现代电力系统分析
Morden Power System Analysis

电气工程学院 王小君

xjwang1@bjtu.edu.cn

内容提要

- 概述
- 最优潮流的数学模型
- 最优潮流算法综述
- 简化梯度算法
- 解耦最优潮流计算

插入视频

一、概述

➢ 基本潮流

- 基本潮流：对一定的扰动变量p（负荷情况），根据给定的控制变量u（发电机有功、无功、节点电压模值等），求出相应的状态变量x。

- 一次基本潮流计算，决定了电力系统的一个运行状态。

- 基本潮流计算结果主要满足了变量间等约束条件。

$$f(x，u，p)=0$$

一、概述

➢ 最优潮流(Optimal Power Flow，OPF)

- 系统状态变量及有关函数变量的上下限值间有一定间距，控制变量可以在一定范围内调节，因而对某一种负荷情况，理论上有众多可行解。选出最佳方案。

- 最优潮流就是当系统的结构参数及负荷情况给定时，通过控制变量的优选，所找到的能满足所有指定的约束条件，并使系统的某一个性能指标或目标函数达到最优时的潮流分布。

一、概述

➢ 最优潮流与基本潮流的区别：

- 基本潮流计算时控制变量u是事先给定的；而最优潮流中的u则是可变而待优选的变量，为此必然有一个作为u优选准则的函数。
- 最优潮流计算除了满足潮流方程这一等式条件之外，还必须满足与运行限制有关的大量不等式的约束条件。

一、概述

➢ 最优潮流与基本潮流的区别：

- 进行基本潮流计算是求解非线性代数方程组；而最优潮流计算由于其模型从数学上讲是一个非线性规划问题，因此需要采用最优化方法来求解。

- 基本潮流计算所完成的仅仅是一种计算功能，即从给定的u求出相应的x；而最优潮流计算则能够根据特定目标函数并在满足相应约束条件的情况下，自动优选控制变量，这便具有指导系统进行优化调整的决策功能。

一、概述

➢ 最优潮流与经济调度的区别：

- 最优潮流使用潮流方程作为平衡条件，模型更加精确，可以精确地处理网损最小化等多种问题，而经济调度仅使用忽略网损的有功平衡约束，受制于简化模型，仅能求解经济性最优问题；
- 最优潮流引入支路潮流约束、节点电压约束等安全性约束，甚至可以引入可靠性约束、随机性约束等一系列非线性约束条件，经济调度仅能处理简单的安全约束问题；

一、概述

➢ 最优潮流的关键内容

数学模型的建立

目标函数：根据不同需求，如辅助服务定价与节点实时电价、网络阻塞管理、输电费用计算、可用传输容量估计

约束条件：如何将一些实际的约束(如节点电压波动范围、发电机出力限制、线路传输容量约束、紧急事故约束、环境约束等)用数学形式精确表达出来。

如何快速有效地求解OPF问题

大量非线性不等式约束的处理，随着约束条件的增多和复杂，导致问题成为优化理论中最难求解的非线性混合整数问题，部分问题甚至具有高度非线性、不可微、非凸、多峰等特点；

早期使用简化梯度法、牛顿法、线性规划法等求解较为简单的OPF问题，现在一般使用现代内点算法、进化类方法求解较为复杂的OPF问题。

二、最优潮流的数学模型

➢ 最优潮流的控制变量

- 最优潮流的变量分为控制变量（u）及状态变量（x）。

一般常用的控制变量有：

（1）除平衡节点外，其它发电机的有功出力；

（2）所有发电机节点及具有可调无功补偿设备节点的电压模值；

（3）移相器抽头位置

（4）带负荷调压变压器的变比

（5）并联电抗器/电容器容量

一般常用的状态变量有：

（1）除平衡节点外，其它所有节点的电压相角；

（2）除发电机节点以及具有可调无功补偿设备节点之外，其它所有节点的电压模值。

二、最优潮流的数学模型

➢ 最优潮流的目标函数

（1）全系统发电燃料总耗量（或总费用）

$$f = \sum_{i \in NG} K_i(P_{Gi}) \qquad (1)$$

式中：*NG*为全系统发电机的集合，其中包括平衡节点s的发电机组。

$K_i(P_{Gi})$是发电机组G_i的耗量特性。

由于平衡节点s的电源有功出力不是控制变量，其节点注入功率必须通过潮流计算才能决定，是节点电压模值U及相角θ的函数，于是

$$P_{Gs} = P_s(U, \theta) + P_{Ls} \qquad (2)$$

式中：$P_S(U, \theta)$为注入节点s而通过与节点相关的线路输出的有功功率；P_{LS}为节点s的负荷功率。

所以（1）式可写成：

$$f = \sum_{\substack{i \in NG \\ i \neq s}} K_i(P_{Gi}) + K_s(P_{Gs}) \qquad (3)$$

二、最优潮流的数学模型

(2) 有功网损

$$f = \sum_{i,j \in NL} (P_{ij} + P_{ji}) \qquad (4)$$

式中：*NL*为所有支路的集合。

可直接采用平衡节点的有功注入作有功网损最小化的目标函数

$$\min f = \min P_s(U, \theta) \qquad (5)$$

除此之外，最优潮流还可以采用其它类型的目标函数，如偏移量最小、控制设备调节量最小、投资及年运行费用之和最小等。

由上可见，最优潮流的目标函数不仅与控制变量***u***有关，同时和状态变量***x***有关。因此可用简洁的形式表示

$$f = f(u, x) \qquad (6)$$

二、最优潮流的数学模型

➢ 等式约束条件及不等式约束条件

- 最优潮流分布必须满足基本潮流方程，这就是最优潮流问题的等式约束条件。即$f(x,u,p)=0$。由于扰动变量p是给定的，该式可简化为

$$g(x,u)=0 \qquad (7)$$

- 不等式约束条件

（1）有功电源出力上下限约束；
（2）可调无功电源出力上下限约束；
（3）带负荷调压变压器变比K调整范围约束；
（4）节点电压模值上下限约束；
（5）输电线路或变压器元件中通过的最大电流或视在功率约束；
（6）线路通过的最大有功潮流或无功潮流约束
（7）线路两端节点电压相角差约束，等等。

统一表示为 $h(u,x)<=0$ （8）

二、最优潮流的数学模型

- 电力系统最优潮流的数学模型可表示为

$$\left.\begin{aligned} &\min_{u} f(u,x) \\ &s.t. \quad g(u,x)=0 \\ &\qquad h(u,x)\le 0 \end{aligned}\right\} \qquad (9)$$

- 采用不同的目标函数并选择不同的控制变量，再和相应的约束条件结合，就可构成不同的最优潮流问题。

（1）对有功及无功进行综合优化的通常泛称最优潮流问题。

（2）有功最优潮流

（3）无功优化潮流

模型扩展

电力市场中的应用领域	OPF数学模型扩展			
	目标函数	网络模型	特殊约束	特殊控制变量
实时电价计算	社会利益最大	DC/AC	发电机爬坡约束、相关备用约束	发电方与需求方报价(大部分为分段线性函数)
输电费用确定	发电成本最小/用户净收益最大	DC/AC	事故约束	发电出力、负荷功率、FACTS设备
网络阻塞管理	阻塞管理费用最小	DC/AC	各种运行约束、事故约束、稳定性约束	发电机出力增减量、双边合同削减量、FACTS设备
ATC计算	总输电容量最大	AC	事故约束、稳定性约束、随机操作约束	FACTS设备
辅助服务支持	辅助服务费用最小	DC	相关备用约束	发电侧备用容量(负荷侧的削减可看做一种形式的备用)
输电权利分配	输电权利拍卖收益最大	DC	相关报价约束、事故约束	供、需双方购买输电权利的报价、FACTS设备

三、最优潮流算法综述

(一)最优潮流算法分类

- 按处理约束的不同分类
- 按选择的修正量不同分类
- 按如何确定修正量的方向分类

1. 按处理约束的方法分类

- 可以分成三类
  - 罚函数法
  - Kuhn-Tucker罚函数类（简称KT罚函数类）、
  - Kuhn-Tucker类（简称KT类）

三、最优潮流算法综述

➢ 按处理约束的方法：罚函数法

- 把等式及不等式约束都用罚函数引入目标函数，将有约束优化问题转化为无约束优化问题，（9）式优化问题：

$$\left.\begin{aligned} &\min_{u} f(\boldsymbol{u},\boldsymbol{x}) \\ &s.t. \quad \boldsymbol{g}(\boldsymbol{u},\boldsymbol{x})=0 \\ &\qquad \boldsymbol{h}(\boldsymbol{u},\boldsymbol{x})\le 0 \end{aligned}\right\}$$

变成：

$$\min F(\boldsymbol{u},\boldsymbol{x}) = f(\boldsymbol{u},\boldsymbol{x}) + \sum_i \omega_{1i} g_i^2(\boldsymbol{u},\boldsymbol{x}) + \sum_i \omega_{2i} h_i^2(\boldsymbol{u},\boldsymbol{x}) \qquad (10)$$

式中ω1i和ω2i是罚因子，取充分大的正数。对越界的不等式约束通过罚函数引入目标函数。对未越界者相应罚因子为0，在罚函数中不出现。

三、最优潮流算法综述

➢ 按处理约束的方法：KT-罚函数法

- 只将越界的不等式约束通过罚函数引入目标函数，保留等式约束方程，即：

$$\left.\begin{aligned}&\min F(\boldsymbol{u},\boldsymbol{x}) = f(\boldsymbol{u},\boldsymbol{x}) + \sum_i \omega_i h_i^2(\boldsymbol{u},\boldsymbol{x})\\ &\text{s. t.} \quad \boldsymbol{g}(\boldsymbol{u},\boldsymbol{x})=0\end{aligned}\right\} \tag{11}$$

再用拉格朗日乘子将等式约束引入目标函数，构造拉格朗日函数：

$$L(\boldsymbol{u},\boldsymbol{x}) = F(\boldsymbol{u},\boldsymbol{x}) + \boldsymbol{\lambda}^{\mathrm{T}}\boldsymbol{g}(\boldsymbol{u},\boldsymbol{x}) \tag{12}$$

L满足最优解的条件是满足Kuhn-Tucker条件(K-T条件)：

$$\frac{\partial L}{\partial \boldsymbol{x}} = 0 \quad , \frac{\partial L}{\partial \boldsymbol{u}} = 0, \quad \frac{\partial L}{\partial \boldsymbol{\lambda}} = 0 \tag{13}$$

求解上面方程得到最优解。

三、最优潮流算法综述

➢ 按修正的变量空间分类

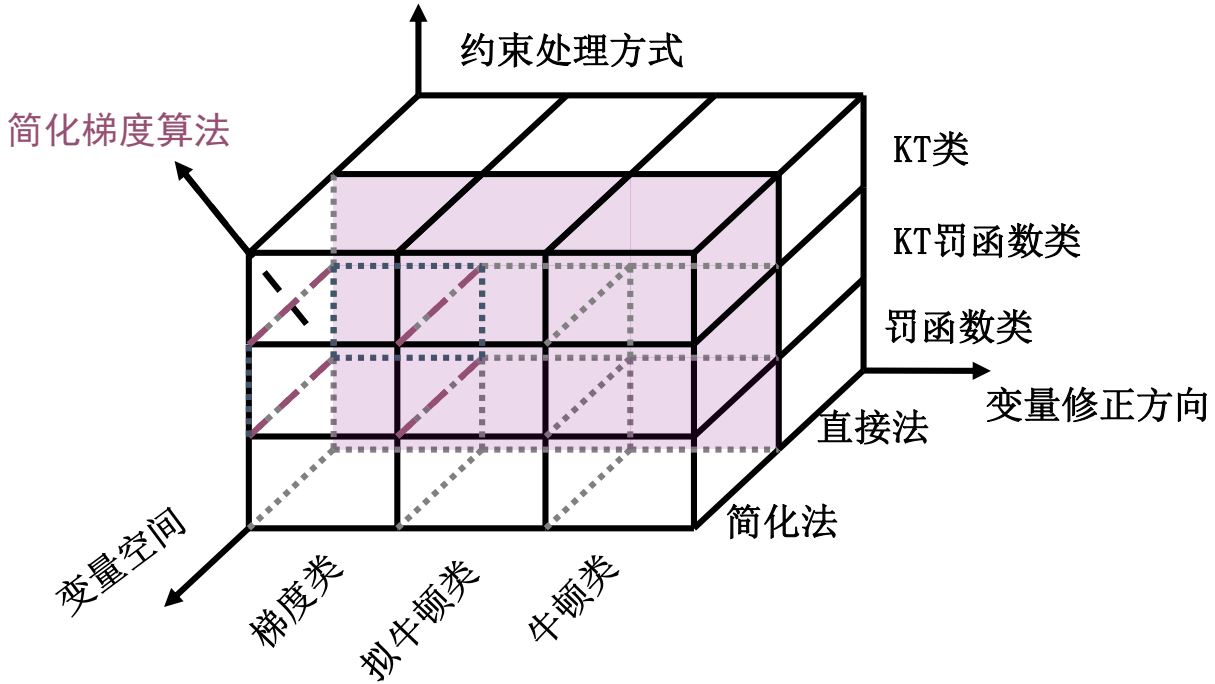
- 在迭代过程中，可以是同时修正全变量空间，包括控制变量***u***和状态变量***x***，称为直接类算法。
- 也可以只修正控制变量***u***，而状态变量通过求解约束方程(潮流方程)得到。称为简化类算法。

➢ 按变量修正的方向分类

- 确定变量修正的方向有三类方法：
  - 第一类为梯度类算法，包括梯度法即最速下降法，这类方法具有一阶收敛性；
  - 第二类为拟牛顿类算法，如共扼梯度法和各种变尺度法，这类方法收敛性介于一阶和二阶之间；
  - 第三类为牛顿法，例如海森矩阵法，这类方法有二阶收敛性。

三、最优潮流算法综述

➢ 三类最优潮流算法的三维分类图形表示



三、最优潮流算法综述

(二)求解算法

- 主要方法有:
  - 广义拉格朗日乘子法
  - 非线性规划法
  - 内点法
  - 人工智能方法等。

3.1 按处理约束的方法分类
3.2 按修正的变量空间分类
3.3 按变量修正的方向分类
3.4 三类最优潮流算法的三维分类图形表示
3.5 广义拉格朗日乘子法
3.10 内点算法
3.11 人工智能方法
3.12进化类算法

三、最优潮流算法综述

➢ 广义拉格朗日乘子法—>KKT条件

模型的一般表达

$$\min f(x)$$
$$s.t.\ \ h_k(x) = 0$$
$$g_j(x) \le 0$$

x为向量，$(x_1, x_2, \dots, x_n)^T$

不等式约束如何处理？

能不能也用类似拉格朗日法来求解？

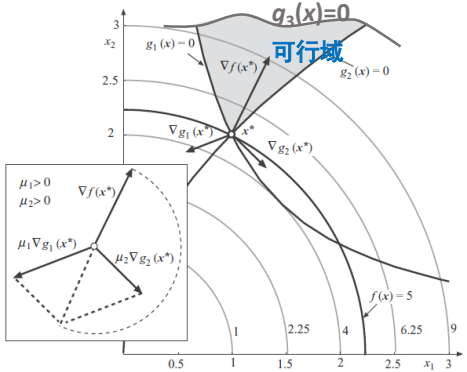
$$f(x) + \sum_{k=1}^{m} \lambda_k h_k(x) + \sum_{j=1}^{n} \mu_j g_j(x) = 0$$

证明省略

Karush-Kuhn-Tucker, KKT条件

$$\nabla f(x) + \sum_{k=1}^{m} \lambda_k \nabla h_k(x) + \sum_{j=1}^{n} \mu_j \nabla g_j(x) = 0$$
$$h_k(x) = 0, \quad k = 1, 2, \cdots, m$$
$$g_j(x) \le 0, \quad j = 1, 2, \cdots, n$$
$$\mu_j g_j(x) = 0, \quad j = 1, 2, \cdots, n$$
$$\mu_j \ge 0, \quad j = 1, 2, \cdots, n$$

互补松弛条件



尽管KKT里面还有不等式约束，但是可以通过互补松弛条件$\mu_j g_j(x) = 0$,来分别讨论

$$\mu_j > 0, g_j(x) = 0; g_j(x) < 0, \mu_j = 0$$

的各种组合情况，此时就变成了线性或非线性方程组求解。

x维数高时，组合同样非常多，同样难解

三、最优潮流算法综述

➢ 拉格朗日乘子的经济学意义

模型的一般表达

$$\min f(x)$$
$$s.t.\ \ h_k(x)=0$$
$$g_j(x)\le 0$$

拉格朗日方程

$$f(x)+\sum_{k=1}^{m}\lambda_k h_k(x)+\sum_{j=1}^{n}\mu_j g_j(x)=0$$

在电力系统最优潮流模型中，目标函数为发电成本，等式约束包括功率平衡约束，不等式约束包括线路传输容量约束，如果考虑网络损耗，等式约束中还应加入网损约束。则优化模型可以表示为：

$$\min \sum_{k=1}^{n} C_k(I_k)$$
$$s.t.\ \sum_{k=1}^{n} I_k = L(I_1,\dots,I_{n-1})$$
$$F_l(I_1,I_2,\dots,I_{n-1})\le F_l^{max},\ l=1,2,\dots,m$$

n为系统总节点数，I_k为节点注入净功率，C_k为功率注入成本；L为系统网损；m为系统支路数，F_l为支路传输容量

$$l=\sum_{k=1}^{n} C_k(I_k)+\pi[L(I_1,\dots,I_{n-1})-\sum_{k=1}^{n} I_k]+\sum_{l=1}^{m}\mu_l[F_l(I_1,I_2,\dots,I_{n-1})-F_l^{max}]$$

拉格朗日乘子π和μ_l的经济学意义：代表当约束条件变动时，目标函数极值的变化。如当某条支路容量越限时，即有$F_l>F_l^{max}$，目标函数中将增加一部分成本($\mu_l\ge 0$)，即对违背约束产生惩罚。

三、最优潮流算法综述

➢ 拉格朗日乘子的经济学意义

$$l=\sum_{k=1}^{n} C_k(I_k)+\pi[L(I_1,\dots,I_{n-1})-\sum_{k=1}^{n} I_k]+\sum_{l=1}^{m}\mu_l[F_l(I_1,I_2,\dots,I_{n-1})-F_l^{max}]$$

应用KKT条件：

1) $\sum_{k=1}^{n}\nabla C_k(I_k)+\pi\nabla L-\pi\sum_{k=1}^{n}\nabla I_k+\sum_{l=1}^{m}\mu_l\nabla F_l=0$

2) $\sum_{k=1}^{n} I_k=L(I_1,\dots,I_{n-1})$　3) $\mu_l[F_l(I_1,I_2,\dots,I_{n-1})-F_l^{max}]=0,\ \mu_l\geq 0$

由KKT条件1）有：$\frac{\partial l}{\partial I_k}=\frac{dC_k}{dI_k}+\pi\left(\frac{\partial L}{\partial I_k}-1\right)+\sum_{l=1}^{m}\mu_l\frac{\partial F_l}{\partial I_k}=0 \quad k=1,\dots,n-1$

$\frac{\partial l}{\partial I_n}=\frac{dC_n}{dI_n}-\pi=0$ ⇒ $\frac{dC_n}{dI_n}=\pi$

*n*为平衡节点

$\frac{dC_k}{dI_k}=\pi-\pi\frac{\partial L}{\partial I_k}-\sum_{l=1}^{m}\mu_l\frac{\partial F_l}{\partial I_k} \quad k=1,\dots,n-1$

导数含义：单位节点注入功率变化引起的发电成本变化

由上式可知，除平衡节点外的其他节点，单位节点注入功率变化引起的发电成本，即边际发电成本，由三部分构成：

1）π：平衡节点的边际成本，对所有节点一致；

2）$-\pi\frac{\partial L}{\partial I_k}$：网络损耗引起的发电成本，与节点位置有关；

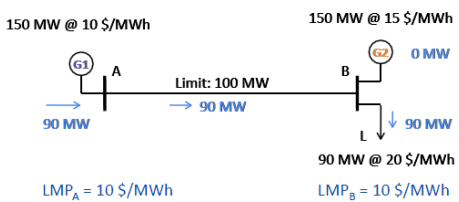
3）$-\sum_{l=1}^{m}\mu_l\frac{\partial F_l}{\partial I_k}$：线路潮流约束引起的发电成本，与节点位置有关

上述边际发电成本在电力市场环境下定义为节点边际电价(locational marginal price, LMP)

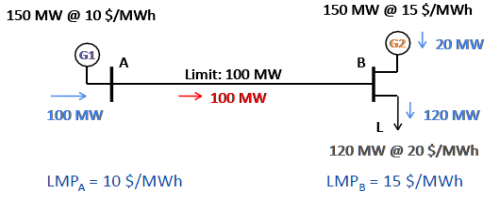
三、最优潮流算法综述

➢ 节点边际电价与阻塞管理

节点边际电价表达式中的第三部分$-\sum_{l=1}^{m}\mu_l\frac{\partial F_l}{\partial I_k}$为由线路潮流约束引起的发电成本，即阻塞费用。在最优潮流模型中加入线路传输容量约束，即实现了阻塞管理。



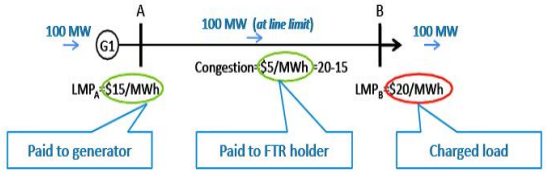
线路容量约束未起作用，两节点的节点边际电价相同



节点B负荷增加后，受线路容量约束影响，节点B的节点边际电价提高

为什么存在线路传输容量限制：电力线路有容许电流与容许电压损失值，容许电流是根据热平衡条件确定的导线长期允许通过的电流；容许电压损失指为了保证各用户的电压偏移不超出允许的范围而确定的电压损失值；根据容许电流、线路的额定电压和功率因数即可确定线路的允许传输容量。

节点边际电价的意义：由于网损及线路传输容量的限制，网络中各个节点具有不同的边际发电成本。市场环境下，节点边际电价实现对每个节点单独结算，决定了负荷对发电商及金融输电权持有者等市场参与者的分别支付，能够在维持系统稳定的前提下提高电力市场的经济效率。



三、最优潮流算法综述

➢ 内点算法

- 1954年，Frish提出了最早的内点法，它是一种仅限于求解无约束优化问题的障碍参数法。
- 1984年，Karmarkar提出了线性规划的一种新的内点算法，证明该算法具有多项式计算复杂性，该算法在求解大规模线性规划问题时，计算速度比单纯形法快50倍以上。
- 随后，Gill将内点法的应用进一步推广到非线性规划领域。
- 近年来，许多学者对Karmarkar算法进行了广泛深入的研究，一些新的变型算法相继出现，最有发展潜力的是路径跟踪法(Path Following)，又称为跟踪中心轨迹法。

三、最优潮流算法综述

➢ 内点法——障碍函数

按罚函数原理将不等式约束的原问题通过示性函数转换为等式约束问题

$$\min f(x)$$
$$s.t.\ \ h_k(x)=0$$
$$g_j(x)\le 0$$

x为向量，$(x_1, x_2, \dots, x_n)^T$

等价问题

$$\min\ f(x)+\sum_{i=1}^{m} I_-(g_i(x))$$
$$s.t.\ \ h_k(x)=0$$
$$I_-(g_i(x))=\begin{cases}0, & g_i(x)\le 0\\ \infty, & g_i(x)>0\end{cases}$$

目标函数不可微，无法用牛顿法求解

思路：找一个接近的连续可微函数来代替不可微的示性函数

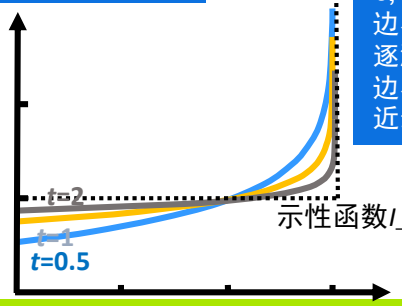
$$\hat{I}_-(g_i(x))=-(1/t)log(-g_i(x))\approx I_-(g_i(x))$$
$$(t>0)$$

目标函数变为：

$$\min\ f(x)+\sum_{i=1}^{m}-(1/t)log(-g_i(x))$$

单增凸函数，可用牛顿法解

效果如何？



通过逐渐增加t，将等效的边界障碍函数逐渐逼近实际边界，从而逼近最优解

三、最优潮流算法综述

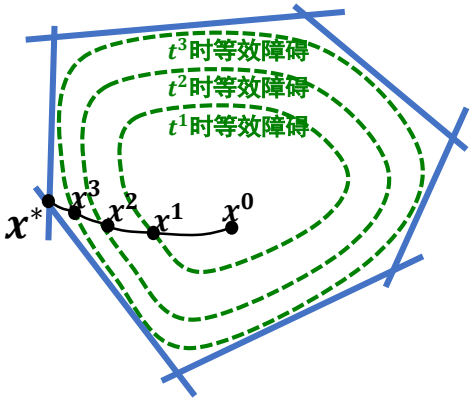
➢ 内点法——中心路径

按罚函数原理将不等式约束的原问题通过示性函数转换为等式约束问题

$$\min\ f(x)+\sum_{i=1}^{m}-(1/t)log(-g_i(x))$$
$$s.t.\ h_k(x)=0\quad (h_k(x)=Ax-b=0)$$

假设对任意t，$x^*(t)$为问题的解，称$[t,\ x^*(t)]$为中心点，这样一系列点构成的集合定义为问题的中心路径。

这样在给定一个t和初始内点x^0的情况下，可以通过逐步增大t的方式形成一系列最小化问题，通过这一系列问题跟踪中心路径找到原始问题的最优解。



对每个t，该问题拉格朗日函数为：

$$f(x)+\sum_{i=1}^{m}-(1/t)log(-g_i(x))+(A^Tx-b)\mu$$

每个中心点$[t,\ x^*(t)]$按拉格朗日乘子法应满足：

$$\nabla f(x^*(t))+\sum_{i=1}^{m}\frac{1}{-g_i(x^*(t))\cdot t}\nabla g_i(x^*(t))+A^T\mu=0$$

$$g_j(x^*(t))<0$$
$$Ax^*(t)-b=0$$

三、最优潮流算法综述

➢ 内点法——中心路径的对偶间隙

按罚函数原理将不等式约束的原问题通过示性函数转换为等式约束问题

中心点$[t,\ x^*(t)]$满足条件：

$$\nabla f(x^*(t))+\sum_{i=1}^{m}\frac{1}{-g_i(x^*(t))\cdot t}\nabla g_i(x^*(t))+A^T\mu=0$$

$$g_j(x^*(t))<0$$
$$Ax^*(t)-b=0$$

$$\lambda_i^*(t)=\frac{1}{-g_i(x^*(t))\cdot t}\quad \eta^*(t)=\frac{\mu}{t}\cdot t$$

假设有一个$f_D(\lambda,\eta)$的对偶函数存在，并假设其拉格朗日函数表达为：

$$\max_{\lambda,\mu:\lambda_i\geq 0} f_D(\lambda,\eta)=\max_{\lambda,\mu:\lambda_i\geq 0}\min_x L(x,\lambda,\eta)$$
$$=\max_{\lambda,\mu:\lambda_i\geq 0}\min_x f(x)+\sum_{i=1}^{m}\lambda_i g_i(x)+\mu^T(Ax-b)$$

中心点$[t,\ x^*(t)]$满足条件的情况下，对偶函数的拉格朗日函数$L(x,\lambda,\eta)$在$x^*(t)$达到最小

对偶函数$f_D(\lambda,\eta)$最小值为：

$$f_D(\lambda^*(t),\eta^*(t))=f(x^*(t))+\sum_{i=1}^{m}\underbrace{\lambda_i^*(t)\cdot g_i(x^*(t))}_{=-1/t}+\mu^T\underbrace{(Ax^*(t)-b)}_{=0}$$

$$=f(x^*(t))-\frac{m}{t}$$

$x^*(t)$和对偶可行解$\lambda^*(t),\eta^*(t)$之间对偶间隙为$\frac{m}{t}$，随着t增大，将收敛至最优解

三、最优潮流算法综述

➢ 内点法——中心路径的KKT解释

中心点$[t,\ x^*(t)]$满足条件：

$$\nabla f(x^*(t)) + \sum_{i=1}^{m} \frac{1}{-g_i(x^*(t)) \cdot t} \nabla g_i(x^*(t)) + A^T \mu = 0$$

$$g_j(x^*(t)) < 0$$

$$Ax^*(t) - b = 0$$

$$\lambda_i^*(t) = \frac{1}{-g_i(x^*(t)) \cdot t} \qquad \eta^*(t) = \frac{\mu}{t} \cdot t$$

那么也就是说，对于一个x点来说，其是$x^*(t)$的充要条件是：

$$\nabla f(x) + \sum_{i=1}^{m} \lambda_i \nabla g_i(x) + A^T \mu = 0$$

$$Ax - b = 0$$

$$g_j(x) \le 0$$

$$-\lambda_i g_i(x) = \frac{1}{t}$$

$$\lambda_i > 0$$

KKT条件的变形

原KKT条件互补松弛条件$\lambda_i g_i(x) = 0$变成了$-\lambda_i g_i(x) = \frac{1}{t}$，对于很大的t，几乎可以认为满足原始KKT条件。

该KKT条件是n+p+m个变量x,μ和λ的n+p+m个非线性方程组，对每个t，求解该方程组，既可以得到$x^*(t)$

三、最优潮流算法综述

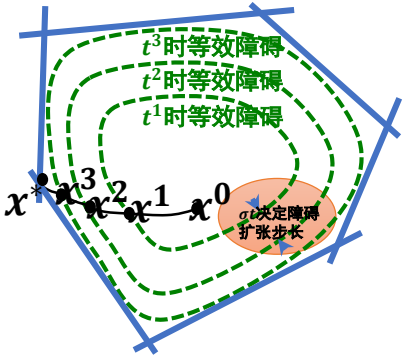
➢ 内点法——中心跟踪序列最小化求解

算法步骤

给定严格可行点x^0、初始步长t^0、迭代步长σ、误差阈值ϵ

重复进行

1. 中心点步骤：从x^0开始，求解KKT非线性方程组，确定$x^*(t)$
2. 改进：$x^{n+1}=x^*(t)$
3. 停止判断：如果$(m/t)<\epsilon$，则收敛退出
4. 增加步长：$t^{n+1}=\sigma t^n$



为什么不把t设得足够大，一步到位？

t设得太大，则初始点过远，很难用牛顿法求解上述变形的KKT方程组，Hessian矩阵在靠近边界时会剧烈变动

那么t的步长σ怎么设？

由于$x^{*(n)}$到$x^{*(n+1)}$中间是通过牛顿法求解KKT变形条件的线性方程组，如果步长σ取得过大，则距离越远，内部牛顿迭代次数就会增多，取得过小，外层跟踪次数就会大增，对一般问题σ取在10-20效果较好，内外层计算时间总体平衡。

三、最优潮流算法综述

➢ 内点法——原对偶方法

$$\nabla f(x)+\sum_{i=1}^{m}\lambda_i\nabla g_i(x)+A^T\mu=0$$

$$Ax-b=0 \qquad g_j(x)\le 0$$

$$-\lambda_i g_i(x)=\frac{1}{t} \qquad \lambda_i>0$$

KKT 条件变形

矩阵形式

$$r_t(x,\lambda,\mu)=\begin{bmatrix}\nabla f(x)+\lambda\nabla g(x)+A^T\mu\\ -diag(\lambda)g(x)-\left(\frac{1}{t}\right)I\\ Ax-b\end{bmatrix}=0$$

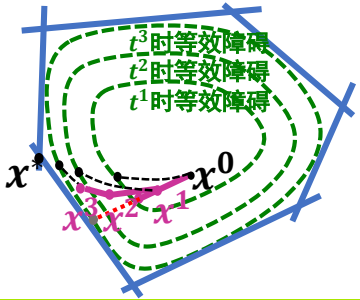
$$g(x)=[g_1(x),\cdots,g_m(x)]^T \qquad f(x)=[f_1(x),\cdots,f_m(x)]^T$$

牛顿步径的线性方程：

$$y=(x,\lambda,\mu)$$

$$r_t(y+\Delta y)\approx r_t(y)+r_t(y)\Delta y$$

$$\begin{bmatrix}\nabla^2 f(x)+\sum_{i=1}^{m}\lambda_i\nabla^2 g_i(x) & \nabla g(x) & A^T\\ -diag(\lambda)\nabla g(x) & -diag(g(x)) & 0\\ A & 0 & 0\end{bmatrix}$$



算法步骤

给定点x^0、满足$g(x)<0$，迭代步长σ、误差阈值ϵ

重复进行

1. 计算t:
2. 计算原对偶搜索方向
3. 直线搜索与更新
4. 判断收敛条件

相对中心跟踪的两层迭代，原对偶内点法只有一层迭代，中心跟踪内层需要通过牛顿法迭代求精确解，而原对偶只搜索一步，不求精确解

三、最优潮流算法综述

➢ 内点法的核心思想

- 用障碍函数把不等式约束处理到目标函数里，变成等式约束问题
- 用拉格朗日乘子法处理等式约束问题
- 选择一个满足所有约束的内点x^0和t作为起始点，使用一系列的障碍函数精度参数t，去逼近最优解，每次求出的作为下一步的初值x^n作为下一步初值，即沿中心路径搜索。
- 用牛顿法求解拉格朗日函数法形成的方程组得到x^n，或使用原对偶方向直接搜索近似x^n

总体上来说，原对偶法收敛速度要快于中心跟踪，尤其是精度提高时，迭代次数相对中心跟踪增加很少，因此目前最优潮流计算往往使用原对偶内点搜索法。

（matpower最优潮流缺省默认算法即为原对偶内点搜索法）

三、最优潮流算法综述

➢ 原对偶内点法求解步骤

第一步：将不等式约束转化为等式约束

数学模型

$$\min f(x)$$

$$s.t. \quad h(x) = 0$$

$$\underline{g} \le g(x) \le \overline{g}$$

其中：$x = [x_1, \cdots, x_n]^T$ n个变量

$h(x) = [h_1(x), \cdots, h_m(x)]^T$

$g(x) = [g_1(x), \cdots, g_r(x)]^T$

$\underline{g}, \overline{g}$ 非线性不等式约束的下限、上限

通过引入松弛变量u和l，将不等式约束转化为等式约束和松弛变量不等式

$$\min f(x)$$

$$s.t. \quad h(x) = 0$$

$$g(x) + u = \overline{g}$$

$$g(x) - l = \underline{g}$$

$$u > 0, l > 0$$

使用对数障碍函数消去松弛变量的非负性不等式约束：

$$\min f(x) - \mu \sum_{j=1}^{r} \log l_r - \mu \sum_{j=1}^{r} \log u_r$$

$$s.t. \quad h(x) = 0 \quad g(x) + u = \overline{g} \quad g(x) - l = \underline{g}$$

$$\mu > 0$$ （这个不等式的μ不是变量，而是拉格朗日因子）

也可以把不等式约束直接使用障碍函数计入，省略u和l松弛变量这一步

三、最优潮流算法综述

➢ 原对偶内点法求解步骤

第二步：构造拉格朗日函数，对变量及乘子求偏导数

引入拉格朗日乘子：$\boldsymbol{y}=[y_1,\cdots,y_m],\boldsymbol{z}=[z_1,\cdots,z_r],\boldsymbol{w}=[w_1,\cdots,w_r]$

得到拉格朗日函数：

$$L=f(x)-\boldsymbol{y}^{\mathrm{T}}\boldsymbol{h}(x)-\boldsymbol{z}^{T}\left[\mathbf{g}(x)-\boldsymbol{l}-\underline{\mathbf{g}}\right]-\boldsymbol{w}^{T}\left[\mathbf{g}(x)+\boldsymbol{u}-\overline{g}\right]-\mu\sum_{j=1}^{r}\log l_r-\mu\sum_{j=1}^{r}\log u_r$$

极值存在的必要条件：拉格朗日函数对所有变量及乘子的偏导数为0

$$L_x=\frac{\partial L}{\partial x}\equiv\nabla_x f(x)-\nabla_x\boldsymbol{h}(x)\boldsymbol{y}-\nabla_x\mathbf{g}(x)(\boldsymbol{z}+\boldsymbol{w})=\mathbf{0}$$

$$L_y=\frac{\partial L}{\partial y}\equiv\boldsymbol{h}(x)=\mathbf{0}$$

$$L_z=\frac{\partial L}{\partial z}\equiv\mathbf{g}(x)-\boldsymbol{l}-\underline{\mathbf{g}}=\mathbf{0}$$

$$L_w=\frac{\partial L}{\partial w}\equiv\mathbf{g}(x)+\boldsymbol{u}-\overline{\mathbf{g}}=\mathbf{0}$$

$$L_l=\frac{\partial L}{\partial l}\equiv\boldsymbol{z}-\mu\boldsymbol{L}^{-1}\boldsymbol{e}\quad\Rightarrow\ L_l^{\mu}=\boldsymbol{LZe}-\mu\boldsymbol{e}=\mathbf{0}$$

$$L_u=\frac{\partial L}{\partial u}\equiv-\boldsymbol{w}-\mu\boldsymbol{U}^{-1}\boldsymbol{e}\quad\Rightarrow\ L_u^{\mu}=\boldsymbol{UWe}+\mu\boldsymbol{e}=\mathbf{0}$$

其中：

$$\boldsymbol{L}=\boldsymbol{diag}(l_1,\cdots,l_r)$$
$$\boldsymbol{U}=\boldsymbol{diag}(u_1,\cdots,u_r)$$
$$\boldsymbol{Z}=\boldsymbol{diag}(z_1,\cdots,z_r)$$
$$\boldsymbol{W}=\boldsymbol{diag}(w_1,\cdots,w_r)$$

三、最优潮流算法综述

➢ 原对偶内点法求解步骤

第三步：牛顿法求解非线性方程组

对上组方程，写成修正方程如下：

$$
\begin{cases}
-\left[\nabla_x^2 f(x) - \nabla_x^2 h(x)y - \nabla_x^2 g(x)(z+w)\right]\Delta x + \nabla_x h(x)\Delta y + \nabla_x g(x)(\Delta z + \Delta w) = L_x \\
\nabla_x h(x)^T \Delta x = -L_y \\
\nabla_x g(x)^T \Delta x - \Delta l = -L_z \\
\nabla_x g(x)^T \Delta x + \Delta u = -L_w \\
Z\Delta l + L\Delta z = -L_l^{\mu} \\
W\Delta u + U\Delta w = -L_u^{\mu}
\end{cases}
$$

写成矩阵形式：(4r+m+n)× (4r+m+n)

原变量

$$
\begin{bmatrix}
H & \nabla_x h(x) & \nabla_x g(x) & \nabla_x g(x) & 0 & 0 \\
\nabla_x^T h(x) & 0 & 0 & 0 & 0 & 0 \\
\nabla_x^T g(x) & 0 & 0 & 0 & -I & 0 \\
\nabla_x^T g(x) & 0 & 0 & 0 & 0 & I \\
0 & 0 & L & 0 & Z & 0 \\
0 & 0 & 0 & U & 0 & W
\end{bmatrix}
\begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \\ \Delta w \\ \Delta l \\ \Delta u \end{bmatrix}
=
\begin{bmatrix} L_x \\ -L_y \\ -L_z \\ -L_w \\ -L_l^{\mu} \\ -L_u^{\mu} \end{bmatrix}
$$

对偶变量

其中：$H = -[\nabla_x^2 f(x) - \nabla_x^2 h(x)y - \nabla_x^2 g(x)(z+w)]$

三、最优潮流算法综述

③ 对角矩阵

$$\begin{bmatrix} I & L^{-1}Z & 0 & 0 & 0 & 0 \\ 0 & I & 0 & 0 & -\nabla_x^{\mathrm{T}} g(x) & 0 \\ 0 & 0 & I & U^{-1}\omega & 0 & 0 \\ 0 & 0 & 0 & I & \nabla_x^{\mathrm{T}} g(x) & 0 \\ 0 & 0 & 0 & 0 & H' & \nabla_x h(x) \\ 0 & 0 & 0 & 0 & \nabla_x^{\mathrm{T}} h(x) & 0 \end{bmatrix} \begin{bmatrix} \Delta z \\ \Delta l \\ \Delta w \\ \Delta u \\ \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} -L^{-1}L_l^{\mu} \\ L_z \\ -U^{-1}L_u^{\mu} \\ -L_w \\ L_x' \\ -L_v \end{bmatrix}$$

② 等式约束雅可比矩阵

① 不等式约束雅可比矩阵

式中：

$$L_x' = L_x + \nabla_x g(x)[L^{-1}(L_l^{\mu} + ZL_z) + U^{-1}(L_u^{\mu} - WL_w)]$$

$$H' = H - \nabla_x g(x)[L^{-1}Z - U^{-1}W]\nabla_x^{\mathrm{T}} g(x)$$

三、最优潮流算法综述

➢ 原对偶内点法求解步骤

第三步：牛顿法求解非线性方程组

求解上述修正方程，得到$\Delta \boldsymbol{x}, \Delta \boldsymbol{y}, \Delta \boldsymbol{z}, \Delta \boldsymbol{w}, \Delta \boldsymbol{l}, \Delta \boldsymbol{u}$,更新变量：

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^{k} + \alpha_p \Delta \boldsymbol{x}$$

原变量

$$\begin{aligned} \boldsymbol{l}^{k+1} &= \boldsymbol{l}^{k} + \alpha_p \Delta \boldsymbol{l} \\ \boldsymbol{u}^{k+1} &= \boldsymbol{u}^{k} + \alpha_p \Delta \boldsymbol{u} \\ \boldsymbol{y}^{k+1} &= \boldsymbol{y}^{k} + \alpha_d \Delta \boldsymbol{y} \\ \boldsymbol{z}^{k+1} &= \boldsymbol{z}^{k} + \alpha_d \Delta \boldsymbol{z} \\ \boldsymbol{w}^{k+1} &= \boldsymbol{w}^{k} + \alpha_d \Delta \boldsymbol{w} \end{aligned}$$

对偶变量

两个方向同时搜索更新

其中α_p, α_d为原始步长和对偶步长

$$\alpha_p = 0.9995 \min\left\{\min_i \left(\frac{-l_i}{\Delta l_i}, \Delta l_i < 0; \frac{-u_i}{\Delta u_i}, \Delta u_i < 0\right), 1\right\}$$

$$\alpha_d = 0.9995 \min\left\{\min_i \left(\frac{-z_i}{\Delta z_i}, \Delta z_i < 0; \frac{-w_i}{\Delta w_i}, \Delta w_i > 0\right), 1\right\}$$

上述步长的选取，严格保证$\boldsymbol{u} > \boldsymbol{0}, \boldsymbol{l} > \boldsymbol{0}$

三、最优潮流算法综述

➢ 原对偶内点法求解步骤

第四步：判断收敛条件和计算新的障碍因子

$$\begin{cases} L_l^{\mu} = LZe - \mu e = 0 \\ L_u^{\mu} = UWe + \mu e = 0 \end{cases} \Rightarrow \mu = \frac{l^T z - u^T w}{2r}$$

$2r$：不等式个数

定义互补间隙，用于判断是否收敛：

$$Gap = l^T z - u^T w$$

因此：

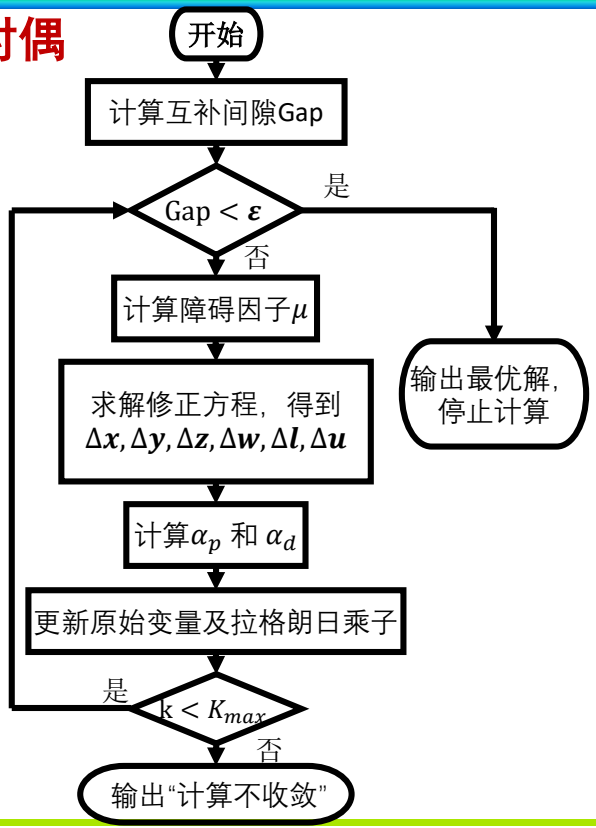
$$\mu = \frac{Gap}{2r}$$

中心参数$\sigma \in (0,1)$，一般设为0.1（对应前面$\sigma = 10 - 20$，此处与前面σ是倒数关系）

$$\mu = \sigma \frac{Gap}{2r} \qquad \sigma \in (0,1)$$

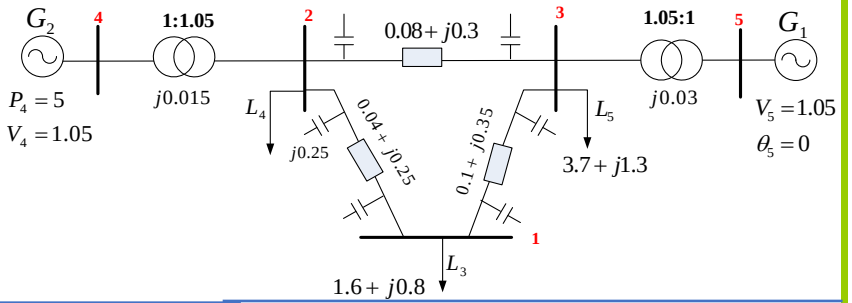
三、最优潮流算法综述

- **最优潮流的原对偶内点法流程**



三、最优潮流算法综述

➢ 算例分析



优化变量包括系统状态变量10个：

$$\tilde{x} = \{\theta_1\ V_1\ \theta_2\ V_2\ \theta_3\ V_3\ \theta_4\ V_4\ \theta_5\ V_5\}$$

控制变量4个：

$$\tilde{u} = \{P_{G1}\ P_{G2}\ Q_{G1}\ Q_{G1}\}\ ,$$

优化变量总计14个：

$$x = \{P_{G1}\ P_{G2}\ Q_{G1}\ Q_{G1}\ \theta_1\ V_1\ \theta_2\ V_2\ \theta_3\ V_3\ \theta_4\ V_4\ \theta_5\ V_5\}$$

支路号	线路首末母线号	线路传输功率边界
1	1-2	2
2	1-3	0.65
3	2-3	2
4	2-4	6
5	3-5	5

发电机序号	母线号	出力上限		出力下限		燃料耗费曲线特性		
		有功	无功	有功	无功	二次系数	一次系数	常数
1	4	8	3	1	-3	50.4395	200.4335	1200.6485
2	5	8	5	1	-2.1	200.55	500.746	1857.201

三、最优潮流算法综述

➢ 算例分析

目标函数: $\boldsymbol{min\ F(x)=\sum_{i=0}^{N} f_i(P_{Gi})}$ 其中: $\boldsymbol{f_i(P_{Gi})=c_iP_{Gi}^2+b_iP_{Gi}+a_i}$

有功发电费用最小

等式约束$h(x)$——节点功率平衡约束: （每个节点有2个潮流方程，共5个节点，10个方程）

$$\Delta P_i = P_{Gi} - P_{Di} - V_i \sum_{j=0}^{m} V_j\left(G_{ij}\cos\theta_{ij} + B_{ij}\sin\theta_{ij}\right) = 0$$

$$\Delta Q_i = Q_{Gi} - Q_{Di} - V_i \sum_{j=0}^{m} V_j\left(G_{ij}\sin\theta_{ij} - B_{ij}\cos\theta_{ij}\right) = 0$$

不等式约束 $g(x)$ ——运行边界约束: （共14个方程）

$\underline{P_{Gi}} \le P_{Gi} \le \overline{P_{Gi}}$ $i=(1,2)$，发电机有功运行边界约束，共两个发电机，两个方程

$\underline{Q_{Gi}} \le Q_{Gi} \le \overline{Q_{Gi}}$ $i=(1,2)$，发电机无功运行边界约束，共两个发电机，两个方程

$\underline{V_i} \le V_i \le \overline{V_i}$ $i=(1-5)$，节点电压运行边界约束，共五个节点，五个方程

$\underline{P_{ij}} \le P_{ij} \le \overline{P_{ij}}$ $i=(1-5)$，线路潮流运行边界约束，共五条线路，五个方程

三、最优潮流算法综述

➢ 算例分析

1)形成系数矩阵:

③ 对角矩阵

$$\begin{bmatrix} I & L^{-1}Z & 0 & 0 & 0 & 0 \\ 0 & I & 0 & 0 & -\nabla_x^{\mathrm{T}} g(x) & 0 \\ 0 & 0 & I & U^{-1}\omega & 0 & 0 \\ 0 & 0 & 0 & I & \nabla_x^{\mathrm{T}} g(x) & 0 \\ 0 & 0 & 0 & 0 & H' & \nabla_x h(x) \\ 0 & 0 & 0 & 0 & \nabla_x^{\mathrm{T}} h(x) & 0 \end{bmatrix} \begin{bmatrix} \Delta z \\ \Delta l \\ \Delta w \\ \Delta u \\ \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} -L^{-1}L_l^{\mu} \\ L_z \\ -U^{-1}L_u^{\mu} \\ -L_w \\ L_x' \\ -L_v \end{bmatrix}$$

② 等式约束雅可比矩阵

① 不等式约束雅可比矩阵

式中:

$$L_x' = L_x + \nabla_x g(x)[L^{-1}(L_l^{\mu} + ZL_z) + U^{-1}(L_u^{\mu} - WL_w)]$$

$$H' = H - \nabla_x g(x)[L^{-1}Z - U^{-1}W]\nabla_x^{\mathrm{T}} g(x)$$

三、最优潮流算法综述

➢ 算例分析

1)形成系数矩阵:

① 等式约束的雅可比矩阵:

$$\nabla_x h(x) = \begin{bmatrix} \dfrac{\partial h}{\partial P_G} \\ \dfrac{\partial h}{\partial Q_R} \\ \dfrac{\partial h}{\partial x} \end{bmatrix}_{14\times 10}$$

$$\frac{\partial h}{\partial x} = \begin{bmatrix} \frac{\partial \Delta P_1}{\partial \theta_1} & \frac{\partial \Delta Q_1}{\partial \theta_1} & \cdots & \frac{\partial \Delta P_5}{\partial \theta_1} & \frac{\partial \Delta Q_5}{\partial \theta_1} \\ \frac{\partial \Delta P_1}{\partial V_1} & \frac{\partial \Delta Q_1}{\partial V_1} & \cdots & \frac{\partial \Delta P_5}{\partial V_1} & \frac{\partial \Delta Q_5}{\partial V_1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \frac{\partial \Delta P_1}{\partial \theta_5} & \frac{\partial \Delta Q_1}{\partial \theta_5} & \cdots & \frac{\partial \Delta P_5}{\partial \theta_5} & \frac{\partial \Delta Q_5}{\partial \theta_5} \\ \frac{\partial \Delta P_1}{\partial V_5} & \frac{\partial \Delta Q_1}{\partial V_5} & \cdots & \frac{\partial \Delta P_5}{\partial V_5} & \frac{\partial \Delta Q_5}{\partial V_5} \end{bmatrix}_{10\times 10}$$

$$\frac{\partial h}{\partial P_G} = \begin{bmatrix} \frac{\partial \Delta P_1}{\partial P_{G1}} & \frac{\partial \Delta Q_1}{\partial P_{G1}} & \cdots & \frac{\partial \Delta P_5}{\partial P_{G1}} & \frac{\partial \Delta Q_5}{\partial P_{G1}} \\ \frac{\partial \Delta P_1}{\partial P_{G2}} & \frac{\partial \Delta Q_1}{\partial P_{G2}} & \cdots & \frac{\partial \Delta P_5}{\partial P_{G2}} & \frac{\partial \Delta Q_5}{\partial P_{G2}} \end{bmatrix}_{2\times 10}$$

$$\frac{\partial h}{\partial Q_R} = \begin{bmatrix} \frac{\partial \Delta P_1}{\partial Q_{R1}} & \frac{\partial \Delta Q_1}{\partial Q_{R1}} & \cdots & \frac{\partial \Delta P_5}{\partial Q_{R1}} & \frac{\partial \Delta Q_5}{\partial Q_{R1}} \\ \frac{\partial \Delta P_1}{\partial Q_{R2}} & \frac{\partial \Delta Q_1}{\partial Q_{R2}} & \cdots & \frac{\partial \Delta P_5}{\partial Q_{R2}} & \frac{\partial \Delta Q_5}{\partial Q_{R2}} \end{bmatrix}_{2\times 10}$$

$$\begin{cases} \dfrac{\partial \Delta Q_j}{\partial P_{Gi}} = 0 \\ \dfrac{\partial \Delta P_j}{\partial P_{Gi}} = \begin{cases} -1 & (i \in j) \\ 0 & (i \notin j) \end{cases} \end{cases} \qquad \begin{cases} \dfrac{\partial \Delta P_j}{\partial Q_{Ri}} = 0 \\ \dfrac{\partial \Delta Q_j}{\partial Q_{Ri}} = \begin{cases} -1 & (i \in j) \\ 0 & (i \notin j) \end{cases} \end{cases}$$

三、最优潮流算法综述

➢ 算例分析

② 不等式约束的雅可比矩阵:

$$\nabla_x g(x) = \begin{bmatrix} \frac{\partial g_1}{\partial P_G} & \frac{\partial g_2}{\partial P_G} & \frac{\partial g_3}{\partial P_G} & \frac{\partial g_4}{\partial P_G} \\ \frac{\partial g_1}{\partial Q_R} & \frac{\partial g_2}{\partial Q_R} & \frac{\partial g_3}{\partial Q_R} & \frac{\partial g_4}{\partial Q_R} \\ \frac{\partial g_1}{\partial x} & \frac{\partial g_2}{\partial x} & \frac{\partial g_3}{\partial x} & \frac{\partial g_4}{\partial x} \end{bmatrix}_{14\times 14}$$

③ 对角矩阵:

$$L^{-1}Z = \mathrm{diag}(z_1/l_1, \cdots, z_{14}/l_{14})$$
$$U^{-1}W = \mathrm{diag}(w_1/u_1, \cdots, w_{14}/u_{14})$$

④ 海森伯矩阵:

$$H' = -\nabla_x^2 f(x) + \nabla_x^2 h(x) y + \nabla_x^2 g(x)(z+w) - \nabla_x g(x)[L^{-1}Z - U^{-1}W]\nabla_x^{\mathrm{T}} g(x)$$

三、最优潮流算法综述

➢ 算例分析

2)形成常数项:

L_y、L_z、L_w、L_l^μ 和 L_u^μ 都很容易求出

$$L_x' = \frac{\partial L}{\partial x} \equiv \nabla_x f(x) - \nabla_X h(x) y - \nabla_x g(x)(z + w) + \nabla_x g(x)[L^{-1}(L_l^\mu + ZL_z) + U^{-1}(L_u^\mu - WL_w)]$$

至此，与例题有关的公式已全部推导完毕。

三、最优潮流算法综述

➢ 算例分析

算例的寻优过程：

设4、5节点在发电机均能由算法调节其出力，在初始化过程中各变量初值是根据实际问题自行设置的。当收敛条件取 $\varepsilon = 10^{-6}$时，需要进行 17 次迭代。将各次迭代过程中 Gap 变化情况绘制成曲线，可以显示出跟踪中心轨迹内点法最优潮流的收敛特性，见图 3-3。

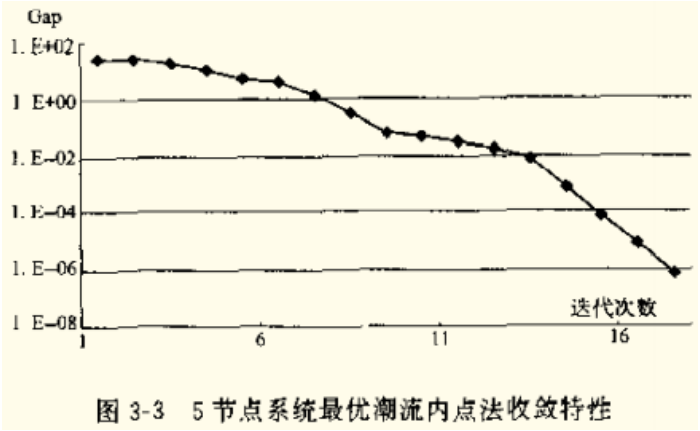


图 3-3　5 节点系统最优潮流内点法收敛特性

三、最优潮流算法综述

➢ 算例分析

结果分析:

计算结果与原潮流计算结果比较见表 3-9 到表 3-11 。从表中看出，由于机组比组的燃科耗费曲线系数小，因此机组有功出力增加，机组有功出力减少。同时系统的网损、无功出力都有所增加。这是由要将节点电压抬高至其下界以满足不等式约束的要求而产生的副作用。但是网损的增加并不影响目标函数的优化，整个旱统的燃科费用与 不优化的潮流计算相比仍然减少了 243. 76 $。

表 3－9　各有功源有功及无功源无功出力

发电机序号	母线序号	有功出力		无功出力		燃料费用/$	
		OPF	PF	OPF	PF	OPF	PF
1	4	5. 505 6	5. 000 0	1. 778 0	1. 831 1	3 833. 06	3 463. 80
2	5	2. 156 8	2. 579 4	2. 619 4	2. 299 4	3 870. 13	4 483. 15
总计		7. 662 4	7. 579 4	4. 397 4	4. 130 5	7 703. 19	7 946. 95

表 3－10　各节点电压向量

母线序号	电压幅值		电压相角/rad	
	OPF	PF	OPF	PF
1	0. 900 00	0. 862 2	−0. 006 97	−0. 083 40
2	1. 100 00	1. 077 9	0. 404 91	0. 311 60
3	1. 081 75	1. 036 4	−0. 057 126	−0. 074 73
4	1. 069 70	1. 050 00	0. 478 67	0. 311 60
5	1. 100 00	1. 050 00	0	0

表 3－11　支路有功功率

支路号	首末端母线号	支路有功功率			
		P_{ij}		P_{ji}	
		OPF	PF	OPF	PF
1	1-2	−1. 606 4	−1. 466 2	1. 734 7	1. 584 5
2	1-3	−0. 006 4	−0. 133 8	−0. 020 3	0. 156 9
3	2-3	1. 770 9	1. 415 5	−1. 563 5	−1. 277 4
4	2-4	−5. 505 6	−5	5. 505 6	5
5	3-5	−2. 156 8	−2. 579 4	2. 156 8	2. 579 4

三、最优潮流算法综述

➢ 算例分析

结果分析:

如果固定发电机组 的有功出力为5，最优潮流计算只能起到减少网损、优化系统无 功的作用。从表 3-12 到表 3-14 的结果可以看出，系统的网损减少了0.0178，即 1.78MW，从而整个系统的燃科费用减少了 27.27 $，节点的电压抬高至 0.9129，整个 系统无功出力减少 0.2339，即 23.39 MVA。

表 3－12　各有功源有功及无功源无功出力

发电机序号	母线序号	有功出力		无功出力		燃料费用/$	
		OPF	PF	OPF	PF	OPF	PF
1	4	5.000 0	5.000 0	2.358 5	1.831 1	3 463.80	3 463.80
2	5	2.561 6	2.579 4	1.538 1	2.299 4	4 455.88	4 483.15
总计		7.561 6	7.579 4	3.896 6	4.130 5	7 919.68	7 946.95

表 3－13　各节点电压向量

母线序号	电压幅值		电压相角/rad	
	OPF	PF	OPF	PF
1	0.912 9	0.862 2	−0.069 17	−0.083 4
2	1.100 0	1.077 9	0.300 03	0.311 60
3	1.085 5	1.036 4	0.067 87	−0.074 73
4	1.066 9	1.050 0	0.367 18	0.381 23
5	1.096 0	1.050 0	0	0

表 3－14　支路有功功率

支路号	首末端母线号	支路有功功率			
		P_{ij}		P_{ji}	
		OPF	PF	OPF	PF
1	1-2	−1.477 7	−1.466 2	1.584 0	1.584 5
2	1-3	−0.122 3	−0.133 8	0.144 8	0.156 9
3	2-3	1.416 0	1.415 5	−1.283 2	1.277 4
4	2-4	−5	−5	5	5
5	3-5	−2.561 6	−2.579 4	2.561 6	2.579 4

三、最优潮流算法综述

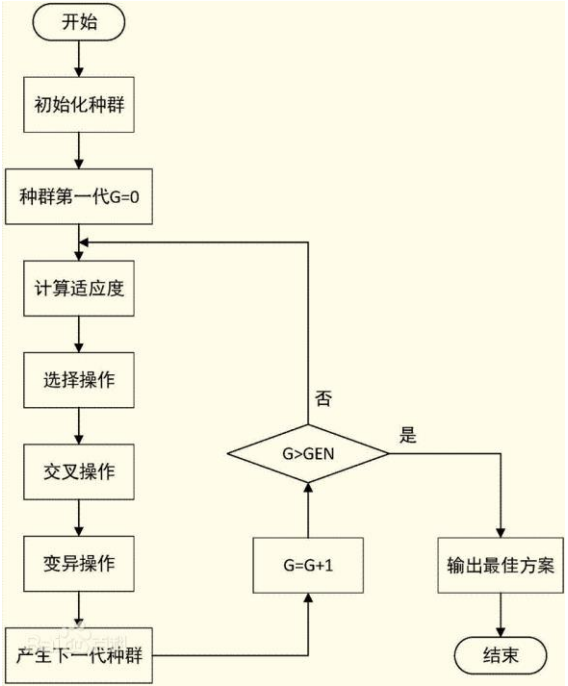
➢ 人工智能方法

- 近几年随着计算机和人工智能等技术的发展，不断有新的方法出现，模拟进化规划方法、模糊集理论、模拟退火算法等人工智能方法先后用于电力系统最优潮流问题。
- 人工智能方法解决了寻找全局最优解的问题，能精确处理问题中离散变量，但由于这类方法通常属于随机搜索方法，具有计算速度慢的先天缺陷，难以适应在线计算及电力市场的要求。

三、最优潮流算法综述

➢ 遗传算法

是模拟达尔文生物进化论的自然选择和遗传学机理的生物进化过程的计算模型，是一种通过模拟自然进化过程搜索最优解的方法。该算法通过数学的方式，利用计算机仿真运算，将问题的求解过程转换成类似生物进化中的染色体基因的交叉、变异等过程。在求解较为复杂的组合优化问题时，相对一些常规的优化算法，通常能够较快地获得较好的优化结果。



三、最优潮流算法综述

➢ 遗传算法

遗传算法的基本运算过程如下：

（1）**初始化**：设置进化代数计数器t=0，设置最大进化代数T，随机生成M个个体作为初始群体P(0)。

（2）**个体评价**：计算群体P(t)中各个个体的适应度。

（3）**选择运算**：将选择算子作用于群体。选择的目的是把优化的个体直接遗传到下一代或通过配对交叉产生新的个体再遗传到下一代。选择操作是建立在群体中个体的适应度评估基础上的。

（4）**交叉运算**：将交叉算子作用于群体。遗传算法中起核心作用的就是交叉算子。

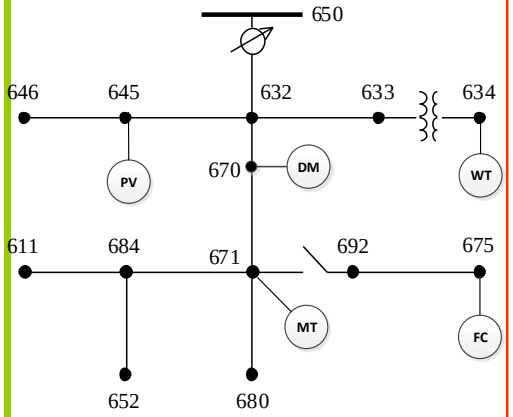
（5）**变异运算**：将变异算子作用于群体。即是对群体中的个体串的某些基因座上的基因值作变动。群体P(t)经过选择、交叉、变异运算之后得到下一代群体P(t+1)。

（6）**终止条件判断**：若t=T,则以进化过程中所得到的具有最大适应度个体作为最优解输出，终止计算。遗传操作包括以下三个基本遗传算子(genetic operator):选择(selection)；交叉(crossover)；变异(mutation)。

三、最优潮流算法综述

➢ 基于遗传算法的OPF问题求解

问题描述（例）



IEEE-13节点标准测试系统

OPF模型：

$$\min TC = \sum_{t=1}^{T}\left\{\sum_{i=1}^{N}\left[C_f(P_{gi}^t)+C_{om}(P_{gi}^t)+C_e(P_{gi}^t)\right]+\left(C_b(P_{ext}^t)-I_s(P_{ext}^t)\right)\right\}$$

$$s.t.\begin{cases}\sum_{i=1}^{N_D} P_{gi}^t + P_{pv}^t + P_{wt}^t + P_{ext}^t + P_{loss}^t = P_{load}^t \\ P_{gi}^{\min} \le P_{gi}^t \le P_{gi}^{\max} \quad i \in \{1 \ldots N_D\} \\ P_{ext}^{\min} \le P_{ext}^t \le P_{ext}^{\max} \\ V_j^{\min} \le V_j \le V_j^{\max} \quad j \in \{1 \ldots S_N\}\end{cases}$$

- **目标：**微网运行总费用最低（机组燃耗费用，维护费用，污染排放折算费用，购电费用）
- **约束：**发电机输出功率约束，线路潮流约束，节点电压约束等
- 采用MATLAB与Opendss混合调度平台进行迭代求解

三、最优潮流算法综述

➢ 基于遗传算法的OPF问题求解

主函数及遗传算法的参数设置

```
% Number of variables
nvars = 3;
% Lower bound & Upper bound
UB = [600 300 500];
LB = [0 0 0];

ObjectiveFunction = @GA_fitness;
ConstraintFunction = @GA_constraint;

options=gaoptimset('PopulationSize',100,'PopInitRange',[LB;UB],'EliteCount',10,'CrossoverFraction',0.6,'Generations',100,...
    'StallGenLimit',100,'TolFun',1e-100,'PlotFcns',{@gaplotbestf,@gaplotbestindiv}); % 算法参数设置
[x,fval] = ga(ObjectiveFunction,nvars,[],[],[],[],LB,UB,ConstraintFunction,options);

myopt.global_min_point = x;
myopt.global_min=fval;
```

变量数及变量上下边界设置

指定目标函数和约束条件对应的函数文件的句柄

GA参数的设置（交叉概率、变异概率，种群数等）

输出最优目标解和目标解对应的方案

三、最优潮流算法综述

➢ 基于遗传算法的OPF问题求解

约束条件（对应于主函数中ConstraintFunction）

```
function [c, ceq]=GA_constraint(x)
DSS_Global;
DSS_Do_Update_by_Text_Command(x);   %下发随机产生的机组功率组合点
DSSCircuit.Solution.Solve; % 潮流计算
PCCPower = sum(myLoads.kW)+myExtract.Losses(1)/1000-sum(myGenerators.kW); %计算联络线交换功率
V1 = DSSCircuit.AllNodeVmagPUByPhase(1); %计算各节点A相电压
V2 = DSSCircuit.AllNodeVmagPUByPhase(2); %计算各节点B相电压
V3 = DSSCircuit.AllNodeVmagPUByPhase(3); %计算各节点C相电压

% 联络线交换功率约束
c(:,1)=PCCPower-1000;
c(:,2)=-PCCPower;
% 节点电压约束
Vamax = max(V1(1,:));
Vamin = min(V1(1,:));
Vbmax = max(V2(1,:));
Vbmin = min(V2(1,:));
Vcmax = max(V3(1,:));
Vcmin = min(V3(1,:));
c(:,3) = Vamax-1.1;
c(:,4) = 0.9-Vamin;
c(:,5) = Vbmax-1.1;
c(:,6) = 0.9-Vbmin;
c(:,7) = Vcmax-1.1;
c(:,8) = 0.9-Vcmin;

ceq = [];
```

下发随机生成解

潮流计算，得到当前随机解对应的系统潮流数据

节点电压约束、联络线潮流约束

3.1 按处理约束的方法分类
3.2 按修正的变量空间分类
3.3 按变量修正的方向分类
3.4 三类最优潮流算法的三维分类图形表示
3.5 广义拉格朗日乘子法
3.10 内点算法
3.11 人工智能方法
3.12进化类算法

三、最优潮流算法综述

目标函数（对应于主函数中ObjectiveFunction）

```
function y = GA_fitness(x)
DSS_Do_Update_by_Text_Command(x);
DSSCircuit.Solution.Solve;
pccpower=sum(myLoads.kW)+DSSCircuit.Losses(1)/1000-sum(myGenerators.kW);
% 定义各类机组的维护管理系数(元/kW.h)
Kom_PV=0.0096;Kom_WI=0.0296;Kom_FC=0.0286;Kom_MI=0.0401;Kom_DE=0.0859;
% 定义污染治理费用系数(元/kg)
Kec_CO2=0.21;Kec_SO2=14.842;Kec_NOX=62.964;
% 定义各类机组的污染排放系数(g/kW.h)
Kef_FC_CO2=489;Kef_FC_SO2=0.003;Kef_FC_NOX=0.01;Kef_MI_CO2=724;Kef_MI_SO2=0.004;Kef_MI_NOX=0.1467;
Kef_DE_CO2=649;Kef_DE_SO2=0.206;Kef_DE_NOX=9.89;Kef_GRID_CO2=889;Kef_GRID_SO2=1.8;Kef_GRID_NOX=1.6;
% 定义各类燃耗机组的燃耗系数(元/kW.h)
Kf_FC=0.59415;Kf_MI=0.38445;Kf_DE_a=0.05032;Kf_DE_b=1.58644;Kf_DE_c=2.94644;
% 定义投资折旧系数
Caz_PV=2;Caz_WI=1.2;Caz_FC=2.8;Caz_MI=1;Caz_DE=0.8;Caz_BAI=0.0667;
r=0.05;L_PV=20;L_WI=10;L_FC=10;L_MI=10;L_DE=10;L_BAI=10;
% 定义电价(元/kw.h)
Cb=Climate.powerprice(myloop.present_step)*6;% Cs=0.65;
%**********燃耗成本*******
Cfuel=(Kf_DE_a*x(1).^2+Kf_DE_b*x(1)+Kf_DE_c)+Kf_FC.*x(2)+Kf_MI.*x(3);
%**********维护成本*******
Com=Kom_WI*Climate.wind(myloop.present_step)+Kom_DE.*x(1)+Kom_FC.*x(2)+Kom_MI.*x(3);
%**********折旧成本*******
Cdp_PV=10000*Caz_PV*r*((1+r).^L_PV)*Climate.pv(myloop.present_step)/8760/((1+r).^L_PV-1);
Cdp_WI=10000*Caz_WI*r*((1+r).^L_WI)*Climate.wind(myloop.present_step)/8760/((1+r).^L_WI-1);
Cdp_DE=10000*Caz_DE*r*((1+r).^L_DE)*x(1)/8760/((1+r).^L_DE-1);
Cdp_FC=10000*Caz_FC*r*((1+r).^L_FC)*x(2)/8760/((1+r).^L_FC-1);
Cdp_MI=10000*Caz_MI*r*((1+r).^L_MI)*x(3)/8760/((1+r).^L_MI-1);
Cdp=Cdp_WI+Cdp_DE+Cdp_FC+Cdp_MI;
%**********排放惩罚成本*******
Cec_DE=Kef_DE_CO2*x(1)/1000*Kec_CO2+Kef_DE_SO2*x(1)/1000*Kec_SO2+Kef_DE_NOX*x(1)/1000*Kec_NOX;
Cec_FC=Kef_FC_CO2*x(2)/1000*Kec_CO2+Kef_FC_SO2*x(2)/1000*Kec_SO2+Kef_FC_NOX*x(2)/1000*Kec_NOX;
Cec_MI=Kef_MI_CO2*x(3)/1000*Kec_CO2+Kef_MI_SO2*x(3)/1000*Kec_SO2+Kef_MI_NOX*x(3)/1000*Kec_NOX;
Cec_GRID=Kef_GRID_CO2*abs(pccpower)/1000*Kec_CO2+Kef_GRID_SO2*abs(pccpower)/1000*Kec_SO2+Kef_GRID_NOX*abs(pccpower)/1000*Kec_NOX;
Cec=Cec_DE+Cec_FC+Cec_MI+Cec_GRID;
%**********购电成本及售电收益*******
Csp=Cb*pccpower;
%**********网损费用*******
Closs=Cb*DSSCircuit.Losses(1)/1000;
Cost=Cfuel+Com+Cdp+Cec+Csp+Closs;
y = Cost;
```

燃耗费用

维护费用

折旧费用

排放惩罚费用

购电费用

三、最优潮流算法综述

➢ 粒子群算法

- 粒子群算法的思想源于对鸟群捕食行为的研究．
- 模拟鸟集群飞行觅食的行为，鸟之间通过集体的协作使群体达到最优目的，是一种基于Swarm Intelligence的优化方法。
- 马良教授在他的著作《蚁群优化算法》一书的前言中写到：

“自然界的蚁群、鸟群、鱼群、羊群、牛群、蜂群等，其实时时刻刻都在给予我们以某种启示，只不过我们常常忽略了大自然对我们的最大恩赐！.....”

三、最优潮流算法综述

➢ 粒子群算法的基本思想

设想这样一个场景：一群鸟在随机搜索食物

已知
- 在这块区域里只有一块食物;
- 所有的鸟都不知道食物在哪里;
- 但它们能感受到当前的位置离食物还有多远.

那么:找到食物的最优策略是什么呢?

搜寻目前离食物最近的鸟的周围区域 .
根据自己飞行的经验判断食物的所在。

PSO正是从这种模型中得到了启发.

PSO的基础：信息的社会共享

三、最优潮流算法综述

➢ 粒子群算法介绍

- 每个寻优的问题解都被想像成一只鸟，称为“粒子”。所有粒子都在一个D维空间进行搜索。
- 所有的粒子都由一个fitness function 确定适应值以判断目前的位置好坏。
- 每一个粒子必须赋予记忆功能，能记住所搜寻到的最佳位置。
- 每一个粒子还有一个速度以决定飞行的距离和方向。这个速度根据它本身的飞行经验以及同伴的飞行经验进行动态调整。

三、最优潮流算法综述

➢ 粒子群算法介绍

1. ***Initial:***

   初始化粒子群体（群体规模为n），包括随机位置和速度。

2. ***Evaluation:***

   根据fitness function，评价每个粒子的适应度。

3. ***Find the Pbest:***

   对每个粒子，将其当前适应值与其个体历史最佳位置（pbest）对应的适应值做比较，如果当前的适应值更高，则将用当前位置更新历史最佳位置pbest。

4. ***Find the Gbest:***

   对每个粒子，将其当前适应值与全局最佳位置（gbest）对应的适应值做比较，如果当前的适应值更高，则将用当前粒子的位置更新全局最佳位置gbest。

5. ***Update the Velocity:***

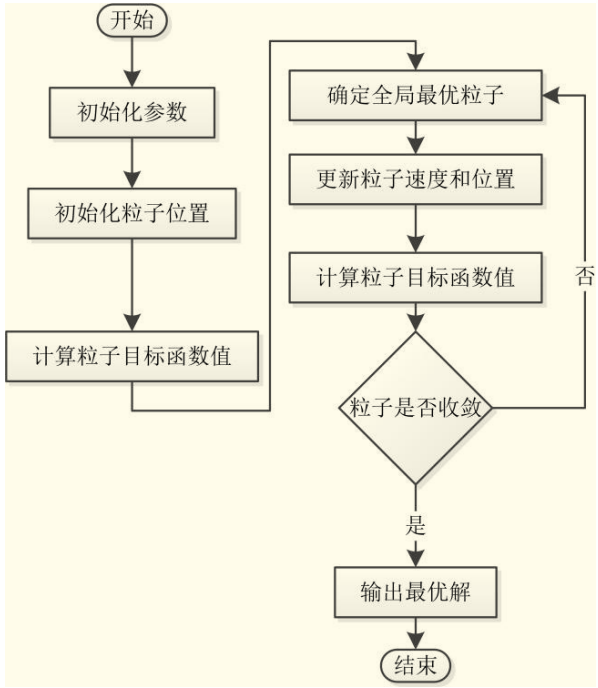
   根据公式更新每个粒子的速度与位置。

6. 如未满足结束条件，则返回步骤2

通常算法达到最大迭代次数G_{max}或者最佳适应度值的增量小于某个给定的阈值时算法停止。

三、最优潮流算法综述

- 粒子群算法的基本流程



三、最优潮流算法综述

➢ 进化类算法

- **优势**

进化类算法无需进行假设和近似，无需目标函数的导数信息，且具有内在并行性，能处理非连续的、非光滑的、高度非线性解空间的复杂优化问题，提高了处理复杂优化问题的速度和精度，展现了良好的收敛特性和全局寻优能力。进化类算法目前已经在电力系统最优潮流中得到了成功应用。

- **求解原理**

进化类算法对于约束处理一般较困难，可首先基于罚函数原理将约束条件以罚函数的方式转换到目标函数中，只保留边界约束

$$\min f(x)$$

$$s.t.\ \ h_k(x) = 0$$

$$g_j(x) \le 0$$

x为待求向量，$(x_1, x_2, \dots, x_n)^T$

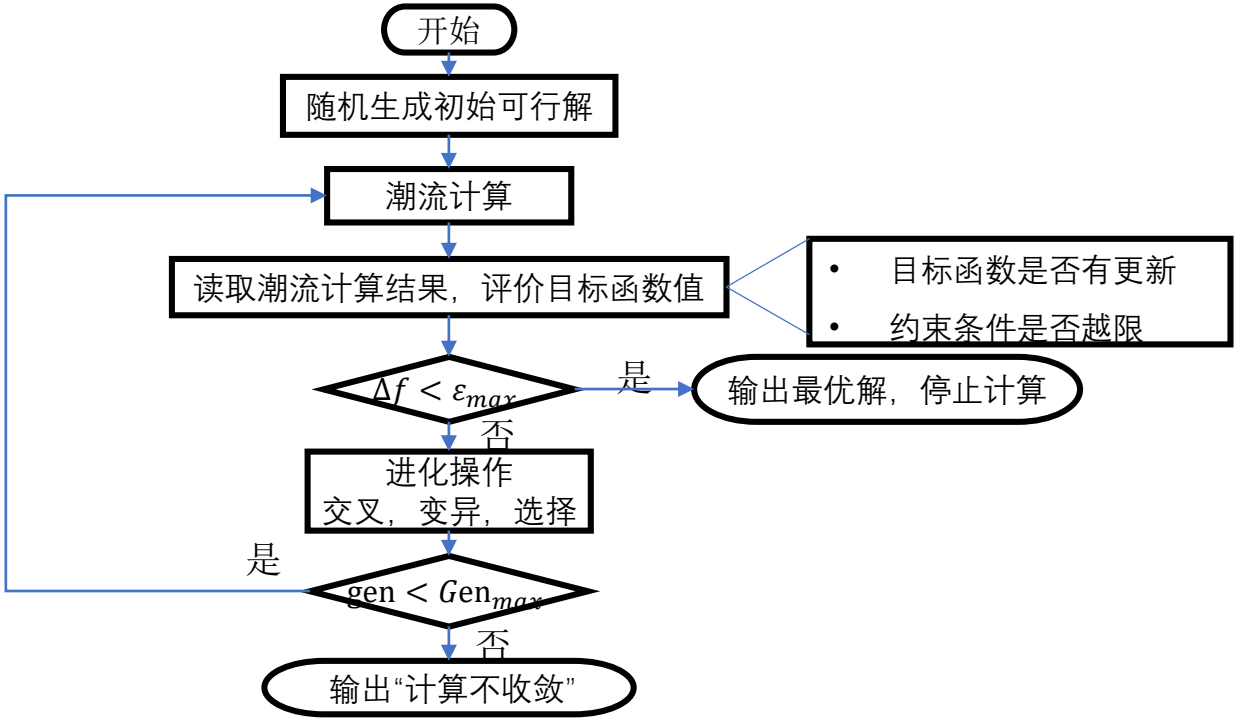
$$\min\ f(x) + \sum_{i=1}^{m} \gamma_i(h_i(x)) + \sum_{j=1}^{n} I_-(g_j(x))$$

γ_i为较大的正数

$$I_-(g_i(x)) = \begin{cases} 0, & g_i(x) \le 0 \\ \infty, & g_i(x) > 0 \end{cases}$$

三、最优潮流算法综述

➢ 基于进化类算法求解OPF一般流程



五、解耦最优潮流计算

5.1有功子优化问题
5.2无功子优化问题

- 不是象FDLF算法上的解耦
- 解耦最优潮流是从问题的本身或问题的模型上把最优潮流的最优化问题分解为有功优化和无功优化两个子优化问题
- 模型的建立
  - 将控制变量$\boldsymbol{u}$分成$\boldsymbol{u}_p$及$\boldsymbol{u}_q$两组，状态变量$\boldsymbol{x}$分为$\boldsymbol{x}_p$及$\boldsymbol{x}_q$两组。
    - $\boldsymbol{u}_p$为除平衡节点外，其它发电机的有功出力
    - $\boldsymbol{x}_p$为除平衡节点外，其它所有节点的电压相角
    - $\boldsymbol{u}_q$为所有发电机节点(含平衡)及有无功补偿设备节点的电压模值，此外还要调压变压器变比。
    - $\boldsymbol{x}_q$为除上述$\boldsymbol{u}_q$中所列节点以外的其余节点的电压模值。
  - 等式及不等式约束也可以分成$\boldsymbol{g}_p$、$\boldsymbol{g}_q$及$\boldsymbol{h}_p$、$\boldsymbol{h}_q$两组。

五、解耦最优潮流计算

➢ 有功子优化问题

- 通常用全系统的发电燃料总耗量或总费用 $f=\sum_{i\in NG}K_i(P_{Gi})$ 作为目标函数，
- 与无功有关的控制变量u_q及状态变量x_q均作为不变的常数处理，设用u_q^0及x_q^0表示，于是有功子优化的数学模型可写成如下的普遍形式

$$\left.\begin{aligned}&\min_{u_p} f_p=\min_{u_p} f_p(u_p,x_p,u_q^0,x_q^0)\\ &s.t.\quad g_p(u_p,x_p,u_q^0,x_q^0)=0\\ &\qquad h_p(u_p,x_p,u_q^0,x_q^0)\le 0\end{aligned}\right\}\qquad (36)$$

式中：等约束g_p=0表示节点有功功率方程组；不等式约束h_p可包括以上提到的有关控制变量u_p及状态变量x_p的不等式约束等。

五、解耦最优潮流计算

➢ 无功子优化问题

- 目前较多以系统的有功损耗 $\min f = \min P_s(\boldsymbol{U},\boldsymbol{\theta})$ 作为目标函数。把控制变量$\boldsymbol{u}_p$及状态变量$\boldsymbol{x}_p$均作为不变的常数处理，并以$\boldsymbol{u}_q^0$及$\boldsymbol{x}_q^0$表示。无功子优化问题的数学模型写为

$$\left.\begin{aligned} &\min_{u_q} f_q = \min_{u_q} f_q(\boldsymbol{u}_q,\boldsymbol{x}_q,\boldsymbol{u}_p^0,\boldsymbol{x}_p^0) \\ &s.t. \quad \boldsymbol{g}_q(\boldsymbol{u}_q,\boldsymbol{x}_q,\boldsymbol{u}_p^0,\boldsymbol{x}_p^0) = 0 \\ &\qquad \boldsymbol{h}_p(\boldsymbol{u}_q,\boldsymbol{x}_q,\boldsymbol{u}_p^0,\boldsymbol{x}_p^0) \le 0 \end{aligned}\right\} \qquad (37)$$

式中：等约束$\boldsymbol{g}_q$=0表示节点无功功率方程组；不等式约束$\boldsymbol{h}_q$中除包括对以上提到的有关控制变量$\boldsymbol{u}_q$及状态变量$\boldsymbol{x}_q$的限制外，还可以包括能表示成$\boldsymbol{u}_q$及$\boldsymbol{x}_q$函数的平衡节点的无功功率及线路无功潮流等不等式约束条件。

五、解耦最优潮流计算

- 求解过程：
- 以上有功、无功子优化问题可以独立求解
- 而要达到有功、无功综合优化的解耦最优潮流计算则要交替地迭代求解这两个子问题，步骤如下：

(1) 通过初始潮流计算，设定$\boldsymbol{u}_q^{(0)}$、$\boldsymbol{x}_q^{(0)}$、$\boldsymbol{u}_p^{(0)}$、$\boldsymbol{x}_p^{(0)}$。

(2) 令$\boldsymbol{u}_q^0=\boldsymbol{u}_q^{(0)}$，$\boldsymbol{x}_q^0=\boldsymbol{x}_q^{(0)}$,迭代记数$k=1$。

(3) 保持$\boldsymbol{u}_q^0$及$\boldsymbol{x}_q^0$不变，解有功子优化问题，得到$\boldsymbol{u}_p$的最优值$\boldsymbol{u}_p^{*(k)}$及相应$\boldsymbol{x}_p^{*(k)}$。

(4) 令$\boldsymbol{u}_p^0=\boldsymbol{u}_p^{*(k)}$，$\boldsymbol{x}_p^0=\boldsymbol{x}_p^{*(k)}$。

(5) 保持$\boldsymbol{u}_p^0$及$\boldsymbol{x}_p^0$不变，解无功子优化问题，得到$\boldsymbol{u}_q$的最优值$\boldsymbol{u}_q^{*(k)}$及相应$\boldsymbol{x}_q^{*(k)}$。

(6) 检验$\|\boldsymbol{u}_q^{*(k)}-\boldsymbol{u}_q^{*(k-1)}\|<\varepsilon$, $\|\boldsymbol{u}_p^{*(k)}-\boldsymbol{u}_p^{*(k-1)}\|<\varepsilon$是否满足。

(7) 若满足上列收敛条件，计算结束；否则令$\boldsymbol{u}_q^0=\boldsymbol{u}_q^{*(k)}$，$\boldsymbol{x}_q^0=\boldsymbol{x}_q^{*(k)}$。

(8) $k=k+1$，转向步骤3。

五、解耦最优潮流计算

- 算法分析
- 解耦最优潮流法的另一个优点在于容许两个子优化问题根据各自的特性而采用不同的求解算法。