SIAM J. OPTIMIZATION
Vol. 6, No. 3, pp. 600–616, August 1996

A NEW FINITE CONTINUATION ALGORITHM FOR LINEAR PROGRAMMING*

KAJ MADSEN[†], HANS BRUUN NIELSEN[‡], AND MUSTAFA Ç. PINAR[‡]

Abstract. We describe a new finite continuation algorithm for linear programming. The dual of the linear programming problem with unit lower and upper bounds is formulated as an ℓ_1 minimization problem augmented with the addition of a linear term. This nondifferentiable problem is approximated by a smooth problem. It is shown that the minimizers of the smooth problem define a family of piecewise-linear paths as a function of a smoothing parameter. Based on this property, a finite algorithm that traces these paths to arrive at an optimal solution of the linear program is developed. The smooth problems are solved by a Newton-type algorithm. Preliminary numerical results indicate that the new algorithm is promising.

Key words. finite algorithms, continuation methods, linear programming, ℓ_1 optimization, Huber function, Newton's method

AMS subject classifications. 90C05, 65K05

1. Introduction.

This paper introduces a new finite algorithm for linear programming (LP) by taking an unconstrained view of the problem. The algorithm operates on a special form of the linear program. We consider an LP problem where the variables are restricted to be between -1 and $+1$. We call this problem a "normalized" linear program. It is well known [3, 14] that the dual of this linear program can be formulated as an unconstrained "augmented" ℓ_1 minimization problem. The term "augmented" is used in this context to indicate that the function to be minimized is composed of an ℓ_1 term and a linear term. In this paper, the ℓ_1 norm is approximated by a smooth "Huber" function [10]. The approximation involves a positive smoothing parameter γ. We also refer to this smooth problem as the "Huber" problem. The following properties of the approximation are emphasized as the main contributions of our analysis in this paper.

(P0) The minimizers of the smooth problem define a family of piecewise-linear paths as a function of the smoothing parameter γ (Lemma 3).

(P1) The solution to the LP problem is detected for sufficiently small $\gamma > 0$ using these paths. That is, γ does not have to be decreased to zero in order to obtain an exact solution to the LP problem (Theorem 1).

(P2) The structure of the set of minimizers of the smooth problem closely "mimics" the structure of the solution set of the ℓ_1 and LP problems (Theorems 1 and 4).

(P3) The nature of the approximation allows a constructive duality link to the original primal problem, thereby giving the resulting algorithm a primal-dual flavor (Theorem 4).

These results generalize the results of an earlier study on the ℓ_1 problem by the first two authors [13].

These properties suggest an algorithm to trace these piecewise-linear paths to arrive at a solution of the linear program. We refer to these paths as "solution paths" throughout the paper. We note that the solution path could be unique as is the case

* Received by the editors November 16, 1993; accepted for publication (in revised form) December 28, 1994. This research was partially supported by Danish National Science Council grant 11-0505.

† Institute of Mathematical Modelling, Numerical Analysis Group, Technical University of Denmark, 2800 Lyngby, Denmark (km@imm.dtu.dk).

‡ Department of Industrial Engineering, Bilkent University, 06533 Bilkent, Ankara, Turkey.

with the concept of the central path in interior point methods. Our algorithm is best interpreted as a continuation algorithm since it possesses the following main features of continuation algorithms.

1. The solution of a parametrized family of subproblems as a parameter varies over an interval—in our case the smooth "Huber" problem as a function of the smoothing parameter γ.
2. The use of information from the solutions of the previous problems to aid in the solution of the next member of the family. Here, we utilize the piecewise-linear dependence of the solution paths on γ and the duality gap to compute an initial point for subsequent Huber problems. Using information from the previous solution makes the convergence of subsequent iterations faster.
3. A method for adjusting the continuation parameter for the efficient and accurate solution of subsequent problems. Piecewise linearity of the solution paths is again exploited in devising an efficient scheme for adjusting γ.
4. The use of a local iterative method to solve the subproblems. We use a finite Newton method [12] to solve the smooth Huber problem.

As a result of (P1), the continuation algorithm is a finite procedure provided that γ is decreased by at least a certain factor after each unconstrained minimization. We discuss these ideas more precisely in the forthcoming sections.

Our algorithm is related to many ideas in the optimization literature. In particular, there exists a close relationship between our method and penalty function algorithms since these methods can also be interpreted as continuation methods. Penalty methods usually treat all or a subset of the constraints of an optimization problem appending a penalty term to the objective function. The resulting penalty problem is solved by an iterative method, and solutions of the previous problems are used as starting points for the solution of subsequent problems. The process is controlled using a penalty parameter that is adjusted after the solution of each penalty problem if certain optimality criteria are not met. Barrier function and interior point algorithms can also be subsumed under this category, as well as more recent noninterior continuation algorithms [2]. In this regard our algorithm offers a novelty in the utilization of piecewise linearity and the absence of a penalty function.

The use of the ℓ_1 function in LP was proposed by Conn in [5] and later studied by Bartels in [1]. In these studies the ℓ_1 function is used as a penalty function. The resulting unconstrained piecewise-linear problem is solved using a combined projected descent and active set strategy.

Another related idea is due to Pinar and Zenios [19], where the Huber function is utilized as a linear-quadratic penalty function to penalize the side constraints in network models with side constraints and multicommodity network models. Linear-quadratic functions have also been the subject of study by Rockafellar [20] and Sun [21].

A more recent idea was proposed by Coleman and Li in [4, 3]. Their departure point is identical to ours in that they also work on the augmented ℓ_1 dual of a normalized linear program. They develop an affine scaling algorithm to solve the augmented ℓ_1 problem after a slight reformulation. Unfortunately, an implementation of their algorithm is not available for comparison.

To the best of our knowledge, our algorithm stands as a novel contribution to the linear optimization literature while it displays similarities with its chief competitors. In particular, the main computational burden is the solution of symmetric positive (semi)definite systems of linear equations reminiscent of interior point methods. On the other hand, two successive systems are closely related enough to warrant an update of the previous factorization, much like the simplex method. However, the iterates are neither required to stay in the interior nor on the surface of the polytope. Furthermore, the new algorithm stops when the duality gap is closed.

We develop a preliminary, but stable and efficient, implementation of the new algorithm for problems with full matrices. The algorithm is tested both on randomly generated problems and some problems from the Netlib collection. The results indicate a modest growth in the number of iterations as the problem size increases. We compare our results with a dense simplex subroutine from the NAG subroutine library based on the package LSSOL of Stanford Systems Optimization Library. The algorithm outperforms the dense simplex subroutine by a wide margin on the randomly generated problems. It is also competitive with the simplex subroutine on most of the Netlib problems tried. We also offer some comparisons with interior point methods and in particular with the CPLEX$^{\text{TM}}$ Barrier package which implements the primal-dual interior point algorithm [11]. Several issues that remain unresolved about the algorithm will be addressed in future studies, including the development of a sparse implementation.

The rest of the paper is organized as follows. After some preliminaries, we study the structure of the smooth Huber problem in §2. The connection between the ℓ_1 and LP problems and the smooth Huber problem is studied in §3. The results of these two sections form a constructive theory in that they suggest an algorithm. The algorithm is described in §4. Numerical results along with some implementation details are given in §5.

1.1. Preliminaries. We wish to solve the following problem:
[NormLP]

$$\begin{array}{ll} \underset{y}{\text{maximize}} & c^T y \\ \text{subject to} & \begin{array}{rcl} Ay & = & b, \\ -e \leq & y & \leq e, \end{array} \end{array}$$

where $c \in \Re^m$, $A \in \Re^{n \times m}$, $b \in \Re^n$, and $e = (1, \dots, 1)$. More general problems can be transformed into [NormLP] if the bounds on y can be replaced by general upper and lower bounds. Furthermore, as we see in §5, we will also be able to solve the general LP problem via [NormLP] using the idea of slack of variables and some artificial bounds. Therefore, we limit our study to [NormLP].

The new method is based on solving the dual problem to [NormLP]:
[AL1]

$$\text{minimize } G(x) \equiv \|A^T x - c\|_1 + b^T x. \tag{1}$$

A proof of the duality correspondence can be found, for instance, in [14]. If x^* and y^* are dual solutions then

$$a_i^T x^* - c_i < 0 \Rightarrow y_i^* = 1, \tag{2}$$
$$a_i^T x^* - c_i > 0 \Rightarrow y_i^* = -1, \tag{3}$$

where a_i is the ith column of A. Therefore, where the last equality is the duality equality and expresses the necessary condition for a solution of [AL1]:

$$-Ay^* + b = 0, \tag{5}$$

$$-e \le y^* \le e. \tag{6}$$

The method for solving [AL1] is similar to that in [13] for minimizing the linear ℓ_1 function. We estimate a minimizer of G by solving a sequence of approximating smooth problems, each of which depends on a parameter $\gamma > 0$. These problems are defined as follows. Let

$$r(x) = A^T x - c, \tag{7}$$

and for a given threshold $\gamma > 0$ define

$$\mathbf{s}_\gamma(x) = [s_{\gamma 1}(x), \ldots, s_{\gamma m}(x)] \tag{8}$$

with

$$s_{\gamma i}(x) = \begin{cases} -1 & \text{if } r_i(x) < -\gamma, \\ 0 & \text{if } |r_i(x)| \le \gamma, \\ 1 & \text{if } r_i(x) > \gamma. \end{cases} \tag{9}$$

$\mathbf{s}_\gamma(x)$ is a *sign vector*, and in general a sign vector is any vector $\mathbf{s} \in \Re^m$ with components $s_i \in \{-1, 1, 0\}$. For any sign vector $\mathbf{s}$ we define

$$W_{\mathbf{s}} = \mathrm{diag}(w_1, \ldots, w_m), \tag{10}$$

where

$$w_i = 1 - s_i^2. \tag{11}$$

If $\mathbf{s} = \mathbf{s}_\gamma(x)$ then we also denote $W_{\mathbf{s}}$ by $W_\gamma(x)$ or W_γ if no confusion is possible.

Now, the nondifferentiable problem [AL1] is approximated by the smooth "Huber problem" [10]:
[SAL1]

$$\text{minimize } G_\gamma(x) \equiv \frac{1}{2\gamma} r^T W_\gamma r + \mathbf{s}_\gamma^T \left[r - \frac{1}{2}\gamma \mathbf{s}_\gamma \right] + b^T x, \tag{12}$$

where the argument x is dropped for notational convenience. Clearly, G_γ measures the "small" residuals ($|r_i(x)| \le \gamma$) by their squares while the "large" residuals are measured by the ℓ_1 function. Thus, G_γ is a piecewise quadratic function, and it is continuously differentiable in $\Re^n$. We prove that when $\gamma \to 0_+$ any solution of [SAL1] is close to a solution of [AL1]. Furthermore, we show that dual solutions to [AL1] and [NormLP] can be detected directly when γ is below a certain (problem dependent) threshold $\gamma_0 > 0$. Therefore, the new algorithm consists of solving [SAL1] for a decreasing set of threshold values γ until dual solutions are detected. We summarize the algorithm as follows.

```
Compute initial $\gamma$
repeat
    compute a solution of [SAL1]
    decrease $\gamma$
until duality gap is closed.
```

Since solving [SAL1] $\gamma > 0$ is a finite procedure [12], the whole algorithm is finite provided γ is decreased at least by a certain factor in each iteration.

2. Properties of G_γ. In this section we describe some essential properties of G_γ.

We can assume without loss of generality that A has rank n and that every column a_i of A is nonzero. Otherwise, the problem could easily be reformulated to have these properties.

Clearly G_γ is composed of a finite number of quadratic functions. In each domain $D \subseteq \Re^n$, where $\mathbf{s}_\gamma(x)$ is constant and G_γ is equal to a specific quadratic function as seen from the above definition. These domains are separated by the following union of hyperplanes:

$$B_\gamma = \{x \in \Re^n | \exists\, i : |r_i(x)| = \gamma\}. \tag{13}$$

A sign vector $\mathbf{s}$ is γ-*feasible* at x if

$$\forall\, \varepsilon > 0 \; \exists\, z \in \Re^n \setminus B_\gamma : \|x - z\| < \varepsilon \;\wedge\; \mathbf{s} = \mathbf{s}_\gamma(z). \tag{14}$$

If $\mathbf{s}$ is a γ-feasible sign vector at some point x, then let $Q_\mathbf{s}^\gamma$ be the quadratic function which equals G_γ on the subset

$$\mathcal{C}_\mathbf{s}^\gamma = \mathrm{cl}\{z \in \Re^n | \mathbf{s}_\gamma(z) = \mathbf{s}\}. \tag{15}$$

$\mathcal{C}_\mathbf{s}^\gamma$ is called a Q-*subset* of $\Re^n$. Notice that any $x \in \Re^n \setminus B_\gamma$ has exactly one corresponding Q-subset ($\mathbf{s} = \mathbf{s}_\gamma(x)$), whereas a point $x \in B_\gamma$ belongs to two or more Q-subsets. Therefore, in general, we must give a sign vector $\mathbf{s}$ in addition to x in order to specify which quadratic function we are currently considering as representative of G_γ. However, the gradient of G_γ is independent of the choice of $\mathbf{s}$.

$Q_\mathbf{s}$ can be defined as follows:

$$Q_\mathbf{s}(z) = \frac{1}{2\gamma}(z - x)^T(AW_\mathbf{s}A^T)(z - x) + G_\gamma'^T(x)(z - x) + G_\gamma(x). \tag{16}$$

The gradient of the function G_γ is given by

$$G_\gamma'(x) = A\left[\frac{1}{\gamma}W_\mathbf{s}r + \mathbf{s}\right] + b, \tag{17}$$

where $\mathbf{s}$ is a γ-feasible sign vector at x. For $x \in \Re^n \setminus B_\gamma$, the Hessian of G_γ exists and is given by

$$G_\gamma''(x) = \frac{1}{\gamma}AW_\gamma A^T. \tag{18}$$

The set of indices corresponding to "small" residuals

$$\mathcal{A}_\gamma(z) = \{i | 1 \le i \le m \;\wedge\; s_{\gamma i}(z) = 0\} \tag{19}$$

is called the γ-*active* set at z and the subspace

$$\mathcal{V}_\gamma(z) = \mathrm{span}\{a_i | i \in \mathcal{A}_\gamma(z)\} \tag{20}$$

is called the γ-*active* subspace at z. The set of minimizers of G_γ is denoted by M_γ. In [12] it is shown that, for the case $b = \mathbf{0}$, there exists a minimizer $x_\gamma \in M_\gamma$ for which $\mathcal{V}_\gamma(x_\gamma) = \Re^n$. The idea of the proof is that if $\mathcal{V}_\gamma(z_\gamma) \ne \Re^n$ for a minimizer z_γ then there exists a connected set of minimizers of the form $z_\gamma + \alpha h$, where $h \perp \mathcal{V}_\gamma(z_\gamma)$.

By letting α increase (or decrease) from zero one can find another minimizer where the rank of the γ-active subspace is increased. Repeating this argument we finally find a minimizer where the rank of the corresponding γ-active subspace is maximal (i.e., the rank is equal to n). Clearly, this proof extends to the case $b \neq \mathbf{0}$.

Let $\mathbf{s}^\gamma(x)$ be the γ-feasible sign vector at x with components

$$s_i^\gamma(x) = \begin{cases} -1 & \text{if } r_i(x) \leq -\gamma, \\ 0 & \text{if } |r_i(x)| < \gamma, \\ 1 & \text{if } r_i(x) \geq \gamma. \end{cases} \tag{21}$$

Then we have the following.

LEMMA 1. *$\mathbf{s}^\gamma(x)$ is constant for $x \in M_\gamma$. Furthermore, $r_i(x)$ is constant for $x \in M_\gamma$ if $s_i^\gamma = 0$.*

Proof. Let $z \in M_\gamma$ and let $\mathbf{s} = \mathbf{s}^\gamma(z)$; i.e., $G_\gamma(x) = Q_\mathbf{s}(x)$ for $x \in \mathcal{C}_\mathbf{s}^\gamma$. If $x \in \mathcal{C}_\mathbf{s}^\gamma \cap M_\gamma$ then $Q_\mathbf{s}''(x)(x-z) = 0$. Therefore, if $|r_i(z)| < \gamma$ then $a_i^T(x-z) = 0$ (see (16)), and hence $r_i(x) = r_i(z)$. Thus r_i is constant in $\mathcal{C}_\mathbf{s}^\gamma \cap M_\gamma$. Using the fact that M_γ is connected and r_i is continuous, it is easily seen by repeating the argument above that r_i is constant in M_γ. Next suppose $r_i(z) \geq \gamma$. Then $r_i(x) \geq \gamma$ for all $x \in M_\gamma$ because existence of $x \in M_\gamma$ with $r_i(x) < \gamma$ is excluded by the convexity of M_γ, the continuity of r_i, and the first part of the lemma. Similarly, $r_i(z) \leq -\gamma \implies r_i(x) \leq -\gamma$ for $x \in M_\gamma$. This proves Lemma 1. □

Following the lemma it is meaningful to use the notation $\mathbf{s}^\gamma(M_\gamma) = \mathbf{s}^\gamma(x), x \in M_\gamma$ as the sign vector corresponding to the solution set. Lemma 1 has the following consequences which characterize the solution set M_γ.

COROLLARY 1. *M_γ is a convex set that is contained in one Q-subset: $\mathcal{C}_\mathbf{s}^\gamma$ where $\mathbf{s} = \mathbf{s}^\gamma(M_\gamma)$.*

Proof. The proof follows immediately from the linearity of the problem and Lemma 1. □

COROLLARY 2. *Let $z \in M_\gamma$, and $\mathbf{s} = \mathbf{s}^\gamma(M_\gamma)$. Let $\mathcal{N}_\mathbf{s}$ be the orthogonal complement of $\mathcal{V}_\mathbf{s} = span\{a_i | s_i = 0\}$. Then*

$$M_\gamma = (z + \mathcal{N}_\mathbf{s}) \cap \mathcal{C}_\mathbf{s}^\gamma.$$

Proof. It follows from (17) that $G_\gamma'(z+u) = 0$ if $u \in \mathcal{N}_\mathbf{s}$ and $z + u \in \mathcal{C}_\mathbf{s}^\gamma$. Thus

$$M_\gamma \supseteq (z + \mathcal{N}_\mathbf{s}) \cap \mathcal{C}_\mathbf{s}^\gamma.$$

If $x \in M_\gamma$ then $r_i(x) = r_i(z)$ for $s_i = 0$, and hence $x - z \in \mathcal{N}_\mathbf{s}$. Therefore, Corollary 1 implies

$$M_\gamma \subseteq (z + \mathcal{N}_\mathbf{s}) \cap \mathcal{C}_\mathbf{s}^\gamma,$$

which proves Corollary 2. □

Notice that Corollary 2 has the following implication. Suppose the minimizer z of G_γ belongs to B_γ. Then M_γ is the intersection of $\mathcal{C}_\mathbf{s}^\gamma$ with a space of dimension at least p where $p = n - \text{rank}\{a_i^T \| r_i(z)| < \gamma\}$. Therefore, one can normally expect multiple solutions if there exists a solution in B_γ.

3. The connection between G_γ, G, and [NormLP].

In this section we let the threshold γ be variable and we analyze how the solution set M_γ changes as γ decreases and approaches zero. Furthermore, we show how to identify a solution to [NormLP].

Assume $x_\gamma \in M_\gamma$, and let $\mathbf{s} = \mathbf{s}^\gamma(M_\gamma)$. Let $\mathcal{V}_\mathbf{s}$ and $\mathcal{N}_\mathbf{s}$ be defined as in Corollary 2 (i.e., if $x_\gamma \notin B_\gamma$ then $\mathcal{V}_\mathbf{s} = \mathcal{V}_\gamma(x_\gamma)$).

Since x_γ satisfies the necessary condition for a minimizer of G_γ, i.e.,

$$\mathbf{0} = AW_\mathbf{s}(A^T x_\gamma - c) + \gamma(A\mathbf{s} + b), \tag{22}$$

the following linear system is consistent:

$$(AW_\mathbf{s}A^T)d = A\mathbf{s} + b. \tag{23}$$

Inserting (23) into (22) we see that $x_0 \equiv x_\gamma + \gamma d$ is the least squares solution to the linear system

$$W_\mathbf{s}(A^T z - c) = \mathbf{0}. \tag{24}$$

LEMMA 2. *Assume that the linear system* (24) *is consistent. Then*

$$\frac{1}{\gamma}W_\mathbf{s} r(x_\gamma) = -W_\mathbf{s}A^T d, \tag{25}$$

where d is a solution to (23).

Proof. The proof follows by inserting the solution $(x_\gamma + \gamma d)$ into (24). $\square$

Now let d solve (23) and assume $\mathbf{s}^{\gamma-\varepsilon}(x_\gamma + \varepsilon d) = \mathbf{s}$; i.e., $x_\gamma + \varepsilon d \in \mathcal{C}_\mathbf{s}^{\gamma-\varepsilon}$ for some $\varepsilon > 0$. The linearity of the problem implies $x_\gamma + \delta d \in \mathcal{C}_\mathbf{s}^{\gamma-\delta}$ for $0 \le \delta \le \varepsilon$. Therefore (22) and (23) show that $(x_\gamma + \delta d)$ is a minimizer of $G_{\gamma-\delta}$. Thus we have the following consequence of Corollary 2.

LEMMA 3. *Let $x_\gamma \in M_\gamma$ and let $\mathbf{s} = \mathbf{s}^\gamma(M_\gamma)$. Let d solve* (23). *If $\mathbf{s}^{\gamma-\varepsilon}(x_\gamma + \varepsilon d) = \mathbf{s}$ for $\varepsilon > 0$ then $\mathbf{s}^{\gamma-\delta}(x_\gamma + \delta d) = \mathbf{s}$, and*

$$M_{\gamma-\delta} = (x_\gamma + \delta d + \mathcal{N}_\mathbf{s}) \cap \mathcal{C}_\mathbf{s}^{\gamma-\delta} \tag{26}$$

for $0 \le \delta \le \varepsilon$.

Lemma 3 states that the minimizers of G_γ form a family of piecewise-linear paths as a function of the parameter γ.

THEOREM 1. *There exists $\gamma_0 > 0$ such that $\mathbf{s}^\gamma(M_\gamma)$ is constant for $0 < \gamma \le \gamma_0$. Furthermore,*

$$M_{\gamma-\delta} = (x_\gamma + \delta d + \mathcal{N}_\mathbf{s}) \cap \mathcal{C}_\mathbf{s}^{\gamma-\delta} \textit{ for } 0 \le \delta < \gamma \le \gamma_0,$$

where $\mathbf{s} = s^\gamma(M_\gamma)$ and d solves (23).

Proof. Since there are only a finite number of different sign vectors the theorem is a consequence of Lemma 3. $\square$

COROLLARY 3. *Let $0 < \gamma \le \gamma_0$ and $\mathbf{s} = \mathbf{s}^\gamma(M_\gamma)$. Let $x_\gamma \in M_\gamma$. If $s_i = 0$ then*

$$r_i(x_\gamma) = -(a_i^T d)\gamma, \tag{27}$$

where d is any solution of (23); *i.e., $r_i(x_\gamma)/\gamma$ is constant.*

Proof. Let $s_i = 0$. Since Theorem 1 implies $|r_i(x_\gamma + \delta d)| < \gamma - \delta$ for $0 \le \delta < \gamma$ we obtain $r_i(x_\gamma + \gamma d) = 0$. Therefore (24) is consistent and the result is a consequence of Lemma 2. $\square$

If $x_\gamma \in M_\gamma$ then $y_\gamma = -(W_\mathbf{s} r(x_\gamma)/\gamma + \mathbf{s})$, where $\mathbf{s} = \mathbf{s}^\gamma(M_\gamma)$, is feasible in [NormLP] as it is seen from (22). y_γ and x_γ are called dual solutions, and we show that y_γ solves [NormLP] when γ is small enough.

In the following two theorems we show that one can often expect the gap between the dual objective function G and the primal objective $c^T y_\gamma$ to decrease from x_γ in the direction d. In fact, we prove that the primal objective always increases and that G decreases under a certain assumption. However, all our experience indicates that G always decreases in the direction d from x_γ, but we have not been able to prove it in general.

THEOREM 2. *Let $x_\gamma \in M_\gamma$ and $\mathbf{s} = \mathbf{s}^\gamma(M_\gamma)$. Let d solve* (23) *and let $x_0 = x_\gamma + \gamma d$. Let*

$$y_\delta = -(W_\mathbf{s} r(x_0 - \delta d)/\delta + \mathbf{s}), \quad \delta > 0,$$

and $H(\delta) = c^T y_\delta$. Then $H'(\gamma) \le 0$.

Proof. Since x_0 is the least squares solution of (24), the vector

$$\rho = W_\mathbf{s} A^T x_0 - W_\mathbf{s} c$$

is orthogonal to $W_\mathbf{s} A^T x_0$ and

$$\|W_\mathbf{s} A^T x_0\|_2^2 + \|\rho\|_2^2 = \|W_\mathbf{s} c\|_2^2.$$

Therefore,

$$\begin{aligned} c^T W_\mathbf{s} r(x_0) &= c^T W_\mathbf{s} W_\mathbf{s} A^T x_0 - c^T W_\mathbf{s} c \\ &= (W_\mathbf{s} A^T x_0 - \rho)^T (W_\mathbf{s} A^T x_0) - \|W_\mathbf{s} c\|_2^2 \\ &= -\|\rho\|_2^2 \le 0. \end{aligned} \tag{28}$$

Furthermore,

$$\begin{aligned} H(\gamma) &= -c^T (W_\mathbf{s}(A^T(x_0 - \gamma d) - c)/\gamma + \mathbf{s}) \\ &= -c^T W_\mathbf{s} r(x_0)/\gamma + c^T W_\mathbf{s} A^T d - c^T s. \end{aligned}$$

Therefore (28) implies

$$H'(\gamma) = c^T W_\mathbf{s} r(x_0)/\gamma^2 \le 0. \qquad \square$$

It follows from the proof that if (24) is consistent then $H'(\gamma) = 0$.

We now extend the definition of $\mathbf{s}_\gamma$ to the case $\gamma = 0$:

$$s_{0i}(x) = \begin{cases} -1 & \text{if } r_i(x) < 0, \\ 0 & \text{if } r_i(x) = 0, \\ 1 & \text{if } r_i(x) > 0. \end{cases} \tag{29}$$

Furthermore, let

$$C_\mathbf{s}^0 = \mathrm{cl}\{z \in \Re^n | \mathbf{s}_0(z) = \mathbf{s}\}. \tag{30}$$

THEOREM 3. *Let $x_\gamma \in M_\gamma$. If* (24) *is consistent then the directional derivative*

$$\lim_{\delta \downarrow 0}\{(G(x_\gamma + \delta d) - G(x_\gamma))/\delta\}$$

is nonpositive.

Proof. Let $\mathbf{s} = \mathbf{s}^{\gamma}(M_{\gamma})$, $\mathbf{s}_0 = \mathbf{s}_0(x_{\gamma})$, and $\mathcal{A} = \{i | s_i = 0\}$. First we notice that for $i \in \mathcal{A}$ the consistency of (24) implies that if $r_i(x_{\gamma}) = 0$ then $r_i(x_{\gamma} + \delta d) = 0$ for any δ. Therefore the following holds for δ being a sufficiently small positive number:

$$\begin{aligned} G(x_{\gamma} + \delta d) &= \mathbf{s}^T r(x_{\gamma} + \delta d) + \mathbf{s}_0^T W_{\mathbf{s}} r(x_{\gamma} + \delta d) + b^T(x_{\gamma} + \delta d) \\ &= G(x_{\gamma}) + \delta[\mathbf{s}^T A^T d + b^T d + \mathbf{s}_0^T W_{\mathbf{s}} A^T d]. \end{aligned}$$

Using (23) and (25) we obtain

$$\begin{aligned} [G(x_{\gamma} + \delta d) - G(x_{\gamma})]/\delta &= d^T A W_{\mathbf{s}} A^T d + \mathbf{s}_0^T W_{\mathbf{s}} A^T d \\ &= d^T A W_{\mathbf{s}}^T W_{\mathbf{s}} A^T d + \mathbf{s}_0^T W_{\mathbf{s}} A^T d \\ &= (W_{\mathbf{s}} r(x_{\gamma})/\gamma)^T (W_{\mathbf{s}} r(x_{\gamma})/\gamma) - \mathbf{s}_0^T W_{\mathbf{s}} r(x_{\gamma})/\gamma \\ &= \sum_{i \in \mathcal{A}} \left[\left(\frac{r_i(x_{\gamma})}{\gamma} \right)^2 - s_{0i} \frac{r_i(x_{\gamma})}{\gamma} \right]. \end{aligned}$$

The last sum is nonpositive because $|r_i(x_{\gamma})| < \gamma$ for $i \in \mathcal{A}$. This proves the theorem. □

THEOREM 4. *Let* $0 < \gamma \leq \gamma_0$, *where* γ_0 *is given in Theorem* 1, *and let* $\mathbf{s} = \mathbf{s}^{\gamma}(M_{\gamma})$. *Let* $x_{\gamma} \in M_{\gamma}$ *and* d *solve* (23). *Then any point in the set*

$$M_0 = (x_{\gamma} + \gamma d + \mathcal{N}_{\mathbf{s}}) \cap \mathcal{C}_{\mathbf{s}}^0 \tag{31}$$

solves [AL1], *and*

$$y_{\gamma} = -\left(\frac{1}{\gamma} W_{\mathbf{s}} r(x_{\gamma}) + \mathbf{s} \right) \tag{32}$$

solves [NormLP].

Proof. Assume $x_0 \in M_0$. Then there exists a solution d_0 to (23) such that $x_0 = x_{\gamma} + \gamma d_0$. Now the linearity and Theorem 1 imply that $x_{\gamma-\delta} = x_{\gamma} + \delta d_0 \in M_{\gamma-\delta}$ for $0 \leq \delta \leq \gamma$.

Since $\mathbf{s}^{\gamma}(x_{\gamma}) = \mathbf{s}^{\gamma-\delta}(x_{\gamma-\delta})$ for $0 \leq \delta < \gamma$, the continuity of r gives

$$r_i(x_0) \neq 0 \implies \text{sign}(r_i(x_0)) = \text{sign}(r_i(x_{\gamma-\delta})) = s_i = -y_{\gamma i} \tag{33}$$

for δ close enough to γ. Therefore,

$$\begin{aligned} G(x_0) &= -r(x_0)^T y_{\gamma} + b^T x_0 \\ &= -x_0^T A y_{\gamma} + c^T y_{\gamma} + x_0^T b \\ &= c^T y_{\gamma}. \end{aligned}$$

Furthermore, y_{γ} is feasible for [NormLP]. Hence, x_0 and y_{γ} are dual solutions to [AL1] and [NormLP]. Since this holds for any $x_0 \in M_0$ the theorem is proved. □

It is shown in [15] that M_0 coincides with the solution set of [AL1]. Notice that every $x_0 \in M_0$ corresponds to the same dual solution y_{γ}. This means that if M_0 contains more than one point then the Lagrange multipliers corresponding to y_{γ} are nonunique; i.e., y_{γ} is a degenerate solution to [NormLP]. On the other hand, if [AL1] has a degenerate solution (i.e., nonunique Lagrange multiplier) then [NormLP] has multiple solutions. In that case, Theorem 4 picks one specific solution y_{γ} to [NormLP].

4. The algorithm. The new algorithm is based on minimizing the function G_γ for a set of decreasing values of γ. It can be described as follows. Starting from a point x, we find a minimizer of G_γ for some $\gamma > 0$. For example, we locate a path among the possibly infinitely many paths defined by the minimizers of G_γ. Utilizing the piecewise-linear dependence of the paths on γ (Lemma 3) and the duality gap, we compute a guess at the minimizer of G_γ for a smaller value of γ. Starting from this guess, we compute the exact minimizer of G_γ using a Newton-type algorithm. Note that this gives the algorithm a predictor–corrector flavor in that we allow ourselves to deviate from the path only to return to it later from the predicted point using the Newton algorithm. Hence, we follow the solution paths closely without having to stay on any of them. Based on Theorem 4, this process terminates when the duality gap is closed.

To initiate the algorithm, a starting point x^0 and threshold value γ^0 are found as follows. Compute a least squares solution, i.e., a minimum x_{ls} of the quadratic function $f(x) \equiv r^T(x)r(x) + b^T x$, by solving

$$(AA^T)x_{ls} = Ac - \frac{1}{2}b. \tag{34}$$

Then set $x^0 = x_{ls}$ and γ^0 such that $|\mathcal{A}_{\gamma^0}(x^0)| \geq n$.

The algorithm has two main components: (1) the solution of the smooth problem, i.e., minimization of G_γ for a given value of γ, and (2) the reduction of γ and the computation of an initial point for the solution of the subsequent Huber problem. We now consider these two components in detail.

4.1. Computing a minimizer of G_γ. The algorithm for computing a minimizer x_γ of G_γ is based on a modified Newton algorithm given in [12]. A search direction h is computed by minimizing the quadratic $Q_{\mathbf{s}}$ where $\mathbf{s} = \mathbf{s}_\gamma(x)$ and x is the current iterate. More precisely, we consider the equation

$$Q''_{\mathbf{s}}h = -Q'_{\mathbf{s}}(x), \tag{35}$$

where $Q''_{\mathbf{s}}$ and $Q'_{\mathbf{s}}$ denote the gradient and Hessian of $Q_{\mathbf{s}}$, respectively. From (16), (17), and (18) we obtain

$$(AW_{\mathbf{s}}A^T)h = -AW_{\mathbf{s}}r - \gamma(A\mathbf{s} + b). \tag{36}$$

For ease of notation let $C \equiv AW_{\mathbf{s}}A^T$ and $g \equiv -AW_{\mathbf{s}}r - \gamma(A\mathbf{s} + b)$. Furthermore, let $\mathcal{N}(C)$ denote the null space of C. If C has full rank, then h is the solution of (36). Otherwise, if the system of equations (36) is consistent, h is taken to be the minimum norm solution. If the system is inconsistent, h is taken to be the projection of g on $\mathcal{N}(C)$. The motivation for these choices is discussed in [12]. The next iterate is found by a line search aiming for a zero of the directional derivative. This procedure is computationally cheap as a result of the piecewise linear nature of G'_γ. It is shown in [12] that the iteration is finite; i.e., after a finite number of iterations we have $x + h \in \mathcal{C}^\gamma_{\mathbf{s}(x)}$. Therefore, $x + h$ is a minimizer of G_γ as a result of (15), (16), and the convexity of G_γ. We summarize below the modified Newton algorithm.

```
repeat
    $\mathbf{s} = \mathbf{s}_\gamma(x)$
    if (36) consistent then
        find $h$ from (36)
```

```
            if $x + h \in C_{\mathbf{s}}^{\gamma}$ then
                $x \leftarrow x + h$
                stop = true
            else
                $x \leftarrow x + \alpha h$ (line search)
            endif
        else
            compute $h =$ null space projection of $g$
            $x \leftarrow x + \alpha h$ (line search)
        endif
    until stop.
```

4.2. Checking optimality and reducing γ. Let x_γ be a minimizer of G_γ computed using the modified Newton algorithm of the previous section. Then either the continuation algorithm terminates or the modified Newton algorithm is restarted using a reduced value of γ. Let $\mathbf{s} = \mathbf{s}^\gamma(M_\gamma)$ and d be a solution to (23). From the necessary condition (22) it follows that (23) is equivalent to

$$(AW_{\mathbf{s}}A^T)d = -\frac{1}{\gamma}AW_{\mathbf{s}}r(x_\gamma). \tag{37}$$

Let d solve (37), and let $r_\gamma \equiv r(x_\gamma)$. For $0 \leq \delta \leq \gamma$ we define

$$x(\gamma - \delta) = x_\gamma + \delta d. \tag{38}$$

Then

$$r(x(\gamma - \delta)) = r_\gamma + \delta A^T d. \tag{39}$$

Notice that, following Lemma 3, $x(\gamma-\delta)$ is a minimizer of $G_{\gamma-\delta}$ as long as $\mathbf{s}_{\gamma-\delta}(x(\gamma-\delta)) = \mathbf{s}$.

The stopping test is based on Theorem 4. It consists of checking the duality gap $(G(x(0)) - c^T y_\gamma)$, where y_γ is given in (32). If this is zero (within the roundoff tolerance) then the algorithm is stopped provided the residuals satisfy the following sign accordance:

$$s_{\gamma i} = 0 \implies r_i(x(0)) = 0,$$
$$s_{\gamma i} \neq 0 \implies s_{\gamma i} r_i(x(0)) \geq 0.$$

Otherwise, we let $y(\gamma - \delta) = -(\frac{1}{\gamma-\delta}W_{\mathbf{s}}r(x(\gamma-\delta)) + \mathbf{s})$, and then we attempt to find the value δ^* of δ for which $|G(x(\gamma-\delta)) - c^T y(\gamma-\delta)|$ attains its minimum. This is basically done by examining the points $\delta = \frac{k}{10}\gamma$, $k = 1, \ldots, 10$, and choosing the best of these in the above sense provided that $\mathbf{s}_{\gamma-\delta^*}(x(\gamma-\delta^*)) \neq \mathbf{s}$. Then $x(\gamma-\delta^*)$ given by (38), with $r(x(\gamma-\delta^*))$ given by (39), starts the Newton iteration using the new threshold value $\gamma - \delta^*$. The precise description of this procedure is given below.

```
$\Delta_0 = G(x_\gamma) - c^T y(\gamma)$
$k \leftarrow 0$
while not STOP do
    $k \leftarrow k + 1$
    $\delta \leftarrow k * \gamma/10$
```

```
        $x \leftarrow x_\gamma + \delta d$
        $r \leftarrow r_\gamma + \delta A^T d$
        $y \leftarrow y(\gamma - \delta)$
        $\Delta_k = G(x) - c^T y$
    end while
        if $k \leq 2$ then $\delta \leftarrow 0.1 * \delta$
        else $\delta \leftarrow (k-1) * \gamma/10$
        endif
    $x \leftarrow x_\gamma + \delta d$
    $r \leftarrow r_\gamma + \delta A^T d$
    $\gamma \leftarrow \gamma - \delta$
```

STOP is a function that returns true when one of the following conditions hold:

$$\Delta_k > \Delta_{k-1}, \quad \Delta_k < -0.1\Delta_0,$$

provided that $\mathbf{s}_{\gamma-\delta}(x(\gamma - \delta)) = \mathbf{s}$ or when

$$k = 9.$$

The first condition ensures that the procedure stops close to a minimum. The second condition guards against the case where the sign of the difference in the function values is reversed.

We have also experimented with alternative procedures for reducing γ. We used the total iteration count as the performance measure. The experiments showed that there is no significant difference among the alternative procedures in terms of performance. As a result, we adopted the above procedure since it has a sound justification as a result of Theorem 4. We note that this procedure guarantees that unless the duality gap is closed, γ is decreased by at least a factor of 0.9 after each unconstrained minimization. Hence, we have the following theorem.

THEOREM 5. *The continuation algorithm described in* §§4.1 *and* 4.2 *stops at an optimal solution* y^* *of* [NormLP] *after a finite number of iterations.*

Proof. As a result of the observation, γ is reduced by a certain factor after each unconstrained minimization phase unless optimality is reached. Hence, using Theorem 1, γ can only be decreased a finite number of times. Since the modified Newton algorithm is finite [12], the result follows. □

5. Numerical results. In this section we report our numerical experience with a Fortran 77 implementation of the new algorithm, which does not exploit sparsity. We use two sets of test problems:

1. randomly generated full problems,
2. sparse problems from the Netlib collection [7].

We have three goals when we perform numerical experiments. The first goal is to examine the growth in solution time and iteration count of the new algorithm as the problem size is increased. The second goal is to test the numerical accuracy of the algorithm. We use randomly generated test problems to achieve these ends. The third goal is to test the viability of the algorithm in solving problems of practical interest. To accomplish this we choose a set of small, sparse test problems from the Netlib collection. Since these problems are very different in nature from the randomly generated problems this should give us insight into the potential of the new algorithm as a competitor to the well-known algorithms for LP. To this end, we compare our results with an LP subroutine E04MBF from the NAG subroutine library. E04MBF is based on LSSOL from Stanford Systems Optimization Library. It is a Fortran 77 package for constrained linear least squares problems, LP, and convex quadratic programming [9]. It does not exploit sparsity. Hence, it provides a fair comparison with our numerical results.

The major effort in the Newton algorithm of §4.1 is spent in solving the systems (36). We use the AAFAC package of [17] to perform this. The solution is obtained via an LDL^T factorization of the matrix $C_k = AW_\gamma(x_k)A^T$, so D and L are computed directly from the active columns of A, i.e., without squaring the condition number as would be the case if C_k was first computed. It is essential for the efficiency of the continuation algorithm that normally the γ-active set $\mathcal{A}_\gamma(x_k)$ differs relatively little from $\mathcal{A}_\gamma(x_{k-1})$. This implies that the factorization of C_k can be obtained by relatively few up- and downdates of the factorization of C_{k-1}. Therefore, the computational cost of a typical iteration step is $O(n^2)$. Occasionally, a refactorization is performed. This consists of the successive updating $LDL^T \leftarrow LDL^T + a_j a_j^T$ for all j in the active set (starting with $L = I, D = 0$). It is considered only when some columns of A leave the active set, i.e., when downdating is involved. If many columns leave we may refactorize because it is cheaper. Otherwise, a refactorization is used when a downdating results in a rank decrease and there is an indication that rounding errors have marked influence. This part of the algorithm combines ideas from [6, 8]. For details see §2 in [17]. The refactorization is an $O(n^3)$ process.

The solution of (23) is done via (37), and the matrix $AW_\mathbf{s}A^T$ is identical to the last C_k used in the Newton iteration. Thus, the LDL^T factorization is available. In order to enhance accuracy we use one step of iterative refinement when solving (37). For further details see [18]. We refer to the subroutine that implements the continuation algorithm as LPASL1. With the exception of some internal tolerance parameters (e.g., tolerances used for numerical checks for zero), LPASL1 does not allow any control over the execution of the algorithm. Hence, all the results reported in this study were obtained under identical algorithmic choices. We note that LPASL1 is available for distribution as a Fortran 77 subroutine. Details can be found in [16].

First we report computational results using randomly generated problems. Our random problems are generated using ideas similar to the generator detailed in §4.1 of [18]. All the problems are of the form [NormLP]. Half of the matrix A in each problem was taken as the identity matrix to simulate the presence of slack columns. We solve ten problems of each size. The tests were performed on a DEC 3000 Alpha APX Model 500 running DEC OSF1 V1.2. The Fortran code was compiled using the `-O` option of the DEC Fortran compiler. The ratio $\mu = m/n$ is kept constant at 2. In all cases, the duality gap in LPASL1 was below 10^{-8}. In Table 1, we report the average over ten problems of the following LPASL1 statistics: number of iterations, run time in CPU seconds, number of refactorizations, and number of γ-reductions. The column "iter" refers to the total number of iterations, i.e., the total number of solutions to (36) and (37), and "reduc" to the number of γ-reductions. The column "refac" refers to the total number of refactorizations in connection with the computations of the factors L and D. In all cases, the cardinality of the set $\mathcal{A}_0(x_0)$ and the rank of the associated matrix $AW_\mathbf{s}A^T$ on termination are equal to n. The two columns under the heading LSSOL contain the run time statistics of LSSOL. The rightmost column displays the ratio run time of LSSOL/run time of LPASL1. In all cases, the optimal objective function values reported by LPASL1 and LSSOL agree to at least eight digits. All runs with LPASL1 and LSSOL were performed using default parameters; i.e., no fine

TABLE 1
Solution statistics of LPASL1 and LSSOL using randomly generated problems (NA: not solved due to memory shortage).

m, n	LPASL1				LSSOL		ratio
	iter	refac	reduc	CPU	iter	CPU	
(200,100)	56.2	4	16.4	0.6	352.2	4.7	7.83
(320,160)	74.1	4.4	22.2	2.6	713.5	21.8	8.38
(480,240)	104.7	6	30.2	9.9	1300.7	91.3	9.22
(720,360)	117.2	4.8	32.4	30.5	2312	360.4	11.81
(1080,540)	165.2	5.4	44.3	116.5	4305.6	1470.9	12.62
(1620,810)	217.4	8.5	51.7	469.4	8628.6	6501.7	13.85
(2430,1215)	290.7	13.1	62.9	2648.7	16205.1	27777	10.48
(3644,1822)	335	13.2	62.6	8327.3	NA	NA	NA

tuning of the codes was done for any test problem.

It was observed that over a set of ten problems of a given size, the performance statistics of LPASL1 display a larger variation around the mean than those of LSSOL. For instance, the minimum run time for problems of size (2430,1215) was 1400 seconds while the maximum run time was 3569 seconds. For LSSOL, the two extreme values were 26441 seconds and 29113 seconds. However, we observe that LPASL1 performs increasingly better than LSSOL as the problem size is increased.

A least squares fit with the model $\beta \cdot n^{\alpha}$ shows that

$$\text{iter}_{LPASL1} \approx 3.0 \cdot n^{0.63}, \quad \text{iter}_{LSSOL} \approx 0.3 \cdot n^{1.53},$$

$$\text{CPU}_{LPASL1} \approx 1.2 \cdot 10^{-7} \cdot n^{3.31}, \quad \text{CPU}_{LSSOL} \approx 4.7 \cdot 10^{-7} \cdot n^{3.48}.$$

We observe that on both measures LSSOL exhibits a higher growth rate than LPASL1.

Finally we consider ten problems from the Netlib collection. We also used a test problem from a civil engineering application, referred to as PLATE. The characteristics of the problems are given in Table 2.

TABLE 2
Characteristics of the sparse test problems.

Problem name	Constraints	Variables	Slack variables
afiro	27	32	19
sc50b	50	48	30
sc50a	50	48	30
sc105	105	103	60
adlittle	56	97	41
scagr7	129	140	45
stocforl	117	111	54
blend	74	83	31
sc205	205	203	114
share2b	96	79	83
plate	61	73	60

All the problems we consider are of the form

$$\begin{array}{ll} \underset{y}{\text{maximize}} & c^T y \\ \text{subject to} & Ay \le b, \\ & y \ge 0. \end{array}$$

In order to apply the continuation algorithm to these problems, we prescribe a uniform bound u to all the variables. Hence, we artificially transform the problem to the following equivalent problem:

$$\begin{array}{ll} \underset{y}{\text{maximize}} & c^T y \\ \text{subject to} & Ay = b, \\ & 0 \leq y \leq U, \end{array}$$

where $U = (u, \ldots, u)$. Then the problem can be transformed to the form [NormLP] by a simple linear transformation. No preprocessing was used prior to solving the problems. In Table 3, we show the solution statistics of LPASL1 and LSSOL. The tests were performed on an IBM PS/2 486. The Salford F77 compiler was used. The columns "iter," "reduc," and "refac" are as defined in Table 1. The columns "ν_0"and "rank" refer to the cardinality of the set $\mathcal{A}_0(x_0)$ and the rank of the associated matrix AW_sA^T on termination, respectively. Finally, we report the CPU time in seconds exclusive of input/output under "CPU." In all cases, the optimal objective function values reported by LPASL1 and LSSOL agree to at least eight digits. Both LPASL1 and LSSOL were run using default parameters.

TABLE 3
Solution statistics of LPASL1 and LSSOL on sparse problems.

Problem name	LPASL1						LSSOL	
	iter	refac	reduc	ν_0	rank	CPU	iter	CPU
afiro	21	4	4	28	27	0.49	19	0.60
sc50b	22	2	1	48	48	0.93	47	3.07
sc50a	32	3	3	47	47	1.64	40	2.47
sc105	56	8	1	101	101	9.78	78	14.28
adlittle	89	18	15	62	56	8.13	149	14.56
scagr7	76	7	10	129	129	19.06	126	58.07
stocforl	101	13	3	111	111	19.01	73	15.93
blend	60	8	3	73	72	7.25	85	12.96
sc205	152	16	3	196	196	107.63	192	122.74
share2b	87	8	5	91	86	11.86	122	13.07
plate	20	3	4	55	43	2.47	150	9.23

We observe that LPASL1 is faster than LSSOL on ten of the eleven problems. Furthermore, in the remaining problem, stocforl, the performance figures of the two codes are very close. We expect to report results with all the Netlib problems in the future when we have an implementation that exploits sparsity.

Comparison with interior point and related methods. Our algorithm displays some similarities to interior point methods for LP and to the affine scaling methods of Coleman and Li in the main computational step. A common feature of all these algorithms is that their main computational step is the solution of a weighted least squares system of the form

$$(ADA^T)p = f, \tag{40}$$

where D is a diagonal positive definite matrix, and p and f are vectors of appropriate dimensions. Since the numerical values of the entries of D change from one iteration to the next, a refactorization of the matrix ADA^T is performed at each iteration of these methods. Since we do only occasional refactorizations of the matrix AWA^T

in our method, the cost of our average iteration, which consists of a series of up- and downdates of the factors, is lower compared with that of the above-mentioned methods. Hence, the relative standing of our method with respect to these methods can only be established by a thorough computational comparison. To give the reader a feel for how the algorithms compare, we ran some of our randomly generated test problems using the CPLEX Barrier optimizer, which is a state-of-the-art implementation of a primal-dual interior point algorithm. The results are summarized in Table 4 below. These runs were made on a SUN 490 workstation.

TABLE 4
Solution statistics of LPASL1 and CPLEX on randomly generated problems.

(m, n)	LPASL1				CPLEX barrier	
	iter	refac	reduc	CPU	iter	CPU
200,100	43	8	5	19.95	13	16.95
320,160	51	6	8	65.83	14	67.84
400,200	58	4	12	104.90	15	140.39
600,300	68	5	13	356.73	15	451.79

We notice that LPASL1 statistics in Table 4 differ from the results displayed in Table 1 for similar size problems. The explanation of this is the following. To carry out the tests reported in Table 4 we generated random problems where all columns were fully dense in contrast to our problems in Table 1, where half of the matrix A was taken as the identity matrix to simulate slack columns. The reason for this change is to force CPLEX Barrier to use dense linear algebra. Interestingly, we observed that LPASL1 consistently took fewer iterations and reductions on problems of the type used in Table 4 compared with its performance reported in Table 1. Computing times in Tables 1 and 4 also differ due to the difference in the computing platforms used for the two experiments.

We also observe in Table 4 that the iteration number in the interior point method grows very slowly. The time performances of LPASL1 and CPLEX Barrier are very close on these test problems with a slight advantage of LPASL1 as the problems get larger. Given the preliminary nature of our numerical work on the new algorithm, these results are encouraging.

We expect similar behavior from Coleman and Li's algorithm since it is reported in their paper that the iteration count of their algorithm is almost constant as a function of the problem size. We are not able to do any comparisons since they only have a MATLAB$^{\text{TM}}$ code.

6. Summary. In this paper, we described a new finite continuation algorithm for LP based on linear ℓ_1 optimization. The algorithm solves a sequence of smooth nonlinear problems that yield a solution to the LP in a finite number of iterations. We studied the structure of the solution set of the approximating problems and showed how primal and dual solutions can be obtained. We also developed an efficient and stable implementation of the algorithm for full matrices and tested it on both randomly generated problems and some problems from the Netlib collection. Although these results are of a preliminary nature, they indicate that the algorithm exhibits a promising performance on both kinds of problems. We compared our results with a simplex LP routine from the Stanford Systems Optimization Library that is available as a subroutine in the NAG library, and with the CPLEX Barrier Optimizer. The comparison showed that the new algorithm is substantially faster than the NAG routine on randomly generated problems with full matrices. It is also competitive on the

Netlib problems tried. The comparison with CPLEX also gave encouraging results. The following issues remain for further studies:

- "hot" starts,
- preprocessing and automatic choice of an upper bound for problems with only nonnegativity restrictions on the variables,
- exploiting sparsity.

REFERENCES

[1] R. H. BARTELS (1980), *A penalty linear programming method using reduced-gradient basis-exchange techniques*, Linear Algebra Appl., 29, pp. 17–32.

[2] B. CHEN AND P.T. HARKER (1993), *A non-interior continuation algorithm for linear and quadratic programming*, SIAM J. Optim., 3, pp. 503–515.

[3] T. COLEMAN AND Y. LI (1990), *A globally and quadratically convergent affine scaling algorithm for the linear programming problem with bounded variables*, in Large Scale Numerical Optimization, T. Coleman and Y. Li, eds., Society for Industrial and Applied Mathematics, Philadelphia, PA, pp. 49–57.

[4] ——— (1992), *A globally and quadratically convergent affine scaling algorithm for ℓ_1 problems*, Math. Programming, 56, pp. 189–222.

[5] A. R. CONN (1976), *Linear programming via a nondifferentiable penalty function*, SIAM J. Numer. Anal., 13, pp. 145–154.

[6] R. FLETCHER AND M. J. D. POWELL (1974), *On the modification of LDL^T factorizations*, Math. Comp., 28, pp. 1067–1087.

[7] D. M. GAY (1985), *Electronic mail distribution of linear programming test problems*, Math. Prog. Soc. COAL Newsletter, 13, pp. 10–12.

[8] W. M. GENTLEMAN (1973), *Least squares computation by Givens transformations without square roots*, J. Inst. Math. Appl., 12, pp. 329–336.

[9] P. E. GILL, S. HAMMARLING, W. MURRAY, M. A. SAUNDERS, AND M. H. WRIGHT (1986), *User's Guide for LSSOL (Version 1.0): A Fortran Package for Constrained Linear Least Squares and Convex Quadratic Programming*, Report SOL 86-1, Department of Operations Research, Stanford University, Stanford, CA.

[10] P. HUBER (1981), *Robust Statistics*, John Wiley, New York.

[11] I. LUSTIG, R. MARSTEN, AND D. SHANNO (1991), *Computational experience with a primal-dual interior point method for linear programming*, Linear Algebra Appl., 152, pp. 191–222.

[12] K. MADSEN AND H.B. NIELSEN (1990), *Finite algorithms for robust linear regression*, BIT, 30, pp. 682–699.

[13] ——— (1993), *A finite smoothing algorithm for linear ℓ_1 estimation*, SIAM J. Optim., 3, pp. 223–235.

[14] K. MADSEN, H. B. NIELSEN, AND M. C. PINAR (1992), *Linear, Quadratic and Minimax Programming Using ℓ_1 Optimization*, Report NI-92-11, Institute of Mathematical Modelling, Numerical Analysis Group, Technical University of Denmark, Lyngby, Denmark.

[15] ——— (1994), *New characterizations of ℓ_1 solutions to overdetermined linear systems*, Oper. Res. Lett., 16, 3, pp. 159–166.

[16] ——— (1994), *User's Guide to* LPASL1: *A Fortran 77 Subroutine, Based on a Continuation Algorithm, for Dense Linear Programming*, Report IMM-94-01, Institute of Mathematical Modelling, Numerical Analysis Group, Technical University of Denmark, Lyngby, Denmark.

[17] H. B. NIELSEN (1990), *AAFAC: A Package of Fortran 77 Subprograms for Solving $A^TAx = c$*, Report NI-90-11, Institute of Mathematical Modelling, Numerical Analysis Group, Technical University of Denmark, Lyngby, Denmark.

[18] ——— (1991), *Implementation of a Finite Algorithm for Linear ℓ_1 Estimation*, Report NI-91-01, Institute of Mathematical Modelling, Numerical Analysis Group, Technical University of Denmark, Lyngby, Denmark.

[19] M. Ç. PINAR AND S. A. ZENIOS (1994), *On smoothing exact penalty functions for convex constrained optimization*, SIAM J. Optim., 4, pp. 486–511.

[20] T. ROCKAFELLAR (1990), *Computational schemes for large-scale problems in extended linear-quadratic programming*, Math. Programming, 48, pp. 447–474.

[21] J. SUN (1992), *On the structure of convex piecewise quadratic functions*, J. Optim. Theory Appl., 72, pp. 499–510.