线性规划的有效算法*[1)]

卢新明　高自友　赵茂先
（山东矿业学院）

SOME EFFECTIVE ALGORITHMS FOR LINEAR PROGRAMMING

Lu Xin-ming　Gao Zi-you　Zhao Mao-xian
(*Shandong Mining Institute*)

Abstract

In this paper, an improved ellipsoid algorithm is given and a new pivoting method and a new pivoting operation are presented. Then, the pivoting operation and the improved ellipsoid algorithm are combined into a new polynomial algorithm. The complexity analysis and the numerical experiments are given.

一、记号与假设

设讨论的线性规划为

(LP)　$\min f = \{c^T x \mid x \in R_0 \cap S_0 \cap E^n\}$.

其中，$R_0 = \{x \mid a_i^T x \leqslant b_i, i = 1,2,\cdots,m\}$, $S_0 = \{x \mid x^T Q_0^{-1} x \leqslant 1\}$, $Q_0 = \mathrm{diag}(2^L, 2^L, \cdots, 2^L)$, L 为给定的正整数.

假设 $A = [a_1, a_2, \cdots, a_m]^T$, $M = \{1,2,\cdots,m\}$, $N = \{1,2,\cdots,n\}$, $\mathrm{rank}(A) = n$, $\|a_i\|_2 = 1, i = 1,2,\cdots,m$.

如 $S \subset M$，则记 A 的第 j 列中行标在 S 中的元素按 S 中的顺序构成的列向量为 A_{Sj}.

如 $Z \subset N$，则记 A 的第 i 行中列标在 Z 中的元素按 Z 中的顺序构成的行向量为 A_{iZ}.

如 u 为一个向量，则记 u 中正分量的个数为 $\rho_+(u)$. 记集合 S 的元素个数为 $|S|$.

对任一个 $x \in R_0$，记 $J(x) = \{i \mid a_i^T x = b_i, i \in M\}$, $A_J = [a_i, i \in J(x)]^T$, $\hat{J}(x) = \{i \mid a_i$ 构成 $\{a_j, j \in J(x)\}$ 的一个极大线性无关组$\}$, $A_J = [a_i, i \in \hat{J}(x)]^T$, $\hat{n} = |\hat{J}(x)|$.

如 B_J 为 A_J 中的一个 $\hat{n} \times \hat{n}$ 非奇异子阵，则可把 A_J, x, c^T 划分为 $A_J = [B_J, N_J]$, $x = (x_{B_J}, x_{N_J})^T$, $c^T = (c_{B_J}^T, c_{N_J}^T)$.

---

* 1989 年 4 月 11 日收到.
1) 国家自然科学基金资助项目 (18901016).

设 $S_k=\{x|(x-x_k)^TQ_k^{-1}(x-x_k)\leqslant 1\}$ 为第 k 次迭代得到的椭球，x^* 为 (LP) 的最优解．本文规定出现下列情况之一时，计算终止

(i) $x_k=x^*$.

(ii) $c^T(x_k-x^*)\leqslant 2^{-q}$ 且 $x_k\in R_0$.

(iii) $\det(Q_k)\leqslant 2^{-(n+1)q}$.

(iv) $x_k\notin R_0\cup S_{k-1}$，当 $R_0\cap S_{k-1}=\phi$ 时可出现，见后文，其中，q 为给定的正整数．

文中用到的其它概念，如可行解，基可行解，基阵，乘子，退化和转轴运算等，见文献[1,2].

二、可行解到基可行解的迭代

如设 $\hat{x}\in R_0$，则可用下列算法把 $\hat{x}$ 转化为 R_0 的基可行解 $\tilde{x}$.

算法 A:

步骤 0. 取 $\lambda=\min\limits_{i\in M}\left\{\dfrac{b_i-a_i^T\hat{x}}{-a_i^Tc}\middle|a_i^Tc<0\right\}$，$\hat{x}\leftarrow\hat{x}-\lambda c$.

步骤 1. 取 $i\in J(\hat{x})$，$j\in N$，使 $a_{ij}\neq 0$. 置 $J\leftarrow J(\hat{x})$，$\hat{R}\leftarrow\{i\}$，$\hat{c}\leftarrow\{j\}$，$\tilde{J}\leftarrow J(\hat{x})-\{i\}$，$\hat{n}\leftarrow 1$，$B_J^{-1}=[a_{ij}^{-1}]$.

步骤 2. 若存在 $i\in\tilde{J}$，$j\in N-\hat{c}$，使 $\tilde{a}_{ij}=a_{ij}-A_{i\hat{c}}B_J^{-1}A_{\hat{R}j}\neq 0$，转步骤 3．否则转步骤 4.

步骤 3. 置 $\tilde{R}\leftarrow\tilde{R}\cup\{i\}$，$\hat{c}\leftarrow\hat{c}\cup\{j\}$，$\tilde{J}\leftarrow\tilde{J}-\{i\}$，$\hat{n}\leftarrow\hat{n}+1$，

$$B_J^{-1}\leftarrow\begin{bmatrix}B_J^{-1}+\dfrac{B_J^{-1}A_{\hat{R}j}A_{i\hat{c}}B_J^{-1}}{\tilde{a}_{ij}} & -\dfrac{B_J^{-1}A_{\hat{R}j}}{\tilde{a}_{ij}}\\ -\dfrac{A_{i\hat{c}}B_J^{-1}}{\tilde{a}_{ij}} & \dfrac{1}{\tilde{a}_{ij}}\end{bmatrix}\text{转步骤 2.}$$

步骤 4. 如果 $\hat{n}=n$，则置 $\tilde{x}\leftarrow\hat{x}$ 终止．否则令 $A_J=[B_JN_J]$，计算

$$u=c_{B_J}^TB_J^{-1},\quad d_{N_J}=(uN_J-c_{N_J}^T)^T,\quad d_N=\begin{cases}d_{N_J}, & d_{N_J}\neq 0\\ e, & d_{N_J}=0\end{cases}(e=(1,1,\cdots,1)^T),$$

$$\bar{d}=\begin{bmatrix}-B_J^{-1}N_Jd_N\\ d_N\end{bmatrix},\quad\lambda=\min_{i\in M-J}\left\{\frac{b_i-a_i^T\hat{x}}{a_i^T\bar{d}}\middle|a_i^T\bar{d}>0\right\}.$$

置 $\hat{x}\leftarrow\hat{x}+\lambda\bar{d}$，$\tilde{J}\leftarrow\{i|a_i^T\hat{x}=b_i,i\in M-J\}$，$J\leftarrow J\cup\tilde{J}$，转步骤 2.

定理 2.1. 如果 R_0 有界，则算法 A 的长操作数为 $O(n^2+mn^2)$.

证．由假设和算法可知，执行步骤 4 的次数至多为 $(n-1)$，而第 k 次迭代长操作数为 $2mn+(k-k^2)$．故执行步骤 4 的长操作数至多为 $l_1=2m(n-1)(n-2)+\frac{1}{2}(n-1)(n-2)-\frac{1}{6}(n-1)n(2n-1)$，而执行步骤 3 的长操作数为 $l_2=n(n-1)+2n(n+1)(2n+1)$，所以算法 A 的长操作数为 $O(l_1+l_2)=O(n^2+mn^2)$.

定理 2.2. 若 R_0 有界，则由算法 A 求出的基可行解 $\tilde{x}$ 满足 $c^T\tilde{x}\leqslant c^T\hat{x}$.

证. 由算法A可知，步骤0使 $c^T\hat{x}$ 不增，故只要证明在步骤4中有 $c^T(\hat{x}+\lambda\bar{d})\leqslant c^T\hat{x}$ 即可.

三、椭球法的进一步改进

求解不等式组 $\{a_i^Tx\leqslant b_i, i=1,2,\cdots,m\}$ 的 L.G. Khachiyan 的椭球法及深切变形的迭代步骤可简述为 $R_0=\{x|a_i^Tx\leqslant b_i, i\in M\}$, $S_k=\{x|(x-x_k)^TQ_k^{-1}(x-x_k)\leqslant 1\}$ 且 $R_0\cap S_k\neq\phi$. 如果存在 r，使 $a_r^Tx_k>b_r$，则新的椭球矩阵 Q_{k+1} 和球心 x_{k+1} 可描述为

$$\begin{cases} Q_{k+1}=\alpha\left[Q_k-\beta\dfrac{Q_ka_r(Q_ka_r)^T}{a_r^TQ_ka_r}\right], \\ x_{k+1}=x_k-\gamma\dfrac{Q_ka_r}{\sqrt{a_r^TQ_ka_r}}. \end{cases} \tag{3.1}$$

对于 L.G. Khachiyan 的椭球法[3]有

$$\alpha=\frac{n^2}{n^2-1},\quad \beta=\frac{2}{n+1},\quad \gamma=\frac{1}{n+1}. \tag{3.2}$$

对于 N.Z. Shor 和 V.I. Gershovich 的深切法[4]有

$$\alpha=\frac{n^2}{n^2-1}(1-d_r^2),\quad \beta=\frac{2(1+nd_r)}{(n+1)(1+d_r)},\quad \gamma=\frac{1+nd_r}{n+1}, \tag{3.3}$$

其中，

$$d_r=\frac{a_r^Tx_k-b_r}{\sqrt{a_r^TQ_ka_r}}.$$

对于(3.2)或(3.3)均有下列引理成立:

引理 3.1. 如果 $a_r^Tx_k>b_r$，则由(3.1)确定的 S_{k+1} 满足

$$\{x|(x-x_{k+1})^TQ_{k+1}^{-1}(x-x_{k+1})\leqslant 1\}=S_{k+1}\cap R_0\neq\phi.$$

引理 3.2. 设 $V(S_i)$ 为椭球 S_i 的体积，则对(3.1)有

$$\frac{V(S_{k+1})}{V(S_k)}\leqslant c=\sqrt{(1-\beta)\alpha^n}.$$

引理 3.1 与 3.2 可仿[3]中有关结果而得到.

由此可得到下列结果:

定理 3.1. 对 (3.1),(3.2)有 $\dfrac{V(S_{k+1})}{V(S_k)}\leqslant 2^{-\frac{1}{2(n+1)}}$.

定理 3.2. 对 (3.1),(3.3)若 $1\geqslant d_r\geqslant\sqrt{1-\delta+\dfrac{\delta}{n^2}}$ $(\delta\leqslant 1)$，则有

$$\frac{V(S_{k+1})}{V(S_k)}\leqslant\sqrt{(1-\beta)}\delta^{\frac{n}{2}}(\beta\leqslant 1).$$

证. 由条件 $d_r\geqslant\sqrt{1-\delta+\dfrac{\delta}{n^2}}$ 可得

$$\alpha = \frac{n^2}{n^2-1}(1-d_r^2) \leqslant \frac{n^2}{n^2-1}\left[1-\left(1-\delta+\frac{\delta}{n^2}\right)\right] = \delta.$$

再由引理 3.2 立得结果.

定理 3.2. 如果对任一 k 定理 3.1 的条件成立，且其中 δ 为常数，则当 $V(S_0) \leqslant 2^{nL}$ 时，只需进行 $O(q+L)$ 次迭代就有

$$V(S_k) \leqslant 2^{-q(n+1)}.$$

证. 由条件和定理 3.1 可得

$$V(S_k) \leqslant V(S_0)(\delta^{\frac{n}{2}})^k = 2^{nL}(\delta^{\frac{n}{2}})^k,$$

故当 $k \geqslant \dfrac{2\ln 2}{\ln\left(\dfrac{1}{\delta}\right)}\left[L+q+\dfrac{q}{n}\right]$ 时，就有 $V(S_k) \leqslant 2^{-q(n+1)}$.

由定理 3.1 和 3.2 可知，我们在采用(3.1),(3.3)迭代时，只要能多次取到较大的 d_r，就可以加快收缩速度，甚至得到线性迭代步数. 为此我们对[4]中的深切算法又作了下列改进：

(1) 如果相对于 S_k 中的 x_k 使多个约束不成立，则选取使下式成立的 r.

$$\theta = \max\{a_i^T Q_k a_i \mid a_i^T x_k > b_i, i \in M\} = a_r^T Q_k a_r. \tag{3.4}$$

因为 $\max\{a_i^T x \mid x \in S_k\} = a_i^T x_k + \sqrt{a_i^T Q_k a_i}$，故这样选取 r，再利用 (3.1),(3.3) 迭代后，可使 S_{k+1} 的形状靠近于球形.

(2) 如果有多个 r 使(3.4)成立，则取 r 使

$$a_r^T x_k - b_r = \max\{a_i^T x_k - b_i \mid a_i^T Q_k a_i = \theta, a_i^T x_k > b_i, i \in M\}.$$

这样可使 $d_r = \dfrac{a_r^T x_k - b_r}{\sqrt{a_r^T Q_k a_r}}$ 尽可能的大.

(3) 如果 $x_k \in R_0$，则沿 (LP) 在 S_k 的拟负梯度方向 $P = -Q_k c$ 搜索到 R_0 的边界得 $\hat{x} = x_k + \lambda P$，这里 $\lambda = \min\limits_{i \in M}\left\{\dfrac{b_i - a_i^T x_k}{a_i^T P} \middle| a_i^T P > 0\right\}$. 再利用算法 A 把 $\hat{x}$ 转化成 R_0 的极点 $\tilde{x}$，用单纯形法[1]或[2]判断 $\tilde{x}$ 是否是最优解，若是，终止. 否则用 $c^T x \leqslant \tilde{c}(=c^T\tilde{x})$ 去深切 S_k (即在(3.1), (3.3)中用 c 代 a_r，用 $\tilde{c}$ 代 b_r) 得 S_{k+1}. 因此只要 $d = \dfrac{c^T x_k - c^T\tilde{x}}{\sqrt{c^T Q_k c}}$ 接近于 1，$V(S_{k+1})$ 就接近于零.

改进的算法可描述为：

算法 MEA:

步骤 1. 置 $x_1 \leftarrow 0, k \leftarrow 1$.

步骤 2. 如果 $k \geqslant 2(n+1)^2(l+q)$，得到 x_k. 若有 $\max\limits_{i \in M}\{a_i^T x_k - b_i\} \leqslant 2^{-L}$，$x_k$ 为近似解. 否则 (LP) 无解，终止. 如果 $k < 2(n+1)^2(l+q)$ 则转步骤 3.

步骤 3. 如果 $x_k \in R_0$ 转步骤 8，否则转步骤 4.

步骤 4. 计算三维向量

$$w_i = (\mathrm{sign}(a_i^T x_k - b_i), a_i^T Q_k a_i, a_i^T x_k - b_i), i \in M.$$

取 $w_l = \operatorname{lexmax}_{i\in M} w_i$, $d = \dfrac{a_l^T x_k - b_l}{\sqrt{a_l^T Q_k a_l}}$, 置 $\bar{a} \leftarrow a_l$ 转步骤 5.

步骤 5. 若 $d > 1$, 则 $R_0 \cap S_k = \phi$, 终止. 若 $d < 1$, 转步骤 7. 若 $d = 1$, 转步骤 6.

步骤 6. 计算 $x = x_k + \dfrac{Q_k \bar{a}}{\sqrt{\bar{a}^T Q_k \bar{a}}}$, 如果 $x \notin R_0$, 则 $R_0 = \phi$, 如果 $x \in R_0$, 则 x 为 (LP) 的最优解, 终止.

步骤 7. 计算 $\alpha = \dfrac{n^2}{n^2-1}(1-d^2)$, $\beta = \dfrac{2(1+nd)}{(n+1)(1+d)}$, $\gamma = \dfrac{1+nd}{1+n}$. 置 $x_{k+1} \leftarrow x_k - \gamma \dfrac{Q_k \bar{a}}{\sqrt{\bar{a}^T Q_k \bar{a}}}$, $Q_{k+1} \leftarrow \alpha \left[Q_k - \beta \dfrac{Q_k \bar{a} (Q_k \bar{a})^T}{\bar{a}^T Q_k \bar{a}} \right]$, $k \leftarrow k+1$, 转步骤 2.

步骤 8. 若 $\sqrt{c^T Q_k c} \leqslant 2^{-q}$, 则得 x_k 为近似解, 终止. 否则计算 $P = -Q_k c$, $\lambda = \min_{i\in M} \left\{ \dfrac{b_i - a_i^T x_k}{a_i^T P} \,\middle|\, a_i^T P > 0 \right\}$, 置 $\hat{x} \leftarrow x_k + \lambda P$ 转步骤 9.

步骤 9. 用算法 A 把 $\hat{x}$ 转化 R_0 的极点 $\tilde{x}$, 并用单纯形法判断 $\tilde{x}$ 是否为最优解, 若是, 终止. 否则置 $d \leftarrow \dfrac{c^T(x_k - \tilde{x})}{\sqrt{c^T Q_k c}}$, $\bar{a} \leftarrow c$, 转步 7.

定理 3.3. (i) 若算法 MEA 终止于步 2, 则有 $\det(Q_k) \leqslant 2^{-(n+1)q}$. (ii) 若算法 MEA 终止于步 8, 则有 $c^T x_k - c^T x^* \leqslant 2^{-q}$, 且 $x_k \in R_0$. (iii) 若算法终止于步 5 或步 6 $x \notin R_0$, 则 $R_0 = \phi$ 或 $R_0 \cap S_k = \phi$. (iv) 若算法 MEA 终止于步 6 $x \in R_0$ 的情形, 则 $x = x^*$. (v). 若算法终止步 9, 则 $\tilde{x} = x^*$.

证. (i) 由椭球法[3]和 $0 \leqslant d \leqslant 1$ 可知, $\det(Q_{i+1}) \leqslant 2^{-\frac{1}{2(n+1)}} \det(Q_i)$, 而 $\det(Q_0) = 2^{nL}$, 故当 $k \geqslant 2(n+1)^2(l+q)$ 时, 必有

$$\det(Q_k) \leqslant 2^{-\frac{k}{2(n+1)}} 2^{nL} \leqslant 2^{-(n+1)q} \left[2^{[2(n+1)^2(l+q)-k]\frac{1}{2(n+1)}} \right] \leqslant 2^{-(n+1)q}.$$

(ii) 设 $S_k = \{x \mid (x - x_k)^T Q_k^{-1} (x - x_k) \leqslant 1\}$, 有 $c^T x_k - \sqrt{c^T Q_k c} = \min_{x\in S_k} c^T x \leqslant c^T x^*$ (因设 $x^* \in S_0$). 故当 $\sqrt{c^T Q c} \leqslant 2^{-q}$ 时, 有 $c^T x_k - c^T x^* \leqslant 2^{-q}$. 由算法知, 这时必有 $x_k \in R_0$.

(iii) 当算法终止于步 5 时, 有 $d = \dfrac{a_l^T x_k - b_l}{\sqrt{a_l^T Q_k a_l}} > 1$, 即 $a_l^T x_k > \sqrt{a_l^T Q_k a_l} + b_l$, 故有 $\min_{x\in S_k} a_l^T x = a_l^T x_k - \sqrt{a_l^T Q_k a_l^T} > b_l$, 即 $R_0 \cap S_k = \phi$ 或 $R_0 = \phi$.

当算法终止于步 6 且 $x \notin R_0$, 则 $\{y \mid a_l^T y \leqslant b_l\} \cap S_k = \{x\}$, 故 $R_0 \cap S_k = \phi$.

(iv) 这时由 $\{y \mid a_l^T y \leqslant b_l\} \cap S_k = x$, 且 $x \in R_0$ 立得.

(v) 自然可得.

对于算法 MEA 我们在 MC-68K 微机上进行了若干数值实验, 其结果与定理 3.2 估

计的相似.

四、新的转轴算法

对于一般的等式和不等式约束，已有一些直接的转轴方法[1]和处理退化的转轴法则[2]. 但已往关于转轴算法的各种改进只利用了当前极点的有关信息，而没有充分利用LP的几何性质. 本文把不等约束线性规划看作一个混合弱连通图. 图的顶点就是约束多面体的极点，图的边就是多面体的棱. 边的方向定义为目标函数的下降方向，如目标函数在某边上的值不变，该边定义为无向边. 特别对于非退化的有界问题，每个顶点的度恰为 n. 如果从任一顶点(极点)出发，要以较少步迭代达最优解，就需要以较少的步数走到出度为零的顶点. 而任一顶的每个度恰对应一个乘子，当乘子大于零时为出度. 基于这种想法，我们给出了这样的一个转轴算法，即每次迭代均沿下降边向出度最少的邻点移动. 若有多个这样的邻点出度达最少，则向其中目标函数下降最大的顶点移动. 这样每多一步可绕过较多的顶点. 其算法如下:

算法 B.

步骤 1. 对给定的极点 x_0，基指标集 $\hat{R}_0=\{i_1,i_2,\cdots,i_n\}$，基阵的逆 $B\bar{R}^1_0=[s_{i_1},s_{i_2},\cdots,s_{i_n}]$，计算乘子 $u^0\leftarrow(u^0_{i1},u^0_{i2},\cdots,u^0_{in})=c^TB\bar{R}^1_0$，$\bar{N}_0\leftarrow\rho_+(u^0),k\leftarrow 0$.

步骤 2. 若 $u^k\leqslant 0$，则 x_k 为 (LP)的解，终止. 否则求 $\Omega_k=\{i_j\mid u^k_{ij}>0,i_j\in\hat{R}_k\}$.

步骤 3. 如果 $|\Omega_k|=1$，取 $i_l\in\Omega_k$，转步 5. 否则转步 4.

步骤 4. 计算 $V_{ij}=[\rho_+(u^{k+1}_{ij}),-\delta^{k+1}_{ij},i_j],\forall i_j\in\Omega_k,V_i=\underset{i_j\in\Omega_k}{\text{lexmin}}V_{ij}$.

步骤 5. 以 a^T_{il} 为换出行，求出换入行 a^T_r，作转轴[1]

$$\hat{R}_{k+1}\leftarrow\{i_1,\cdots,i_{l-1},r,i_{l+1},\cdots,i_n\},$$
$$B\bar{R}^1_{k+1}\leftarrow B\bar{R}^1_k-S_{il}(a^T_rB\bar{R}^1_k-e^T_l)/a^T_rS_{il},$$
$$x_{k+1}\leftarrow B\bar{R}^1_{k+1}b_{\hat{R}_{k+1}},u^{k+1}\leftarrow c^TB\bar{R}^1_{k+1},\bar{N}_{k+1}\leftarrow\rho_+(u^{k+1}),k\leftarrow k+1,$$

转步 2.

在算法 B 中，u^{k+1}_{ij} 和 δ^{k+1}_{ij} 是以 a^T_{il} 为换出行，作一次转轴运算后的乘子向量和目标函数下降量.

定理 4.1. 若 (LP) 非退化，则算法 B 有限步终止.

该定理可从单纯形方法直接得到.

另外，显见算法 B 与[1,2]中的方法相比，每步迭代仅增加了 $O(mn)$ 的计算量，而且若干实验表明其迭代步数比已有的其它转轴算法少许多.

定理 4.2. 在算法 B 中，若 $\rho_+(u^{k+1})<\rho_+(u^k)$，则算法至多 n 步迭代终止.

证. 由于 $|u^k|=n$，且当 $\rho_+(u^k)=0$ 时，x_k 为最优解. 故结论成立.

定理 4.3. 设 (LP) 非退化，若第 k 个顶点只有一个下降棱，即 $\rho_+(u^k)=1$，且这时换出基的约束为 $a^T_{i_0}x\leqslant b_{i_0}$，则在以下迭代中恒有 $a^T_{i_0}x_{k+j}<b_{i_0}(j\geqslant 1)$.

证. 不妨设 $u^k=(u^k_1,u^k_2,\cdots,u^k_n)$，且 $u^k_1>0,u^k_2\leqslant 0,\cdots,u^k_n\leqslant 0$，即在第 k 次迭代时 $a^T_1x\leqslant b_1$ 换出.

若存在正整数 l, 使

$$a_1^T x_{k+l} = b_1, \tag{4.1}$$

则由 $a_j^T x^{k+l} \leqslant b_j (j = 1,2,\cdots,m)$ 得

$$\sum_{j=1}^{n} u_j^k a_j^T x^{k+l} = u_1^k b_1 + \sum_{j=2}^{n} u_j^k a_j^T x^{k+l} \geqslant u_1^k b_1 + \sum_{j=2}^{n} u_j^k b_j = \sum_{j=1}^{n} u_j^k b_j. \tag{4.2}$$

再由 $u^k = c^T B_k^{-1}$ 得

$$c^T x^{k+l} = u^k B_k x^{k+l} = \sum_{j=1}^{n} u_j^k a_j^T x^{k+l} \geqslant \sum_{j=1}^{n} u_j^k b_j = u^k b_{R_k} = u^k B_k x^k = c^T x^k. \tag{4.3}$$

显然(4.3)与非退假设和目标函数的严格下降性矛盾. (4.1)不可能成立.

定理 4.2 与 4.3 也说明了算法 B 所遵循的良好性质. 另外基于这两个定理,可以将算法 B 修改为一个纯的转轴运算,即当 $\rho_+(u^{k+1}) \geqslant \rho_+(u^k)$ 且 $\rho_+(u^{k+1}) > 1$ 时就终止于转轴的 k 步.

转轴运算 S:

步骤 1～步骤 3. 同算法 B 的步 1—步 3

步骤 4. 计算 $V_{ij} = (\rho_+(u_{ij}^{k+1}), -\delta_{ij}^{k+1}, i_j), \forall i_j \in \Omega_k$, $V_{il} = \operatorname{lexmin}_{i_j \in \Omega_k} V_{ij}$. 若 $\rho_+(u_{il}^{k+1}) < \bar{N}_{1c}$ 转步 5. 否则,终止.

步骤 5. 同算法 B 的步 5.

对于非退化的 (LP), 若干数值例子说明,连续执行运算 S 1～3 次即可求出最优解(这里的连续是指在两次之间执行一次转轴).

定理 4.4. 如 (LP) 非退化,则运算 S 至多有 $(n-1)(m-n)$ 步迭代终止.

该定理可直接由定理 4.2、4.3 和运算 S 得到.

五、算 法 MEAS

为了充分利用单纯形方法平均线性性和转轴运算 S 的优点,加速椭球法的收敛,我们在算法步 9 中对极点 $\tilde{x}$ 进行一次转轴运算 S. 得到新的极点 x^1, 以使尽快得到最优解或使 d 尽可能大.

算法 MEAS:

步骤 1～步骤 8. 同算法 ME 的步 1—步 8.

步骤 9. 用算法 A 把 $\tilde{x}$ 转化为极点 $\tilde{x}$, 同时得到基指标集 $\hat{R}$ 和基阵的逆 $B_{\hat{R}}^{-1}$.

步骤 10. 从 $\tilde{x}$, $\hat{R}$, $B_{\hat{R}}^{-1}$ 开始,执行转轴运算 S, 得 x^1. 转下步.

步骤 11. 如果 x^1 为最优解,终止. 否则,置 $d \leftarrow \dfrac{c^T x_k - c^T x^1}{\sqrt{c^T Q_k c}}$, $\bar{a} \leftarrow c$ 转步 7.

定理 5.1. 算法 MEAS 至多 $2(n+1)^2 l$ 步终止.

其证明可由算法 MEA, S 直接得到.

六、数 值 实 验

由定理 3.2,4.4 和凸多面体的性质可知,如果 x_k 距可行点较远,这时算法 MEAS 的

步 4 中的 d 必取较大的值，这时由定理 3.2 可知椭球可大幅度减小．如果 x_k 距可行点较近，则可很快进入可行集．这时就可利用运算 S 求出最优解或使椭球快速减小．当然后者对可行集体积 $V(R_0)>\varepsilon>0$ 时更为有效．不过对一般线性规划问题 $\min\{c^Tx\mid a_i^Tx\leqslant b_i, i\in M\}$．我们可利用大 M 方法将其变为 $\min\{c^Tx+\bar{M}x_{n+1}\mid a_i^Tx-x_{n+1}\leqslant b_i,\ i\in M\}$，这里 $\bar{M}$ 可取适当大的正数．这时 $\bar{R}=\{(x,x_{n+1})\mid a_i^Tx-x_{n+1}\leqslant b_i, i\in M\}$ 的体积充分大．

对于其它算法的数值结果从略．这里仅给出 MEAS 的若干算例．以下列出的迭代次数均指椭球收缩步数，所得到的解均为精确解．

表 1 给出的例 1,2,3,4 对应于文献[5]中的例 1,2,3,4．

表　1

例	1	2	3	4
$m\times n$	7×2	12×4	28×17	150×50
初始点可行性	不可行	不可行	不可行	不可行
迭代次数	2	6	11	52
CPU 时间	0.28 秒	0.56 秒	39.16 秒	12 分 37 秒
机　型	*MC*－68*K*	*MC*－68*K*	*MC*－68*K*	*MC*－68*K*

另外，对 MEAS 我们还进行了若干个随机实验，问题形式如下：

$$\min f=\sum_{i=1}^{n}c_ix_i,$$

$$(SP)\begin{cases}\sum_{i=1}^{n}a_{ij}x_i\leqslant b_i,\ i=1,2,\cdots,n,\\ x_i\leqslant u_i, i=1,2,\cdots,n,\\ -x_i\leqslant -v_i, i=1,2,\cdots,n,\end{cases}$$

其中，$a_{ij}(i=1,2,\cdots,n, j=1,2,\cdots,n)$ 是区间(1,10)中的随机数，$b_i=\sum_{j=1}^{n}a_{ij}+1$ $(i=1,2,\cdots,n)$．$c_j(j=1,2,\cdots,n)$ 是区间(−4,5)中的随机数，$u_i(i=1,2,\cdots,n)$ 是区间(10,100)中的随机数，$V_i(i=1,2,\cdots,n)$ 是区间(0.05,0.5)中的随机数．

对于问题 (SP)，显然 $a_1=(0,0,\cdots,0)^T$ 对每个问题都是不可行的，而 $a_2=(1,$

表　2

例	$m\times n$	迭代次数	*cpu* 时间	最优目标值
SP_1	30×10	1	9.01 秒	−25.86
SP_{13}	45×15	1	15.3 秒	−22.42
SP_7	60×20	24	1 分 52 秒	−44.37
SP_9	90×30	16	5 分 26 秒	−50.42
SP_{14}	120×40	100	28 分 31 秒	−85.03

$1,\cdots,1)^T$ 对每个约束都是可行的. 取 a_2 为初始点，计算结果列于表 2，取 a_1 为初始点，计算结果列于表 3 .

表 3

例	$m\times n$	迭代次数	cpu 时间	最优目标值
SP_1	30×10	13	9.51 秒	−25.86
SP_2	30×10	12	8.60 秒	−31.23
SP_3	30×10	13	11.30 秒	−20.88
SP_4	30×10	12	10.07 秒	−39.72
SP_5	60×20	21	2 分 35 秒	−83.26
SP_6	60×20	21	2 分 36 秒	−84.62
SP_7	60×20	41	2 分 31 秒	−44.37
SP_8	60×20	21	1 分 22 秒	−37.05
SP_9	90×30	31	8 分 05 秒	−56.42
SP_{10}	90×30	31	5 分 15 秒	−59.50
SP_{11}	90×30	31	5 分 16 秒	−60.62
SP_{12}	150×50	5	6 分 23 秒	−562.42

参 考 文 献

[1] 魏紫銮，具有自由变量线性规划问题的求解，数值计算与计算机应用，4: 3(1983),125—131.

[2] 卢新明，高自友，一般形式线性规划的转轴法则，山东矿业学院学报，2(1989),68—73.

[3] P. G'ács, L. Lov'ász, Khachigan's algorithm for Linear Programming, Mathematical Programming Study, 14(1981), 61—68.

[4] N. Z. Shor, V. I. Gershovich, Family of algorithms for solving convex programming problem, Kibernetika 15(4) (1979) 62—67; translated in: Cybernetics, 15(1979), 502—507.

[5] 刁在筠，Karmarkar 算法的一个变形，高校应用数学学报，1(1988),41—56.