线性规划内点法

文再文

北京大学北京国际数学研究中心

教材《最优化：建模、算法与理论》配套电子教案

http://bicmr.pku.edu.cn/~wenzw/optbook.html

致谢：本教案由谢中林、张轩熙协助准备

提纲

单纯形法与内点法

单纯形法:

- 不断列出可行域的顶点然后一步一步寻找问题的最优解
- 线性规划可行域的顶点数可能多达$\mathcal{O}(2^n)$个(n为自变量维数), 因此单纯形法最坏情况下的复杂度是指数量级

内点法:

- 在可行域内部寻找一条路径最终抵达其边界
- 每个迭代步计算代价都远高于仅仅在可行域边界移动的单纯形法
- 内点法的一步迭代对问题解的改善是显著的, 可以证明内点法实际上是一个多项式时间算法
- 原始-对偶算法是其中效率最高的实用方法, 在大规模问题上是单纯形法强有力的竞争者

原始-对偶算法: 思想

原始及对偶问题

$$\begin{array}{lll} \min & c^T x & \\ \text{s.t.} & Ax = b & \\ & x \geq 0 & \end{array} \qquad \begin{array}{lll} \max & b^T y & \\ \text{s.t.} & A^T y + s = c & \\ & s \geq 0 & \end{array}$$

KKT条件

$$\begin{aligned} Ax &= b, \\ A^{\mathrm{T}} y + s &= c, \\ x_i s_i &= 0,\ i = 1, 2, \cdots, n, \\ x \geq 0, s &\geq 0. \end{aligned}$$

- 利用KKT条件不断在可行域的相对内部产生迭代点
- 算法构造的解满足第一、二、四式, 而只能近似地满足条件三
- 当条件四满足且条件三对任意的i 不满足时, 我们有$x_i s_i > 0,\ \forall\, i$, 这意味着点$(x, s)$ 为可行域的相对内点

对偶间隙与终止条件

- 互补条件$x_i s_i = 0$在内点法中不能严格满足, 而我们想要算法最终收敛到线性规划问题的解, 因此希望$x_i s_i \to 0,\ \forall\, i$. 这个条件就可以作为内点法的终止条件
- 对内点$x > 0, s > 0$定义互补条件违反度的度量

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i s_i = \frac{x^{\mathrm{T}} s}{n},$$

也称为**对偶间隙**. 当μ趋于0时, (x, s)将接近可行域的边界

原始-对偶算法: 通过舍去非线性项推导

- 线性规划原始-对偶算法的目标是给定当前可行点(x, y, s), 寻找下一个点

$$(\tilde{x}, \tilde{y}, \tilde{s}) = (x, y, s) + (\Delta x, \Delta y, \Delta z)$$

使得如下条件成立:

$$\begin{cases} A^{\mathrm{T}}\tilde{y} + \tilde{s} = c, & \tilde{s} > 0, \\ A\tilde{x} = b, & \tilde{x} > 0, \\ \tilde{x}_i\tilde{s}_i = \sigma\mu, & i = 1, 2, \cdots, n. \end{cases}$$

其中$0 < \sigma < 1$ 是取定的常数.

- 上述条件也被称为是**扰动KKT 条件**, 最后一个条件可进一步使用分量乘积简化为$\tilde{x} \odot \tilde{s} = \sigma\mu\mathbf{1}$.

- 最后一个条件: 假设μ 是当前点(x, y, s) 处的对偶间隙, 我们希望迭代下一步时这个度量将会缩小一个比例σ .

原始-对偶算法: 通过舍去非线性项推导

展开方程组可以得到

$$\begin{cases} A(x+\Delta x) = b, \\ A^{\mathrm{T}}(y+\Delta y) + (s+\Delta s) = c, \\ (s+\Delta s) \odot (x+\Delta x) = \sigma\mu\mathbf{1}, \end{cases}$$

去除高阶非线性项$\Delta x \odot \Delta s$ 后得到线性方程组 :

$$\begin{cases} A\Delta x = r_p \overset{\text{def}}{=\!=} b - Ax, \\ A^{\mathrm{T}}\Delta y + \Delta s = r_d \overset{\text{def}}{=\!=} c - s - A^{\mathrm{T}}y, \\ x \odot \Delta s + s \odot \Delta x = r_c \overset{\text{def}}{=\!=} \sigma\mu\mathbf{1} - x \odot s, \end{cases}$$

其中$r = (r_p, r_d, r_c)^{\mathrm{T}}$ 刻画了KKT 条件的残量 .

原始-对偶算法: 通过舍去非线性项推导

记$L_x = \mathrm{Diag}(x)$, $L_s = \mathrm{Diag}(s)$, 我们将方程组化为矩阵形式

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^{\mathrm{T}} & I \\ L_s & 0 & L_x \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} r_p \\ r_d \\ r_c \end{bmatrix}.$$

利用矩阵分块消元, 可以直接求解上述方程, 得到

$$\begin{cases} \Delta y = (AL_s^{-1}L_xA^{\mathrm{T}})^{-1}(r_p + AL_s^{-1}(L_xr_d - r_c)), \\ \Delta s = r_d - A^{\mathrm{T}}\Delta y, \\ \Delta x = -L_s^{-1}(L_x\Delta s - r_c), \end{cases}$$

其中$AL_s^{-1}L_xA^{\mathrm{T}}$ 是对称矩阵, 当A 满秩时, $AL_s^{-1}L_xA^{\mathrm{T}}$ 正定.

原始-对偶算法: 利用牛顿法推导

- 记当前解为(x, y, s), $(x, s) > 0$.记$(\Delta x, \Delta y, \Delta s)$ 为搜索方向.
- 用牛顿法直接求解KKT条件, 即

$$F(x, y, s) = \begin{bmatrix} Ax - b \\ A^T y + s - c \\ x \odot s \end{bmatrix} = 0, \quad (x, s) \geq 0.$$

搜索方向满足

$$J(x, y, s) \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ L_s & 0 & L_x \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = -F(x, y, s) \triangleq \begin{bmatrix} r_p \\ r_d \\ -x \odot s \end{bmatrix}$$

其中J 表示F 的雅各比矩阵.

原始-对偶算法: 利用牛顿法推导

- 做全步长迭代$(x, y, s) + (\Delta x, \Delta y, \Delta s)$ 可能会违反约束$(x, s) \geq 0$.
- 进行线搜索, 新的迭代点为$(x, y, s) + \alpha(\Delta x, \Delta y, \Delta s)$, $\alpha \in (0, 1]$
- 我们经常会得到一个小步长更新（$\alpha << 1$）.纯牛顿方向不能很好的指向一个解, 算法不是很有效.
- 修正的原始-对偶方法不直接指向KKT条件的解, 而是指向一个对偶间隙更小的点, 利用牛顿法求解

$$F(x, y, s) = \begin{bmatrix} Ax - b \\ A^T y + s - c \\ x \odot s \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \sigma\mu\mathbf{1} \end{bmatrix}.$$

其中$0 < \sigma < 1$ 称为中心参数, 希望迭代下一步时对偶间隙将会缩小一个比例σ.

- 修正后的原始-对偶算法所对应的方程与直接舍去非线性项得到的方程相同, 也需利用线搜索得出步长α.

原始-对偶算法: 评注

- 即使初始点(x, y, s) 是可行的, 求解线性方程组产生的更新$(\tilde{x}, \tilde{y}, \tilde{s})$ 也不一定是可行解.
- 因为前两个方程$Ax = b$, $A^{\mathrm{T}}y + s = c$ 是线性的, 在迭代过程中可以一直满足. 但$x > 0, s > 0$ 这个约束**不能**保证一直成立
- 采用线搜索中的回溯法来确定一个合适的更新

$$(x^{k+1}, y^{k+1}, s^{k+1}) = (x^k, y^k, s^k) + \alpha_k(\Delta x^k, \Delta y^k, \Delta s^k),$$

其中$\alpha_k = \alpha_0\rho^{k_0}$, 并选取最小的整数$k_0$ 使得$x^{k+1} > 0, s^{k+1} > 0$, 这里$0 < \rho < 1$, α_0 是给定常数

原始-对偶算法: 总结

1. 选定初始可行点(x^0, y^0, s^0), 令$k = 0$.
2. 根据原始-对偶算法求解搜索方向$(\Delta x, \Delta y, \Delta s)$
3. 选取合适的步长α_k, 做一步更新

$$(x^{k+1}, y^{k+1}, s^{k+1}) = (x^k, y^k, s^k) + \alpha_k(\Delta x, \Delta y, \Delta s)$$

4. 若满足停机条件, 终止; 否则令$k = k + 1$, 转步原始-对偶算法.

算法的计算量主要来自搜索方向的计算, 步长α 和参数σ 的选取对内点法的表现十分关键.

提纲

对数障碍函数与中心的刻画

- 直接用原始-对偶法进行迭代时，$x_i s_i = 0$ 意味着点(x, s) 已经接近可行域的边缘，如果继续进行迭代，则迭代点将会紧贴定义域边缘进行更新，这有违于内点法的思想，容易出现小步长更新
- 修正后的原始对偶方法希望迭代点更靠近“中心”，$x_i s_i$ 能够以一致的速度下降到0
- “中心”这个概念可以通过对数障碍函数来刻画.在原始线性规划问题中，加上点到边界距离的惩罚项

$$\begin{array}{ll} \min & c^T x - \tau \sum_{i=1}^{n} \ln x_i, \\ \text{s.t.} & Ax = b. \end{array}$$

其中$\tau > 0$ 是中心化参数.

- 引入拉格朗日乘子y，KKT条件为

$$\begin{cases} c_i - \dfrac{\tau}{x_i} - A_{\cdot i}^T y = 0, \ i = 1, 2, \ldots, n, \\ Ax = b. \end{cases}$$

中心路径

Definition (中心路径)

令$s_i = \frac{\tau}{x_i}$, 于是最优解(x_τ, y_τ, s_τ) 满足方程：

$$\begin{aligned} Ax &= b, \quad x > 0 \\ A^\top y + s &= c, \quad s > 0 \\ x_i s_i &= \tau, \quad \text{for } i = 1, \ldots, n \end{aligned}$$

定义单参数曲线$\mathcal{C} = \{(x_\tau, y_\tau, s_\tau) \mid \tau > 0\}$ 为中心路径

- 原始对偶可行集和严格可行集定义为

$$\mathcal{F} = \{(x, y, s) | \, Ax = b, A^T y + s = c, (x, s) \geq 0\}$$

$$\mathcal{F}^o = \{(x, y, s) | \, Ax = b, A^T y + s = c, (x, s) > 0\}$$

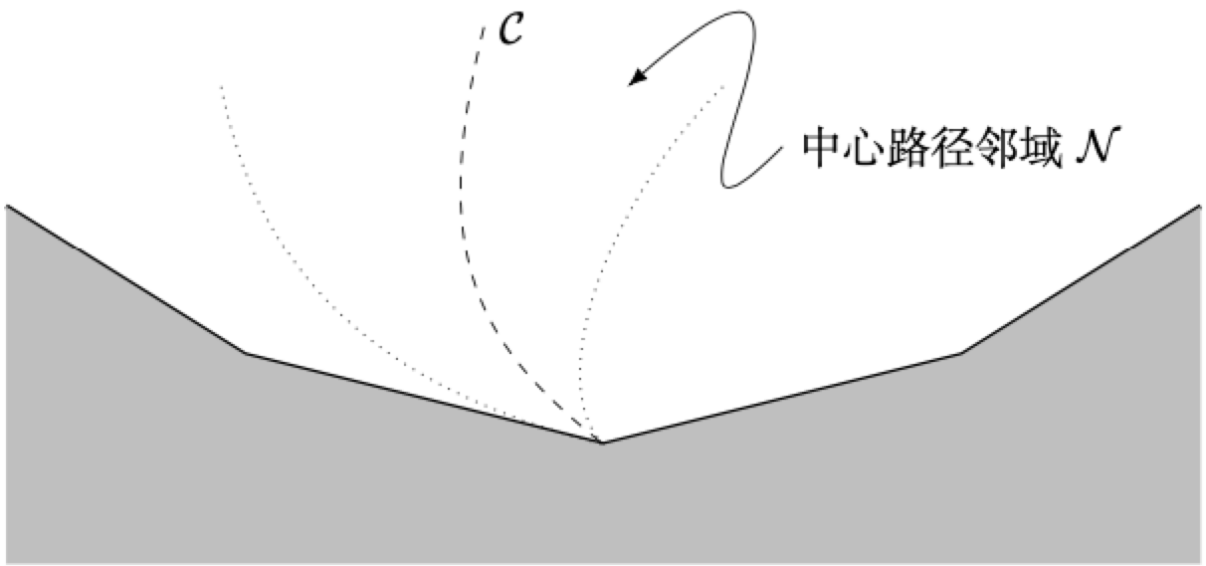
- 中心路径$\mathcal{C}$ 是$\mathcal{F}^o$ 中的一条参数化曲线

中心路径邻域

- 我们希望在原始–对偶算法中，迭代点列在中心路径附近移动，跟随曲线$\mathcal{C}$ 到达最优值点，这样可以保证每一步取较大步长

Definition (中心路径邻域)

$$\mathcal{N}_{-\infty}(\gamma) = \{(x, y, s) \in \mathcal{F}^o \mid x_i s_i \geq \gamma\mu\}, \quad \gamma \in [0, 1).$$



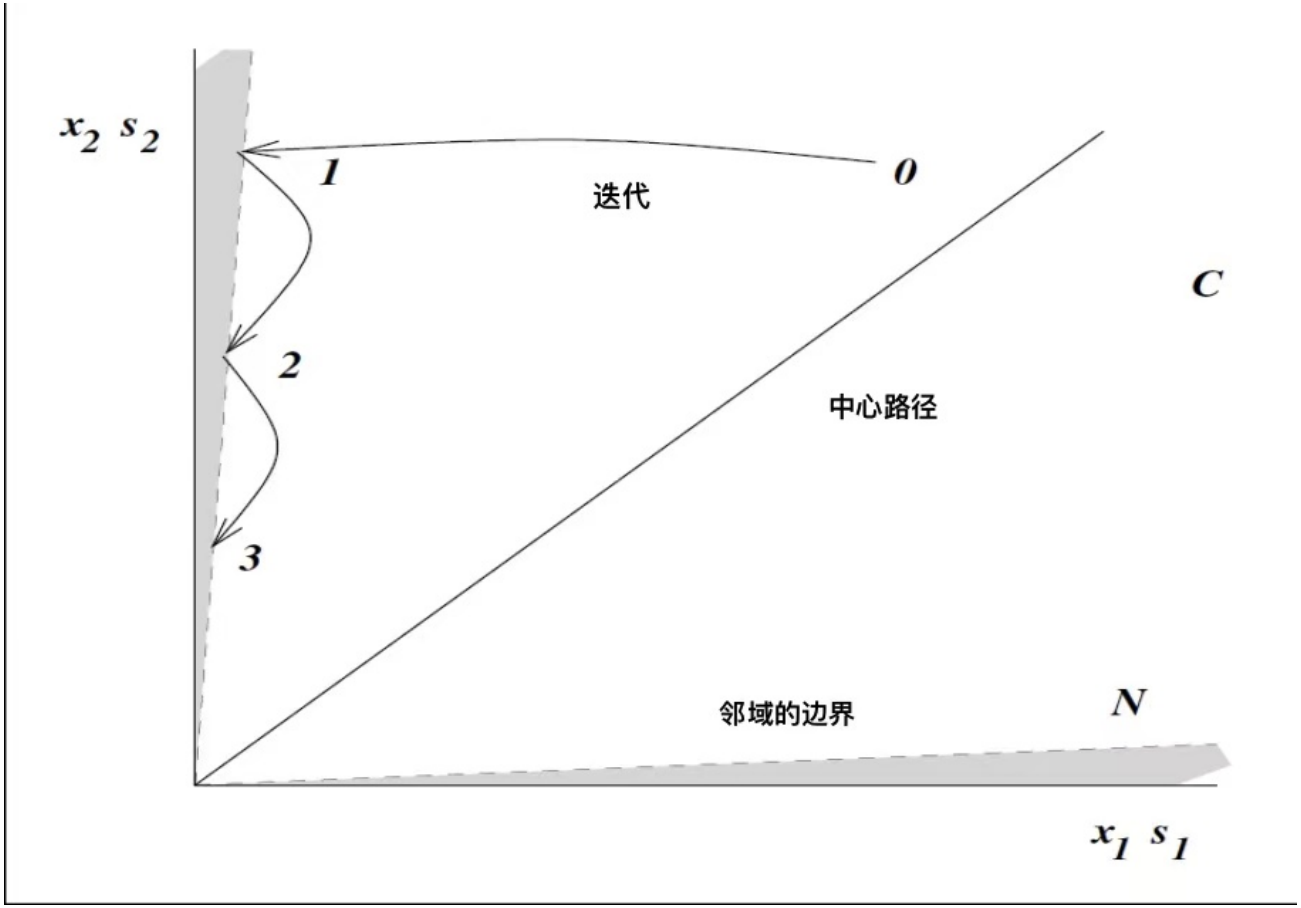
如图为中心路径和中心路径邻域在原始解x 上的投影

带路径追踪的原始– 对偶算法

1. 选择初值点$(x^0, y^0, z^0) \in \mathcal{F}^o$, 选取参数$0 < \gamma < \sigma < 1$, 令$k = 0$.
2. 根据原始-对偶算法求解更新方向$(\Delta x, \Delta y, \Delta s)$
3. 选取最大的步长α_k, 使下一步迭代点落入$\mathcal{N}_{-\infty}(\gamma)$ 内, 做一步更新$(x^{k+1}, y^{k+1}, s^{k+1}) = (x^k, y^k, s^k) + \alpha_k(\Delta x, \Delta y, \Delta s)$
4. 若满足停机条件, 终止; 否则令$k = k + 1$, 转步（2）

- 有了中心路径邻域的概念, 就可以写出带路径追踪的原始– 对偶算法
- 该算法的关键在于如何选取最大的α_k . 实际上, 只需要保证在下一步迭代时(x, y, s) 满足$x_i s_i \geq \gamma\mu$ 即可. 这是关于α 的n 个二次不等式, 求解比较容易 .
- 再结合条件$x > 0, s > 0$ 就能很容易地确定α_k

带路径追踪的原始– 对偶算法



如图为$n = 2$ 的示例, 迭代路径在$x_1s_1 - x_2s_2$ 上的投影, 此时中心路径是一个过原点的倾斜45度的射线, 搜索方向从直线变成了曲线.

提纲

收敛性分析

引理

若$\mathbb{R}^n$ 中两向量u, v 满足$u^{\mathrm{T}}v \geq 0$, 则

$$\|u \odot v\|_2 \leq 2^{-3/2}\|u + v\|_2^2.$$

- 由$u^{\mathrm{T}}v \geq 0$ 知

$$0 \leq u^T v = \sum_{u_i v_i \geq 0} u_i v_i + \sum_{u_i v_i < 0} u_i v_i = \sum_{i \in \mathcal{P}} |u_i v_i| - \sum_{i \in \mathcal{M}} |u_i v_i|,$$

其中$\mathcal{P} = \{i \mid u_i v_i \geq 0\}, \mathcal{M} = \{i \mid u_i v_i < 0\}.$

- 由$\|\cdot\|_2 \leq \|\cdot\|_1$ 得

$$\begin{aligned}
\|u \odot v\|_2 &= (\|[u_i v_i]_{i \in \mathcal{P}}\|_2^2 + \|[u_i v_i]_{i \in \mathcal{M}}\|_2^2)^{1/2} \\
&\leq (\|[u_i v_i]_{i \in \mathcal{P}}\|_1^2 + \|[u_i v_i]_{i \in \mathcal{M}}\|_1^2)^{1/2} \\
&\leq (2\|[u_i v_i]_{i \in \mathcal{P}}\|_1^2)^{1/2} \\
&\leq \sqrt{2}\|[\frac{1}{4}(u_i + v_i)^2]_{i \in \mathcal{P}}\|_1 \leq 2^{-3/2}\|u + v\|_2^2.
\end{aligned}$$

若(x, y, s) 可行, 则$\Delta x^{\mathrm{T}} \Delta s = 0$

引理

若(x, y, s) 可行, 则由原始-对偶算法得到的点$(\Delta x, \Delta y, \Delta s)$ 满足$\Delta x^{\mathrm{T}} \Delta s = 0$.

- 首先, 点$(\Delta x, \Delta y, \Delta s)$ 满足方程:

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^{\mathrm{T}} & I \\ L_s & 0 & L_x \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} r_p \\ r_d \\ r_c \end{bmatrix}.$$

- 由于(x, y, s) 可行, 故

$$r_p = b - Ax = 0,$$
$$r_c = c - s - A^{\mathrm{T}} y = 0.$$

- 于是$A^{\mathrm{T}} \Delta y + \Delta s = 0$, 左右同乘$\Delta x^{\mathrm{T}}$ 并利用$A \Delta x = 0$ 即得

$$\Delta x^{\mathrm{T}} \Delta s = 0.$$

对$\Delta x \odot \Delta s$的估计

由上述两个引理, 可以给出对增量的逐元素积$\Delta x \odot \Delta s$的估计, 进而对步长进行估计.

引理

若$(x, y, s) \in \mathcal{N}_{-\infty}(\gamma)$, 则

$$\|\Delta x \odot \Delta s\| \leq 2^{-3/2}(1 + 1/\gamma)n\mu.$$

- 记$L_x = \mathrm{Diag}(x), L_s = \mathrm{Diag}(s)$, 由$\mathcal{N}_{-\infty}(\gamma) \subset \mathcal{F}_0$知$(x, y, s)$可行, 故$\Delta x^{\mathrm{T}} \Delta s = 0$.
- 对$L_s \Delta x + L_x \Delta s = r_c$两边同乘$(L_s L_x)^{-1/2}$并记$D = L_x^{1/2} L_s^{-1/2}$得

$$D^{-1} \Delta x + D \Delta s = (L_s L_x)^{-1/2} r_c.$$

对$\Delta x \odot \Delta s$的估计

- 由于$(D^{-1}\Delta x)^{\mathrm{T}}(D\Delta s) = 0$, 利用上个引理得

$$\begin{aligned}\|\Delta x \odot \Delta s\| &= \|(D^{-1}\Delta x) \odot (D\Delta s)\| \\ &\le 2^{-3/2}\|D^{-1}\Delta x + D\Delta s\|^2 \\ &= 2^{-3/2}\|(L_x L_s)^{-1/2} r_c\|^2 \\ &= 2^{-3/2}\|(L_x L_s)^{-1/2}(\sigma\mu\mathbf{1} - x \odot s)\|^2.\end{aligned}$$

- 将最后一个估计展开, 由定义$x^{\mathrm{T}}s = n\mu$及$x_i s_i \ge \gamma\mu$得

$$\begin{aligned}\|\Delta x \odot \Delta s\| &= 2^{-3/2}(\sigma\mu\mathbf{1} - x \odot s)^{\mathrm{T}}(L_x L_s)^{-1}(\sigma\mu\mathbf{1} - x \odot s) \\ &\le 2^{-3/2}[x^{\mathrm{T}}s - 2\sigma\mu\mathbf{1}^{\mathrm{T}}\mathbf{1} + \sigma^2\mu^2\sum_{i=1}^{n}\frac{1}{x_i s_i}] \\ &\le 2^{-3/2}[x^{\mathrm{T}}s - 2\sigma\mu\mathbf{1}^{\mathrm{T}}\mathbf{1} + \sigma^2\mu^2\frac{n}{\gamma\mu}] \\ &\le 2^{-3/2}[1 - 2\sigma + \frac{\sigma^2}{\gamma}]n\mu \le 2^{-3/2}(1 + 1/\gamma)n\mu.\end{aligned}$$

路径追踪算法步长的下界

引理 (路径追踪算法步长的下界)

设$(x,y,s)\in\mathcal{N}_{-\infty}(\gamma)$, 记

$$(x(\alpha),y(\alpha),s(\alpha))=(x,y,s)+\alpha(\Delta x,\Delta y,\Delta s).$$

则对任意的$\alpha\in\left[0,2^{3/2}\gamma\frac{1-\gamma}{1+\gamma}\frac{\sigma}{n}\right]$, 有

$$(x(\alpha),y(\alpha),s(\alpha))\in\mathcal{N}_{-\infty}(\gamma).$$

在路径追踪算法算法中至少可以选取

$$\alpha_k=2^{3/2}\frac{\sigma}{n}\gamma\frac{1-\gamma}{1+\gamma},$$

虽然这样选取的α_k 不一定是最大的

证明

- 任取$i = 1, 2, \ldots, n$, 由对$\Delta x \odot \Delta s$ 的估计得

$$|\Delta x_i^k \Delta s_i^k| \leq 2^{-3/2}(1 + 1/\gamma)n\mu_k.$$

- 由$x_i s_i \geq \gamma\mu, L_s\Delta x + L_x\Delta s = -x \odot s + \sigma\mu\mathbf{1}$ 及上式得

$$\begin{aligned} x_i^k(\alpha)s_i^k(\alpha) &= \left(x_i^k + \alpha\Delta x_i^k\right)\left(s_i^k + \alpha\Delta s_i^k\right) \\ &= x_i^k s_i^k + \alpha\left(x_i^k\Delta s_i^k + s_i^k\Delta x_i^k\right) + \alpha^2\Delta x_i^k\Delta s_i^k \\ &\geq x_i^k s_i^k(1-\alpha) + \alpha\sigma_k\mu_k - \alpha^2\left|\Delta x_i^k\Delta s_i^k\right| \\ &\geq \gamma(1-\alpha)\mu_k + \alpha\sigma_k\mu_k - \alpha^2 2^{-3/2}(1+1/\gamma)n\mu_k. \end{aligned}$$

- 定义$\mu_k(\alpha) = \frac{1}{n}\sum_i x_i^k(\alpha)s_i^k(\alpha)$, 则由$\sum_i \Delta x_i^k\Delta s_i^k = 0$ 与$L_s\Delta x + L_x\Delta s = -x \odot s + \sigma\mu\mathbf{1}$ 知

$$u_k(\alpha) = (1 - \alpha(1 - \sigma_k))\mu_k.$$

证明

- 于是为使$x_i^k(\alpha)s_i^k(\alpha) \geq \gamma\mu_k(\alpha)$, 只需

$$\gamma(1-\alpha)\mu_k + \alpha\sigma_k\mu_k - \alpha^2 2^{-3/2}(1+1/\gamma)n\mu_k \geq \gamma\left(1-\alpha+\alpha\sigma_k\right)\mu_k$$

- 整理得

$$\alpha\sigma_k\mu_k(1-\gamma) \geq \alpha^2 2^{-3/2} n\mu_k(1+1/\gamma)$$

- 于是我们得到步长α 需满足的条件:

$$\alpha \leq \frac{2^{3/2}}{n}\sigma_k\gamma\frac{1-\gamma}{1+\gamma}$$

原始– 对偶算法的收敛性

定理 (原始– 对偶算法的收敛性)

给定参数$0 < \gamma < \sigma < 1$, 设$\mu_k = \frac{(x^k)^{\mathrm{T}} s^k}{n}$ 为算法产生的**对偶间隙**, 且初值$(x^0, y^0, s^0) \in \mathcal{N}_{-\infty}(\gamma)$, 则存在与维数$n$ 无关的常数δ, 使得对任意k 有

$$\mu_{k+1} \leq \left(1 - \frac{\delta}{n}\right)\mu_k.$$

且对任意给定的精度$\varepsilon \in (0, 1)$, 存在迭代步数$K = \mathcal{O}\left(n \ln \frac{1}{\varepsilon}\right)$, 使得

$$\mu_k \leq \varepsilon \mu_0, \quad \forall k \geq K.$$

- 定理表明对偶间隙是呈指数式趋于0 的, 且当维数n 越大时收敛于0 的速度也就越慢.
- 这个结果揭示了内点法确实可以做到在多项式时间内产生给定精度的解, 从这方面来看它比单纯形法更加快速.

对偶间隙的估计

- 利用$(\Delta x)^{\mathrm{T}}\Delta s = 0$ 与$(x^k)^{\mathrm{T}}\Delta s^k + (s^k)^{\mathrm{T}}\Delta x^k = (\sigma - 1)x^{\mathrm{T}}s$ 得

$$\begin{aligned}\mu_{k+1} &= x^k(\alpha_k)^{\mathrm{T}}s^k(\alpha_k)/n \\ &= [(x^k)^{\mathrm{T}}s^k + \alpha_k((x^k)^{\mathrm{T}}\Delta s^k + (s^k)^{\mathrm{T}}\Delta x^k) + \alpha_k^2(\Delta x^k)^{\mathrm{T}}\Delta s^k]/n \\ &= \mu_k + \alpha_k(-(x^k)^{\mathrm{T}}s^k/n + \sigma_k\mu_k) \\ &= (1 - \alpha_k(1 - \sigma_k))\mu_k \\ &\leq (1 - \frac{2^{3/2}}{n}\gamma\frac{1-\gamma}{1+\gamma}\sigma_k(1 - \sigma_k))\mu_k\end{aligned}$$

- 由于$\sigma(1-\sigma)$ 是σ 的凹函数, 故任取$\sigma_k \in [\sigma_{\min}, \sigma_{\max}]$

$$\sigma_k(1 - \sigma_k) \geq \min\{\sigma_{\min}(1 - \sigma_{\min}), \sigma_{\max}(1 - \sigma_{\max})\}.$$

- 取

$$\delta = 2^{3/2}\gamma\frac{1-\gamma}{1+\gamma}\min\left\{\sigma_{\min}\left(1 - \sigma_{\min}\right), \sigma_{\max}\left(1 - \sigma_{\max}\right)\right\}$$

即完成证明.

迭代步数的估计

- 在对偶间隙的估计式两边取对数得

$$\log \mu_{k+1} \leq \log \left(1 - \frac{\delta}{n}\right) + \log \mu_k$$

- 递归地使用上述估计, 我们得到

$$\log \mu_k \leq k \log \left(1 - \frac{\delta}{n}\right) + \log \mu_0$$

- 任取$\beta > -1$, 有$\log(1+\beta) \leq \beta$, 故$\log(\mu_k/\mu_0) \leq k(-\frac{\delta}{n})$
- 因此欲使$\mu_k/\mu_0 \leq \epsilon$, 只需$k\left(-\frac{\delta}{n}\right) \leq \log \epsilon$, 这等价于

$$k \geq K \stackrel{\text{def}}{=} \frac{n}{\delta} \log \frac{1}{\epsilon} = \frac{n}{\delta} |\log \epsilon|$$

提纲

实用原始-对偶算法

- 实用原始-对偶算法的基本思想与原始-对偶算法相同: 在中心路径附近迭代并保持x, s 恒正
- 大部分实用原始-对偶算法往往从可行域外的初始点开始迭代
- 为了提升执行效率, 实用原始-对偶算法忽略了一些便于理论分析的性质, 同时加入了一些对实际表现有巨大提升的技巧

矫正步

- 在导出$(\Delta x, \Delta y, \Delta s)$应满足的方程时，为将等式

$$(x + \Delta x) \odot (s + \Delta s) = r_c$$

线性化，我们舍去了$\Delta x \odot \Delta s$这一非线性项，导致了系统性的误差

- 矫正步的目标就是修正这一误差
- 以中心参数$\sigma = 0$，步长$\alpha = 1$的纯牛顿方向全步长所对应的情形为例，此时$r_c = -x \odot s$，于是

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^{\mathrm{T}} & I \\ L_s & 0 & L_x \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} r_p \\ r_d \\ -x \odot s \end{bmatrix} \Rightarrow s \odot \Delta x + x \odot \Delta s + x \odot s = 0$$

- 上式表明$(x + \Delta x) \odot (s + \Delta s) = \Delta x \odot \Delta s$，这正是舍去非线性项引入的误差

矫正步

- 一种自然的想法是通过求解

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^{\mathrm{T}} & I \\ L_s & 0 & L_x \end{bmatrix} \begin{bmatrix} \Delta x^{\mathrm{cor}} \\ \Delta y^{\mathrm{cor}} \\ \Delta s^{\mathrm{cor}} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -\Delta x \odot \Delta s \end{bmatrix}$$

得到矫正步$(\Delta x^{\mathrm{cor}}, \Delta y^{\mathrm{cor}}, \Delta s^{\mathrm{cor}})$, 再将

$$(\Delta x, \Delta y, \Delta s) + (\Delta x^{\mathrm{cor}}, \Delta y^{\mathrm{cor}}, \Delta s^{\mathrm{cor}})$$

用作更新, 此时自然有

$$(x + \Delta x + \Delta x^{\mathrm{cor}}) \odot (s + \Delta s + \Delta s^{\mathrm{cor}}) = 0$$

- 在其他情形中仍可通过该技巧减小对偶间隙

中心参数的选取

- 除了构造矫正步方程的右端项, 纯牛顿方向也可用以选取合适的步长α 与中心参数σ
- 假设$(\Delta x, \Delta y, \Delta s)$ 是点(x, y, s) 对应的纯牛顿方向, 若沿该方向可显著降低对偶间隙, 则应尽量保留, 此时应选取较小的中心参数σ, 反之则应选取较大的中心参数σ, 以使下一迭代点更靠近中心路径
- 为实现上述想法, 首先计算原始与对偶变量对应的最大步长

$$\alpha^{\text{pri}} \stackrel{\text{def}}{=} \min\left(1, \min_{i:\Delta x_i<0} -\frac{x_i}{\Delta x_i}\right),$$

$$\alpha^{\text{dual}} \stackrel{\text{def}}{=} \min\left(1, \min_{i:\Delta s_i<0} -\frac{s_i}{\Delta s_i}\right),$$

即保证更新后x, s 仍为正的步长.

中心参数的选取

- 再计算特殊的对偶间隙

$$\hat{\mu} = (x + \alpha^{\text{pri}} \Delta x)^T (s + \alpha^{\text{dual}} \Delta s)/n$$

- 此时中心参数可选为

$$\sigma = (\frac{\hat{\mu}}{\mu})^3$$

该取法在实践中效果很好, 但未从理论上给出保证

- 需求解的矫正步变为

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^{\text{T}} & I \\ L_s & 0 & L_x \end{bmatrix} \begin{bmatrix} \Delta x^{\text{cor}} \\ \Delta y^{\text{cor}} \\ \Delta s^{\text{cor}} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \sigma\mu \mathbf{1} - \Delta x \odot \Delta s \end{bmatrix}$$

最终的更新方向为

$$(\Delta x, \Delta y, \Delta s) + (\Delta x^{\text{cor}}, \Delta y^{\text{cor}}, \Delta s^{\text{cor}})$$

步长选取

- 实用原始-对偶算法不直接考察迭代点是否在中心路径邻域, 而是分别计算原始变量x 与对偶变量s 在不违背非负约束时的最大步长
- 给定第k 步迭代点(x^k, y^k, s^k), 其中$(x^k, s^k) > 0$, 以及迭代方向$(\Delta x^k, \Delta y^k, \Delta s^k)$, 定义

$$\alpha_{k,\max}^{\text{pri}} \overset{\text{def}}{=} \min_{i:\Delta x_i^k<0} -\frac{x_i^k}{\Delta x_i^k}, \quad \alpha_{k,\max}^{\text{dual}} \overset{\text{def}}{=} \min_{i:\Delta s_i^k<0} -\frac{s_i^k}{\Delta s_i^k}$$

- 实用原始-对偶算法选取$\alpha_k^{\text{pri}} \in (0, \alpha_{k,\max}^{\text{pri}}), \alpha_k^{\text{dual}} \in (0, \alpha_{k,\max}^{\text{dual}})$, 并采取更新

$$x^{k+1} = x^k + \alpha_k^{\text{pri}} \Delta x^k, \quad (y^{k+1}, s^{k+1}) = (y^k, s^k) + \alpha_k^{\text{dual}} (\Delta y^k, \Delta s^k)$$

初始点选取

- 首先找到满足原始与对偶可行性的最小范数解

$$\min_{x} \frac{1}{2} x^{\mathrm{T}} x \quad \text{s.t. } Ax = b,$$

$$\min_{(y,s)} \frac{1}{2} s^{\mathrm{T}} s \quad \text{s.t. } A^{\mathrm{T}} y + s = c.$$

其解为

$$\tilde{x} = A^T (AA^T)^{-1} b, \quad \tilde{y} = (AA^T)^{-1} Ac, \quad \tilde{s} = c - A^T \tilde{y}$$

- 通常$\tilde{x}$与$\tilde{s}$中含有非正元素，不能作为初始点，此时可定义

$$\delta_x = \max(-(3/2) \min_i \tilde{x}_i, 0), \quad \delta_s = \max(-(3/2) \min_i \tilde{s}_i, 0)$$

并将变量修正为

$$\hat{x} = \tilde{x} + \delta_x \mathbf{1} \geq 0, \quad \hat{s} = \tilde{s} + \delta_s \mathbf{1} \geq 0$$

初始点选取

- 为确保初始点(x^0, y^0, s^0) 中x^0, s^0 不过分接近0, 同时保持一定的相似性, 还需定义

$$\hat{\delta}_x = \frac{1}{2}\frac{\hat{x}^T\hat{s}}{\mathbf{1}^T\hat{s}}, \quad \hat{\delta}_s = \frac{1}{2}\frac{\hat{x}^T\hat{s}}{\mathbf{1}^T\hat{x}}$$

  即分别用归一化的$\hat{x}$ 与$\hat{s}$ 为权重求其对偶的加权和

- 初始点取为

$$x^0 = \hat{x} + \hat{\delta}_x\mathbf{1}, \quad \lambda^0 = \tilde{\lambda}, \quad s^0 = \hat{s} + \hat{\delta}_s\mathbf{1}$$

预测-矫正算法(Mehrotra)

1. 按上述描述计算初始点(x^0, y^0, s^0), 并令$k = 0$
2. 计算(x^k, y^k, s^k) 对应的纯牛顿方向$(\Delta x^k, \Delta y^k, \Delta s^k)$, 即预测步
3. 基于纯牛顿方向计算实用的中心参数$\sigma = (\hat{\mu}/\mu)^3$
4. 求解带中心化项矫正步并得到最终的更新方向, 仍记为$(\Delta x^k, \Delta y^k, \Delta s^k)$
5. 选取$\alpha_k^{\text{pri}}, \alpha_k^{\text{dual}}$, 并做更新

$$x^{k+1} = x^k + \alpha_k^{\text{pri}} \Delta x$$
$$(y^{k+1}, s^{k+1}) = (y^k, s^k) + \alpha_k^{\text{dual}} (\Delta y, \Delta s)$$

检验是否满足终止条件, 否则令$k = k + 1$, 转步2

预测-矫正算法: 评注

- 上述Mehrotra算法无收敛性理论保证, 且可以找到使其发散的例子
- 可以通过添加一些限制使得该算法收敛, 但因为发散的例子很罕见, 实际代码往往不这么做
- 在线性规划问题不适定/无界时, 上述算法必然发散, 可通过一些启发式的算法预先检测出这些情形

提纲

势减小(POTENTIAL-REDUCTION)方法

- 势减小方法的步骤与路径追踪算法类似, 也需求解路径追踪算法的步方程, 但它的迭代不依赖于中心路径$\mathcal{C}$
- 该方法使用对数势函数Φ 来度量可行域中点的好坏, 并在每步迭代中减小势函数的值, 从而得到更好的点, 其具备如下性质:

$$\Phi \to \infty \text{ if } x_i s_i \to 0 \text{ for some } i, \text{ while } \mu = x^T s/n \to 0$$

$$\Phi \to -\infty \text{ 当且仅当} (x, y, s) \to \Omega$$

- 第一个性质保证了不会有$x_i s_i$ 单独趋于0, 从而迭代点始终位于非负象限, 第二个性质使得$\Phi \to -\infty$ 时迭代点接近解集Ω
- 一种典型取法为

$$\Phi_\rho(x, s) = \rho \log x^T s - \sum_{i=1}^{n} \log x_i s_i, \quad \rho > n$$

扩展: 原始-对偶算法应用于单调线性互补问题

$$s = Mx + q, \quad (x, s) \geq 0, \quad x^T s = 0$$

其中M 是半正定的$n \times n$ 矩阵, x, s 是待求解的变量, q 是$\mathbb{R}^n$ 中的向量

- 该问题类似于线性规划的KKT条件, 可类比于对偶间隙定义其互补度量$\mu = x^{\mathrm{T}} s/n$
- 该问题等价于

$$\begin{bmatrix} Mx + q - s \\ x \odot s \end{bmatrix} = 0, \quad (x, s) \geq 0$$

从而可用牛顿法推导其迭代步方程

$$\begin{bmatrix} M & -I \\ L_s & L_x \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta s \end{bmatrix} = \begin{bmatrix} -(Mx + q - s) \\ -x \odot s + \sigma\mu \mathbf{1} \end{bmatrix}$$

其中$\sigma \in (0, 1)$, 可利用实用原始-对偶算法中的技巧快速求解

扩展: 内点法应用于二次规划

$$\begin{aligned} \min_x \quad & q(x) = \frac{1}{2}x^T Gx + x^T c \\ \text{s.t.} \quad & Ax \geq b \end{aligned}$$

- 其KKT条件为

$$\begin{aligned} Gx - A^T\lambda + c &= 0 \\ Ax - b &\geq 0 \\ (Ax-b)_i\lambda_i &= 0, \\ \lambda &\geq 0 \end{aligned} \quad \overset{y=Ax-b}{\Longleftrightarrow} \quad \begin{aligned} Gx - A^T\lambda + c &= 0, \\ Ax - y - b &= 0, \\ y_i\lambda_i &= 0, \\ (y,\lambda) &\geq 0. \end{aligned}$$

- 定义互补度量$\mu = y^{\mathrm{T}}\lambda/n$, 其扰动的KKT条件为

$$F(x,y,\lambda;\sigma\mu) = \begin{bmatrix} Gx - A^T\lambda + c \\ Ax - y - b \\ \mathcal{Y}\Lambda e - \sigma\mu\mathbf{1} \end{bmatrix} = 0, \quad \sigma \in (0,1)$$

其中$\mathcal{Y} = \operatorname{diag}(y_1, y_2, \ldots, y_m), \Lambda = \operatorname{diag}(\lambda_1, \lambda_2, \ldots, \lambda_m)$.

扩展: 内点法应用于二次规划

利用牛顿法即得迭代步方程

$$\begin{bmatrix} G & 0 & -A^T \\ A & -I & 0 \\ 0 & \Lambda & \mathcal{Y} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta \lambda \end{bmatrix} = \begin{bmatrix} -r_d \\ -r_p \\ -\Lambda \mathcal{Y} \mathbf{1} + \sigma \mu \mathbf{1} \end{bmatrix}$$

其中

$$r_d = Gx - A^T \lambda + c, \quad r_p = Ax - y - b$$

再进行更新

$$(x^+, y^+, \lambda^+) = (x, y, \lambda) + \alpha(\Delta x, \Delta y, \Delta \lambda)$$

其中α 应在保证$(y^+, \lambda^+) > 0$ 时尽可能满足其他条件.