1 无约束优化 [4]

无约束优化问题可表示为

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}), \qquad \boldsymbol{x} = (x_1, \ldots, x_n)^T \in \mathbb{R}^n \tag{1.1}$$

f 为非线性函数, 则1.1 称为无约束非线性规划. 实际上是一个多元函数无条件极值问题.

极值问题的解都是巨构最优解, 全局最优解只能从局部最优解的比较中得到.

若$f(\boldsymbol{x})$ 二次可导, 则记一阶导数$\nabla f(\boldsymbol{x}) = (f_{x_1}, \ldots, f_{x_n})^T$, 二阶导数记为$\nabla^2 f = (f_{x_i x_j})_{n\times n}$, 即Hessian 矩阵. 多元函数极值条件为:

$$\nabla f(\boldsymbol{x}) = 0, \text{且}\nabla^2 f(\boldsymbol{x})\text{正定} \tag{1.2}$$

使用迭代法求极值: 选取初始解x_0, 对第k 次迭代解, 确定搜索方向$\boldsymbol{d}_k \in \mathbb{R}^n$和步长$a_k \in \mathbb{R}$ 令

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + a_k \boldsymbol{d}_k \tag{1.3}$$

使

$$f(\boldsymbol{x}_{k+1}) < f(\boldsymbol{x}_k) \tag{1.4}$$

当$\boldsymbol{x}_k$ 满足终止条件时终止.

1.1 确定搜索方向

1.1.1 最速下降法(梯度法)

作泰勒展开:

$$\begin{aligned} f(\boldsymbol{x}_{k+1}) &= f(\boldsymbol{x}_k + a_k \boldsymbol{d}_k) \\ &= f(\boldsymbol{x}_k) + a_k \nabla f(\boldsymbol{x}_k)^T \boldsymbol{d}_k + o(\|a_k \boldsymbol{d}_k\|^2) \end{aligned} \tag{1.5}$$

取于是取$\boldsymbol{d^k} = -\nabla f(\boldsymbol{x}^k)$ 可使目标下降最快, 这种方法就是最速下降法.

1.1.2 牛顿法

作二阶展开:

$$\begin{aligned} f(\boldsymbol{x}_{k+1}) &= f(\boldsymbol{x}_k + a_k \boldsymbol{d}_k) \\ &= f(\boldsymbol{x}_k) + a_k \nabla f(\boldsymbol{x}_k)^T \boldsymbol{d}_k + \frac{1}{2} a_k \boldsymbol{d}_k^T \nabla^2 f(\boldsymbol{x}_k) \boldsymbol{d}_k + o(\|a_k \boldsymbol{d}_k\|^3) \end{aligned} \tag{1.6}$$

若$\nabla^2 f(\boldsymbol{x}_k)$ 正定, 则余项$a_k \nabla f(\boldsymbol{x}_k)^T \boldsymbol{d}_k + \frac{1}{2} a_k^2 \boldsymbol{d}_k^T \nabla^2 f(\boldsymbol{x}_k) \boldsymbol{d}_k$ 为$\boldsymbol{d}_k$ 二次型, 必可取到最小值.

且在$\boldsymbol{d}_k = -a_k (\nabla^2 f(\boldsymbol{x}_k))^{-1} \nabla f(\boldsymbol{x}_k)$ 处取到. 这就是牛顿法.

1.1.3 拟牛顿法

$(\nabla^2 f(\boldsymbol{x}_k))^{-1}$ 的不仅计算复杂, 而且会出现病态, 不正定的情况. 为克服这些缺点, 同时保持较快收敛的优点, 构造一个正定矩阵G_{k+1} 近似代替$(\nabla^2 f(\boldsymbol{x}_k)$, 或构造H_{k+1} 近似代替$(\nabla^2 f(\boldsymbol{x}_k))^{-1}$

记:

$$\Delta \boldsymbol{x}_k = \boldsymbol{x}_{k+1} - \boldsymbol{x}_k \tag{1.7}$$

$$\Delta f_k = \nabla f(\boldsymbol{x}_{k+1}) - \nabla f(\boldsymbol{x}_k) \tag{1.8}$$

计算G_{k+1} 有两组迭代公式:

BFGS (Broyden-Fletcher-Goldfarb-Shanno) 公式

$$G_{k+1} = G_k + \frac{\Delta f_k \Delta f_k^T}{\Delta f_k^T \Delta \boldsymbol{x}_k} - \frac{G_k \Delta \boldsymbol{x}_k \Delta \boldsymbol{x}_k^T G_k}{\Delta \boldsymbol{x}_k^T G_k \Delta \boldsymbol{x}_k} \tag{1.9}$$

$$H_{k+1} = H_k + (1 + \frac{\Delta f_k^T H_k \Delta f_k}{\Delta f_k^T \Delta \boldsymbol{x}_k}) \frac{\Delta \boldsymbol{x}_k (\Delta \boldsymbol{x}_k)^T}{\Delta f_k^T \Delta \boldsymbol{x}_k} - \frac{\Delta \boldsymbol{x}_k \Delta f_k^T H_k + H_k \Delta f_k \Delta \boldsymbol{x}_k^T}{\Delta f_k^T \Delta \boldsymbol{x}_k} \tag{1.10}$$

DFP(Davidon-Fletcher-Powell) 公式

$$G_{k+1} = G_k + (1 + \frac{\Delta \boldsymbol{x}_k^T G_k \Delta \boldsymbol{x}_k}{\Delta \boldsymbol{x}_k^T \Delta f_k}) \frac{\Delta f_k \Delta f_k^T}{\Delta f_k^T \Delta \boldsymbol{x}_k} - \frac{\Delta f_k \Delta \boldsymbol{x}_k^T G_k + G_k \Delta \boldsymbol{x}_k \Delta f_k^T}{\Delta \boldsymbol{x}_k^T \Delta f_k} \tag{1.11}$$

$$H_{k+1} = H_k + \frac{\Delta \boldsymbol{x}_k \Delta \boldsymbol{x}_k^T}{\Delta f_k^T \Delta \boldsymbol{x}_k} - \frac{H_k \Delta f_k \Delta f_k^T H_k}{\Delta f_k^T H_k \Delta f_k} \tag{1.12}$$

这两组公式是对偶的, 只要互换G_k, H_k 与$\Delta f_k, \Delta \boldsymbol{x}_k$ 互换即得.

可证明, 只要$\Delta f_k^T \Delta \boldsymbol{x}_k > 0$ 取$H_1 = I$(单位阵) , 这样构造的H_k 正定对称, 迭代过程二阶收敛.

1.2 搜索步长的确定

确定搜索方向$\boldsymbol{d}_k$ 确定, 求步长a_k 实际是一个一维优化问题:

$$\min_{\alpha} f(x_k + \alpha) \tag{1.13}$$

显然其精确解应满足

$$\nabla f(\boldsymbol{x}_k + \alpha \boldsymbol{d}_k)^T \boldsymbol{d}_k = 0 \tag{1.14}$$

当f 复杂的时候计算很复杂, 实际计算的时候用插值方法. 如采用二次插值:

$$q(\alpha) = a\alpha^2 + b\alpha + c \tag{1.15}$$

参数a, b, c 可由$\alpha = 0$ 的函数值和导数$q(0) = f(\boldsymbol{x}_k)$, $q'(0) = \nabla f(x_k)^T \boldsymbol{d}_k$, 以及另一点函数值确定, 而α 的最优值取$\alpha_{min} = -\frac{b}{2a}$ 使$q(\alpha)$ 达到最小值.

还可以用三次插值, 或者二次三次混合插值等.

1.3 非线性最小二乘拟合

有一组数据$(t_i, y_i), i = 1, \ldots, n$, 要拟合一个已知函数$y = f(t, \boldsymbol{x})$
$\boldsymbol{x} = (x_1, \ldots, x_m)^T, m \leqslant n$, $\boldsymbol{x}$ 为待定系数.
记$r_i(\boldsymbol{x}) = y_i - f(t_i, \boldsymbol{x}), r(\boldsymbol{x}) = (r_1(\boldsymbol{x}), \ldots, r_n(\boldsymbol{x}))$
最小二乘发即如下优化模型:

$$\min_{\boldsymbol{x}} R(\boldsymbol{x}) = r^T(\boldsymbol{x}) r(\boldsymbol{x}) \tag{1.16}$$

记$r(\boldsymbol{x})$ 的Jacobi 矩阵为:

$$J(\boldsymbol{x}) = (\frac{\partial r_i}{\partial x_j})_{n \times m} \tag{1.17}$$

则

$$\nabla R = 2J(\boldsymbol{x})^T r(\boldsymbol{x}) \tag{1.18}$$

$$\nabla^2 R = 2J(\boldsymbol{x})^T J(\boldsymbol{x}) + 2S \tag{1.19}$$

$$S = \sum_{i=1}^{n} r_i(\boldsymbol{x}) \nabla^2 r_i(\boldsymbol{x}) \tag{1.20}$$

$$\nabla^2 r_i(\boldsymbol{x}) = (\frac{\partial^2 r_i}{\partial x_k \partial x_l})_{n \times n} \tag{1.21}$$

如用牛顿法计算, S的计算量很大, 因此需要简化. 可忽略S 由此得高斯-牛顿(Guass-Newton) 法. 下降方向$\boldsymbol{d}_k$为:

$$J(\boldsymbol{x}_k)^T J(\boldsymbol{x}_k) d_k = -J(\boldsymbol{x}_k) r(\boldsymbol{x}_k) \tag{1.22}$$

高斯-牛顿法中$J^T J$总是半正定, 但会出现病态, 改进方法是修正的的计算方法:

$$(J(\boldsymbol{x}_k)^T J(\boldsymbol{x}_k) + \alpha_k I) \boldsymbol{d}_k = -J(\boldsymbol{x}_k) r(\boldsymbol{x}_k) \tag{1.23}$$

$\alpha_k > 0$ 是在每次迭代中修正的参数. α 很小时$\boldsymbol{d}_k$ 接近高斯-牛顿法, α 很大时, 接近负梯度方向, 迭代方向在牛顿方向和负梯度方向之间. 这个方法的名字叫LM(Levenbery-Marquardt)法.

2 带约束非线性规划 [4]

带约束线性规划可表示成如下形式:

$$\begin{array}{ll} max & z = f(\boldsymbol{x}), \boldsymbol{x} \in \mathbb{R}^n \\ s.t. & h_i(\boldsymbol{x}), i = 1, 2, \ldots, m \\ & g_i(\boldsymbol{x}) \leqslant 0, j = 1, 2, \ldots, l \end{array} \tag{2.1}$$

若只有等式约束$h_i(\boldsymbol{x}) = 0$ 可用Lagrange 乘子法化为无约束优化:

$$L(\boldsymbol{x}, \boldsymbol{\mu}) = f(\boldsymbol{x}) + \sum_{i=1}^{m} \mu_i h_i(\boldsymbol{x}) \tag{2.2}$$

记可行解域为$G: g_j(\boldsymbol{x}) \leqslant 0, j = 1, 2, \ldots, l$, 若最优解不在$G$ 的边界而在其内部取到, 则问题可简化为无约束优化. 下面只考虑最优解在边界取到.

设$\boldsymbol{x}$ 为可行解, 使

$$g_j(\boldsymbol{x}) = 0, \quad j \in J_1 \tag{2.3}$$

$$g_j(\boldsymbol{x}) < 0, \quad j \in J_2 \tag{2.4}$$

$$\tag{2.5}$$

称$g_j(j \in J_1)$ 为起作用约束, $g_j(j \in J_2)$ 为不起作用约束.

对于$\boldsymbol{x} \in G$ 和一方向$\boldsymbol{d}$ 若存在实数λ_0, 使$\boldsymbol{x} + \lambda \boldsymbol{d} \in G(o < \lambda < \lambda_0)$, 则称$\boldsymbol{d}$ 为$\boldsymbol{x}$ 的可行方向.

$\boldsymbol{x}$ 在有效约束上, 即$g_j(\boldsymbol{x}) = 0(j \in J_i)$, 将$g_j(\boldsymbol{x} + \lambda \boldsymbol{d})$ 泰勒展开,

$$g_i(\boldsymbol{x} + \lambda \boldsymbol{d}) = g_i(\boldsymbol{x}) + \lambda \nabla g_i(\boldsymbol{x})^T \boldsymbol{d} + o(\lambda^2) \tag{2.6}$$

只只要

$$\nabla g_i(\boldsymbol{x})^T \boldsymbol{d} < 0 \tag{2.7}$$

即$\boldsymbol{d}$即为可行方向.

对于$\boldsymbol{x} \in G$ 和一方向$\boldsymbol{d}$, 若存在λ_0 , 使:

$$f(\boldsymbol{x} + \lambda \boldsymbol{d}) < f(x), \quad (0 < \lambda < \lambda_0) \tag{2.8}$$

显然只要

$$\nabla f(\boldsymbol{x})^T \boldsymbol{d} < 0 \tag{2.9}$$

$\boldsymbol{d}$ 即为下降方向.

显然最优解有必要条件, 即不存在同时为可行方向和最优方向的方向.

即: 若$\boldsymbol{x}$ 是最优解, 且$\nabla g_j(x)(j \in J_1)$ 线性无关, 则存在$\lambda_1, \ldots, \lambda_l \geqslant 0$, 使

$$\nabla f(\boldsymbol{x}) + \sum_{j=1}^{l} \lambda_j \nabla g_j(\boldsymbol{x}) = 0 \tag{2.10}$$

$$\lambda_j g_j(\boldsymbol{x}) = 0, \quad j = 1, 2, \ldots, l \tag{2.11}$$

这个条件称为Kuhn-Tucker 条件, 满足这个条件的点称为K-T 点. 最优解只可能在K-T 点中取到.

2.1 二次规划

当目标函数是二次函数, 约束为线性时, 模型2.1 简化为

$$\begin{array}{ll} max & f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T H \boldsymbol{x} + \boldsymbol{c}\boldsymbol{x} \\ s.t. & A\boldsymbol{x} \leqslant \boldsymbol{b} \end{array} \tag{2.12}$$

其中$\boldsymbol{x}, A, \boldsymbol{b}$ 同线性规划, $H \in \mathbb{R}^{n \times n}$ 为对称阵, 叫做二次规划(Quadratic Programming, 记作QP). 特别, 当H 正定时, 目标函数是凸函数, 称为凸二次规划.

QP有良好的性质:

- K-T 条件不仅是最优解的必要条件, 而且是充分条件
- 局部最优解就是全局最优解

2.2 逐步二次规划法

非线性规划的解法很多, 有可行方向法, 罚函数发, 梯度投影法等. 下面简述, 逐步二次规划法(Sequential Quadratic Programming)

基本原理是:构造Lagrange 乘子

$$L(\boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{lambda}) = f(\boldsymbol{x}) + \sum_{i=1}^{m} \mu_i h_i(\boldsymbol{x}) + \sum_{j=1}^{l} \lambda_j g_j(\boldsymbol{x}) \tag{2.13}$$

用二次函数近似$L(\boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{\lambda}$ 后化为QP 问题:

$$\begin{array}{ll} max & \frac{1}{2}\boldsymbol{d}^T G \boldsymbol{d} + \nabla f(\boldsymbol{x}_k)^T \boldsymbol{d} \\ s.t. & \nabla h_i(\boldsymbol{x}_k)^T \boldsymbol{d} + h_i(\boldsymbol{x}) = 0, \quad i = 1, \ldots, m \\ & \nabla g_i(\boldsymbol{x}_k)^T \boldsymbol{d} + g_i(\boldsymbol{x}_k) \leqslant 0, \quad j = 1, \ldots, l \end{array} \tag{2.14}$$

其中$\boldsymbol{x}_k$ 是第k 次迭代的初始点, G_k 是$L(\boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{lambda})$ 的Hesse 矩阵的近似, 可用BFGS 公式.

得到的最优解$\boldsymbol{d}_k$ 取作第k 次迭代的搜索方向. 步长由线性搜索确定.

3 求解非线性规划的软件 [1]

求解非线性规划方法众多, 不像线性规划中基本上单纯形法一统天下, 因此有各种各样的软件和工具包的存在, 这里很不完全的列一些软件, 并介绍下面用到的AMPL 建模系统.

3.1 可用的软件库

Omuses/HQP 是一个开源的c++库, 提供各种解法器. OPT++2.1 也是一个c++库, 提供各种解法器.

我测试过以上两个库, 具体的程序比较复杂, 不再详述.

另外COIN-OR [3] (COmputational INfrastructure for Operations Research - open source for the operations research community -) 是一个不错的软件收集工程.

3.2 AMPL [2]

AMPL ®[2] (A Modeling Language for Mathematical Programming)

专门为优化问题设计的建模语言. 它是由Bell实验室开发的. GLPK 使用的GNU MathProg Language 就是他的一个子集. AMPL 不仅可以处理线性规划问题, 还能处理非线性规划问题. 他将规划文件.mod 转换成专门的.nl 文件. 然后调用解法器(solver) 求解. AMPL本身不是一个求解程序, 而是一种翻译器.

AMPL软件是付费的, 不过可以使用免费的学生版.

`http://www.ampl.com/NEW/TABLES/amplcml.zip`

他还包括了学生版的MINOS, CPLEX 等解法器, 学生版的限制主要是问题的规模限制在300个变量以下. 下面的计算都是在学生版中完成的.

3.2.1 AMPL/Solver

这里主要讨论开源的解法器, Ipopt, 和TRON. Ipopt 和TRON都可以用c之间调用, 具体方法就不给出了.

3.2.2 Neos

http://neos.mcs.anl.gov/

NEOS服务. 是在线解决优化问题的服务器, 可以通过email, XML-RPC, 和web提交问题. 目前NEOS的版本是5. NEOS支持各种各样的的优化解法器, 而且可以自动确定很多求解参数. NEOS支持AMPL格式的建模程序. 如果你有优化问题, 规模比较大, 不妨试一下, 很好用.

4 习题解答

4.1 解下面问题

$$\begin{array}{ll} max & f(\boldsymbol{x}) = 2x_1 - x_2 \\ s.t. & x_1^2 + x_2^2 \leqslant 1 \\ & x_2 \geqslant 0 \end{array} \tag{4.1}$$

使用AMPL语言建模, 求解

```
var x{1..2};
maximize ff: 2 * x[1] - x[2];
subject to ss: x[1]^2 + x[2]^2 <= 1;
subject to bb: x[2] >= 0;
```

运行命令

```
G:\mydoc\my text\建模\modeling\05sum\nonlinear>ampl
ampl: model a2.mod
ampl: solve;
MINOS 5.5: optimal solution found.
9 iterations, objective 2
Nonlin evals: constrs = 32, Jac = 31.
ampl: display x;
x [*] :=
1  1
2  0
;
```

解得最优解在$x_1 = 1, x_2 = 0$ 取得为2.

4.2 第二题解

4.2.1 题目

根据一个地区国民经济发展需要和电力建设规划, 已知到某一年要求水电总装机容量至少为NkW, 现拟同时兴建n座水电站, 而该地区可能筹集到的资金为B, 若采用水电站群的"总年利润最大" 为准则, 试确定各水电站的最优装机容量. 试建立其数学模型.

4.2.2 解

假设电价和可供应电量成负指数关系, 假设发电机组的利润和发电机组容量成负指数关系. 即容量越大, 单位利润越大

具体建模如AMPL程序, 并凑了一组数据试验

```
set P;
param op{P}; #各电站建设支出每千瓦
param bop{P}; #各电站基准建设费用
param bip; #基准价格
param N; #至少装机容量
param B; #可获得的资金
param bue{P}; #最大装机容量
param bbe{P}; #最小装机容量
param bmoq; # 基准维护费用
param mlam; # 单位利润的上升参数
param lam; # 电价的下降参数

var e{P} >=0 ; #装机容量
var ip; # 收入每千瓦
var income; # 年收入
var outcome; # 年维护支出
var bout; # 建设支出
var ea ; #总装机容量
var lucre ; # 收益
var moq{P}; # 每个电站的维护费用

maximize w: lucre;

# 利润等于年收入减去总维护费用
subject to lucreq: lucre - (income - outcome) = 0 ;
# 年支出等于各电站维护费用.
subject to youteq: outcome - sum{i in P} moq[i] * e[i] = 0 ;
# 年收入等于电价乘以装机容量
subject to inm: income - ip * ea = 0;
# 建设支出等于各电站基础建设费用加上装机费用
subject to bouteq: sum{i in P} (op[i] * e[i] + bop[i]) - bout = 0 ;
# 建设支出小于可筹集资金
```

```
subject to allb: bout <= B;
# 总供电量大于N
subject to equic: ea >= N;
# 各电站有可建设容量上下限
subject to beq{i in P}: bbe[i] <= e[i] <= bue[i] ;
# 总供电量为各电站供电量之和
subject to eaq: sum{i in P} e[i] - ea = 0;
# 单位电价和总供电量成负指数关系
subject to eip: bip * lam ^ (ea/N) - ip = 0;
# 单位维护费用随容量上升而下降, 设为负指数关系
subject to moqeq{i in P}:  bmoq * mlam ^ e[i] - moq[i] = 0;

data;
set P:=1,2,3,4,5;
param op:=
1 1.1
2 1.2
3 1.1
4 1.3
5 1.2;
param bop:=
1 2.2
2 2.0
3 2.5
4 2.3
5 2.2;
param bbe:=
1 100
2 110
3 120
4 150
5 170
;
param bue:=
1 200
2 250
3 270
4 290
5 300
;

param bip:=2.0;
param N:=400;
param B:=2000;
param lam:=0.7;
param mlam:= 0.6;
```

```
param bmoq:= 1.1;
end;
```

求解结果

```
ampl: model tp1.mod
ampl: solve;
MINOS 5.5: optimal solution found.
23 iterations, objective 825.1309982
Nonlin evals: constrs = 62, Jac = 61.
ampl: display e;
e [*] :=
1  200
2  250
3  270
4  231.409
5  170
;
```

References

[1] Nonlinear Programming Frequently Asked Questions , http://www-unix.mcs.anl.gov/otc/Guide/faq/nonlinear-programming-faq.html , 2000

[2] AMPL 主页, http://www.ampl.com , 2001

[3] COmputational INfrastructure for Operations Research , http://www.coin-or.org/

[4] 高等教育出版社, 数学实验, 萧铁树等, 1999