

SANDIA REPORT

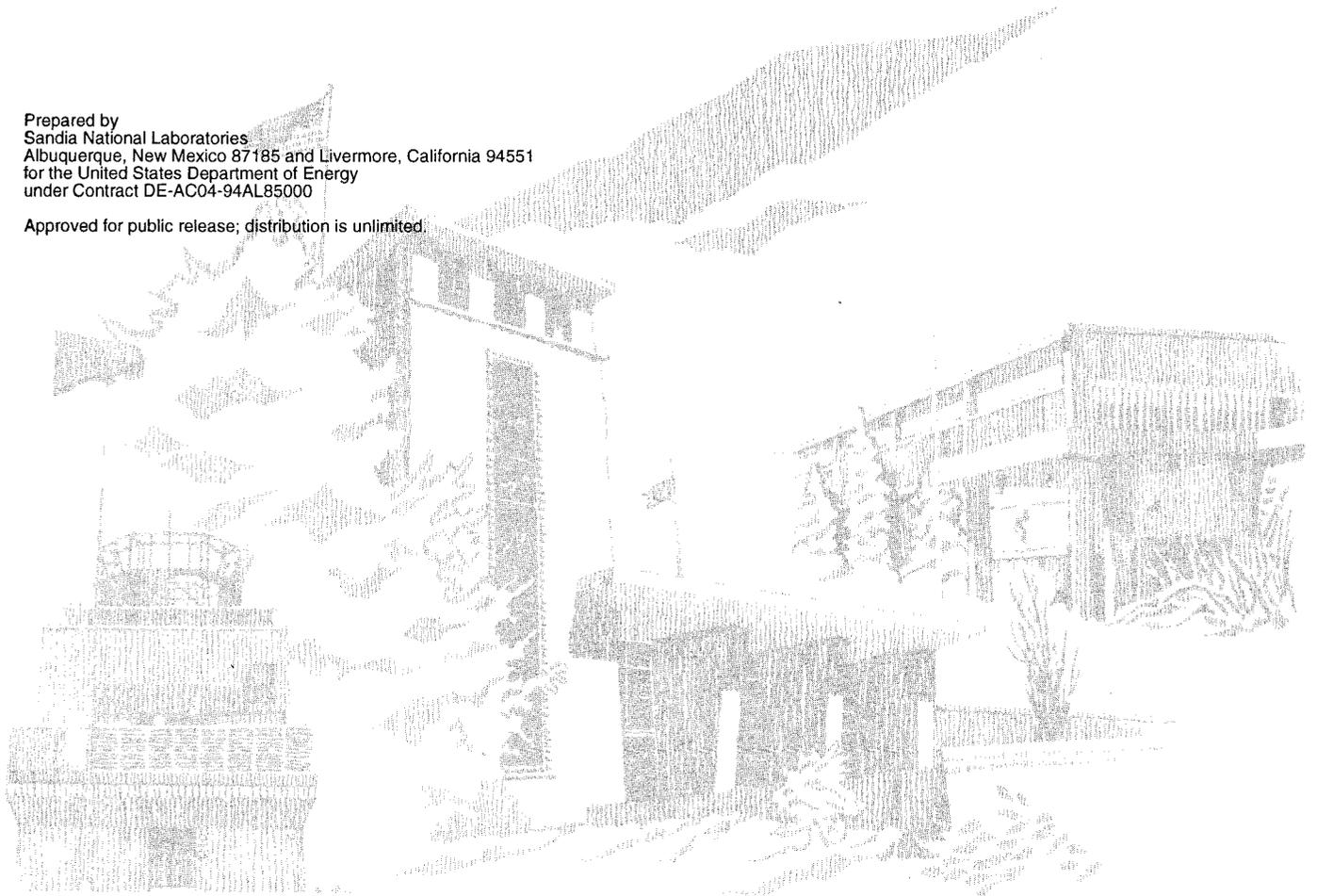
SAND97-8243 • UC-405
Unlimited Release
Printed March 1997

Stable Computation of Search Directions for Near-Degenerate Linear Programming Problems

Patricia D. Hough

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94551
for the United States Department of Energy
under Contract DE-AC04-94AL85000

Approved for public release; distribution is unlimited.



Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

NOTICE: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, nor any of the contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof, or any of their contractors or subcontractors.

SAND97-8243
Unlimited Release
Printed March 1997

Stable Computation of Search Directions for Near-Degenerate Linear Programming Problems *

Patricia D. Hough †
Scientific Computing Department
Sandia National Laboratories
Livermore, CA 94551
pdhough@ca.sandia.gov

Abstract

In this paper, we examine stability issues that arise when computing the search directions $(\Delta \mathbf{x}, \Delta \mathbf{y}, \Delta \mathbf{s})$ for a primal-dual path-following interior point method for linear programming. The dual step $\Delta \mathbf{y}$ can be obtained by solving a weighted least-squares problem for which the weight matrix becomes extremely ill conditioned near the boundary of the feasible region. Hough and Vavasis proposed using a type of complete orthogonal decomposition (the COD algorithm) to solve such a problem and presented stability results. The work presented here addresses the stable computation of the primal step $\Delta \mathbf{x}$ and the change in the dual slacks $\Delta \mathbf{s}$. These directions can be obtained in a straightforward manner, but near-degeneracy in the linear programming instance introduces ill-conditioning which can cause numerical problems in this approach. Therefore, we propose a new method of computing $\Delta \mathbf{x}$ and $\Delta \mathbf{s}$. More specifically, this paper describes an orthogonal projection algorithm that extends the COD method. Unlike other algorithms, this method is stable for interior point methods without assuming nondegeneracy in the linear programming instance. Thus, it is more general than other algorithms in this respect. Numerical tests indicate that it is more reliable than standard algorithms on near-degenerate problems.

*This work was partially supported by ONR grant N00014-96-1-0050 and NSF grant DMS-9505155.

†Part of this work was done while the author was a member of the Center for Applied Mathematics at Cornell University.

Contents

1	Introduction	5
2	Computing Δy	6
3	Computing Δs and Δx	8
4	Numerical Results	13
4.1	The Algorithm of Mizuno, Todd, and Ye	14
4.2	LIPSOL	17
5	Related Work	19
6	Conclusions	20
7	Acknowledgements	21

1 Introduction

We consider solving the following primal-dual pair of linear programming problems, given in standard form:

$$\begin{aligned} & \text{minimize} && \mathbf{c}^T \mathbf{x} \\ & \text{subject to} && A\mathbf{x} = \mathbf{b}, \\ & && \mathbf{x} \geq \mathbf{0}, \end{aligned} \tag{1}$$

and

$$\begin{aligned} & \text{maximize} && \mathbf{b}^T \mathbf{y} \\ & \text{subject to} && A^T \mathbf{y} + \mathbf{s} = \mathbf{c}, \\ & && \mathbf{s} \geq \mathbf{0}. \end{aligned} \tag{2}$$

In this setting, $A \in \mathbb{R}^{m \times n}$ is known and has rank m , $\mathbf{b} \in \mathbb{R}^m$ and $\mathbf{c} \in \mathbb{R}^n$ are given, and $\mathbf{x}, \mathbf{s} \in \mathbb{R}^n$ and $\mathbf{y} \in \mathbb{R}^m$ are unknown. While many algorithms for solving these problems appear in the literature, the focus of this paper is primal-dual path-following interior point algorithms. In particular, we consider the linear algebra problems that arise in the computation of the search directions. For an overview of interior point methods, see Gonzaga [11], M. Wright [28], or S. Wright [30].

We start with a brief summary of the path-following approach, beginning with the optimality conditions which are given by:

$$\begin{aligned} Ax &= \mathbf{b}, \\ A^T \mathbf{y} + \mathbf{s} &= \mathbf{c}, \\ SX\mathbf{e} &= \mathbf{0} \\ \mathbf{x} \geq \mathbf{0} & \quad , \quad \mathbf{s} \geq \mathbf{0}, \end{aligned}$$

where $S = \text{diag}(\mathbf{s})$, $X = \text{diag}(\mathbf{x})$, and \mathbf{e} is the vector of all ones. Introducing a parameter $\mu > 0$ into the third equation perturbs the optimality conditions to the following:

$$\begin{aligned} Ax &= \mathbf{b}, \\ A^T \mathbf{y} + \mathbf{s} &= \mathbf{c}, \\ SX\mathbf{e} &= \mu \mathbf{e}, \\ \mathbf{x} > \mathbf{0} & \quad , \quad \mathbf{s} > \mathbf{0}. \end{aligned} \tag{3}$$

If both (1) and (2) have strictly interior points, then this system of equations has a unique solution for every $\mu > 0$. The set of all solutions as μ ranges from 0 to ∞ is the *central path*, which was first defined by Bayer and Lagarias [1], [2], [3], Megiddo [15], and Sonnevend [21]. Notice that $\mu = \infty$ corresponds to the analytic center of the feasible region, and $\mu = 0$ corresponds to the optimal solution of the linear programming problems. Thus, in order to obtain the optimal solution via the central path, the system of equations (3) is solved repeatedly with a sequence of values for μ that tend to zero.

More specifically, given $(\mathbf{x}, \mathbf{y}, \mathbf{s})$ that approximates a point on the central path, it is necessary to solve (3) with a smaller value of μ for the next approximate central path point $(\mathbf{x} + \Delta\mathbf{x}, \mathbf{y} + \Delta\mathbf{y}, \mathbf{s} + \Delta\mathbf{s})$; i.e., solve

$$\begin{aligned} A(\mathbf{x} + \Delta\mathbf{x}) &= \mathbf{b}, \\ A^T(\mathbf{y} + \Delta\mathbf{y}) + (\mathbf{s} + \Delta\mathbf{s}) &= \mathbf{c}, \\ (S + \Delta S)(X + \Delta X)\mathbf{e} &= \bar{\mu}\mathbf{e}, \\ \mathbf{x} + \Delta\mathbf{x} > \mathbf{0} \quad , \quad \mathbf{s} + \Delta\mathbf{s} > \mathbf{0}, \end{aligned}$$

where $\bar{\mu} < \mu$. Solving for the unknowns $(\Delta\mathbf{x}, \Delta\mathbf{y}, \Delta\mathbf{s})$, applying (3), and linearizing the third condition yield the Newton equations:

$$A\Delta\mathbf{x} = \mathbf{0}, \tag{4}$$

$$A^T\Delta\mathbf{y} + \Delta\mathbf{s} = \mathbf{0}, \tag{5}$$

$$X\Delta\mathbf{s} + S\Delta\mathbf{x} = \bar{\mu}\mathbf{e} - XSe. \tag{6}$$

Notice that the search directions $(\Delta\mathbf{x}, \Delta\mathbf{y}, \Delta\mathbf{s})$ are uniquely determined by this system of equations. The remainder of this paper focuses on the stable computation of the solution to this system of equations.

The following section contains the derivation of the weighted least-squares problem that must be solved in order to obtain $\Delta\mathbf{y}$. This turns out to be a problem of the type addressed by Hough and Vavasis, [14], and thus, the results of that paper apply. A discussion of numerical problems that may arise when computing of $\Delta\mathbf{x}$ and $\Delta\mathbf{s}$ appears in §3. Rather than using the straightforward approach to obtain these directions, we propose using orthogonal projections to compute scaled versions of these directions. This method is an extension of the COD algorithm described in Hough and Vavasis. Numerical experiments in which the extended COD algorithm was incorporated into two different interior point methods were done to test its performance in practice. The results of these experiments appear in §4. An overview of related work and some conclusions make up the remainder of the paper.

2 Computing $\Delta\mathbf{y}$

In the primal-dual setting, it is necessary to compute both a primal step and a dual step at each iteration. As described in the previous section, these steps are related by and uniquely determined by the Newton equations. We first focus on obtaining $\Delta\mathbf{y}$, which can be accomplished by solving a weighted least-squares problem. The derivation of that problem follows. First, the other variables must be eliminated from the system of equations. Solving (6) for $\Delta\mathbf{s}$ and substituting into (5) yield

$$A^T\Delta\mathbf{y} = -\bar{\mu}X^{-1}\mathbf{e} + \mathbf{s} + X^{-1}S\Delta\mathbf{x}.$$

Multiplying both sides of the previous equation by $AS^{-1}X$ and applying (4) give

$$AS^{-1}XA^T\Delta\mathbf{y} = AS^{-1}X(\mathbf{s} - \bar{\mu}X^{-1}\mathbf{e}).$$

Since $S^{-1}X$ is a diagonal matrix, we write

$$ADA^T \Delta \mathbf{y} = AD(\mathbf{s} - \bar{\mu}X^{-1}\mathbf{e}), \quad (7)$$

where $D = S^{-1}X$. Notice that this is the system of normal equations for the weighted least-squares problem given by

$$\min_{\Delta \mathbf{y} \in \mathbb{R}^m} \left\| D^{1/2}(A^T \Delta \mathbf{y} - \mathbf{v}) \right\|, \quad (8)$$

where $\mathbf{v} = \mathbf{s} - \bar{\mu}X^{-1}\mathbf{e}$. Notice also that since $(\mathbf{x}, \mathbf{s}) > \mathbf{0}$, the matrix D is positive definite. A final point to note is that some entries of \mathbf{s} and some entries of \mathbf{x} approach zero near the boundary of the feasible region, so D becomes extremely ill conditioned. Since problems of this type often arise in applications, a great deal of work, both theoretical and algorithmic, has been done with regard to solving them. Of particular interest are some norm bounds that prove to be quite useful in the upcoming analysis. The results were first derived by Dikin in 1974 [6] and have been rediscovered independently by a number of authors. The most recent of these are Stewart in 1989 [22] and Todd in 1990 [23]. For a survey of this and related work, see Forsgren [7]. The norm bounds in question are given in the following theorem.

Theorem 1 *Let \mathcal{D} denote the set of all positive definite $n \times n$ real diagonal matrices. Let A be an $m \times n$ real matrix of rank m . If we define χ_A and $\bar{\chi}_A$ as follows:*

$$\begin{aligned} a) \chi_A &= \sup\{\|(ADA^T)^{-1}AD\| : D \in \mathcal{D}\}, \text{ and} \\ b) \bar{\chi}_A &= \sup\{\|A^T(ADA^T)^{-1}AD\| : D \in \mathcal{D}\}, \end{aligned}$$

then both χ_A and $\bar{\chi}_A$ are finite.

The norm in the previous theorem can be any matrix norm induced by a vector norm, but in this paper, $\|\cdot\| = \|\cdot\|_2$. Similarly, $\kappa(M) = \kappa_2(M)$, where $\kappa(M)$ is the condition number of the matrix M .

The finiteness of χ_A and $\bar{\chi}_A$ suggests that it is possible to compute a solution to a system of equations of the form (7) or (8) which satisfies an error bound that is independent of D . While the construction of an actual algorithm is not apparent, this observation does serve as motivation for the following specialized definition of stability, proposed by Vavasis [25].

Definition 1 *An algorithm for solving a problem of the form of (7) or (8) is stable if, in the presence of finite-precision arithmetic, an error bound of the form*

$$\|\Delta \mathbf{y} - \Delta \hat{\mathbf{y}}\| \leq \epsilon \cdot f(A) \cdot \|\mathbf{v}\| \quad (9)$$

is satisfied, where $\Delta \mathbf{y}$ is the true solution, $\Delta \hat{\mathbf{y}}$ is the computed solution, $f(A)$ is some function of A not depending on D , and $\epsilon > 0$ is machine roundoff.

Note that the right-hand side of (9) is independent of D .

There are many algorithms for solving problems like (7) and (8) that appear in the literature, but stability analyses which prove a bound like (9) do

not currently exist for most of these algorithms. Two exceptions are the NSH method proposed by Vavasis [24] and the COD algorithm proposed by Hough and Vavasis [14]. The COD algorithm has a couple of advantages over the NSH method. Unlike the NSH method, the COD algorithm is based on standard techniques. Additionally, the COD algorithm requires less work because the NSH method requires solving a larger system of equations. Thus, we will use the COD algorithm to solve the weighted least-squares problem (8). For completeness, an outline of the algorithm follows.

Algorithm: Complete Orthogonal Decomposition (COD)

Step 1: QR factor, with column pivoting, $AD^{1/2}$ to get

$$AD^{1/2} = QRP, \tag{10}$$

where Q is an $m \times m$ orthogonal matrix, R is an $m \times n$ upper triangular (“trapezoidal”) matrix, and P is an $n \times n$ permutation matrix.

Step 2: Apply reduced QR factorization (without pivoting) to R^T to get

$$R^T = Z_1U_1, \tag{11}$$

where Z_1 is an $n \times m$ matrix with orthonormal columns, and U_1 is an $m \times m$ upper triangular matrix.

Step 3: Solve the following system, via back substitution, for $\bar{\mathbf{y}}$:

$$U_1\bar{\mathbf{y}} = Z_1^T P D^{1/2} \mathbf{v}. \tag{12}$$

Step 4: To get \mathbf{y} , multiply the result of Step 3 by Q :

$$\Delta \mathbf{y} = Q\bar{\mathbf{y}}. \tag{13}$$

The paper by Hough and Vavasis contains a detailed discussion of the stability of the algorithm and other numerical issues that arise. Therefore, we forgo any further discussion of the computation of $\Delta \mathbf{y}$. Instead, we shift the focus to determining $\Delta \mathbf{x}$ and $\Delta \mathbf{s}$. The following section contains a discussion of numerical issues that arise and a description of how the COD algorithm can be extended to stably compute these directions.

3 Computing $\Delta \mathbf{s}$ and $\Delta \mathbf{x}$

The dual step $\Delta \mathbf{y}$ constitutes only part of the information that is needed at each iteration. It is also necessary to determine the primal step $\Delta \mathbf{x}$ and the change in the dual slack variables $\Delta \mathbf{s}$. Once $\Delta \mathbf{y}$ has been obtained, it is apparent from the central path equations that computing $\Delta \mathbf{x}$ and $\Delta \mathbf{s}$ is straightforward. Solving (5) and (6) yields the following:

$$\Delta \mathbf{s} = A^T \Delta \mathbf{y}, \text{ and} \tag{14}$$

$$\Delta \mathbf{x} = \bar{\mu} S^{-1} \mathbf{e} - S^{-1} X \Delta \mathbf{s}. \tag{15}$$

However, this approach is not numerically stable. The forward error bound (9) is not sufficiently strong to get a suitable accuracy bound on the entries $\Delta\mathbf{x}$ and $\Delta\mathbf{s}$, because some components are very small as convergence is achieved [29], or more generally, as the boundary of the feasible region is approached. This means that there is a demand for more accuracy in some components of $A^T\Delta\mathbf{y}$ than what could be obtained from (9). To clarify this, let us consider the following two-variable example, given in dual form:

$$\begin{aligned} & \text{maximize} && 100y_1 - y_2 \\ & \text{subject to} && l \leq y_1, y_2 \leq u \end{aligned}$$

Since there is significantly more emphasis on y_1 , an interior point method will first maximize with respect to y_1 , then with respect to y_2 . To illustrate, the feasible region and the central path for this problem are shown in Figure 1. Notice that near the boundary (e.g. at point P on the central path), the distance s_1 to the boundary is small. As a result, the magnitude of Δs_1 will also be small (since it is limited by the size of s_1). So $\Delta\mathbf{s}$ must be computed in such a way that there is a small relative error in Δs_1 , i.e., the error in Δs_1 should be small relative to s_1 . However, the step from point P has a very small change in the y_1 direction and a large change in the y_2 direction. Obtaining $\Delta\mathbf{s}$ via (14) implies that the contributions from Δy_1 and Δy_2 are treated as “equal”. This means that any error in Δy_2 will be introduced into Δs_1 , which is small relative to Δy_2 . While the normwise error in $\Delta\mathbf{s}$ may be acceptable, the relative error in Δs_1 may not be since Δy_2 is large compared to s_1 . This large relative error is then passed on to the entries of $\Delta\mathbf{x}$ in (15). Thus, it is necessary to find a way to compute these directions with more accuracy than can be guaranteed by using (14) and (15).

In order to obtain the desired accuracy, we propose first computing scaled versions of $\Delta\mathbf{x}$ and $\Delta\mathbf{s}$ and then unscaling to obtain the desired directions. This approach makes use of an orthogonal projection formed from orthogonal factors in the COD. The scaled $\Delta\mathbf{s}$ can be obtained as follows. The normal equations (7) imply

$$\Delta\mathbf{y} = (ADA^T)^{-1}AD(\mathbf{s} - \bar{\mu}X^{-1}\mathbf{e}).$$

Substituting this into (14) and multiplying both sides by $D^{1/2}$ give

$$D^{1/2}\Delta\mathbf{s} = -D^{1/2}A^T\Delta\mathbf{y} = D^{1/2}A^T(ADA^T)^{-1}AD(\bar{\mu}X^{-1}\mathbf{e} - \mathbf{s}).$$

Notice that on the right-hand side of this equation, D is embedded in a matrix that must be inverted. This cannot be done accurately when D is ill conditioned, but it is not difficult to see that this matrix need not be inverted. Working through the steps of the COD algorithm shows that $D^{1/2}A^T = P^T Z_1 U_1 Q^T$. Substituting for $D^{1/2}A^T$ in the previous equation yields the following:

$$D^{1/2}\Delta\mathbf{s} = PZ_1 Z_1^T P^T D^{1/2}(\bar{\mu}X^{-1}\mathbf{e} - \mathbf{s}). \quad (16)$$

Thus, matrix multiplication and subtraction are the only operations required. Since the goal is to obtain a forward error bound on the entries of $D^{1/2}\Delta\mathbf{s}$,

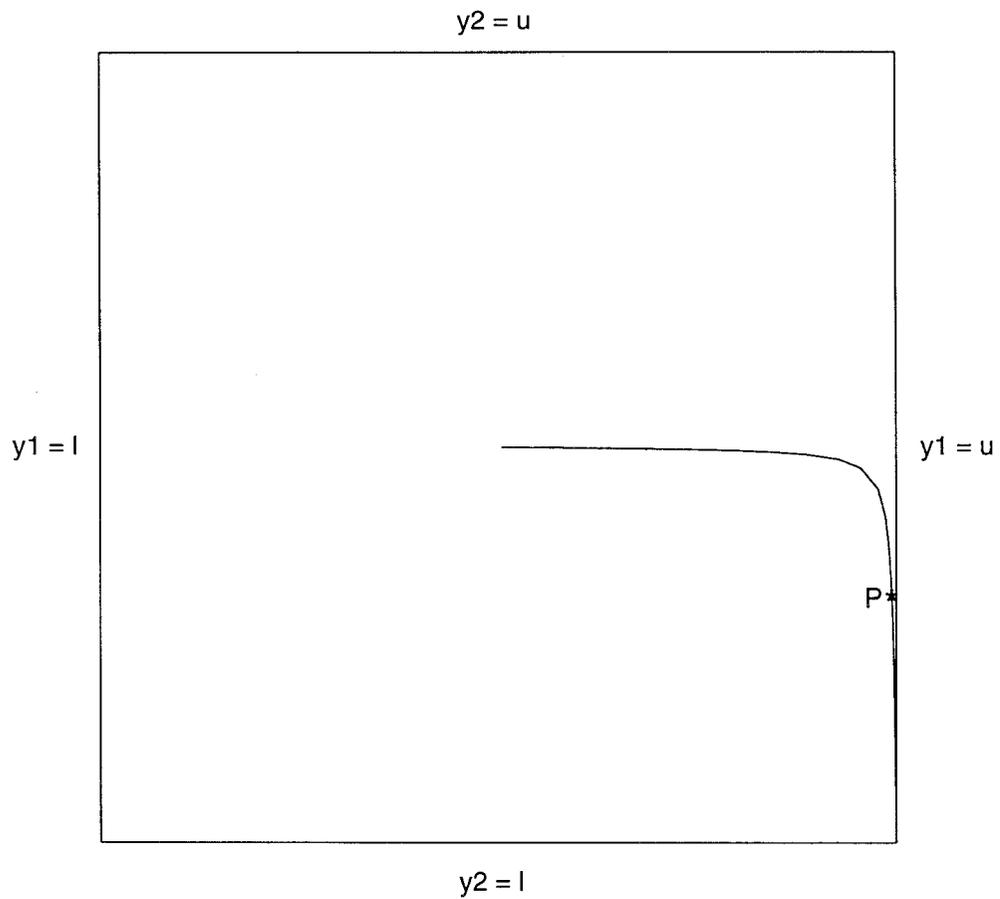


Figure 1: The feasible region and the central path for the given example appear above. The central path approaches the y_1 boundary first and then proceeds to the optimal solution. Notice that the magnitudes of the changes in the two directions can differ by a significant amount near the boundary.

we must have forward error bounds for the factors on the right-hand side. We first show that the error in Z_1 is bounded. (Recall that we are trying to obtain error bounds that are independent of D .) Note that this is asserted in Hough and Vavasis but not proved [14]. The theorem and proof appear below.

Theorem 2 *Let Z_1 be as in step 3 of the COD algorithm, and let \hat{Z}_1 be the corresponding computed matrix. Then*

$$\|Z_1 - \hat{Z}_1\| \leq \epsilon \cdot k \cdot f(A) + O(\epsilon^2), \quad (17)$$

where k is a small constant, ϵ is machine roundoff, and $f(A)$ is a function of A not depending on D .

Proof: Recall from Step 3 of the COD algorithm that $R^T = Z_1 U_1$. In order to use the results of Hough and Vavasis, we consider instead $R^T D^{-1/2} = Z_1 U_0$, where $U_0 = U_1 D^{-1/2}$. Let \hat{Z}_1 and \hat{U}_0 be the corresponding computed matrices, and assume that the signs of the diagonal entries of \hat{U}_0 are the same as those of the corresponding entries of U_0 . Standard backward error analysis (see [13]) implies that there exists a matrix \tilde{Z}_1 with exactly orthonormal columns such that

$$\|\tilde{Z}_1 - \hat{Z}_1\| \leq c\epsilon \quad (18)$$

and such that $\tilde{Z}_1 \hat{U}_0 = Z_1 U_0 + E$, where

$$\|E\| \leq c\epsilon \|R^T D^{-1/2}\| \quad (19)$$

and c is a small constant. We proceed as follows. Multiplying both sides on the right by U_0^{-1} gives

$$\tilde{Z}_1 \hat{U}_0 U_0^{-1} = Z_1 + E U_0^{-1}. \quad (20)$$

We now multiply this equation by its transpose to obtain

$$(\tilde{Z}_1 \hat{U}_0 U_0^{-1})^T (\tilde{Z}_1 \hat{U}_0 U_0^{-1}) = (Z_1 + E U_0^{-1})^T (Z_1 + E U_0^{-1}),$$

or

$$(\hat{U}_0 U_0^{-1})^T (\hat{U}_0 U_0^{-1}) = I + Z_1^T E U_0^{-1} + (Z_1^T E U_0^{-1})^T + O(\epsilon^2).$$

Notice that this is the Cholesky factorization of some matrix.

Now let $E_1 = Z_1^T E U_0^{-1}$. Notice that $E_1 + E_1^T$ can be written as

$$E_1 + E_1^T = L + W + L^T,$$

where L is strictly lower triangular and W is diagonal. It is not difficult to see that

$$(I + W/2 + L^T)^T (I + W/2 + L^T) = I + E_1 + E_1^T + O(\epsilon^2).$$

Also, $I + W/2 + L^T$ is upper triangular, so by uniqueness of the Cholesky factorization

$$\hat{U}_0 U_0^{-1} = I + W/2 + L^T + O(\epsilon^2).$$

Substituting this into (20) yields

$$\tilde{Z}_1(I + W/2 + L^T + O(\epsilon^2)) = Z_1 + EU_0^{-1}$$

and thus

$$Z_1 - \tilde{Z}_1 = \tilde{Z}_1(W/2 + L^T) + EU_0^{-1} + O(\epsilon^2). \quad (21)$$

Since $W/2 + L^T$ is part of $E_1 + E_1^T$, then

$$\|W/2 + L^T\| \leq 2\|E_1\| \leq 2 \cdot \|E\| \cdot \|U_0^{-1}\|. \quad (22)$$

So

$$\begin{aligned} \|Z_1 - \hat{Z}_1\| &\leq \|Z_1 - \tilde{Z}_1\| + \|\tilde{Z}_1 - \hat{Z}_1\| \\ &\leq 3 \cdot \|E\| \cdot \|U_0^{-1}\| + O(\epsilon^2) + c\epsilon \\ &\leq 3 \cdot c \cdot \epsilon \cdot \|R^T D^{-1/2}\| \cdot \|U_0^{-1}\| + c \cdot \epsilon + O(\epsilon^2). \end{aligned}$$

The second inequality comes from (21), (22), and (18). The third line comes from (19). Recall from Hough and Vavasis that $\|R^T D^{-1/2}\| \leq f_1(A)$ and that $\|U_0^{-1}\| = \|D^{1/2}U_1^{-1}\| \leq f_2(A)$, where $f_1(A)$ and $f_2(A)$ do not depend on D . Substituting these bounds into the above inequality and combining terms yield the final result:

$$\|Z_1 - \hat{Z}_1\| \leq \epsilon \cdot k \cdot f(A) + O(\epsilon^2),$$

where k incorporates the constants and $f(A)$ incorporates the functions of A in the previous sequence of inequalities. \square

We now return to the computation of $D^{1/2}\Delta\mathbf{s}$. Recall (16):

$$D^{1/2}\Delta\mathbf{s} = PZ_1Z_1^T P^T D^{1/2}(\bar{\mu}X^{-1}\mathbf{e} - \mathbf{s}).$$

Theorem 2 can be used to show that the forward error in the orthogonal projection $PZ_1Z_1^T P^T$ is bounded by a function that is independent of the weight matrix D . In order to say something helpful about the factor $D^{1/2}(\bar{\mu}X^{-1}\mathbf{e} - \mathbf{s})$, we need the relationships $D^{1/2} = S^{-1/2}X^{1/2}$ and $XSe \approx \mu\mathbf{e}$. These two facts imply that $D^{1/2}(\bar{\mu}X^{-1}\mathbf{e} - \mathbf{s}) \approx \mu^{1/2}\mathbf{e}$. Since forming the right-hand side involves scaling and subtraction (to obtain $\mu^{1/2}\mathbf{e}$) and then multiplication by the orthogonal projection, it is not difficult to show that

$$\|D^{1/2}\Delta\mathbf{s} - D^{1/2}\Delta\hat{\mathbf{s}}\| \leq \epsilon \cdot \mu^{1/2} \cdot c \cdot g(A),$$

where $D^{1/2}\Delta\mathbf{s}$ is the exact result, $D^{1/2}\Delta\hat{\mathbf{s}}$ is the computed result, $\epsilon > 0$ is machine precision, c is a constant, and $g(A)$ is a function of A that does not depend on D . An entrywise error bound can now be obtained as follows. Notice that $(D^{1/2}\Delta\mathbf{s})_i = d_i\Delta s_i$. So

$$\begin{aligned} d_i^{1/2} |\Delta s_i - \Delta \hat{s}_i| &= \left| d_i^{1/2} \Delta s_i - d_i^{1/2} \Delta \hat{s}_i \right| \\ &\leq \left\| D^{1/2} \Delta \mathbf{s} - D^{1/2} \Delta \hat{\mathbf{s}} \right\| \\ &\leq \epsilon \cdot \mu^{1/2} \cdot c \cdot g(A). \end{aligned}$$

Recall that $D^{1/2} = S^{-1/2}X^{1/2}$ and $XSe \approx \mu e$. So $d_i^{1/2} \approx \mu^{1/2}/s_i$ and

$$\frac{\mu^{1/2} |\Delta s_i - \Delta \hat{s}_i|}{s_i} \approx d_i^{1/2} |\Delta s_i - \Delta \hat{s}_i| \leq \epsilon \cdot \mu^{1/2} \cdot c \cdot g(A), \quad (23)$$

and thus,

$$\frac{|\Delta s_i - \Delta \hat{s}_i|}{s_i} \leq \epsilon \cdot c \cdot g(A) \quad (24)$$

for $1 \leq i \leq n$. Notice that this means that the error in each entry of $\Delta \mathbf{s}$ is bounded relative to the corresponding entry in \mathbf{s} , and the bound is independent of D . Therefore, the goal has been obtained.

Computing $\Delta \mathbf{x}$ with the required accuracy is now straightforward. First, recall that (15) says

$$\Delta \mathbf{x} = \bar{\mu} S^{-1} \mathbf{e} - \mathbf{x} - S^{-1} X \Delta \mathbf{s}.$$

Recall also that $D = S^{-1}X$. Substituting this into the previous equation and multiplying both sides by $D^{-1/2}$ yield

$$D^{-1/2} \Delta \mathbf{x} = D^{-1/2} (\bar{\mu} S^{-1} \mathbf{e} - \mathbf{x}) - D^{1/2} \Delta \mathbf{s}.$$

It is not difficult to show that the corresponding entries of both terms on the right-hand side are of similar magnitude. Since only scaling and subtraction are involved, it is easy to show that the forward error in each entry of $D^{-1/2} \Delta \mathbf{x}$ is bounded by a function independent of D . Using an argument similar to the one for $\Delta \mathbf{s}$, we find that the entries of $\Delta \mathbf{x}$ also satisfy an error bound of the form

$$\frac{|\Delta x_i - \Delta \hat{x}_i|}{x_i} \leq \epsilon \cdot c \cdot g(A), \quad (25)$$

where $\epsilon > 0$ is machine precision, c is a constant, and $g(A)$ is a function of A . These error bounds demonstrate that despite contributions from directions of different magnitudes, this method computes $\Delta \mathbf{x}$ and $\Delta \mathbf{s}$ with suitable relative accuracy in each component.

Now that the stability results have been established, it remains to be seen how the extended COD algorithm performs in practice. The results of computational experiments appear in the next section.

4 Numerical Results

The extended COD algorithm has been implemented and incorporated into two primal-dual interior point methods. The first is a feasible predictor-corrector method developed by Mizuno, Todd, and Ye [17], and the second is the infeasible predictor-corrector method of Mehrotra [16] (as implemented by Zhang in LIPSOL [31]). Each interior point algorithm was also implemented with some variation of Cholesky factorization for comparison purposes. The algorithms and the results of the experiments are described in the following subsections.

Two types of test problems were used. The first group includes small shortest path problems designed to be ‘‘torture’’ tests for the algorithms. Even

though there are specialized algorithms for solving shortest path problems, they make nice test problems for two reasons. First, the matrix A for these problems is a reduced node-arc incidence (RNAI) matrix. Vavasis showed that χ_A and $\bar{\chi}_A$ are bounded by the size of the matrix for RNAI matrices [24]. Thus, it is easy to evaluate the strength of the error bounds in this case. Secondly, the solution can be determined by inspection, so it is possible to determine whether or not the COD algorithm is performing according to expectations.

The other set of problems consists of the small NETLIB problems. These were tested in two ways. First, the COD implementations of the algorithms were used to compute the solutions to the problems. These solutions were compared to known solutions in order to determine whether or not the COD algorithm computed the correct solution. The results were also compared to those obtained using the Cholesky implementations. The problems were then modified in the following way: near-degeneracies were introduced by making some inactive constraints at the solution near active and then modifying the original problems accordingly. This was done in such a way that the solutions are unaffected. The results obtained using both the COD and the Cholesky implementations were compared in order to determine which solver performed better.

4.1 The Algorithm of Mizuno, Todd, and Ye

One of the interior point algorithms used for computational experiments is a predictor-corrector method proposed by Mizuno, Todd, and Ye [17]. Briefly, each main iteration consists of the following steps. A “predictor” step is computed by solving (4), (5), and (6) with $\bar{\mu} = 0$. Then a line search is performed to determine the longest step for which the next iterate will be within a specified distance $\alpha \in (0, 1)$ of the central path. For these experiments, the parameter α is chosen to be close to 1 to allow for longer steps rather than shorter, more centered steps. After $\mathbf{x}, \mathbf{y}, \mathbf{s}$, and μ have been updated according to the predicted step, the iterates are centered. This requires repeatedly solving (4), (5), and (6) with the new value of μ and updating \mathbf{x}, \mathbf{y} , and \mathbf{s} until the result is within $\beta \in (0, 1)$ of the central path, where $\beta < \alpha$. In contrast to α , β is close to 0 since the goal is to bring the iterates in close to the central path. An outline of the algorithm is given below.

Algorithm: Mizuno, Todd, and Ye

Given an approximately centered starting point $(\mathbf{x}^{(0)}, \mathbf{y}^{(0)}, \mathbf{s}^{(0)})$ and $\mu^{(0)}$, proceed as follows.

While $(\mathbf{x}^{(k)})^T \mathbf{s}^{(k)} > tol$

(predictor step)

compute $\Delta \mathbf{x}_p^{(k)}, \Delta \mathbf{y}_p^{(k)}, \Delta \mathbf{s}_p^{(k)}$

determine largest θ such that $\left\| \frac{(\mathbf{x}^{(k)} + \theta \Delta \mathbf{x}_p^{(k)}) \cdot (\mathbf{s}^{(k)} + \theta \Delta \mathbf{s}_p^{(k)})}{\mu^{(k)}} - 1 \right\| < \alpha$

set $\bar{\mathbf{x}} = \mathbf{x}^{(k)} + \theta \Delta \mathbf{x}_p^{(k)}, \bar{\mathbf{y}} = \mathbf{y}^{(k)} + \theta \Delta \mathbf{y}_p^{(k)}, \bar{\mathbf{s}} = \mathbf{s}^{(k)} + \theta \Delta \mathbf{s}_p^{(k)}, \mu^{(k)} = (1 - \theta) \mu^{(k)}$

While $\left\| \frac{\bar{\mathbf{x}} \cdot \bar{\mathbf{s}}}{\mu^{(k)}} - 1 \right\| > \beta$

```

      (corrector step)
      compute  $\Delta \mathbf{x}_c^{(k)}, \Delta \mathbf{y}_c^{(k)}, \Delta \mathbf{s}_c^{(k)}$ 
      set  $\bar{\mathbf{x}} = \bar{\mathbf{x}} + \Delta \mathbf{x}_c^{(k)}, \bar{\mathbf{y}} = \bar{\mathbf{y}} + \Delta \mathbf{y}_c^{(k)}, \bar{\mathbf{s}} = \bar{\mathbf{s}} + \Delta \mathbf{s}_c^{(k)}$ 
    end
    set  $k = k + 1, \mathbf{x}^{(k)} = \bar{\mathbf{x}}, \mathbf{y}^{(k)} = \bar{\mathbf{y}}, \mathbf{s}^{(k)} = \bar{\mathbf{s}}$ 
  end
end

```

The question that remains is that of how to obtain the feasible starting point $(\mathbf{x}^{(0)}, \mathbf{y}^{(0)}, \mathbf{s}^{(0)})$ and μ_0 . In order to do this, the algorithm is implemented in a Phase I - Phase II manner. The linear programming problem itself is solved in Phase II, but the initial point is determined in Phase I. The procedure is as follows. Artificial variables are introduced into the original problem in order to construct a larger linear programming problem for which a feasible starting point is known. This larger problem can be solved to optimality, and it would be possible to obtain the optimal solution to the original problem. However, this would require more work than solving the original problem, so the interior point method is run on the larger problem only until a suitable starting point for the original problem can be extracted. Then the interior point method is used to solve the original problem with the starting point found in Phase I. The results of the computational tests are taken from Phase II, so we go into no further detail about Phase I.

An example from the shortest path test problems is given in Figure 2. The matrix A is the reduced node-arc incidence matrix for this graph, the entries of the vector \mathbf{c} are the edge weights, and the vector \mathbf{b} contains a -1 in the fifth entry and a 1 in the second entry (thus describing the locations of the source and the sink). Recall that two different methods for solving the weighted least-squares problems were used. The first is the COD algorithm. The other is solving the normal equations via Cholesky factorization. For these tests, the MATLAB Cholesky function was used. When $\delta = 10^{-6}$, the Cholesky implementation of the Mizuno, Todd, and Ye algorithm reaches a point where the coefficient matrix of the normal equations is no longer recognized by MATLAB as being positive definite. The algorithm stops at this point without finding the shortest path. On the other hand, the COD implementation computes the shortest path with accuracy on the order of machine precision.

The next set of examples includes some of the small problems in the NETLIB test set. The purpose of these tests is to verify that the implementations work correctly for problems where the solution is already known. The results are shown in Table 1. Notice that both the Cholesky and the COD implementations compute the correct solution, and the same number of calls to the solver of the weighted least-squares problem is made in each case. This is not surprising since the Cholesky solvers are quite robust, and we expect the real advantage of the COD algorithm to be seen in near-degenerate problems.

When near-degeneracy is introduced into the problems, we again find that both implementations make the same number of calls to the solver as they converge to the solution. (See Table 1.) Similar results were obtained for other NETLIB test problems used for numerical tests. It is unclear at this point why the performance of the Cholesky algorithm and the COD algorithm on the near-

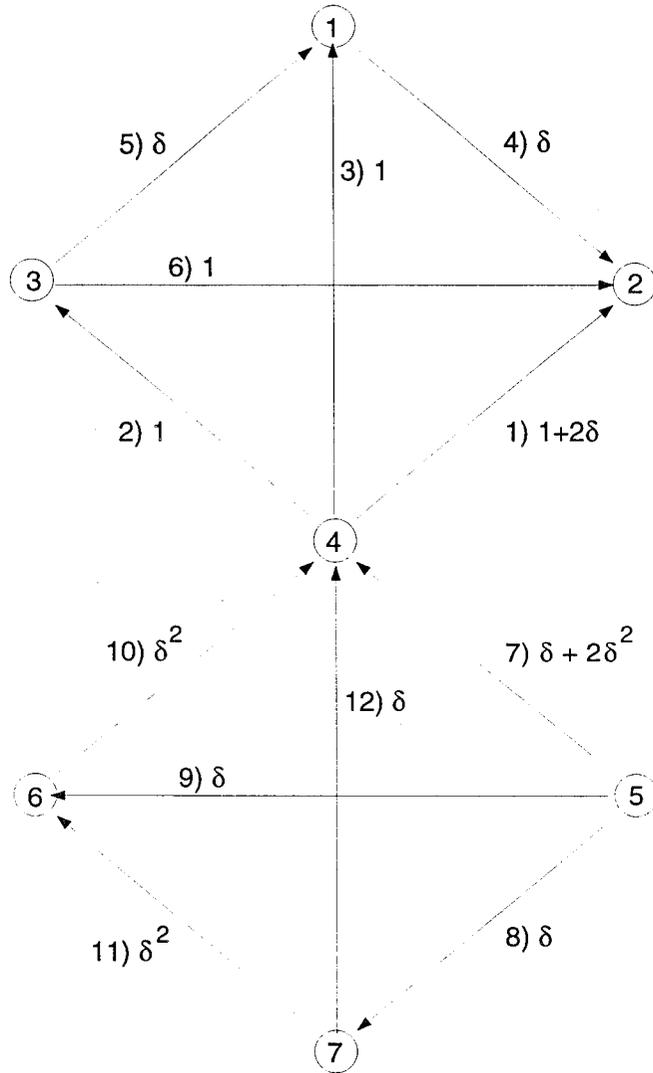


Figure 2: The above graph illustrates a near-degenerate shortest path problem. It is clear that the shortest path from the source (node 5) to the sink (node 2) consists of edges 9, 10, 3, and 4. However, the Cholesky implementations of the interior point methods fail with small values of δ while the COD implementations find the correct path.

Table 1: This table contains a summary of the numerical results for the NETLIB test problems solved using the Mizuno, Todd, and Ye algorithm. Notice that the performance of the two implementations is comparable.

	problem	number of calls to Cholesky	number of calls to COD
NETLIB problems unchanged	afiro	21	22
	sc50a	30	30
	sc50b	27	27
with near- degeneracy	afiro	42	43
	sc50a	51	52
	sc50b	25	25

degenerate NETLIB problems is so similar. However, while these results may be inconclusive, the results obtained for the shortest path problems (like the one in Figure 2) indicate that there are cases in which it is clearly advantageous to use the COD algorithm.

4.2 LIPSOL

LIPSOL is an implementation (written by Yin Zhang) of the interior point method of Mehrotra. This is an infeasible second-order predictor-corrector method. Because it is an infeasible method, the iterates move toward feasibility as they move toward the solution, and the equation which forces centrality is not linearized since the method is second order. Therefore, the system of equations to be solved is somewhat different than (4), (5), and (6). In this setting, it is necessary to compute $(\Delta\mathbf{x}, \Delta\mathbf{y}, \Delta\mathbf{s})$ such that

$$\begin{aligned}
 A\Delta\mathbf{x} &= \mathbf{b} - A\mathbf{x}, \\
 A^T\Delta\mathbf{y} + \Delta\mathbf{s} &= \mathbf{c} - A^T\mathbf{y} - \mathbf{s}, \\
 X\Delta\mathbf{s} + S\Delta\mathbf{x} &= \mu\mathbf{e} - X\mathbf{S}\mathbf{e} - \Delta X\Delta\mathbf{S}\mathbf{e}.
 \end{aligned} \tag{26}$$

The solution of this system of equations is broken down into two steps. The first is the predictor step, which finds the affine scaling search direction. This is accomplished by solving

$$\begin{aligned}
 A\Delta\mathbf{x}_p &= \mathbf{b} - A\mathbf{x}, \\
 A^T\Delta\mathbf{y}_p + \Delta\mathbf{s}_p &= \mathbf{c} - A^T\mathbf{y} - \mathbf{s}, \\
 X\Delta\mathbf{s}_p + S\Delta\mathbf{x}_p &= -X\mathbf{S}\mathbf{e}.
 \end{aligned} \tag{27}$$

Notice that the nonzero terms on the right-hand side introduce additional terms into the least-squares problem defined in §2. Following a derivation similar to that in §2 yields the following system normal equations:

$$ADA^T\Delta\mathbf{y}_p = AD[(\mathbf{c} - A^T\mathbf{y}) + D^{-1}\mathbf{p}], \tag{28}$$

where $\mathbf{p} = A^T \mathbf{q}$, $AA^T \mathbf{q} = \mathbf{b} - A\mathbf{x}$, and $D = S^{-1}X$. Notice that $\mathbf{c} - A^T \mathbf{y}$ looks like \mathbf{s} and \mathbf{p} looks like $\Delta \mathbf{x}_p$, so the corresponding entries of the terms on the right-hand side have the same order of magnitude.

We take a moment here to point out that the Cholesky implementation used in LIPSOL is the sparse Cholesky algorithm of Ng and Peyton [20], with a slight modification in the handling of small diagonal entries which is described in [31]. Thus, LIPSOL is expected to be quite robust.

Similarly, the equations defining $\Delta \mathbf{x}_p$ and $\Delta \mathbf{s}_p$ are somewhat different. Continuing with the derivation as in §3 yields the following equation for the scaled version of $\Delta \mathbf{s}_p$:

$$D^{1/2} \Delta \mathbf{s}_p = D^{1/2}(\mathbf{c} - A^T \mathbf{y} - \mathbf{s}) - PZ_1 Z_1^T P^T \left[D^{1/2}(\mathbf{c} - A^T \mathbf{y}) + D^{-1/2} \mathbf{p} \right]. \quad (29)$$

We offer an intuitive explanation of why $\Delta \mathbf{s}_p$ computed in this manner satisfies (24). As the solution is approached, the first term on the right-hand side tends to zero and makes a negligible contribution. In the second term, $\mathbf{c} - A^T \mathbf{y}$ tends to \mathbf{s} , and \mathbf{p} looks like $\Delta \mathbf{x}_p$. So arguments similar to those in §3 imply that the nonorthogonal factor in the second term looks like $\mu^{1/2} \mathbf{e}$, and an error bound like (24) can be obtained. Arguments similar to those in §3 imply that if $\Delta \mathbf{x}_p$ is computed by using

$$D^{-1/2} \Delta \mathbf{x}_p = -D^{-1/2} \mathbf{x} - D^{1/2} \Delta \mathbf{s}_p, \quad (30)$$

then it will satisfy an error bound like (25).

After the predictor step is determined, the corrector step is computed. This takes into account the centering and the second-order term. The values of $\Delta \mathbf{x}_p$ and $\Delta \mathbf{s}_p$ computed in the previous step are used to “predict” the second-order term, and the system of equations solved at this step is given by

$$\begin{aligned} A \Delta \mathbf{x}_c &= \mathbf{0}, \\ A^T \Delta \mathbf{y}_c + \Delta \mathbf{s}_c &= \mathbf{0}, \\ X \Delta \mathbf{s}_c + S \Delta \mathbf{x}_c &= \mu \mathbf{e} - \Delta X_p \Delta S_p \mathbf{e}. \end{aligned} \quad (31)$$

Again, the equations look somewhat different from those for a feasible method, but the stability arguments are similar. Thus, the specific equations and discussions are omitted here.

To obtain the solution to (26), the solutions to (27) and (31) are combined as follows:

$$\begin{aligned} \Delta \mathbf{x} &= \Delta \mathbf{x}_p + \Delta \mathbf{x}_c, \\ \Delta \mathbf{y} &= \Delta \mathbf{y}_p + \Delta \mathbf{y}_c, \\ \Delta \mathbf{s} &= \Delta \mathbf{s}_p + \Delta \mathbf{s}_c. \end{aligned}$$

There was one minor change made to the LIPSOL code. In LIPSOL, there are constant upper bounds imposed on the entries of the weight matrix D . These prevent D becoming arbitrarily ill conditioned, and it is not known how these caps affect the convergence of the interior point algorithm (since the systems of equations are being solved with a perturbed D). As we are interested in

Table 2: This table contains a summary of the numerical results for the NETLIB test problems solved using LIPSOL. Notice that while the performances of the Cholesky implementation and the COD implementation are comparable for the unchanged problems, the COD implementation is more reliable for the near-degenerate problems. Here n.c. indicates that the algorithm halted without converging.

	problem	number of iterations, Cholesky	number of iterations COD
NETLIB problems unchanged	afiro	7	7
	sc50a	9	9
	sc50b	8	8
with near-degeneracy	afiro	n.c.	11
	sc50a	n.c.	12
	sc50b	n.c.	9

the performance of algorithms when D becomes arbitrarily ill conditioned, the caps were removed.

LIPSOL was used to solve the same set of test problems as the Mizuno, Todd, and Ye algorithm. For the example illustrated in Figure 2 with $\delta = 10^{-8}$, the Cholesky implementation converges to

$$[0 \ 0 \ 1 \ 1 \ 0 \ 0 \ .2 \ .8 \ 0 \ .8 \ .8 \ 0]^T,$$

which is not even a valid path. On the other hand, the COD implementation finds the correct shortest path.

Table 2 contains the results for the NETLIB problems. For the unmodified problems, both implementations converge to the solution in the same number of iterations (with two calls to the linear system solver per iteration). For the near-degenerate problems, however, the Cholesky implementation halts without converging, but the COD implementation converges to the solution. Thus, the advantage of using the extended COD algorithm is clear in these experimental results.

5 Related Work

The problem of stably computing search directions for interior point methods has received a fair amount of attention in the literature [9]. Among those who have studied such problems are Coleman and Liu [5], Forsgren, Gill, and Shinnerl [8], Gill, Saunders, and Shinnerl [10], Gould [12], Murray [18], Nash and Sofer [19], M. Wright [27], and S. Wright [29]. One difference between these other works and this one can be summarized as follows. The other authors typically consider a more general problem, one in which the weighted least-squares problem has the form $\min \|H^{1/2}(A\mathbf{y} - \mathbf{b})\|$, where H is symmetric and positive definite, but not necessarily diagonal. This is a problem that we

currently cannot address with the techniques presented in this paper. In some recent work, Forsgren [7] derives a result similar to Theorem 1 for such matrices H that are also diagonally dominant. This is accomplished by defining a new type of diagonal decomposition which is used to transform the problem into one to which Theorem 1 applies. If this decomposition can be found accurately, then the methods of this paper can be used. However, the applicability of the extended COD algorithm to the general case has not yet been determined.

When the problems are specialized to those with diagonal weight matrices D , the aforementioned authors consider a more restricted problem in that they all make an assumption that the large and small entries on the diagonal of D have some correlation with the columns of A^T . This corresponds to a nondegeneracy assumption about the underlying optimization problem. In contrast, the stability analysis of the COD method does not involve any restrictions about where “large” versus “small” entries of D can appear. Thus, it holds without nondegeneracy assumptions and is more general than other algorithms in this respect.

6 Conclusions

A common approach to solving linear programming problems is to use a primal-dual interior point method. In order to compute the search directions $(\Delta\mathbf{x}, \Delta\mathbf{y}, \Delta\mathbf{s})$ needed at each iteration, it is necessary to solve a linear system of equations, the Newton equations. In particular, a weighted least-squares problem must first be solved to obtain $\Delta\mathbf{y}$. However, near-degeneracy in the linear programming instance can lead to ill-conditioning in the weight matrix, which can cause standard methods for solving weighted least-squares problems to compute solutions with only a few or, in some cases, no digits of accuracy. This is the problem addressed by Hough and Vavasis, and the COD algorithm proposed in that paper can be used in this situation.

As in the computation of $\Delta\mathbf{y}$, near-degeneracy can cause numerical problems in the obvious method for finding $\Delta\mathbf{s}$ and $\Delta\mathbf{x}$. This is the issue that has been addressed in this paper. To obtain these search directions, we use orthogonal projections (formed by factors from the COD algorithm) to compute scaled directions. The actual search directions are then determined by unscaling. We have demonstrated that when the search directions are computed in this way, each component has a relative error that is bounded despite the ill-conditioning. Unlike those of other methods, the stability analysis presented here makes no nondegeneracy assumptions. Thus, we expect the advantages of the extended COD algorithm to be seen when it is used to solve near-degenerate problems, and numerical experiments do indicate that the COD algorithm is more reliable in this case.

This paper has addressed accuracy issues that arise in the computation of search directions for interior point methods. Another important consideration is efficiency. The systems of equations that arise in this setting are typically large and sparse. Since the COD algorithm is based on dense methods, it is clearly not the most efficient way to solve these problems. Bobrovnikova and Vavasis have investigated using iterative methods in this setting. They have proposed

algorithms (for solving the weighted least-squares problem) based on conjugate gradients and MINRES [4]. In the case of MINRES, they have proved that their method converges and that the solution satisfies an error bound like (9). Numerical results are promising; however, the algorithm experiences a loss of orthogonality in the search directions for some examples, causing the algorithm to take more iterations to converge than one would hope. Additionally, this algorithm has not yet been extended to compute Δs and Δx . These and other issues (see [4]) are the topics of continuing work.

Another approach is to use the LIP method proposed by Vavasis and Ye [26]. This algorithm accelerates classical path-following algorithms by interleaving traditional steps with longer steps, known as layered-least squares steps. As with the extended COD algorithm, the advantage of the LIP method is most notable near the solution of the optimization problem and when there is near-degeneracy in the linear programming instance. The weighted least-squares problem that describes the layered least-squares step can be solved using a modification of standard sparse Cholesky techniques, and the solution satisfies (9). It is also possible to compute Δs and Δx with suitable accuracy. Again, numerical results are promising, but there are some issues that still must be resolved. A more detailed discussion of this approach is postponed until a future work.

7 Acknowledgements

The author wishes to thank Steve Vavasis and Elena Bobrovnikova for many helpful discussions and comments.

References

- [1] D. A. Bayer and J. C. Lagarias. The nonlinear geometry of linear programming. I. Affine and projective scaling trajectories. *Transactions of the AMS*, 314:499–526, 1989.
- [2] D. A. Bayer and J. C. Lagarias. The nonlinear geometry of linear programming. II. Legendre transform coordinates and central trajectories. *Transactions of the AMS*, 314:527–581, 1989.
- [3] D. A. Bayer and J. C. Lagarias. The nonlinear geometry of linear programming. III. Projective Legendre transform coordinates and Hilbert geometry. *Transactions of the AMS*, 320:193–225, 1990.
- [4] E. Bobrovnikova and S. Vavasis. Accurate solution of weighted least squares by iterative methods. Technical Report ANL/MCS-P644-0297, Mathematics and Computer Science Division, Argonne National Laboratory, Chicago, IL, 1997.
- [5] T. F. Coleman and J. Liu. An interior Newton method for quadratic programming. Technical Report CTC93TR153, Advanced Computing Research Institute, Cornell University, Ithaca, NY, 1993.

- [6] I. I. Dikin. On the speed of an iterative process. *Upravlyaemye Sistemi*, 12:54–60, 1974.
- [7] A. L. Forsgren. On linear least-squares problems with diagonally dominant weight matrices. Report TRITA-MAT-1995-OS2, Optimization and Systems Theory, Department of Mathematics, Royal Institute of Technology, S-100 44 Stockholm, Sweden, 1995.
- [8] A. L. Forsgren, P. E. Gill, and J. R. Shinnerl. Stability of symmetric ill-conditioned systems arising in interior methods for constrained optimization. *SIAM J. Matrix Anal. Appl.*, 17:187–211, 1996.
- [9] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization*. Academic Press, London, 1981.
- [10] P. E. Gill, M. A. Saunders, and J. R. Shinnerl. On the stability of the Cholesky factorization for symmetric quasi-definite systems. *SIAM J. Matrix Anal. Appl.*, 17:35–46, 1996.
- [11] C. C. Gonzaga. Path-following methods for linear programming. *SIAM Review*, 34:167–224, 1992.
- [12] N. Gould. On the accurate determination of search directions for simple differentiable penalty functions. *IMA Journal of Numerical Analysis*, 6:357–372, 1986.
- [13] N. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM Press, 1996.
- [14] P. D. Hough and S. A. Vavasis. Complete orthogonal decomposition for weighted least squares. To appear in *SIAM J. Matrix Anal. Appl.*, 1997.
- [15] N. Megiddo. Pathways to the optimal set in linear programming. In N. Megiddo, editor, *Progress in Mathematical Programming: Interior Point and Related Methods*, pages 131–158. Springer, New York, 1989.
- [16] S. Mehrotra. On the implementation of a primal-dual interior point method. *SIAM J. Optimization*, pages 575–601, 1992.
- [17] S. Mizuno, M. J. Todd, and Y. Ye. A surface of analytic centers and infeasible interior-point algorithms for linear programming. *Mathematics of Operations Research*, 20:135–162, 1995.
- [18] W. Murray. Analytical expressions for the eigenvalues and eigenvectors of the Hessian matrices of barrier and penalty functions. *J. Optimization Theory and Applications*, 7:189–196, 1971.
- [19] S. G. Nash and A. Sofer. A barrier method for large-scale constrained optimization. *ORSA Journal on Computing*, 5:40–53, 1993.
- [20] E. Ng and B. W. Peyton. Block sparse Cholesky algorithms on advanced uniprocessor computers. *SIAM J. Scientific Computing*, 14:1034–1056, 1993.
- [21] G. Sonnevend. An “analytic center” for polyhedrons and new classes of global algorithms for linear (smooth, convex) programming. In A. Prekopa, J. Szelezsan, and B. Strazicky, editors, *System Modelling and Optimization: Proceedings of the 12th IFIP-Conference held in Budapest, Hungary*,

September 1985, volume 84 of *Lecture Notes in Control and Information Sciences*, pages 866–876. Springer Verlag, Berlin, Germany, 1986.

- [22] G. W. Stewart. On scaled projections and pseudoinverses. *Linear Algebra and its Applications*, 112:189–193, 1989.
- [23] M. J. Todd. A Dantzig-Wolfe-like variant of Karmarkar’s interior-point linear programming algorithm. *Operations Research*, 38:1006–1018, 1990.
- [24] S. A. Vavasis. Stable numerical algorithms for equilibrium systems. *SIAM J. Matrix Anal. Appl.*, 15:1108–1131, 1994.
- [25] S. A. Vavasis. Stable finite elements for problems with wild coefficients. *SIAM J. Num. Anal.*, 33:890–916, 1996.
- [26] S. A. Vavasis and Y. Ye. A primal-dual interior point method whose running time depends only on the constraint matrix. *Math Programming*, 74:79–120, 1996.
- [27] M. H. Wright. Determining subspace information from the Hessian of a barrier function. Technical Report 92-02, AT&T Bell Laboratories Numerical Analysis Manuscript, 1992.
- [28] M. H. Wright. Interior methods for constrained optimization. In *Acta Numerica 1992*. Cambridge University Press, Cambridge, 1992.
- [29] S. J. Wright. Stability of linear algebra computations in interior-point methods for linear programming. Technical Report MCS-P446-0694, Mathematics and Computer Science Division, Argonne National Laboratory, Chicago, IL, 1994.
- [30] S. J. Wright. *Primal-dual Interior-point Methods*. SIAM, Philadelphia, PA, 1997.
- [31] Y. Zhang. Solving large-scale linear programs by interior-point methods under the MATLAB environment. Technical Report TR96-01, Department of Mathematics and Statistics, University of Maryland Baltimore County, Baltimore, MD, 1996.

UNLIMITED RELEASE

INITIAL DISTRIBUTION:

Stephen Vavasis
MCS Division, Building 221
Argonne National Laboratory
9700 S. Cass Ave.
Argonne, IL 60439

Kim-Chuan Toh
National University of Singapore
10 Kent Ridge Crescent
Singapore 0511
Singapore

Javier Peña
Center for Applied Mathematics
657 Rhodes Hall
Cornell University
Ithaca, NY 14853

Stephen Wirkus
Center for Applied Mathematics
657 Rhodes Hall
Cornell University
Ithaca, NY 14853

MS 9004 M. E. John, 8100
MS 9214 L. M. Napolitano, Jr., 8117
MS 9214 J. C. Meza, 8117
MS 9214 P. D. Hough (10), 8117
MS 9021 Technical Communications, 8815, for OSTI (10)
MS 9021 Technical Communications, 8815/Technical Library, MS 0899, 4414
MS 0899 Technical Library (4), 4414
MS 9018 Central Technical Files (3), 8940-2