An exact primal---dual penalty method approach to warmstarting interior-point methods for linear programming — Source link ↗

Hande Y. Benson, David F. Shanno

Institutions: Drexel University, Rutgers University

Topics: Penalty method, Linear-fractional programming, Interior point method, Linear programming and Branch and price

Related papers:

- Warm-Start Strategies in Interior-Point Methods for Linear Programming

- Warm start of the primal-dual method applied in the cutting-plane scheme

- Primal-Dual Interior-Point Methods

- A New Unblocking Technique to Warmstart Interior Point Methods Based on Sensitivity Analysis

- Interior-point methods for nonconvex nonlinear programming: regularization and warmstarts

Share this paper: 📘 🐦 💼 ✉

View more about this paper here: https://typeset.io/papers/an-exact-primal-dual-penalty-method-approach-to-warmstarting-4gna6ulhd3

AN EXACT PRIMAL-DUAL PENALTY METHOD APPROACH TO WARMSTARTING INTERIOR-POINT METHODS FOR LINEAR PROGRAMMING

HANDE Y. BENSON AND DAVID F. SHANNO

ABSTRACT. One perceived deficiency of interior-point methods in comparison to active set methods is their inability to efficiently re-optimize by solving closely related problems after a warmstart. In this paper, we investigate the use of a primal-dual penalty approach to overcome this problem. We prove exactness and convergence and show encouraging numerical results on a set of linear and mixed integer programming problems.

1. INTRODUCTION

In this paper, we consider the re-optimization of linear programming problems (LPs) of the form

$$
\begin{array}{lrcl}
\text{maximize} & c^T x \\
\text{subject to} & Ax & \leq & b \\
& x & \geq & 0,
\end{array}
\tag{1}
$$

where $x \in \mathbb{R}^n$ are the decision variables, and $c \in \mathbb{R}^n, A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$ are the problem data. The data, especially c and b, are frequently subject to change, so it may be necessary to solve a series of closely related optimization problems. Engineering problems with changing specifications and business problems with changing market prices and demand are examples of such situations. In the context of a branch-and-bound algorithm to solve problems with binary variables, even the lower bound on x can change. Other approaches for mixed integer programming may require the addition of new constraints or variables before resolving a growing master problem.

If the initial problem is large-scale and the solution time considerable, it is reasonable to expect that solving the rest of the problems in the series will require a similar amount of effort. However, it is reasonable to also expect that if the change in the problem data is small enough, the change in the optimal solution will be small. *Warmstarting* is the use of information obtained during the solution of the initial problem to solve the subsequent perturbed problems. When no such information is used, the new problem is solved from a *coldstart*.

In this paper, we consider the issue of warmstarting when using an interior-point method. We will focus here on the algorithm implemented in the software LOQO [22], but the general deficiency of interior-point methods with respect to

warmstarting is well-known and documented. To summarize, the main issue is that the solution of the initial problem is on the boundary and well-centered, but when the problem is perturbed, the old solution can be still close to the boundary but badly centered. Because of its behavior around the boundary, the interior-point method will then take a series of short steps and make slow progress toward the new optimum. Also, numerical problems may arise, requiring the use of iterative refinement techniques and possibly resulting in step calculations moving the iterate to a point much farther away from the new optimum.

Several remedies have been proposed for warmstarting an interior-point method for linear programming. Among them are Freund's shifted barrier method [7] to allow infeasibility of the nonnegative variables, Gondzio and Grothey's [9] and Yildirim and Wright's [23] storage of a nearly optimal, well-centered point, Lustig et.al.'s [16] perturbation of the problem to move the boundary, and Polyak's modified barrier function [19]. Of these papers, only [9] and [16] report numerical results, and only on a few classes of problems. The results show improvement over coldstarts.

Our contribution here is the use of an exact penalty method to aid in warmstarting. We have documented on an ℓ_∞ penalty method for mathematical programs with equilibrium constraints (MPECs) in [2], and in the same paper, discussed possible advantages to using a penalty method to bound optimal sets of Lagrange multipliers, to find nonKKT optimal solutions, and to detect primal infeasibility for general nonlinear programming problems (NLP). Another additional benefit mentioned in that paper is that the ℓ_1 penalty method can be used to overcome jamming, which occurs when the algorithm gets stuck at a nonoptimal (possibly infeasible) point. As discussed in [3], an interior-point method such as LOQO can fail to make progress under certain conditions if the slack variables vanish prematurely. We will show that the deficiency of interior-point methods with respect to warmstarting is similar to jamming, and therefore, an ℓ_1 penalty method can be used as a remedy.

Another issue is that the standard ℓ_1 penalty approach is of a primal nature and can be useful when the perturbation in the problem is due to changes in the constraints. However, its use is only limited to this case, and it has very little, if any, effect when the perturbation in the problem is due to changes in the objective function. Therefore, we introduce here a primal-dual penalty approach, in which both the primal and the dual problems are relaxed and constrained simultaneously. For the resulting model, we will show that it yields an exact penalty approach, which is useful for any type of data perturbation.

In this paper, we have chosen to focus on the case of linear programming and report encouraging numerical results on problems from the Netlib test suite and on some mixed integer linear programming problems. The outline of our paper is as follows: We start with a description of the interior-point method in Section 2. In Section 3, we illustrate how such a method can fail when warmstarting. Then, in Section 4, we introduce the primal-dual penalty approach and demonstrate how it can remedy our problem. Exactness of the penalty problem and convergence analysis for this approach are also provided. In Section 5, we describe the implementation and testing of our warmstart approach and provide numerical results.

2. An Infeasible Interior-Point Method for Linear Programming

We begin with an overview of the interior-point algorithm implemented in the solver LOQO. Although LOQO is currently known best as a nonlinear programming solver, it was originally designed to solve LPs. The form of the problem (1) only allows inequality constraints and nonnegative variables, but this simplification is for purely pedagogical reasons. The algorithm described below and the warmstart work to follow is implemented for more general problems, which is also demonstrated in the numerical results. Further details can be found in [21] and [22].

The dual problem of (1) is given by

$$
\begin{array}{rll}
\text{minimize} & b^T y \\
\text{(2)} \qquad \text{subject to} & A^T y & \geq & c \\
& y & \geq & 0,
\end{array}
$$

and its optimality conditions can be written as

$$
\begin{array}{rll}
Ax + w &=& b \\
A^T y - z &=& c \\
x_j z_j &=& 0, \quad j = 1, \ldots, n \\
w_i y_i &=& 0, \quad i = 1, \ldots, m,
\end{array}
\tag{3}
$$

where w and z are the slack variables associated with the primal and dual constraints, respectively. The first set of equations ensures *primal feasibility*, the second *dual feasibility*, and the last two *complementarity*.

We start by relaxing the complementarity conditions as

$$
\begin{array}{rll}
x_j z_j &=& \mu, \quad j = 1, \ldots, n \\
w_i y_i &=& \mu, \quad i = 1, \ldots, m,
\end{array}
$$

where μ is a nonnegative parameter controlled by the algorithm. μ is referred to as the *barrier parameter* since the above conditions can also be derived from the logarithmic barrier problem of Fiacco and McCormick [4]. It is started at a (possibly large) positive number and driven to zero as the optimum is approached. This ensures that the iterates are kept away from the boundary until the optimal solution is obtained.

At each iteration, we have that $(x, w, y, z) > 0$, and our goal is to find step directions $(\Delta x, \Delta w, \Delta y, \Delta z)$ such that the new point $(x + \Delta x, w + \Delta w, y + \Delta y, z + \Delta z)$ lies on the central path, that is, satisfies the optimality conditions for the current value of μ, which is calculated as

$$
\mu = r \frac{w^T y + x^T z}{m + n},
\tag{4}
$$

where r is a constant between 0 and 1. Newton's Method is used to obtain the step directions, and it gives the following system to solve at each iteration:

$$
\begin{bmatrix}
A & 0 & I & 0 \\
0 & A^T & 0 & -I \\
Z & 0 & 0 & X \\
0 & W & Y & 0
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
\Delta y \\
\Delta w \\
\Delta z
\end{bmatrix}
=
\begin{bmatrix}
b - Ax - w \\
c - A^T y + z \\
\mu e - XZe \\
\mu e - YWe
\end{bmatrix},
$$

where $X, W, Y,$ and Z denote diagonal matrices with entries from the vectors $x, w, y,$ and z, respectively, and e is a vector of ones of appropriate dimension.

This system can be reduced further without providing any additional fill-in. Thus, we factor out Δz and Δw to obtain the *reduced KKT system*:

$$(5) \qquad \begin{bmatrix} -E & A \\ A^T & F \end{bmatrix} \begin{bmatrix} \Delta y \\ \Delta x \end{bmatrix} = \begin{bmatrix} \rho - E\gamma_w \\ \sigma + \gamma_z \end{bmatrix},$$

where

$$(6) \qquad \begin{aligned} E &= Y^{-1}W \\ F &= X^{-1}Z \\ \rho &= b - Ax - w \\ \sigma &= c - A^T y + z \\ \gamma_w &= \mu W^{-1}e - y \\ \gamma_z &= \mu X^{-1}e - z. \end{aligned}$$

We also have that

$$(7) \qquad \begin{aligned} \Delta z &= \gamma_z - F\Delta x \\ \Delta w &= E(\gamma_w - \Delta y). \end{aligned}$$

LOQO then proceeds to a new estimate of the optimum by

$$\begin{aligned} x^{(k+1)} &= x^{(k)} + \gamma\alpha_p^{(k)}\Delta x^{(k)} \\ w^{(k+1)} &= w^{(k)} + \gamma\alpha_p^{(k)}\Delta w^{(k)} \\ y^{(k+1)} &= y^{(k)} + \gamma\alpha_d^{(k)}\Delta y^{(k)} \\ z^{(k+1)} &= z^{(k)} + \gamma\alpha_d^{(k)}\Delta z^{(k)} \end{aligned}$$

where the superscripts denote the iteration number, γ is a parameter close to but less than 1, the primal steplength, $\alpha_p^{(k)}$, is chosen to ensure that $(x^{(k+1)}, w^{(k+1)}) > 0$ and the dual steplength, $\alpha_d^{(k)}$, is chosen to ensure that $(y^{(k+1)}, z^{(k+1)}) > 0$.

3. Warmstarting in Interior-Point Methods

In this section, we will examine the potential difficulties for an interior-point method after a warmstart. Consider the following problem:

$$\begin{aligned} \text{maximize} \quad & x_1 + 3x_2 \\ \text{subject to} \quad & x_1 + x_2 \leq 3 \\ & 2x_1 + x_2 \leq 2 \\ & x_1, x_2 \geq 0. \end{aligned}$$

The optimal primal and dual solutions for this problem are

$$(8) \qquad \begin{aligned} x &= (0, 2), & w &= (1, 0) \\ y &= (0, 3), & z &= (5, 0). \end{aligned}$$

Therefore, we have that the first constraint is inactive and the second constraint is active at the optimum.

Next, we perturb the first constraint as follows:

$$x_1 + x_2 \leq 1.$$

The optimal primal and dual solutions for the modified problem are

$$\begin{aligned} x &= (0, 1), & w &= (0, 1) \\ y &= (3, 0), & z &= (2, 0). \end{aligned}$$

Thus, the first constraint has now become active and the second constraint has become inactive.

Iteration	x	w	y	z
0	(0,2)	(1,0)	(0,3)	(5,0)
1	(0,2)	(0.05,0)	(0,3)	(5,0)
2	(0,2)	(0.0025,0)	(0,3)	(5,0)
3*	(0,2)	(0.000125,0)	(1.6,1.4)	(3.05,0)
4*	(0,2)	(0.00000625,0)	(2.93,0.07)	(2.07,0)

TABLE 1. Variable values for the first four iterations after a cold-start. Iterative refinement was performed on iterations marked by "*".

If we start to solve the modified problem from the initial solution (8), we will run into a range of numerical difficulties. Variable values of initial iterations are provided in Table 1. First, the only change to the reduced KKT system (5) is to primal infeasibility, ρ, for the inactive constraint. The corresponding entry of the diagonal matrix, E, is on the order of 10^8 (theoretically, it is equal to $1/0$). Therefore, a very small value for this constraint's Δy will allow the system to be satisfied. In fact, in the first several iterations, Δx and Δy are both no more than 10^{-8} for all components.

Second, the barrier parameter, μ, starts at 0, as its computation (4) is not affected by a perturbation to the data. Therefore, the step direction formulas for the slack variables result in Δz also being close to 0. This also means that primal feasibility can be re-obtained without disturbing the primal variables, and by simply moving the primal slacks. In the first iteration, we have $\Delta w_1 = -2$ to obtain primal feasibility, but the steplength α_p will be cut to prevent $w_1 + \alpha_p \Delta w_1$ from becoming negative. Therefore, the steplengths will become shortened due to the warmstart.

The numerical trouble occurs when the short steps decrease w_1 to a value sufficiently close to 0. Since y_1 has not moved away from 0, we are now at a degenerate pair. Thus, the Cholesky factorization and the subsequent backsolve start producing numerical errors, and iterative refinement measures are needed. However, such measures provide step directions that move the iterates away from the warmstart solution. In this case, they have the effect of moving the dual variables to values around 1.5 and the dual slack z_1 to around 3.05.

Continuing from this point, the algorithm finds the optimal solution in another 12 iterations, but after losing all the advantage of a warmstart. In fact, the cold start solution of the same problem takes 12 iterations, and the warmstart brings it to 15 iterations.

An appropriate remedy would be to start with a reduced KKT system where the 0's on the diagonal and on the right-hand-side have been perturbed. One way to accomplish this is to shift the values of the primal and dual slack variables. However, suitable shifting values may be hard to determine, and such a shift is somewhat contrary to performing a warmstart. In the next section, we propose the use of an exact penalty approach to provide a suitable perturbation. This penalty approach also relaxes the lower bounds on the slack variables so that steplengths do not get shortened.

4. An Exact Primal-Dual Penalty Method

The ℓ_1 penalty function has received much interest in recent literature, especially in the solution of mathematical programming problems with equilibrium constraints (MPECs). Anitescu [1], Benson et.al. [2], and Leyffer et.al. [15] consider using such an approach to resolve the unboundedness of the set of optimal Lagrange multipliers arising in MPECs. The application of this approach to general nonlinear programming problems are discussed in [2] and by Gould et.al. in [11], and it is shown in [2] that an ℓ_1 penalty approach allows an interior-point solver to detect infeasible problems, find nonKKT optima, and resolve jamming.

The penalized problem considered in this paper has the form

$$
(9) \quad
\begin{array}{llrcccl}
\text{maximize} & & \multicolumn{4}{l}{c^T x - d_x^T \xi_x - d_w^T \xi_w} \\
\text{subject to} & & \multicolumn{3}{l}{Ax + w} & = & b \\
& -\xi_x & \leq & x & & \leq & u_z \\
& -\xi_w & \leq & w & & \leq & u_y \\
& & & \xi_x, \xi_w & & \geq & 0,
\end{array}
$$

where ξ_x and ξ_w are the relaxation variables for the lower bounds on x and w, respectively, and d_x and d_w are their corresponding penalty parameters. Since the ℓ_1 penalty function is exact, we can find sufficiently large but finite values for the penalty parameters for which the solution of (9) matches that of (1).

The form of the problem (9) is different than the usual penalty approaches in the literature, such as [5] and [18], in that upper bounds, u_z and u_y, have also been added to the relaxed variables. Since we assume that the problem (1) has an optimum, setting u_z and u_y sufficiently large will not impact the solution. Thus, for the primal problem, these bounds will not have any serious consequences. They do, however, play a significant role for the dual problem, which has the form

$$
(10) \quad
\begin{array}{llrcccl}
\text{minimize} & & \multicolumn{4}{l}{b^T y + u_y^T \psi_y + u_z^T \psi_z} \\
\text{subject to} & & \multicolumn{3}{l}{A^T y - z} & = & c \\
& -\psi_y & \leq & y & & \leq & d_w - \psi_y \\
& -\psi_z & \leq & z & & \leq & d_x - \psi_z \\
& & & \psi_y, \psi_z & & \geq & 0,
\end{array}
$$

where ψ_y and ψ_z are the relaxation variables for the lower bounds on y and z, respectively. The form of the upper bounds in (10) differs slightly from those in (9), but since the relaxation variables will be driven to zero, the upper bound can be kept sufficiently large. Due to the primal-dual nature exhibited by the relaxing-bounding scheme of (9) and (10), we will call our approach a *primal-dual penalty*. Such an approach is necessary to accommodate perturbations to all data elements of (1), as changes in b impact primal feasibility, changes in c impact dual feasibility, and changes in A impact both primal and dual feasibility.

Note that the resulting problem (9) is still an LP. Therefore, the interior-point algorithm described in Section 2 can be applied to solve this problem. Moreover, we can show that the effort required per iteration is not substantially different than

solving the original LP. The optimality conditions for the penalty problem are

$$
\begin{aligned}
\text{(11a)} && Ax + w &= b \\
\text{(11b)} && A^T y - z &= c \\
\text{(11c)} && (z_j + \psi_{zj})(x_j + \xi_{xj}) &= 0, && j = 1, \ldots, n \\
\text{(11d)} && (y_i + \psi_{yi})(w_i + \xi_{wi}) &= 0, && i = 1, \ldots, m \\
\text{(11e)} && \psi_{zj}(u_{zj} - x_j) &= 0, && j = 1, \ldots, n \\
\text{(11f)} && \psi_{yi}(u_{yi} - w_i) &= 0, && i = 1, \ldots, m \\
\text{(11g)} && \xi_{xj}(d_{xj} - z_j - \psi_{zj}) &= 0, && j = 1, \ldots, n \\
\text{(11h)} && \xi_{wi}(d_{wi} - y_i - \psi_{yi}) &= 0, && i = 1, \ldots, m.
\end{aligned}
$$

After relaxing the complementarity constraints and applying Newton's Method, the step directions can be obtained from the corresponding reduced KKT system for this problem. The system has the same form as (5) with

(12)

$$
\begin{aligned}
E &= \left(\left((Y + \Psi_y)^{-1}(W + \Xi_w) + \Xi_w(D_w - Y - \Psi_y)^{-1} \right)^{-1} + \Psi_y(U_y - W)^{-1} \right)^{-1} \\
F &= \left((Z + \Psi_z)^{-1}(X + \Xi_x) + \Xi_x(D_x - Z - \Psi_z)^{-1} \right)^{-1} + \Psi_z(U_z - X)^{-1} \\
\gamma_w &= \left((Y + \Psi_y)^{-1}(W + \Xi_w) + \Xi_w(D_w - Y - \Psi_y)^{-1} \right)^{-1} \\
&\quad \left(\mu(Y + \Psi_y)^{-1}e - \mu(D_w - Y - \Psi_y)^{-1}e - w \right) - \left(\mu(U_y - W)^{-1}e - \psi_y \right) \\
\gamma_z &= \left((Z + \Psi_z)^{-1}(X + \Xi_x) + \Xi_x(D_x - Z - \Psi_z)^{-1} \right)^{-1} \\
&\quad \left(\mu(Z + \Psi_z)^{-1}e - \mu(D_x - Z - \Psi_z)^{-1}e - x \right) - \left(\mu(U_z - X)^{-1}e - \psi_z \right).
\end{aligned}
$$

The step directions for the relaxation variables can be calculated as follows:

(13)

$$
\begin{aligned}
\Delta\psi_y &= (U_y - W)^{-1} \left(\mu e + \Psi_y \Delta w \right) \\
\Delta\psi_z &= (U_z - X)^{-1} \left(\mu e + \Psi_z \Delta x \right) \\
\Delta\xi_x &= (D_x - Z - \Psi_z)^{-1} \left(\mu e + \Xi_x(\Delta z + \Delta\psi_z) \right) \\
\Delta\xi_w &= (D_w - Y - \Psi_y)^{-1} \left(\mu e + \Xi_w(\Delta y + \Delta\psi_y) \right).
\end{aligned}
$$

In these expressions, Ξ_x, Ξ_w, Ψ_y, and Ψ_z are the diagonal matrices with entries from the relaxation variables ξ_x, ξ_w, ψ_y, and ψ_z, respectively. Similarly, D_x, D_w, U_z, and U_y are the diagonal matrices with entries from the penalty parameters d_x, d_w, u_z, and u_y, respectively. The formulas for the step directions for the original slack variables remain in the same form as (7). The barrier parameter is computed as follows:

(14)
$$
\mu = r \frac{(z + \psi_z)^T(x + \xi_x) + (y + \psi_y)^T(w + \xi_w) + \psi_z^T(u_z - x) + \psi_y^T(u_y - w) + \xi_x^T(d_x - z - \psi_z) + \xi_w^T(d_w - y - \psi_y)}{3m + 3n}
$$

Despite the lengthy expressions, it is clear that E and F are diagonal matrices with nonnegative entries for sufficiently large primal and dual penalty parameters. Therefore, the reduced KKT matrix has the same sparsity structure as (5) and it is quasidefinite. Since the factorization of this matrix is the bottleneck of an interior-point method, using the primal-dual penalty approach does not significantly increase the amount of work required per iteration.

The reduced KKT system of (9) achieves the goal of perturbing the diagonal of the matrix and the right hand side of the equation. It also allows the slack variables of the original primal and dual problems to become negative, preventing

the excessive shortening of steplengths. Therefore, the interior-point algorithm will not stall when warmstarting if the new variables ξ_x, ξ_w, ψ_y, and ψ_z are initialized appropriately.

4.1. **Establishing the Exactness of the Primal-Dual Penalty Formulation.** We mentioned above that setting the penalty parameters large enough to admit the optimal solutions of (1) and (2) would be beneficial. Let us now formally show the exactness of the primal-dual penalty approach.

Theorem 1. *Let (x^*, w^*) be a solution to (1) and let (y^*, z^*) be a corresponding solution for its dual (2). If the following conditions hold*

$$
\begin{array}{ccc}
d_x & > & z^* \\
d_w & > & y^* \\
u_y & > & w^* \\
u_z & > & x^*,
\end{array}
$$

then $(x, w, \xi_x, \xi_w) = (x^*, w^*, 0, 0)$ is a solution to (9) and $(y, z, \psi_y, \psi_z) = (y^*, z^*, 0, 0)$ is a corresponding solution for its dual (10).

Proof. Let (x^*, w^*) be a solution to (1) and let (y^*, z^*) be a corresponding solution for its dual (2). Letting

$$
\begin{array}{ccc}
d_x & > & z^* \\
d_w & > & y^* \\
u_y & > & w^* \\
u_z & > & x^*,
\end{array}
$$

we have that $(x, w, \xi_x, \xi_w) = (x^*, w^*, 0, 0)$ is feasible for (9) and $(y, z, \psi_y, \psi_z) = (y^*, z^*, 0, 0)$ is feasible for its dual (10). Thus, the penalized problem and its dual have feasible solutions, and therefore, they must have optimal solutions. Let a primal optimal solution be denoted by $(\bar{x}, \bar{w}, \bar{\xi}_x, \bar{\xi}_w)$ and let a corresponding dual optimal solution be denoted by $(\bar{y}, \bar{z}, \bar{\psi}_y, \bar{\psi}_z)$, and let the following conditions hold

$$
\begin{array}{ccc}
d_x & > & \bar{z} \\
d_w & > & \bar{y} \\
u_y & > & \bar{w} \\
u_z & > & \bar{x}.
\end{array}
$$

Then, the conditions

$$
\begin{array}{rcll}
\bar{\psi}_{zj}(u_{zj} - \bar{x}_j) & = & 0, & j = 1, \ldots, n \\
\bar{\psi}_{yi}(u_{yi} - \bar{w}_i) & = & 0, & i = 1, \ldots, m
\end{array}
$$

of (11e)-(11f) require that $\bar{\psi}_y = 0$ and $\bar{\psi}_z = 0$. Therefore, we have that

$$
\begin{array}{rcll}
d_{xj} - \bar{z}_j - \bar{\psi}_{zj} & = & d_{xj} - \bar{z}_j, & j = 1, \ldots, n \\
d_{wi} - \bar{y}_i - \bar{\psi}_{yi} & = & d_{wi} - \bar{y}_i, & i = 1, \ldots, m.
\end{array}
$$

and the conditions

$$
\begin{array}{rcll}
\bar{\xi}_{xj}(d_{xj} - \bar{z}_j - \bar{\psi}_{zj}) & = & 0, & j = 1, \ldots, n \\
\bar{\xi}_{wi}(d_{wi} - \bar{y}_i - \bar{\psi}_{yi}) & = & 0, & i = 1, \ldots, m.
\end{array}
$$

of (11g)-(11h) require that $\bar{\xi}_x = 0$ and $\bar{\xi}_w = 0$. The remaining conditions, (11a)-(11d), reduce to (3) which are the optimality conditions for (1). Thus, $(\bar{x}, \bar{w}, \bar{\xi}_x, \bar{\xi}_w) = (x^*, w^*, 0, 0)$ is a solution to (9) and $(\bar{y}, \bar{z}, \bar{\psi}_y, \bar{\psi}_z) = (y^*, z^*, 0, 0)$ is a corresponding solution for its dual (10). $\qquad\square$

Corollary 2. *Let $(\bar{x}, \bar{w}, \bar{\xi}_x, \bar{\xi}_w)$ be an optimal solution to (9), and let $(\bar{y}, \bar{z}, \bar{\psi}_y, \bar{\psi}_z)$ be a corresponding optimal solution to (10). If $\bar{\xi}_x$, $\bar{\xi}_w$, $\bar{\psi}_y$, $\bar{\psi}_z$ all equal 0, then $(\bar{x}, \bar{w})$ is an optimal solution to (1) and $(\bar{y}, \bar{z})$ is a corresponding optimal solution to (2).*

Proof. Setting $\bar{\xi}_x$, $\bar{\xi}_w$, $\bar{\psi}_y$, $\bar{\psi}_z$ equal to 0 reduces the optimality conditions (11a)-(11h) to the optimality conditions (3) of the original problem. $\square$

Note that, for an LP, the sets of optimal primal and dual values are either both empty or both have at least one finitely valued member, and therefore, if an optimal solution exists for the original problem, the penalty parameters can be made sufficiently large for finite values. Therefore, the primal-dual penalty approach is exact.

Also note that the penalized primal and dual problems (9) and (10) will always have optimal solutions. Because the bounds on the primal variables and primal slacks are relaxed, all primal constraints can be satisfied, and therefore the penalized primal problem always has a feasible solution. Since the primal variables and slacks are also bounded above, the penalized primal problem is not unbounded. Thus, the penalized primal problem always has an optimal solution, which means that the penalized dual always has a corresponding optimal solution. As in the proof of Theorem 1, we will refer to the optimal solutions of the penalized primal and dual problems as $(\bar{x}, \bar{w}, \bar{\xi}_x, \bar{\xi}_w)$ and $(\bar{y}, \bar{z}, \bar{\psi}_y, \bar{\psi}_z)$, respectively.

Certificates of infeasibility for the original primal and/or dual problems can be obtained by examining the optimal solution of the penalized problem. If any components of ξ_x or ξ_w cannot be driven to zero even as the corresponding components of d_x and d_w are driven to ∞, then the problem is primal infeasible. Similarly, if any components of ψ_y and ψ_z cannot be driven to zero as the corresponding components of u_y and u_z are increased to ∞, then the problem is dual infeasible. We can get these certificates by repeatedly solving the problem with increasing values of the penalty parameters if

$$\|\bar{\xi}_x\| + \|\bar{\xi}_w\| + \|\bar{\psi}_y\| + \|\bar{\psi}_z\| > 0.$$

The scheme of updating the penalty parameters and resolving the problem only after an unsatisfactory solution has been obtained is referred to as a *static* penalty approach. Later in this section, we will also discuss a *dynamic* penalty approach, where the parameters may be updated from one iteration to the next.

Any provably convergent interior-point method for LP can solve the penalized primal-dual problems (9) and (10) to an optimal solution. The algorithm described in Section 2 is such an algorithm, as proved by Kojima et.al. in [13]. Therefore, it can find solutions to the original primal-dual problems (1) and (2) or provide certificates of infeasibility when coupled with a static approach to updating the penalty parameters.

4.2. **Computational Issues.** Two questions still remain:
  (1) How do we initialize the penalty parameters, d_x, d_w, u_y, and u_z, and the relaxation variables, ξ_x, ξ_w, ψ_y, and ψ_z, in order to reach the new optimum quickly after a warmstart?
  (2) If necessary, how do we update the penalty parameters in order to obtain the solution to the perturbed LP?

Before answering these questions, we first formally establish some notation and terminology. We will refer to the *initial problem* as the first LP to be solved and the *perturbed problem* as any subsequent problem to be solved. We assume that the initial problem has the optimal primal-dual solution pairs (x^I, w^I) and (y^I, z^I), which we can use to warmstart, and that if the perturbed problem has an optimal solution, such a primal-dual pair is denoted by (x^P, w^P) and (y^P, z^P). The perturbed problem will be solved using the primal-dual penalty approach outlined in the previous section, and we will refer to that problem as the *penalized problem*. The penalized problem will have the primal-dual solutions $(\bar{x}, \bar{w}, \bar{\xi}_x, \bar{\xi}_w)$ and $(\bar{y}, \bar{z}, \bar{\psi}_y, \bar{\psi}_z)$. The initial values of the relaxation variables in the perturbed problem will be denoted by $\xi_x^{(0)}, \xi_w^{(0)}, \psi_y^{(0)}$, and $\psi_z^{(0)}$. Also, we will refer to the data elements of the perturbed problem as $\tilde{b}, \tilde{A}$, and $\tilde{c}$. We will also let $\tilde{l}$ denote the new, possibly nonzero, lower bounds on the variable x.

To determine the best initial values for the relaxation variables, our goal is to allow the nonnegative variables of the perturbed problem to become negative so that progress may be made in the first few iterations. Therefore, we set them proportional to the perturbation, as follows:

$$
(15) \quad
\begin{aligned}
\xi_{xj}^{(0)} &= \max(\tilde{l}_j - x_j^I, 0) + \epsilon, & j &= 1, \ldots, n \\
\xi_{wi}^{(0)} &= \max(\tilde{b}_i - \sum_{j=1}^{n} \tilde{a}_{ij} x_j^I - w_i^I, 0) + \epsilon, & i &= 1, \ldots, m \\
\psi_{yi}^{(0)} &= \epsilon, & i &= 1, \ldots, m \\
\psi_{zj}^{(0)} &= \max(\tilde{c}_j - \sum_{i=1}^{m} \tilde{a}_{ij} x_j^I + z_j^I, 0) + \epsilon, & j &= 1, \ldots, n,
\end{aligned}
$$

where ϵ is a small parameter, currently set to $10^{-5}M$, where M is the greater of 1 and the largest primal or dual slack value. The magnitude has been determined to ensure that the Cholesky factorization remains stable if there is no perturbation to either the primal or the dual, while being small enough to reach the optimal solution in one or two iterations if it has not changed by much. Consideration of the primal and dual slack values also allows for a consideration of scale.

As mentioned in Theorem 1, we need the following conditions to hold to find an optimal solution to the perturbed problem using the primal-dual penalty approach

$$
\begin{aligned}
d_x &> z^P \\
d_w &> y^P \\
u_y &> w^P \\
u_z &> x^P
\end{aligned}
$$

if the perturbed problem has a solution. Otherwise, we need to increase at least one of the penalty parameters indefinitely to provide a certificate of infeasibility.

In the absence of any information on the magnitudes of the optimal solutions to the perturbed problem, one may consider setting all penalty parameters to very large values. Nevertheless, doing so may cause numerical instability and does not guarantee that the penalty parameters are large enough or that the perturbed problem is not primal and/or dual infeasible.

However, for a warmstart, we can use the solution of the initial problem to set the penalty parameters for the perturbed problem. If the perturbation is small enough, it is reasonable to assume that in most cases the new primal and dual

solutions will exist and be of the same order of magnitude as those of the initial problem. Since the active set can change, we use the following initialization scheme for the penalty parameters:

$$
\begin{array}{rcl}
d_x & = & 10(z^I + \delta) \\
d_w & = & 10(y^I + \delta) \\
u_y & = & 10(w^I + \psi_y^{(0)} + \delta) \\
u_z & = & 10(x^I + \psi_z^{(0)} + \delta),
\end{array}
$$
(16)

where δ is a constant set to 100.

For the static penalty approach, we can increase the penalty parameters and resolve the problem if the optimal values of at least one relaxation variable is sufficiently larger than 0. One way to implement this is to multiply the penalty parameters by 10 and resolve the problem.

For the dynamic penalty approach, we need to be able to detect that the penalty parameters are not large enough while the penalized problem is being solved. As shown in (9) and (10), the primal and dual penalty parameters act as upper bounds on the dual and primal variables. If one or more variables among x, w, y, and z are converging to their upper bounds, (11c)-(11h) require that these variables cannot provide a feasible solution to the original problem. Therefore, the corresponding penalty parameters must be increased. Doing so will ensure that the penalty parameters are larger than the optimal values of these variables, which is sufficient for exactness. Supplementing the interior-point algorithm with such an updating scheme retains its convergence properties.

Theorem 3. *If the perturbed problem has an optimal solution, a dynamic penalty update approach will be able to find this solution after a finite number of penalty parameter updates. Otherwise, the dynamic penalty update approach will provide a certificate of infeasibility.*

Proof. Let us assume that the perturbed problem has an optimal solution and that at least one of the penalty parameters is not large enough to admit this solution. There still exists an optimal solution to the penalized problem, but at this solution, at least one of the relaxation variables will be nonzero, as shown by Theorem 1. For the relaxation variable that is nonzero, the corresponding decision or slack variable must be equal to its upper bound in order to satisfy the optimality conditions (11e)-(11h). Therefore, the dynamic update approach will not stop at such a solution and will instead update the penalty parameter until it is large enough to admit the solution of the perturbed problem.

Note that there are a finite number of penalty parameters, so the dynamic penalty updating scheme cannot update the penalty parameters indefinitely and must either stop the updates after a finite number of times or have one or more of the penalty parameters approach ∞. If the updates stop after a finite number of times, then according to the above argument, all the penalty parameters must be large enough to admit the solution of the perturbed problem. Then, as shown in [13], the algorithm described in Section 2 will converge to the optimal solution of the penalized problem, which is the optimal solution of the perturbed problem.

If the penalty parameters are updated an infinite number of times, then at least one of the decision or slack variables must approach infinity. Therefore, the perturbed problem must be infeasible, and a certificate of infeasibility can be issued. □

The particular scheme implemented in the code for updating penalty parameters checks each of $x, w, y,$ and z against their upper bounds, which are the corresponding primal or dual penalty parameters. If one of these variables is greater than one-tenth of its upper bound, the penalty parameter is increased tenfold. Although this scheme may require an excessive distance between the variable and its bound, it has worked well in practice. The details of the exact primal-dual penalty approach with a dynamic update are provided as Algorithm 1.

Algorithm 1. *Initialize the solution* $x = x^{(0)}$, $w = w^{(0)}$, $y = y^{(0)}$, $z = z^{(0)}$. *Initialize the relaxation variables using* (15). *Initialize the penalty parameters using* (16). *Let* $k = 1$, $r = 0.95$, $\tau = 10^{-8}$.

Step 1: *Compute* $\mu^{(k)}$ *using* (14).

Step 2: *Obtain step directions* $(\Delta x^{(k)}, \Delta w^{(k)}, \Delta \xi_x^{(k)}, \Delta \xi_w^{(k)})$ *and* $(\Delta y^{(k)}, \Delta z^{(k)},$ $\Delta \psi_y^{(k)}, \Delta \psi_z^{(k)})$ *by solving* (5) *and* (7) *with* (12) *and* (13).

Step 3: *Compute the primal and dual steplengths,* $\alpha_P^{(k)}$ *and* $\alpha_D^{(k)}$ *as follows:*

$$
\alpha_P^{(k)} = \max\left(r \max_{i,j}\left(\frac{\xi_{xj}^{(k)}}{\Delta \xi_{xj}^{(k)}}, \frac{\xi_{wi}^{(k)}}{\Delta \xi_{wi}^{(k)}}, \frac{(x_j^{(k)}+\xi_{xj}^{(k)})}{(\Delta x_j^{(k)}+\Delta \xi_{xj}^{(k)})}, \frac{(w_i^{(k)}+\xi_{wi}^{(k)})}{(\Delta w_i^{(k)}+\Delta \xi_{wi}^{(k)})} \right), 1 \right).
$$
$$
\alpha_D^{(k)} = \max\left(r \max_{i,j}\left(\frac{\psi_{yi}^{(k)}}{\Delta \psi_{yi}^{(k)}}, \frac{\psi_{zj}^{(k)}}{\Delta \psi_{zj}^{(k)}}, \frac{(y_i^{(k)}+\psi_{yi}^{(k)})}{(\Delta y_i^{(k)}+\Delta \psi_{yi}^{(k)})}, \frac{(z_j^{(k)}+\psi_{zj}^{(k)})}{(\Delta z_j^{(k)}+\Delta \psi_{zj}^{(k)})} \right), 1 \right).
$$

Step 4 (Dynamic Update): *For* $j = 1, \ldots, n$,

$$
\begin{aligned}
\text{If } d_{xj} &< 10(z_j^{(k)} + \Delta z_j^{(k)}), \text{ then } d_{xj} = 10 d_{xj}, \text{ and} \\
\text{if } u_{zj} &< 10(x_j^{(k)} + \Delta x_j^{(k)}), \text{ then } u_{zj} = 10 u_{zj}.
\end{aligned}
$$

For $i = 1, \ldots, m$,

$$
\begin{aligned}
\text{If } d_{wi} &< 10(y_i^{(k)} + \Delta y_i^{(k)}), \text{ then } d_{wi} = 10 d_{wi}, \text{ and} \\
\text{if } u_{yi} &< 10(w_i^{(k)} + \Delta w_i^{(k)}), \text{ then } u_{yi} = 10 u_{yi}.
\end{aligned}
$$

Step 5: *Let*

$$
\begin{aligned}
x^{(k+1)} &= x^{(k)} + \alpha_P^{(k)} \Delta x^{(k)} \\
w^{(k+1)} &= w^{(k)} + \alpha_P^{(k)} \Delta w^{(k)} \\
\xi_x^{(k+1)} &= \xi_x^{(k)} + \alpha_P^{(k)} \Delta \xi_x^{(k)} \\
\xi_w^{(k+1)} &= \xi_w^{(k)} + \alpha_P^{(k)} \Delta \xi_w^{(k)} \\
y^{(k+1)} &= y^{(k)} + \alpha_D^{(k)} \Delta y^{(k)} \\
z^{(k+1)} &= z^{(k)} + \alpha_D^{(k)} \Delta z^{(k)} \\
\xi_y^{(k+1)} &= \xi_y^{(k)} + \alpha_D^{(k)} \Delta \xi_y^{(k)} \\
\xi_z^{(k+1)} &= \xi_z^{(k)} + \alpha_D^{(k)} \Delta \xi_z^{(k)}.
\end{aligned}
$$

If (11a)-(11h) *are satisfied within tolerance* τ, *then Stop.* $(x^{(k)}, w^{(k)}, \xi_x^{(k)}, \xi_w^{(k)})$ *is the optimal primal solution and* $(y^{(k)}, z^{(k)}, \psi_y^{(k)}, \psi_z^{(k)})$ *is the optimal dual solution. Otherwise, let* $k = k + 1$ *and go to Step 1.*

Computationally, it is not possible to update penalty parameters an infinite number of times. Therefore, in the code, we update the penalty parameter three times before discarding the original objective function and minimizing the infeasibility. If the penalty objective is sufficiently close to zero, then the problem continues from the current solution with the largest value of the penalty parameter encountered. Otherwise, a certificate of infeasibility can be issued. Note that this scheme can

also be performed to detect dual infeasibility by discarding the objective function on the dual side.

4.3. **A Simpler Model.** We can achieve the goal of relaxing the primal and dual constraints using a simpler set of models. The primal problem can be expressed as follows:

$$
\begin{array}{llrcl}
\text{maximize} & & c^T x - d^T \xi \\
\text{subject to} & & Ax + w & = & b \\
& 0 \leq & x & \leq & u \\
& -\xi \leq & w & & \\
& & \xi & \geq & 0,
\end{array}
$$

and the dual problem is

$$
\begin{array}{llrcl}
\text{minimize} & & b^T y + u^T \psi \\
\text{subject to} & & A^T y - z & = & c \\
& 0 \leq & y & \leq & d \\
& -\psi \leq & z & & \\
& & \psi & \geq & 0.
\end{array}
$$

The reduced KKT system for this pair of problems has the same form as (5), with

$$
\begin{aligned}
E &= Y^{-1}(W + \Xi) + \Xi(D - Y)^{-1} \\
F &= (Z + \Psi)^{-1}X + \Psi(U - X)^{-1} \\
\gamma_w &= \left(Y^{-1}(W + \Xi) + \Xi(D - Y)^{-1}\right)^{-1}\left(\mu Y^{-1}e - \mu(D - Y)^{-1}e - w\right) \\
\gamma_z &= \mu X^{-1}e - \mu(U - X)^{-1}e - z
\end{aligned}
$$

All the same theoretical results presented above apply to this simpler model. However, the numerical performance of this approach is not stable, as any entry of E corresponding to a former inactive constraint or any entry of F corresponding to a former active bound would start at a very high value. For some problems, this causes numerical difficulties right after the warmstart and increases the number of iterations needed to reach the optimum. Therefore, we have used the formulation (9)-(10) as our approach. In the next section, we provide testing results on the implementation of this approach, coupled with a dynamic updating of the penalty parameters.

5. Numerical Results

The penalty approach presented in the previous section was able to solve the example of Section 3 in 8 iterations. This is an improvement over the 12 iterations performed with a coldstart and remedies the shortcomings of the original method which resulted in 15 iterations after a warmstart. In this section, we provide further results to illustrate the performance of our approach.

For perturbation of the data, we performed our numerical testing on a selection of 37 problems from the Netlib test suite. The main selection criterion was the size of the problem, in that we considered only those problems with $n + m < 1000$. We also eliminated all instances where the perturbation did not change the solution significantly or resulted in an infeasible problem. We solved these problems using the solver LOQO Version 6.06 [20] which implements the algorithm described in Section 2. The problems, expressed in MPS format, were submitted directly to the solver, and no preprocessing was performed.

Changed	δ	Avg Warmstart Iters	Avg Coldstart Iters	Avg Reduction
b	0.001	10.38	21.76	52%
b	0.01	10.41	21.67	52%
b	0.1	12.07	21.58	44%
c	0.001	11.55	22.74	49%
c	0.01	11.78	22.09	47%
c	0.1	13.00	22.21	41%
A	0.001	12.01	21.78	45%
A	0.01	12.41	21.78	43%
A	0.1	13.95	21.87	36%

TABLE 2. Summary numerical results for penalty warmstart approach.

The problems were perturbed randomly. The random numbers were obtained using the random number generator Mersenne Twister [17], and we generated a list of numbers uniformly distributed between -1 and 1. For the case of perturbing the right hand sides of the constraints, two vectors of length m each were read in. The first vector, ϵ_1, determined whether a constraint would be perturbed. Constraint i was perturbed if

$$\epsilon_{1i} < \text{MIN}(-0.80, \frac{40}{m} - 1),$$

which ensures that no more than 10% or 20 constraints (on the average) will be perturbed. The second vector, ϵ_2, was used to determine the magnitude of the perturbation. In order to match the scale of the numbers in the problem, the following scheme was used:

$$\tilde{b_i} = \left\{ \begin{array}{ll} \delta\epsilon_{2i}, & \text{if } b_i = 0 \\ b_i(1 + \delta\epsilon_{2i}), & \text{otherwise.} \end{array} \right.$$

when perturbing constraint i. The parameter δ was varied to observe the effects of different levels of perturbation. A similar scheme was employed to perturb the objective coefficients, c, and constraint coefficients, A. The only difference with the constraint coefficient updates is that the sparsity structure of A was preserved.

For each original problem, we present the numerical results obtained by 5 different perturbations to b, c, and A at three different levels of δ (0.001, 0.01, 0.1). We report on the distance between the optimal solutions to the original and perturbed problems, along with the number of changes to the active set, and compare iteration counts for the warmstart and the coldstart solutions. These results are summarized in Table 2 and presented in detail in the Appendix.

The distance between the solutions is computed as

$$\frac{\|x^I - x^P\|}{1.0 + \|x^I\|},$$

and the distance between the Lagrange multipliers is computed as

$$\frac{\|y^I - y^P\|}{1.0 + \|y^I\|}.$$

When the distance between the solutions is small, generally 10^{-2} or less, the warmstart scheme works very well, as evidenced by the decrease in the number of iterations from solving the perturbed problem with a coldstart. However, when the distance is large, such as $e226$ in Table 3 of the Appendix, the performance deteriorates. Nonetheless, such a distance represents a significant enough change in the problem that there is no reason to perform a warmstart.

We also implemented a branch-and-bound algorithm to solve mixed-integer linear programming (MILP) problems, using LOQO to solve the linear relaxation at each node of the tree. While not being competitive with a MILP solver, we were able to examine the performance of our approach on warmstarting after variable bound perturbations. Table 9 shows the results of this exercise. We chose three small problems, a four-choice version of *diet* from [6], *hl415* from [12], and *synthes3* originally from [14] and linearly formulated by Hans Mittelman. The results present a comparison of the iteration counts between a coldstart, which goes back to the original initial solution, and a warmstart, which starts the node from the optimal solution of its parent.

We have made several observations during the solution of the MILPs. First, all three problems included binary variables, and for such a problem, changing variable bounds only changes the slacks for the bounds by at most 1. Therefore, letting the initial values of the relaxation variables on the bounds and δ for the corresponding penalty parameters equal 1 worked the best. Second, scaling issues arise which can greatly affect the number of iterations or whether the problem can be solved at all. This behavior is due to the fact that an interior-point method stops when it satisfies the optimality conditions within a small tolerance, and, as in many codes, LOQO scales primal and dual infeasibilities and the duality gap by $(\|b\| + 1)$, $(\|c\| + 1)$, and $(|c^T x| + 1)$, respectively. When any of these data elements are large, even binary variables which may be fixed at 0 or 1 can take on non-integer "optimal" values. This problem is easy to remedy for the case of a binary variable by preprocessing the fixed variable out of the problem, which we have implemented for the coldstart results shown in this paper. Another related problem, however, is that due to the magnitude of the objective function, penalty parameters may need to be quite large to ensure that the objective function in the original variables will not improve significantly in exchange for a small penalty. Since a dynamic update of the penalty parameter will take several iterations to detect this behavior and increase the penalty parameter sufficiently, scaling issues can greatly affect the performance of the primal-dual penalty approach. While this behavior can occur for any problem, it is especially prevalent in MILPs which involve cost-minimization objectives. The fixed cost portion of such an objective can be quite large (a magnitude of 10^5 is not unusual), and in numerical testing on larger problems not documented in this paper, we have found it beneficial to rescale the objective function. We have employed such an approach to improve the performance of two such models here: we rescaled the range constraints of *diet* and the objective function of *synthes3* by a factor of 1000. The coldstart results are also for these rescaled problems.

As stated above, we do not expect our branch-and-bound code to be competitive with other MILP solvers, but it may be improved by not solving the intermediate subproblems to optimality. In [8] and [10], Gondzio and Vial present an approach where nearly-optimal well-centered points are obtained for the subproblems, and report substantial improvements in iteration counts when warmstarting. Subsequent work on MILP would benefit from consideration of and comparisons to this approach.

Finally, we provide an example to illustrate the effects of adding variables and/or constraints to the problem. The cutting-stock, or roll-trimming, problem as formulated in [6] provides a column generation approach, implemented using looping

constructs within AMPL. A linear relaxation of the master problem is solved with subsequently increasing numbers of variables. The new variables are obtained by solving a different integer programming problem. Each new variable in the master problem is a new constraint in its dual. For the warmstart, we use the optimal solution of the previous master problems's primal and dual variable values and re-calculate the slacks. The new primal variables are started at initial values of 0. The numerical results are presented in Table 10.

6. Conclusion

In this paper, we discussed the use of an exact penalty approach to perform warmstarts using an interior-point method. We showed that without such an approach, an interior-point method can stall when starting from the optimal solution of a closely related problem and run into numerical troubles. The penalty approach provides a regularization to prevent this behavior, and by carefully controlling the penalty parameter, we can get fast convergence to the solution of the perturbed problem. The approach is similar to using a shifted barrier approach, except the shift is a variable that is driven to 0. The numerical performance of this approach on the problems from the Netlib test suite and on a group of mixed integer linear programming problems shows an average decrease of 40-50% in the number of iterations, which is quite encouraging. The average decrease is around 65-70% if the primal and the dual solutions change by no more than a scaled difference of 10^{-3}.

As mentioned in the introduction, the exact penalty approach has been useful for nonlinear programming to handle ill-behaved problems. A natural extension of the work in this paper is to the case of nonlinear programming, and we hope to report on our findings in the near future.

Finally, it should be emphasized that there is a definite lack of test sets for re-optimization purposes. We encourage researchers to submit problem instances with meaningful data perturbations. Such a test set will be especially important for nonlinear programming, where randomly changing the data may not be possible.

References

[1] Mihai Anitescu. Nonlinear programs with unbounded lagrange multiplier sets. Technical Report ANL/MCS-P793-0200, Argonne National Labs.

[2] H. Y. Benson, A. Sen, D.F. Shanno, and R. J. Vanderbei. Interior point algorithms, penalty methods and equilibrium problems. *Computational Optimization and Applications*, 2005. (to appear).

[3] H.Y. Benson, D.F. Shanno, and R.J. Vanderbei. Interior-point methods for nonconvex nonlinear programming: Jamming and comparative numerical testing. *Mathematical Programming A*, 99(1):35–48, 2004.

[4] A.V. Fiacco and G.P. McCormick. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*. Research Analysis Corporation, McLean Virginia, 1968. Republished in 1990 by SIAM, Philadelphia.

[5] R. Fletcher. *Practical Methods of Optimization*. J. Wiley and Sons, Chichester, England, 1987.

[6] R. Fourer, D.M. Gay, and B.W. Kernighan. *AMPL: A Modeling Language for Mathematical Programming*. Scientific Press, 1993.

[7] R.M. Freund. Theoretical efficiency of a shifted barrier function algorithm for linear programming. *Linear Algebra and Its Applications*, 152:19–41, 1991.

[8] J. Gondzio. Warm Start of the Primal-Dual Method Applied in the Cutting Plane Scheme. *Mathematical Programming*, 83:125–143, 1998.

[9] J. Gondzio and A. Grothey. Reoptimization with the primal-dual interior point method. *SIAM Journal on Optimization*, 13(3):842–864, 2003.

[10] J. Gondzio and J.-Ph. Vial. Warm Start and ϵ-subgradients in Cutting Plane Scheme for Block-Angular Linear Programs. *Computational Optimization and Applications*, 14:17–36, 1999.

[11] N.I.M. Gould, D. Orban, and Ph.L. Toint. An interior-point l1-penalty method for nonlinear optimization. Technical Report RAL-TR-2003-022, Rutherford Appleton Laboratory Chilton, Oxfordshire, UK, November 2003.

[12] F.S. Hillier and G.J. Lieberman. *Introduction to Mathematical Programming*. McGraw-Hill, New York, 2 edition, 1977.

[13] M. Kojima, N. Megiddo, and S. Mizuno. A primal-dual infeasible-interior-point algorithm for linear programming. *Mathematical Programming*, 61:263–400, 1993.

[14] S. Leyffer. Integrating SQP and branch-and-bound for mixed integer nonlinear programming. Technical Report NA-182, Department of Mathematics, University of Dundee, August 1998.

[15] S. Leyffer, G. Lopez-Calva, and J. Nocedal. Interior methods for mathematical programs with complementarity constraints. Technical Report OTC 2004-10, Northwestern University, Evanston, IL, December 2004.

[16] I.J. Lustig, R.E. Marsten, and D.F. Shanno. Interior point methods for linear programming: computational state of the art. *ORSA J. on Computing*, 6:1–14, 1994.

[17] M. Matsumoto and T. Nishimura. Mersenne twister: A 623-dimensionally equidistributed uniform pseudorandom number generator. *ACM Transactions on Modeling and Computer Simulation*, 8(1):3–30, 1998.

[18] S.G. Nash and A. Sofer. *Linear and Nonlinear Programming*. McGraw-Hill, New York, 1996.

[19] R. Polyak. Modified barrier functions (theory and methods). *Math. Programming*, 54:177–222, 1992.

[20] L.A. Shepp and R.J. Vanderbei. The complex zeros of random polynomials. *Transactions of the AMS*, 347(11):4365–4384, 1995.

[21] R.J. Vanderbei. LOQO: An interior point code for quadratic programming. *Optimization Methods and Software*, 12:451–484, 1999.

[22] R.J. Vanderbei and D.F. Shanno. An interior-point algorithm for nonconvex nonlinear programming. *Computational Optimization and Applications*, 13:231–252, 1999.

[23] E.A. Yildirim and S.J. Wright. Warm-start strategies in interior-point methods for linear programming. *SIAM Journal on Optimization*, 12(3):782–810, 2002.

Problem	δ	#Pert	WarmIters	ColdIters	$\|x^I - x^P\|$	$\|y^I - y^P\|$	Δ
adlittle	0.001	5.6	2.6	18	0.003202181752	0.000014586238	2
adlittle	0.01	5.6	3.8	17.8	0.036400409638	0.011466355782	4
adlittle	0.1	5.6	11	17	0.3924837644	1.813057443798	7.8
afiro	0.001	2.4	3.2	12	0.0011678264504	0.0947489358	1.2
afiro	0.01	2.4	3.2	12	0.0007989962986	0.089257872	1.4
afiro	0.1	2.4	3.6	12	0.014892467386	0.20098398	1.2
agg2	0.001	20.2	5	25.2	0.00034187646	0.00022341066	0
agg2	0.01	20.2	5	25.4	0.0034148048	0.00025784402	0
agg2	0.1	20.2	9.2	25.4	0.03478068	0.173200086	0
agg3	0.001	20.2	5	24	0.00044449582	0.00052089212	0.2
agg3	0.01	20.2	6.6	24	0.004965259	0.25230334344	11.6
agg3	0.1	20.2	9.6	24	0.04459604	0.92526735	23.4
bandm	0.001	20.2	10.4	19	0.000123982844	1.0857212	6.2
bandm	0.01	20.2	10.6	19.2	0.0012381135	0.9262959	0.8
bandm	0.1	20.2	11.2	20	0.0110012632	0.9239741	3.4
blend	0.001	7.8	4	16	0.000303079904	0.00007971808	0
blend	0.01	7.8	4.6	16	0.0030224	0.00050603523	2.6
blend	0.1	7.8	8.4	16	0.0330820446	0.0322738007	0.8
boeing1	0.001	19.8	17	27.6	0.0039221134	0.40535092	44.4
boeing1	0.01	19.8	17	27.4	0.0041235082	0.71336096	21.4
boeing1	0.1	19.8	17.4	27.8	0.0184638934	2.185173	20.8
capri	0.001	19.8	17.2	25.2	0.00011520664	0.0104491832	3
capri	0.01	19.8	17	25	0.00083342632	0.0104612872	3
capri	0.1	19.8	17.6	25.8	0.0107436474	0.013281188	5.2
e226	0.001	19.6	21.4	24.6	17.7051726	1.96924962	2.2
e226	0.01	19.6	18	23.4	14.1649736	0.81973026	3
e226	0.1	19.6	18.4	22.8	6.6063288	1.55621424	7.8
grow15	0.001	20.6	12	21	0.060740534	0.0000004200672	0
grow15	0.01	20.6	12	21	0.060741118	0.0000004200672	0
grow15	0.1	20.6	12	21	0.06073097	0.0000004200673	0
grow7	0.001	14.2	7	20	0.0011366912	0.00000008207383	0
grow7	0.01	14.2	7	20	0.001136728	0.00000008207383	0
grow7	0.1	14.2	7	20	0.0011370982	0.000000082073828	0
israel	0.001	17.4	7	29	0.0028336566	0.0000000104360078	0
israel	0.01	17.4	8.2	29	0.0087307272	0.000000204793846	5.2
israel	0.1	17.4	15.8	29	0.0358459662	0.018162129758424	16.8
kb2	0.001	4.2	4	17	0.00000240630814	0.000000043956492	2
kb2	0.01	4.2	4	17	0.0000240627742	0.000000043692692	2
kb2	0.1	4.2	4.2	16.8	0.000240628326	0.0000000508571	1.6
lotfi	0.001	15.2	26.2	38	1.7512536	0.07372925	1.4
lotfi	0.01	15.2	25.6	38.4	1.7343354	0.084457026	10.4
lotfi	0.1	15.2	25.8	37.8	1.446603	0.107815326	7
sc105	0.001	9.6	8.2	15.4	0.00000400174778	0.38601766	4
sc105	0.01	9.6	8.2	15.2	0.000040045703	0.73229842	3.6
sc105	0.1	9.6	9.2	15.2	0.001181680612	0.91226018	4.2
sc205	0.001	18.4	11.4	17	0.0000423845518	2.2519524	6.4
sc205	0.01	18.4	11.2	17	0.00042382646	1.95001354	6.4
sc205	0.1	18.4	10	17	0.0042103408	2.34346894	9.2
sc50a	0.001	4.8	5.2	13.8	0.0000104318696	0.04535423749	4.4
sc50a	0.01	4.8	5.6	13.8	0.000104337232	0.15409781758	2.6
sc50a	0.1	4.8	7.6	13.8	0.00104335806	3.168897931354	2.8
sc50b	0.001	4.8	3	12	0.0000033665062	0.00009285385	12
sc50b	0.01	4.8	3	12	0.000033741186	0.000092854346	12.4
sc50b	0.1	4.8	3.6	12	0.00033751108	0.000093370066	4.8
scagr25	0.001	20.4	9.8	29.2	0.000128856676	0.00683496614	20.6
scagr25	0.01	20.4	11.2	29.4	0.00132497322	0.0085391073	11
scagr25	0.1	20.4	15.6	28.4	0.013220882	0.01434111488	18.8
scagr7	0.001	13	18.2	26	0.000382898824	0.000000089276834	0.8
scagr7	0.01	13	18	25.8	0.00342266954	0.00032196158677	0.4
scagr7	0.1	13	18	25.8	0.03173249408	0.00042891180376	0.8

TABLE 3. Numerical performance of LOQO on the Netlib test suite when warmstarting a problem whose constraint right-hand sides, b, have been modified. Each problem labeled under the column Problem, δ is the perturbation parameter, and #Pert is the number of perturbed constraints. The average iteration counts for the warmstart and coldstart solution of five perturbed problems are given, and the last 3 columns are the scaled Euclidean norm of the distance between the optimal solutions and the Lagrange multipliers of the initial and the perturbed problem and the number of changes to the active set.

Problem	δ	#Pert	WarmIters	ColdIters	$\|x^I - x^P\|$	$\|y^I - y^P\|$	Δ
scfxm1	0.001	22	14.6	26.2	0.02370614	3.798183	29.4
scfxm1	0.01	22	14.6	26.4	0.030488384	4.415954	16
scfxm1	0.1	22	14.6	26.2	0.05384319	4.4868972	8.6
scsd1	0.001	8.2	9.4	16.6	0.096794814	0.111085056	34.6
scsd1	0.01	8.2	8.8	16.4	0.098801772	0.111035914	34.6
scsd1	0.1	8.2	12.8	15.6	0.21426694	0.128279912	35.4
sctap1	0.001	20.6	8	20.8	0.042360862	0.22468786	48
sctap1	0.01	20.6	8	20.6	0.041823544	0.22457918	51
sctap1	0.1	20.6	12.8	19	0.14872671	0.22883418	52.6
share1b	0.001	11.6	18	41.4	0.0000731204072	0.000000059338672	0
share1b	0.01	11.6	18	40.2	0.000731183464	0.000000099496234	0
share1b	0.1	11.6	18.6	41	0.00747868162	0.0080980673404	2.6
share2b	0.001	8.8	5.8	15.4	0.0101454252	0.0002785847	2.4
share2b	0.01	8.8	6	15.6	0.006371865	0.00040580978	2
share2b	0.1	8.8	6.8	15.6	0.0256612606	0.00041397668	0.6
stair	0.001	20.6	21	19	3.2124368	0.005843676	0.2
stair	0.01	22.6	20.4	19	3.1126782	0.0057625792	0.2
stair	0.1	22.6	20.2	19.8	3.3041412	0.1209035974	6.8
stocfor1	0.001	11.6	5.6	18	0.0000200767266	0.04280102012	5.2
stocfor1	0.01	11.6	5.4	18.2	0.000200754128	0.042390892396	4.6
stocfor1	0.1	11.6	5	17.8	0.00200753612	0.042389969544	4.6

TABLE 4. Numerical performance of LOQO on the Netlib test suite when warmstarting a problem whose constraint right-hand sides, b, have been modified. Each problem labeled under the column Problem, δ is the perturbation parameter, and #Pert is the number of perturbed constraints. The average iteration counts for the warmstart and coldstart solution of five perturbed problems are given, and the last 3 columns are the scaled Euclidean norm of the distance between the optimal solutions and the Lagrange multipliers of the initial and the perturbed problem and the number of changes to the active set.

Problem	δ	#Pert	WarmIters	ColdIters	$\|x^I - x^P\|$	$\|y^I - y^P\|$	Δ
adlittle	0.001	9	5	18.2	0.135710876	0.01638875038	7.4
adlittle	0.01	9	6.2	18.2	0.145065362	0.0394711904	0
adlittle	0.1	9	8.2	16.4	0.223525338	0.121248116	1
afiro	0.001	2.8	3.6	12.2	0.12707564555828	0.036102098	0.4
afiro	0.01	2.8	3.8	12	0.12707638895	0.01945450558	0
afiro	0.1	2.8	4.4	12	0.1270766100472	0.0553325986	0
agg	0.001	15.8	12	23	0.035645717	1.665991	0
agg	0.01	15.8	12.4	23	0.1803810822	4.3241394	0
agg	0.1	15.8	13.8	23	0.1996938048	10.2106634	0
agg2	0.001	20.6	26.2	53.6	0.094664568	0.0302227282	15.6
agg2	0.01	20.6	12.2	26.6	0.183489666	2.825332006	2
agg2	0.1	20.6	13.8	26.6	0.31286746	5.07462946	28.8
agg3	0.001	20.6	10.2	27.4	0.133549792	0.98392198	32.4
agg3	0.01	20.6	14	25.4	0.183808204	4.7505676	33
agg3	0.1	20.6	14.2	25.2	0.30053	25.5355478	30.4
bandm	0.001	20.2	10	19	0.000000225161542	0.97298358	1.8
bandm	0.01	20.2	9.8	19	0.0041196892616	1.13319632	3.2
bandm	0.1	20.2	11.8	19.6	0.014135233895302	0.64995658	5.8
beaconfd	0.001	20	3.6	14	0.097714860852	0.61656227446	0
beaconfd	0.01	20	6.4	14	0.141751120816	8.98905990512	0.4
beaconfd	0.1	20	7	14	0.164060189734	0.47591091454	0
blend	0.001	8.6	4.2	16	0.00544061752	0.000530666724	2.6
blend	0.01	8.6	4.6	16	0.005477636028	0.00199724728	0
blend	0.1	8.6	7.6	16	0.02731423806	0.0226377252	1
boeing1	0.001	20.2	16	27.4	0.012759034	2.2167058	28
boeing1	0.01	20.2	16.6	27.6	0.0147586332	2.2220476	1.6
boeing1	0.1	20.2	16.6	28	0.081041606	2.1617016	39.6
boeing2	0.001	14.6	15.2	21	0.1844679	6.9880136	0
boeing2	0.01	14.6	14.4	20.8	0.18433618	7.1433668	0
boeing2	0.1	14.6	13.2	20	0.18462308	1.5341246	5
bore3d	0.001	21.2	10	21	0.00000223061184	0.88693624	0
bore3d	0.01	21.2	11	21.6	0.11544673507682	3.41384604	0
bore3d	0.1	21.2	12	21	0.3547573390784	8.06042338	0.2
brandy	0.001	20.6	8.8	19	0.039623248	0.34686414	8.8
brandy	0.01	20.6	9	19	0.046975972	0.79027314	12.4
brandy	0.1	20.6	11.2	18.8	0.502859746	1.01120956	14.4
capri	0.001	20.4	18.4	26.2	0.00016831052	0.0072130768	0
capri	0.01	20.4	17.8	26	0.00016677984	0.007665238	1.6
capri	0.1	20.4	18	26.6	0.25046158722	0.0155090318	0.2
degen2	0.001	20.2	3	17	0.000005645633	0.0000796302	20.8
degen2	0.01	20.2	7	17.2	0.0000031817642	0.84839611872	0
degen2	0.1	20.2	11.2	17	0.1181135960782	1.1345114	4
e226	0.001	20.4	19	24	7.5850646	0.97988628	0.8
e226	0.01	20.4	15.8	23	1.86746298	0.34696046	4.8
e226	0.1	20.4	18.8	22.8	19.8835146	1.933322	12
grow15	0.001	18.4	16.2	25.4	0.1865341	0.000085384567	0
grow15	0.01	18.4	17.2	26.6	0.19973484	0.000854905	0
grow15	0.1	18.4	17.8	28.2	0.22845346	0.00855273388	0
grow7	0.001	20.6	12.2	23.6	0.186796516	0.000016800818	0
grow7	0.01	20.6	14.2	25.8	0.228411068	0.000163084214	0
grow7	0.1	20.6	14.8	25.2	0.246693378	0.0016308457	0
israel	0.001	14.2	8.4	30	0.020295981	0.000346152246	5.6
israel	0.01	14.2	9.8	30.2	0.0286291916	0.00343975728	7.8
israel	0.1	14.2	11.4	30	0.056422872	0.030134059	11.2
kb2	0.001	3.8	4	16.6	0.000000019004656	0.0007842734	2
kb2	0.01	3.8	5	16.6	0.000369616376414	0.00856494636	1.6
kb2	0.1	3.8	8.2	17.2	0.020672638639716	0.1821661314	0.8
lotfi	0.001	21.6	25.8	38	1.7805836	0.120510004	1.8
lotfi	0.01	21.6	24.2	35.8	1.40789576	2.76309842	45.2
lotfi	0.1	21.6	23.8	35.8	1.530596	170.6490454	96.6

TABLE 5. Numerical performance of LOQO on the Netlib test suite when warmstarting a problem whose objective function coefficients, c, have been modified. Each problem labeled under the column Problem, δ is the perturbation parameter, and #Pert is the number of perturbed coefficients. The average iteration counts for the warmstart and coldstart solution of five perturbed problems are given, and the last 3 columns are the scaled Euclidean norm of the distance between the optimal solutions and the Lagrange multipliers of the initial and the perturbed problem and the number of changes to the active set.

Problem	δ	#Pert	WarmIters	ColdIters	$\|x^I - x^P\|$	$\|y^I - y^P\|$	Δ
sc105	0.001	9.4	6	14	0.0000000090200132	0.096662278	0.4
sc105	0.01	9.4	6.6	14.2	0.009727429689914	1.41571990642	0.4
sc105	0.1	9.4	10.6	15.2	0.844300438	5.3586719	9.4
sc205	0.001	18.8	10.2	16.2	0.0100416708095964	7.5806026	3.2
sc205	0.01	18.8	14.4	21.2	3.339876804	192.510708	11.2
sc205	0.1	18.8	14.4	20.2	3.64762244	106.203382	20.4
sc50a	0.001	4.8	3.8	13	0.000000016398826	0.000088376558	1.4
sc50a	0.01	4.8	4	13	0.000000014525946	0.00020353714	0
sc50a	0.1	4.8	7.2	13.4	0.033163870830478	5.762721344	1
sc50b	0.001	4.8	3.8	12	0.000000015517022	0.000094448346	2.4
sc50b	0.01	4.8	4	12	0.000000013537162	0.00019306788	0
sc50b	0.1	4.8	6.8	12.6	0.119834681284491	4.60649102	3.6
scagr25	0.001	20	9.8	29	0.00000285870502	0.00012296934	18
scagr25	0.01	20	9.4	29.2	0.00826375588036	0.00033113884	54.2
scagr25	0.1	20	11.6	29.6	0.039512082	0.00328769164	17.4
scagr7	0.001	14.2	18	26	0.000019408696	0.000068523248	0.6
scagr7	0.01	14.2	18	26	0.00006011668	0.000685447222	0.4
scagr7	0.1	14.2	18.2	26.2	0.0621536287312	0.00720193758	0.2
scfxm1	0.001	20.2	13.2	25	0.017501582	2.5048588	11
scfxm1	0.01	20.2	13.2	25	0.020984438	2.483063	0
scfxm1	0.1	20.2	13	25.8	0.020431872	3.4034368	13
scorpion	0.001	21	2.8	16	0.0000113990926	0.001766280248	1.2
scorpion	0.01	21	4.2	16	0.000000702533632	0.051455088	0
scorpion	0.1	21	6.8	16	0.088872504976634	0.3754302498	4
scsd1	0.001	18.2	5.6	14.2	0.030131356	0.0032454821	0
scsd1	0.01	18.2	5.6	14.2	0.031630776	0.00369138608	0
scsd1	0.1	18.2	6.2	14.2	0.030782104	0.01019002166	0
sctap1	0.001	20.2	12.4	24.4	1.1860689	0.0039026358	7.2
sctap1	0.01	20.2	19.2	23	1.18318768	0.0039616488	2.2
sctap1	0.1	20.2	18	22.2	1.18225046	0.0104209616	1
share1b	0.001	18.4	18	42	0.000361202950494	0.000259397786	0.8
share1b	0.01	18.4	17.6	41.4	0.001467506287294	0.00691697746	3.4
share1b	0.1	18.4	17.4	40.8	0.002776452505214	0.0301112166	0.8
share2b	0.001	8.2	7.4	16.6	0.086333116	0.00016688116	1.4
share2b	0.01	8.2	6.8	16.2	0.113452678	0.00095789916	2
share2b	0.1	8.2	8.8	15.8	0.159204882	0.0103449676	1
stair	0.001	20.4	20.4	19	3.3805094	0.0056217272	1
stair	0.01	20.4	20	19.6	3.8394062	0.0572294056	1.2
stair	0.1	20.4	20.6	21	4.4421358	0.41332886	28.6
stocfor1	0.001	10.6	3	17	0.0000000021162428	0.000097536572	0
stocfor1	0.01	10.6	3	17	0.0000000020014738	0.00045571615	0.6
stocfor1	0.1	10.6	3.4	17	0.0000000137254068	0.004760905648	3.8
tuff	0.001	19	30.6	27.8	0.416179104	39.6999	36.2
tuff	0.01	19	27.6	28.8	0.56144268	42.527566	50
tuff	0.1	19	28.2	31.6	1.5340454	45.267418	84.8
vtp.base	0.001	18.8	19	33.8	0.000000000068848036	11.968334	0
vtp.base	0.01	18.8	19	34.2	0.000000000068848036	11.968334	0
vtp.base	0.1	18.8	19	34.6	0.000000000068848036	11.968334	0

TABLE 6. Numerical performance of LOQO on the Netlib test suite when warmstarting a problem whose objective function coefficients, c, have been modified. Each problem labeled under the column Problem, δ is the perturbation parameter, and #Pert is the number of perturbed coefficients. The average iteration counts for the warmstart and coldstart solution of five perturbed problems are given, and the last 3 columns are the scaled Euclidean norm of the distance between the optimal solutions and the Lagrange multipliers of the initial and the perturbed problem and the number of changes to the active set.

Problem	δ	#Pert	WarmIters	ColdIters	$\|x^I - x^P\|$	$\|y^I - y^P\|$	Δ
adlittle	0.001	20	5.8	18.8	0.23004568	0.0307304172	4
adlittle	0.01	20	7.4	18.8	0.2423504	0.079378044	12.2
adlittle	0.1	20	10.4	17.4	0.26085276	0.2326881	2.2
afiro	0.001	8.6	2.2	12	0.000081411596	0.0012162842	1.4
afiro	0.01	8.6	3.4	12	0.00093170246	0.0098871706	0.2
afiro	0.1	8.6	5.2	11.8	0.0138997684	0.119891346	0.4
agg	0.001	17.8	12	23.2	0.0016293708	1.7770602	0
agg	0.01	17.8	12.4	23.2	0.0023571514	4.1736046	0
agg	0.1	17.8	12.4	23	0.0058083328	3.9331038	0.6
agg2	0.001	18.6	5.6	25.2	0.015322110548	0.00483277902	0
agg2	0.01	18.6	7.2	25.2	0.017689878706	0.06946087488	28.8
agg2	0.1	18.6	8	25.2	0.019425219474	0.34482405216	1.4
agg3	0.001	18.4	6	24	0.020046419958	1.23183767408	0
agg3	0.01	18.4	6.2	24	0.020141893888	0.5744438939	0
agg3	0.1	18.4	6.6	24	0.02136065331	2.50835156138	0
bandm	0.001	18	10	19	0.0001677733578	0.97037126	2.8
bandm	0.01	18	10.6	19.2	0.002256803846	1.01341196	0.2
bandm	0.1	18	12	19	0.02387359544	1.36662542	5.2
beaconfd	0.001	18.2	3	14	0.001084700882	0.00077815982	8.2
beaconfd	0.01	18.2	3.6	14	0.03035784145	0.02878392712	8.4
beaconfd	0.1	18.2	6.6	14.2	0.20251008	1.83049378	4.2
blend	0.001	19.6	6	16	0.0236584826	0.00125026344	0
blend	0.01	19.6	8.2	16.4	0.028081222	0.0036489594	0.2
blend	0.1	19.6	11	15.6	0.068060578	0.06687189	3.2
boeing1	0.001	17.4	16	27	0.0058299596	2.1942368	0
boeing1	0.01	17.4	16	27	0.0057763142	2.0608648	18.4
boeing1	0.1	17.4	16.4	26.8	0.0085252768	2.2106048	24.6
boeing2	0.001	19.8	14.8	20.8	0.162697554	5.1748438	0.4
boeing2	0.01	19.8	13.8	20.4	0.170450074	1.54258968	10.2
boeing2	0.1	19.8	14.2	21.2	0.171995624	1.23068666	3.4
bore3d	0.001	19.4	28	21	0.0785505912382	213.37428428	0
bore3d	0.01	19.4	28	21	0.07719931111	3113.18141414	0
bore3d	0.1	19.4	28.2	22.2	0.09650122318	27539.2610609	8
brandy	0.001	20.8	9.4	19	0.099916238	0.40077668	9.8
brandy	0.01	20.8	9.2	19.2	0.109523462	0.3771333	10.6
brandy	0.1	20.8	13.8	18.8	1.632442678	3.41448172	14
capri	0.001	18.4	17.4	25.4	0.000421063788	0.0138256384	2
capri	0.01	18.4	17.8	25.2	0.00401641782	0.0147313122	2.6
capri	0.1	18.4	18.4	25.8	0.0441783178	0.0349884738	10.8
e226	0.001	18.2	18.8	24.2	8.11769	1.55227514	1.4
e226	0.01	18.2	19.2	24	10.1792914	1.2938678	0.6
e226	0.1	18.2	15.8	23	3.7509858	0.43574222	4.8
grow15	0.001	19.2	16.2	25.4	0.17699266	0.0001555718526	0
grow15	0.01	19.2	16.8	26.2	0.20803348	0.064972741064	0
grow15	0.1	19.2	17.2	26.2	0.2168174	0.06225957088	0
grow7	0.001	19.2	11.6	24.4	0.17873502	0.0000501711682	0
grow7	0.01	19.2	13.8	26.4	0.26178442	0.000501958564	0
grow7	0.1	19.2	15.8	26.4	0.27726752	0.03709622704	0
israel	0.001	18.6	7.8	29.4	0.0024163726	0.0000163038748	4.2
israel	0.01	18.6	8	29.2	0.0039731858	0.000163013298	0.2
israel	0.1	18.6	11	28.4	0.0394124382	0.00234963732	5.6
kb2	0.001	20.6	9.8	17.2	0.00210601376	0.00577251792	0
kb2	0.01	20.6	10.2	17.2	0.0053032874	0.014157776	0.8
kb2	0.1	20.6	16.8	16.6	0.265510688	1880.918933588	2.4
lotfi	0.001	19.4	30.6	39.8	1.6736976	0.037889778	0.8
lotfi	0.01	19.4	25.6	38	1.602055	0.038145572	0.8
lotfi	0.1	19.4	25.6	42.2	1.39840218	0.060610612	7.6
recipe	0.001	18.4	11.2	13	4.2583008	9.4780032	0
recipe	0.01	18.4	11.2	13	4.8460016	8.323369	0
recipe	0.1	18.4	11.4	13.2	5.6405112	10.6306476	0

TABLE 7. Numerical performance of LOQO on the Netlib test suite when warmstarting a problem whose constraint coefficients, A, have been modified. Each problem labeled under the column Problem, δ is the perturbation parameter, and #Pert is the number of perturbed coefficients. The average iteration counts for the warmstart and coldstart solution of five perturbed problems are given, and the last 3 columns are the scaled Euclidean norm of the distance between the optimal solutions and the Lagrange multipliers of the initial and the perturbed problem and the number of changes to the active set.

Problem	δ	#Pert	WarmIters	ColdIters	$\|x^I - x^P\|$	$\|y^I - y^P\|$	Δ
sc105	0.001	20	6.4	14	0.00069372118	0.153107678	0
sc105	0.01	20	7.4	14	0.0069127822	1.1976098	0
sc105	0.1	20	10.8	14.2	0.06527052	6.4907776	0.4
sc205	0.001	20	9.4	16	0.00027682266	2.02464148	0
sc205	0.01	20	9	16	0.002766188	0.423780368	0
sc205	0.1	20	10.8	16.6	0.02770365	1.7525939	1.2
sc50a	0.001	13	3.4	13	0.00037282702	0.000101846704	4.6
sc50a	0.01	13	4.4	13	0.0037243434	0.00089431732	4.8
sc50a	0.1	13	9.2	13.4	0.036835694	4.1847840722	1.6
sc50b	0.001	11.6	3.4	12	0.00046263898	0.000093891172	8.4
sc50b	0.01	11.6	4.2	12	0.0046249472	0.00037081124	0.8
sc50b	0.1	11.6	6	12	0.04618263	0.707129808	0
scagr25	0.001	18.2	9.6	29	0.00020719086	0.00021514086	18
scagr25	0.01	18.2	11.2	28.8	0.0149594926	0.00358720982	3.2
scagr25	0.1	18.2	15.4	30	0.045162464	0.0560662974	36.4
scagr7	0.001	20.4	18.4	26	0.00134188584	0.000114529798	0.8
scagr7	0.01	20.4	18	26	0.0049032864	0.00118178268	0
scagr7	0.1	20.4	18.2	25.4	0.201980066	0.0126154122	0.2
scfxm1	0.001	18.6	13	25	0.01682499	2.6291784	3.2
scfxm1	0.01	18.6	13	25	0.015106814	2.118531	0
scfxm1	0.1	18.6	12.8	25.4	0.01846191	2.4163766	3.8
scsd1	0.001	18.4	6.2	14.2	0.08413639	0.00715237406	0
scsd1	0.01	18.4	6.4	14.2	0.077907088	0.00689536048	0
scsd1	0.1	18.4	7	14.2	0.088189282	0.02312750224	0
sctap1	0.001	18.8	10.6	20.4	0.26710713912	0.00349713106	2.4
sctap1	0.01	18.8	12.8	19.8	0.273224044	0.0043261482	3.2
sctap1	0.1	18.8	14.2	19	0.273168796	0.050630594	0.4
share1b	0.001	19.4	18	41	0.0014607127582	0.00149440534	0
share1b	0.01	19.4	18.2	39.8	0.012810718898	0.0359196054	8
share1b	0.1	19.4	20.2	42.2	0.0988513284	0.4266846438	3
share2b	0.001	19	6	15.6	0.0374120312	0.001074656708	2.8
share2b	0.01	19	7.2	15.6	0.0669569982	0.01214286038	0.4
share2b	0.1	19	13.4	15.6	0.27288212	0.1687331604	4.6
stair	0.001	18.2	21	19	3.0647778	0.0012958324	0
stair	0.01	18.2	20.2	19	3.0503776	0.0018312654	4
stair	0.1	18.2	20.4	19.2	3.4561404	0.0242050408	0.4
stocfor1	0.001	20.2	4.6	17.2	0.000160629168	0.01661990654	0.8
stocfor1	0.01	20.2	4.8	17.4	0.00160079038	0.0216713722	1.8
stocfor1	0.1	20.2	8.2	17.8	0.0155442492	0.03954027	3
tuff	0.001	19	29.2	27	0.0049594306	28.749366	0
tuff	0.01	19	29.8	27.2	0.00521817	29.915192	0
tuff	0.1	19	26.8	24.6	0.0135730138	28.544668	7.6
vtp.base	0.001	19	19	34.2	0.000075436772	11.478906	0.2
vtp.base	0.01	19	19	35	0.00075576422	10.774968	0.2
vtp.base	0.1	19	18.2	34.8	0.00765695436	11.2667912	0.6

TABLE 8. Numerical performance of LOQO on the Netlib test suite when warmstarting a problem whose constraint coefficients, A, have been modified. Each problem labeled under the column Problem, δ is the perturbation parameter, and #Pert is the number of perturbed coefficients. The average iteration counts for the warmstart and coldstart solution of five perturbed problems are given, and the last 3 columns are the scaled Euclidean norm of the distance between the optimal solutions and the Lagrange multipliers of the initial and the perturbed problem and the number of changes to the active set.

Problem	WarmIters	ColdIters	Δ
Diet-2	6	11	1
Diet-3	6	10	1
Diet-4	6	10	1
Diet-5	6	10	1
Diet-6	6	11	1
Diet-7	6	11	1
Diet-8	6	10	2
Diet-9	6	11	1
Diet-10	7	11	1
Diet-11	6	10	1
HL415-2	7	12	1
HL415-3	8	11	2
HL415-4	(inf)	(inf)	(inf)
HL415-5	9	10	1
HL415-6	7	11	1
HL415-7	7	11	1
Synthes3-2	11	15	5
Synthes3-3	9	14	4
Synthes3-4	11	15	1
Synthes3-5	9	14	2
Synthes3-6	9	13	3
Synthes3-7	10	13	1
Synthes3-8	10	15	2
Synthes3-9	9	13	3
Synthes3-10	11	14	1
Synthes3-11	10	16	4
Synthes3-12	9	15	4
Synthes3-13	10	15	1
Synthes3-14	9	14	2
Synthes3-15	10	15	2

TABLE 9. Numerical performance of LOQO on the LPs at each node of a branch and bound tree arising in the solution of three problems. Each problem is warmstarted from the optimal solution of its parent problem. The iteration counts for the warmstart and coldstart solution of each problem are given, as well as the number of changes to the active set from the parent node's problem. (inf) indicates that the subproblem was found to be infeasible.

Problem	WarmIters	ColdIters	Δ
Master-2	6	11	5
Master-3	6	11	5
Master-4	6	11	5

TABLE 10. Numerical performance of LOQO on the master problems of the cutting stock model. Each problem has one additional variable which is initialized to 0. All other primal and dual variables and slacks are initialized from the optimal solution of the previous master problem. The iteration counts for the warmstart and coldstart solution of each master problem are given, as well as the number of changes to the active set from the previous master problem.

HANDE Y. BENSON, DREXEL UNIVERSITY, PHILADELPHIA, PA

DAVID F. SHANNO, RUTCOR - RUTGERS UNIVERSITY, NEW BRUNSWICK, NJ