Journal of Mathematical Research and Exposition
Vol.10 No.3 Aug. 1990

Some Improved Results on Ellipsoid Algorithm for Linear Programming*

Xu Shurong
(Zhongshan University, Guangzhou, China)

Nie Yiyong
(Shenyang Institute of Computing Techenology, Academia Sinica, Shenyang, China)

Abstract In the paper, new procedures on Khachiyan ellipsoid algorithm for linear programming are presented. The results of the paper are essentially the extended and improved versions of that previously given in Khachiyan's paper [1]. Application of the new algorithm for solving systems of linear equations are presented. The efficiency of new algorithms is demonstrated by numerical experiments.

§ 0. Introduction

It is well known that Khachiyan's ellipsoid algorithm is the first polynomial time algorithm for linear programming problems. The ellipsoid algorithm is of considerable interest from a theoretical point of view. However, it may not be satisfactory for some practical problems because of the slow convergence rates. Acceleration for ellipsoid algorithm and developing new efficient polynomial time algorithm have been attempted by many researchers.

In this paper, two improved algorithms based on the original Khachiyan paper [1] are presented. The reduction of the entry length and an accurate estimation on the maximum number of iterations are given. Applications of the improved algorithm for system of linear equations are summarized and many quantitative results for the solutions of the system can be obtained. Comparing the improved algorithms with the original algorithm by some numerical examples shows the efficiency of them.

§ 1. First Improved Algorithm

We consider the inequalities

$$Ax < b \tag{1.1}$$

* Received Oct. 22, 1988. This research work was supported in part by Chinese National Foundation of Natural Science and in Part by Zhongsha University Advanced Research Centre Foundation.

where $A=(a_{ij})$ is $m\times n$ matrix, x and b are n and m dimensional vectors, respectively. Let a_i denote the i-th row of matrix A. The entry length L is defined by

$$L=\sum_{i=1}^{m}\sum_{j=1}^{n}\log_2(|a_{ij}|+1)+\sum_{i=1}^{m}\log_2(|b_i|+1)+\log_2 mn+1. \tag{1.2}$$

Khachiyan [1] presented a polynomial-time algorithm for solution of (1.1), then Gacs and Lovasz [2] modified the algorithm as follows.

Let $x^0=0$, I_n be an identity matrix and $A^0=2^{2L}I_n$. Assume that we have x^k, A^k, then the following steps are implemented:

Step 1. Check $Ax^k<b$. If it's true, then x^k is a solution; otherwise there is i_k such that

$$a_{i_k}x^k-b_{i_k}\geqslant 0,\quad \text{set}\quad a=-a_{i_k}^T$$

Step 2. Let

$$x^{k+1}=x^k+(1/(n+1))(A^k a/\sqrt{a^TA^ka}) \tag{1.3}$$

$$A^{k+1}=(n^2/(n^2-1))(A^k-2/(n+1)\cdot(A^ka)(A^ka)^T/a^TA^ka). \tag{1.4}$$

Step 3. Go to Step 1.

The number of iterations is not more than $11n^2L$.

Using expressions (1.3) and (1.4) for computation on computer, one often finds that the round-off errors will change the positive definite property of A^k as k is increasing. It is possible to meet the mistakes such as square root of negetive number and division of two small numbers which are close to zero. There computing can not be continued. In order to put into operation of the ellipsoid algorithm on computer. We introduced the Discriminating Condition

$$\sqrt{a^TA^ka}<a_{i_0}x^k-b_{i_0} \tag{1.5}$$

and developed an improved algortihm agreed with the deeply cutting method. The algorithm is as follows.

Let $x^0=0$, $A^0=2^{2L}I_n$ be identity matrix multiplied by positive number. Assume that we have x^k, A^k, then the following steps are implemented:

Step 1. Check (1.1), if it is true, then x^k is a solution; otherwise there is i_0 such that

$$a_{i_0}x^k\geqslant b_{i_0}\quad\text{and}\quad a_{i_0}x^k-b_{i_0}=\max_i(a_ix^k-b_i).$$

Step 2. Check (1.5), if it's true, then the computation is stopped; other wise set $a=-a_{i_0}^T$ and

$$\mu=(a_{i_0}x^k-b_{i_0})/\sqrt{a^TA^ka}.$$

Step 3. Let

$$x^{k+1}=x^k+((\mu n+1)/(n+1))\cdot(A^ka/\sqrt{a^TA^ka}) \tag{1.6}$$

$$A^{k+1}=(n^2(1-\mu^2)/(n^2-1))(A^k-(2(\mu n+1)/(n+1)(1+\mu))\cdot((A^ka)(A^ka)^T/a^TA^ka)). \tag{1.7}$$

Step 4. Go to Step 1.

After $O(n^2L)$ iterations we shall be able to obtain a solution of (1.1) or assert that (1.1) is not consistent. Because of factor μ, the contraction of the ellipsoid is accelerated and the number of iterations is decreased in the improved algorithm, when $\mu=0$, improved algorithm (1.6) and (1.7) became original algorithm (1.3) and (1.4) respectively.

§ 2. Reduction for Entry Length and Maximum Number of Iterations.

The entry length L is larger than $m\cdot n$ by (1.2). The larger L is, the more iterations may be. Even 2^L may be infinite in the computer. Therefore we must reduce L as far as possble in theory. It's easy to show that L can be replaced by

$$L^*=\sum_{i=1}^{m}\log_2(b_i^2+\sum_{j=1}^{n}a_{ij}^2)+\log_2 n+0.11. \tag{2.1}$$

In fact we have

Lemma 2.1 Arbitrary vertex $V=(V_1,V_2,\cdots,V_n)^T$ of set $\{x|a_ix\leqslant b_i, i=1,2,\cdots, m, x\geqslant 0\}$. Satisfies

$$\|V\|_\infty=\max_j|V_j|<2^{L^*}/n.$$

Lemma 2.2 Let $\lambda(E^0\cap S)$ be the volume of intersection set of sphere $E^0=\{x|\|x\|_2\leqslant 2^{L^*}\}$ and solution set $s=\{x|a_ix<b_i,\ i=1,2,\cdots,m\}$, then

$$\lambda(E^0\cap S)>2^{-(n+1)L^*}.$$

Since L is replaced by L^* in A^0, the number of iterations is considerably reduced.

The maximum number of iterations $16n^2L$ in ellipsoid algorithm is higher estimation. We have

Theorem 2.1 Assume $n\geqslant 3$ in (1.1). Using the improved ellipsoid algorithm (1.6), (1.7) for

$$K_m'=2\ln 2(2n^2+3n+1)L^*<1.4(2n^2+3n+1) \tag{2.2'}$$

steps, we can obtain a solution of (1.1) or assert that (1.1) is not consistent.

Theorem 2.2 Assume $n\geqslant 2$ in (1.1). Using the improved ellipsoid algorithm (1.6), (1.7) for

$$K_m=(4n^2+6n+2)L^* \tag{2.2}$$

steps, we can obtain a solution of (1.1) or assert that (1.1) is not consistent.

In practice, the iterations determined by (1.5) are less than K_m and K_m' for inconsistent system.

The proof of above Lemma 2.1, 2.2 and theorem 2.2 can be found in [3].

§ 3. Second Improved Algorithm-Two-side Cutting Ellipsoid Algorithm

In the first improved ellipsoid algorithm a contraction of the ellipsoid is made from center x^k to tangent point P_k. If the ellipsoid is contracted in two directions from x^k to P_k and from P_k to x^k simultaneously, then the contraction of ellipsoid may be expected to quicken. Based on the idea, we construct the flowing second improved algorithm-two-side cutting ellipsoidal algorithm.

Step 1. Find i_0 such that

$$\rho_{i_0}^k/\gamma_{i_0}^k = \max_{1\le i\le m}\{\rho_i^k/\gamma_i^k\} \tag{3.1}$$

where

$$\rho_i^k = a_i x^k - b_i\ ,$$
$$\gamma_i^k = \sqrt{a_i A^k a_i^T} \qquad i = 1,2,\cdots, m.$$

Let $\mu_1 = \rho_{i_0}^k/\gamma_{i_0}^k$. Obviously, if $\mu_1 < 0$, x^k is a solution of (1.1); if $\mu_1 \ge 1$, there is no solution on the computer. Therefore we suppose $0 < \mu_1 < 1$.

Step 2. Let

$$z = A^k a_{i_0}^T/\gamma_{i_0}^k\ ,\quad f_i = a_i z,\quad i = 1,2,\cdots, m. \tag{3.2}$$

Step 3. If $\rho_i^k \ge f_i$; computing

$$\eta_i = (f_i/\gamma_i^k)(\rho_i^k/\gamma_i^k) + \sqrt{1-(\rho_i^k/\gamma_i^k)^2}\sqrt{1-(f_i/\gamma_i^k)^2} \tag{3.3}$$

Let $\mu_2 = \min\{\eta_i\}$. If $\rho_i^k < f_i$, $\forall i$, then $x^k - z$ is a solution.

Step 4. If $\mu_2 < \mu_1$, then (1.1) is not consistent; Otherwise let

$$\left.\begin{aligned}
&a_1 = (n+1)/(\mu_1+\mu_2),\\
&a_2 = 2(1-\mu_1^2) + n(\mu_2^2-\mu_1^2),\\
&a_3 = (1-\mu_1^2)(\mu_2-\mu_1),\\
&d = (a_2 - \sqrt{a_2^2 - 4a_1a_3})/(2a_1),\\
&v = (\mu_1+\mu_2)/(\mu_2-\mu_1-2d),\\
&x^{k+1} = x^k + (\mu_1+d)z,\\
&A^{k+1} = (1-\mu_1^2+vd^2)(A^k - 2(\mu_1+d)/(\mu_1+\mu_2)\cdot zz^T),\\
&\rho_i^{k+1} = \rho_i^k - (\mu_1+d)f_i\ ,\\
&\gamma_i^{k+1} = \sqrt{(1-\mu_1^2+vd^2)((\gamma_i^k)^2 - 2(\mu_1+d)/(\mu_1+\mu_2)\cdot f_i^2)}\ .
\end{aligned}\right\} \tag{3.4}$$

Step 5. Go to Step 1.

The second improved algorithm (3.1)- (3.4) became first improved algorithm (1.6), (1.7) when $\mu_2 = 1$. It also became original algorithm (1.3), (1.4) when $\mu_2 = 1, \mu_1 = 0$. Since there are factors μ_1 and μ_2 in algorithm (3.1)-(3.4). So the contraction of ellipsoid is quickened.

We prove the following two lemmas which assure the improved algorithm (3.1)-(3.4) to be valid.

Lemma 3.1 A^{k+1} is symmetric and positive definite matrix.

Proof Since A^k is symmetric, obviously A^{k+1} is also symmetric. According to the expression of d, it is easy to obtain $0<d<(\mu_2-\mu_1)/2$. So we have

$$0<2(\mu_1+d)<\mu_2+\mu_1, \quad \text{or} \quad 2(\mu_1+d)/(\mu_2+\mu_1)<1.$$

Therefore $\forall x\in R^n$, $(x\neq 0)$ we have

$$x^TA^{k+1}x=(1-\mu_1^2+vd^2)(x^TA^kx-2(\mu_1+d)/(\mu_1+\mu_2)\cdot x^TA^ka_{i_0}^Ta_{i_0}A^kx/a_{i_0}A^ka_{i_0}^T)$$
$$>(1-\mu_1^2+vd^2)(x^TA^kx\cdot a_{i_0}A^ka_{i_0}^T-(x^TA^ka_{i_0}^T)^2)/a_{i_0}A^ka_{i_0}^T.$$

According to positive definite of A^k, we obtain $x^TA^{k+1}x>0$, so A^{k+1} is a positive definite matrix.

Lemma 3.2 Let ellipsoid $E^k=\{x|(x-x^k)(A^k)^{-1}(x-x^k)\leqslant 1\}$, then E^{k+1} includes $E^k\cap\{z_1|\mu_1\leqslant z_1\leqslant\mu_2\}$.

Proof Let $z_1=\mu_1+\theta(\mu_2-\mu_1)$, $(0<\theta<1)$. We only need to prove the following expression

$$\mu_1^2+v\theta^2(\mu_2-\mu_1)^2-2\theta d(\mu_2-\mu_1)v\leqslant z_1^2$$

to hold. Since $\mu_2/\mu_1>1$ so $(\mu_2-\mu_1)(\theta-1)\mu_2/\mu_1\cdot d-(\theta+1)d<(\mu_2-\mu_1)-2d$. We have

$$2\theta(\mu_2-\mu_1)/(\mu_2-\mu_1-2d)\cdot(\mu_1+d)\theta(\mu_2-\mu_1)-d(\mu_2+\mu)<2\theta\mu_1(\mu_2-\mu_1).$$

Therefore

$$\mu_1^2+v\theta^2(\mu_2-\mu_1)^2-2\theta d(\mu_2-\mu)v=\mu_1^2+\theta^2(\mu_2-\mu_1^2)+2\theta(\mu_2-\mu_1)/(\mu_2-\mu_1-2d)$$
$$\cdot((\mu_1+d)\theta(\mu_2-\mu_1)\theta-d(\mu_1+\mu_2))<\mu_1^2+\theta^2(\mu_2-\mu_1)^2+2\theta\mu_1(\mu_2-\mu_1)$$
$$=(\mu_1+\theta(\mu_2-\mu_1))^2=z_1^2.$$

Since $(1-\mu_1^2-vd^2)-v(z_1-\mu_1-d)^2=1-(\mu_1^2+v\theta^2(\mu_2-\mu_1)^2-2\theta d(\mu_2-\mu_1)v)$, so $(1-\mu_1^2+vd^2)-v(z_1-\mu_1-d)^2\geqslant 1-z_1^2$ holds. Lemma is proved.

§ 4. Numerical Examples

In order to compare the efficiency as mentioned above ellipsoidal algorithms we computed some examples on an IBM 370 computer. One of them is as follows.

$$Ax=\begin{pmatrix}A_1\\A_2\end{pmatrix}x<\begin{pmatrix}b_1\\b_2\end{pmatrix}, \tag{4.1}$$

where $A_2=-I_9$ is 9 order identity matrix, b_2 is 9-dimension zero vector, and

$$A_1=\begin{bmatrix}1&1&1&0&0&0&0&0&0\\0&0&0&1&1&1&0&0&0\\0&0&0&0&0&0&1&1&1\\1&0&0&1&0&0&1&0&0\\0&1&0&0&1&0&0&1&0\\0&0&1&0&0&1&0&0&1\\-1&-1&-1&0&0&0&0&0&0\\0&0&0&1&-1&-1&0&0&0\\0&0&0&0&0&0&-1&-1&-1\\-1&0&0&-1&0&0&-1&0&0\\0&-1&-0&0&-1&0&0&-1&0\\0&0&1&1&0&-1&0&0&-1\\-5&-4&-7&6&-7&-3&-8&1&-2\end{bmatrix},\quad b_1=\begin{bmatrix}1+\varepsilon\\1+\varepsilon\\1+\varepsilon\\1+\varepsilon\\1+\varepsilon\\1+\varepsilon\\-1+\varepsilon\\-1+\varepsilon\\-1+\varepsilon\\-1+\varepsilon\\-1+\varepsilon\\-1+\varepsilon\\-1+\varepsilon\\-24+\varepsilon\end{bmatrix}$$

$\varepsilon=0.000005$.

We solve the inequalities (4.1) by means of five algorithms. Rounding results to four decimals, we obtain the same solution for five algorithms, that is

$$x_1 = x_2 = x_5 = x_6 = x_7 = x_9 = 0.0000, \quad x_3 = x_4 = x_8 = 1.0000.$$

The following table is a comparison of effciency among these algorithms, where L is entry length, K_m is maximum number of iterations in theory, K_p is maximum number of iterations in practice, T is CPU time.

Table 1

No.	Algorithm	L	K_m	K_p	T
1	Khachiyan Algorithm	71 computed by (1.2)	91361 ($K_m = 16n^2L$)	11463	24min 9.14sec
2	Khachiyan Algorithm	29 computed by (2.1)	37584 ($K_m = 16n^2L^*$)	6000	12min 58.1sec
3	Algorithm (1.3) (1.4)	29	10695 computed by (2.2)	4675	3min 17.53sec
4	Algorithm (1.6)(1.7)	29	10695	1315	Omin 57.46sec
5	Algorithm (3.1)(3.4)	29	10695	465	

§ 5. Applications to Linear Systems

We applied the improved ellipsoid algorithm to solving the systems of linear equations.

Suppose that

$$A_0 x = b_0 \tag{5.1}$$

where A_0 is $n \times n$ matrix, x and b_0 are n dimensional vectors. Each equality of (5.1) is replaced by two perturbation inequalities. For instance, we may consider that

$$A_0 x < b_0 + \varepsilon e, \quad -A_0 x < -b_0 + \varepsilon e \tag{5.2}$$

here $\varepsilon > 0, e = (1, \cdots, 1)^T$. If ε is sufficiently small, each solution of (5.2) will be approximate solution of (5.1). The errors between solution of (5.2) and (5.1) was described quantitatively in [4]. Clearly, (5.2) may be solved by the above improved ellipsoid algorithm.

We note the case in which there is no solution of (5.2). For certain computer the following possibilities to cause no solution:

(a) (5.1) is inconsistent, so is (5.2);

(b) ε is too small to solve (5.2) on the computer.

Using our improved algorithm we may show these possibilities with the maximum number of iterations and with discriminating condition; respectively. Therefore we may obtain many quantitative results for the solution of (5.1) with the aid of (5.2).

The efficiency of solving the equations (5.1) will be raised with each improvement of ellipsoid algorithm. It is quite natural that the second improved algorithm can be applied to solving (5.2).

Using the method described in the section, we have computed the solution of several linear systems successfully. One of them is linear system with bad conditions [6]

$$A_0 x = b_0 \tag{5.3}$$

where $A_0 = (a_{ij})$ is Hilbert matrix, $A_{ij} = \frac{1}{i+j}$, $b_0 = (b_i)$, $b_i = \sum_{j=1}^{n} a_{ij}$, $n = 40$.

We solve it on computer NORD-100 with mantissa 30. Using complete pivot Gaussian elimination and QR decomposition method the mistaken solutions (error exceed ± 1) are obtained. Then we applied as mentioned above improved algorithm (1.6) (1.7) to finding column perturbaton of (5.2), (5.3) and take $\varepsilon = 10^{-8}$. When number of iterations is 80, 1000 and 9000, the accuracy of the computing solutions to ideal solutions is 10^{-1}, 10^{-2}, 10^{-3}, respectively. We give the solutions with the number of iterations 9000 as follows:

1.00029, 0.99814, 1.00155, 1.00373, 0.99915, 0.99617, 0.99764, 0.99814,
1.00056, 1.00177, 1.00234, 1.00297, 1.00180, 1.00113, 1.00069, 0.99881,
0.99946, 0.99847, 0.99815, 0.99859, 0.99784, 0.99851, 1.00199, 0.99877,
0.99940, 0.99657, 1.00305, 1.00138, 1.00374, 0.99961, 0.99984, 1.00043,
0.99960, 0.99866, 1.00391, 0.99660, 1.00173, 1.00244, 0.99783, 0.99856.

The solutions are in accord with results of paper [6] which are computed on computer with mantissa 40. They have the same accuracy. This shows our algorithm is successful for solving linear systems with bad conditions.

References

[1] L. G. Khachiyan, Doklady Akademii Nauk SSSR 244 (1979), 1093—1096.

[2] P. Gacs, L. Lovasz, Mathematical programming study 14(1981), 61—68.

[3] Xu Shurong, Nie Yiyong, Acta Scientiarum Nuturalium Universitatis Sunxatseni, (1983), 4, 60—66.

[4] Nie Yiyong, Xu Shurong, Namerical Mathematics Journal of Chinese Universities, 6: 1 (1984), 25—34.

[5] Xu Shurong, Fan xiaoyun, Journal of engineering mathematics, Vol.4, No.1, (1987) p.105—107.

[6] Zhao jin-xi, Numerical Mathematics, Journal of Chinese Universities, 3:1 (1981), 8—17.

线性规划椭球算法若干改进结果

徐 树 荣　　聂 义 勇

（中山大学计算机科学系，广州）　（中国科学院沈阳计算技术研究所）

摘　　要

本文给出线性规划哈奇杨椭球算法的两个改进形式，推广了哈奇杨文[1]的结果，给出了对解线性代数方程组的应用和若干数值算例.