SCHOOL OF OPERATIONS RESEARCH
AND INDUSTRIAL ENGINEERING
COLLEGE OF ENGINEERING
CORNELL UNIVERSITY
ITHACA, NEW YORK 14853

TECHNICAL REPORT NO. 997.

January 1992

THE ELLIPSOID ALGORITHM USING PARALLEL CUTS[1]

by
Ai-Ping Liao and Michael J. Todd

[1]Research supported in part by NSF, AFORS and ONR through NSF grant DMS-8920550.

The Ellipsoid Algorithm Using Parallel Cuts[1]

Ai-Ping Liao and Michael J. Todd

School of Operations Research and Industrial Engineering,
Cornell University,
Ithaca, New York 14853.

Abstract

We present an ellipsoid algorithm using parallel cuts which is robust and conceptually simple. If the ratio of the distance between the parallel cuts under consideration and the corresponding radius of the current ellipsoid is less than or equal to some constant, it is called the "canonical case". Applying our algorithm to this case the volume of the next ellipsoid decreases by a factor which is, at worst, $\exp(-\frac{1}{2(n+1)})$. For the noncanonical case, we first add an extra constraint to make it a canonical case in a higher dimensional space, then apply our algorithm to this canonical case, and finally reduce it back to the original space.

Key words. Ellipsoid algorithm, parallel cuts, complexity, linear system.

[1]Research supported in part by NSF, AFORS and ONR through NSF grant DMS-8920550.

1 Introduction

We will be concerned with methods for determining the feasibility of a linear system. In particular, we are going to try to find an n-vector x such that

$$l \leq A^T x \leq u \tag{1}$$

where A is an $n \times m$ matrix and l, u are m-vectors. We assume $n, m \geq 2$ and A is of full rank. If $l \not\leq u$, then obviously, (1) is infeasible; and if there is an index, say i, such that $l_i = u_i$, then system (1) can be reduced to a subsystem obtained by deleting the i-th inequality and then solving (1) over the subspace $\{x \in R^n : a_i^T x = l_i (= u_i)\}$. If $m \leq n$, by performing a QR-factorization of A, we have

$$A = [Q_1, Q_2] \begin{bmatrix} R \\ 0 \end{bmatrix} = Q_1 R$$

and thus (1) is equivalent to

$$l \leq R^T Q_1^T x \leq u \tag{2}$$

which has solution(s) such that

$$Q_1^T x = R^{-T}(\frac{l+u}{2}).$$

Thus we will always assume $m > n$ and $l < u$. We also denote $r := \frac{l+u}{2}$ and $s := \frac{u-l}{2}$ for convenience.

In [2] Burrell and Todd proposed a parallel-cut ellipsoid algorithm based on the result of Todd [9].

Let

$$P = \{x \in R^n : l \leq A^T x \leq u\}. \tag{3}$$

P can be alternately written as

$$P = \{x \in R^n : (a_i^T x - l_i)(a_i^T x - u_i) \leq 0, i = 1, \ldots, m\} \tag{4}$$

where a_i is the i-th column of A and l_i, u_i are the corresponding components of $\boldsymbol{l}$, u respectively.

Now choose a nonnegative diagonal matrix $D = \text{diag}(d) = \text{diag}(d_1, \ldots, d_m)$, and combine the inequalities above with weights d_i. We thus obtain a set

$$E := E(d) := \{x \in R^n : (A^T x - l)^T D(A^T x - u) \leq 0\}. \tag{5}$$

It is obvious that $P \subset E$. We suppose that ADA^T is nonsingular. Thus E is actually an ellipsoid.

Further calculation shows that

$$E = \{x \in R^n : (x - x_c)^T ADA^T (x - x_c) \leq x_c^T (ADA^T) x_c - l^T Du\} \tag{6}$$

where

$$x_c := x_c(d) := (ADA^T)^{-1} ADr \tag{7}$$

is the center of E.

If the current center violates some constraint, say, $l_i \leq a_i^T x \leq u_i$, by the result of Todd [9] we can construct a new ellipsoid that contains that part of the previous one between the parallel hyperplanes $a_i^T x = l_i$ and $a_i^T x = u_i$, and the volume of the ellipsoid decreases by a factor which is, at worst, $\exp(-\frac{1}{2(n+1)})$.

As we know, the ellipsoid method has to be carried out to a high accuracy in order that each of the sequence $\{E_k\}$ of ellipsoids contains the feasible region P which makes it difficult to implement. But Burrell and Todd's algorithm has the advantage that, with the representation (5), P is contained in each E_k regardless of round-off error

as long as $D_k \geq 0$. In Section 2 we propose, with the "up and down" technique, a simpler scheme based on Burrell and Todd's algorithm; and its complexity analysis is given in Section 3. We also show that why it is necessary to update bounds in order to get a polynomial bound for this algorithm as well as the Burrell and Todd algorithm. Finally, in Section 4 we describe several variants of the algorithm.

2 The Algorithm

By calculation, the volume of E is

$$\begin{aligned} \text{vol}(E) &= \kappa_n \cdot \frac{(x_c^T A D A^T x_c - l^T D u)^{\frac{n}{2}}}{(\det(ADA^T))^{\frac{1}{2}}} \\ &=: \kappa_n \cdot (v(d))^{\frac{n}{2}} \end{aligned} \tag{8}$$

where κ_n is the volume of the unit ball in R^n, a constant depending on n only, and $v(d) = f(d) \cdot h(d)$, where

$$f(d) := x_c^T A D A^T x_c - l^T D u, \tag{9}$$

$$h(d) := (\det(ADA^T))^{-\frac{1}{n}}. \tag{10}$$

The idea of the method is as follows: we use the center of $E(d)$ in (6) as the test point, we focus on the function $v(d)$ and by using the coordinate descent algorithm to minimize $v(d)$, a polynomial bound can be obtained for the feasibility problem. More sophisticated methods for minimizing $v(d)$ are discussed in Liao [8]. Geometrically, if the current center violates some constraint, say, $l_j \leq a_j^T x \leq u_j$, then we increase d_j to make the j-th constraint "more important" and shrink the volume of E. In the "canonical case" (defined below in the Algorithm) a constant decreasing factor for $v(d)$ can be obtained; for the "non-canonical case", we lift up the problem to R_+^{m+1} space by

adding one extra constraint so that the "lifted" problem is a "canonical case", apply the algorithm to the canonical case and then reduce the problem back to R_+^m by combining the "lifting" constraint and the j-th one.

We now state the algorithm. Strictly speaking, this algorithm should be called a coordinate descent algorithm with scaling and updating modifications.

Algorithm 2.1

- Initialization. Choose $d^0 > 0$, and scale it so that $f(d^0) = 1$; set $k = 0$.

- For $k = 0, 1, \ldots,$ do

  If $x_c^k := x_c(d^k) \in P$, stop with the feasible solution x_c^k; otherwise, choose j with the j-th constraint violated by x_c^k. Let

$$\beta := \frac{s_j^2}{a_j^T (ADA^T)^{-1} a_j}, \quad \gamma := a_j^T (ADA^T)^{-1} a_j.$$

  (i) (canonical case) If $\beta \leq \frac{1}{4}$, we take $\lambda^k = \frac{1}{2n\gamma}$, and $d^{k+1} = d^k + \lambda^k e_j$.

  (ii) If $\beta > \frac{1}{4}$, we take $\lambda^k = \frac{1}{2n\gamma}$, and $d^{k+1} = d^k + \lambda^k e_j$, AND update the bounds as follows:

$$l_j \leftarrow \frac{d_j^k l_j + \tilde{d}_0^{k+1} l_0}{d_j^k + \tilde{d}_0^{k+1}} \quad \text{if } a_j^T x_c > u_j, \text{ or}$$

$$u_j \leftarrow \frac{d_j^k u_j + \tilde{d}_0^{k+1} u_0}{d_j^k + \tilde{d}_0^{k+1}} \quad \text{if } a_j^T x_c < l_j$$

  where $\tilde{d}_0^{k+1} = \frac{1}{2n\gamma}$ and

$$(l_0, u_0) = \begin{cases} (l_j, l_j + \sqrt{\gamma}) & \text{if } a_j^T x_c < l_j \\ (u_j - \sqrt{\gamma}, u_j) & \text{if } a_j^T x_c > u_j. \end{cases}$$

  Then, scale d^{k+1} so that $f(d^{k+1}) = 1$, where f is computed with the updated j-th bounds; set $k \leftarrow k + 1$, and repeat.

□

We can get better bounds l_0, u_0 from z_l and z_u which respectively minimize and maximize $a_0^T x$ over $E_k := E(d^k)$. This leads to

$$l_0 = \begin{cases} l_j & \text{if } a_j^T x_c < l_j \\ a_j^T z_l & \text{if } a_j^T x_c > u_j, \end{cases} \quad u_0 = \begin{cases} u_j & \text{if } a_j^T x_c > u_j \\ a_j^T z_u & \text{if } a_j^T x_c < l_j, \end{cases}$$

where

$$\begin{aligned} z_l &= x_c - \gamma^{-\frac{1}{2}}(ADA^T)^{-1}a_j, \\ z_u &= x_c + \gamma^{-\frac{1}{2}}(ADA^T)^{-1}a_j. \end{aligned}$$

Then, if $l_0 > u_j$ or $u_0 < l_j$, stop with the conclusion that the system is not consistent.

Note that case (ii) corresponds to an instance of case (i) in R_+^{m+1} space by adding a new constraint, say a 0-th constraint, $l_0 \leq a_0^T x \leq u_0$, where $a_0 = a_j$, with $d_0^k = 0$. It is easy to see that $\tilde{\beta} \leq \frac{1}{4}$ and $\tilde{x}_c^k$ violates the 0-th constraint, where $\tilde{\beta}$ and $\tilde{x}_c^k$ are the corresponding quantities in the new system. It thus becomes case (i) in this system. Therefore, we take $\lambda^k = \frac{1}{2n\gamma}$ and $\tilde{d}^{k+1} = \tilde{d}^k + \lambda^k e_0$, where $\tilde{d}^k := (0, (d^k)^T)^T \in R_+^{m+1}$. If we denote the quantities for the new system with tildes we have $\tilde{v}(\tilde{d}^k) = v(d^k)$. Note that, since $a_0 = a_j$,

$$\tilde{A}\tilde{D}\tilde{A}^T = \sum_{i=0}^{m} \tilde{d}_i a_i a_i^T = \sum_{i=1}^{m} d_i a_i a_i^T = ADA^T$$

where $d_i = \tilde{d}_i, i = 1, \ldots, m, i \neq j$, $d_j = \tilde{d}_0 + \tilde{d}_j$. Thus $\tilde{h}(\tilde{d}^{k+1}) = h(d^{k+1})$ where $d^{k+1} = d^k + \tilde{d}_0^{k+1} e_j$; on the other hand, direct calculation shows that $f(d^{k+1})$, with the updated bounds, is equal to $\tilde{f}(\tilde{d}^{k+1})$ with the original bounds and the 0-th bounds l_0, u_0. Therefore, $\tilde{v}(\tilde{d}^{k+1}) = v(d^{k+1})$. We also note that the system with the updated bounds has the same solution set as the original one. See Figure 1.

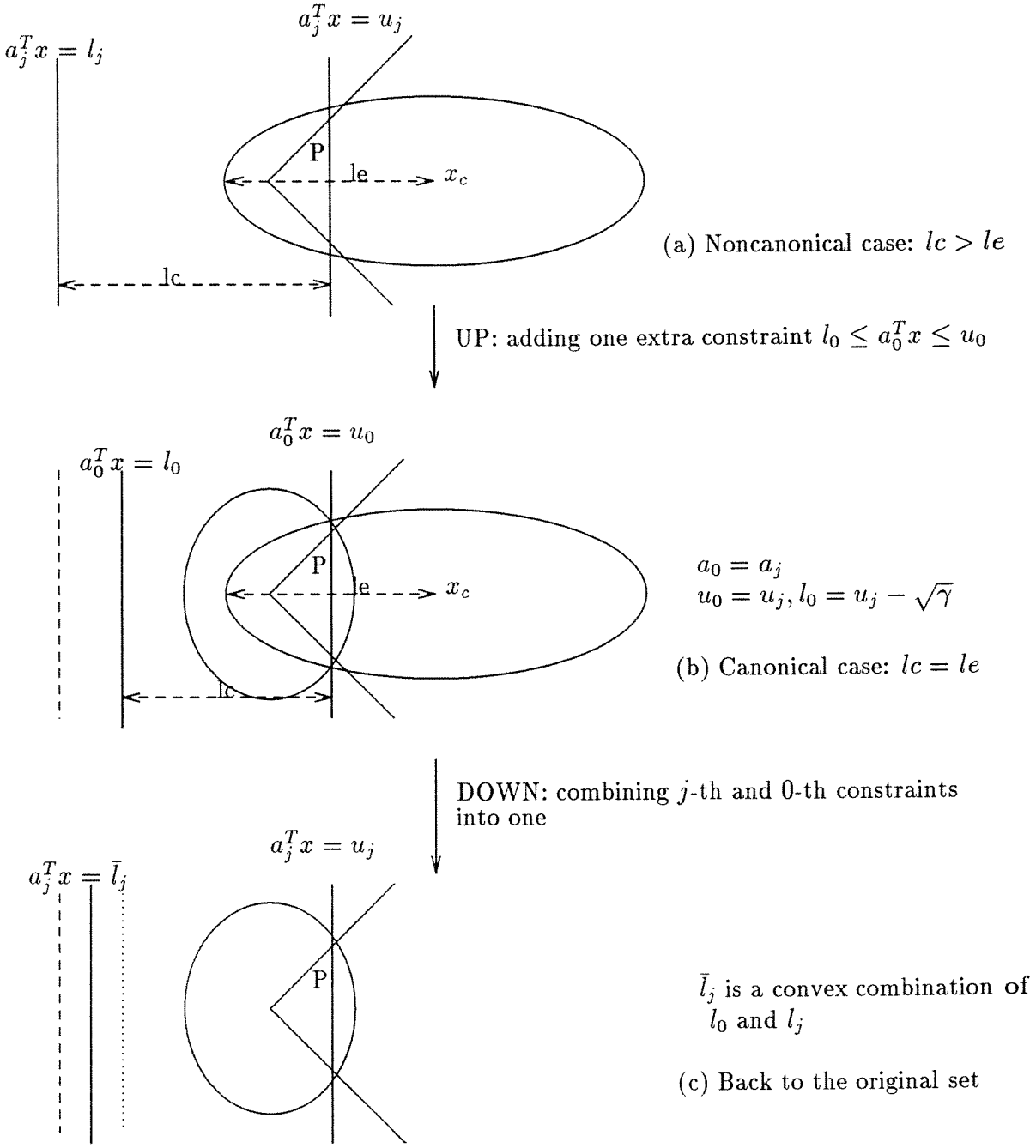

Figure 1: Up and down: dealing with the noncanonical case.

The above argument can be summarized as follows: case (ii) is essentially the same as case (i), i.e. if a constant reduction of $v(d)$ can be obtained in case (i), it can also be obtained in case (ii). We call the above method the "up and down" technique. We will see in the next section that $\beta \leq \frac{1}{4}$ is important in the analysis for getting a polynomial bound.

Finally, we note that scaling is not necessary if β is defined as

$$\beta := \frac{s_j^2}{f(d)a_j^T(ADA^T)^{-1}a_j};$$

and that the inverse of $AD_{k+1}A^T$ can be obtained from that of AD_kA^T by performing a rank-1 update by the Sherman-Morrison Formula. Thus our algorithm is as simple as the ellipsoid method yet more robust in practice.

3 Complexity Analysis

This section provides the analysis of the algorithm.

We denote the current point as d and the next point as $d^+ = d(\lambda) := d + \lambda e_j$; in general, quantities with "+" are those defined at d^+. Suppose that the j-th constraint is violated by the current center $x_c(d)$. By the argument in the previous section, it is enough to deal with case (i); we thus assume that $\beta \leq \frac{1}{4}$.

The following two rank-1 update formulae can be found, for example, in Householder [6].

Lemma 3.1 (Sherman-Morrison Formula) *Let u, v be two n-vectors, and M be a nonsingular $n \times n$ matrix. Then, if $\bar{\sigma} := 1 + v^T M^{-1} u \neq 0$, $M + uv^T$ is nonsingular and*

$$(M + uv^T)^{-1} = M^{-1} - \frac{1}{\bar{\sigma}} M^{-1} u v^T M^{-1}.$$

□

Lemma 3.2 *Under the same assumptions as in Lemma 3.1, we have*

$$\det(M + uv^T) = \det(M)(1 + v^T M^{-1} u).$$

□

Now we calculate $\ln v(d + \lambda e_j) - \ln v(d)$. By the Sherman-Morrison formula,

$$\begin{aligned} x_c^+ &= x_c(d + \lambda e_j) = (ADA^T + \lambda a_j a_j^T)^{-1}(ADr + \lambda a_j r_j) \\ &= (ADA^T)^{-1}ADr + \lambda r_j (ADA^T)^{-1} a_j - \frac{\lambda}{\sigma}(ADA^T)^{-1} a_j a_j^T (ADA^T)^{-1} ADr \\ &\quad - \frac{\lambda^2 \gamma}{\sigma} r_j (ADA^T)^{-1} a_j \\ &= x_c + \frac{\theta(r_j - a_j^T x_c)}{(1+\theta)\gamma}(ADA^T)^{-1} a_j \end{aligned}$$

where $\sigma := 1 + \lambda\gamma = 1 + \theta$, and $\theta := \lambda\gamma$. Correspondingly,

$$A^T x_c^+ = A^T x_c + \frac{\theta(r_j - a_j^T x_c)}{(1+\theta)\gamma} A^T (ADA^T)^{-1} a_j. \tag{11}$$

Thus, if we denote $\delta(f) := f(d^+) - f(d)$,

$$\begin{aligned} \delta(f) &= r^T D^+ A^T x_c^+ - l^T D^+ u - r^T D A^T x_c + l^T D u \\ &= r^T D A^T x_c^+ + \lambda r_j a_j^T x_c^+ - \lambda l_j u_j - r^T D A^T x_c \\ &= \frac{\theta(r_j - a_j^T x_c)}{(1+\theta)\gamma} a_j^T x_c + \lambda r_j a_j^T x_c + \frac{\lambda r_j \theta (r_j - a_j^T x_c)}{1+\theta} - \lambda l_j u_j \\ &= \frac{\lambda(r_j - a_j^T x_c) a_j^T x_c}{1+\theta} + \lambda r_j a_j^T x_c + \frac{\lambda r_j \theta (r_j - a_j^T x_c)}{1+\theta} - \lambda l_j u_j \\ &= \frac{\lambda}{1+\theta}[r_j a_j^T x_c - (a_j x_c)^2 + (1+\theta) r_j a_j^T x_c + \theta r_j^2 - \theta r_j a_j^T x_c - (1+\theta) l_j u_j] \\ &= \frac{\lambda}{1+\theta}[-(a_j^T x_c - l_j)(a_j^T x_c - u_j) + \theta s_j^2] \\ &= \frac{\beta\theta}{1+\theta}[\theta - \theta_0], \end{aligned} \tag{12}$$

where $\theta_0 := \frac{(a_j^T x_c - l_j)(a_j^T x_c - u_j)}{s_j^2}$. On the other hand, by Lemma 3.2,

$$\ln h(d^+) - \ln h(d) = -\frac{1}{n} \ln \frac{\det(AD^+A^T)}{\det(ADA^T)} = -\frac{1}{n} \ln(1+\theta). \tag{13}$$

Thus, noting that $f(d) = 1$,

$$q(\theta) := \ln v(d^+) - \ln v(d) \;\; = \;\; \ln(1 + \frac{\beta\theta}{1+\theta}(\theta - \theta_0)) - \frac{1}{n}\ln(1+\theta). \tag{14}$$

We note that since x_c does not satisfy $l_j \leq a_j^T x_c \leq u_j$, $\theta_0 > 0$. Thus,

$$\begin{aligned} \ln(1 + \beta\frac{\theta}{1+\theta}(\theta - \theta_0)) - \frac{1}{n}\ln(1+\theta) \leq \\ \leq \beta\frac{\theta^2}{1+\theta} - \frac{1}{n}\ln(1+\theta). \end{aligned} \tag{15}$$

Hence, if we choose $\theta = \frac{1}{2n}$, we have

$$\begin{aligned} \ln v(d^+) - \ln v(d) &\leq \beta\frac{1}{2n+1}\frac{1}{2n} - \frac{1}{n}\ln(1+\frac{1}{2n}) \\ &\leq \frac{1}{4}\cdot\frac{1}{2n(2n+1)} - \frac{1}{n}\ln(1+\frac{1}{2n}). \end{aligned}$$

One property of the logarithmic function is:

$$\ln(1+\delta) \geq \delta - \frac{\delta^2}{2(1-|\delta|)}, \text{ for } |\delta| < 1.$$

Thus, for $n \geq 2$,

$$\begin{aligned} \frac{1}{4}\cdot\frac{1}{2n(2n+1)} - \frac{1}{n}\ln(1+\frac{1}{2n}) &\leq \frac{1}{4}\cdot\frac{1}{4n^2+2n} - \frac{1}{n}(\frac{1}{2n} - \frac{1}{4(2n^2-n)}) \\ &\leq \frac{1}{n}(\frac{1}{4n} - \frac{1}{2n} + \frac{1}{4(2n^2-n)}) \\ &= -\frac{1}{n^2}(\frac{1}{4} - \frac{1}{4(2n-1)}) \\ &\leq -\frac{1}{n^2}(\frac{1}{4} - \frac{1}{12}) \leq -\frac{1}{6n^2}. \end{aligned}$$

Hence

$$\frac{v(d^+)}{v(d)} \leq \exp(-\frac{1}{6n^2}),$$

or, in term of the reduction in volume of the corresponding ellipsoids,

$$\frac{\text{volume}(E^+)}{\text{volume}(E)} = \left(\frac{v(d^+)}{v(d)}\right)^{\frac{n}{2}} \leq \exp(-\frac{1}{12n}).$$

We thus have

Theorem 3.3 *Suppose there is a known $\rho > 0$ such that, if P is nonempty, it contains a ball $\mathcal{B}(y^*, \rho)$ of radius ρ. Then the algorithm stops in at most*

$$12n \log(\frac{volume(E(d^0))}{volume(\mathcal{B}(Y^*, \rho))}) = 6n^2 \log(\frac{f(d^0)h(d^0)}{\rho^2})$$

steps.

□

For getting a polynomial bound for our algorithm, we suppose all the input data are integers and the input length of system (1) is:

$$L := \sum_{i=-1}^{n} \sum_{j=1}^{m} (\log(|a_{ij}| + 1) + 1) + 1$$

where $a_{(-1)j} = l_j$ and $a_{0j} = u_j$.

It is easy to see that the system (1) is equivalent to

$$\begin{aligned} A^T x &\leq u \\ -A^T x &\leq -l \end{aligned} \tag{16}$$

whose input length L' satisfies $L \leq L' \leq 2L$.

Since the assumption in the theorem may fail to hold, we consider the perturbed system:

$$\begin{aligned} A^T x &\leq u' := u + 2^{-L'} e \\ -A^T x &\leq -l' := -l + 2^{-L'} e \end{aligned} \tag{17}$$

which is equivalent to

$$l - 2^{-L'}e =: l' \leq A^T x \leq u' := u + 2^{-L'}e \tag{18}$$

and let P_p be its solution set; then one can prove (see, e.g., Khachian [7] or Gács and Lovász [3]) that $P = \emptyset$ if and only if $P_p = \emptyset$ and that, if $P \neq \emptyset$, there is some y^* with $\mathcal{B}(y^*, 2^{-2L'}) \subseteq P_p$. If we apply the algorithm to system (18) with the initial point $d^0 = e$, then

$$f(d^0)h(d^0) \leq f(e) = r'^T A^T (AA^T)^{-1} Ar' - l'^T u'^T$$

where $r' = \frac{l'+u'}{2}$, since $h(d^0) = h(e) \leq 1$ due to the nonsingularity of AA^T and the input data being all integers. Define

$$maxdet(AA^T) := \max\{|\det B| : B \text{ is a submatrix of } AA^T\};$$

then it can be shown that $maxdet(AA^T) \leq 2^{size(AA^T)}$, where

$$size(AA^T) := n^2 + \sum_{i,j=1}^{n} \log(|(a^i)^T a^j| + 1)$$

and we use superscripts to denote the rows of A. Using the Cauchy-Schwarz inequality and noting that $\|a^i\|_2 \leq \|a^i\|_1 \leq 2^L$ for all i, we have

$$size(AA^T) \leq n^2 + (2L+1)n^2 \leq 3n^2 L,$$

so that $maxdet(AA^T) \leq 2^{3n^2 L}$. Similarly, $maxdet(AA^T, Ar') \leq 2^{3n^2 L}$. Thus, by Cramer's theorem, each component of $(AA^T)^{-1}Ar'$ is less than or equal to $2^{3n^2 L}$. Therefore,

$$\begin{aligned} f(e) &= r'^T A^T (AA^T)^{-1} Ar' - l'^T u'^T \\ &\leq r'^T A^T (AA^T)^{-1} Ar' + m2^{2L} \end{aligned}$$

$$\begin{aligned} &\leq \sum_{j=1}^{n}(m2^{2L}\cdot 2^{3n^2L}) + m2^{2L} \\ &= nm2^{3n^2L+2L} + m2^{2L} \\ &\leq m(n+1)2^{3n^2L+2L}. \end{aligned}$$

Since

$$6n^2\log(\frac{f(d^0)h(d^0)}{\rho^2}) \leq 6n^2L(3n^2+10)\log(m(n+1)) \leq 24n^5(m+1)L,$$

we can determine feasibility of the system (1) in at most $24n^5(m+1)L$ steps. We note that this bound can be improved, for example, if we can find n columns of A, say the first n columns of A, which are linearly independent, by setting $d^0 = e_{1,n} := (e^T, 0)^T$ where $e \in R^n$. Then the bound becomes $24n^5(n+1)L$. Finally, we note that a feasible solution of (1) can be obtained in polynomial time from one of (18) using a rounding method (see Grötschel, Lovász and Schrijver [4]).

If l_j, u_j do not satisfy $\beta \leq \frac{1}{4}$, the constant decrease factor might not be bounded away from 1. From (14) we have

$$\begin{aligned} q'(\theta) &= (1+\frac{\beta\theta}{1+\theta}[\theta-\theta_0])^{-1}(\frac{\beta(2\theta-\theta_0+\theta^2)}{(1+\theta)^2}) - \frac{1}{n(1+\theta)} \\ &= \mu(\beta(1-\frac{1}{n})\theta^2 + (2\beta+\frac{\beta\theta_0}{n}-\frac{1}{n})\theta - (\beta\theta_0+\frac{1}{n})), \end{aligned} \tag{19}$$

where

$$\mu^{-1} = (1+\theta)^2(1+\beta\frac{\theta}{1+\theta}(\theta-\theta_0)).$$

If

$$\beta\theta_0 = \frac{(a_j^Tx_c - l_j)(a_j^Tx_c - u_j)}{\gamma} = \frac{1}{n},$$

then $q'(\theta_0) = 0$. Thus, by Lemma 4.2 below, $q(\theta)$ is minimized by $\theta = \theta_0$. So

$$\frac{v(d^+)}{v(d)} = (1+\theta_0)^{-\frac{1}{n}}$$

and if $\theta_0 \longrightarrow 0$ (so $\beta \longrightarrow \infty$), $\frac{v(d^+)}{v(d)} \longrightarrow 1$. Therefore, we cannot get a decrease factor that is bounded away from 1.

A typical example is when $d_j = 0$. In this case, $f(d), a_j^T(ADA^T)a_j$, and x_c do not depend on l_j, u_j, so it follows that we can let

$$\beta\theta_0 = \frac{1}{n}, \quad \frac{v(d^+)}{v(d)} \approx 1$$

by choosing l_j and u_j appropriately.

4 Variant Algorithms

In this section we give some variants of Algorithm 2.1.

Algorithm 4.1

Same as Algorithm 2.1 except that the step length in the canonical case is taken as

$$\lambda^* = argmin\{v(d^k + \lambda e_j) : \lambda \geq 0\},$$

and $d^{k+1} = d^k + \lambda^* e_j$. □

In the following we show that $v^n(d)$ is a pseudo-convex function of each component of d; thus λ^* can be obtained by setting the derivative of $v^n(d^k + \lambda e_j)$ with respect to λ to zero. We first cite Cauchy's formula which is a special case of the Cauchy-Binet formula that can be found, for example, in Horn and Johnson [5].

Lemma 4.1 (Cauchy's Formula) *Let A be an n by m and B be an m by n matrix, $m > n$, and let $C = AB$. Then*

$$\det C = \sum_{\gamma} \det A(,\gamma) \det B(\gamma,)$$

where the sum is taken over all index sets $\gamma \subset \{1, \ldots, m\}$ *of cardinality* n, $A(,\gamma)$ *is the submatrix formed by the columns of* A *whose indices are in* γ, *and* $B(\gamma,)$ *is analogously formed by the rows of* B *indexed by* γ.

□

Lemma 4.2 $v^n(d)$ *is a pseudo-convex function of each component of* $d \in R^m_+$.

Proof. By Cauchy's Formula,

$$\det(ADA^T) = \sum_{\gamma=\{i_1,\ldots,i_n\}} \alpha_{(\gamma)} d_{i_1} d_{i_2} \cdots d_{i_n}$$

where $\alpha_{(\gamma)} = \det A(,\gamma)^2 \geq 0$. Thus $h^{-n}(d)$ is a positive linear function of each component d_i; on the other hand, $f(d)$ is a convex function, hence so is f^n. By theorem 6.9 of Avriel [1], v^n is a pseudo-convex function of each d_i and d_i^* minimizes v^n(thus v) if $\frac{\partial v}{\partial d_i} = 0$. □

Since x_c^k violates the j-th constraint, $q'(0) < 0$ and so $\frac{\partial v(d^k)}{\partial d_j} < 0$; on the other hand, from (14), $v(d^k + \lambda e_j) \longrightarrow +\infty$ as $\lambda \longrightarrow +\infty$. Hence, λ^* exists. Actually, using our expression (19) for $q'(\theta)$, it can be explicitly expressed as $\lambda^* = \frac{\theta^*}{\gamma}$, where

$$\theta^* = \frac{\sqrt{\rho^2 + 4\beta(1-\frac{1}{n})(\beta\theta_0 + \frac{1}{n})} - \rho}{2\beta(1-\frac{1}{n})},$$

and

$$\rho = 2\beta - \frac{1}{n} + \frac{\beta\theta_0}{n}.$$

With this λ^* the corresponding ellipsoid E^{k+1} is the smallest one with $E^{k+1} \supset \{x \in E^k : l_j \leq a_j^T x \leq u_j\}$ ([2]). Thus, by the result of Todd [9], the volume of the corresponding ellipsoid decreases by a factor of $\exp(-\frac{1}{2(n+1)})$.

In our algorithms we use the so called "up and down" technique. As a matter of fact, the "down and up" technique is also possible; that is, when we meet a non-canonical case, instead of seeking help in a higher dimensional space, we seek help in a lower dimensional space. This leads to the Burrell-Todd algorithm [2].

Algorithm 4.2

Same as Algorithm 4.1 except that we use the following "down and up" technique for non-canonical case: we first delete the j-th constraint by setting

$$\tilde{d} := (d_1, \ldots, d_{j-1}, d_{j+1}, \ldots, d_m)^T \in R^{m-1},$$

and denote the quantities computed at $\tilde{d}$ with tildes. Similarly, let $\tilde{A}$ denote A with its j-th column deleted. We then set the new j-th lower and upper bounds l_j and u_j as

$$(\bar{l}_j, \bar{u}_j) = \begin{cases} (l_j, a_j^T \tilde{z}_u) & \text{if } a_j^T x_c < l_j \\ (a_j^T \tilde{z}_l, u_j) & \text{if } a_j^T x_c > u_j. \end{cases}$$

where

$$\begin{aligned} \tilde{z}_l &= \tilde{x}_c - \tilde{\gamma}^{-\frac{1}{2}} (\tilde{A}\tilde{D}\tilde{A}^T)^{-1} a_j, \\ \tilde{z}_u &= \tilde{x}_c + \tilde{\gamma}^{-\frac{1}{2}} (\tilde{A}\tilde{D}\tilde{A}^T)^{-1} a_j. \end{aligned}$$

are the minimizer and the maximizer, respectively, of the function $a_j^T x$ over $\tilde{E}$, the ellipsoid obtained by removing the effect of the j-th constraint on the current ellipsoid. Then take $\lambda^* = argmin\{v(d^k + \lambda e_j) : \lambda \geq 0\}$ where $v(d)$ is defined with the new j-th bounds, and $d^{k+1} = d^k + \lambda^* e_j$. □

Actually, Burrell and Todd [2] give even better updated bounds by considering the dual variables.

This algorithm also decreases the volume of the corresponding ellipsoids by a factor of $\exp(-\frac{1}{2(n+1)})$ at each step.

In the above algorithms, the significant decrease occurs in the so called "canonical case". We may also define other kinds of canonical cases and get different algorithms. But we should note that, in the previous algorithms the updated bound is at least as good as the previous one; on the other hand, in the following the updated bound may enlarge the feasibility region. To overcome this difficulty we use the original system (1) for testing the feasibility of $x_c(d^k)$; by doing so, the feasible solution to the original system will be not lost yet the volume of E^k keeps shrinking as long as x_c^k is not feasible. For convenience, we denote by l^k, u^k the current bounds at k-th iteration. The new canonical case is defined as follows:

If $a_j^T x_c > u_j$, $l_j = a_j^T z_l$; if $a_j^T x_c < l_j$, $u_j = a_j^T z_u$, where

$$\begin{aligned} z_l &= x_c - \gamma^{-\frac{1}{2}}(ADA^T)^{-1}a_j, \\ z_u &= x_c + \gamma^{-\frac{1}{2}}(ADA^T)^{-1}a_j. \end{aligned}$$

Algorithm 4.3

- Initialization. Choose $d^0 > 0$, scale it so that $f(d^0) = 1$ and let $(l^0, u^0) := (l, u)$; set $k = 0$.

- For $k = 0, 1, \ldots$, do

  If $x_c^k := x_c(d^k) \in P$, stop with the feasible solution x_c^k; otherwise, choose j with the j-th constraint violated by x_c^k. Let

  $$\beta := \frac{s_j^2}{a_j^T(ADA^T)^{-1}a_j}, \quad \gamma := a_j^T(ADA^T)^{-1}a_j$$

  where $s_j := \frac{u_j - l_j}{2}$.

  UP: We add a new constraint in current system:

  $$l_0 \leq a_0^T x \leq u_0$$

where $a_0 := a_j$ and

$$(l_0, u_0) = \begin{cases} (l_j, a_j^T z_u) & \text{if } a_j^T x_c < l_j \\ (a_j^T z_l, u_j) & \text{if } a_j^T x_c > u_j \end{cases}$$

where

$$\begin{aligned} z_l &= x_c^k - \gamma^{-\frac{1}{2}}(AD^kA^T)^{-1}a_j, \\ z_u &= x_c^k + \gamma^{-\frac{1}{2}}(AD^kA^T)^{-1}a_j. \end{aligned}$$

(If $l_0 > u_j$ or $u_0 < l_j$, stop with the conclusion that the system is not consistent.) It is thus the canonical case. Take $\tilde{d}^{k+1} = \tilde{d}^k + \lambda^k e_0$ with λ^k being either $\frac{1}{2n\gamma}$ or $argmin\{\tilde{v}(\tilde{d}^k + \lambda e_0) : \lambda \geq 0\}$. The quantities with tildes are defined in the same way as those in the paragraphs right after Algorithm 2.1.

DOWN: Set $d^{k+1} := d^k + \lambda^k e_j$, AND update the bounds as follows:

Let $\alpha = \frac{\tilde{d}_0^{k+1}}{d_j^k + \tilde{d}_0^{k+1}}$ and $\delta = 1 - \alpha$, $t_0 = l_0 u_0$, and $t_j^k = l_j^k u_j^k$. Set the new bounds as:

$$\begin{aligned} l_j^{k+1} &= \alpha r_0 + \delta r_j^k - \sqrt{(\alpha r_0 + \delta r_j^k)^2 - (\alpha t_0 + \delta t_j^k)} \\ u_j^{k+1} &= \alpha r_0 + \delta r_j^k + \sqrt{(\alpha r_0 + \delta r_j^k)^2 - (\alpha t_0 + \delta t_j^k)}, \end{aligned}$$

with other components remaining the same.

Then, scale d^{k+1} so that $f(d^{k+1}) = 1$, where f is computed with the updated j-th bounds; set $k \leftarrow k + 1$, and repeat.

□

It is easy to see that this "down" technique preserves the value of v. For the above algorithm to make sense, we have to show that:

(i) $(\alpha r_0 + \delta r_j^k)^2 - (\alpha t_0 + \delta t_j^k) \geq 0$;

(ii) $l_j^{k+1} \leq \max\{l_0, l_j^k\}$ and $u_j^{k+1} \geq \min\{u_0, u_j^k\}$.

If (i) holds then such l_j^{k+1} and u_j^{k+1} exist; (ii) ensures that the solution set of the updated system includes the original one.

For (i), we have

$$\begin{aligned}
(\alpha r_0 + \delta r_j^k)^2 - (\alpha t_0 + \delta t_j^k) &= \alpha^2 r_0^2 + \delta^2 (r_j^k)^2 + 2\alpha\delta r_0 r_j^k - \alpha^2 t_0 - \alpha\delta t_0 - \delta^2 t_j^k - \alpha\delta t_j^k \\
&= \alpha^2 s_0^2 + \delta^2 (s_j^k)^2 + 2\alpha\delta r_0 r_j^k - \alpha\delta t_0 - \alpha\delta t_j^k \\
&\geq 2\alpha\delta s_0 s_j^k + 2\alpha\delta r_0 r_j^k - \alpha\delta t_0 - \alpha\delta t_j^k \\
&= \alpha\delta(u_0 u_j^k + l_0 l_j^k - u_0 l_0 - u_j^k l_j^k) \\
&= \alpha\delta(u_j^k - l_0)(u_0 - l_j^k) \geq 0.
\end{aligned}$$

For (ii), we first suppose that $u_j^k \leq u_0$. We show that $u_j^{k+1} \geq u_j^k$. If $u_j^k - (\alpha r_0 + \delta r_j^k) \leq 0$ we are done, so we suppose otherwise. Then

$$\begin{aligned}
&\left(\sqrt{(\alpha r_0 + \delta r_j^k)^2 - (\alpha t_0 + \delta t_j^k)}\right)^2 - (u_j^k - (\alpha r_0 + \delta r_j^k))^2 \\
= \; & 2u_j^k(\alpha r_0 + \delta r_j^k) - (\alpha t_0 + \delta t_j^k) - (u_j^k)^2 \\
= \; & \alpha(u_j^k u_0 + u_j^k l_0 - u_0 l_0 - (u_j^k)^2) \\
= \; & \alpha(u_j^k - l_0)(u_0 - u_j^k) \geq 0.
\end{aligned}$$

Thus $u_j^{k+1} \geq u_j^k$. If $u_0 \leq u_j^k$, by exchanging the subscripts of j and 0 in the above argument, we have $u_j^{k+1} \geq u_0$. Therefore, $u_j^{k+1} \geq \min\{u_0, u_j^k\}$.

The same method can be used for showing $l_j^{k+1} \leq \max\{l_0, l_j^k\}$.

We note that Algorithm 4.3 with $\lambda^k := argmin\{\tilde{v}(\tilde{d}^k + \lambda e_0) : \lambda \geq 0\}$ is just the ellipsoid method with deep cuts with the "up and down" technique. That is, when

adding a new constraint Algorithm 4.1 does the same as the ellipsoid method with deep cuts. See Figure 2.

Finally, we have

Algorithm 4.4

Same as Algorithm 4.3 except that we define the canonical case as follows:

either $l_j = a_j^T z_l$ and $u_j = a_j^T x_c$, or $l_j = a_j^T x_c$ and $u_j = a_j^T z_u$, where

$$\begin{aligned} z_l &= x_c - \gamma^{-\frac{1}{2}}(ADA^T)^{-1}a_j, \\ z_u &= x_c + \gamma^{-\frac{1}{2}}(ADA^T)^{-1}a_j. \end{aligned}$$

□

We note that Algorithm 4.4 is just the ellipsoid method with the "up and down" technique. That is, when dealing with a canonical case, Algorithm 4.4 does the same as the ellipsoid method does. See Figure 3.

Finally, we can just use the "up" part of the "up and down" technique in all the above algorithms except Algorithm 4.2, and they are still polynomial algorithms for the feasibility problem, but the number of constraints is getting larger and larger. They can also be used to solve convex programs and preserve polynomiality.

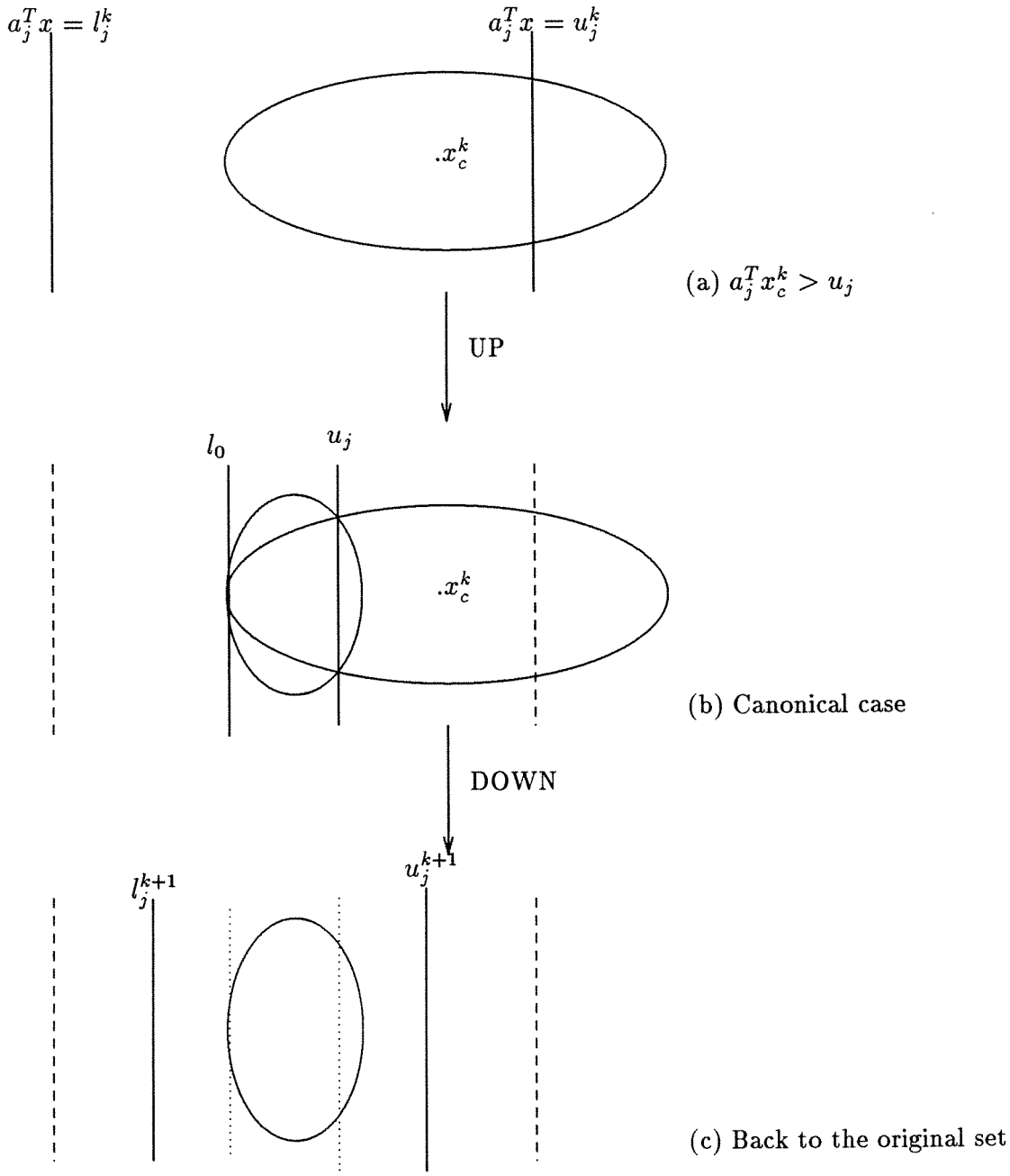

Figure 2: Algorithm 4.3.

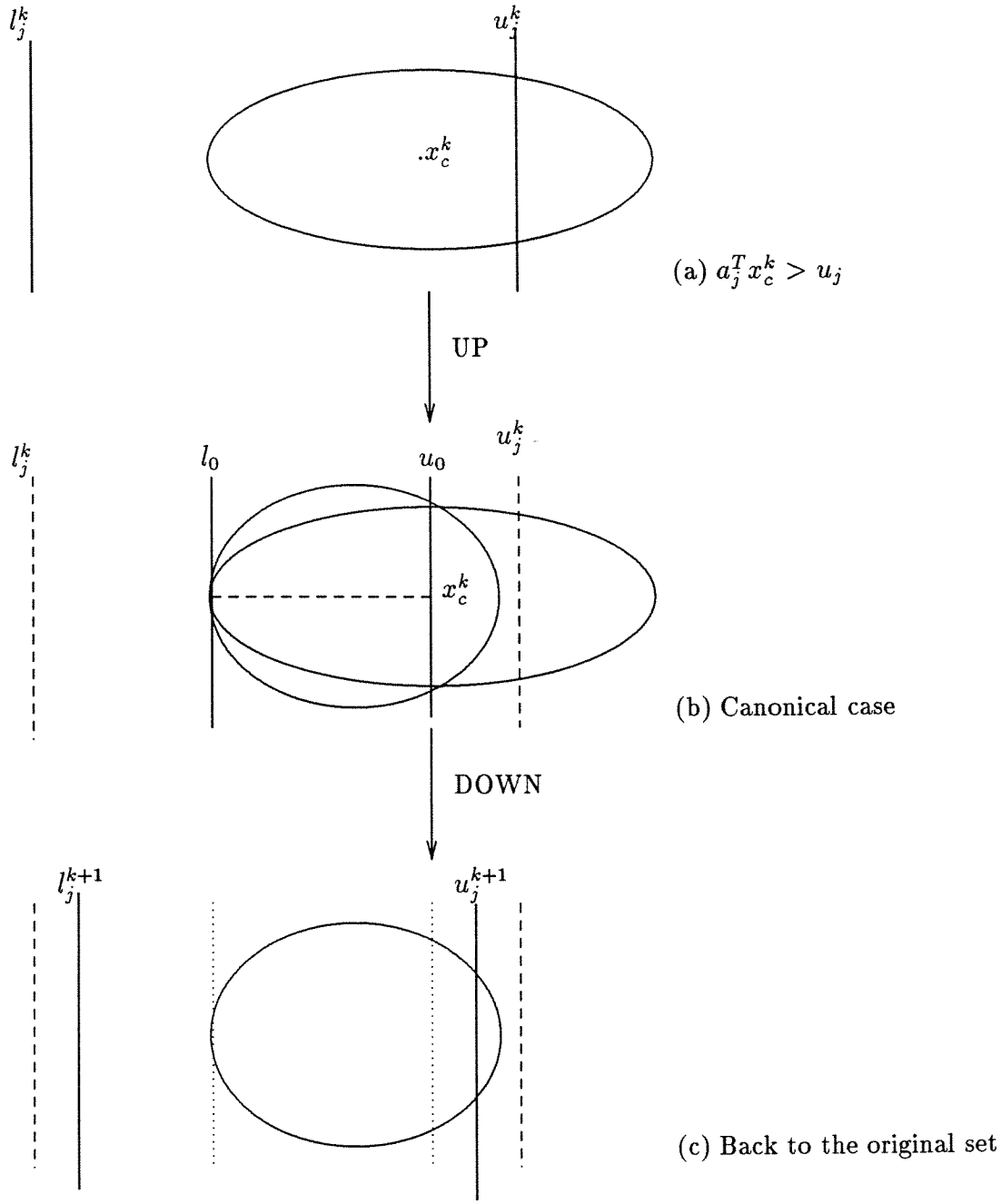

Figure 3: Algorithm 4.4.

References

[1] M. Avriel. *Nonlinear Programming: Analysis and Methods.* Prentice-Hall, Inc., 1976.

[2] B. P. Burrell and M. J. Todd. The ellipsoid method generates dual variables. *Mathematics of Operations Research*, 10:688–700, 1985.

[3] P. Gács and L. Lovász. Khachiyan's algorithm for linear programming. *Mathematical Programming Study*, 14:61–68, 1981.

[4] M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1:169–197, 1981.

[5] R. A. Horn and C. R. Johnson. *Matrix Analysis.* Cambridge University Press, 1990.

[6] A. S. Householder. *The Theory of Matrices in Numerical Analysis.* Ginn(Blaisdell), 1964.

[7] L. G. Khachian. Polynomial algorithms for linear programming. *USSR Computational Mathematics and Mathematical Physics*, 20:53–72, 1980.

[8] Ai-Ping Liao. *Algorithms for Linear Programming via Weighted Centers.* PhD thesis, Cornell University, Ithaca, New York, 1992.

[9] M. J. Todd. On minimum volume ellipsoids containing part of a given ellipsoid. *Mathematics of Operations Research*, 7:253–261, 1980.