Applied and
Computational
Mathematics
Division

NISTIR 5406

Computing and Applied Mathematics Laboratory

An Interior-Point Method for General Large-Scale Quadratic Programming Problems

P. T. Boggs, P. D. Domich and J. E. Rogers

January 1995

U.S. DEPARTMENT OF COMMERCE
Technology Administration
National Institute of Standards and Technology
Gaithersburg, MD 20899

An Interior-Point Method for General Large Scale Quadratic Programming Problems *

Paul T. Boggs Paul D. Domich Janet E. Rogers

September 8, 1995

Abstract

In this paper we present an interior point algorithm for solving both convex and nonconvex quadratic programs. The method, which is an extension of our interior point work on linear programming problems, efficiently solves a wide class of large scale problems and forms the basis for a sequential quadratic programming (SQP) solver for general large scale nonlinear programs. The key to the algorithm is a 3-dimensional cost-improvement subproblem, which is solved at every iteration. We have developed an approximate recentering procedure and a novel, adaptive big-$\mathcal{M}$ Phase I procedure that are essential to the success. We describe the basic method along with the recentering and big-$\mathcal{M}$ Phase I procedures. Details of the implementation and computational results are also presented.

Keywords: big-$\mathcal{M}$ Phase I procedure, convex quadratic programming, interior point methods, linear programming, method of centers, multi-directional search direction, nonconvex quadratic programming, recentering.

*Contribution of the National Institute of Standards and Technology and not subject to copyright in the United States. This research was supported in part by ONR Contract N-0014-87-F0053.

Applied and Computational Mathematics Division, National Institute of Standards and Technology, Gaithersburg, MD 20899. INTERNET: boggs@cam.nist.gov

Applied and Computational Mathematics Division, National Institute of Standards and Technology, Boulder, CO 80303-3328. INTERNET: domich@boulder.nist.gov

Applied and Computational Mathematics Division, National Institute of Standards and Technology, Boulder, CO 80303-3328. INTERNET: jrogers@boulder.nist.gov

1. Introduction

We have been working on a particular class of interior point methods for solving linear programming problems for several years. (See, e.g., [1] and [2].) In this paper we extend these ideas both to convex and nonconvex quadratic programs. Our experiments, some of which are reported here, show that our algorithm efficiently solves a wide class of problems, including the Netlib linear programming LP problems, and the dual of these Netlib problems modified to include various quadratic objective functions. The code also forms the basis for a sequential quadratic programming (SQP) solver for general large scale nonlinear programs. (See [3] and [4].) In this context, it has been used to solve numerous large scale quadratic programs with indefinite Hessian matrices. To our knowledge, our quadratic programming code has few competitors in terms of robustness and size of problems that can be efficiently handled. We describe below the basic method along with some novel enhancements and details that are essential to the success of the method.

We take the quadratic programming problem to be

$$\begin{aligned} \min_{u} \ & c^{\mathsf{T}} u + \tfrac{1}{2} u^{\mathsf{T}} Q u \\ \text{subject to: } & Au \le b \end{aligned} \tag{QP}$$

where $u, c \in \Re^n$, $Q \in \Re^{n \times n}$, $A \in \Re^{m \times n}$, and $b \in \Re^m$. (Equality constraints or variable bounds in the original problem formulation can be handled as described in, for example, [5].) We assume that Q is symmetric, but possibly indefinite. We make the standard assumptions that the problem has a finite optimal solution and that A has full column rank. We do *not* assume that the problem has a full-dimensional interior, nor do we assume that the problem is nondegenerate.

The basic strategy employed in our algorithm is to solve sequentially low-dimensional approximations to (QP). Specifically, given an initial feasible point, say u^c, we generate q independent directions, solve (QP) restricted to the affine space formed by those directions anchored at the point u^c, and compute a new, strictly feasible point by moving a certain distance in the direction of that solution. This process can then be repeated, as described in our previous work on linear programs. Because of our success with $q = 3$, we call our algorithm O3D for Optimizing over 3-Dimensional Subspaces.

A major feature of our method is that it applies both to (QP) and LP. In fact, except as specifically noted below, the algorithm for LP can be obtained from the description for (QP) by simply setting $Q = 0$. Our method also applies, with minor safeguards, to nonconvex (QP). This is especially important in the SQP application mentioned above since, in the general nonlinear programming case, Q is (an approximation to) the Hessian of the Lagrangian and there is no guarantee that Q will be positive definite.

Our algorithm includes two strategies that have led to significant improvements in computational performance. First, we recenter after each subspace iteration. This improves overall performance in that it results in fewer iterations and improved robustness. Complete recentering is, of course, prohibitively expensive, but we have derived and implemented a novel, approximate recentering strategy that achieves, at reasonable cost, many of the benefits of complete recentering. Second, since we certainly do not require that an initial feasible point be specified, we implement an effective "big $\mathcal{M}$" Phase I procedure that provides an adaptive heuristic for choosing and dynamically adjusting $\mathcal{M}$.

The directions that we use to construct the subproblem can be derived from the method of centers originally proposed by Huard [6]. As noted above, these are natural extensions of the directions we developed for LP. We believe that the inequality formulation (QP) used in this framework provides some advantages over the equality formulation frequently used in a primal-dual approach. As will be shown below, even when the natural form of the problem includes equality constraints that must be converted to inequality form, our method requires the factorization of a matrix that is the sum of Q and a matrix of the form $[A^{\mathsf{T}}A] \in \Re^{n\times n}$. The normal equations for the primal-dual method, on the other hand, result in a matrix of the form $[AQ^{-1}A^{\mathsf{T}}] \in \Re^{m\times m}$ when the problem is in equality form. Unless Q has special structure so that factoring Q is especially easy, forming and factoring $[A^{\mathsf{T}}Q^{-1}A]$ is likely to be significantly more computationally expensive than forming and factoring $[A^{\mathsf{T}}A+Q]$. A primal-dual formulation applied to a problem with only inequality constraints, however, as described by Vanderbei in [7] would not have this problem. In addition, some of the disadvantage of applying a primal dual approach to a problem in strict equality form can also be overcome by working from the reduced KKT system rather than the normal equations, as described in, for example, [7] and [8].

In 2 we present a brief description of the O3D features that form the basis for our quadratic programming work. We discuss the generation of the directions that drive the algorithm, the recentering procedure that we employ to improve the robustness of our code, our adaptive Phase I procedure, and the modifications needed to handle an indefinite Q. The remaining details of our implementation are given in 3. In 4 we present a summary of our numerical results. These results indicate that the proposed algorithm can solve (QP) in approximately the same number of iterations as required for LP. Finally, in 5 we note some areas for future research.

Interior point methods have been extended to the quadratic programming problem by several authors, in addition to the work of Vanderbei and Carpenter described above. For example, Shanno, *et al.* [9] have been working on a primal-dual method, and Anstreicher has suggested a log barrier approach [10] and a related dual version [11] that is similar to our approach, although his work is primarily theoretical. Recently Coleman and Liu [12] have suggested an interior Newton method with strong convergence properties for general quadratic programming problems, but their method has not been extended to the large scale case.

2. O3D

2.1. General Description

In this section we review the basic structure of O3D and describe the important new features as they pertain to (QP). Our algorithm begins by generating three independent directions, s^1, s^2, and s^3, emanating from a given strictly feasible point, say u^c. We then form a "cost-improvement subproblem"

$$\begin{aligned} \min_{\zeta} \; & c^{\mathsf{T}}(u^c + S\zeta) + \tfrac{1}{2}(u^c + S\zeta)^{\mathsf{T}} Q (u^c + S\zeta) \\ \text{subject to: } & A(u^c + S\zeta) \le b \, , \end{aligned} \tag{CI}$$

where $S \in \Re^{n\times 3}$ denotes the matrix with columns containing the three directions and $\zeta \in \Re^3$. Using a simple dual affine method, we solve (CI) for ζ^* and form an intermediate new point

$$\bar{u} = u^c + \alpha S\zeta^* \tag{2.1}$$

for an appropriate value of the step length $\alpha \in (0, 1)$. As noted in the introduction, we have found that approximate recentering improves the robustness of the algorithm considerably. Thus we take the next iterate, u^+, to be the result of finding an approximate center, as discussed in 2.3, of the region formed by the intersection of the feasible region of (QP) with the set $\mathcal{L} \equiv \{u \mid c^{\mathsf{T}}u + \frac{1}{2}u^{\mathsf{T}}Qu \le c^{\mathsf{T}}\bar{u} + \frac{1}{2}\bar{u}^{\mathsf{T}}Q\bar{u}\}$.

For completeness we briefly discuss the formal center of such a region. Define the residual for the objective function of (QP) to be

$$r_0(u, t) = t - (c^{\mathsf{T}}u + \tfrac{1}{2}u^{\mathsf{T}}Qu),$$

for some value of t, and the residual for each of the constraints of (QP) to be

$$r_k(u, t) = (b - Au)_k, \qquad k = 1, \ldots, m \, ,$$

where the subscript k denotes the kth component of the vector. The *center* of the feasible region that corresponds to points where the objective function is strictly less than $\bar{t} = c^\mathsf{T}\bar{u} + \frac{1}{2}\bar{u}^\mathsf{T}Q\bar{u}$ is then defined as the solution to the "recentering problem"

$$\max_u \prod_{k=0}^{m} r_k(u, \bar{t}) \ ,$$

or equivalently,

$$\max_u \sum_{k=0}^{m} \log r_k(u, \bar{t}) \ . \tag{RP}$$

We present a brief description of the directions comprising (CI) in 2.2; the derivation for these directions is a straightforward extension of the material in [2]. The details of our procedure for approximating the solution to (RP) are in 2.3.

2.2. Directions for (CI)

Here we define the directions that make up the cost-improvement subproblem and show that they are scale invariant. These directions are generalizations of those developed for the linear programming case; the details may be found in [1]. We define four directions, but, at a given step of the algorithm, only use three. The directions are given as follows, where $D = \operatorname{diag}\left\{\frac{1}{r_k(u,t)}, k = 1, \ldots, m\right\}$:

- The first direction is the generalized *dual-affine* direction [13] given by

$$s^1 = -\left[A^\mathsf{T}D^2A + \frac{Q}{r_0(u,t)}\right]^{-1}(c + Qu) \ . \tag{2.2}$$

- The second direction is derived from the Newton direction for (RP). This direction can be shown to be a linear combination of s^1 and our second direction,

$$s^2 = \left[A^\mathsf{T}D^2A + \frac{Q}{r_0(u,t)}\right]^{-1} A^\mathsf{T}De \ ,$$

  where e denotes the vector of all ones.

- Our third direction is derived by considering the effect on $\left[A^\mathsf{T}D^2A + \frac{Q}{r_0(u,t)}\right]$ of the first constraint encountered in the s^1 direction. It is easily shown that the rank-one change caused by this constraint should be dominated in early iterations by the row vector $A_{\tilde{k}}$, where $\tilde{k}$ denotes the index of the first constraint encountered. The resulting direction is

$$s^3 = \left[A^\mathsf{T}D^2A + \frac{Q}{r_0(u,t)}\right]^{-1} A_{\tilde{k}}^\mathsf{T} \ .$$

- Finally, using the notation of factorable functions [14], we obtain our fourth direction, a third-order correction to s^1, namely

$$s^4 = \left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} \sum_{k=1}^{m} A_k^\top \left(\frac{[A_k s^1]^2}{r_k(u,t)^3}\right) ,$$

where A_k denotes the kth row of A, $A_k s^1$ is the dot product of the kth row with direction s^1, and $r_k(u,t)^3$ is the cube of the kth residual.

It is usually desirable to develop algorithms that do not depend on an arbitrary scaling. Unfortunately, not all aspects of O3D satisfy this property, but we now show that the search directions are scale invariant.

Proposition 2.1. *The four O3D search directions are scale invariant.*

Proof: Let $S_{\mathcal{R}} \in \Re^{m\times m}$ denote a diagonal scaling matrix for the rows of A, and let $S_C \in \Re^{n\times n}$ denote a diagonal scaling matrix for the columns of A. The scaled problem (QP) is then

$$\begin{aligned} \min_{(S_C^{-1}u)} \; & [S_C c]^\top [S_C^{-1}u] + \frac{1}{2}[S_C^{-1}u]^\top [S_C Q S_C][S_C^{-1}u] \\ \text{subject to: } & [S_{\mathcal{R}} A S_C][S_C^{-1}u] \le [S_{\mathcal{R}} b] \, . \end{aligned} \tag{2.3}$$

The dual affine direction, s^1, for (2.3) is easily shown to be scale invariant, since

$$\begin{aligned} s^1_{scld} &= -\left[[S_{\mathcal{R}} A S_C]^\top [S_{\mathcal{R}}^{-1} D^2 S_{\mathcal{R}}^{-1}][S_{\mathcal{R}} A S_C] + \frac{[S_C Q S_C]}{r_0(u,t)}\right]^{-1} \left([S_C c] + [S_C Q S_C][S_C^{-1}u]\right) \\ &= -\left(S_C^{-1}\left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} S_C^{-1}\right) S_C(c + Qu) \\ &= -S_C^{-1}\left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} (c + Qu) \\ &= S_C^{-1} s^1 \, . \end{aligned}$$

The scale invariance of s^2 and s^4 can be demonstrated analogously. For s^3 we have

$$\begin{aligned} s^3_{scld} &= \left(S_C^{-1}\left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} S_C^{-1}\right)\left[S_C A^\top S_{\mathcal{R}} e_{\tilde{k}}\right] \\ &= S_C^{-1}\left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} A_{\tilde{k}}^\top S_{\mathcal{R}}^{\tilde{k}\tilde{k}} , \end{aligned}$$

where $e_{\tilde{k}}$ denotes the vector corresponding to the $\tilde{k}$th column of an $m \times m$ identity matrix, and $S_{\mathcal{R}}^{\tilde{k}\tilde{k}}$ is the $(\tilde{k}, \tilde{k})$th element of $S_{\mathcal{R}}$. Since $S_{\mathcal{R}}^{\tilde{k}\tilde{k}}$ is a scalar, it does not affect the s_{scld}^3 direction. Thus to show scale invariance for this direction, we must simply show that the first constraint encountered in the s_{scld}^1 direction, whose index we denote $\tilde{k}_{scld}$, is the same as that encountered in the unscaled direction s^1. This constraint is found by selecting from the set of constraints such that $A_k s_{scld}^1 > 0, k = 1, \ldots, m$, the constraint $\tilde{k}_{scld}$ that minimizes

$$\frac{S_{\mathcal{R}}^{kk} r_k(u, t)}{S_{\mathcal{R}}^{kk} A_k S_C s_{scld}^1} .$$

But

$$\frac{S_{\mathcal{R}}^{kk} r_k(u, t)}{S_{\mathcal{R}}^{kk} A_k S_C s_{scld}^1} = \frac{S_{\mathcal{R}}^{kk} r_k(u, t)}{S_{\mathcal{R}}^{kk} A_k S_C \left(-S_C^{-1} \left[A^\top D^2 A + \frac{Q}{r_0(u,t)} \right]^{-1} (c + Qu) \right)} = \frac{r_k(u, t)}{A_k s^1} .$$

Hence, since $\tilde{k}_{scld} = \tilde{k}$, the $A_{\tilde{k}}$ direction is also scale invariant. $\square$

2.3. Subproblem Recentering

The performance of interior-point procedures depends on the proximity of the current iterate to a face of the polytope and can be improved by recentering after the cost-improvement phase. Since the cost of exactly locating a center is prohibitive, we use an approximate recentering procedure that moves the iterate close to the center at a reasonable cost. This recentering reduces the effect of initial starting values and produces a more stable algorithm.

The recentering subproblem closely resembles the subproblem procedure previously developed and is applied after the cost improvement step (2.1) has been taken. Three recentering directions are computed. These three directions, specified below, define a recentering subproblem for which the center is computed efficiently by Newton's method. The recentering subproblem at the cost improvement point, $\bar{u}$, is

$$\max_{\xi} \sum_{k=0}^{m} \log r_k(\bar{u} + W\xi, \bar{t})$$

where W now contains the three recentering directions described below and $\xi \in \Re^3$. The step length is determined using a quadratic line search and the procedure terminates after a prescribed number of iterations, or after determining a center to sufficient accuracy.

The three linearly independent directions that are used to form the recentering subproblem consist of two second-order directions and a cost-improvement direction. The first direction is the Newton direction for (RP) given by

$$w^1 = -\left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} \left(A^\top De + \frac{c+Qu}{r_0(u,t)}\right) . \tag{2.4}$$

The next second-order recentering direction,

$$w^2 = -\left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} \left(A^\top D^2 e + \frac{c+Qu}{r_0(u,t)^2}\right) ,$$

is motivated using the big-$\mathcal{M}$ Phase I formulation, where driving the artificial variable to zero can be interpreted as a form of recentering. As shown in the following proposition, w^2 arises from the dual affine direction for the Phase I problem.

Proposition 2.2. *Using the big-$\mathcal{M}$ Phase I formulation*

$$\begin{array}{c} \min\limits_{u,\theta}\ c^\top u + \frac{1}{2}u^\top Qu + \mathcal{M}\theta \\ \text{subject to: } Au - \theta e \le b \end{array} \tag{2.5}$$

with artificial variable θ, the step in u given by the dual affine search direction (2.2) can be expressed as

$$\left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} \left(\beta_1(A^\top D^2 e) + \beta_2(c+Qu)\right) ,$$

for scalars β_1 and β_2.

Proof: Forming the dual affine search direction for (2.5) requires the solution of the linear system

$$\begin{bmatrix} \left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right] & -(A^\top D^2 e) \\ -(A^\top D^2 e)^\top & e^\top D^2 e \end{bmatrix} \begin{bmatrix} \Delta u \\ \Delta\theta \end{bmatrix} = -\begin{bmatrix} c+Qu \\ \mathcal{M} \end{bmatrix} .$$

The partitioned inverse for the matrix is

$$\frac{1}{\mathcal{K}} \begin{bmatrix} \mathcal{K}\left[\left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right] - \frac{(A^\top D^2 e)^\top (A^\top D^2 e)}{e^\top D^2 e}\right]^{-1} & \left(\left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} (A^\top D^2 e)\right) \\ \left(\left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} (A^\top D^2 e)\right)^\top & 1 \end{bmatrix}$$

where the scalar $\mathcal{K} \equiv e^\top D^2 e - (A^\top D^2 e)^\top \left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} (A^\top D^2 e)$. The change in u, Δu, becomes the sum of two terms,

$$-\left(\left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right] - \frac{(A^\top D^2 e)^\top (A^\top D^2 e)}{e^\top D^2 e}\right)^{-1} (c+Qu) \tag{2.6}$$

and

$$-\left(\frac{\mathcal{M}}{\mathcal{K}}\right)\left[A^{\top}D^2A+\frac{Q}{r_0(u,t)}\right]^{-1}A^{\top}D^2e\,.$$

Applying the Sherman-Morrison-Woodbury formula to (2.6) and rearranging the terms produces the form required and completes the proof of the proposition. □

Since the direction and not the magnitude of w^2 is important, it follows that we can take $\beta_1 = 1$ and $\beta_2 = r_0(u,t)^{-2}$.

The third direction is a cost-improvement direction that is required to prevent the objective function from increasing during the recentering. We use the dual affine direction s^1 (2.2) and specify $r_0(u,t)$, the residual on the objective row, so that the value of the subproblem objective function does not increase.

In the case of the LP formulation, an additional first-order recentering direction replaces the cost-improvement direction. Using (RP), the gradient of

$$L(u,t)=\sum_{k=0}^{m}\log r_k(u,t)$$

provides the first-order LP search direction to determine the center of the polytope, namely

$$w^3=\nabla_u L(u,t)=-\left(A^{\top}De+\frac{c}{r_0(u,t)}\right).$$

For LP we construct recentering search directions orthogonal to c and hence the recentering does not change the objective value.

2.4. Phase I and an Adaptive Big-$\mathcal{M}$ Procedure

When necessary, an initial feasible solution is obtained using the big-$\mathcal{M}$ Phase I formulation (2.5), with an adaptive procedure for updating the value of $\mathcal{M}$. Recently, Ishihara and Kojima [15] and Kojima, Mizuno and Yoshise [16] have also suggested a procedure for adaptively increasing the value of $\mathcal{M}$ for primal-dual interior point algorithms. Their procedure is based on a theoretical result that indicates that there exists an $\mathcal{M}^*$ such that if $\mathcal{M} > \mathcal{M}^*$, then (2.5) is equivalent to (QP). Unfortunately, the value $\mathcal{M}^*$ is unknown, and merely setting $\mathcal{M}$ to an excessively large value can cause numerical difficulties. Thus they propose initializing $\mathcal{M}$ to some nominal value and then increasing it by a factor $\kappa > 1$ when the current estimates of the primal and dual variables indicate that it is necessary. This scaling of $\mathcal{M}$ is continued until it is "large enough", or until it is determined that either there is no minimal solution or the artificial variable is greater than zero at the minimal solution. This adaptive method does not appear to be applicable to our method, however, since it requires feasible estimates of both the primal and dual variables.

Our procedure guarantees by construction that the Phase I dual affine direction, which is the only one of our directions that depends directly on $\mathcal{M}$, will decrease the artificial variable θ at a given iteration. The procedure permits $\mathcal{M}$ to be increased or decreased from the previous iteration as is deemed appropriate. The ability to reduce $\mathcal{M}$ at early iterations appears to be important for the performance of our code; at later iterations the value stabilizes and is essentially unchanged from one iteration to the next. The value selected for $\mathcal{M}$ is a function of the value $\mathcal{M}_{\min}$, defined below in Proposition 2.5. Although we cannot prove that our other directions will always produce a reduction in θ, our empirical evidence indicates that selecting a value of $\mathcal{M}$ large enough to reduce θ in the dual affine direction almost always results in a reduction for the composite direction $S\zeta^*$.

Proposition 2.3. *When Q is positive definite,*

$$\left((v^\top v) - (A^\top Dv)^\top \left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} (A^\top Dv)\right) \;\geq\; 0\,,$$

for any vector $v \in \Re^n$.

Proof:

Under our assumptions, $A^\top D^2 A$ is positive definite, and thus when Q is positive definite

$$(A^\top Dv)^\top \left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} (A^\top Dv) \;\leq\; (A^\top Dv)^\top \left[A^\top D^2 A\right]^{-1} (A^\top Dv)\,,$$

and hence

$$\begin{aligned} &(v^\top v) - (A^\top Dv)^\top \left[A^\top D^2 A + \tfrac{Q}{r_0(u,t)}\right]^{-1} (A^\top Dv) \\ &\quad \geq\; (v^\top v) - (A^\top Dv)^\top \left[A^\top D^2 A\right]^{-1} (A^\top Dv)\,. \end{aligned}$$

Therefore, since

$$\left((v^\top v) - (A^\top Dv)^\top \left[A^\top D^2 A\right]^{-1} (A^\top Dv)\right) \;=\; v^\top P v \;=\; (Pv)^\top (Pv) \;\geq\; 0\,,$$

for $P = \left[I - [DA]\left[[DA]^\top [DA]\right]^{-1} [DA]^\top\right]$, it follows that

$$\left((v^\top v) - (A^\top Dv)^\top \left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} (A^\top Dv)\right) \;\geq\; 0\,,$$

for positive definite Q. $\square$

Proposition 2.4. *Given the Phase I problem (2.5) cost-improvement direction*

$$s = \begin{bmatrix} \left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right] & -(A^\top D^2 e) \\ -(A^\top D^2 e)^\top & e^\top D^2 e \end{bmatrix}^{-1} \begin{pmatrix} g \\ \gamma \end{pmatrix},$$

where $g \in \Re^n$ denotes the first n elements and γ the last element of the appropriate Phase I right hand side, the artificial variable θ will be reduced if

$$\gamma \quad < \quad -g^\top \left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} (A^\top D^2 e).$$

Proof: Using a partitioned inverse as introduced in 2.3, the search direction can be rewritten as

$$s = \begin{bmatrix} \left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} g \\ 0 \end{bmatrix} + \Psi \begin{bmatrix} \left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} (A^\top D^2 e) \\ 1 \end{bmatrix},$$

where

$$\Psi \equiv \frac{\gamma + g^\top \left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} (A^\top D^2 e)}{e^\top D^2 e - (A^\top D^2 e)^\top \left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} (A^\top D^2 e)}. \tag{2.7}$$

Applying Proposition 2.3 with $v = De$ to the denominator of (2.7), it follows that

$$\mathcal{K} \quad \equiv \quad e^\top D^2 e - (A^\top D^2 e)^\top \left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} (A^\top D^2 e) \quad \geq \quad 0\,.$$

Thus, if θ is decreased by search direction s, and $\mathcal{K}$ is positive, it must be that

$$\gamma + g^\top \left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} (A^\top D^2 e) \quad < \quad 0\,,$$

and thus that

$$\gamma \quad < \quad -g^\top \left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]^{-1} (A^\top D^2 e)\,.$$

$\square$

Proposition 2.5. *The artificial variable θ is reduced by the Phase I dual affine direction s^1 if*

$$\mathcal{M} \;>\; \mathcal{M}_{\min} \;\equiv\; -\left((c+Qu)^{\mathsf{T}}\left[A^{\mathsf{T}}D^2A+\tfrac{Q}{r_0(u,t)}\right]^{-1}(A^{\mathsf{T}}D^2e)\right).$$

Proof: The proof follows directly from Proposition 2.4 and the definition of the right hand side used to compute the Phase I dual affine direction, for which $g = -(c+Qu)$ and $\gamma = -\mathcal{M}$. □

Factoring $\left[A^{\mathsf{T}}D^2A+\frac{Q}{r_0(u,t)}\right]$ and solving for

$$\Gamma = \left[A^{\mathsf{T}}D^2A+\frac{Q}{r_0(u,t)}\right]^{-1}(A^{\mathsf{T}}D^2e)$$

allow us to compute $\mathcal{M}_{\min} = -(c+Qu)^{\mathsf{T}}\Gamma$ and $\mathcal{K} = e^{\mathsf{T}}D^2e - (A^{\mathsf{T}}D^2e)^{\mathsf{T}}\Gamma$ directly. Then, each of the Phase I search directions can be calculated equally efficiently using

$$s = \begin{bmatrix} \left[A^{\mathsf{T}}D^2A+\frac{Q}{r_0(u,t)}\right]^{-1}g \\ 0 \end{bmatrix} + \left(\frac{\gamma+g^{\mathsf{T}}\Gamma}{\mathcal{K}}\right)\begin{bmatrix}\Gamma\\ 1\end{bmatrix}.$$

Our adaptive computation of $\mathcal{M}$ and the construction of the Phase I search directions require an additional back-solve at each Phase I iteration, but have the advantage that $\left[A^{\mathsf{T}}D^2A+\frac{Q}{r_0(u,t)}\right]$ is created without including the dense artificial column e and thus is less costly both to form and factor. In a practical implementation, a procedure is needed for selecting $\mathcal{M} > \mathcal{M}_{\min}$; these details are given in 3.4.

2.5. Indefinite Quadratic Programming

The extension of our algorithm to include an indefinite Q presents two specific difficulties to overcome. The first arises in computing the directions for (CI). The second difficulty arises in the solution of (CI) where the (3×3) matrix $S^{\mathsf{T}}QS$ may also be indefinite. We address these two problems in the order given.

From 2.2, all of the (CI) directions are of the form

$$\left[A^{\mathsf{T}}D^2A+\frac{Q}{r_0(u,t)}\right]^{-1}g$$

for $r_0(u,t)$ equal to the current residual on the objective row. Although A is assumed full column rank and $\left[A^{\mathsf{T}}D^2A\right]$ is positive definite, for Q indefinite, $\left[A^{\mathsf{T}}D^2A+\frac{Q}{r_0(u,t)}\right]$ will be indefinite for sufficiently small values of $r_0(u,t)$. In this case, we could simply use a general symmetric solver, but such procedures require pivoting to maintain numerical stability. Thus they tend to destroy sparsity and to generate additional costs since the sparsity structure cannot be determined in advance. Instead, we use a modification to the Cholesky factorization based on that proposed by Gill, Murray, and Wright in [17], whereby we transform the indefinite matrix into a positive semidefinite one. A better approach to generating search directions is an area for future work (see also, 5).

For the second problem, we simply solve the appropriate (3×3) system using a general dense Gaussian elimination procedure for the subproblem dual affine direction and check that the problem is bounded in that direction. Clearly, if (CI) is unbounded then so is the original problem and we terminate the procedure with an appropriate indication.

As noted above this procedure has allowed the solution of numerous indefinite problems that have arisen in the solution of general nonlinear programming problems by an SQP approach.

3. Implementation Details

3.1. General

In this section we supply some of the specifics of our code, beginning with general information, followed by specialized details in subsequent subsections. As noted in 2.2, the O3D procedure uses three cost-improvement directions to form the subproblem (CI). In our current implementation, these are s^1 and s^4, and either s^2 or s^3, where s^3 is used in early iterations, and s^2 is used in the final iterations. The "switch-over" from the s^3 direction to the s^2 direction is based on the duality gap in a manner similar to that reported in [1]. The O3D procedure stops when it has satisfied at least one of three convergence criteria: (a) the relative difference between the primal and dual objective values is less than 10^{-8} (see, e.g., [13] and also 3.5); (b) the relative change in the objective function is less than 10^{-10}; or (c) the relative step length is less than 10^{-16}.

The matrices A, Q, and $\left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]$ are stored in a sparse format. The residual on the objective function, needed for the formation of $\left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]$, is computed using the value originally suggested by [18] but extended for (QP), namely

$$r_0(u,t) = \left| \frac{(c+Qu)^\top (c+Qu)}{(c+Qu)^\top (A^\top De)} \right| > 0\,;$$

equivalently, the right hand side t for the "objective function constraint" is

$$t = (c^\top u + \tfrac{1}{2}u^\top Qu) + r_0(u,t) = (c^\top u + \tfrac{1}{2}u^\top Qu) + \left| \frac{(c+Qu)^\top (c+Qu)}{(c+Qu)^\top (A^\top De)} \right| .$$

A minimum-local-fill ordering procedure [19] is invoked once at the beginning of the procedure to find a permutation of the rows and columns of $\left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]$ that reduces fill-in. The reordered matrix is factored and solved at every iteration using the Yale Sparse Matrix Package SMPAK [20], modified in a manner based on that described in [17] to force positive semi-definiteness. (See also 2.5.) All code is written in double precision Fortran 77, and the results reported in 4 are executed on an IBM RISC System/6000 Model 590 workstation running at 66MHz using the xlf version 3.1 compiler with the -O option.

3.2. Constraint and Variable Preprocessing

Preprocessing steps are changes made to the problem data to improve the performance characteristics of the O3D procedure. In general, these are steps that are completed prior to beginning the actual minimization. The reordering of $\left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]$ described above is a preprocessing step. Scaling would also be a preprocessing step, although the results reported in this paper were obtained using unscaled data, since the cost-improvement subproblem (CI) that forms the heart of our algorithm is scale invariant (see 2.2). The significance of scaling is discussed further in, e.g., [21], [22], [23] and [24].

Preprocessing steps also include dropping unused, redundant and unnecessary constraints and variables from the problem, thereby reducing the problem size. This is important because many problems contain redundant constraints; variables whose values are fixed by the constraints; and elements of u that have a corresponding cost coefficient and row and column of Q that are zero, but are unbounded by the constraint matrix. For example, almost two-thirds of the dual forms of the problems in the Netlib LP test suite can easily be shown to have one or more elements of u that are unbounded by the constraint matrix and have a corresponding zero cost coefficient. Such unbounded variables can become excessively large in magnitude, especially in the presence of the recentering procedure of 2.3. While these extremes do not theoretically affect the optimal objective value, they could result in significant roundoff errors when evaluating the residuals, and therefore could affect numerical accuracy. As part of the O3D preprocessing, therefore, we provide a procedure that iteratively drops unnecessary rows and columns of the constraint matrix A, while concurrently making the appropriate adjustments to c, b, Q and the objective value.

This reduction procedure results in a substantial decrease in $\|u\|_2$ at the solution for many problems, even though it cannot guarantee a bounded region. Although our empirical evidence indicates that problem reduction does not significantly affect the total number of iterations, it does have a computational advantage because it reduces the number of Phase I iterations, and because it reduces, often significantly, the dimension of the matrix $\left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]$ which must be factored at each iteration.

For the Netlib LP test suite, we found that our reduction procedure removes up to 47% of the constraints, with at least 10% of the constraints dropped in 50 of the 88 problems that we examined. The reduction procedure also removes up to 57% of the variables in any given problem, although at least 15 of these problems still appear to include one or more unbounded variables.

3.3. Initial Values

A solution procedure for (QP) requires initialization of u, and, when the big-$\mathcal{M}$ Phase I formulation (2.5) is used, of θ as well. Empirically, the performance of interior point algorithms is sensitive to each of these quantities, although as already noted, our recentering procedure greatly reduces this sensitivity. We now discuss our selection of initial values for u and θ. In 3.4 we specify $\mathcal{M}$.

We have tested our O3D procedure using three different initial values of u, namely:

(i) 0;

(ii) e; and

(iii) the unweighted least squares solution $\begin{bmatrix} A^\top A & -(A^\top e) \\ -(A^\top e)^\top & e^\top e \end{bmatrix}^{-1} \begin{bmatrix} A^\top \\ -e^\top \end{bmatrix} b$.

The choice is an option in the code.

Using (i), 85 out of 88 problems from the Netlib LP test set require the solution of a Phase I problem to obtain a feasible starting point; for (ii) and (iii), all 88 problems require a Phase I start. Only (i) is scale invariant under the assumption that scaling would be performed before selecting the initial value for u.

Without the recentering procedure, (iii) requires on average almost 50% more iterations than either (i) or (ii), although some of the 88 problems do solve in fewer iterations using (iii). The performance of (i) and (ii) are roughly the same, but, as expected, some individual problems show significant differences. With recentering, however, the number of iterations required for each of the three starting values is almost identical for all problems. For the results reported in 4, we initialize u to 0.

For the work reported in this paper, we have simplified the selection of an initial value for θ from that used in our original work [1]. For our current work, we set

$$\theta = \begin{cases} \mid r_{\min} \mid + \frac{1}{2}(r_{\min_+} + r_{\max_+}) & \text{if } r_{\max_+} > 0 \\ \mid r_{\min} \mid + 1 & \text{otherwise,} \end{cases}$$

where $r_{\min_+}$ and $r_{\max_+}$ denote the minimum and maximum, respectively, of the *positive* residuals at the initial value of u, and $\mid r_{\min} \mid$ is the magnitude of smallest residual at this point. This value ensures that the initial value of $(u, \theta)^{\mathsf{T}}$ is sufficiently feasible while only moderately increasing the magnitude of each of the residuals.

3.4. Setting big-$\mathcal{M}$

In 2.4 we prove that selecting $\mathcal{M} > \mathcal{M}_{\min}$ guarantees that the dual affine direction will reduce the artificial variable θ. In addition, we note therein that empirical evidence indicates that when the dual affine direction reduces θ then our composite step $S\zeta^*$ also reduces θ. Empirical evidence further indicates that selecting $\mathcal{M}$ relatively larger at early iterations, and then decreasing its value in later iterations once the value $\mathcal{M}\theta$ is dominated by the objective value, is beneficial in terms both of reducing the total number of iterations and of improving robustness. We thus set $\mathcal{M} = \mu \max\{1, \mathcal{M}_{\min}\}$, where $\mu = 100$ at iteration 1 and while $\mathcal{M}_{\min}\theta \left(1+ \mid c^{\mathsf{T}}u + u^{\mathsf{T}}Qu \mid\right)^{-1} > 10^{-1}$; $\mu = 10$ while $10^{-1} > \mathcal{M}_{\min}\theta \left(1+ \mid c^{\mathsf{T}}u + u^{\mathsf{T}}Qu \mid\right)^{-1} > 10^{-3}$; and $\mu = 2$ otherwise.

3.5. Estimating Dual Variables and the Duality Gap

The dual of (QP) when Q is positive definite can be written as

$$\begin{aligned} &\min_{y,u} \; b^{\mathsf{T}}y + \tfrac{1}{2}u^{\mathsf{T}}Qu \\ &\text{subject to: } A^{\mathsf{T}}y = -(c + Qu) \;, \quad y \geq 0 \;, \end{aligned}$$

and the dual variables y can be approximated by

$$\tilde{y} = -D^2A\left[A^{\mathsf{T}}D^2A\right]^{-1}(c + Qu) \;, \tag{3.1}$$

which, if $\tilde{y} \geq 0$ and $\tilde{y}^{\mathsf{T}}r = 0$, will be a dual feasible point. The computation of $\tilde{y}$ using (3.1) requires significant extra cost, however, since we have a factorization of $\left[A^{\mathsf{T}}D^2A + \frac{Q}{r_0(u,t)}\right]$ but not of $\left[A^{\mathsf{T}}D^2A\right]$. Since $r_0/r_i \approx y_i$ near the solution, $A^{\mathsf{T}}D^2A$ will dominate $r_0^{-1}Q$ in

$\left[A^{\mathsf{T}}D^2A + \frac{Q}{r_0(u,t)}\right]$ near the solution, and therefore $\left[A^{\mathsf{T}}D^2A + \frac{Q}{r_0(u,t)}\right]$ goes to $\left[A^{\mathsf{T}}D^2A\right]$ in the limit. We therefore approximate $\tilde{y}$ by first calculating

$$\hat{y} \ = \ -D^2A\left[A^{\mathsf{T}}D^2A + \frac{Q}{r_0(u,t)}\right]^{-1}(c+Qu) \ = \ D^2As^1 \ , \tag{3.2}$$

then setting $\hat{y}_j = \max\{\hat{y}_j, 0\}$, $j = 1, \ldots, m$. Since this value may fail to solve the system $A^{\mathsf{T}}y = -(c + Qu)$, we calculate a relative error for $\hat{y}$ using

$$\sigma_y \quad \equiv \quad \frac{\left\|A^{\mathsf{T}}\hat{y} + (c+Qu)\right\|_2}{1 + \|(c+Qu)\|_2} \ .$$

The relative duality gap is specified by

$$\delta_1 \ \equiv \ \frac{|\ c^{\mathsf{T}}u - b^{\mathsf{T}}y\ |}{1+|\ c^{\mathsf{T}}u + \frac{1}{2}u^{\mathsf{T}}Qu\ |} \qquad \text{and} \qquad \delta_2 \ \equiv \ \frac{\max\{|\ y_ir_i\ |, i = 1, \ldots, m\}}{1+|\ c^{\mathsf{T}}u + \frac{1}{2}u^{\mathsf{T}}Qu\ |} \ .$$

If $\sigma_y \leq 10^{-8}$, $\delta_1 \leq 10^{-8}$, and $\delta_2 \leq 10^{-8}$, then our duality-gap convergence criterion is satisfied.

We have found that using (3.2) to approximate the dual variables for (QP) works quite well in practice. If we terminate by a different criterion, however, we recompute the dual variables at the end using (3.1). For most problems, this recomputation is unnecessary. For those problems for which the dual variables are recomputed, final dual variables satisfy $\delta_1, \delta_2 \leq 10^{-8}$ and $\sigma_y \leq 10^{-8}$ about one third of the time.

4. Results

We analyze one set of LP problems, and six sets of QP problems. The LP problem set is comprised of the dual of the standard form of 88 of the problems publicly available through Netlib [25]; the QP problem sets are constructed by augmenting these linear programming problems with various matrices ρQ, where the scalar ρ is selected so that, given the LP solution u_{lp}, $(c^{\mathsf{T}}u_{lp})(u_{lp}^{\mathsf{T}}[\rho Q]u_{lp})^{-1} = 1$. Three positive definite Q matrices are used:

- Q_I is an appropriately dimensioned identity matrix;
- Q_T is an appropriately dimensioned tridiagonal matrix with 2 on the diagonal, and -1 on the off diagonals; and
- Q_G is the general positive definite matrix $\tilde{A}^{\mathsf{T}}\tilde{A}$, where $\tilde{A}$ indicates a matrix formed from the constraint matrix A by substituting 1 for each nonzero entry, and then randomly permuting the columns. The resulting matrix Q_G will thus have approximately the same density as $A^{\mathsf{T}}A$ but a different nonzero structure.

Table 1: Summarized Results for all 88 Problems of Set

Problem Set	d2q06c Iterations	d2q06c Time (sec)	All Problems Iterations	All Problems Time (sec)	Number with σ_y, δ_1, δ_2 $\leq 10^{-8}$	Number with σ_y, δ_1, δ_2 $\leq 10^{-6}$
LP						
	30	29	1785	686	85	88
QP (positive definite Q matrix)						
Q_I	38	36	2112	790	76	88
Q_T	37	42	2183	1123	73	88
Q_G	37	1300	2328	11979	74	87
QP (indefinite Q matrix)						
Q_{I_i}	79	62	3599	1363	63	85
Q_{T_i}	47	51	3554	1517	56	85
Q_{G_i}	39	658	2532	6042	68	88

From these positive definite Q matrices, three indefinite Q matrices are constructed, Q_{I_i}, Q_{T_i} and Q_{G_i}. These are the same as Q_I, Q_T and Q_G, respectively, except that 50% of their diagonal entries have been randomly negated. For the QP problem sets with indefinite quadratic terms, we have also augmented the constraint matrices with bound constraints on each of the variables prior to forming Q_{G_i}. The bounds are $u_{lp}(i) \pm \max\{50\%\ u_{lp}(i),\ 0.1\}$, $i = 1, \ldots, n$, where $u_{lp}(i)$ denotes the ith component of the LP solution.

The 88 problems we constructed range in size from 27 (afiro) to 5,467 (80bau3b) variables, and from 52 (afiro) to 25,621 (80bau3b) constraints. The maximum number of nonzeros in the constraint matrix A is 49,483 (pilot), and the maximum number of nonzeros in (the lower triangular portion of the symmetric matrix) Q is 197,676 (fit1p). The largest problem (based on the number of nonzeros in the Cholesky factor of $\left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]$) was constructed from Netlib problem d2q06c and has 2,160 variables, 10,139 constraints, 37,349 nonzeros in A, 29,086 nonzeros in Q, and 1,038,127 nonzeros in each factor of $\left[A^\top D^2 A + \frac{Q}{r_0(u,t)}\right]$. A summary of our results is presented in Table I.

5. Conclusions

Overall, this study has demonstrated an important new procedure for solving large quadratic programs. Using a subproblem approach, the procedure has effectively solved large general quadratic programs. A Phase I problem with a dynamic big-$\mathcal{M}$ approach has been developed, as well as a procedure that approximately recenters the current iterate to improve the performance and the numerical stability of the method. For each of these algorithmic developments we provide some theoretical justification; we are currently pursuing a more complete theoretical basis.

Our extension of the procedure for nonconvex quadratic programs has been successfully used to solve numerous problems arising from a SQP approach for solving general nonlinear problems. This extension, however, is *ad hoc*, and is an area for future work.

References

[1] Paul D. Domich, Paul T. Boggs, Janet E. Rogers, and Christoph Witzgall. Optimizing over three-dimensional subspaces in an interior-point method for linear programming. *Linear Algebra and its Applications*, 152, July 1991.

[2] Paul T. Boggs, Paul D. Domich, Janet E. Rogers, and Christoph Witzgall. An interior point method for linear and quadratic programming problems. *Mathematical Programming Society Committee on Algorithms Newsletter*, 19:32–40, 1991.

[3] Paul T. Boggs, Jon W. Tolle, and Anthony J. Kearsley. A practical algorithm for general large scale nonlinear optimization problems. NISTIR 5407, National Institute of Standards and Technology, 1994. (To appear in *SIAM Journal of Optimization.*).

[4] Paul T. Boggs, Jon W. Tolle, and Anthony J. Kearsley. A truncated SQP algorithm for large scale nonlinear programming problems. In *Advances in Optimization and Numerical Analysis: Proceedings of the Sixth Workshop on Optimization and Numerical Analysis, Oaxaca, Mexico*, Dordrecht, The Netherlands, 1994. Kluwer Academic Publishers.

[5] Roger Fletcher. *Practical Methods of Optimization, Second Edition*. John Wiley and Sons, Inc., New York, 1987.

[6] Pierre Huard. Resolution of mathematical programming with nonlinear constraints by the method of centres. In J. Abadie, editor, *Nonlinear Programming*, pages 209–219. North Holland, Amsterdam, 1967.

[7] Robert J. Vanderbei. Interior-point methods: algorithms and formulations. *ORSA Journal of Computing*, 6:32–34, 1994.

[8] Robert J. Vanderbei and Tamra J. Carpenter. Symmetric indefinite systems for interior-point methods. *Mathematical Programming*, 58:1–32, 1993.

[9] David F. Shanno, 1991. Private communication.

[10] K. M. Anstreicher, D. den Hertog, and C. Roos. A long step barrier method for convex quadratic programming. *Algorithmica*, 10(5):365, 1993.

[11] Kurt M. Anstreicher. On long step path following and SUMT for linear and quadratic programming. Manuscript, Yale School of Organization and Management, August 1990.

[12] Thomas F. Coleman and Jianguo Liu. An interior newton method for quadratic programming. Technical Report TR 93-1388, Cornell University, October 1993.

[13] Ilan Adler, Mauricio G. C. Resende, and Geraldo Veiga. An implementation of Karmarkar's algorithm for linear programming. *Mathematical Programming*, 44(3):297–335, 1989.

[14] Richard H. F. Jackson and Garth P. McCormick. The polyadic structure of factorable function tensors with applications to higher-order minimization techniques. *Journal of Optimization Theory and Applications*, 51(1):63–93, 1986.

[15] Tohru Ishihara and Masakazu Kojima. On the big $\mathcal{M}$ in the affine scaling algorithm. *Mathematical Programming*, 62(1):85–94, 1993.

[16] Masakazu Kojima, Shinji Mizuno, and Akiko Yoshise. A little theorem of the big $\mathcal{M}$ in interior point algorithms. *Mathematical Programming*, 59(3):361–375, 1993.

[17] Philip E. Gill, Walter Murray, and Margaret H. Wright. *Practical Optimization*. Academic Press, Inc., 1981.

[18] Anthony V. Fiacco and Garth P. McCormick. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*. John Wiley and Sons, Inc., New York, 1968.

[19] Jiri Vlach and Kishore Singhal. *Computer Methods for Circuit Analysis and Design*. Van Nostrand Reinhold Company, New York, 1983.

[20] *SMPAK User's Guide Version 1.0*, 1985.

[21] Paul T. Boggs, Paul D. Domich, Janet R. Donaldson, and Christoph Witzgall. Algorithmic enhancements to the method of centers for linear programming programming. *ORSA Journal on Computing*, 1(3):159–171, 1989.

[22] 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 Applications*, 152:191–222, 1991.

[23] Roy E. Marsten, Matthew J. Saltzman, David F. Shanno, George S. Pierce, and J. F. Ballintijn. Implementation of a dual affine interior point algorithm for linear programming. *ORSA Journal on Computing*, 1:287–297, 1990.

[24] A. Marxen. *Primal Barrier Methods for Linear Programming*. PhD thesis, Department of Operations Research, Stanford University, Stanford, California, 1989.

[25] David M. Gay. Electronic mail distribution of linear programming test problems. Mathematical Programming Society COAL Newsletter 13, December 1985.