可行方向法

李宗军

青岛理工大学 · 理学院

2016 年

要点

1. 可行方向法
   - 简约梯度法

复习 I

1. 等式约束的 K-T 条件
2. 不等式约束的 K-T 条件
3. 一般的 NLP 的 K-T 条件

$$\nabla f(x) + \sum_{i \in I} \lambda_i \nabla g_i(x) + \sum_{j \in J} u_j \nabla h_j(x) = 0$$

$$\lambda_i g_i(x) = 0, i \in I$$

$$\lambda_i \geq 0, i \in I$$

4. K-T 条件的计算
5. K-T 条件是必要条件，注意其梯度们是线性无关的
6. 对凸规划，K-T 条件是充要条件

复习 II

7. LP 问题的 K-T 条件
8. K-T 的作用是把最优化问题转换为方程组问题

简约梯度法

1. 考虑的问题

$$\min f(x)$$
$$st. \begin{cases} Ax & = b \\ x & \geq 0 \end{cases}$$

2. 算法思想

仿照 LP 中的单纯形法，在每一轮迭代的当前点 x^k 处，把 x^k 分为基变量部分和非基变量部分，则 $f(x) = F(x_N)$, 采用 $F(x_N)$ 的负梯度方向构造可行下降方向 p，迭代步长采用一维搜索方法。

算法推导

- $Ax = b \Rightarrow Bx_B + Nx_N = b, x_B = B^{-1}b - B^{-1}Nx_N$

算法推导

- $Ax = b \Rightarrow Bx_B + Nx_N = b, x_B = B^{-1}b - B^{-1}Nx_N$
- 代入目标函数，有 $f(x) = f(B^{-1}b - B^{-1}Nx_N, x_N)$, 令为 $F(x_N)$.

算法推导

- $Ax = b \ \Rightarrow Bx_B + Nx_N = b, x_B = B^{-1}b - B^{-1}Nx_N$
- 代入目标函数，有 $f(x) = f(B^{-1}b - B^{-1}Nx_N, x_N)$, 令为 $F(x_N)$.
- 取 $\nabla F(x_N) = -\left(B^{-1}N\right)^T \nabla_B f(x) + \nabla_N f(x) \overset{\text{LET}}{=} r_N$ 作为下降方向。下面考察可行性。

算法推导

- $Ax = b \;\Rightarrow Bx_B + Nx_N = b, x_B = B^{-1}b - B^{-1}Nx_N$
- 代入目标函数，有 $f(x) = f(B^{-1}b - B^{-1}Nx_N, x_N)$, 令为 $F(x_N)$.
- 取 $\nabla F(x_N) = -\left(B^{-1}N\right)^T \nabla_B f(x) + \nabla_N f(x) \overset{\text{LET}}{=} r_N$ 作为下降方向。下面考察可行性。
- 取 $p = (p_B, p_N)^T = (p_B, r_N)^T$, 则下个迭代点为 $x' = x + tp = (x_B, x_N)^T + t\,(p_B, p_N)^T$.

算法推导

- $Ax=b \ \Rightarrow Bx_B+Nx_N=b, x_B=B^{-1}b-B^{-1}Nx_N$
- 代入目标函数，有 $f(x)=f(B^{-1}b-B^{-1}Nx_N, x_N)$, 令为 $F(x_N)$.
- 取 $\nabla F(x_N)=-\left(B^{-1}N\right)^T\nabla_B f(x)+\nabla_N f(x)\overset{\text{LET}}{=}r_N$ 作为下降方向。下面考察可行性。
- 取 $p=(p_B,p_N)^T=(p_B,r_N)^T$, 则下个迭代点为 $x'=x+tp=(x_B,x_N)^T+t\,(p_B,p_N)^T$.
- 要使 x' 满足可行性，即 $Ax'=b, x'\geq 0$. 从而

算法推导

- $Ax = b \Rightarrow Bx_B + Nx_N = b, x_B = B^{-1}b - B^{-1}Nx_N$
- 代入目标函数，有 $f(x) = f(B^{-1}b - B^{-1}Nx_N, x_N)$, 令为 $F(x_N)$.
- 取 $\nabla F(x_N) = -\left(B^{-1}N\right)^T \nabla_B f(x) + \nabla_N f(x) \stackrel{\text{LET}}{=} r_N$ 作为下降方向。下面考察可行性。
- 取 $p = (p_B, p_N)^T = (p_B, r_N)^T$, 则下个迭代点为 $x' = x + tp = (x_B, x_N)^T + t\,(p_B, p_N)^T$.
- 要使 x' 满足可行性，即 $Ax' = b, x' \geq 0$. 从而
- $Ax' = A(x + tp) = Ax + tAp = b + tAp = b$, 从而 $Ap = 0$, 即

算法推导

- $Ax = b \ \Rightarrow Bx_B + Nx_N = b, x_B = B^{-1}b - B^{-1}Nx_N$
- 代入目标函数，有 $f(x) = f(B^{-1}b - B^{-1}Nx_N, x_N)$, 令为 $F(x_N)$.
- 取 $\nabla F(x_N) = -\left(B^{-1}N\right)^T \nabla_B f(x) + \nabla_N f(x) \overset{\text{LET}}{=} r_N$ 作为下降方向。下面考察可行性。
- 取 $p = (p_B, p_N)^T = (p_B, r_N)^T$, 则下个迭代点为 $x' = x + tp = (x_B, x_N)^T + t\,(p_B, p_N)^T$.
- 要使 x' 满足可行性，即 $Ax' = b, x' \geq 0$. 从而
- $Ax' = A(x + tp) = Ax + tAp = b + tAp = b$, 从而 $Ap = 0$, 即
- $(B, N)\,(p_B, p_N)^T = Bp_B + Np_N = 0$, 解得 $p_B = -B^{-1}Np_N$

算法推导 II

- $x' = x + tp \geq 0 \Rightarrow$

$$\begin{cases} \forall t > 0, & p \geq 0 \\ t \leq \left\{ -\frac{x_i^k}{p_i^k} | p_i^k < 0, i = 1, \cdots, n \right\} & \text{else} \end{cases},$$

算法推导 II

- $x' = x + tp \geq 0 \ \Rightarrow$
$$\begin{cases} \forall t > 0, & p \geq 0 \\ t \leq \left\{ -\frac{x_i^k}{p_i^k} | p_i^k < 0, i = 1, \cdots, n \right\} & \text{else} \end{cases},$$
- 故取 $t_{\max} = \begin{cases} +\infty, & p^k \geq 0 \\ \min\left\{ -\frac{x_i^k}{p_i^k} | p_i^k < 0, i = 1, \cdots, n \right\}, & \text{else} \end{cases}$

算法推导 II

- $x' = x + tp \geq 0 \ \Rightarrow$
$$\begin{cases} \forall t > 0, & p \geq 0 \\ t \leq \left\{ -\frac{x_i^k}{p_i^k} | p_i^k < 0, i = 1, \cdots, n \right\} & \text{else} \end{cases},$$
- 故取 $t_{\max} = \begin{cases} +\infty, & p^k \geq 0 \\ \min\left\{ -\frac{x_i^k}{p_i^k} | p_i^k < 0, i = 1, \cdots, n \right\}, & \text{else} \end{cases}$
- 对 $p_N = r_N, x'_N = x_N + tr_N$, 其分量形式 $x'_i = x_i + tr_i$, 为保证 $x'_i \geq 0$，选取 $p_i = \begin{cases} -r_i, & r_i \leq 0 \\ -x_i r_i, & r_i > 0 \end{cases}$

定理 4.5.3

1. 若 $p \neq 0$, 则 p 是 $f(x)$ 在 x^k 处的可行下降方向;
2. $p = 0$ 的充要条件是 x^k 是 K-T 点。

迭代步骤 I

1. 算法初始化，给定初始点 $x^0, \varepsilon > 0, k := 0$
2. 设 I_B^k 是 x^k 的 m 个最大分量的下标集，对矩阵 A 进行分解 $A = (B_k, N_k)$
3. 计算 $\nabla f(x^k) = \left(\nabla_B f(x^k), \nabla_N f(x^k)\right)^T$，然后计算简约梯度 $r_N^k = \left(B_k^{-1} N_k\right)^T \nabla_B f(x^k) - \nabla_N f(x^k)$. 记 r_N^k 的第 i 个分量为 r_i^k
4. 构造 $p^k = \left(p_B^k, p_N^k\right)^T$, 其中
$$p_N^k = \left(p_i^k\right) = \begin{cases} r_i^k, & r_i^k \le 0 \\ x_i^k r_i^k, & r_i^k > 0 \end{cases}, p_B^k = -B_k^{-1} N_k p_N^k.$$ 若 $\left\|p^k\right\| \le \varepsilon$, 停止迭代，输出 x^k.
5. 采用一维搜索，求解 $\min\left\{f\left(x^k + tp^k\right) \middle| 0 \le t \le t_{\max}^k\right\}$, 得到最优解 t_k, 其中
$$t_{\max}^k = \begin{cases} +\infty, & p^k \ge 0 \\ \min\left\{-\frac{x_i^k}{p_i^k} \middle| p_i^k < 0\right\}, & \text{else} \end{cases}$$

迭代步骤 II

6. $x^{k+1} = x^k + t_k p^k, k := k + 1$, 转向 2.

程序 MMA

MMA 演示

算法说明

1. 数值实验表明：对于大规模的线性约束的 NLP，简约梯度法是最好的
2. 对约束最优化而言，迭代方向必须是可行下降方向，此类算法统称为可行下降算法。
3. 由定理 4.5.3（$p^k = 0$ 的充要条件是 x^k 是 K-T 点。），从而迭代终止条件采用简单的 $\|p^k\| \leq \varepsilon$, 而不是采用复杂的 K-T 条件。
4. 初始点，可采用 LP 中的两阶段法获得。

小结

1. 简约梯度法，适用的问题是 $\min\{f(x)|Ax = b, x \geq 0\}$
2. 算法思想
3. 算法推导：迭代方向、迭代步长、终止条件、初始点
4. 迭代步骤
5. MMA 程序

作业

T25