PULL THE WEIGHTED CENTER TOWARDS THE SOLUTION OF LP

Aiping Liao[1]

Advanced Computing Research Institute
Cornell Theory Center
Cornell University
Ithaca, NY 14853

July 25 1993

Abstract

In the paper of Liao and Todd [3] two weighted centers are introduced and used to design algorithms for solving systems of linear inequalities. The linear programming problems can be solved via the weighted centers of a sequence of linear inequalities formed by letting the objective be an extra constraint and increasing the lower bound corresponding to the objective function as long as it is possible. In this paper we study the second kind of weighted center of [3] which is more computationally oriented and show that, under a regularity assumption, the weighted center of the linear inequality with the objective as an extra constraint converges to the solution of the linear programming problem under consideration as the upper bound corresponding to the objective function is pulled towards the infinity. We also propose a relaxed version of one of the algorithms of [3]. This modified version does not try to find an accurate center during each iteration; instead, an approximate center which is the k-th feasible iterate is determined in the k-th iteration. We show that this modified algorithm finds an ε-solution in finitely many iterations. Some limited numerical results are presented to compare our algorithm with the simplex method and indicate that our algorithm is promising.

Key words. weighted center, Newton's method, linear programming.

[1] This research was conducted using the resources of the Cornell Theory Center, which receives major funding from the National Science Foundation, and New York State. Additional funding comes from the Advanced Research Projects Agency, the National Institutes of Health, IBM Corporation and other members of the center's Corporate Research Institute.

1 Introduction

In this paper we will consider linear programming problems with the following form:

$$\begin{aligned} &\max c^T x \\ &l \le A^T x \le u \end{aligned} \qquad \text{(LP)}$$

where A is an $n \times m$ matrix with full rank. We denote by a_i the i-th column of A and l_i, u_i the corresponding components of l, u respectively. We assume that $m \ge n \ge 2$ and $l < u$, and define $r := \frac{1}{2}(l+u)$ for convenience. We also denote by e the all-one vector. As shown in Liao and Todd [3], this form of linear programming is not very restrictive in theory as well as in practice since most linear programming problems can be reduced to form (LP).

Liao and Todd [3] introduce two weighted centers associated with the linear system

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

These centers are actually the centers of the ellipsoids with smallest volume among certain sets of ellipsoids that contain P, the feasible set of (1). In the following we are mainly interested in the second kind of weighted center which is more computationally oriented. This weighted center is defined by $x_c(d^*) := (AD^*A^T)^{-1}AD^*r$, where $D^* := \mathrm{diag}(d^*)$ and the weight $d^* \in R^m$ is the solution to:

$$\begin{aligned} &\min \;\; F(d) := f(d) + B(d) \\ &\text{s. t. } d \in \mathcal{D} \end{aligned} \qquad (P_{f+B})$$

where

$$\begin{aligned} f(d) &:= x_c^T ADA^T x_c - l^T Du, \\ x_c(d) &:= (ADA^T)^{-1} ADr, \\ B(d) &:= e^T d^{-1}, \\ \mathcal{D} &:= \{d : d \ge 0, (ADA^T) \text{ is nonsingular}\}. \end{aligned}$$

Liao and Todd [3] prove that if $\mathrm{int}(P) \ne \emptyset$ then (P_{f+B}) has a unique solution, say d^*, and the associated center $x_c(d^*)$ is an interior point of P. Noting that $f(d)$ and $B(d)$ are homogeneous functions of degree 1 and -1 respectively, they thus propose a Newton-scaling algorithm:

Algorithm 1.1 [Algorithm 3.3 of Liao and Todd [3]]

- Initialization. Take $\tilde{d}^0 = e$ and scale it to d^0, i.e., $d^0 = \lambda^0 \tilde{d}^0$ with $\lambda = \sqrt{\frac{B(\tilde{d}^0)}{f(\tilde{d}^0)}}$; set $k = 0$.

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

  If a convergence condition holds, stop. Otherwise, perform a line search along the Newton direction, that is: take

  $$\tilde{d}^{k+1} = d^k + \lambda^* d_{nt}$$

  where $d_{nt} := -(\nabla^2 F(d^k))^{-1} \cdot \nabla F(d^k)$, and

  $$\lambda^* = \arg\min\{F(d^k + \lambda d_{nt}) : \lambda \geq 0\}.$$

  Let

  $$d^{k+1} = \lambda^{k+1} \tilde{d}^{k+1}$$

  with $\lambda^{k+1} = \sqrt{\frac{B(\tilde{d}^{k+1})}{f(\tilde{d}^{k+1})}}$. Set $k = k + 1$ and repeat.

$\square$

It is shown in [3] that if $\mathrm{int}(P) \neq \emptyset$ then the iterates generated by Algorithm 1.1 converges to d^* quadratically; and Algorithm 1.1 finds a feasible solution of system (1) in finitely many iterations. Based on Algorithm 1.1, Liao and Todd [3] propose a sliding objective algorithm for solving linear programming problem with form (LP).

Algorithm 1.2 [Algorithm 4.2 of Liao and Todd [3]]

- Initialization. Find approximately the center x_c^0 of the system:

  $$l \leq A^T x \leq u, \tag{2}$$

  and let d^0 be the corresponding weights. Let $l_0^0 := c^T x_c^0$ and

  $$\begin{aligned} u_0^{-1} &:= c^T x_c^0 - (f(d^0) c^T (A D^0 A^T)^{-1} c)^{\frac{1}{2}}, \\ u_0^0 &:= u_0^{-1} + q, \end{aligned}$$

  where $q \geq \frac{3n^2(u_0^{-1} - l_0^0)}{4}$. Set $A \leftarrow (c, A)$, i.e., let c be the 0-th column, and

  $$l^0 := (l_0^0, l^T)^T, \;\; u^0 := (u_0^0, u^T)^T.$$

  Set $k = 0$.

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

  (1) Use Algorithm 1.1 to solve, approximately, the system

$$l^k \leq A^T x \leq u^k, \qquad (FP)_k$$

  i.e., find d_k^* such that $\|\nabla F^k(d_k^*)\|_2 \leq \epsilon$ where ϵ is a small number which is determined by (A, c, l, u). The quantities with superscripts are those corresponding to $(FP)_k$. Then scale d_k^* and denote the scaled point d^k.

  (2) Let x_c^k be the current approximate center and set

$$l_0^{k+1} := c^T x_c^k \text{ and } l^{k+1} := (l_0^{k+1}, l^T)^T;$$

  and $u^{k+1} := (u_0^0, u^T)^T$. Set $k \leftarrow k+1$, and repeat.

$\square$

If we define the ε-solution x^ε as a feasible solution with $z^* - c^T x^\varepsilon \leq \varepsilon$ where z^* is the optimal objective value, then it is shown in [3] that Algorithm 1.2 provides an ε-solution of (LP) in finitely many iterations if $\mathrm{int}(P) \neq \emptyset$. It is also observed from the numerical tests in [3] that the factor q is important with regard to the number of major iterations. The bigger q, the fewer major iterations. In the next section we prove this observation under a regularity assumption. In section 3 a modified version of Algorithm 1.2 is proposed. This modified algorithm does not try to find an accurate center during each iteration; instead, an approximate center which is the k-th feasible iterate is determined in the k-th iteration. We show that this modified algorithm finds an ε-solution in finitely many iterations. Some implementation techniques are discussed in section 4, together with some limited numerical tests comparing our algorithm with the simplex method. The numerical results indicate that our algorithm is promising.

2 The pulling technique

According to Algorithm 1.2, the solution of (LP) can be found by solving a sequence of linear systems with the form

$$l_F \leq A_F^T x \leq u_F, \qquad (FP)$$

which is actually an expanded system of (1) by adding the constraint $l_0 \leq c^T x \leq u_0$ as the 0-th constraint: $A_F = [c, A]$, $l_F = [l_0, l^T]^T$ and $u_F = [u_0, u^T]^T$. We denote

$P_F = \{x : l_F \le A_F^T x \le u_F\}$. In this section we will prove that, under the regularity condition defined below, the center $x_c^*(u_0) := x_c(d^*)$ of system (FP) approaches the solution set of (LP) as $u_0 \longrightarrow \infty$.

The following definition of a regular point is similar to that of Luenberger [4].

Definition 1 *A point x^* satisfying the constraints of (LP) is said to be a regular point if the vectors a_j, $j \in J$ are linearly independent, where J, the active set associated with x^*, is the set of indices j for which $a_j^T x^* = l_j$ or $a_j^T x^* = u_j$.*

We now show that the function $F(d) = f(d) + B(d)$ associated with (FP) is invariant under affine transformations on variable x.

Lemma 2.1 *Let $\mathcal{T} : R^n \to R^n$ be an affine transformation, i.e., $\bar{x} := \mathcal{T}x := Tx + x_0$, where x_0 is some vector in R^n and T is an invertible matrix. Under this transformation, the system (FP) becomes*

$$\bar{l}_F \le \bar{A}_F^T \bar{x} \le \bar{u}_F, \qquad (\overline{FP})$$

where $\bar{l}_F = l_F + A_F^T T^{-1} x_0$, $\bar{u}_F = u_F + A_F^T T^{-1} x_0$ and $\bar{A}_F = T^{-T} A_F$. If we denote the quantities for the new system with bars then $\bar{x}_c(d) = Tx_c(d) + x_0$ and $\bar{F}(d) = F(d)$, for all $d > 0$.

Proof. Plugging $x = T^{-1}(\bar{x} - x_0) = T^{-1}\bar{x} - T^{-1}x_0$ into (FP) we have the equivalent system $(\overline{FP})$. By direct calculations, we have

$$\begin{aligned} \bar{x}_c(d) &= (\bar{A}_F D \bar{A}_F^T)^{-1} \bar{A}_F D \bar{r}_F \\ &= T(A_F D A_F^T)^{-1} A_F D \bar{r}_F \\ &= T(A_F D A_F^T)^{-1} A_F D (r_F + A_F^T T^{-1} x_0) \\ &= Tx_c(d) + x_0, \end{aligned}$$

and

$$\begin{aligned} \bar{f}(d) &= (\bar{x}_c(d))^T (\bar{A}_F D \bar{A}_F^T) \bar{x}_c(d) - \bar{l}_F^T D \bar{u}_F \\ &= (Tx_c(d) + x_0)^T (T^{-T} A_F D A_F^T T^{-1})(Tx_c(d) + x_0) - \bar{l}_F^T D \bar{u}_F \\ &= (x_c(d))^T (A_F D A_F^T) x_c(d) + 2x_0^T T^{-T} A_F D r_F + x_0^T T^{-T} A_F D A_F^T T^{-1} x_0 \\ &\quad -(l_F + A_F^T T^{-1} x_0)^T D (u_F + A_F^T T^{-1} x_0) \\ &= (x_c(d))^T (A_F D A_F^T) x_c(d) - l_F^T D u_F = f(d). \end{aligned}$$

Thus, $\bar{F}(d) = \bar{f}(d) + B(d) = f(d) + B(d) = F(d)$. □

It follows from Lemma 2.1 that there is a one-to-one correspondence between system (FP) and $(\overline{FP})$ and the objective function F does not change. Thus the pulling technique applies to system (FP), i.e.,

$$x_c(d^*) \longrightarrow x^*, \quad \text{as } u_0 \to \infty$$

where x^* is some solution of (LP), if and only if it applies to system $(\overline{FP})$, i.e.,

$$\bar{x}_c(d^*) \longrightarrow \bar{x}^*, \quad \text{as } \bar{u}_0 \to \infty$$

where $\bar{x}^* = Tx^* + x_0$.

The following lemma can be proven easily.

Lemma 2.2 *If we define a subproblem of a linear programming problem as another linear programming problem with a subset of the constraints of the original one, then any solution of this subproblem is a solution to the original one, provided that it is feasible to the original problem.*

$\square$

We now prove the main result.

Theorem 2.3 *Suppose that* $\mathrm{int}(P_F) \neq \emptyset$, $c \neq 0$ *and* $l_0 < \min\{c^T x : x \in P\}$. *Let* $x_c(d(u_0))$ *be the weighted center associated with the system (FP). Then any limit point of* $\{x_c(d(u_0))\}$ *as* $u_0 \longrightarrow \infty$ *is on the boundary of P; moreover, if* x^* *is a limit point of this sequence and* x^* *is a regular point, then* x^* *is a solution of (LP).*

Proof. Since A is of full rank, the feasible set P_F is thus uniformly bounded in u_0. $\mathrm{int}(P_F) \neq \emptyset$ implies that the weighted center $x_c(d(u_0))$ is always feasible and $d(u_0)$ satisfies

$$-(a_i^T x_c(d(u_0)) - l_i)(a_i^T x_c(d(u_0)) - u_i) = \frac{1}{(d(u_0))_i^2}, \text{ for } i = 0, 1, \ldots, m,$$

where $a_0 = c$. Obviously,

$$(c^T x_c(d(u_0)) - l_0)(d(u_0))_0^2 u_0 \to 1, \tag{3}$$

as $u_0 \to \infty$ (thus $(d(u_0))_0 \to 0$). Suppose that, as $u_0 \to \infty$, $\{x_c(d(u_0))\}$ stays away from the boundary of P. Then we have $(d(u_0))_i < \kappa$, for some $\kappa > 0$ and $i = 1, 2, \ldots, m$. By the definition $x_c(d(u_0)) = (ADA^T)^{-1}ADr$, we have

$$(ADA^T) \cdot x_c = ADr;$$

or

$$(c^T x_c - l_0) \cdot d_0 ADA^T x_c = (c^T x_c - l_0) \cdot d_0 ADr. \tag{4}$$

Let $u_0 \longrightarrow \infty$, the left hand side of (4) $\longrightarrow 0$ since $(c^T x_c - l_0)ADA^T x_c$ is bounded and $\lim_{u_0 \to \infty} d_0 = 0$. Noting that $d_0 u_0 \to \infty$ (by (3)), we have

$$\begin{aligned}
(c^T x_c - l_0) \cdot d_0 ADr &= (c^T x_c - l_0)(d_0)^2 u_0 ADr/(d_0 u_0) \\
&= (c^T x_c - l_0)(d_0)^2 u_0 A \begin{bmatrix} \frac{d_0 r_0}{d_0 u_0} \\ \frac{d_1 r_1}{d_0 u_0} \\ \vdots \\ \frac{d_m r_m}{d_0 u_0} \end{bmatrix} \\
&\to \frac{c}{2}.
\end{aligned}$$

Therefore, the right hand side of (4) $\longrightarrow \frac{c}{2}$ as $u_0 \to \infty$, which is a contradiction.

Now we suppose that x^* is a limit point of $\{x_c(d(u_0))\}$. Without loss of generality, we assume that the active set associated with x^* $J = \{1, 2, \ldots, |J|\}$ and $A = [A_1, A_2]$ with A_1 being nonsingular. Let $\mathcal{T}$ be an affine transformation defined by $\mathcal{T}(x) = A_1 x - r_1$ where $r_1 := r(1:n)$. Then under the transformation $x \to \mathcal{T}(x)$ (LP) is reduced to the form

$$\begin{array}{ll}
\max c^T x & \\
-u \le Ix \le u & \qquad \text{(LP2)} \\
\tilde{l} \le B^T x \le \tilde{u} &
\end{array}$$

where I is the n by n identity matrix, $B = A_1^{-T} A_2$, $u > 0$ and $c \leftarrow A_1^{-T} c$. It follows from Lemma 2.1 that the linear systems associated with (LP2) are equivalent to those associated with (LP). It is thus enough to show that the theorem is true for problem (LP2). Partitioning d into $d = [d_0, \bar{d}^T, \tilde{d}^T]^T$ according to the structure of (LP2) (we denote $A_F = [c, I, B]$) we thus have, using the Sherman-Morrisson formula (see Householder [2], for example),

$$\begin{aligned}
(A_F D A_F^T)^{-1} &= (d_0 cc^T + \bar{D} + B\tilde{D}B^T)^{-1} \\
&= (I + (d_0 cc^T + \bar{D})^{-1} B\tilde{D}B^T)^{-1} \cdot (d_0 cc^T + \bar{D})^{-1} \\
&= (I + (d_0 cc^T + \bar{D})^{-1} B\tilde{D}B^T)^{-1} \cdot (\bar{D}^{-1} - \frac{d_0 \bar{D}^{-1} cc^T \bar{D}^{-1}}{1 + d_0 c^T \bar{D}^{-1} c}) \\
&= \Delta^{-1}(\bar{D}^{-1} - \frac{d_0 \bar{D}^{-1} cc^T \bar{D}^{-1}}{1 + d_0 c^T \bar{D}^{-1} c}),
\end{aligned}$$

where

$$\Delta := I + (d_0 cc^T + \bar{D})^{-1} B\tilde{D}B^T.$$

Thus

$$\begin{aligned} x_c &= (A_F D A_F^T)^{-1} A_F D r_F = (A_F D A_F^T)^{-1}(d_0 r_0 c + B\tilde{D}\tilde{r}) \\ &= \frac{d_0 r_0}{1 + d_0 c^T \bar{D}^{-1} c} \Delta^{-1} \bar{D}^{-1} c + \Delta^{-1}(\bar{D}^{-1} B\tilde{D}\tilde{r} - \frac{d_0 \bar{D}^{-1} cc^T \bar{D}^{-1} B\tilde{D}\tilde{r}}{1 + d_0 c^T \bar{D}^{-1} c}) \\ &= \frac{d_0 r_0}{1 + d_0 c^T \bar{D}^{-1} c} \Delta^{-1} \bar{D}^{-1} c + \Delta^{-1} q \end{aligned} \tag{5}$$

where

$$q := \bar{D}^{-1} B\tilde{D}\tilde{r} - \frac{d_0 \bar{D}^{-1} cc^T \bar{D}^{-1} B\tilde{D}\tilde{r}}{1 + d_0 c^T \bar{D}^{-1} c}. \tag{6}$$

Since x^* is a limit point of $\{x_c(d(u_0))\}$ there is a sequence $\{u_0^k : k = 1, 2, \ldots\}$ such that $u_0^k \to \infty$ and $x_c(d(u_0^k)) \to x^*$ as $k \to \infty$. From (3) we have $(c^T x_c - l_0) \cdot (d(u_0^k))_0^2 u_0^k \to 1$. We claim that for any j with $c_j \neq 0$ we have $d_j \to \infty$, i.e., $j \in J$. Suppose that this claim is not true. Then we can find an index $p \notin J$ such that $c_p \neq 0$ and there is a subsequence of $\{u_0^k\}$ (for convenience we denote it by the current sequence and drop the superscript k) such that $d_p := d_p(u_0) \to d_p^* < \infty$. Since J is the active set $d_j \to \infty$, for $j \in J$. We can assume without loss of generality that $d_j \to d_j^*$ for all $j \notin J$ where $0 < d_j^* < \infty$, $j \notin J$. Moreover, $\tilde{D}^*$ is nonsingular since $|J| \leq n$. Using the Sherman-Morrison formula it is easy to show that, as $u_0 \longrightarrow \infty$,

$$\begin{aligned} \Delta &\to I + \mathrm{diag}(0, \ldots, 0, (d_{|J|+1}^*)^{-1}, \ldots, (d_n^*)^{-1}) \cdot B\tilde{D}^* B^T \\ &=: \begin{bmatrix} I_J & 0 \\ B_J & B_{nJ} \end{bmatrix} =: \Delta^* \end{aligned}$$

where I_J is the $|J| \times |J|$ identity matrix, B_J is some $(n - |J|) \times |J|$ matrix and B_{nJ} is a $(n - |J|) \times (n - |J|)$ nonsingular matrix. Accordingly, the inverse of Δ^* can be expressed as

$$(\Delta^*)^{-1} = \begin{bmatrix} I_J & 0 \\ E_J & E_{nJ} \end{bmatrix}, \tag{7}$$

where E_{nJ} is nonsingular. It follows from (6) and (7) that

$$\begin{aligned} \Delta^{-1} q &\longrightarrow \begin{bmatrix} I_J & 0 \\ E_J & E_{nJ} \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 \\ 0 & D_{nJ}^{-1} \end{bmatrix} B\tilde{D}\tilde{r} \\ &=: \begin{bmatrix} 0 \\ z \end{bmatrix}, \end{aligned} \tag{8}$$

where z is some vector. By (5), (7) and (8) we have (denoting $[c_J/d_J]$ the vector whose components are $c_j/d_j, j \in J$ and similarly defining the vector $[c_{nJ}/d_{nJ}]$)

$$\begin{aligned}
\lim_{u_0\to\infty} x_c &= \lim_{u_0\to\infty}(d_0^2 r_0 \frac{\Delta^{-1}\bar{D}^{-1}c}{d_0(1+d_0 c^T \bar{D}^{-1}c)} + \Delta^{-1}q) \\
&= \lim_{u_0\to\infty}(\frac{1}{2(c^Tx^*-l_0)d_0}\begin{bmatrix} I_J & 0 \\ E_J & E_{nJ}\end{bmatrix}\begin{bmatrix} c_J/d_J \\ c_{nJ}/d_{nJ}\end{bmatrix}) + \begin{bmatrix} 0 \\ z\end{bmatrix} \\
&= \lim_{u_0\to\infty}(\frac{1}{2(c^Tx^*-l_0)d_0}\begin{bmatrix} [c_J/d_J] \\ E_J[c_J/d_J] + E_{nJ}[c_{nJ}/d_{nJ}]\end{bmatrix}) + \begin{bmatrix} 0 \\ z\end{bmatrix}. \quad (9)
\end{aligned}$$

By our assumption $[c_J/d_J]/(d(u_0))_0$ goes to some vector. Noting that E_{nJ} is nonsingular it thus follows from (9) that x_c is unbounded since at least one component of $[c_{nJ}/d_{nJ}]$, c_p/d_p, is nonzero. This is a contradiction and the claim is thus proven. We now show that for all $j \in J$ if $c_j > 0$ then the j-th component of x_c, $(x_c)_j$ goes to u_j; if $c_j < 0$ then $(x_c)_j$ goes to $-u_j$. It follows from (9) that

$$\begin{aligned}
\lim_{u_0\to\infty} x_c &= \lim_{u_0\to\infty}(\frac{1}{2(c^Tx^*-l_0)d_0}\begin{bmatrix} I_J & 0 \\ E_J & E_{nJ}\end{bmatrix}\begin{bmatrix} D_J^{-1} & 0 \\ 0 & D_{nJ}^{-1}\end{bmatrix}\begin{bmatrix} c_J \\ 0\end{bmatrix}) + \begin{bmatrix} 0 \\ z\end{bmatrix} \\
&= \lim_{u_0\to\infty}(\frac{1}{2(c^Tx^*-l_0)d_0}\begin{bmatrix} D_J^{-1}c_J \\ E_J D_J^{-1}c_J\end{bmatrix}) + \begin{bmatrix} 0 \\ z\end{bmatrix}.
\end{aligned}$$

Therefore

$$x_J^* = \lim_{u_0\to\infty}(x_c)_J = \lim_{u_0\to\infty}\frac{1}{2(c^Tx^*-l_0)d_0}D_J^{-1}c_J.$$

Hence for all $j \in J$

$$(x_c)_j \longrightarrow \begin{cases} u_j & \text{if } c_j > 0 \\ -u_j & \text{if } c_j < 0. \end{cases}$$

We thus prove that x^* is an optimal solution to the subproblem

$$\begin{aligned} &\max c^T x \\ &-u \le Ix \le u. \end{aligned} \qquad \text{(LP3)}$$

The theorem thus follows from Lemma 2.2. □

Although this theorem is proven under the regularity assumption, our numerical tests indicate that $x_c(d(u_0)) \to x^*$ still holds even if the regularity assumption does not hold. We also note that the assumption $l_0 < \min\{c^Tx : x \in P\}$ can be satisfied by choosing

$$l_0 < \min\{c^Tx : x \in E\}$$

where

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

is an ellipsoid and $P \subset E$ [3].

3 A modified algorithm

In this section we propose a modified version of Algorithm 1.2. The idea is that we first set u_0 very large (pulling) so that, according to Theorem 2.3, the corresponding center $x_c(d(u_0))$ may be close to the optimal set and then we update the lower bound by letting $l_0 = c^T x_c(d(u_0))$ (pushing) to push the next center further towards the solution set. On the other hand, intuitively, the weighted center has a tendency to stay at the midpoint $(l_i + u_i)/2$, $j = 0, 1, \ldots, m$; we thus believe that the k-th feasible approximate center x_c would be good enough for the k-th iteration. We now describe our new algorithm.

Algorithm 3.1

- Initialization. Find an approximate center x_c of the system:

  $$l \leq A^T x \leq u, \tag{10}$$

  and let d be the corresponding weights. Let $l_0^0 := c^T x_c$ and

  $$u_0^0 := c^T x_c + (f(d^0)(c^T (ADA^T)^{-1} c)^{\frac{1}{2}} + q$$

  where q is some large positive number, $q = 10^8$ for example. Set $A \leftarrow (c, A)$, i.e., let c be the 0-th column, and

  $$l^0 := (l_0^0, l^T)^T, \; u^0 := (u_0^0, u^T)^T.$$

  Set $k = 0$.

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

  (1) Use Algorithm 1.1 to find the k-th feasible solution to

  $$l^k \leq A^T x \leq u^k, \qquad (FP)_k$$

  and denote this solution as $x_c^k := x_c(d^k)$.

(2) Check the convergence. If it converges, stop; otherwise go to step (3).

(3) Update the lower bound.

$$l_0^{k+1} := c^T x_c^k \text{ and } l^{k+1} := (l_0^{k+1}, l^T)^T.$$

Set $k \leftarrow k+1$, and repeat.

□

The following theorem shows that this algorithm, like Algorithm 1.2, also finds an ε-solution in finitely many iterations.

Theorem 3.1 *Suppose that* $\text{int}(P) \neq \emptyset$. *Then Algorithm 2.1 provides an ε-solution of (LP) in finitely many iterations.*

Proof. Suppose $\{x_c^k\}$ is a sequence of approximate centers generated by Algorithm 2.1 and suppose that for all $k = 0, 1, \ldots$, $|c^T x_c^k - z^*| > \varepsilon$. Without loss of generality, we assume that $l_0^k \longrightarrow l_0^*$ and there is a ball with some positive number as its radius contained in P^k for all k, where P^k is the feasible set of the linear system $(FP)_k$. Using a continuity argument one can show that Algorithm 2.1 is exactly Algorithm 1.2 when k is sufficiently large. The same argument as in the proof of Theorem 3.7 of Liao and Todd [3] thus applies and this theorem follows.
□

4 Implementation techniques and numerical results

In the previous section we sketch out a modified algorithm. Now we provide some details on the implementation of this algorithm. First of all, we suppose that the problem is scaled so that $\|a_i\|_2 = 1$ for all $i = 0, 1, \ldots, m$ to make the convergence test scale independent.

(1). The choice of q. The pulling technique is carried out by choosing a large q. Theoretically speaking, the bigger the q, the better. However, if q is too large the system $(FP)_k$ will become ill-conditioned. From our experience, $10^8 \leq q \leq 10^{11}$ would be fine.

(2). Convergence test. It is proven in Liao and Todd [3] that the duality gap of (LP) and its dual can be bounded as follows:

$$gap \leq (f(d)c^T(ADA^T)^{-1}c)^{\frac{1}{2}} =: tol(d)$$

where $x_c(d)$ is a feasible solution of (LP). Naturally, we can use the following condition as a stopping test

$$\frac{tol(d)}{1+|c^Tx_c|} < \epsilon,$$

where ϵ is a user predetermined positive number. Since $tol(d)$ is half the width of some ellipsoid containing the feasible set P^k along direction c, it is often that $tol(d)$ is too relaxed, i.e., $tol(d)$ is not small yet $x_c(d)$ is very close to the optimal solution. We thus add two extra convergence tests as options:

(a) At each iteration we find an approximate active set defined by n constraints corresponding to the first n smallest quantities among

$$\{\min(|a_i^Tx_c - l_i|, |a_i^Tx_c - u_i|) : i = 1, 2, \ldots, m\},$$

and solve the subproblem of (LP) induced by this approximate active set. This subproblem is easy to solve and if the solution is feasible to (LP) then it is also a solution of (LP). Of course, in practice, (a) may not determine a solution for some problems, but it is worthwhile using it as an option.

(b) Theorem 2.6 of Liao and Todd [3] shows that if $f(d) < 0$ for some $d > 0$, then the corresponding system is inconsistent. Therefore, whenever we suspect that the current iterate is close to the optimal solution, say $tol(d) < 0.2$, we take an extra run, i.e., we take $l_0 = c^Tx_c + 10^{-8}$ and run Algorithm 1.1 a small number of iterations, say 10, for the system (FP). If, during these iterations, some d is found such that $f(d) < 0$, then we get an approximate solution x_c to (LP) with $|c^Tx_c - z^*| < 10^{-8}$.

Our algorithm (with $q = 10^8$) was coded in MATLAB. The test problems we use are generated similarly to that of Avis and Chvatal [1] but with upper bounds on each component of x, namely,

$$\begin{aligned} &\max e^Tx \\ &l \leq \tilde{N}x \leq u \end{aligned} \tag{11}$$

Table 1: Computational results of Algorithm 3.1

Problem	Obj. Value	SIMPLEX		ALGORITHM 3.1	
		iter	CPU time	iter(minor iter)	CPU time
Test10	2.396239831E+01	17	9.8	2(27)	1.2
Test20	2.928953918E+01	55	12.1	3(35)	4.7
Test30	3.209169912E+01	88	15.6	3(28)	9.1
Test40	1.674106771E+01	168	30.4	3(28)	18.2
Test50	3.449813978E+01	275	53.9	3(22)	25.5

with

$$\tilde{N} = \begin{pmatrix} N \\ I \end{pmatrix}, \quad u = (10^4 e^T, 10 e^T)^T,$$

N is an $n \times n$ matrix with integer elements chosen randomly in the range $1, \ldots, 1000$. The five test problems: test10, test20, test30, test40 and test50 correspond to $n = 10, 20, 30, 40$, and 50 respectively. The lower bounds l are chosen so that none of them will be active at the solution. Therefore (11) is equivalent to

$$\begin{aligned} &\max e^T x \\ &\tilde{N} x \leq u, \end{aligned} \tag{12}$$

which, again, is the dual of problem

$$\begin{aligned} &\min u^T y \\ &\tilde{N}^T y = e \\ &y \geq 0. \end{aligned} \tag{13}$$

Then the simplex method is used to solve this problem. We believe this gives a "fair" comparison between the simplex method and Algorithm 3.1. The MATLAB version of the simplex method was obtained from netlib. (It was actually obtained by sending netlib@ornl.gov. the following message "send simplex1 from matlab/optimization".) All the displays are discarded from this MATLAB version of the simplex method. All the tests were performed on a Sun sparc station. The numerical results are presented in Table 1. It is clear that our algorithm does better than the simplex method on these problems, although our convergence test (a) happens to apply to these problems.

Table 2: Solving the first 5 test problems of netlib

Prob #	Prob Name	MINOS		ALGORITHM 3.1	
		iter	Obj. Value	iter(minor iter)	Obj. Value
1	afiro	9	-4.6475314E+02	6(48)	-4.6475314E+02
2	sc50a	31	-6.4575077E+01	4(42)	-6.4575077E+01
3	sc50b	31	-7.0000000E+01	3(30)	-7.0000000E+01
4	adlittle	115	2.2549496E+05	10(90)	2.2549496E+05
5	blend	107	-3.0812150E+01	4(74)	-3.0812150E+01

We also solve the first 5 problems of the test problems provided in netlib using Algorithm 3.1. These problems are first formed in the form

$$\begin{aligned} &\min c^T y \\ &Ay = b \\ &y \geq 0. \end{aligned} \tag{14}$$

The dual of (14) is

$$\begin{aligned} &\max b^T x \\ &A^T x \leq c. \end{aligned} \tag{15}$$

Then some dummy lower bounds $l_j, j = 1, 2, \ldots, m$ which are some large negative numbers are added so that problem (15) is equivalent to

$$\begin{aligned} &\max b^T x \\ &l \leq A^T x \leq c. \end{aligned} \tag{16}$$

Algorithm 3.1 is then applied to (16). Table 2 presents the results, where the results of MINOS are cited from Lustig, Marsten and Shanno [5].

We should note that in our algorithm the most computational effort is spent in solving the $m \times m$ linear system:

$$\nabla^2 F(d) d_{ir} = -\nabla F(d), \tag{17}$$

which defines the search direction, since the Hessian $\nabla^2 F(d)$ is usually a dense matrix. There are two possible ways to overcome this difficulty. First, we can use parallel computation technique to solve system (17); second, as shown in Liao and Todd [3],

$$\nabla^2 F(d) = 2 \cdot \mathrm{diag}(A^T x_c - r) A^T (ADA^T)^{-1} A \cdot \mathrm{diag}(A^T x_c - r) + 2D^{-3},$$

it is thus possible to use the conjugate gradient method with some appropriate preconditioner to solve (17). These are the subjects of current research.

5 Conclusions

The pulling technique is an efficient method for the weighted center algorithm introduced by Liao and Todd [3]. We prove that under the regularity assumption the weighted center is indeed pulled towards to the optimal solution set as u_0 – the upper bound corresponding to the constraint formed by the objective – goes to ∞. Thus, by choosing a large u_0 the corresponding center will be close to the optimal solution set. We also propose a modification of Algorithm 3.3 of [3]. This modified algorithm does not try to find the exact center, instead, the k-th feasible approximate center is determined. Our limited numerical tests show that this new approach is more efficient.

This pulling technique can also be used to find an initial vertex that is close to the optimal solution set and from this vertex (a basic solution) the standard simplex method is carried out. By doing so, if the vertices of the LP problem are evenly distributed, an improvement on efficience can be expected.

It is still an open question whether Theorem 2.3 still holds without the regularity assumption, though our numerical experiment suggests so.

Acknowledgment: The author would like to thank Mike Todd for a careful reading of this paper and his many valuable comments, corrections and suggestions.

References

[1] D. Avis and V. Chvatal. Notes on Bland's pivoting rule. *Mathematical Programming Study*, 8:24–34, 1978.

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

[3] A. Liao and M. J. Todd. Solving LP Problems via Weighted Centers. Technical Report CTC93TR145, Advanced Computing Research Institute, Cornell University, 1993.

[4] D. G. Luenberger. *Linear and Nonlinear Programming*. Addison-Wesley, Reading, Mass., 1984.

[5] I. Lustig, R. Marsten, and D. Shanno. Computational experience with a primal-dual interior point method for linear programming. *Linear Algebra and Its Applications*, 152:191–222, 1991.