算法
- 潮流计算的中止标准
- 最优潮流（OPF）算法
- 对不同成本模型的默认 OPF 算法
- OPF 的成本转换参数
- OPF 的中止标准
- 冗余水平
- 结果输出方式

细节如下：

>>help mpotion

MPOPTION 用来设置和恢复 MATPOWER 选项向量。

opt = mpoption

返回默认的选项向量

opt = mpoption(name1,value1,name2,value2,…)

返回默认的选项向量，除了修改的选项值之外（最多 7 个），name#是选项的名称，value#是选项的新值。比如：options = mpotion('PF_ALG',2,'PF_TOL',le-4)

Opt = mpoption(opt,name1,value1,name2,value2,…)

除了使用 opt 代替默认的选项向量作为基础向量之外，其他的和上面一样。

当前定义的选项如下所示：

idx – NAME,default	description [options]
潮流计算选项	
1 — PF_ALG, 1	潮流算法
[1 – Newton's method	牛顿法]
[2 – Fast-Decoupled(XB version)	快速解耦算法（XB）]
[3 – Fast-Decoupled(BX version)	快速解耦算法（BX）]
[4 –Gauss Seidel	高斯－赛德尔法]
2 — PF_TOL, le-8	每一个单元（节点）的有功－无功最大的允许偏差。
3 — PF_MAX_IT,10	牛顿法的最大迭代次数
4 — PF_MAX_IT_FD,30	快速解耦算法的最大迭代次数
5 — PF_MAX_IT_GS,1000	高斯－赛德尔法的最大迭代次数
6 — ENFORCE_Q_LIMS,0	机组电压无功控制限制[0 或者 1]
10 — PF_DC,0	采用直流潮流模型
[0 — 使用交流模型，采用交流算法选项]	
[1 — 使用直流模型，忽略交流算法选项]	
OPF 选项（最优潮流计算选项）	
11 — OPF_ALG, 0	OPF 计算所采用的算法
[0 — 根据以下的顺序采用最好的默认算法，500,520 然后是 100/200]	
[否则是第一个数字指定问题模型并且第二个数字指定解决方法]	
[如下所示，（更加详细的信息请参看用户手册）]	
[100 — 标准模型（旧），constr]	
[120 — 标准模型（旧），dense LP]	
[140 — 标准模型（旧），sparse LP （松弛的）]	
[160 — 标准模型（旧），sparse LP (满的)]	
[200 — CCV 模型（旧），constr]	
[220 — CCV 模型（旧），dense LP]	
[240 — CCV 模型（旧），sparse LP （松弛的）]	
[260 — CCV 模型（旧），sparse LP (满的)]	
[500 — 通用模型，MINOS]	
[520 — 通用模型，fmincon]	
[更多关于模型的和操作的细节见用户手册]	
12 — OPF_ALG_POLY, 100	采用多项式成本函数的默认 OPF 算法（

只有当通用模型无解时才采用）

13 － OPF_ALG_PWL, 200 采用分段线性成本函数的默认 OPF 算法（只有当通用模型无解时才采用）

14 － OPF_POLY2PWL_PTS, 10 将多项式模型转化成分段线性模型时拐点的个数

16 － OPF_VIOLATION, 5e-6 不平衡量的限制值

17 － CONSTR_TOL_X, 1e-4 copf 和 fminopf 中 x 的停止范围

18 － CONSTR_TOL_F, 1e-4 copf 和 fminopf 中 F 的停止范围

19 － CONSTR_MAX_IT, 0 copf 和 fminopf 的最大迭代次数[0 表示 2×nb+150]

20 － LPC_TOL_GRAD,3e-3 lpopf 对斜率的允许最大误差

21 － LPC_TOL_X,1e-4 lpopf 对 x（最小步长）允许最大误差

22 － LPC_TOL_IT,400 lpopf 的最大迭代次数

23 － LPC_TOL_RESTART, 5 lpopf 的最大重开始次数

24 － LPC_P_LINE_LIM, 0 用有功代替实在功率潮流限制[0 或 1]

25 － LPC_IGNORE_ANG_LIM, 0 忽略相角差限制（无论指定与否）[0 或 1]

输出选项

31 － VERBOSE, 1 打印进程信息的数量

[0 – 不打印进程信息]

[1 － 打印一点进程信息]

[2 － 打印大量的进程信息]

[3 － 打印所有的进程信息]

32 － OUT_ALL, -1 结果的打印控制

[-1 － 用分散的标志来控制哪些需要输出]

[0 － 不打印任何东西]

[(覆盖除了 OUT-RAW 之外的分散控制变量)]

[1 － 输出所有]

[(覆盖除了 OUT-RAW 之外的分散控制变量)]

33 － OUT_SYS_SUM, 1 打印系统概要信息 [0 或者 1]

34 － OUT_AREA_SUM, 0 打印区域概要信息 [0 或者 1]

35 － OUT_BUS, 1 打印母线细节信息 [0 或者 1]

36 － OUT_BRANCH, 1 打印支路细节信息 [0 或者 1]

37 － OUT_GEN, 0 打印机组细节信息 [0 或者 1]（OUT_BUS 也包含机组信息）

38 － OUT_ALL_LIM, -1 控制打印约束信息

[-1 － 用分散的标志来控制哪些需要输出]

[0 － 不打印任何约束(覆盖分散控制变量)]

[1 － 有约束力的约束 (覆盖分散控制变量)]

[1 － 所有约束 (覆盖分散控制变量)]

39 － OUT_V_LIM, 1 控制打印电压限制信息

[0 － 不打印任何信息]

[1 － 只打印有约束力的限制信息]

[2 － 所有约束]

[(对 OUT_LINE_LIM,OUT_PG_LIM,OUT_QG_LIM 都一样)]

40 － OUT_LINE_LIM,1 控制打印线路限制信息
41 － OUT_PG_LIM,1 控制打印机组有功限制信息
42 － OUT_QG_LIM,1 控制打印机组无功限制信息
43 － OUT_RAW,0 打印 perl 数据库接口代码的初始数据[0 或 1]

其他选项

51 － SPARSE_QP, 1 对 QP 和 LP 解法采用稀疏矩阵技术[0 或 1]

MINOPF 选项

61 － MNS_FEASTOL, 0(1E-3) 原始可行区间，默认为 OPF_VIOLATION 的值
62 － MNS_ROWTOL, 0(1E-3) 行的误差范围，默认为 OPF_VIOLATION 的值
61 － MNS_XTOL, 0(1E-3) x 的误差范围，默认为 CONSTR_TOL_X 的值
64 － MNS_MAJDAMP, 0(0.5) 最大阻尼系数
65 － MNS_MINDAMP, 0(2.0) 最小阻尼系数
66 － MNS_PENALTY_PARM,0(1.0) 惩罚系数
67 － MNS_MAJOR_IT,0(200) 主迭代次数
68 － MNS_MINOR_IT,0(2500) 辅迭代次数
69 － MNS_MAX_IT,0(2500) 迭代限制
70 － MNS_VERBOSITY, -1
[-1 － 通过 VERBOSE 标志控制（一下的 0 或者 1）]
[0 － 无输出]
[1 － 只输出停止状态信息]
[2 － 输出打印停止状态和屏幕进程]
[3 － 输出打印屏幕进程，报告文件（通常是 fort.9）]
71 － MNS_CORE , 1200*nb+5000
72 － MNS_SUPBASIC_LIM, 0(2*ng) superbasics 限制
73 － MNS_MUTE_PRICE,0 (30) 复合价格

典型的选项向量的使用方式如下所示：

取得默认的选项向量

```
>>opt = mpoption;
```

使用快速解耦算法来做潮流计算

```
>>opt = mpotion(opt,’PF_ALG’,2);
```

只输出系统概要信息和机组信息:

```
>>opt = mpotion(opt,’OUT_BUS’,0,’OUT_BRANCH’,0,’OUT_GEN’,1);
```

显示所有的进程信息：

```
>>opt = mpotion(opt,’VERBOSE’,3);
```

然后使用这些设置来运行以下的潮流计算代码：

```
>> runpf(‘case57’, opt)
>>runpf(‘case118’,opt)
>>runpf(‘case300’,opt)
```

3.7 文件汇总

文档文件：

README MATPOWER 的基本介绍
README.txt MATPOWER 的基本介绍，为 windows 用户使用

docs/CHANGES	MATPOWER 的修改历史
docs/CHANGES.txt	MATPOWER 的修改历史，为 windows 用户使用
docs/manual.pdf	MATPOWER 的用户手册，pdf 版

高层方案

cdf2matp.m	将数据从 IEEE CDF 的格式转换成 MATPOWER 的格式
runcomp.m	运行两个最优潮流并且比较他们的结果
rundcopf.m	运新一个直流最优潮流计算
rundcpf.m	运行一个直流潮流计算
runduopf.m	运行一个可以处理高价机组停机的直流 OPF
runopf.m	运行一个最优潮流计算程序
runpf.m	运行一个潮流计算程序
runuopf.m	运行一个可以处理高价机组停机的 OPF

（以下的其他 opf.m,copf.m,fmincopf.m,lpopf.m 也可以当作高层方案使用 ）

数据输入文件：

caseformat.m	输入数据格式匹配的文档
case_ieee30.m	IEEE30 节点系统
case118.m	IEEE118 节点系统
case14.m	IEEE14 节点系统
case30.m	改进的 IEEE30 节点系统
case300.m	IEEE300 节点系统
case30pwl.m	分段线性成本结构的 case30.m
case30Q.m	带无功成本的 case30.m
case39.m	39 节点系统
case4gs.m	从 Grainger & Steveson 转化的 4 节点系统
case57.m	IEEE57 节点系统
case6ww.m	来自于 Wood & Wollenberg 的 1 节点系统
case9.m	3 机 9 节点系统（默认案例）
case9Q.m	带无功成本的 9 节点系统

各种方案使用的通用源文件和功能函数：

bustypes.m	创建参考节点，PV 节点和 PQ 节点的节点向量
compare.m	输出两种解法之间的差别概要信息
dAbr_dV.m	计算线路视在功率对电压的偏导，OPF 使用
dSbr_dV.m	计算线路视在功率对电压的偏导，OPF 和状态估计使用
dSbus_dV.m	计算节点注入复功率对电压的偏导，OPF、牛顿法潮流计算和状态估计使用
Ext2int.m	将数据矩阵从外部节点编号转换为内部节点编号
hasPQcap.m	检查机组的 P－Q 容量曲线约束
have_fcn.m	检查选项功能能否获得
idx_area.m	对 area 矩阵的列索引定义命名
idx_brch.m	对 branch 矩阵的列索引定义命名
idx_bus.m	对 bus 矩阵的列索引定义命名
idx_cost.m	对 gencost 矩阵的列索引定义命名
idx_gen.m	对 gen 矩阵的列索引定义命名
int2ext.m	将内部编号的数据矩阵转换为按外部编号的矩阵

isload.m	检查机组是否是可调度负荷
loadcase.m	将数据从文件或者结构体中导入到数据矩阵中
makeB.m	形成快速解耦算法所需要的 B 矩阵
makeBdc.m	形成直流 PF 和 OPF 所需的 B 矩阵
makePTDF.m	形成直流 PTDF 矩阵
makeSbus.m	对指定的机组和负荷形成节点复功率注入
makeYbus.m	形成节点导纳矩阵
mp_lp.m	用可获得的最佳算法计算线性规划问题
mp_qp.m	用可获得的最佳算法计算二次规划问题
mpver.m	输出 MATPOWER 版本信息
printpf.m	PF 或者 OPF 解的完美输出
savecase.m	保存数据矩阵中的数据到案例文件中
mpoption.m	设置 MATPOWER 选项

潮流计算（PF）

dcpf.m	执行直流潮流计算
fdpf.m	执行快速解耦潮流计算
gausspf.m	执行高斯－赛德尔潮流计算
newtonpf.m	执行牛顿法潮流计算
pfsoln.m	用潮流计算的解更新数据矩阵

最优潮流（OPF）：

各种 OPF 算法的通用文件

opf_form.m	返回提供 OPF 算法规则的代码
opf_slvr.m	返回提供 OPF 算法解法的代码
opf.m	高层算法 OPF 解的程序
poly2pwl.m	生成用对多项式成本的分段线性逼近的函数
pqcost.m	将机组成本矩阵 gencost 分为用功和无功成本
totcost.m	对指定的调度情况计算所有的成本

仅由直流 OPF 使用的文件

dcopf.m	执行直流最优潮流计算

仅由传统 OPF 所使用的文件（基于 constr 和 LP 的 OPF）

fg_names.m	返回函数的名字和给定算法的斜率
fun_ccv.m	对 CCV 规则计算目标函数和约束
fun_std.m	对标准规则计算目标函数和约束
grad_ccv.m	计算 CCV 规则下目标函数和约束的斜率
grad_std.m	计算标准规则下目标函数和约束的斜率
opfsoln.m	用 OPF 的解来根新数据矩阵

仅由基于 constr 的 OPF 使用的文件

copf.n	执行基于 constr 的 OPF 计算

仅由基于线性规划 LP 的 OPF 使用的文件

lpopf.m	执行基于 LP 的 OPF 计算

LPconstr.m 以连续线性规划解非线性优化问题
LPeqslvr.m 运行牛顿潮流计算
LPrelax.m 放松约束下解线性规划问题
LPsetup.m 以指定的方法解线性规划问题

仅由通用 OPF 规则使用的文件（基于 fmincon 和 MINOS）
genform.m 通用 OPF 规则的相关文档
makeAy.m 为通用 OPF 规则形成 A 矩阵和 b 向量

仅由基于 fminconOPF 使用的文件
consfmin.m 计算约束的值和斜率
costfmin.m 计算目标函数的值和斜率
fmincopf.m 执行基于 fmincon 的 OPF

进由带机组组合的 OPF 使用的文件
fairmax.m 除了实现随机中断之外，和 MATLAB 的内部函数 max（）一样
uopf.m 执行带机组组合的 OPF

其他：（位于 extras 子文件夹中）
拍卖市场软件（位于 smartmarket 子文件夹中）
auction.m 清除由竞价规则和 OPF 解产生的竞价和供应集合
case2off.m 根据 gen 和 gencost 矩阵创建价格/出力的竞价/供应集合
idx_disp.m 为 dispatch 矩阵命名列索引
off2case.m 基于价格/出力的竞价/供应更新 gen 和 gencost 矩阵
printmkt.m 输出打印市场结果
runmkt.m 高级程序，运行一个基于 OPF 的拍卖
SM_CHANGES smartmarket 软件的更新史

测试：（位于 t 文件夹中）
soln9_dcopf.mat 测试用数据
soln9_dcpf.mat 测试用数据
soln9_opf.mat 测试用数据
soln9_opf_ang.mat 测试用数据
soln9_opf_extrals1.mat 测试用数据
soln9_opf_Plim.mat 测试用数据
soln9_opf_PQcap.mat 测试用数据
soln9_pf.mat 测试用数据
soln9_opf.mat 测试用数据
t_auction.m 测试 extras/smartmarket 中的 auction.m
t_aution_case.m t_auction 的测试案例
t_aution_fmincopf.m 基于 fmincon 的对 extras/smartmarket 中的 auction.m 的测试
t_begin.m 开始一个测试集合
t_case9_opf.m OPF 测试的案例文件（版本 1 的格式）
t_case9_opfv2.m OPF 测试的案例文件（版本 2 的格式）

t_case9_pf.m	潮流计算测试的案例文件（版本 1 的格式）
t_case9_pfv2.m	潮流计算测试的案例文件（版本 2 的格式）
t_end.m	停止一个测试集合并且输出数值
t_hasPQcap.m	测试 hasPQcap.m
t_is.m	测试两个矩阵是否在一定的范围内是一样的
t_jacobian.m	对偏导进行数值测试
t_loadcase.m	测试 load_case.m
t_makePTDF.m	测试 makePTDF.m
t_off2case.m	测试 off2case.m
t_ok.m	测试一个表达式是否为真
t_opf.m	测试 OPF 解法
t_pf.m	测试潮流计算解法
t_run_tests.m	运行一系列测试的框架
t_runmarket.m	测试 runmarket.m
t_skip.m	跳过指定的测试
test_matpower.m	运行所有的可运行的 MATPOWER 测试

4 致谢

作者需要感谢许多人。感谢 Chris DeMarco，威斯康星大学的 PSERC 的伙伴之一，为他所提供的雅克比矩阵建立的方法。感谢 Bruce Wollenberg 对版本 1 的所有改进的意见。版本 2 基本上就是按照他的意见修改的。感谢 Andrew Ward 对我们 OPF 中优化有功成本的编码和测试所作的工作。感谢 Alberto Borghetti 对高斯－赛德尔潮流计算所提供的代码。同时要感谢这么多年来为我们提供代码，漏洞报告和建议的人们。最后，感谢科奈尔大学 PSERC 中心的 Bob Thomas 对整个 MATPOWER 的开发过程中所作出的贡献。

5 参考文献

1. R. van Amerongen, “A General-Purpose Version of the Fast Decoupled Loadflow”, *IEEE Transactions*
on Power Systems, Vol. 4, No. 2, May 1989, pp. 760-770.
2. O. Alsac, J. Bright, M. Prais, B. Stott, “Further Developments in LP-based Optimal Power Flow”,
IEEE Transactions on Power Systems, Vol. 5, No. 3, Aug. 1990, pp. 697-711.
3. R. Fletcher, *Practical Methods of Optimization*, 2nd Edition, John Wiley & Sons, p. 96.
4. P. E. Gill, W. Murry, M. H. Wright, *Practical Optimization*, Academic Press, London, 1981.
5. A. F. Glimm and G. W. Stagg, “Automatic calculation of load flows”, *AIEE Transactions (Power*
Apparatus and Systems), vol. 76, pp. 817-828, Oct. 1957.
6. A. Grace, *Optimization Toolbox*, The MathWorks, Inc., Natick, MA, 1995.

7. C. Li, R. B. Johnson, A. J. Svoboda, “A New Unit Commitment Method”, *IEEE Transactions on*
Power Systems, Vol. 12, No. 1, Feb. 1997, pp. 113-119.
8. C. Mészáros, “The efficient implementation of interior point methods for linear programming and
their applications”, *Ph.D. Thesis*, Eötvös Loránd University of Sciences, 1996.
9. B. Stott, “Review of Load-Flow Calculation Methods”, *Proceedings of the IEEE*, Vol. 62, No. 7,
July 1974, pp. 916-929.
10. B. Stott and O. Alsac, “Fast decoupled load flow”, *IEEE Transactions on Power Apparatus and*
Systems, Vol. PAS-93, June 1974, pp. 859-869.
11. B. Stott, J. L. Marino, O. Alsac, “Review of Linear Programming Applied to Power System Rescheduling”,
1979 PICA, pp. 142-154.
12. W. F. Tinney and C. E. Hart, “Power Flow Solution by Newton’s Method”, *IEEE Transactions on*
Power Apparatus and Systems, Vol. PAS-86, No. 11, Nov. 1967, pp. 1449-1460.
13. A. J. Wood and B. F. Wollenberg, “Power Generation, Operation, and Control, 2nd Edition, John
Wiley & Sons, p. 108-111.
14. B.A Murtagh andM.A. Saunders, “MINOS 5.5 User’s Guide”, *Stanford University Systems Optimization*
Laboratory Technical Report SOL83-20R.