Final Iterations in Interior Point Methods — Preconditioned Conjugate Gradients and Modified Search Directions

Weichung Wang

August 4, 1996

Abstract

In this article we consider modified search directions in the endgame of interior point methods for linear programming. In this stage, the normal equations determining the search directions become ill-conditioned. The modified search directions are computed by solving perturbed systems in which the systems may be solved efficiently by the preconditioned conjugate gradient solver. We prove the convergence of the interior point methods using the modified search directions and show that each barrier problem is solved with a superlinear convergence rate. A variation of Cholesky factorization is presented for computing a better preconditioner when the normal equations are ill-conditioned. These ideas have been implemented successfully and the numerical results show that the algorithms enhance the performance of the preconditioned conjugate gradients-based interior point methods.

1. Introduction

The development of interior point methods has led to many successful implementations that may efficiently solve linear programming problems

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Ax = b, \\ & x \geq 0, \end{array} \tag{1}$$

where c and x are real n-vectors, b is a real m-vector, and $A \in \mathbb{R}^{m \times n}$ is a real matrix of rank m with $m \leq n$. These methods eliminate the inequalities in (1) by applying a logarithmic barrier function with a barrier parameter μ and then forming the Lagrangian of the barrier subproblem. A sequence of Lagrangians corresponding to a sequence of barrier parameters $\{\mu_k\}$, with μ_k decreasing to zero, are solved for iterates converging to the optimal solution to the linear programming problem. The first order optimality conditions of the Lagrangian are

$$\begin{cases} XZe - \mu e = 0, \\ c - A^T y - z = 0, \\ Ax - b = 0. \end{cases} \tag{2}$$

The vector z here is a dual slack variable, y contains Lagrangian multipliers, and e is a vector with all 1's. The diagonal matrices X and Z contain x and z in their main diagonals respectively.

Newton's method is used to solve the nonlinear system (2) and the search direction is then determined by solving the KKT (Karush-Kuhn-Tucker) system

$$\begin{pmatrix} X & Z & 0 \\ -I & 0 & A^T \\ 0 & A & 0 \end{pmatrix} \begin{pmatrix} \Delta z \\ \Delta x \\ -\Delta y \end{pmatrix} = - \begin{pmatrix} Xz - \mu e \\ c - A^T y - z \\ Ax - b \end{pmatrix}. \tag{3}$$

Equivalently, we may first solve the normal equations

$$(A\Theta A^T)\Delta y = A\Theta(r_d + Ze - \mu X^{-1}e) + r_p, \tag{4}$$

where $r_p = b - Ax$, $r_d = c - A^T y - z$, and $\Theta = Z^{-1}X$, and then compute

$$\Delta x = \Theta(A^T \Delta y - r_d - Ze + \mu X^{-1} e) \tag{5}$$

and

$$\Delta z = r_d - A^T \Delta y. \tag{6}$$

Either direct methods or iterative methods may be used to solve the systems to determine the search directions. The computation of the search directions is the bulk of the computational effort for interior point methods and thus accelerating this computation is a key problem.

In this article, we focus on solving the normal equations by the preconditioned conjugate gradient method for determining the search directions. Good preconditioners are necessary to make it competitive, but they are difficult to find: the requirements of accuracy for beginning and later stages are greatly different, the matrix Θ may change wildly, and Θ becomes very ill-conditioned when iterates become close to a optimum. To overcome these difficulties, Wang and O'Leary [23] recently proposed an algorithm that adaptively chooses either a using direct method or preconditioned conjugate gradients. They also discussed adaptive preconditioning strategies that either recompute a Cholesky factorization $A\Theta A^T = LPL^T$, where L is an $m \times m$ unit lower triangular matrix and P is diagonal, or apply α rank-1 updates. That is, the current preconditioner is computed as

$$LPL^T + \sum_{\alpha\, largest} \Delta\Theta_{ii} a_i a_i^T,$$

where $\Delta\Theta$ is the difference between the current Θ and the previous $\hat{\Theta}$ satisfying $A\hat{\Theta}A^T = LPL^T$ and a_i is the i-th column of A. The adaptive algorithm switches to a direct method whenever P contains a zero element in its main diagonal. This situation is due to ill-conditioning in Θ and may be found in the endgame of many linear programming problems. Consequently, though the computational results reported in [23] are promising, there is room for improvement.

We improve the algorithm in [23] by considering modified search directions in the endgame. When the iterates are close to optimal solutions, we perturb small entries in the slack variables z in the left hand side of equation (3), so that preconditioned conjugate gradients converges rapidly. We also show that the iteration using the modified search directions converges to the solution of the first order optimality conditions (2) with a fixed μ. A superlinear convergence rate of the iterations is also proved.

All the ideas are implemented by modifying a well-coded direct method based interior point method program, OB1-R [15]. Numerical experiments demonstrate that the timing may be improved by using preconditioned conjugate gradients through the whole interior point method process and using the modified search directions in the endgame.

We survey some other related works. Many papers (i.e. [11], [25], and [15]) address theoretical and implementation aspects of interior point methods. Direct methods relying on sparse Cholesky factorization were used to solve the normal equations by Lustig, Marsten, and Shanno (OB1-R) [15], Czyzyk, Mehrotra, and Wright (PCx) [2], Zhang (LIPSOL) [26], and other researchers. Iterative methods, in contrast, were also considered, since iterative methods may take advantage the fact that approximate solutions are allowed in the early stage of an interior point method. See, for example, Freund and Jarre [7], Portugal, Resende, Veiga, and Júdice [18], and Mehrotra and Wang [16]. Mizuno and Jarre [17] proposed and further analyzed an infeasible-interior-point algorithm using inexact solutions of the reduced KKT system as a search directions. On the other hand, many recent studies concentrated on the stability of the highly ill-conditioned systems which may be found in the endgame of interior point methods. Hough and Vavasis [12] considered weighted least-squares problems with a highly ill-conditioned weight matrix. They proposed a complete orthogonal decomposition algorithm which is stable in the sense that its forward error bound is independent of the matrix Θ. In [6], Forsgren, Gill, and Shinnerl presented a perturbation analysis of a class of symmetric diagonally ill-conditioned systems and gave a rounding-error analysis for symmetric indefinite matrix factorization.

In the next section, we discuss ideas for perturbing the normal equations to obtain modified search directions and then propose an algorithm based on the ideas. The modification is closely akin to that proposed by Karmarkar [14] in order to reduce the complexity of his interior point method to $O(n^{2.5})$ by updating a matrix rather than recomputing it. The differences in formulation are that his was a primal algorithm, while ours is primal-dual, our choice of parameters is somewhat different, and we solve the linear systems iteratively, taking advantage of the fact that the modified systems are much easier to solve than the original ones. We also prove the convergence properties of the algorithm and give a superlinear convergence rate for solving the first order optimality conditions. Section 3 discusses implementation issues for finding the modified search directions. Numerical results are presented in § 5. Finally, we conclude the article in § 6.

1.1. Assumptions and Notations

We assume that

$(A1)$ matrix $A \in \mathbb{R}^{m\times n}$, with $n > m$, has full row rank;

$(A2)$ $X_b = \{x \in \mathbb{R}^n \mid Ax = b,\ x \geq 0\}$ is compact.

We introduce the following notation to be used throughout the article. Let e be the vector in which all elements are 1's, and let e_i be the vector with all 0's except that the i-th component is equal to 1. Let K denote the matrix $A\Theta A^T$. If C is a square matrix, $diag(C)$ is the vector formed from the main diagonal of C; if v is a vector, $diag(v)$ is a diagonal matrix with the elements of v on the main diagonal.

The variables x_j, y_j, and z_j denote the j-th vector in the sequence $\{x_j\}$, $\{y_j\}$, and $\{z_j\}$, respectively. The Greek variable χ_i denotes the i-th component of the vector x_j, where the index of x will be clear from the context, i.e. $x_j = (\chi_1, \cdots, \chi_n)^T$. Similarly, we let $y_j = (\eta_1, \cdots, \eta_m)^T$ and $z_j = (\zeta_1, \cdots, \zeta_n)^T$ for $y_j \in \mathbb{R}^m$ and $z_j \in \mathbb{R}^n$.

The solution of (2) for a fixed μ is denoted as $x^*(\mu)$, $y^*(\mu)$, and $z^*(\mu)$. Capital letters X, Y, and Z denote diagonal matrices containing vectors x, y, and z on the main diagonals respectively. Let $S_Y = \{y \in \mathbb{R}^m \mid \|y\| \leq \Lambda_Y\}$ and $S_Z = \{z \in \mathbb{R}^n \mid 0 < \Omega_Z e \leq z \leq \Lambda_Z e\}$, where $\Lambda_Y, \Omega_Z, \Lambda_Z$ are positive numbers.

2. Modified search directions for the endgame

We consider the course of the algorithm in the endgame, where iterates x, y, and z are close to the solution of equation (2). The strict complementarity implies that, for each i, either χ_i or ζ_i is close to zero in the relative interior of a non-singleton solution set. See, for example, [26]. The resulting diagonal matrix Θ, in which the i-th diagonal entry is

$$\Theta_{ii} = \frac{\chi_i}{\zeta_i},$$

consequently contains some very small positive entries and some irregularly distributed large entries corresponding to small ζ_i's. Moreover, these wildly changing entries may cause troubles for the preconditioned conjugate gradient solver using the updated preconditioner. The observation, however, that

$$A\Theta A^T = \sum_{i=1}^{n} \Theta_{ii} a_i a_i^T$$

suggests that only large diagonal elements in Θ are significant, where a_i is the i-th column of matrix A. We further observe that a slight perturbation in the small ζ_i's may result in a significant change in the corresponding large Θ_{ii}'s. The following question is then raised: is it possible to slightly perturb those small ζ_i's, such that the preconditioned conjugate gradient method may benefit? In other words, we hope to find a modified search direction by solving perturbed normal equations where the new system can be easily solved by preconditioned conjugate gradients. At the same time, the outer iterations of the interior point method can still converge and the performance will not be degraded.

The answer to the question is positive and we propose a method to achieve this goal. We define some notation first. Let $\hat{\Theta}$ be a previous diagonal matrix for which we have a preconditioner $C_{\hat{\Theta}} = LPL^T = A\hat{\Theta}A^T$. We may partition the diagonal entries of $\hat{\Theta}$ as $[\hat{\Theta}^B, \hat{\Theta}^S]$, where $\hat{\Theta}^B$ and $\hat{\Theta}^S$ contain the big and small entries in $\hat{\Theta}$ respectively. The matrix Θ is then partitioned compatibly as $[\Theta^B, \Theta^S]$. The main idea is that we slightly perturb the small ζ's so that the resulting perturbed matrix $\overline{\Theta}^B = \kappa\hat{\Theta}^B$. Rather than perturb all the Θ^B entries, however, we may wish to perturb only Θ^{B_1}, the part of Θ^B containing really small ζ's and fairly large χ's. Therefore, the corresponding perturbation sizes remain small. Let

$$\Theta^B = [\Theta^{B_1},\ \Theta^{B_2}]. \tag{7}$$

We may choose $\varepsilon_i \in \mathbb{R}, \forall\, i \in B_1$, such that

$$\overline{\zeta}_i = \zeta_i + \varepsilon_i,\ \text{where } \overline{\zeta}_i > 0 \tag{8}$$

and

$$\begin{aligned} (\overline{\Theta}^{B_1})_{ii} &= \tfrac{\chi_i}{\overline{\zeta}_i} \\ &= \tfrac{\chi_i}{\zeta_i+\varepsilon_i} \\ &= \kappa(\hat{\Theta}^{B_1})_{ii}. \end{aligned} \tag{9}$$

The perturbed system is then

$$\begin{aligned} A\overline{\Theta}A^T &= \sum \overline{\Theta}_{ii}^{B_1} a_i a_i^T \; + \; \sum \Theta_{ii}^{B_2} a_i a_i^T \; + \; \sum \Theta_{ii}^{S} a_i a_i^T \\ &= \kappa(LPL^T) \; + \; \sum \Delta\Theta_{ii}^{B_2} a_i a_i^T \; + \; \sum \Delta\Theta_{ii}^{S} a_i a_i^T, \end{aligned} \tag{10}$$

where $\Delta\Theta_{ii}^{B_2} = \Theta_{ii}^{B_2} - \kappa\hat{\Theta}_{ii}^{B_2}$. Using LPL^T as a preconditioner of (10),

$$\begin{aligned} &(LPL^T)^{-1}(A\overline{\Theta}A^T) = \\ &\qquad \kappa I \; + \; (LPL^T)^{-1} \sum \Delta\Theta_{ii}^{B_2} a_i a_i^T \; + \; (LPL^T)^{-1} \sum \Delta\Theta_{ii}^{S} a_i a_i^T. \end{aligned} \tag{11}$$

If we perturb most of the small ζ's, the second term in (11) will be a small rank matrix and the third term has small rank or norm. Consequently, the preconditioned conjugate gradient method will converge rapidly. For further improving the performance of the preconditioned conjugate gradients, we may apply rank-1 updates on some largest $|\Delta\Theta_{ii}^{B_2}|$'s.

2.1. The theoretical algorithm

The discussion above leads to Algorithm 1. The algorithm is a *theoretical* algorithm and presented using the form of KKT system. Section 3 discusses a implementable variety of the algorithm in normal equations form. Briefly speaking, to determine the search directions for each μ, the algorithm solves a sequence of perturbed 3×3 block KKT systems where the j-th in the sequence is

$$\begin{pmatrix} X_j & \overline{Z}_j & 0 \\ -I & 0 & A^T \\ 0 & A & 0 \end{pmatrix} \begin{pmatrix} \Delta z_j \\ \Delta x_j \\ -\Delta y_j \end{pmatrix} = - \begin{pmatrix} X_j z_j - \mu e \\ c - A^T y_j - z_j \\ A x_j - b \end{pmatrix}, \tag{12}$$

Only the entries in z smaller than ν_j are perturbed to create $\overline{Z}$, where $\{\nu_j\}$ is a sequence containing small positive numbers that converge to 0. We also modify the y so that $\eta_i \in S_Y$. Other parts of the algorithm are similar to standard interior point methods.

We mention that Algorithm 1 is similar to an algorithm recently proposed by Gill, Murray, Ponceleón, and Saunders [10]. Their algorithm allows the variables y and z to be chosen arbitrarily within two bounded sets. Our algorithm, in contrast, explicitly states the way we choose y and z and allows the matrix $\overline{Z}$ and the vector z to differ. This feature leads to a proof of superlinear convergence of the (inner) Newton iterations. Let

$$M(x,\rho) = c^T x - \mu \sum_{i=1}^{n} \ln \chi_i + \rho \|Ax - b\|_1, \tag{13}$$

where ρ is a positive number. Note that a barrier subproblem with a barrier parameter μ corresponding to linear program (1) is

$$\begin{array}{ll} \min & c^T x - \mu \sum_{i=1}^n \ln \chi_i \\ \text{s.t.} & Ax = b, \end{array} \tag{14}$$

and the inner loop of Algorithm 1 solves the barrier subproblems.

Algorithm 1 (The IPM with modified search directions).

Initialize $k \leftarrow 0,\ \mu_0 > 0,\ \Omega_Z > 0,\ \Lambda_Z \gg 0,\ \Lambda_Y \gg 0,$
  $x(\mu_0) > 0,\ y(\mu_0) \in S_Y,\ z(\mu_0) \in S_Z.$
if (the relative duality gap is large) then
  Set $j \leftarrow 0;\ x_0 = x(\mu_k)\ ;\ y_0 = y(\mu_k);\ z_0 = z(\mu_k).$
  until (converge to the optimality conditions (2))

Choose the perturbed vector $\overline{z}_j$.

if $(j = 0)$ then
  Choose ν_0 satisfying $\Omega_Z < \nu_0 \ll 1$.
else
  Choose ν_j such that $\nu_j < \nu_{j-1}$.
  Determine $\kappa_j > 0$.
  Perturb $\overline{z}_j$ such that
  $\begin{cases} \frac{\chi_i}{\overline{\zeta}_i} = \kappa_j \times (\text{previous } \hat{\Theta}_{ii} \text{ involving refactor.}), & \text{if } \zeta_i \le \nu_j, \\ \overline{\zeta}_i = \zeta_i, & \text{otherwise.} \end{cases}$
  Modify $\overline{z}_j$ such that $\begin{cases} \overline{\zeta}_i = \Omega_Z, & \text{if } \overline{\zeta}_i < \Omega_Z, \\ \overline{\zeta}_i = \Lambda_Z, & \text{if } \Lambda_Z < \overline{\zeta}_i. \end{cases}$
end if

Choose the vector $y_j \in S_Y$.
Choose y_j such that $\begin{cases} \eta_i = \eta_i, & \text{if } \lvert\eta_i\rvert \le \frac{\Lambda_Y}{m}, \\ \eta_i = \frac{\Lambda_Y}{m}, & \text{otherwise.} \end{cases}$

Update the vectors.
Determine Δx_j, Δy_j, and Δz_j by solving equation (12). Determine α_j and $x_{j+1} = x_j + \alpha_j \Delta x_j$ such that $M(x_{j+1}, \rho) < M(x_j, \rho)$. Compute $y_{j+1} = y_j + \Delta y_j$ and $z_{j+1} = z_j + \Delta z_j$. Set $j \leftarrow j + 1$.

  end until
  Set $x(\mu_{k+1}) = x_j,\ y(\mu_{k+1}) = y_j,\ z(\mu_{k+1}) = z_j$, and $k \leftarrow k + 1$.
  Choose a positive $\mu_k < \mu_{k-1}$.
end if

2.2. Convergence analysis

We prove the global convergence of Algorithm 1 and establish the rate of convergence in the (inner) Newton iterations. To prove the convergence of Algorithm 1, we adopt the procedure described by Gill et al. [10]. Lemmas and theorems are similar to the ones in [10]; some proofs are different [22]. Once we prove that an iterative method converges to the solution of (2) for a fixed μ, the global convergence of the interior point method, Algorithm 1, follows from the classical results by Fiacco and McCormick [5]. We first consider two lemmas to be used for proving the main convergence theorem.

Lemma 2. *Let $B(x) = c^T x - \mu \sum_{i=1}^{n} \ln \chi_i$ and $M(x,\rho) = B(x) + \rho\|Ax-b\|_1$, where ρ is a positive number. For any given positive numbers Λ_r and Λ_M, the level set $S = \{x \in \mathbb{R}^n \mid \|Ax-b\|_1 \le \Lambda_r, M(x,\rho) \le \Lambda_M\}$ is compact.*

We next observe that if $x \in S$, x is uniformly bounded away from zero due to the properties of B.

Lemma 3. *There exist uniform lower and upper bounds $\Omega_S, \Lambda_S > 0$, such that $\Omega_S e \le x \le \Lambda_S e$ for any $x \in S$.*

Now we show that, with mild restrictions on x, y, and z, a descent direction to the $M(x,\rho)$ may be determined by solving equation (12). Note that, in (12), the diagonal matrix $\overline{Z}$ may be different from the vector z in the right hand side.

Lemma 4. *Let $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, $diag(\overline{Z}) \in \mathbb{R}^n$, $r = (Ax-b) \in \mathbb{R}^m$, and Ω_Z, Λ_y, Λ_z, $\Lambda_r > 0$. Assume further that $x \in S$, $\|y\| < \Lambda_Y$, $\Omega_Z e < z < \Lambda_Z e$, and $\|r\| = \|Ax-b\|_1 < \Lambda_r$. If $\Delta x \in \mathbb{R}^n$ is the solution of equation (12) and ρ is large enough, Δx is a descent direction for $M(x,\rho)$ whenever $N^T(c-\mu X^{-1}e)$ or $r = Ax-b$ is not a zero vector, where the columns of N form a basis for the full space of A. Furthermore, Δx is a descent direction for $\|Ax-b\|_1$ whenever $r = Ax-b$ is nonzero.*

Proof. Eliminating Δz and then Δy from equation (12), we obtain the reduced 2×2 KKT system

$$\begin{pmatrix} \overline{Z}X^{-1} & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ -\Delta y \end{pmatrix} = -\begin{pmatrix} c - \mu x^{-1}e - A^T y \\ Ax - b \end{pmatrix}. \tag{15}$$

Therefore,

$$\overline{Z}X^{-1}\Delta x - A^T\Delta y = -g + A^T y, \tag{16}$$

where $g = c - \mu X^{-1}e$. Furthermore, the solution Δx is bounded from the assumptions.

Our goal is to show that the inner product of Δx and $\nabla_x M(x,\rho)$ is less than zero. Note that the assumption $x \in S$ implies that all components of x are uniformly bounded away from zero, and $\nabla_x M(x,\rho)$ is well defined for all $x > 0$ as follows:

$$\nabla_x M(x,\rho) = \nabla_x B(x,\mu) + \rho A^T \overline{e} = g + \rho A^T \overline{e},$$

where $B(x,\mu)$ is defined in Lemma 2 and the i-th component of $\overline{e}$ is either equal to 1 if the i-th component of r is non-negative, or -1, otherwise.

We first build equation (17), (18), and (19) to be used for computing the inner product. There exists $\Delta x_N \in \mathbb{R}^{n-m}$ and $\Delta x_A \in \mathbb{R}^m$ such that

$$\Delta x = N\Delta x_N + A^T \Delta x_A. \tag{17}$$

Multiplying N^T on both sides of equation (16) and using the fact that $AN = 0$,

$$N^T \overline{Z} X^{-1} \Delta x = -N^T g.$$

The decomposition (17) suggests that

$$N^T \overline{Z} X^{-1} N \Delta x_N = -N^T \overline{Z} X^{-1} A^T \Delta x_A - N^T g.$$

or

$$H_N \Delta x_N = -N^T (g + \overline{Z} X^{-1} A^T \Delta x_A), \tag{18}$$

where $H_N = N^T \overline{Z} X^{-1} N$ is a positive definite matrix with full rank. Furthermore, equation (15) and (17) imply that

$$-r = -(Ax - b) = A\Delta x = AA^T \Delta x_A,$$

or

$$\Delta x_A = -(AA^T)^{-1} r. \tag{19}$$

Now, by using equation (17), (18), (19), and the fact that $AN = 0$, we manipulate the product of Δx and $\nabla_x M(x, \rho)$ as follows.

$$
\begin{aligned}
&(\Delta x)^T \nabla_x M(x,\rho) \\
&= (\Delta x_N)^T N^T (g + \rho A^T \overline{e}) + (\Delta x_A)^T A (g + \rho A^T \overline{e}) \\
&= [-H_N^{-1} N^T (g + \overline{Z} X^{-1} A^T \Delta x_A)]^T N^T (g + \rho A^T \overline{e}) + (\Delta x_A)^T A (g + \rho A^T \overline{e}) \\
&= -(g + \overline{Z} X^{-1} A^T \Delta x_A)^T N H_N^{-T} N^T (g + \rho A^T \overline{e}) + (\Delta x_A)^T A (g + \rho A^T \overline{e}) \\
&= -g^T N H_N^{-T} N^T g - (\Delta x_A)^T A \overline{Z} X^{-1} N H_N^{-T} N^T g - g^T N H_N^{-T} N^T \rho A^T \overline{e} + \qquad (20) \\
&\quad (\Delta x_A)^T A \overline{Z} X^{-1} N H_N^{-T} N^T \rho A^T \overline{e} + (\Delta x_A)^T A g + \rho (\Delta x_A)^T A A^T \overline{e} \\
&= -g^T N H_N^{-1} N^T g + r^T (AA^T)^{-1} A \overline{Z} X^{-1} N H_N^{-1} N^T g - r^T (AA^T)^{-1} A g - \\
&\quad \rho r^T (AA^T)^{-1} A A^T \overline{e} \\
&= -g^T N H_N^{-1} N^T g - r^T u - \rho r^T \overline{e},
\end{aligned}
$$

where $u = (AA^T)^{-1} A (I - X^{-1} \overline{Z} N H_N^{-1} N^T) g$.

Now we give upper bounds for the last three terms in equation (20). For the first term, we use the fact that, if H_N^{-1} is symmetric, the magnitude of $g^T N H_N^{-1} N^T g$ is bounded by the largest and smallest eigenvalue of H_N^{-1} times $\|N^T g\|_2^2$ [24]. We therefore obtain

$$-g^T N H_N^{-1} N^T g \leq -c_1 \|N^T g\|_2^2,$$

where $c_1 = \frac{1}{\lambda_{max}}$, λ_{max} is the largest eigenvalue of H_N, and $\lambda_{max} > 0$ since H_N is positive definite.

Moreover, to bound the summation of the second and third terms, let

$$h(x) = r^T u = (Ax - b)^T (AA^T)^{-1} A (I - X^{-1} \overline{Z} N H_N^{-1} N^T) g$$

for any $x \in S$ and a fixed $\overline{Z}$. Since $h(x)$ is continuous and S is compact, there exists $x_{min} \in S$ minimizing $h(x)$ over $x \in S$. That is, $h(x_{min}) = \min\{h(x) \mid x \in S\}$. If $h(x_{min}) \geq r^T \overline{e} = \|r\|_1 \geq 0$, then

$$-r^T u - \rho r^T \overline{e} \leq -h(x_{min}) - \rho r^T \overline{e} \leq -\rho \|r\|_1.$$

Otherwise, if $h(x_{min}) < r^T \overline{e}$, then for any $\rho \geq 1 - \frac{h(x_{min})}{r^T \overline{e}} \geq 0$,

$$-r^T u - \rho r^T \overline{e} \leq -h(x_{min}) - \rho r^T \overline{e} \leq -\|r\|_1.$$

Thus, for a given $\rho \geq \max(1 - \frac{r^T u_{min}}{r^T \overline{e}}, 0)$, we may choose $c_2 = \min\{\rho, 0\}$, such that

$$-r^T u - \rho r^T \overline{e} \leq -c_2 \|r\|_1.$$

Therefore, we conclude that

$$\begin{aligned}
&(\Delta x)^T \nabla_x M(x,\rho) \\
&= -g^T N H_N^{-1} N^T g - r^T u - \rho r^T \overline{e}. \qquad (21)\\
&\leq -c_1 \|N^T g\|^2 - c_2 \|r\|_1 \leq 0,
\end{aligned}$$

where $c_1, c_2 > 0$.

This completes the proof that the product $(\Delta x)^T \nabla_x M(x,\rho)$ is strictly less than zero whenever $N^T g$ or r is not a zero vector.

To see that Δx is a decent direction for $\|Ax - b\|_1$, we simply compute the product of Δx and $\nabla_x(\|Ax - b\|_1)$ by using equation (17) and (19).

$$\begin{aligned}
&(\Delta x)^T \nabla_x(\|Ax - b\|_1) \\
&= ((\Delta x_N)^T N^T + (\Delta x_A)^T A)(A^T \overline{e}) \\
&= (\Delta x_A)^T (AA^T)\overline{e}) \qquad (22)\\
&= -r^T \overline{e} \\
&= -\|r\|_1.
\end{aligned}$$

It is thus straightforward that if $r \neq 0$, $(\Delta x)^T \nabla_x(\|Ax - b\|)$ is negative. ∎

Lemma 5. *If Δx is defined as in Lemma 4, then there exist positive numbers α and Λ_X such that the sufficient Goldstein-Armijo conditions [13] are satisfied. with $x + \alpha\Delta x > \Lambda_S e$.*

Proof. For simplicity, we adopt the notation $M(x)$ instead of $M(x,\rho)$ in this proof. Let α_M be the largest feasible step length along Δx; that is, $x + \alpha_M \Delta x \geq 0$ and some elements of $x + \alpha_M \Delta x$ equal zero. By the continuity of $M(x)$ and the fact that Δx is a decent direction for $M(x)$ (Lemma 4), for sufficiently small $\alpha > 0$, we have

$$M(x + \alpha\Delta x) < M(x) + \xi\alpha\Delta x^T \nabla_x M(x),$$

where $0 < \xi < 1$. Note that $M(x + \alpha\Delta x) \to \infty$ as $\alpha \to \alpha_M$, and $M(x) + \xi\alpha\Delta x^T \nabla_x M(x)$ decreases as α increases, and hence there exists $\hat{\alpha} < \alpha_M$ such that

$$M(x + \hat{\alpha}\Delta x) = M(x) + \xi\hat{\alpha}\Delta x^T \nabla_x M(x). \qquad (23)$$

That is, the inequality

$$M(x + \alpha\Delta x) - M(x) \leq \xi\alpha\Delta x^T \nabla_x M(x) \qquad (24)$$

is true for every $\alpha \in (0, \hat{\alpha}]$.

Moreover, $x + \alpha \Delta x > \Lambda_S e$ holds by Lemma 4 and definition of the set S. ∎

Theorem 6. *Given positive constants, Ω_Z, Λ_Y, and Λ_Z, let $\{\overline{Z}_j\}$ be a sequence of diagonal matrices with $0 < \Omega_Z e \leq diag(\overline{Z}_j) \leq \Lambda_Z e$, and let $\{y_j\}$ be a sequence of vectors satisfying $\|y_j\| \leq \Lambda_Y$. Assume $\{x_j\}$ is a sequence generated by $x_{j+1} = x_j + \alpha_j \Delta x_j$ and $x_0 > 0$, where Δx_j is defined by (12) and α_j satisfies the sufficient Goldstein-Armijo conditions on $M(x_j, \rho)$ with $x_j > 0$. If ρ is large enough, $\lim_{j \to \infty} x_j = x^*(\mu)$.*

Proof. We first choose $x_0 \in \mathbb{R}$ and set $\Lambda_r = \|Ax_0 - b\|$ and $\Lambda_M = M(x_0, \rho)$. Then $S = \{x \mid \|Ax - b\| \leq \Lambda_r \text{ and } M(x, \rho) \leq \Lambda_M\}$. Lemma 5 shows a stepsize may be found to decrease $M(x_j, \mu)$ and $\|Ax_j - b\|$. The sequence $\{x_j\}$ generated by the iteration $x_{j+1} = x_j + \alpha_j \Delta x_j$ thus belongs to S. Since $M(x, \rho)$ is continuous and S is compact, $M(x, \rho)$ is bounded below over the set $\{x \mid x \in S\}$. Lemma 4 also shows that $\{M(x_j, \rho)\}$ is monotonically decreasing. The fact that $\{M(x_j, \rho)\}$ is bounded below and monotonically decreasing implies the sequence $\{M(x_j, \rho)\}$ converges. That is, $\lim_{i \to \infty}(M(x_{j+1}, \rho) - M(x_j, \rho)) = 0$.

On the other hand, Lemma 4 and equation (24) give

$$\begin{aligned} -(M(x_j + \alpha_j \Delta x_j) - M(x_j)) &\geq -\xi \alpha_j \Delta x_j^T \nabla_x M(x_j) \\ &\geq c_1 \|N^T g(x_j)\|^2 + c_2 \|r(x_j)\|_1 \\ &\geq 0. \end{aligned}$$

Therefore,

$$\lim_{i \to \infty} (\alpha_j \Delta x_j^T \nabla_x M(x_j, \rho)) = \lim_{i \to \infty} (c_1 \|N^T g(x_j)\|^2 + c_2 \|r(x_j)\|_1) = 0,$$

or

$$\|N^T g(x_j)\| \to 0 \text{ and } \|r(x_j)\|_1 \to 0.$$

The fact that problem (14) is a convex problem for a given μ suggests that the solution of (2), $x^*(\mu)$, is unique. Finally by the continuity of $N^T g(x)$ and $r(x)$ we conclude that $x_j \to x^*(\mu)$. ∎

Finally, we show that $\{y_j\} \to y^*(\mu)$ and $\{z_j\} \to z^*(\mu)$ from Theorem 6.

Corollary 7. *Under the assumptions of Theorem 6, $\lim_{i \to \infty} y_j + \Delta y_j = y^*(\mu)$ and $\lim_{i \to \infty} z_j + \Delta z_j = z^*(\mu)$, where $y^*(\mu)$ and $z^*(\mu)$ are the solution of equation (2).*

Proof. Expand (12) and then combine the first two equations to obtain

$$A^T(y + \Delta y) = c - \mu X^{-1} e + \overline{Z} X^{-1} \Delta x, \tag{25}$$

or

$$AX\overline{Z}^{-1}A^T(y + \Delta y) = AX\overline{Z}^{-1}(c - \mu X^{-1}e) + A\Delta x. \qquad (26)$$

Note that the vector z, which may different from $\overline{Z}$, has been canceled. Theorem 6 and equation (2) imply that $c - \mu X^{-1}e$ converges to $A^T y^*(\mu)$ and $A\Delta x = b - Ax$ converges to 0. Therefore, since $AX\overline{Z}^{-1}$ is symmetric positive definite, $\lim_{j\to\infty} y_j + \Delta y_j = y^*(\mu)$.

Similarly, equation (12) implies

$$X_j \Delta z_j + \overline{Z}_j \Delta x_j = -X_j z_j + \mu e,$$

or

$$z_j + \Delta z_j = X_j^{-1}\overline{Z}_j \Delta x_j + \mu X_j^{-1} e.$$

By equation (2), the fact that x_j is bounded above zero, and $\{x_j\} \to x^*(\mu)$ (or $\{\Delta x_j\} \to 0$),

$$\lim_{i\to\infty} (z_j + \Delta z_j) = z^*(\mu).$$

∎

We have proved the convergence of the minor iteration in Algorithm 1 for arbitrary choice of $y \in S_Y$, $diag(\overline{Z}) \in S_Z$, and $z \in S_Z$. In the next section, we focus on implementation issues of the modified search directions algorithm. We next show a superlinear convergence rate for the minor iteration.

2.3. The rate of convergence

Assuming primal feasibility, (i.e. the starting point satisfies $Ax = b$), we establish the superlinear convergence rate of the iteration in the inner loop of Algorithm 1.

We first show that the full "Newton" step may be taken when x_j is close to the optima x^* by showing that the following lemma is applicable.

Lemma 8 (Dennis and Moré [4]). *Let $f : \mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable in an open set D and consider iteration $x_{j+1} = x_j + \alpha_j p_j$, where $j = 0, 1, \cdots$, $\nabla f(x_j)^T p_j < 0$, and α_j is chosen to satisfy the Goldstein-Armijo conditions. If $\{x_j\}$ converges to a point x^* in D at which $\nabla^2 f(x^*)$ is positive definite and*

$$\lim_{j\to\infty} \frac{\|\nabla f(x_j) + \nabla^2 f(x_j) p_j\|}{\|p_j\|} = 0, \qquad (27)$$

then there is an index $j_0 \geq 0$ such that $\alpha_j = 1$ is admissible for $j \geq j_0$.

Note that the barrier problem of the model linear problem (1) is described in (14). Let $P = I - A^T(AA^T)^{-1}A$ be a projection to the null space of the matrix A, $w_j \in \mathbb{R}^n$,

$diag(Pw_j)$ be a $n \times n$ diagonal matrix containing the n-vector Pw_j in its main diagonal, and

$$x_j = \bar{x} + Pw_j, \tag{28}$$

where $\bar{x} = (\bar{\chi}_1, \cdots, \bar{\chi}_n)^T$ satisfies $A\bar{x} = b$. The barrier problem (14) thus is equivalent to the following unconstrained problem

$$\min_w c^T Pw - \mu \sum_{i=1}^n \ln(\bar{\chi}_i + P_i w),$$

where P_i is the i-th row of the matrix P. We claim that the search direction Δx_j defined in (12) is a descent direction for the function

$$\overline{B}(w) = c^T Pw - \mu \sum_{i=1}^n \ln(\bar{\chi}_i + P_i w). \tag{29}$$

The gradient and the Hessian matrix of $\overline{B}(w)$ are

$$\nabla \overline{B} = P^T c - \mu P^T (\bar{X} + diag(Pw_j))^{-1} e$$

and

$$\nabla^2 \overline{B} = \mu P^T (\bar{X} + diag(Pw_j))^{-2} P,$$

respectively. Since we assume $Ax_0 = b$, $A\Delta x_j = 0$ for all j, or $P\Delta x_j = \Delta x_j$. We compute

$$\begin{aligned} (\nabla \overline{B})^T \Delta x_j &= (c^T P - \mu e^T (\bar{X} + diag(Pw_j))^{-1} P) \Delta x_j \\ &= (c^T P - \mu e^T (X_j)^{-1}) \Delta x_j. \end{aligned} \tag{30}$$

By the primal feasibility assumption and an argument similar to that in the proof of Lemma 4, we conclude that

$$(\nabla_x B)^T \Delta x_j = (c^T P - \mu e^T (X_j)^{-1}) \Delta x_j < 0. \tag{31}$$

Thus Δx_j is a descent direction for $\overline{B}(w)$.

In short, starting from any $w_0 \in \mathbb{R}^n$, the iteration

$$w_{j+1} = w_j + \alpha_j \Delta x_j \tag{32}$$

may induce the iteration

$$x_{j+1} = x_j + \alpha_j \Delta x_j,$$

since $P\Delta x_j = \Delta x_j$.

We now show that the condition corresponding to equation (27) is satisfied in our problem.

Lemma 9. *Let $\overline{B}(w)$ be defined as equation (29) and Δx_j be the solution of the system (12) with the assumption that $Ax_j = b$. Then*

$$\lim_{j\to\infty} \frac{\|\nabla \overline{B}(w_j) + \nabla^2 \overline{B}(w_j)\cdot \Delta x_j\|}{\|\Delta x_j\|} = 0. \tag{33}$$

Proof. Using equations (16), (28), and the facts that $P\Delta x_j = \Delta x_j$ and $AP = 0$, we compute

$$\begin{aligned} & \nabla \overline{B} + \nabla^2 \overline{B}(w_j)\cdot \Delta x_j \\ = \;& P^T c - \mu P^T(\bar{X} + diag(Pw_j))^{-1}e + \mu P^T(\bar{X} + diag(Pw_j))^{-2}\Delta x_j \\ = \;& P^T c - \mu P^T X_j^{-1} e + \mu P^T X_j^{-1}\overline{Z}_j^{-1}(A^T(y_j + \Delta y_j) - c + \mu X_j^{-1}e) \\ = \;& P^T(I_j - I)(A^T(y_j + \Delta y_j) - c + \mu X_j^{-1}e) + P^T A^T(y_j + \Delta y_j) \\ = \;& P^T(I_j - I)\Delta x_j, \end{aligned} \tag{34}$$

where $I_j = \mu X_j^{-1}\overline{Z}_j^{-1}$.

Equation (34) implies that

$$\frac{\|\nabla \overline{B}(w_j) + \nabla^2 \overline{B}(w_j)\cdot \Delta x_j\|}{\|\Delta x_j\|} = \frac{\|P^T(I_j - I)\Delta x_j\|}{\|\Delta x_j\|} \le \|P^T\|\cdot\|(I_j - I)\|.$$

Theorem 6 and Corollary 7 show that X_j and $\overline{Z}_j$ converge to the solution of the optimality condition (2), and therefore $\{I_j\} \to I$. This completes the proof. ∎

Lemma 8 and 9 imply that full Newton step may be taken when $\{x_j\}$ approaches x^*. A classical result by Dennis and Moré [3] shown in Lemma 10 leads to Theorem 11 establishing the convergence rate.

Lemma 10 (Dennis and Moré [3]). *Let $F : \mathbb{R}^p \to \mathbb{R}^p$ be differentiable in the open convex set D in $\mathbb{R}^p$, and assume that for some s^* in D, F' is continuous at s^* and $F'(s^*)$ is nonsingular. Let $\{J_j\}$ in $L(\mathbb{R}^p)$ be a sequence of nonsingular matrices and suppose that for some s_0 in D the sequence $\{s_j\}$ where*

$$s_{j+1} = s_j - J_j^{-1}F(s_j), \tag{35}$$

remains in D and converges to s^*. Then $\{s_j\}$ converges Q-superlinearly to s^* and $F(s^*) = 0$ if and only if

$$\lim_{j\to\infty} \frac{\|[J_j - F'(s^*)](s_{j+1} - s_j)\|}{\|s_{j+1} - s_j\|} = 0. \tag{36}$$

Theorem 11. *Under the assumptions of Theorem 6, assume that the sequence* $\{(z_j, x_j, y_j)^T\}$ *is generated by the inner loop of Algorithm 1. The sequence* $\{(z_j, x_j, y_j)^T\}$ *converges to* $\{(z^*(\mu), x^*(\mu), y^*(\mu))^T\}$ *Q-superlinearly.*

Proof. Our main goal is to show that (36) holds for our problem. Let

$$J_j = \begin{pmatrix} X_j & \overline{Z}_j & \\ -I & & A^T \\ & A & \end{pmatrix} \begin{pmatrix} I & & \\ & \alpha_j^{-1} I & \\ & & -I \end{pmatrix} = \begin{pmatrix} X_j & \alpha_j^{-1}\overline{Z}_j & \\ -I & & -A^T \\ & \alpha_j^{-1} A & \end{pmatrix},$$

where α_j is the step length for Δx_j, and let

$$F(z, x, y) = \begin{pmatrix} Xz - \mu e \\ c - A^T y - z \\ Ax - b \end{pmatrix}.$$

The Jacobian matrix of $F(z, x, y)$ is

$$F'(z, x, y) = \begin{pmatrix} X & Z & \\ -I & & -A^T \\ & A & \end{pmatrix}.$$

Let F'_j and $(F^*)'$ denote $F'(z_j, x_j, y_j)$ and $F'(z^*(\mu), x^*(\mu), y^*(\mu))$, respectively. The iteration in the inner loop of Algorithm 1 may be written as

$$\begin{pmatrix} z_{j+1} \\ x_{j+1} \\ y_{j+1} \end{pmatrix} = \begin{pmatrix} z_j \\ x_j \\ y_j \end{pmatrix} + \begin{pmatrix} \Delta z_j \\ \alpha_j \Delta x_j \\ \Delta y_j \end{pmatrix} = \begin{pmatrix} z_j \\ x_j \\ y_j \end{pmatrix} - (J_j)^{-1} F'_j. \tag{37}$$

Using the facts that $\{X_j\} \to X^*(\mu)$, $\{\overline{Z}_j\} \to Z^*(\mu)$, and $\alpha_j = 1$ satisfies the sufficient Goldstein-Armijo conditions for j sufficiently large, we can conclude that

$$\|J_j - (F^*)'\| = \| \begin{pmatrix} X_j - X^*(\mu) & \alpha_j^{-1}\overline{Z}_j - Z^*(\mu) & 0 \\ 0 & 0 & 0 \\ 0 & (\alpha_j^{-1} - 1)A & 0 \end{pmatrix} \|$$

approaches zero as j goes to infinity. That is,

$$\lim_{j\to\infty} \frac{\|(J_j - (F^*)') \cdot (s_{j+1} - s_j)\|_1}{\|(s_{j+1} - s_j)\|_1} = 0. \tag{38}$$

holds for $s_j = (z_j, x_j, y_j)^T$. The Q-superlinear convergence thus follows from Lemma 10. Note that we take advantage of the fact that the vector z in $F(x, y, z)$ may be different from the perturbed matrix $\overline{Z}_j$ in J_j; otherwise, equation (37) would not hold. ∎

3. Implementation

We now discuss a practical way to implement the idea of using a perturbed system for determining a modified search direction. Key differences are that, for each μ, the theoretical algorithm (Algorithm 1) whose convergence is proved solves the KKT system exactly, while in practice we solve the normal equations approximately. Furthermore, in practice, only one step of Newton's method is applied for each fixed μ. We perturb the diagonal matrix Θ, rather than the variable z's, so that the large entries in the current Θ are proportional to the corresponding entries of $\hat{\Theta}$.

Eliminating Δx and Δz from the perturbed KKT system (12), we obtain the normal equations corresponding to the perturbed system

$$(A\overline{\Theta}A^T)\Delta y = A\overline{\Theta}(r_d + Ze - \mu X^{-1}e) + r_p. \tag{39}$$

The search directions are

$$\Delta x = \overline{\Theta}(A^T\Delta y - r_d - Ze + \mu X^{-1}e) \tag{40}$$

and

$$\Delta z = r_d - A^T\Delta y. \tag{41}$$

We now focus on how we modify the current diagonal matrix Θ. Suppose that we have a Cholesky factorization of $A\hat{\Theta}A^T = LPL^T$. Recall that our goal is to determine the index set B_1 and the proportionality factor κ such that the corresponding $\Theta^{B_1} = \kappa\hat{\Theta}^{B_1}$. See equation (7) for the definition of Θ^{B_1}. After the modified matrix $\overline{\Theta}$ has been determined, we update the preconditioner using $\overline{\Theta}$ and then use the preconditioned conjugate gradient method to solve the normal equations involving $A\overline{\Theta}A^T$.

Algorithm 12 (Θ modified algorithm).

```
Initialize B_1 ← ∅ and γ ← 0 ∈ ℝ^n.
if ( relative gap is small ) then
    do (i = 1 : n)
        if [(ζ_i is small) and (Θ_ii is large) ] then
            Compute γ_i = Θ̂_ii/Θ_ii,
            Set B_1 = B_1 ∪ {i},
        else
            set γ_i = 0.
        end if
    end do
    Compute mean and variance of the nonzero γ_i's.
    if (the variance is small) then
        Set κ = the mean.
    else
        Set B_1 = B_1 \ {the indices of some largest and smallest γ_i's }
        Compute mean of the γ_i's corresponding to the new B_1.
        Set κ = the mean.
    end if
    do (i = 1 : n)
        if [ (i ∈ B_1) and (γ_i is close to κ)] then
            Compute Θ̄_ii = κ × Θ̂_ii
        else
            Set Θ̄_ii = Θ_ii
        end if
    end do
else
    Set Θ̄ = Θ
end if
```

Algorithm 12 explains how we determine the index set B_1 as well the proportionality factor κ. The algorithm first sets the index set B_1 containing all the large entries of the current Θ. We then find the ratios of $\hat{\Theta}_{ii}$ to Θ_{ii}, for every $i \in B_1$. The mean and the variance of those ratios are computed and κ is assigned as the mean value. If the variance value is small, we calculate

$$\overline{\Theta}_{ii} = \kappa \Theta_{ii},$$

for $i \in B_1$. Otherwise, we take out the indices corresponding to some largest and smallest ratios from B_1 to form a new B_1. The mean value of the ratios corresponding to the new B_1 is re-computed to obtain a new κ and then the current Θ is perturbed using the new κ.

We present a theorem showing that the difference between the original search direction Δy and the modified search direction $\Delta\overline{y}$ approaches zero if the difference between Z and $\overline{Z}$ goes to zero. First, we introduce a lemma, by Stewart [19] and Todd [20] independently, that will be used in the proof of the theorem.

Lemma 13. (Stewart [19]) *Let D_+ be the set of all $n \times n$ diagonal matrices with positive diagonal elements. Let A be of full row rank. Then there exist two finite positive numbers $\Lambda_1(A)$ and $\Lambda_2(A)$ such that, for any $D \in D_+$,*

$$\begin{aligned} \sup_{D \in D_+} \|(ADA^T)^{-1}AD\| &\leq \Lambda_1(A) \\ \sup_{D \in D_+} \|A^T(ADA^T)^{-1}AD\| &\leq \Lambda_2(A). \end{aligned}$$

Theorem 14. *Let Δy and $\Delta\overline{y}$ be computed by equations (4) and (39), respectively. Then we conclude that, for a positive diagonal matrix Θ, $\|\Delta y - \Delta\overline{y}\|$ approaches zero as $\|Z - \overline{Z}\|$ goes to zero.*

Proof. Using the formula

$$M^{-1} = N^{-1} - M^{-1}(M - N)N^{-1}, \tag{42}$$

we first rewrite

$$(A\overline{\Theta}A^T)^{-1} = (A\Theta A^T)^{-1} - (A\overline{\Theta}A^T)^{-1}(A\Delta\Theta A^T)(A\Theta A^T)^{-1}, \tag{43}$$

where M and N are two arbitrary invertible matrices and $\Delta\Theta = \overline{\Theta} - \Theta$.

Thus by the definition of Δy and $\Delta\overline{y}$, equation (43), and Lemma 13, the upper bound of $\|\Delta y - \Delta\overline{y}\|$ may be computed as follows.

$$
\begin{aligned}
& \|\Delta y - \Delta \overline{y}\| \\
= \;& \|[(A\Theta A^T)^{-1}A\Theta - (A\overline{\Theta}A^T)^{-1}A\overline{\Theta}]\psi(\mu)\| \\
\leq \;& \|(A\Theta A^T)^{-1}A\Theta - (A\overline{\Theta}A^T)^{-1}A\Theta + (A\overline{\Theta}A^T)^{-1}A\Theta - (A\overline{\Theta}A^T)^{-1}A\overline{\Theta}\| \; \|\psi(\mu)\| \\
= \;& \|(A\Theta A^T)^{-1}A\Theta - (A\Theta A^T)^{-1}A\Theta + \\
& (A\overline{\Theta}A^T)^{-1}(A\Delta\Theta A^T)(A\Theta A^T)^{-1}A\Theta - (A\overline{\Theta}A^T)^{-1}A(\Delta\Theta)\|\|\psi(\mu)\| \\
= \;& \|(A\overline{\Theta}A^T)^{-1}A\Delta\Theta(A^T(A\Theta A^T)^{-1}A\Theta - I)\| \; \|\psi(\mu)\| \\
\leq \;& \|(A\overline{\Theta}A^T)^{-1}A\overline{\Theta}\| \; \|\overline{\Theta}^{-1}\Delta\Theta\| \; \|A^T(A\Theta A^T)^{-1}A\Theta - I\| \; \|\psi(\mu)\| \\
\leq \;& \Lambda_1(A)\,(\Lambda_2(A)+1)\,\|\psi(\mu)\|\,\|\overline{\Theta}^{-1}(\overline{\Theta}-\Theta)\|,
\end{aligned}
\tag{44}
$$

where $\Lambda_1(A)$ and $\Lambda_2(A)$ are constants arising in Lemma 13 and $\psi(\mu) = r_d + Ze - \mu X^{-1}e$, the right hand side of the normal equations (4). By the definitions of $\overline{\Theta}$ and Θ, the i-th element of the diagonal matrix $\overline{\Theta}^{-1}(\overline{\Theta} - \Theta)$ is

$$
\frac{\zeta_i - \overline{\zeta}_i}{\zeta_i}. \tag{45}
$$

Equation (45) also implies that if $\overline{Z}$ goes to Z, $\|\overline{\Theta}^{-1}(\overline{\Theta} - \Theta)\|$ approaches zero. Furthermore, since $\Lambda_1(A)$, $\Lambda_2(A)$, and $\|\psi(\mu)\|$ are bounded constants, the right hand side of (44) converges to zero provided $\|\overline{Z} - Z\| \to 0$. We thus complete the proof. ∎

A similar result is derived for $\|\Delta x - \Delta \overline{x}\|$.

Theorem 15. *Let Δx and $\Delta \overline{x}$ be defined in equations (51) and (40), respectively. Then $\|\Delta x - \Delta \overline{x}\| \to 0$ if $\|Z - \overline{Z}\| \to 0$.*

Proof. By Lemma 13, Theorem 14, and the definition of Δx, $\Delta \overline{x}$, Δy, and $\Delta \overline{y}$,

$$
\begin{aligned}
& \|\Delta x - \Delta \overline{x}\| \\
= \;& \|\Theta A^T\Delta y - \overline{\Theta}A^T\Delta\overline{y} + \Theta\psi(\mu) - \overline{\Theta}\psi(\mu)\| \\
\leq \;& \|\Theta A^T\Delta y - \Theta A^T\Delta\overline{y}\| + \|\Theta A^T\Delta\overline{y} - \overline{\Theta}A^T\Delta\overline{y}\| + \|\Theta\psi(\mu) - \overline{\Theta}\psi(\mu)\| \\
\leq \;& \|\Theta A^T\| \; \|\Delta y - \Delta\overline{y}\| + \|\Theta - \overline{\Theta}\| \; \|\psi(\mu)\| \, \Lambda_2 + \|\Theta - \overline{\Theta}\| \; \|\psi(\mu)\| \\
\leq \;& \|\Theta A^T\| \, \Lambda_1(A)\,(\Lambda_2(A)+1)\,\|\psi(\mu)\|\,\|\overline{\Theta}^{-1}(\overline{\Theta}-\Theta)\| + \\
& (1+\Lambda_2)\,\|\psi(\mu)\|\,\|\Theta - \overline{\Theta}\|,
\end{aligned}
\tag{46}
$$

where $\Lambda_1(A)$ and $\Lambda_2(A)$ are constants defined in Lemma 13 and $\psi(\mu) = r_d + Ze - \mu X^{-1}e$. An argument similar to that at the end of Theorem 14 completes the proof. ∎

The ideas discussed above are implemented by modifying OB1-R, which considers linear programming problems with simple upper bounds

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Ax = b, \\ & x + s = u, \\ & x \geq 0, \\ & s \geq 0, \end{array} \tag{47}$$

where $u \in \mathbb{R}^n$ contains upper bounds for the entries of x and some of them may be infinite. Since the problem considered by OB1-R is slightly different from our model problem (1), we elaborate the corresponding equations for clarity. The optimality conditions corresponding to the linear programming problem (47) is

$$\left\{ \begin{array}{lcl} Ax - b & = & 0, \\ x + s - u & = & 0, \\ A^T y - w + z - c & = & 0, \\ XZe - \mu e & = & 0, \\ SWe - \mu e & = & 0, \end{array} \right. \tag{48}$$

where y and w are the Lagrangian multipliers. The Newton search directions may be determined by the block 5×5 matrix

$$\begin{pmatrix} A & & & & \\ I & I & & & \\ & & A^T & -I & I \\ Z & & & & X \\ & W & & & S \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta s \\ \Delta y \\ \Delta w \\ \Delta z \end{pmatrix} = \begin{pmatrix} Ax - b \\ x + s - u \\ A^T y - w + z - c \\ XZe - \mu e \\ SWe - \mu e \end{pmatrix}, \tag{49}$$

or the equations

$$\Delta y = (A\Theta A^T)^{-1}(A\Theta(\widetilde{\psi}(\mu) + r_d) + (b - Ax)), \tag{50}$$

$$\Delta x = \Theta(A^T \Delta y - (\widetilde{\psi}(\mu) + r_d)), \tag{51}$$

$$\Delta w = \mu S^{-1}e - We - S^{-1}W\Delta s, \tag{52}$$

$$\Delta z = \mu X^{-1}e - Ze - X^{-1}Z\Delta x, \tag{53}$$

$$\Delta s = x + s - u - \Delta x, \tag{54}$$

(55)

where $\widetilde{\psi}(\mu) = \mu(S^{-1} + X^{-1})e - (W - Z)e$, and $\Theta^{-1} = (S^{-1}W + X^{-1}Z)$.

The modified search directions, by perturbing the diagonal matrix Z to $\overline{Z}$, may be written as

$$\begin{aligned}
\Delta y &= (A\overline{\Theta}A^T)^{-1}(A\overline{\Theta}(\widetilde{\psi}(\mu) + r_d) + (b - Ax)), && (56)\\
\Delta x &= \overline{\Theta}(A^T\Delta y - (\widetilde{\psi}(\mu) + r_d)), && (57)\\
\Delta w &= \mu S^{-1}e - We - S^{-1}\overline{W}\Delta s, && (58)\\
\Delta z &= \mu X^{-1}e - Ze - X^{-1}\overline{Z}\Delta x, && (59)\\
\Delta s &= x + s - u - \Delta x, && (60)\\
& && (61)
\end{aligned}$$

where $\overline{\Theta}^{-1} = (S^{-1}W + X^{-1}\overline{Z})$ and Δw and Δs are same as (52) and (54), respectively. All the discussion may be easily extended to the problems (47). However, it is worth mentioning that the perturbed matrix $\overline{Z}$ is needed for determining Δz and may be computed by solving

$$\overline{\Theta}^{-1} = (S^{-1}W + X^{-1}\overline{Z}),$$

after $\overline{\Theta}$ has been determined.

We now present computational results of some test problems to show the modified search direction may improve performance in the endgame. In other words, in the endgame, we perform Algorithm 12 to determine the perturbed matrix $\overline{\Theta}$ and then solve equations (56) to (60) to find the search directions. Table 1 illustrates the performance of the preconditioned conjugate gradient solver in the last μ values. The original and the perturbed normal equations are solved for the problems `pilot` and `pilot87` from the NETLIB collection [8]. The number of preconditioned conjugate gradient iterations and the time for forming and solving the normal equations (in seconds) are compared for both approaches. The preconditioned conjugate gradient solvers use the same stopping criterion. The artificial variables are kept in `pilot87`. Complete Cholesky factorization is performed to determine the preconditioners at the 73-th and 77-th iterations in `pilot` and at the 75-th, 79-th, and 83-th iterations in `pilot87`. All other iterations use updated preconditioners. From the table, we observe that, by perturbing the normal equations, we may improve both the preconditioned conjugate gradient iteration numbers and timing. Furthermore, the number of outer iterations remains the same for using OB1-R.

4. Combined algorithm

We now present Algorithm 18, the algorithm combining all the ideas discussed above to solve the (perturbed) normal equations using preconditioned conjugate gradients. To factor an ill-conditioned matrix, we use a variety of the standard Cholesky factorization shown in Algorithm 16.

Problem : pilot						
		Outer iteration				
Prtrb.		73	74	75	76	77
Yes	PCG iter.	3	26	38	40	3
	Time (s)	6.86	3.34	4.71	4.96	6.89
No	PCG iter.	3	31	45	57	3
	Time (s)	6.71	3.81	5.38	6.73	6.75

Problem : pilot87										
		Outer iteration								
Prtrb.		75	76	77	78	79	80	81	82	83
Yes	PCG iter.	1	32	50	49	1	22	25	33	1
	Time (s)	22.50	8.51	14.68	17.58	22.50	6.04	6.28	8.26	22.50
No	PCG iter.	1	37	72	100	1	35	43	54	1
	Time (s)	22.51	9.68	20.34	35.00	22.61	9.23	10.80	13.36	22.54

Table 1: Comparison for solving the normal equations with or without perturbing.

Algorithm 16 (A hybrid modified Cholesky factorization).

Initialize `pii_tiny` $\leftarrow$ a positive tiny number.
do $(i = 1 : m)$
    Determine P_{ii} by the modified Cholesky factorization in [9].
    if $(P_{ii} \leq$ `pii_tiny`$)$
        Set $P_{ii} = 0$.
        Skip computation of the i-th column of L.
    else
        Determine the i-th column of L using P_{ii}.
    end if
end do

The preconditioned conjugate gradient solver is used through the whole interior point method, except for the first μ, with one exception. If we factor the matrix $A\Theta A^T$ and preconditioned conjugate gradients converges in more than, for example, 50 iterations, even if the hybrid modified Cholesky factorization is used, we switch to a direct method in the next μ iteration. This situation occurs in the case that the matrix $A\Theta A^T$ is too ill-conditioned to make the refactored Cholesky factors an efficient preconditioner. This is an unusual occurrence: only one problem (`dfl001`) met the criterion among all the problems we tested using all the default parameters; but we include the criterion as a "safe guard" for efficiency.

If the ratio of the last barrier parameter to the current barrier parameter is large near the endgame, we refactor the matrix $A\Theta A^T$ to obtain the preconditioner for the current iteration. Since the barrier parameter is proportional to the duality gap, a large change in the two successive barrier parameters implies a large change in the corresponding duality gaps. In this case, the iterates made a "big" improvement and thus the variables and the current resulting matrix Θ may change widely. The update strategy is thus not suitable.

Algorithm 17 describes the interior point algorithm solving the normal equations by the combined preconditioned conjugate gradient solver, Algorithm 18, from the second μ through the entire course.

Algorithm 17 (Interior point algorithm with adaptive solver).

Initialize $k \leftarrow 1$; $\mu_0 > 0$; $x_0, y_0, z_0 > 0$; `UseDirect` $\leftarrow$ False.

while (not convergent)

  if [($k > 1$) and (`UseDirect` = False)] then

Solve using PCG. (See Algorithm 18 for details.)
Determine the preconditioner. Iterate the PCG method using Algorithm 18.

  end if

  if [($k = 1$) or (`UseDirect` = True)] then

Solve using direct solver.
Form the matrix $A\Theta A^T$. Factor $A\Theta A^T = LPL^T$. Solve the normal equations using LPL^T,   applying iterative refinement if necessary. Compute `drct_cost` as the elapsed time of the direct solver.

  end if

Update the primal and dual variables.
Compute $x_{k+1} \leftarrow x_k + \alpha_p \Delta x$; $y_{k+1} \leftarrow y_k + \alpha_d \Delta y$; $z_{k+1} \leftarrow z_k + \alpha_d \Delta z$.

  Choose $\mu_{k+1} < \mu_k$.

  Set $k \leftarrow k + 1$.

end while

Algorithm 18 (The modified PCG solver).

Solve using PCG.

Determine the preconditioner.

```
updt_nmbr ← the number of rank-one updates to be performed.
pred_cost ← predicted cost of iterative solution.
if [ (prev_cost > .8 × drct_cost) or (pred_cost > drct_cost) or
   (the ratio of μ_{k-1} to μ_k is large near endgame) ] then
    UseRefact ← True
    Form the matrix AΘA^T.
    if [UseStdChol = True] then
        Factor AΘA^T using the standard Cholesky algorithm.
        Factor AΘA^T using the hybrid Cholesky algorithm (Algorithm 16).
    end if
else
    UseRefact ← False
    Compute the modified diagonal matrix Θ̄ from Algorithm 12.
    Perform updt_nmbr rank-one updates based on AΘ̄A^T.
end if
```

Iterate the PCG method.

```
pcg_itn ← 0
until ( convergent)
    Execute a PCG iteration using the modified AΘ̄A^T.
    pcg_itn ← (pcg_itn + 1)
    if (pcg_itn > max_pcg_itn) then
        if (this happened for the previous μ) then ( UseDirect ← True )
        UseRefact ← True
        Factor (AΘA^T) to reinitialize the preconditioner.
        Restart the PCG iteration.
    end if
end until
if [(UseRefact = True) and (pcg_itn ≥ 5)] then UseStdChol ← False
else if [(UseRefact = True) and (pcg_itn ≥ 50)] then UseDirect ← True
end if
```

5. Numerical results

We modify OB1-R, by Lustig, Marsten, and Shanno [15] and dated December 1989, to implement Algorithm 17 and 18 (Adap2). We choose some problems tested in [23] to make comparisons with the computational results of OB1-R and the adaptive algorithm (Adap1) reported in [23].

All the algorithms are coded in FORTRAN using double precision arithmetic. The codes are compiled on a SUN SPARCstation 20 containing 64 megabytes main memory and running SunOS Release 4.1.3. Optimization level `-O3` is turned on for compiling the programs. Numerical experiments are performed on the same platform. The timings reported are CPU time in seconds. Since all three codes use the same preprocessor HPREP, we omit the preprocessing time. All the statistical tables in the section are extracted from [23].

5.1. The NETLIB problems

We first present our numerical results on the NETLIB problem collection [8], a standard linear programming test problem set. Small problems have a relatively small cost for forming and factoring the coefficient matrix in the normal equations, as mentioned in [7] and [23], so we do not expect that a interior point algorithm based on iterative solvers may prevail over a direct solver based algorithm. We consequently run a few small problems from the NETLIB collection, but concentrate on the larger problems, those containing more than $25,000$ nonzero entries in the coefficient matrix A. The problem `fit2p`, however, is neglected since all three codes fail to solve the problem on our workstation in a reasonable time, since the problem contains a large dense matrix $A\Theta A^T$.

Table 2 shows the characteristics of the tested NETLIB problems. The numbers of rows, columns, and nonzeros of coefficient matrix A are reported. The numbers are obtained from output of the preprocessor HPREP and may not be identical with the data in [8]. Only the nonzero elements in the lower sub-diagonal part of AA^T and L are counted and tabulated. We calculate the density of the matrix AA^T and L by computing the ratio of the number of nonzeros to the number of the entries in the lower sub-diagonal parts of the matrices.

Numerical results of OB1-R and the code Adap2 on the NETLIB problems are shown in Table 3. The table indicates the name of the problem and compares the number of μ values needed by the interior point methods for both codes, final relative duality gaps, and CPU time used by the both codes in seconds. The time differences between the two programs are shown in the last column. Our Adap2 codes are faster for the problems with positive time difference.

Both OB1-R and Adap2 take the same number of μ numbers to achieve similar small relative duality gaps, except on the problem `pilot87` and `greenbea`. Adap2 takes one addition μ value in `pilot87`, achieves a slightly smaller relative duality gap, and uses less time. In the problem `greenbea`, both algorithms stop unsuccessfully since they fail to converge with a small duality gaps. The problem, as mentioned in [21], is difficult to solve by interior point methods. On the problem `d6cube`, our algorithm attains a duality gap two orders smaller and is a little quicker.

Performance of the two algorithms is similar for the problems taking four minutes or less. On the costly problems, like `maros-r7` and `pilot87`, our algorithm tends to outperform OB1-R. In the most expensive problem `dfl001`, our algorithm is significantly faster than OB1-R.

We keep the artificial variables, the slack variables of the equality constraints, to prevent rank deficiency on `dfl001`. Without doing so, neither method terminates successfully, for the matrix $A\Theta A^T$ is very ill-conditioned.

5.2. The "Kennington" problems

Another large problem set found in the NETLIB site is the "Kennington" problems used by Carolan, Hill, Kennington, Niemi, and Wichmann [1]. We present the problems from the set containing $25,000$ to $370,000$ nonzero elements in the matrix A.

Table 4 and Table 5 give the statistics and results of these problems, respectively. If we allow the algorithms to discard artificial variables, both algorithms perform similarly, except that on `pds-10` Adap2 is much faster. Both of the algorithms perform similarly in the problems with small cost (4 minutes or less).

Moreover, our algorithm significantly outperforms OB1-R on the costly problems if we keep all artificial variables to prevent rank deficiency. Even if OB1-R eliminates the artificial variables and solves the smaller problems, the cost of Adap2 keeping the artificial variables is still less than that of OB1-R.

Smaller NETLIB problems							
Problem	LP size and nonzeros			Nonzeros		Density	
Name	Rows	Columns	Nonzeros	AA^T	L	AA^T	L
maros	845	1443	9614	11409	24839	.03	.07
scfxm3	990	1371	7777	8749	13520	.02	.03
seba	515	1028	4352	51400	53748	.39	.41
ship12l	1042	5427	16170	10673	11137	.02	.02
vtp.base	198	203	909	1575	2121	.08	.11

Larger NETLIB problems							
Problem	LP size and nonzeros			Nonzeros		Density	
Name	Rows	Columns	Nonzeros	AA^T	L	AA^T	L
80bau3b	2237	9799	21002	9972	40895	.00	.02
d2q06c	2171	5167	32417	26991	165676	.01	.07
d6cube	404	6184	37704	13054	54445	.16	.67
degen3	1503	1818	24646	50178	119403	.04	.11
dfl001	6071	12230	35632	38098	1634257	.00	.09
fit2d	25	10500	129018	296	299	.99	1.00
greenbea	2389	5405	30877	33791	81914	.01	.03
greenbeb	2389	5405	30882	33766	80503	.01	.03
maros-r7	3136	9408	144848	330472	1195107	.07	.24
pilot	1441	3652	43167	59540	193137	.06	.19
pilot87	2030	4883	73152	115951	421194	.06	.20
stoch3	16675	15695	64875	103360	206731	.00	.00
truss	1000	8806	27836	12561	52509	.03	.11
wood1p	244	2594	70215	18046	18082	.61	.61
woodw	1098	8405	37474	20421	47657	.03	.08

Table 2: Statistics for the larger test problems from NETLIB.

Smaller NETLIB problems							
	IPM ite.		Rel. dual gap		Time		
Problem	OB1-R	Adap2	OB1-R	Adap2	OB1-R	Adap2	Diff
maros	45	45	.11e-08	.12e-08	11.17	11.62	-0.45
scfxm3	39	39	.21e-08	.21e-08	4.25	4.63	-0.38
seba	30	30	.15e-08	.18e-09	43.05	35.79	7.26
ship12l	26	26	.53e-08	.53e-08	4.15	5.22	-1.07
vtp.base	26	26	.33e-08	.38e-08	0.45	0.58	-0.13

Larger NETLIB problems							
	IPM ite.		Rel. dual gap		Time		
Problem	OB1-R	Adap2	OB1-R	Adap2	OB1-R	Adap2	Diff
80bau3b	78	78	.44e-08	.44e-08	46.15	47.15	-1.00
d2q06c	55	55	.25e-08	.46e-08	257.13	245.23	11.90
d6cube	77	77	.67e-06	.85e-08	113.90	110.80	3.10
degen3	30	30	.16e-09	.16e-09	66.22	75.12	-9.90
dfl001	98	98	.27e-06	.27e-06	19844.37	14436.65	5407.72
fit2d	54	54	.21e-08	.21e-08	46.80	49.03	-2.23
greenbea	52	52	-.62e-04	-.62e-04	52.03	53.78	-1.75
greenbeb	74	74	.80e-09	.22e-09	69.15	71.20	-2.05
maros-r7	29	29	.31e-09	.31e-09	1952.93	1743.57	209.36
pilot	77	77	.71e-08	.61e-08	485.08	388.92	96.16
pilot87	82	83	.94e-08	.16e-08	1948.82	1430.97	517.85
stoch3	87	87	.70e-09	.70e-09	142.22	159.12	-16.90
truss	30	30	.80e-09	.80e-09	19.55	21.23	-1.68
wood1p	18	18	.35e-08	.35e-08	12.95	18.23	-5.28
woodw	37	37	.27e-08	.27e-08	25.30	27.05	-1.75

Table 3: Computational results for the larger test problems from NETLIB.

Kennington problems							
Problem	LP size and nonzeros			Nonzeros		Density	
Name	Rows	Columns	Nonzeros	AA^T	L	AA^T	L
cre-b	7240	72447	256095	194579	940374	.01	.04
cre-d	6476	69980	242646	181670	853300	.01	.04
ken-11	14694	21349	49058	33880	118869	.00	.00
ken-13	28632	42659	97246	66586	315642	.00	.00
osa-07	1118	23949	143694	52466	54783	.08	.09
osa-14	2337	52460	314760	113843	116160	.04	.04
pds-06	9881	28655	62524	39061	582158	.00	.01
pds-10	16558	48763	106436	66550	1674872	.00	.01

Table 4: Statistics for the Kennington problems.

5.3. Comparison with the adaptive algorithm

We compare the numerical performance of our algorithm with the adaptive algorithm of [23]. The main differences between the two approaches are as follows.

- Adap2 uses modified search directions in the endgame; however, Adap1 does not.
- Adap2 uses the OB1-R Cholesky factorization first until the OB1-R Cholesky factorization fails to generate a good preconditioner, in the sense that the preconditioned conjugate gradient solver does not converge within 5 iterations by using the refactored preconditioner. We then switch to the hybrid modified Cholesky factorization (Algorithm 16). In contrast, Adap1 uses only the OB1-R Cholesky factorization.
- Adap2 allows zero in the diagonal Cholesky factor P while Adap1 can not handle the situation. Adap2 uses a portion of the modified Cholesky factor by Gill and Murray [9]. See [22, Chap. 5] for details.

Table 6 compares Adap1 and Adap2 in the costly problems that take Adap1 more than $1,500$ seconds to solve. Both algorithms perform similarly for other cheaper problems not listed. Adap2 outperforms in all the problems except the problems `cre-b` and `cre-d` without artificial variables. These two problems are not suitable for iterative solvers since the Θ's are ill-conditioned and change wildly in first μ values. Consequently, Adap2 detects two successive μ values that the number of preconditioned conjugate gradient iterations exceeds the maximum number of iterations allowed. We thus decide to

Kennington problems without artificial variables							
	IPM ite.		Rel. dual gap		Time		
Problem	OB1-R	Adap2	OB1-R	Adap2	OB1-R	Adap2	Diff
cre-b	91	91	-.16e-07	-.16e-07	5020.10	4954.07	66.03
cre-d	92	92	-.69e-08	-.56e-08	3872.00	3839.33	32.67
ken-11	33	33	.16e-09	.16e-09	53.28	68.50	-15.22
ken-13	51	51	.43e-08	.43e-08	244.95	277.92	-32.97
osa-07	53	53	.11e-05	.11e-05	80.68	91.08	-10.40
osa-14	55	55	-.81e-06	-.81e-06	191.52	225.02	-33.50
pds-06	102	102	.48e-09	.44e-09	2817.62	2742.75	74.87
pds-10	128	128	.19e-08	.29e-08	19650.00	14320.02	5329.98

Kennington problems keeping artificial variables							
	IPM ite.		Rel. dual gap		Time		
Problem	OB1-R	Adap2	OB1-R	Adap2	OB1-R	Adap2	Diff
cre-b	102	103	-.99e-09	.14e-08	5365.32	4157.33	1207.99
cre-d	103	103	-.60e-08	.10e-08	4415.48	3511.63	903.85
ken-11	44	44	.21e-08	.21e-08	73.70	93.07	-19.37
ken-13	48	47	-.60e-09	.67e-08	238.77	272.45	-33.68
osa-07	53	53	.11e-05	.11e-05	78.43	90.32	-11.89
osa-14	55	55	-.81e-06	-.81e-06	187.63	221.37	-33.74
pds-06	117	116	.50e-08	.76e-08	3216.82	2470.22	746.60
pds-10	131	131	.69e-08	.32e-08	19978.53	12965.85	7012.68

Table 5: Results for the Kennington problems.

Problem	IPM ite. Adap1	Adap2	Rel. dual gap Adap1	Adap2	Time Adap1	Adap2	Diff
Without artificial variables							
maros-r7	29	29	.31e-09	.31e-09	1825.87	1743.57	82.30
pilot87	82	83	.94e-08	.16e-08	1626.55	1430.97	195.58
cre-b	91	91	-.16e-07	-.16e-07	4872.30	4954.07	-81.77
cre-d	92	92	-.69e-08	-.56e-08	3761.92	3839.33	-77.41
pds-06	102	102	.48e-09	.44e-09	2781.30	2742.75	38.55
pds-10	128	128	.19e-08	.29e-08	18718.57	14320.02	4398.55
Keeping artificial variables							
dfl001	98	98	.27e-06	.27e-06	16644.35	14436.65	2207.70
cre-b	102	103	-.10e-08	.14e-08	4472.75	4157.33	315.42
cre-d	103	103	-.60e-08	.10e-08	3698.33	3511.63	186.70
pds-06	116	116	.64e-08	.76e-08	2554.35	2470.22	84.13
pds-10	131	131	.69e-08	.32e-08	13546.32	12965.85	580.47
NET0416	53	53	.50e-09	.45e-09	9265.85	8281.15	984.70

Table 6: Computational results for Adap1 and our algorithm.

use a direct method at the 9-th and 14-th μ value in the problem `cre-b` and `cre-d`, respectively. In contrast, Adap1 detects zero in the diagonal Cholesky factor P in the second μ value and thus switches a direct method for the two problems.

For the problems that switch to a direct method in the later phases in Adap1 (for example, `dfl001` and `pds-10`), Adap2 achieves significant saving.

6. Conclusion

We have presented an algorithm using the preconditioned conjugate gradient solver through the whole process of interior point methods for linear programming problems. If the algorithm recomputes the preconditioners in later phases, we adopt the hybrid modified Cholesky factorization as an alternative to the Cholesky factorization used by OB1-R. The hybrid modified Cholesky factorization generates a more efficient preconditioner. We modify the preconditioned conjugate gradient solver and rank-1 update and downdate procedure to handle zero component in the diagonal Cholesky factor P.

In the endgame, we perturb the diagonal matrix Θ for determining modified search directions. The resulting coefficient matrix $A\Theta A^T$ is thus more closely related to the preconditioner. We discuss the motivation of the modified search directions and prove the convergence of the interior point method. Numerical results show that the algorithms enhance the performance of OB1-R and the adaptive algorithm in [23].

Acknowledgment

I thank Dianne P. O'Leary for many helpful discussions and useful suggestions during the preparation of this article. The idea used to prove the superlinear convergence rate in § 2.3 is also attributed to her.

References

[1] W. Carolan, J. Hill, J. Kennington, S. Niemi, and S. Wichmann. An empirical evaluation of the KORBX algorithms for military airlift applications. *Operations Research*, 38(2):240–248, 1990.

[2] Joseph Czyzyk, Sanjay Mehrotra, and Stephen J. Wright. PCx user guide. Technical Report OTC 96/01, Optimization Technology Center, Argonne National Laboratory and Northwestern University, 1996.

[3] J. E. Dennis and Jorge J. Moré. A characterization of superlinear convergence and its application to quasi-newton methods. *Mathematics of Computation*, 28(126):549–560, 1974.

[4] J. E. Dennis and Jorge J. Moré. Quasi-newton methods, motivation and theory. *SIAM Review*, 19(1):46–89, 1977.

[5] A. V. Fiacco and G. P. McCormick. *Nonlinear Programming : Sequential Unconstrained Minimization Techniques.* John Wiley & Sons, New York, 1968. Reprint : Volume 4 of *SIAM Classics in Applied Mathematics*, SIAM Publications, Philadelphia, PA 19104–2688, USA, 1990.

[6] Anders Forsgren, Philip E. Gill, and Joseph R. Shinnerl. Stability of symmetric ill-conditioned systems arising in interior methods for constrained optimization. *SIAM Journal on Matrix Analysis and Applications*, 17:187–211, 1996.

[7] Roland W. Freund and Florian Jarre. A QMR-based interior-point algorithm for solving linear programs. Technical report, AT&T Bell Laboratories and Institut für Angewandte Mathematik und Statistik, 1995.

[8] D. M. Gay. Electronic mail distribution of linear programming test problems. *Mathematical Programming Soc. COAL Newsletter*, 1985.

[9] Philip E. Gill and Walter Murray. Newton-type methods for unconstrained and linearly constrained optimization. *Mathematical Programming*, 7:311–350, 1974.

[10] Philip E. Gill, Walter Murray, Dulce B. Ponceleón, and Michael A. Saunders. Primal-dual methods for linear programming. *Mathematical Programming*, 70:251–277, 1995.

[11] Clovis C. Gonzaga. Path-following methods for linear programming. *SIAM Review*, 34(2):167–224, June 1992.

[12] Patricia D. Hough and Stephen A. Vavasis. Complete orthogonal decomposition for weighted least squares. Center of Applied Mathematics and Department of Computer Science, Cornell University, May 1996.

[13] Jr. J. E. Dennis and Robert B.Schnabel. *Numerical methods for unconstrained optimization and nonlinear equations.* Prentice-Hall, Englewood Cliffs, N.J., 1983.

[14] N. K. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4:373–395, 1984.

[15] Irvin J. Lustig, Roy E. Marsten, and David F. Shanno. Computational experience with a primal-dual interior point method for linear programming. *Linear Algebra and Its Application*, 152:191–222, 1991.

[16] Sanjay Mehrotra and Jen-Shan Wang. Conjugate gradient based implementation of interior point methods for network flow problems. Technical Report 95-70.1, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL 60208-3119, U.S.A., October 1995.

[17] Shinji Mizuno and Florian Jarre. Global and polynomial-time convergence of an infeasible-interior-point algorithm using inexact computation. May be found in the IPM archive, April 1996.

[18] L. F. Portugal, M. G. C. Resende, G. Veiga, and J. J. Júdice. A truncated primal-infeasible dual-feasible network interior point method. November 1994.

[19] G. W. Stewart. On scaled projections and pseudo-inverses. *Linear Algebra and Its Applications*, 112:189–194, 1989.

[20] Michael J. Todd. A Dantzig-Wolfe-like variant of Karmarkar's interior-point linear programming algorithm. *Operation Research*, 38:1006–1018, 1990.

[21] Robert J. Vanderbei. LOQO : An interior point code for quadratic programming. Program in Statistics and Operations Research, Princeton University. `rvdb@princeton.edu`, 1995.

[22] Weichung Wang. Iterative methods in interior point algorithms for linear programming. Ph.d. dissertation, Applied Mathematics Program, University of Maryland, August 1996.

[23] Weichung Wang and Dianne P. O'Leary. Adaptive use of iterative methods in interior point methods for linear programming. Technical Report CS-TR-3560, Computer Science Department Report, University of Maryland, November 1995.

[24] J. H. Wilkinson. *The Algebraic Eigenvalue Problem.* Claredon Press, Oxford, England, 1965.

[25] M. H. Wright. Interior methods for constrained optimization. In A. Iserles, editor, *Acta Numerica 1992*, pages 341–407. Cambridge University Press, New York, USA, 1992.

[26] Yin Zhang. Solving large-scale linear programs by interior-point methods under the MATLAB environment. Technical Report TR 96–01, Department of Mathematics and Statistics, University of Maryland Baltimore County, February 1996.