INTERIOR-POINT METHODS FOR NONCONVEX NONLINEAR PROGRAMMING: REGULARIZATION AND WARMSTARTS

HANDE Y. BENSON AND DAVID F. SHANNO

ABSTRACT. In this paper, we investigate the use of an exact primal-dual penalty approach within the framework of an interior-point method for nonconvex nonlinear programming. This approach provides regularization and relaxation, which can aid in solving ill-behaved problems and in warmstarting the algorithm. We present details of our implementation within the LOQO algorithm and provide extensive numerical results on the CUTEr test set and on warmstarting in the context of quadratic, nonlinear, mixed integer nonlinear, and goal programming.

1. INTRODUCTION

A nonlinear programming problem (NLP), in its most general form, can be expressed as

$$\begin{array}{ll} \min\limits_{x} & f(x) \\ \text{s.t.} & 0 \leq h(x) \leq r \\ & l \leq x \leq u, \end{array} \tag{1}$$

where $x \in \mathbb{R}^n$ are the decision variables, $f : \mathbb{R}^n \to \mathbb{R}$ is the objective function, $h : \mathbb{R}^n \to \mathbb{R}^m$ are the constraint functions, $r \in \mathbb{R}^m$ is a nonnegative (possibly infinite) parameter, and $l \in \mathbb{R}^n$ and $u \in \mathbb{R}^n$ are lower and upper bounds on the decision variables, one or both of which can be infinite.

The algorithm we will use to solve such problems is the infeasible primal-dual interior-point method implemented in LOQO [23]. As such, we will assume that all objective and constraint functions are twice continuously differentiable. If f is convex, h_i are concave for $r_i = \infty$ and linear otherwise, then the problem is convex and the algorithm, if it terminates successfully, will locate a global optimum. If the problem is not convex, the algorithm will be searching for a local optimum.

Interior-point codes, such as LOQO [23], IPOPT [25] and KNITRO [19], have been shown in numerical studies such as [5] and [18] to be more efficient than other codes using sequential quadratic programming (SQP) or reduced gradient approaches, especially on large and sparse problems. Despite this computational advantage, however, ill-behaved problems and problems without Karush-Kuhn-Tucker (KKT) solutions have posed serious difficulties for claiming reliability of these codes. For example, in [24], Wächter and Biegler showed that a class of interior-point methods become stuck at a nonoptimal, and in fact infeasible, point on a particular problem.

Date: June 13, 2006.

Key words and phrases. interior-point methods, nonlinear programming, warmstarting, penalty methods.

Research of the first author is sponsored by ONR grant N00014-04-1-0145. Research of the second author is supported by NSF grant DMS-0107450.

In [7], this phenomenon was called "jamming." A large set of problems known as mathematical programs with equilibrium constraints (MPECs) have been shown in [1] and others to have an unbounded set of optimal Lagrange multipliers, and in [2] it is shown that a primal-dual interior-point code such as LOQO will have convergence problems while looking for the analytic center of the face of optimal dual solutions. Additionally, proper stopping procedures, similar to the elastic mode of the SQP code SNOPT [11], need to be investigated for detecting lack of KKT solutions, whether it be from problems that are primal and/or dual infeasible or where the duality gap at the optimal solution is nonzero.

There have been several recent studies into using a penalty method approach for resolving some of these issues. As discussed in [2], an ℓ_1 penalty approach can be implemented within the framework of an interior-point code to resolve the issues of jamming, infeasibility identification, and the location of nonKKT solutions. This recent interest in penalty methods arose in the context of solving MPECs, and [1], [2], and [16] have proposed the use of these methods to resolve the issue of unbounded Lagrange multipliers. Numerical results presented in [2] and [16] show improved numerical performance on MPECs. Additionally, in [12], an interior-point method using an ℓ_1 penalty approach was proposed in the context of general NLP.

In the recent paper [4], the authors have shown that an exact primal-dual penalty approach can also be useful for warmstarting an interior-point method for linear programming. *Warmstarting* is the use of information obtained during the solution of an initial problem to solve subsequent, closely-related problems. When no such information is used, the new problem is solved from a *coldstart*. As demonstrated in [4], an interior-point method can become stuck at the initial problem's solution, thereby requiring more iterations than a coldstart to solve a problem. The relaxation aspect of a penalty approach is ideally suited to making iterates "unstuck," and the authors in [4] proposed a new primal-dual penalty framework so that the relaxation/bounding scheme of a penalty method could be applied on both the primal and the dual sides to accommodate various types of perturbations to the problem data.

The main goal of this paper is to extend the exact primal-dual penalty approach of [4] to NLP in order to show that it will be useful for warmstarting NLPs as well. In addition, we show that the primal-dual penalty approach retains the theoretical advantages of the purely primal penalty method for solving ill-behaved problems. We have implemented the primal-dual penalty in LOQO, and we will present extensive numerical results for warmstarting in the context of quadratic, nonlinear, and mixed-integer nonlinear programming. We also demonstrate the effectiveness of the primal-dual penalty approach for improving the solution of general NLPs from the CUTEr [13] test set. As shown in [7], LOQO is a very robust code, and, therefore, we will use the algorithm without modification as much as possible and employ the penalty approach only as necessary, such as in the case of warmstarting. Therefore, we start by presenting the interior-point algorithm currently implemented in LOQO.

2. LOQO: An Interior-Point Code for Nonlinear Programming

We begin with an overview of the LOQO algorithm. A more detailed explanation can be found in [23].

We concentrate here on the inequality constrained problem

$$\begin{array}{lll} \min\limits_{x} & f(x) & \\ \text{s.t.} & h(x) & \geq \quad 0. \end{array} \tag{2}$$

This simplification is made for purely pedagogical reasons, and a description of how to handle more general problems will follow.

First, slack variables, w, are added to each of the constraints to convert them to equalities, and the nonnegativity constraints on the slack variables are eliminated by placing them in a barrier objective function, giving the Fiacco and McCormick [9] logarithmic barrier problem:

$$\begin{array}{lll} \min\limits_{x,w} & f(x) - \mu \sum\limits_{i=1}^{m} \log w_i & \\ \text{s.t.} & h(x) - w & = \quad 0, \end{array}$$

where $w \in \mathbb{R}^m$. The scalar $\mu > 0$ is called the *barrier parameter*. Now that we have an optimization problem with no inequalities, we form the Lagrangian

$$L_\mu(x, w, y) = f(x) - \mu \sum_{i=1}^{m} \log w_i - y^T(h(x) - w), \tag{3}$$

where $y \in \mathbb{R}^m$ are called the *Lagrange multipliers* or the *dual variables.*

In order to achieve a stationary point of the Lagrangian function, we solve the primal-dual system arising from the first-order optimality conditions:

$$\begin{array}{rcl} \nabla f(x) - A(x)^T y & = & 0 \\ -\mu e + WYe & = & 0 \\ h(x) - w & = & 0, \end{array} \tag{4}$$

where $A(x)$ is the transpose of the Jacobian of the constraints, W and Y are the diagonal matrices with the entries of w and y, respectively, and e is the vector of ones of appropriate size.

In order to solve this system, we use Newton's Method. Doing so gives the following system to solve:

$$\begin{bmatrix} -H(x,y) & A(x)^T \\ A(x) & E \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} \sigma \\ \rho + E\gamma \end{bmatrix} \tag{5}$$

$$\Delta w = WY^{-1}(\gamma - \Delta y), \tag{6}$$

where

$$\begin{array}{lcl} H(x,y) & = & \nabla^2 f(x) - \sum_{i=1}^{m} y_i \nabla^2 h(x) \\ E & = & WY^{-1} \\ \sigma & = & \nabla f(x) - A(x)^T y \\ \gamma & = & \mu W^{-1} e - y \\ \rho & = & -h(x) + w. \end{array}$$

Note that the algorithm is a primal-dual algorithm, which means that solving (5) yields step directions for both the primal and the dual variables. This system is solved using LDL^T *factorization*, which is a modified version of Cholesky factorization, and then performing a backsolve to obtain the step directions. If the entries of the diagonal matrix D have the wrong sign, the problem is nonconvex, and $H(x, y)$ is replaced in the current iteration by $H(x, y) + \lambda I$, where I is the $n \times n$ identity matrix and λ is a sufficiently large perturbation parameter. For details, refer to [20].

The algorithm starts at an initial solution $(x^{(0)}, w^{(0)}, y^{(0)})$ such that $w^{(0)} > 0$ and $y^{(0)} > 0$, and proceeds iteratively toward the solution through a sequence of points which are determined by the search directions obtained from the reduced KKT system (5) as follows:

$$\begin{array}{rcl} x^{(k+1)} & = & x^k + \alpha^{(k)}\Delta x^{(k)}, \\ w^{(k+1)} & = & w^k + \alpha^{(k)}\Delta w^{(k)}, \\ y^{(k+1)} & = & y^k + \alpha^{(k)}\Delta y^{(k)}, \end{array}$$

where $0 < \alpha \leq 1$ is the *steplength* and the superscripts denote the iteration number. $\alpha^{(k)}$ is chosen to ensure that $w^{(k+1)} > 0$, $y^{(k+1)} > 0$, and either the barrier function or the primal infeasibility is reduced subject to an Armijo condition. This approach to steplength control is called a "Markov filter." For full details, the reader is referred to [5].

LOQO concludes that it has reached an optimal solution when the first-order conditions (4) with $\mu = 0$ are satisfied within a tolerance level, which is set to 10^{-7} by default.

2.1. **Problems in General Form.** Even though we described the algorithm for the inequality constrained problem (2), LOQO can handle problems involving variable bounds, as well as range and equality constraints. A variable with the bounds

$$l \leq x \leq u,$$

is converted to

$$\begin{array}{rcl} x - g & = & l \\ x + t & = & u \\ g, t & \geq & 0. \end{array} \tag{7}$$

A range constraint of the form

$$0 \leq h(x) \leq r$$

is converted to

$$\begin{array}{rcl} h(x) - w & = & 0 \\ w + p & = & r \\ w, p & \geq & 0. \end{array} \tag{8}$$

An equality constraint is treated as a range constraint as well, with $r = 0$. Therefore, each equality constraint has two slack variables, w, and p, associated with it. This has important implications for the Cholesky factorization, as LOQO performs a *dual ordering*, where the initial pivots are picked from the lower right block of the matrix in (5). According to the computational experience reported in [20], a dual ordering can serve to decrease computation time per iteration and to provide a more numerically stable ordering. Including slack variables for equality constraints (rather than not slacking them at all) ensures that there will always be strictly positive entries on the diagonal of the lower right block, regardless of the type of constraint, so that a dual ordering can always be performed.

2.2. **The Dual Problem.** Although dual feasibility is included among the first-order conditions (4), the dual objective function, and thereby the form of the dual problem for an NLP is not generally of concern to solvers (unlike in linear programming). Having its roots in linear and quadratic programming, LOQO works with the dual problem of (2)

$$\begin{array}{lll} \max\limits_{y} & f(x) - \nabla f(x)^T x - (h(x) - A(x)x)^T y & \\ \text{s.t.} & \nabla f(x) - A(x)^T y & = \ 0 \\ & y & \geq \ 0 \end{array} \tag{9}$$

in the context of NLP. The dual objective function is the first-order Taylor series expansion of the Lagrangian (3). In fact, it can be seen that given a primal and dual feasible solution, the requirement that the duality gap be zero reduces to the complementarity condition:

$$[f(x)] - \left[f(x) - \nabla f(x)^T x - (h(x) - A(x)x)^T y\right] = 0 \tag{10a}$$

$$h(x)^T y + (\nabla f(x) - A(x)^T y)^T x = 0 \tag{10b}$$

$$w^T y = 0 \tag{10c}$$

$$WYe = 0, \tag{10d}$$

where equation (10a) is the requirement that the duality gap be zero, which simplifies to (10b), and equation (10c) is obtained by the feasibility of the primal and dual solutions and simplifies to (10d) by the nonnegativity of w and y. In the past, the duality gap was a measure of optimality in LOQO for NLPs, as well, but in the current implementation, it has been replaced by the complementarity condition and matches (4) with $\mu = 0$. We have chosen to include the dual problem here in order to illustrate the symmetry of the primal-dual penalty approach in the next section.

3. An Exact Primal-Dual Penalty Approach to Regularization and Relaxation

In this paper, we will use the primal-dual penalty approach introduced in [4] for warmstarting interior-point methods for linear programming. The classical penalty method as described in [10] is purely primal, and we will need the dual aspect mainly for detecting dual infeasibility and for warmstarting. Doing so is necessary for warmstarting because perturbations to the problem which change the objective gradient can lead to the current solution becoming dual infeasible. In order to restore dual feasibility, it becomes necessary to relax the dual constraints.

The primal penalty problem corresponding to (2) has the form

$$\begin{array}{ll} \min\limits_{x,w,\xi} & f(x) + c^T \xi \\ \text{s.t.} & h(x) - w = 0 \\ & -\xi \leq w \leq b \\ & \xi \geq 0, \end{array} \tag{11}$$

where $\xi \in \mathbb{R}^m$ are the new decision variables corresponding to the relaxation of the constraints and $c \in \mathbb{R}^m$ are their penalty parameters. This new form of the primal penalty problem differs from the classical approach presented in [10] in two crucial aspects: (1) The slacks, w, rather than the constraint functions are relaxed, and (2) upper bounds are also added to the slacks. The new upper bounds, b, on the slack

variables are the dual penalty parameters, and the dual penalty problem, then, has the form

$$\begin{array}{ll} \max\limits_{y,\psi} & f(x) - \nabla f(x)^T x - (h(x) - A(x)x)^T y - b^T \psi \\ \text{s.t.} & \nabla f(x) - A(x)^T y = 0 \\ & -\psi \leq y \leq c - \psi \\ & \psi \geq 0, \end{array} \tag{12}$$

where $\psi \in \mathbb{R}^m$ are the dual relaxation variables. We have that $c > 0$ and $b > 0$. As the first-order conditions of the Lagrangian function will show in the discussion to follow, the relaxation of the primal slack variables and the penalty term in the primal objective function result in the bounding of the Lagrange multipliers, as in the purely primal approach. The bounding of the primal slack variables has the reverse effect on the dual and results in the relaxation of the Lagrange multipliers from below by ψ and introduces a dual penalty term. The upper bound on the Lagrange multipliers, $c - \psi$ differs from the purely primal approach, which would have c as the upper bound, but as ψ is driven to 0, the upper bound can be kept sufficiently large. The primal-dual penalty approach (11)-(12) is *exact*, that is, given an optimal primal-dual pair (x^*, y^*) to (2) and (9), the pair (11)-(12) also has a solution at (x^*, y^*) for sufficiently large but finite b and c. The proof of exactness is identical to the one provided in [4] for the case of linear programming. Note again that we have chosen to allow the slack variables and the Lagrange multipliers to go negative rather than directly relax the constraints, as discussed in [10], [12], and [2]. This will allow for better warmstarting by changing the terms in the barrier objective of our infeasible interior-point method.

Solving (11) instead of (2) has several benefits, including the relaxation of the problem to achieve feasibility and the bounding of the primal and dual variables. Many of these benefits were illustrated and/or proved in [2] for the purely primal penalty, and we will provide a quick overview here. However, we will first discuss the application of the algorithm of Section 2 to (11)-(12).

3.1. **The Penalty Approach within the IPM Framework.** We follow the development of Section 2 to present the algorithm to solve (11). The logarithmic barrier problem associated with (11)-(12) is

$$\begin{array}{ll} \min\limits_{x,w,\xi} & f(x) + c^T \xi - \mu \sum\limits_{i=1}^{m} \log(\xi_i) - \mu \sum\limits_{i=1}^{m} \log(w_i + \xi_i) - \mu \sum\limits_{i=1}^{m} \log(b_i - w_i) \\ \text{s.t.} & h(x) - w = 0, \end{array} \tag{13}$$

where $w \in \mathbb{R}^m$ are the slack variables and μ is the (nonnegative) barrier parameter. The first-order conditions for the barrier problem are derived from the Lagrangian of (13) and can be stated as follows:

$$\nabla f(x) - A(x)^T y = 0 \tag{14}$$

$$h(x) - w = 0 \tag{15}$$

$$\mu e - (W + \Xi)(Y + \Psi)e = 0 \tag{16}$$

$$\mu e - \Xi(C - Y - \Psi)e = 0 \tag{17}$$

$$\mu e - \Psi(B - W)e = 0 \tag{18}$$

where C, B, Ξ and Ψ are the diagonal matrices with the entries of c, b, ξ, and ψ, respectively. (14) and (15) correspond to the dual and primal feasibility conditions of

the original problem. The new complementarity conditions, (16)-(18), perturb the original complementarity condition and bound the primal slacks and the Lagrange multipliers above by the primal and the dual penalty parameters.

Applying Newton's Method to (14)-(18) and eliminating Δw, $\Delta\xi$, and $\Delta\psi$, the the reduced KKT system for the primal-dual penalty problem has the same form as (5) with

$$
\begin{array}{rcl}
E & = & \Big(\big((Y+\Psi)^{-1}(W+\Xi)+\Xi(C-Y-\Psi)^{-1}\big)^{-1}+\Psi(B-W)^{-1}\Big)^{-1} \\
\gamma & = & \big((Y+\Psi)^{-1}(W+\Xi)+\Xi(C-Y-\Psi)^{-1}\big)^{-1} \\
 & & \big(\mu(Y+\Psi)^{-1}e-\mu(C-Y-\Psi)^{-1}e-w\big)-\big(\mu(B-W)^{-1}e-\psi\big)
\end{array}
\tag{19}
$$

and

$$
\begin{array}{rcl}
\Delta w & = & -(W+\Xi)(Y+\Psi)^{-1}(\mu\Psi^{-1}e+y+\psi+\Delta y+\Delta\psi)-\Delta\xi, \\
\Delta\xi & = & (C-Y-\Psi)^{-1}\Xi(\Delta y+\Delta\psi)+\mu(C-Y-\Psi)^{-1}e-\xi, \\
\Delta\psi & = & -\big((Y+\Psi)^{-1}(W+\Xi)+(B-W)\Psi^{-1}+(C-Y-\Psi)^{-1}\Xi\big)^{-1} \\
 & & \big(b-\mu\Psi^{-1}e-\mu(Y+\Psi)^{-1}e+\mu(C-Y-\Psi)^{-1}e\big) \\
 & & -\big((Y+\Psi)^{-1}(W+\Xi)+(B-W)\Psi^{-1}+(C-Y-\Psi)^{-1}\Xi\big)^{-1} \\
 & & \big((Y+\Psi)^{-1}(W+\Xi)+(C-Y-\Psi)^{-1}\Xi\big)\,\Delta y.
\end{array}
$$

Thus, we again have a primal-dual algorithm that solves (11) and (12) simultaneously, yielding step directions for the primal and dual variables (both original and relaxation) at each iteration.

There are several things to note about this system. First, it has the same sparsity structure as (5), the reduced KKT system of the original problem. Therefore, there would be no significant computational effort to switch back and forth between the solutions of these two related systems. Second, by changing the terms on the lower right diagonal of the matrix, the penalty method is said to *regularize* the system. This is achieved using the scheme of relaxing/penalizing via a variable and as the variable goes to 0, so does the regularization. Third, the original term in the lower right diagonal, WY^{-1}, has been changed to E as defined in (19). Doing so will allow for positive entries if the slack variables prematurely go to 0 and aid in warmstarting by eliminating 0 diagonal entries. This is essential for our implementation in the solver LOQO as it uses a dual ordering in its Cholesky factorization routine.

The steplength, $\alpha^{(k)}$, at each iteration k is chosen to ensure that

$$
\begin{array}{lcl}
w^{(k+1)}+\xi^{(k+1)} & > & 0, \\
y^{(k+1)}+\psi^{(k+1)} & > & 0, \\
\xi^{(k+1)} & > & 0, \\
\psi^{(k+1)} & > & 0, \\
b-w^{(k+1)} & > & 0, \\
c-y^{(k+1)}-\psi^{(k+1)} & > & 0,
\end{array}
$$

and either the barrier function or the infeasibility is reduced. The algorithm concludes that it has reached an optimum when the optimality conditions for the original problem are satisfied.

3.2. **Problems in General Form.** The penalty problem, (11), is obtained from (2) by relaxing the nonnegativities of the slack variables. This approach will be used for problems in general form. Slack variables g and t on the variable bounds as well as w and p on the range and equality constraints are all allowed to take on

negative values, at a penalty cost. It should be noted that it is essential to rewrite equality constraints as range constraints in order to relax them.

3.3. **Setting and Updating the Penalty Parameters.** The most important aspect of setting the initial values of the penalty parameters is to ensure that they are sufficiently larger than those components of the current iterate for which they serve as upper bounds. Therefore, the penalty parameters are set as follows:

$$\begin{aligned} b &= 10(w + \kappa) \\ c &= 10(y + \psi + \kappa), \end{aligned}$$

where κ is a constant with a default value of 1. The relaxation variables are initialized as

$$\begin{aligned} \xi &= \max(h(x) - w, 0) + \tau \\ \psi &= \tau, \end{aligned} \tag{20}$$

where τ is a small parameter, currently set to $10^{-5}M$, where M is the greater of 1 and the largest primal slack or dual variable value. These initializations are generally sufficient for switching to the penalty method when we are far from the optimum, and close to the solution, especially after a warmstart, they allow starting the penalty method without moving the iterates too far from the current point. Note that the relaxation is performed using a variable, so if a larger relaxation is needed, the variables, ξ and ψ, will move as necessary. As we point out in the context of the numerical results in Section 5, when warmstarting nonconvex problems, it may be better to take a smaller value of τ, such as 10^{-14}, precisely because we need to ensure that the iterates leave the neighborhood of the current solution only when needed.

Since the initial values of the penalty parameters, b and c, may not be large enough to admit the optimal solution, we also need an updating scheme for these parameters. Given the relaxations, a pair of optimal solutions can always be found for (11)-(12), and one possible "static" updating scheme is to solve a problem to optimality and to increase the penalty parameters if their corresponding relaxation variables are not sufficiently close to zero. However, this may require multiple solves of a problem and substantially increase the number of iterations necessary to find the optimal solution. We have instead implemented a "dynamic" updating scheme in LOQO, where the penalty parameters are checked at the end of each iteration and updated as follows:

$$\begin{aligned} &\text{If} \quad w_i^{(k)} > 0.9 b_i^{(k)}, \quad &&\text{then} \quad b_i^{(k+1)} = 10 b_i^{(k)}, \quad &&i = 1, \ldots, m. \\ &\text{If} \quad y_i^{(k)} + \psi_i^{(k)} > 0.9 c_i^{(k)}, \quad &&\text{then} \quad c_i^{(k+1)} = 10 c_i^{(k)}, \quad &&i = 1, \ldots, m. \end{aligned}$$

3.4. **Benefits of a Penalty Approach.** Many of the benefits of using an ℓ_1 penalty approach have been discussed in literature. Although we are relaxing the nonnegativities on the slack variables, rather than the constraints themselves directly, and we are also relaxing the dual variables, the same theoretical properties follow. We will now summarize them here. The main contribution of this paper, which is the use of the primal-dual penalty approach to warmstart the interior-point method, will be discussed in the next section.

3.4.1. *Regularization and Jamming.* As shown in [24], an interior-point method can become stuck at a nonoptimal point as the slack variables prematurely go to 0. This phenomenon is called *jamming* in [7], where the authors also show that it can be avoided if the Jacobian of the active constraints is nonsingular. The nonsingularity of the Jacobian ensures that the reduced KKT matrix also remains nonsingular, even as the entries in the lower diagonal approach zero.

An ℓ_1 penalty approach can have a similar effect by ensuring that since a different relaxation variable is used for each constraint, the Jacobian is always nonsingular. In our formulation of the primal-dual penalty method, primal relaxation also improves the conditioning of the reduced KKT matrix, and examination of (19) shows that a dual relaxation further improves this conditioning. Additionally, we update the penalty parameter whenever the relaxed slacks approach their upper bounds. This ensures that the diagonal terms will not prematurely go to 0. In fact, while the algorithm described in Section 2 jams at an infeasible point on the so-called Wächter-Biegler problem of [24], the penalty approach solves it easily to optimality.

3.4.2. *Relaxation and Infeasibility Detection.* Another benefit of a penalty approach is due to the relaxation of the slack variables. The primal-dual pair (11)-(12) always has a feasible solution pair, and therefore, optimal primal-dual solutions to these problems always exist. Therefore, a provably convergent NLP algorithm is guaranteed to find a local optimal solution to (11). If this solution has the property that for at least one $i = 1, \ldots, m$, $\xi_i \to a$ for some scalar $a > 0$ as $c_i \to \infty$, then the original problem is locally primal infeasible, that is the algorithm fails due to its inability to reduce infeasibility even though there may be a feasible solution reachable from a different starting point. Similarly, if $\psi_i \to a$ as $b_i \to \infty$ for any $i = 1, \ldots, m$ for some strictly positive scalar a, then the original problem is locally dual infeasible.

It is impractical to allow a penalty parameter to become infinite. Therefore, we have implemented both a primal and a dual feasibility restoration phase. The primal feasibility restoration phase is similar to Phase I of SNOPT [11], in that the problem

$$\begin{array}{ll} \min\limits_{x,w,\xi} & c^T\xi \\ \text{s.t.} & h(x) + w = 0 \\ & -\xi \le w \le b \\ & \xi \ge 0 \end{array} \tag{21}$$

is solved in order to minimize infeasibility. It differs from SNOPT's version in that the primal slack variables are still bounded above by the dual penalty parameters. Since these parameters get updated whenever necessary, we can always find a feasible solution to (21). If the optimal objective function value is nonzero (numerically, greater than the infeasibility tolerance), a certificate of primal infeasibility is issued.

Similarly, the dual feasibility restoration phase solves the dual feasibility problem:

$$\begin{array}{ll} \max\limits_{y,\psi} & f(x) - \nabla f(x)^T x - b^T\psi \\ \text{s.t.} & \nabla f(x) - A(x)^T y = 0 \\ & -\psi \le y \le c - \psi \\ & \psi \ge 0. \end{array} \tag{22}$$

If the penalty objective is nonzero at the optimal solution to (22), then the problem (2) is determined to be dual infeasible. It should be noted that both the primal and the dual certificates of infeasibility apply only to *local* infeasibility, where local infeasibility refers to a point which is a local nonzero minimizer of the infeasibility.

3.4.3. *Finding nonKKT Solutions.* Unlike linear programming, the duality gap at the optimal solution of an NLP does not have to be 0. As in the case of problem 13 from the Hock and Schittkowski test set [14], the unique primal optimal solution may not have a corresponding dual solution. Thus, the optimum is not a KKT point, and no other KKT points exist.

As shown in [2], the purely primal penalty approach produces a KKT point close to the unique optimum. In fact, the KKT point approaches the optimum as the penalty parameters, and therefore the dual variables, go to ∞. Therefore, the exact penalty method behaves as an inexact method in the absence of KKT solutions. In [2], this behavior is proved for the general NLP case. This is still consistent with the primal-dual penalty approach. Numerical results given in Table 13 of Section 5.2 show that the primal-dual penalty approach produces results that exhibit this behavior on problem 13 from the Hock and Schittkowski test set.

3.4.4. *Unboundedness of Optimal Sets of Primal and Dual Variables.* Optimality condition (17) ensures that the Lagrange multipliers are bounded above by the penalty parameter. For problems such as mathematical programs with equilibrium constraints (MPECs), this feature is particularly attractive as their optimal Lagrange multiplier sets are unbounded. Similarly, an optimal set of primal variables can also be unbounded, for example

$$\begin{array}{lrcl} \min\limits_{x_1,x_2} & x_1^2 - x_2^2 & & \\ \text{s.t.} & x_1 - x_2 & \geq & 0 \\ & x_1, x_2 & \geq & 0. \end{array}$$

For a primal-dual interior-point method, such as LOQO, such unboundedness will cause numerical difficulties as the algorithm will try to find the analytic center of the face of optimal solutions, and this center will be at infinity. Therefore, employing a penalty approach with sufficiently large penalty parameters to admit at least one set of optimal primal-dual solutions will allow the algorithm to find the center at a finite solution.

3.5. **Switching in and out of a Penalty Approach.** As numerical studies such as [6] and [18] show, LOQO (and other interior-point NLP solvers) are fairly robust and efficient just as they are. Bringing relaxation variables and penalty parameters into the model may require additional iterations to be performed, especially if the solver will need to first increase the penalty parameters to admit an optimal solution. In fact, in the next section, we will show that using a primal-dual penalty approach to solve our problems from start to finish significantly increases the total number of iterations and the total runtime on a standard set of problems.

Therefore, given the benefits outlined above, we have decided to use the penalty approach only as necessary. That is, the problem will be solved using the algorithm described in Section 2, without the penalty method, unless one of the following occurs:

(1) Numerical problems arise: If any of the diagonal entries in the lower right corner of the matrix in (5) go to 0 prematurely, as in the Wächter-Biegler

problem, or if the reduced KKT system (5) becomes inconsistent and requires iterative refinement, the regularization aspect of a penalty method would help to remedy the numerical problems that arise in the solution of the reduced KKT system. The numerical problems may also arise due to functions that become nondifferentiable at certain points along the algorithm's trajectory toward a solution. Restarting the algorithm, this time in penalty mode, would change this trajectory, and unless the point of nondifferentiability is at the optimal solution, it can be avoided.

(2) There are too many step cuts: The length of a step taken along the computed step direction may have to be substantially cut before the step can sufficiently reduce the barrier objective or the primal infeasibility. In this case, the algorithm will not be able to make good progress from the current point. If the steplength is halved at least 10 times, we will switch over to penalty mode.

(3) Sufficient progress toward optimality cannot be made: The Armijo condition employed in the steplength control determines measures of sufficient progress, based on the computed step direction and the gradient of the Lagrangian at the current solution. If the norm of either of these vectors is too small or if the Armijo condition would admit an increase in either the barrier objective or the primal infeasibility, then the Armijo criteria are no longer reliable, and we cannot be expected to make sufficient progress toward optimality. In this case, we will switch over to penalty mode.

If any of these problems arise in penalty mode, and primal or dual feasibility have not been attained, LOQO will switch into feasibility restoration mode on either the primal side or the dual side, as necessary. As noted before, feasible solutions always exist for the penalty problems, and the algorithm proceeds in penalty mode with the original objective functions after a feasible point has been attained. If the problems above are encountered again, the algorithm enters primal or dual feasibility restoration mode, this time solving the feasibility problem to optimality. If the relaxation on either the primal or the dual side cannot be driven to 0, we have a certificate of (local) primal or dual infeasibility. Otherwise, the algorithm will leave penalty mode and proceed from a primal-dual feasible pair of solutions.

4. Warmstarting

Warmstarting is the use of information obtained during the solution of a problem to solve the subsequent, closely-related problems. Interior-point methods have long been perceived as being inferior in terms of warmstarting, especially in linear programming when compared to a simplex method that only requires a few pivots after a small data perturbation. For NLP, however, even small data perturbations can greatly affect the structure of the problem when they are embedded in highly nonlinear functions. Little is known about warmstarting in such cases. A major thrust of this paper is that, given limited problem sets to work with, we demonstrate that an interior-point method with an exact primal-dual penalty approach shows promise.

In [4], the authors highlight three possible problems which can arise with warmstarts:

(1) The diagonal entries of the matrix in (5) are either 0 or ∞ (numerically, a very large number) at the optimal solution of the original problem. In the

case of NLP, a perturbation to the problem can affect the Hessian, Jacobian, and right-hand side of (5), but the entries on the lower right diagonal of (5) will remain the same as before. Any change to the row of an inactive constraint, which has an infinite diagonal entry, can be absorbed by a very small change to its Lagrange multiplier. If the inactive constraint is to be active for the new problem, however, the Lagrange multiplier may need to be moved significantly away from 0.

(2) If an active constraint is perturbed, then the step direction for its slack variable, which is currently at 0, may be negative. The barrier then forces the steplength to be cut until it is very small. The algorithm stalls at a nonoptimal point, thus exhibiting the same behavior as jamming.

(3) If an active constraint is to become inactive, its Lagrange multiplier may move to 0 even though the slack variable cannot make progress. Similarly, if an inactive constraint is to become active, its slack variable may move to 0 even though its Lagrange multiplier cannot make progress. In both of these cases, degeneracy arises, and 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, effectively producing a coldstart.

A small numerical example illustrating these points is provided in [4]. The remedy proposed in that paper is the use of an exact primal-dual penalty approach in the case of linear programming, and we are expanding the same argument to the case of NLP in this paper. The numerical results of the next section illustrate the effectiveness of this approach.

5. Numerical Results

The interior-point method outlined in Section 2 is implemented in LOQO, Version 6.06. We will refer to this code as the "default" in our numerical results. We also implemented the primal-dual penalty approach in LOQO. We first start with a presentation of numerical results from using the primal-dual penalty approach in warmstarting several groups of problems from the CUTEr test set [13], the Hock and Schittkowski test set [14], nonlinear programming problems arising in engineering [22], goal programming problems [15], and mixed-integer nonlinear programming (MINLP) problems from the MINLPLib [8] test set.

We also present results for the use of the primal-dual penalty approach for general nonlinear programming. We have a version of LOQO that uses the penalty method from start to finish, without the switching conditions outlined in Section 3, and we will call this version the "penalty" code. Most of our numerical analysis will instead focus on the "hybrid" code, which implements the primal-dual penalty, but only invokes it in accordance with the switching conditions. The testing for these results was performed on the CUTEr set [13].

Finally, one of the advantages of a primal-dual, rather than just a primal, penalty approach is the ability to perform dual infeasibility detection. We present some numerical results to demonstrate this capability.

5.1. **Warmstarts.**

As discussed in the previous section, an important benefit of the primal-dual penalty approach is the ability to warmstart by using the solution of

Problem	Avg Warmstart Iters	Avg Coldstart Iters	Avg Reduction
antenna	34.0	27.8	-24.28%
blend	11.2	20.6	45.44%
chemeq	40.2	299.2	86.46%
fir_convex	37.8	42.0	8.61%
hang_midpt	9.6	45.8	78.96%
hydrothermal	28.2	28.0	-0.71%
kowalik	7.4	11.0	32.73%
markowitz	7.4	14.0	47.14%
nb_L2	6.2	23.0	72.82%
nnls	7.6	13.0	41.54%
robotarm	19.6	22.8	14.07%
rocket	32.8	57.4	36.72%
sawpath	2.8	11.0	74.55%
shekel	5.2	18.6	70.30%
springs	5.0	59.4	89.76%
steiner	17.8	23.0	23.75%
wbv	11.8	17.6	28.86%

TABLE 1. Summary numerical results for penalty warmstart approach on NLPs arising in engineering applications.

an initial problem to solve a closely related, possibly perturbed, problem. Gathering a test suite is not challenging in the case of linear programming, as discussed in [4], because we can simply perturb the data for any such problem. In nonlinear programming, however, a meaningful test set is hard to obtain, and we report here on several examples from literature. We will also report on reoptimization of the same problem from a boundary point close to the optimum, which has issues similar to warmstarting.

5.1.1. *NLPs from Engineering Applications.* We attempted to develop an initial test set by using NLPs arising in engineering and perturbing problem parameters. These parameters can be the wavelength in an antenna array problem, bounds on the controls applied in the range maximization of a hang glider, initial or final positions of a rocket launching problem, coordinates of points in a nonlinear least squares problem, or the annual returns in a financial engineering problems. The sample problems used were gathered from [23] by taking all problems that arise from real-world applications and contain tunable parameters. For each instance, we applied one or more random perturbations to the problem parameters, and the order of the perturbation was determined to ensure that the solution to the original problem would no longer be optimal for the perturbed problem. The model files and the command files used in the perturbation can be obtained from [3]. Summary results for each problem are provided in Table 1, and detailed numerical results are given in Tables 4 and 5 in the Appendix. Note that the averages in Table 1 are provided over 5 instances of each problem and the average reduction is the average of the individual reductions.

Several observations can be made from the results. First, there are both convex and nonconvex problems in this test suite, and the performance of the primal-dual penalty approach differs between these two categories of problems. Nonconvex problems such as *antenna* and *hydrothermal* can vary greatly in the effectiveness of a warmstart, while convex problems such as *chemeq* and *markowitz* are fairly consistent. This difference can be attributed to the modifications made to the diagonal of the Hessian during Cholesky factorization, and different modifications can produce very different step directions. As the size of the diagonal modification increases,

the resulting step direction gets closer to the steepest descent direction, which is not effective on a nonconvex problem. Since the magnitude of the modification depends on the Hessian at the current point, coldstarts and warmstarts may apply different modifications. This results in great variations in the effectiveness of the warmstart as compared to the coldstart.

For nonconvex problems, one might also expect that the objective function might be convex in a neighborhood of the optimal solution. If the solution of the perturbed problem is in this neighborhood, a warmstart may save time and iterations by avoiding the modifications to the diagonal of the Hessian. In order to promote this behavior, the initialization parameter τ in (20) has been changed to 10^{-14} for the nonconvex problems. The nonconvex problems in the test set are: *antenna*, *blend*, *hang_midpt*, *hydrothermal*, *kowalik*, *robotarm*, *rocket*, *sawpath*, and *shekel*.

Overall, the warmstart approach reduces the number of iterations by an average of 43%. This reduction is especially crucial for problems such as *nb_L2*, where the solution of the problem is time-consuming, and each iteration saved results in significant savings in runtime. Similarly, being able to re-solve the portfolio optimization problem *markowitz* more efficiently can allow for more frequent updating of allocations in order to respond to market changes better.

5.1.2. *Quadratic Programming.* Since quadratic programming problems (QPs) have well-structured data elements, it is relatively simple to perturb these elements to obtain new, closely related problems. Given a QP with the following structure

$$\begin{array}{lll} \min\limits_{x} & c^T x + \frac{1}{2} x^T Q x & \\ \text{s.t.} & Ax & \geq \quad b \\ & x & \geq \quad 0, \end{array}$$

where $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, and $A \in \mathbb{R}^{m \times n}$ are the data elements, we can apply random perturbations to c, A, and b. In order to obtain a meaningful comparison between cold and warmstarts, we avoided the possibility of the approaches going to different solutions by ensuring that the numerical testing was performed only on convex QPs. To that end, each problem tested has a positive definite Q, which does not get perturbed to avoid introducing nonconvexity. The problems were obtained from the CUTEr test suite [13], and we picked all convex QPs solvable to optimality by LOQO.

The random numbers used to perturb the data elements 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:

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

Changed	δ	Avg Warmstart Iters	Avg Coldstart Iters	Avg Reduction
b	0.0001	24.63	53.54	45%
b	0.001	34.40	68.48	49%
b	0.01	35.97	66.63	42%
c	0.0001	10.89	39.80	49%
c	0.001	10.29	39.07	49%
c	0.01	10.08	39.51	48%
A	0.0001	17.19	53.91	55%
A	0.001	16.89	54.11	55%
A	0.01	18.11	54.64	49%

TABLE 2. Summary numerical results for penalty warmstart approach on convex QPs from the CUTEr test set.

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 (0.0001, 0.001, 0.01). We report on the distance between the optimal solutions to the original and perturbed problems, and compare iteration counts for the warmstart and the coldstart solutions. These results are summarized by perturbation type and level in Table 2. Average performance of the warmstart approach on each problem is presented in the Tables 16-18 of the Appendix, and detailed results on each perturbation instance can be obtained from [3]. Note that the average reductions reported in Table 2 are the averages of the individual reductions for each instance.

The distance between the primal 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 *qpcboei1* in Table 16 of the Appendix, the performance may deteriorate. Nonetheless, such a distance represents a significant enough change in the problem that a coldstart may be just as close to the optimal solution as, if not closer than, a warmstart.

It is also important to note that while the data elements of a QP may lend themselves well to being randomly perturbed, the problem may have a special structure that should not be changed. For example, the problem *ksip* represents the discretization of a semi-infinite QP and has constraints of the form

$$\sum_{j=1}^{20} \left(\frac{i}{1000}\right)^{j-1} x_j \geq \sin\left(\frac{i}{1000}\right), \quad i = 1, \ldots, 1000.$$

In a real world example, such a special structure would not be perturbed. Additionally, the constraint coefficients are quite small (some of them on the order of 10^{-57}) which causes some numerical issues. Perturbing the coefficients themselves

Changed	δ	Avg Warmstart Iters	Avg Coldstart Iters	Avg Reduction
c	0.0001	10.48	28.82	57%
c	0.001	10.67	29.24	57%
c	0.01	10.45	30.43	56%

TABLE 3. Summary numerical results for penalty warmstart approach on NLPs from the CUTEr test set that have linear objective functions.

using the scheme described above may lead to negligible changes. Accordingly, despite trying for several dozen instances, we were not able to obtain a perturbation of the constraint coefficients that changed the optimal solution. Also due to the magnitude of the constraint coefficients, applying a perturbation to the right-hand sides of the constraints would require the solution to moved significantly in order to satisfy the new constraints. Therefore, the distance between the old and the new optimal solutions is sizable, and a warmstart was not beneficial. Nevertheless, we have decided to include this problem in our numerical tests in order to emphasize the need for a more extensive and appropriate test suite for warmstarting.

5.1.3. *Nonlinear Programming with Linear Components.* The random perturbation approach described above for QPs can be extended to NLPs that have linear objective functions or linear constraints. NLPs with linear objective functions have the form

$$\begin{array}{ll} \min\limits_{x} & c^T x \\ \text{s.t.} & 0 \leq h(x) \leq r \\ & l \leq x \leq u, \end{array} \tag{23}$$

where $c \in \mathbb{R}^n$. Similarly, NLPs with linear constraints have the form

$$\begin{array}{ll} \min\limits_{x} & f(x) \\ \text{s.t.} & Ax \leq b \\ & l \leq x \leq u, \end{array} \tag{24}$$

where $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$. We present here numerical results for perturbing c for problems of the form (23) and for perturbing A and b for problems of the form (24).

All test problems were obtained from the CUTEr test set [13]. In order to ensure that the coldstart and warmstart comparisons reach the same solution, we picked convex problems. Since for an NLP, proving convexity analytically can be challenging, we used only those problems for which LOQO did not need to modify the diagonal of the Hessian during the Cholesky factorization phase. While this approach may eliminate more problems than necessary, it is much simpler to implement than checking individual solutions and still resulted in a sufficiently large test set.

Summary numerical results are provided in Tables 3 and 4, and average results for individual problems are provided in Tables 19-23 in the Appendix. Detailed results for each problem instance can be obtained from [3]. Note that the average reductions reported in Tables 3 and 4 are the averages of the individual reductions for each instance.

Changed	δ	Avg Warmstart Iters	Avg Coldstart Iters	Avg Reduction
b	0.0001	9.42	16.79	46%
b	0.001	10.51	17.09	39%
b	0.01	11.80	16.89	31%
A	0.0001	11.06	17.54	33%
A	0.001	11.52	17.89	32%
A	0.01	14.01	18.11	21%

TABLE 4. Summary numerical results for penalty warmstart approach on NLPs from the CUTEr test set that have linear constraints.

5.1.4. *Goal Programming.* Goal programming is a technique for solving multicriteria optimization problems. There are several objective functions, representing several goals, that are prioritized in order of importance. For each goal, a new problem is solved with the objective function corresponding to the goal and additional constraints implying that the level of attainment of the more important goals cannot be weakened.

In this paper, we consider a nonlinear goal programming problem from [15]. The problem is as follows:

- Priority 1 goal: $3x_1^2 + x_2^2 - 12x_1 - 8x_2 \leq -25$
- Priority 2 goal: $(1 - x_1)^2 + (2 - x_2)^2 \leq 1$.

We would like to get each of these goals satisfied as much as possible, so we start by adding nonnegative slack variables, d^- and d^+, into each goal. The first subproblem to solve is

$$\begin{array}{lrcl} \min\limits_{x,d^-,d^+} & d_1^+ & & \\ \text{s.t.} & 3x_1^2 + x_2^2 - 12x_1 - 8x_2 + d_1^- - d_1^+ & = & -25 \\ & (1 - x_1)^2 + (2 - x_2)^2 + d_2^- - d_2^+ & = & 1 \\ & x_1, x_2, d_1^-, d_1^+, d_2^-, d_2^+ & \geq & 0. \end{array}$$

At the optimal solution to this problem, we have that d_1^+ is equal to 0. Therefore, we have been able to satisfy our Priority 1 goal. The second subproblem must try to achieve the Priority 2 goal without weakening our attainment of the first goal. It can be expressed as:

$$\begin{array}{lrcl} \min\limits_{x,d^-,d^+} & d_2^+ & & \\ \text{s.t.} & 3x_1^2 + x_2^2 - 12x_1 - 8x_2 + d_1^- & = & -25 \\ & (1 - x_1)^2 + (2 - x_2)^2 + d_2^- - d_2^+ & = & 1 \\ & x_1, x_2, d_1^-, d_2^-, d_2^+ & \geq & 0. \end{array}$$

This problem has one less variable and has a different objective function. The primal infeasibility does not change, but the dual infeasibility gets affected by the change in the objective function. Therefore, it is important that a penalty method to relax this problem should have a dual component which relaxes the dual slacks. In fact, without the penalty approach, LOQO solves the second problem in 20 iterations. With the primal-dual penalty turned on to perform a warmstart from the first problem's solution, LOQO solves the second problem in 10 iterations.

5.1.5. *Hock and Schittkowski problems.* As discussed above, it is particularly challenging to meaningfully perturb an NLP. Therefore, to test our approach, we took

a different point of view, and perturbed the optimal solution of a problem and re-optimized it from this nearby point. In order for the test to appropriately challenge the solver, it is not sufficient to just perturb the primal and the dual variables. The warmstart solution should have complementarity preserved, with changes to the active set whenever appropriate. If the nonzero factor in the complementary pair is significantly larger than 0, moving it to 0 can move a dual variable or a slack too far from the optimal solution, effectively creating a coldstart that can be worse than the initial solution to the problem. Thus, we have perturbed only those complementary pairs where the nonzero element is no more than 10^{-2}. In this case, we randomly picked the member of the complementary pair to be zero, and then assigned a perturbed value to the nonzero member. Doing so preserved complementarity, and such a solution can be the optimum for a closely related problem. The only issue in this case is that the number of active set changes is not significant. This problem would be remedied by the availability of a good test set for warmstarting.

We tested 103 problems from the Hock and Schittkowski [14] test suite. The variables (decision and slack) 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 each perturbation, a scalar, ϵ, was read from this list to determine the magnitude of the perturbation. In order to match the scale of the numbers in the problem, the following scheme was used when perturbing variable x_j:

$$x_j^{(0)} = \begin{cases} \delta\epsilon, & \text{if } x_j^* = 0 \\ x_j^*(1+\delta\epsilon), & \text{otherwise.} \end{cases}$$

where the initial, perturbed solution is denoted by $x^{(0)}$, the optimal solution is denoted by x^*, and δ is a parameter varied to observe the effects of different levels of perturbation. The dual variables, y, and the primal slacks, w, were perturbed using the same scheme, while ensuring that they did not become negative. For each complementary pair, (w_i, y_i), $i = 1, \ldots, m$, where the nonzero element is no more than 10^{-2}, another random number ϵ_2 was drawn. If $\epsilon_2 < 0$, we set $w_i = 0$. Otherwise, we set $y_i = 0$. For each original problem, we present the numerical results obtained by 3 different perturbations at three different levels of δ (0.0001, 0.001, 0.01). We report on the distance between the optimal solutions to the original and perturbed problems, and compare iteration counts for the warmstart and the coldstart solutions. These results are summarized in Table 5 and presented in detail online at [3]. Note that the average reductions reported in Table 5 are the averages of the individual reductions for each instance.

5.1.6. *MINLPs.* We also implemented a branch-and-bound algorithm to solve mixed-integer nonlinear programming (MINLP) problems, using LOQO to solve the nonlinear relaxation at each node of the tree, and examined the performance of our approach on warmstarting after variable bound perturbations. The test suite is a group of 37 problems from MINLPLib [8], which were chosen among the problems with less than 10 discrete variables. The criteria for choosing these problems were that the coldstart and the warmstart arrived at the same optimal solution (there can be a difference for problems with nonconvex subproblems) and that there be no more than 50 subproblems. Full testing results for the 37 problems are provided at [3].

δ	Reduction in Iters	Primal Dist	Dual Dist
0.0001	39%	3.70e-03	2.13e-03
0.001	33%	6.14e-03	1.72e-02
0.01	24%	1.90e-02	3.40e-02

TABLE 5. Average performance of the primal-dual penalty approach when warmstarting 103 problems from the Hock and Schittkowski test set. δ is the perturbation factor, the second column is the average reduction in iterations from the coldstart to the warmstart, and Primal Dist and Dual Dist are the scaled Euclidean distances between the warmstart and the optimal solutions for the primal and the dual variables, respectively.

In Table 6, we provide a few representative examples of our results. 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. For each problem, we show the results for every node in the branch-and-bound tree.

We have made several observations during the solution of the MINLPs. First, many 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, note that for many of the subproblems, the scaled distance between the solutions (both primal and dual) of the problem and its parent is quite large. When the distance is small, we can generally see an improvement in the number of iterations, but as is consistent with our observation above, when either of the distances is large, a warmstart may indeed hurt the problem because the initial solution is too far away. Third, perhaps the biggest advantage of using a penalty approach is the infeasibility detection scheme which is necessary for many subproblems. Without this capability, the algorithm will run to iteration limit, wasting runtime and not giving a conclusive result as to the status of the subproblem. As seen in problem *nvs03* in Table 6, the penalty approach is able to detect infeasibility of the subproblems after a reasonable number of iterations.

5.2. **The Primal-Dual Penalty Approach for Nonlinear Programming.** We have tested the primal-dual penalty approach, both the "penalty" and the "hybrid" codes, on 1073 problems from the CUTEr test suite [13] and compare our results to the nonpenalty version of LOQO (Version 6.06) as described in Section 2. Summary results are provided in this section. Full testing results are not included with this paper due to space limitations but are available online at [3].

Tables 7 and 8 present an overview performance for each code on the full set of problems. The first column is the number of problems solved by each code, and the second and third columns are the total number of iterations and the total runtime (in CPU seconds) over the commonly solved problems. By "commonly solved problems," we refer to those problems solved by both codes used in each comparison. Note that the total iterations and total runtimes for the default code differ between the two tables, and this is due to the fact that the sets of commonly

Problem	Node	ColdIters	WarmIters	Δ	PrimalDist	DualDist
ex1223a	1	14	13	1	2.08E-002	9.66E-001
ex1223a	2	11	10	8	3.18E-002	1.70E+000
ex1223a	3	12	11	0	2.81E-002	6.73E-002
ex1223a	4	12	10	1	1.79E-002	1.39E-001
ex1223a	5	12	14	1	5.85E-003	5.10E-002
ex1223a	6	10	10	1	4.47E-018	8.16E-001
ex1223a	7	12	10	2	3.43E-004	1.07E-001
ex1223a	8	12	10	0	2.67E-002	2.38E-001
ex1223a	9	12	11	1	4.93E-015	2.53E-001
ex1223a	10	22	28	16	7.02E+000	5.07E-001
ex1224	1	19	13	2	2.83E-002	1.33E+002
ex1224	2	19	16	2	1.05E-001	5.93E-005
ex1224	3	18	17	1	2.67E-002	1.85E-004
ex1224	4	17	17	2	2.57E-005	1.37E+002
ex1224	5	17	21	2	3.10E-002	3.69E-004
ex1224	6	20	30	2	1.25E-001	8.02E-005
ex1224	7	35	13	1	1.05E-002	1.25E-003
ex1224	8	30	33	2	3.48E-004	9.29E+001
ex1224	9	51	36	0	1.70E-003	1.31E-004
ex1224	10	56	53	0	1.98E-004	1.95E-005
ex1224	11	47	28	3	1.01E-001	6.24E-005
ex1224	12	55	48	3	1.36E-002	1.51E-003
ex1224	13	25	18	1	3.04E-003	6.20E-005
ex1224	14	24	27	1	2.53E-002	1.32E-006
ex1224	15	31	41	2	1.27E-001	3.66E-004
nvs03	1	21	16	3	8.92E-003	9.84E-001
nvs03	2	11	15	0	8.33E-003	7.73E+000
nvs03	3	21	18	2	8.26E-003	9.98E-001
nvs03	4	(IL)	59(INF)	4	1.55E-002	2.21E+002
nvs03	5	21	21	1	3.57E-002	1.44E+002
nvs03	6	(IL)	60(IMP)	2	1.64E-002	2.37E+001
nvs03	7	23	21	2	3.42E-003	6.89E-001
nvs03	8	(IL)	56(INF)	4	9.22E-001	4.87E+000

TABLE 6. Numerical performance of LOQO on the MINLPLib test suite. "Node" indicates the number of the node on the branch-and-bound tree, "ColdIters" is the coldstart iteration count, "WarmIters" is the warmstart iteration count, Δ is the number of changes to the active set, "PrimalDist" and "DualDist" are the relative distances between the parent and the child node for the primal and the dual variables, respectively. (IL) indicates that the algorithm reached the iteration limit, (INF) indicates that the algorithm determined that the problem is infeasible, and (IMP) indicates that further improvement could not be made on a problem.

	Problems Solved	Total Iterations	Total Runtime
Default	986	50520	3029.73
Penalty	977	60820	7096.78

TABLE 7. Comparison of the performance of the default and the penalty codes on the CUTEr test suite. The total iterations and runtime reported are only for commonly solved problems.

solved problems are different for the two comparisons. In Tables 7 and 8, there are 941 and 974 commonly solved problems, respectively.

	Problems Solved	Total Iterations	Total Runtime
Default	986	57116	3927.50
Hybrid	1002	39777	4881.23

TABLE 8. Comparison of the performance of the default and the hybrid codes on the CUTEr test suite. The total iterations and runtime reported are only for commonly solved problems.

In Table 7, a comparison between the default and the penalty codes are provided. The penalty code, whose performance can be greatly affected by the choice of the penalty parameters, does suffer from the initializations described in Section 3. Since the initial estimates of the slack variables may be far from the optimal solution, setting the penalty parameters too small may need too many iterations to update them to the right values. On the other hand, setting them too large may seek to minimize the infeasibility first. For an infeasible interior-point NLP solver such as LOQO, focusing first on the attainment of feasibility may be detrimental to the algorithm's performance. We have verified that many of the small problems that the penalty code fails on can be solved by a careful calibration of the initial values of the slack variables. This shows promise for the hybrid code, which will start after several iterations of the algorithm or after a warmstart, with good estimates for the slack variable values and, therefore, for the penalty parameters. Additionally, using the penalty code from start to finish on the commonly solved problems takes almost 20% more iterations and twice as much time. The difference in the total runtime is mainly due to two problems, *porous1* and *msqrtb*, where the default code finds the solution in fewer iterations and each additional iteration is very expensive. These two problems combined contribute to about 2800 seconds of the difference between the total runtimes.

As shown in Table 8, the hybrid code retains the theoretical robustness of the penalty code and improves on its computational robustness by using better estimates of the penalty parameters. By switching over only as necessary, it is far more efficient than the penalty code. In fact, by providing regularization when necessary, it also improves on the efficiency of the default code. Therefore, the rest of the numerical results in this section will concentrate on the hybrid code. It is also important to make three more notes here: (1) the hybrid method switched into penalty mode on 253 problems, (2) the difference between the default and the hybrid performance in iterations is attributable in most part to five problems, *s380*, *s380a*, *palmer7a*, *rk23*, and *haifam*, which together account for a difference of about 14,000 iterations, and (3) the difference between the default and the hybrid performance in runtime is due entirely to one problem *reading7*, which accounts for a difference of approximately 1000 CPU seconds.

We now examine the problems documented in Table 9 that were solved by the hybrid code, but not by the default code. There are 28 such problems, and of these, one of the problems, *s220*, switched to the penalty after predicting that sufficient progress cannot be made using the step directions calculated without the penalty. The rest of the problems switch over after making too many steplength cuts in the Markov filter. These cuts may be due to numerical issues, unboundedness of the optimal set, or any other reason that may affect the quality of the step direction.

artif	brainpc0	brainpc5	catenary
cresc100	cresc132	discs	eigmaxc
eqc	gausselm	hager4	kiwcresc
palmer5a	palmer7e	polak5	polak6
reading1	robotarm	s220	s355
s355a	sineali	snake	spiral
trainf	twirimd1	vanderm3	zamb2

TABLE 9. Problems solved by the hybrid code and not by the default.

Local infeasibility: *powellsq*
Numerical problems in penalty mode: *yao*
Less accuracy: *deconvc*, *dixchlnv*, *hvycrash*, *spanhyd*, *steenbre*, *steenbrg*, *vanderm1*
Iteration Limit: *lukvli10*, *methanol*, *reading4*

TABLE 10. Problems solved by the default code and not by the hybrid code.

There are also 12 problems that are solved to optimality by the default code but not by the hybrid code. These problems are categorized in Table 10. Some of this difference can be attributed to the different trajectories taken by the two approaches. For example, in solving problem *powellsq*, the hybrid code goes to a neighborhood in which feasibility cannot be attained. The seven problems that end with less accuracy do so because they have reached the optimal objective function value on the primal side and are primal feasible, but the dual objective function and/or dual infeasibility have not yet reached their optimal values. In this case, the purely primal nature of the steplength control mechanism is the cause of the problem, as we check for predicted decrease in either the barrier objective or the primal infeasibility. Since the primal solution does not need to move to satisfy the stopping criteria, the predicted decrease is too small. We conclude by deciding that the current solution cannot be improved. This is the correct conclusion for the primal solution, but the dual solution can still improve. One remedy for this situation is to consider including dual measures in the steplength control mechanism, as in [21].

Next, we examine the problems that remain unsolved by both the default and the hybrid codes. A distinct advantage of the hybrid code is that having tried the penalty code and, if necessary, the feasibility restoration approaches, we can make a determination as to whether the problem is locally primal or dual infeasible and whether we have made all the possible progress toward an optimal solution. To that end, the hybrid code has identified the problems in Table 11 as being locally primal infeasible.

These three problems are in addition to *powellsq* as discussed in Table 10. Since the algorithm is trying to attain a local optimum, the problems are classified as

argauss
himmelbd
lewispol

TABLE 11. Primal infeasible problems identified by the hybrid code.

Problem Name	Complementarity	Primal Feasibility	Dual Feasibility
allinitc	1.00e-5	1.00E-008	5.00E-008
brainpc6	1.00e-8	8.00E-004	4.00E-008
brainpc9	1.00e-7	1.00E-005	2.00E-005
hs013	1.00e-3	8.00E-008	9.00E-007
s221	1.00e-4	5.00E-008	6.00E-007
steenbrb	1.00e-7	4.00E-017	9.00E-007
steenbrf	1.00e-3	4.00E-013	1.00E-003

TABLE 12. Problems not solved by the default code and where sufficient progress cannot be made, as identified by the hybrid code.

being locally infeasible. We will discuss results of dual infeasibility detection later in this section.

The hybrid code stops the problems in Table 12 before reaching the desired levels of accuracy because further progress cannot be made, even after trying the penalty approach. In Table 12, we provide the scaled measures of optimality at the point the hybrid code stops working on the problem and signals the user that the "current point cannot be improved." Two of these problems, *hs013* and *s221*, have a unique optimum that is not a KKT point, and as discussed in the previous section, such a point can be approached as the penalty parameters increase to infinity. The hybrid code determines that it has increased the penalty too many times and stops. It is also important to note that the optimality measures reported in Table 12 are the KKT conditions of the original problem and not the relaxed penalty problem.

There are still 50 problems that remain unsolved by both the default and the hybrid codes. Some of these problems have points of nondifferentiability that cannot be avoided and some have scaling issues that cause numerical problems. While we cannot overcome the nondifferentiability, we are looking into rescaling approaches for future versions of LOQO.

Next, we compare the hybrid code to the interior-point solver IPOPT [25] (Version 3.0.0). It should be noted that the basic approach of each algorithm is as described in Section 2. They differ in details such as Cholesky factorization routines, steplength control, handling of equality constraints, and regularization. The two codes also differ in their stopping criteria. While IPOPT relaxes variable bounds and uses the relaxed measures to stop, LOQO always computes the optimality measures of the original problem, even in penalty mode. This results in accuracy differences between the two codes. For example, on problem *hs013*, the solution reported by IPOPT as being optimal with less than 10^{-9} in all three unscaled measures of optimality is determined by LOQO to be locally dual infeasible and with only 10^{-4} as the complementarity.

	Problems Solved	Total Iterations
Hybrid LOQO	1016	55819
IPOPT	1013	39061

TABLE 13. Overview of the performance of hybrid LOQO and IPOPT on the CUTEr test suite. Total iterations are reported only on the commonly solved problems.

We provide a performance comparison in Table 13 showing that the hybrid approach is competitive in terms of robustness. After accounting for 7 problems (*chain*, *huestis*, *qpnboei1*, *reading1*, *sawpath*, *trainf*, and *zamb2*) which together make up about 14,000 of the difference in the numbers of iterations, we can see that the hybrid code is also competitive in terms of efficiency on the commonly solved problems. In order to deflect some of the accuracy level differences, we have included the problems attaining lower levels of accuracy than LOQO's default settings, as listed in Tables 10 and 12. Doing so ensures that problems such as *hs013* where LOQO stops within 3 digits of the true optimum and IPOPT stops within 2 digits are counted as solved by both codes. We have also included problems listed as "Solved to Acceptable Level" by IPOPT as solved in the above table. We have verified that for all of those problems both methods are approaching the same optimum.

5.3. **Dual Infeasibility Detection.** As discussed in the previous section, the primal-dual penalty approach can be used to detect dual infeasible problems. Given that a primal feasible solution exists, dual infeasibility indicates that a problem is primal unbounded. There are three such problems in the CUTEr test suite (*s255*, *s340*, and *s340a*). However, all three of these problems have free variables, and there are no corresponding slack variables to be relaxed to test our dual infeasibility detection approach. One possible remedy is to split a free variable, x, into

$$x - g + t = 0, \quad g \geq 0, \quad t \geq 0,$$

and relax and bound the primal slacks g and t and their corresponding dual slacks. Free variable splitting of this form has been removed from LOQO in its latest version due to efficiency issues, but it would have an advantage in terms of dual infeasibility detection.

Since a suitable test problem was not available in the CUTEr set, we formulated one of our own:

$$\begin{array}{lrcl} \min\limits_{x_1,x_2} & -x_1^4 - x_2^4 & & \\ \text{s.t.} & x_2^2 - x_1^2 & \geq & 0 \\ & x_1, x_2 & \geq & 0. \end{array}$$

In fact, LOQO enters dual feasibility restoration mode and finds that the optimal solution has a nonzero penalty. Therefore, the problem is determined to be locally dual infeasible.

In order to enter dual feasibility restoration mode, we monitored the increase in the dual penalty parameters due to the increase in the primal slacks. In this case, as x_1 and x_2 tend to infinity, so do the slack variables on their lower bounds. Therefore, the dual penalty parameters, which act as their upper bounds, are increased too many times.

In comparison, the primal feasibility restoration mode is also entered when the penalty approach has problems with the steplength control. This is due to the fact that the steplength control mechanism, which monitors the barrier objective and the primal infeasibility, is purely primal. Therefore, any issues with obtaining sufficient decrease may be remedied by re-initializing the penalty parameters and focusing solely on reducing primal feasibility.

If dual feasibility measures are to be considered within the steplength control scheme, we may be able to detect situations in which a dual feasibility restoration may also be used. Such a scheme is used in [21] as a filter with criteria from the KKT conditions (4).

6. Conclusion

While penalty methods have been studied before, either as a regularization or an elastic mode in the context of SQP, they have been limited to being purely primal. In this paper, we have extended the primal-dual penalty approach of [4] to NLPs, and this approach retains the desirable theoretical properties of its primal counterpart, but improves dual infeasibility identification and warmstarting capabilities. It also preserves the symmetry of a primal-dual interior-point method.

Also, as evidenced by the numerical results, the primal-dual penalty approach improves the robustness of an interior-point method on NLPs. In order to retain the efficiency of these methods, however, a hybrid approach that switches over to the penalty method only when needed works best.

While we have demonstrated the effectiveness of the primal-dual penalty approach for warmstarting NLPs, we would like to emphasize that more examples are needed to develop a truly meaningful test suite for warmstarting problems. As the work in this important area progresses, such a test suite will be vital to assessing the success of various proposed methods. We encourage all researchers to submit examples toward the development of this test suite.

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. Numerical testing results for the primal-dual penalty approach. http://www.pages.drexel.edu/~hvb22/penaltyweb.

[4] H.Y. Benson and D.F. Shanno. An exact primal-dual penalty method approach to warmstarting interior-point methods for linear programming. *Computational Optimization and Applications*, 2006. (to appear).

[5] H.Y. Benson, D.F. Shanno, and R.J. Vanderbei. A comparative study of large scale nonlinear optimization algorithms. In *Proceedings of the Workshop on High Performance Algorithms and Software for Nonlinear Optimization, Erice, Italy*, 2001.

[6] H.Y. Benson, D.F. Shanno, and R.J. Vanderbei. Interior-point methods for nonconvex nonlinear programming: Filter methods and merit functions. *Computational Optimization and Applications*, 23(2):257–272, November 2002.

[7] 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.

[8] M.R. Bussieck, A.S. Drud, and A Meeraus. MINLPLib - a collection of test models for mixed-integer nonlinear programming. *INFORMS Journal on Computing*, 15(1).

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

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

[11] P.E. Gill, W. Murray, and M.A. Saunders. User's guide for SNOPT 5.3: A Fortran package for large-scale nonlinear programming. Technical report, Systems Optimization Laboratory, Stanford University, Stanford, CA, 1997.

[12] 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.

[13] N.I.M. Gould, D. Orban, and Ph.L. Toint. CUTEr and SifDec: A Constrained and Unconstrained Testing Environment, revisited. *ACM Transactions on Mathematical Software*, 29(4):373–394, December 2003.

[14] W. Hock and K. Schittkowski. *Test examples for nonlinear programming codes*. Lecture Notes in Economics and Mathematical Systems 187. Springer Verlag, Heidelberg, 1981.

[15] G. Kuriger and A.R. Ravindran. Intelligent search methods for nonlinear goal programming problems. *INFOR: Information Systems and Operational Research*, 43(2):79–92, 2005.

[16] 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.

[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] J. Nocedal, J.L. Morales, R. Waltz, G. Liu, and J.P. Goux. Assessing the potential of interior-point methods for nonlinear optimization. Technical Report Report OTC-2001-6, Optimization Technology Center, 2001.

[19] J. Nocedal and R. A. Waltz. Knitro 2.0 user's manual. Technical Report OTC 02-2002, Optimization Technology Center, Northwestern University, January 2002.

[20] D.F. Shanno and R.J. Vanderbei. Interior-point methods for nonconvex nonlinear programming: Orderings and higher-order methods. *Math. Prog.*, 87(2):303–316, 2000.

[21] M. Ulbrich, S. Ulbrich, and L. Vicente. A globally convergent primal-dual interior point filter method for nonconvex nonlinear programming. *Mathematical Programming*, 100:379–410, 2004.

[22] R.J. Vanderbei. AMPL models. http://orfe.princeton.edu/~rvdb/ampl/nlmodels

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

[24] A. Wächter and L. Biegler. Failure of global convergence for a class of interior point methods for nonlinear programming. *Mathematical Programming*, 88(3):565–587, 2000.

[25] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Technical Report RC 23149, IBM T. J. Watson Research Center, Yorktown, USA, March 2004.

Problem	ColdIters	WarmIters	$\|x^I - x^P\|$	$\|y^I - y^P\|$
antenna-1	28	59	6.50E-02	2.23E-02
antenna-2	26	25	7.68E-03	7.53E-04
antenna-3	26	45	6.35E-03	9.79E-02
antenna-4	27	23	4.05E-02	3.85E-02
antenna-5	32	18	1.44E-02	2.30E-03
antenna-ave	27.8	34.0	2.68E-02	3.23E-02
blend-1	22	9	2.56E-05	4.67E-02
blend-2	20	12	4.56E-04	4.81E-02
blend-3	21	13	3.54E-04	3.11E-02
blend-4	20	12	4.38E-05	3.31E-02
blend-5	20	10	3.65E-04	2.98E-02
blend-ave	20.6	11.2	2.49E-04	3.78E-02
chemeq-1	292	27	3.89E-04	3.22E-02
chemeq-2	323	44	8.92E-05	2.60E-03
chemeq-3	312	43	3.75E-04	1.45E-03
chemeq-4	248	43	3.89E-04	1.43E-03
chemeq-5	321	44	5.55E-05	2.58E-03
chemeq-ave	299.2	40.2	2.59E-04	8.05E-03
fir_convex-1	33	38	2.33E-03	7.39E-01
fir_convex-2	39	31	7.35E-04	1.01E-03
fir_convex-3	45	43	3.49E-03	7.29E-01
fir_convex-4	53	42	3.62E-03	7.29E-01
fir_convex-5	40	35	4.09E-04	5.64E-04
fir_convex-ave	42.0	37.8	2.12E-03	4.40E-01
hang_midpt-1	43	11	1.39E-05	1.00E-04
hang_midpt-2	47	9	1.39E-05	9.99E-05
hang_midpt-3	40	7	1.39E-05	9.98E-05
hang_midpt-4	45	11	1.39E-05	1.00E-04
hang_midpt-5	54	10	1.39E-05	9.98E-05
hang_midpt-ave	45.8	9.6	1.39E-05	9.99E-05
hydrothermal-1	28	29	2.01E-03	5.24E-04
hydrothermal-2	28	26	6.50E-04	1.63E-04
hydrothermal-3	28	32	1.64E-03	7.93E-04
hydrothermal-4	28	29	4.34E-03	8.23E-04
hydrothermal-5	28	25	1.08E-03	9.10E-05
hydrothermal-ave	28.0	28.2	1.94E-03	4.79E-04
kowalik-1	11	9	3.69E-03	0.00E+00
kowalik-2	11	9	1.14E-03	0.00E+00
kowalik-3	11	7	5.61E-03	0.00E+00
kowalik-4	11	6	5.86E-03	0.00E+00
kowalik-5	11	6	6.37E-04	0.00E+00
kowalik-ave	11.0	7.4	3.39E-03	0.00E+00
markowitz-1	14	3	8.93E-04	1.81E-05
markowitz-2	14	5	3.11E-04	6.86E-06
markowitz-3	14	12	9.73E-04	2.09E-05
markowitz-4	14	11	9.05E-04	2.00E-05
markowitz-5	14	6	1.37E-04	3.62E-06
markowitz-ave	14.0	7.4	6.44E-04	1.39E-05
nb_L2-1	25	3	1.02E-03	4.76E-02
nb_L2-2	20	8	1.11E-02	4.66E-01
nb_L2-3	22	4	2.36E-03	1.09E-01
nb_L2-4	23	5	8.65E-03	3.74E-01
nb_L2-5	25	11	1.16E-02	4.80E-01
nb_L2-ave	23.0	6.2	6.95E-03	2.95E-01
nnls-1	13	6	5.04E-04	0.00E+00
nnls-2	13	9	9.18E-04	0.00E+00
nnls-3	13	9	7.11E-04	0.00E+00
nnls-4	13	9	8.08E-04	0.00E+00
nnls-5	13	5	4.95E-04	0.00E+00
nnls-ave	13.0	7.6	6.87E-04	0.00E+00

TABLE 14. Numerical performance of LOQO on NLPs when warmstarting a problem whose problem data have been modified. The iteration counts for the coldstart and warmstart solution of five perturbed problems are given, and the last 2 columns are the scaled Euclidean norm of the distance between the optimal solutions and the Lagrange multipliers of the initial and the perturbed problem. The problems labeled as "ave" refer to the average performance over the five problems.

Problem	ColdIters	WarmIters	$\|x^I - x^P\|$	$\|y^I - y^P\|$
robotarm-1	22	18	1.05E-04	1.41E-04
robotarm-2	23	20	6.82E-05	3.82E-04
robotarm-3	23	20	1.33E-04	2.10E-04
robotarm-4	23	20	7.77E-05	1.50E-04
robotarm-5	23	20	2.80E-05	1.57E-04
robotarm-ave	22.8	19.6	8.25E-05	2.08E-04
rocket-1	68	29	5.92E-03	1.10E-04
rocket-2	51	37	1.84E-03	3.43E-05
rocket-3	48	27	8.95E-03	1.66E-04
rocket-4	31	31	9.30E-03	1.72E-04
rocket-5	89	40	1.03E-03	1.91E-05
rocket-ave	57.4	32.8	5.41E-03	1.00E-04
sawpath-1	11	3	8.76E-10	6.84E-03
sawpath-2	11	3	2.64E-10	2.24E-03
sawpath-3	11	3	1.21E-09	9.97E-03
sawpath-4	11	3	1.25E-09	1.03E-02
sawpath-5	11	2	1.48E-10	1.26E-03
sawpath-ave	11.0	2.8	7.48E-10	6.12E-03
shekel-1	20	5	1.53E-05	0.00E+00
shekel-2	13	5	3.18E-06	0.00E+00
shekel-3	17	6	2.80E-05	0.00E+00
shekel-4	27	5	9.71E-03	0.00E+00
shekel-5	16	5	2.47E-06	0.00E+00
shekel-ave	18.6	5.2	1.95E-03	0.00E+00
springs-1	37	5	4.36E-04	4.13E-04
springs-2	68	4	1.36E-04	1.29E-04
springs-3	31	6	6.60E-04	6.25E-04
springs-4	98	6	6.85E-04	6.49E-04
springs-5	63	4	7.57E-05	7.17E-05
springs-ave	59.4	5.0	3.99E-04	3.78E-04
steiner-1	27	29	3.72E-04	9.49E-03
steiner-2	22	11	3.28E-04	3.51E-02
steiner-3	22	19	4.20E-04	1.19E-02
steiner-4	21	17	5.03E-03	4.30E-02
steiner-5	23	13	4.35E-04	4.38E-02
steiner-ave	23.0	17.8	1.32E-03	2.87E-02
wbv-1	15	9	3.29E-03	1.08E-02
wbv-2	22	11	4.88E-03	1.00E-02
wbv-3	22	11	5.71E-03	9.61E-03
wbv-4	15	9	2.24E-03	1.29E-02
wbv-5	14	19	2.10E-02	4.55E-01
wbv-ave	17.6	11.8	7.42E-03	9.97E-02

TABLE 15. Numerical performance of LOQO on NLPs when warmstarting a problem whose problem data have been modified. The iteration counts for the coldstart and warmstart solution of five perturbed problems are given, and the last 2 columns are the scaled Euclidean norm of the distance between the optimal solutions and the Lagrange multipliers of the initial and the perturbed problem. The problems labeled as "ave" refer to the average performance over the five problems.

Problem	δ	#Pert	ColdIters	WarmIters	$\|x^I - x^P\|$	$\|y^I - y^P\|$
biggsc4	0.0001	2.0	29.8	16.6	2.71E-06	5.54E-04
biggsc4	0.001	2.0	29.0	14.6	2.71E-05	1.73E-03
biggsc4	0.01	2.0	31.0	12.0	2.71E-04	5.77E-03
hatfldh	0.0001	2.0	25.0	16.6	2.70E-06	5.46E-04
hatfldh	0.001	2.0	25.0	14.6	2.71E-05	1.73E-03
hatfldh	0.01	2.0	25.0	12.0	2.71E-04	5.77E-03
hs035	0.0001	1.0	9.0	5.0	1.03E-06	1.29E-03
hs035	0.001	1.0	9.0	4.4	1.03E-05	4.08E-03
hs035	0.01	1.0	9.0	5.6	1.03E-04	1.29E-02
hs044	0.0001	2.4	15.0	3.4	2.14E-08	1.22E-04
hs044	0.001	2.4	15.0	4.2	2.68E-08	2.46E-04
hs044	0.01	2.4	15.0	5.0	3.78E-08	2.00E-03
hs076	0.0001	1.0	10.0	4.0	4.43E-06	2.49E-03
hs076	0.001	1.0	10.0	4.2	4.43E-05	7.86E-03
hs076	0.01	1.0	10.0	4.6	4.44E-04	2.49E-02
hs118	0.0001	4.2	15.0	3.6	1.23E-08	1.89E-04
hs118	0.001	4.2	15.0	4.4	9.29E-09	2.96E-04
hs118	0.01	4.2	15.0	5.8	7.16E-09	1.83E-03
hs44new	0.0001	1.4	19.0	4.0	1.30E-08	9.57E-05
hs44new	0.001	1.4	19.0	4.8	1.98E-08	2.44E-04
hs44new	0.01	1.4	19.0	5.8	2.16E-08	2.22E-03
lsqfit	0.0001	1.0	8.0	3.8	5.97E-06	2.39E-03
lsqfit	0.001	1.0	8.0	4.0	5.99E-05	7.58E-03
lsqfit	0.01	1.0	8.0	6.0	5.99E-04	2.40E-02
mosarqp1	0.0001	20.8	13.0	5.0	2.52E-04	9.91E-03
mosarqp1	0.001	20.8	13.0	5.0	2.52E-04	9.90E-03
mosarqp1	0.01	20.8	13.0	5.0	3.21E-04	1.11E-02
mosarqp2	0.0001	20.6	16.6	8.0	4.94E-04	3.23E-03
mosarqp2	0.001	20.6	23.2	8.0	5.13E-04	3.30E-03
mosarqp2	0.01	20.6	15.2	8.0	7.66E-04	4.09E-03
powell20	0.0001	21.8	350.0	15.0	2.60E-05	5.65E-03
powell20	0.001	21.8	350.6	15.0	2.59E-04	5.69E-03
powell20	0.01	21.8	362.0	15.0	2.60E-03	9.83E-03
qpcboei1	0.0001	20.4	63.0	89.2	3.54E-06	3.42E+00
qpcboei1	0.001	20.4	63.0	85.8	3.36E-05	3.42E+00
qpcboei1	0.01	20.4	63.0	86.6	3.29E-04	3.42E+00
qpcboei2	0.0001	19.0	101.0	68.6	1.19E-04	4.10E+00
qpcboei2	0.001	19.0	101.0	67.2	1.46E-03	4.01E+00
qpcboei2	0.01	19.0	101.0	81.6	3.92E-03	4.00E+00
qpcstair	0.0001	19.8	189.0	28.0	9.59E-05	2.99E-02
qpcstair	0.001	19.8	189.0	28.0	9.69E-05	3.00E-02
qpcstair	0.01	19.8	189.4	28.0	1.45E-04	3.01E-02
s224	0.0001	1.2	10.0	3.4	6.09E-05	5.62E-04
s224	0.001	1.2	10.0	4.4	6.07E-04	1.78E-03
s224	0.01	1.2	10.0	6.4	6.07E-03	5.91E-03
s392	0.0001	11.2	32.0	13.0	6.00E-05	9.31E-04
s392	0.001	11.2	29.0	13.6	5.99E-04	9.21E-04
s392	0.01	11.2	32.2	14.2	5.98E-03	4.04E-03
zecevic2	0.0001	1.0	11.0	5.0	9.41E-06	1.64E-03
zecevic2	0.001	1.0	11.0	5.0	9.41E-05	5.18E-03
zecevic2	0.01	1.0	11.0	6.2	9.41E-04	1.64E-02

TABLE 16. Numerical performance of LOQO on the convex quadratic problems from the CUTEr 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 2 columns are the scaled Euclidean norm of the distance between the optimal solutions and the Lagrange multipliers of the initial and the perturbed problem.

Problem	δ	#Pert	ColdIters	WarmIters	$\|x^I - x^P\|$	$\|y^I - y^P\|$
biggsc4	0.0001	1.0	27.2	17.0	1.17E-05	1.67E-03
biggsc4	0.001	1.0	24.6	14.8	1.17E-04	5.27E-03
biggsc4	0.01	1.0	23.6	13.0	1.17E-03	1.68E-02
hatfldh	0.0001	1.0	22.2	17.0	1.17E-05	1.67E-03
hatfldh	0.001	1.0	20.2	14.8	1.17E-04	5.27E-03
hatfldh	0.01	1.0	18.2	13.0	1.17E-03	1.68E-02
hs035	0.0001	1.0	9.0	4.8	1.96E-05	5.67E-03
hs035	0.001	1.0	9.0	5.0	1.96E-04	1.79E-02
hs035	0.01	1.0	9.0	5.4	1.96E-03	5.67E-02
hs044	0.0001	1.2	15.0	3.2	1.73E-05	3.19E-03
hs044	0.001	1.2	15.0	3.6	1.73E-04	1.01E-02
hs044	0.01	1.2	14.8	4.4	1.73E-03	3.19E-02
hs076	0.0001	1.0	10.0	4.0	2.69E-05	5.94E-03
hs076	0.001	1.0	10.0	4.0	2.70E-04	1.88E-02
hs076	0.01	1.0	10.0	4.0	2.70E-03	5.95E-02
hs118	0.0001	2.0	15.0	3.0	2.31E-05	8.25E-03
hs118	0.001	2.0	15.0	3.4	2.31E-04	2.61E-02
hs118	0.01	2.0	15.0	5.4	2.31E-03	8.25E-02
hs44new	0.0001	1.0	19.0	3.4	2.69E-05	3.97E-03
hs44new	0.001	1.0	19.0	3.8	2.70E-04	1.26E-02
hs44new	0.01	1.0	18.2	4.6	2.70E-03	3.98E-02
ksip	0.0001	24.4	45.4	41.8	2.12E-03	4.10E-01
ksip	0.001	24.4	46.6	42.8	6.30E-03	5.82E-01
ksip	0.01	24.4	42.6	64.6	1.40E-02	6.55E-01
lsqfit	0.0001	1.0	8.0	4.0	2.02E-05	4.54E-03
lsqfit	0.001	1.0	8.0	4.2	2.02E-04	1.43E-02
lsqfit	0.01	1.0	8.0	4.6	2.02E-03	4.54E-02
mosarqp1	0.0001	20.8	13.0	5.0	2.52E-04	9.91E-03
mosarqp1	0.001	20.8	13.0	5.0	2.53E-04	9.95E-03
mosarqp1	0.01	20.8	13.0	5.0	3.75E-04	1.27E-02
mosarqp2	0.0001	19.2	17.4	8.0	4.93E-04	3.23E-03
mosarqp2	0.001	19.2	31.0	8.0	5.00E-04	3.25E-03
mosarqp2	0.01	19.2	15.4	8.0	5.82E-04	3.52E-03
powell20	0.0001	23.8	351.4	15.4	1.60E-04	5.01E-03
powell20	0.001	23.8	373.0	18.0	1.71E-03	4.73E-03
powell20	0.01	23.8	363.4	23.6	1.74E-02	4.91E-02
qpcboei1	0.0001	19.4	62.4	113.4	1.87E-05	3.42E+00
qpcboei1	0.001	19.4	63.2	111.8	1.87E-04	3.41E+00
qpcboei1	0.01	19.4	62.0	112.8	1.87E-03	3.34E+00
qpcboei2	0.0001	12.6	104.6	153.2	2.42E-05	1.60E+01
qpcboei2	0.001	12.6	341.0	329.4	2.42E-04	1.87E+02
qpcboei2	0.01	12.6	341.8	326.8	2.42E-03	2.17E+03
qpcstair	0.0001	21.0	190.0	28.4	8.73E-05	1.64E-01
qpcstair	0.001	21.0	190.4	28.2	2.33E-04	1.53E-01
qpcstair	0.01	21.0	188.6	28.2	1.83E-03	2.00E-01
s224	0.0001	1.2	10.0	3.0	4.90E-05	5.09E-04
s224	0.001	1.2	10.0	3.6	4.90E-04	1.62E-03
s224	0.01	1.2	10.0	4.8	4.90E-03	5.40E-03
s392	0.0001	2.8	33.2	13.8	2.19E-05	7.89E-03
s392	0.001	2.8	32.6	13.8	2.19E-04	7.96E-03
s392	0.01	2.8	34.8	14.0	2.19E-03	8.88E-03
zecevic2	0.0001	1.0	11.0	5.0	3.34E-05	3.13E-03
zecevic2	0.001	1.0	11.0	5.0	3.34E-04	9.91E-03
zecevic2	0.01	1.0	11.0	5.2	3.34E-03	3.13E-02

TABLE 17. Numerical performance of LOQO on the convex quadratic problems from the CUTEr 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 coefficients. The average iteration counts for the warmstart and coldstart solution of five perturbed problems are given, and the last 2 columns are the scaled Euclidean norm of the distance between the optimal solutions and the Lagrange multipliers of the initial and the perturbed problem.

Problem	δ	#Pert	ColdIters	WarmIters	$\|x^I - x^P\|$	$\|y^I - y^P\|$
biggsc4	0.0001	1.0	27.4	16.4	3.12E-06	5.93E-04
biggsc4	0.001	1.0	25.4	13.8	3.11E-05	1.77E-03
biggsc4	0.01	1.0	23.0	11.2	3.11E-04	6.01E-03
hatfldh	0.0001	1.0	22.6	16.4	3.12E-06	5.89E-04
hatfldh	0.001	1.0	20.0	13.8	3.11E-05	1.78E-03
hatfldh	0.01	1.0	17.4	11.2	3.11E-04	6.01E-03
hs035	0.0001	1.0	9.0	5.0	1.03E-06	1.29E-03
hs035	0.001	1.0	9.0	5.0	1.03E-05	4.08E-03
hs035	0.01	1.0	9.0	5.2	1.03E-04	1.29E-02
hs044	0.0001	1.2	15.0	3.2	2.01E-08	1.20E-04
hs044	0.001	1.2	15.0	3.6	1.66E-08	2.69E-04
hs044	0.01	1.2	15.0	4.2	1.54E-08	2.30E-03
hs076	0.0001	1.0	10.0	4.0	3.46E-06	2.12E-03
hs076	0.001	1.0	10.0	4.2	3.45E-05	6.68E-03
hs076	0.01	1.0	10.0	4.4	3.45E-04	2.11E-02
hs118	0.0001	1.2	15.0	3.0	1.48E-09	7.80E-05
hs118	0.001	1.2	15.0	3.0	1.48E-09	2.63E-04
hs118	0.01	1.2	15.0	4.2	3.80E-09	2.50E-03
hs21mod	0.0001	1.2	18.8	16.8	4.03E-01	1.24E+00
hs21mod	0.001	1.2	18.8	16.8	4.03E-01	1.24E+00
hs21mod	0.01	1.2	18.6	16.8	4.03E-01	1.24E+00
hs44new	0.0001	1.2	19.0	3.2	2.06E-08	1.22E-04
hs44new	0.001	1.2	19.0	3.6	1.73E-08	2.70E-04
hs44new	0.01	1.2	18.2	4.2	1.68E-08	2.30E-03
ksip	0.0001	2.2	33.0	45.4	1.48E+00	1.28E+00
ksip	0.001	2.2	30.2	40.0	1.48E+00	1.28E+00
ksip	0.01	2.2	29.6	37.8	1.48E+00	1.28E+00
lsqfit	0.0001	1.0	8.0	4.0	7.58E-06	2.81E-03
lsqfit	0.001	1.0	8.0	4.2	7.57E-05	8.88E-03
lsqfit	0.01	1.0	8.0	4.6	7.57E-04	2.81E-02
powell20	0.0001	20.0	352.4	15.0	3.58E-04	6.30E-03
powell20	0.001	20.0	349.4	15.0	3.57E-04	6.30E-03
powell20	0.01	20.0	354.4	15.0	3.53E-04	6.30E-03
s224	0.0001	1.0	10.0	3.0	5.46E-05	5.44E-04
s224	0.001	1.0	10.0	3.4	5.47E-04	1.74E-03
s224	0.01	1.0	10.0	4.0	5.47E-03	6.17E-03
s268	0.0001	1.0	13.8	10.0	4.85E-01	1.78E+00
s268	0.001	1.0	12.8	10.0	4.85E-01	1.78E+00
s268	0.01	1.0	11.6	10.0	4.85E-01	1.78E+00
s392	0.0001	2.4	32.0	13.0	3.31E-05	9.42E-04
s392	0.001	2.4	32.4	13.0	3.31E-04	1.01E-03
s392	0.01	2.4	41.8	13.0	3.31E-03	2.17E-03
zecevic2	0.0001	1.0	11.0	5.0	1.23E-05	1.83E-03
zecevic2	0.001	1.0	11.0	5.0	1.23E-04	5.79E-03
zecevic2	0.01	1.0	11.0	5.4	1.23E-03	1.85E-02

TABLE 18. Numerical performance of LOQO on the convex quadratic problems from the CUTEr 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 2 columns are the scaled Euclidean norm of the distance between the optimal solutions and the Lagrange multipliers of the initial and the perturbed problem.

Problem	δ	#Pert	ColdIters	WarmIters	$\|x^I - x^P\|$	$\|y^I - y^P\|$
dtoc1l	0.0001	20.8	6.0	2.8	9.33E-07	3.69E-04
dtoc1l	0.001	20.8	6.0	3.2	2.03E-06	5.39E-04
dtoc1l	0.01	20.8	6.0	4.0	1.79E-05	1.59E-03
expfita	0.0001	7.0	18.8	13.0	1.00E-05	1.23E-02
expfita	0.001	7.0	18.8	13.0	4.94E-05	2.39E-02
expfita	0.01	7.0	18.6	13.8	4.50E-04	6.89E-02
expfitb	0.0001	20.2	22.8	19.0	2.18E-05	1.37E-02
expfitb	0.001	20.2	24.4	19.0	1.08E-04	3.98E-02
expfitb	0.01	20.2	22.8	22.2	1.26E-03	1.36E-01
expfitc	0.0001	23.4	32.2	30.0	8.12E-05	3.24E-02
expfitc	0.001	23.4	31.0	29.8	8.92E-05	3.11E-02
expfitc	0.01	23.4	32.4	35.4	5.71E-04	7.95E-02
himmelbi	0.0001	11.4	22.0	7.6	2.75E-01	7.22E+01
himmelbi	0.001	11.4	22.0	8.2	3.53E-01	8.82E+01
himmelbi	0.01	11.4	22.0	11.2	2.74E-01	7.18E+01
himmelbj	0.0001	9.4	70.0	8.4	7.74E-03	2.41E-01
himmelbj	0.001	9.4	70.0	12.0	6.45E-02	3.21E-01
himmelbj	0.01	9.4	70.0	27.8	4.02E-01	7.30E-01
hong	0.0001	1.0	18.0	37.0	2.09E-07	6.41E-02
hong	0.001	1.0	18.0	37.0	2.08E-07	6.41E-02
hong	0.01	1.0	17.8	37.4	2.03E-07	1.04E-01
hs009	0.0001	1.0	11.0	11.8	9.14E-08	1.07E+00
hs009	0.001	1.0	18.2	13.6	9.14E-08	1.07E+00
hs009	0.01	1.0	23.6	17.2	9.13E-08	1.07E+00
hs024	0.0001	1.0	12.0	4.0	4.19E-09	6.58E-05
hs024	0.001	1.0	12.0	4.4	1.11E-08	1.21E-04
hs024	0.01	1.0	12.0	5.8	4.66E-09	6.68E-04
hs036	0.0001	1.0	13.0	4.0	2.69E-08	1.21E-05
hs036	0.001	1.0	13.2	4.2	2.45E-08	7.08E-05
hs036	0.01	1.0	13.4	6.2	2.63E-08	6.64E-04
hs037	0.0001	1.0	10.0	3.0	1.28E-05	1.30E-04
hs037	0.001	1.0	10.0	4.0	1.28E-04	4.23E-04
hs037	0.01	1.0	10.0	6.4	1.28E-03	1.63E-03
hs041	0.0001	1.0	8.0	5.2	3.09E-06	2.66E-03
hs041	0.001	1.0	8.0	5.0	2.57E-05	7.07E-03
hs041	0.01	1.0	8.0	5.0	2.52E-04	2.13E-02
hs049	0.0001	1.0	16.0	1.8	4.12E-03	1.15E-01
hs049	0.001	1.0	16.0	2.2	4.71E-03	1.24E-01
hs049	0.01	1.0	16.0	3.2	5.81E-03	1.37E-01
hs050	0.0001	1.0	14.0	2.0	4.86E-07	1.19E-03
hs050	0.001	1.0	14.0	3.0	2.62E-07	8.82E-04
hs050	0.01	1.0	14.0	3.6	4.58E-07	1.13E-03
hs054	0.0001	1.0	10.0	2.6	7.52E-06	1.64E-03
hs054	0.001	1.0	10.0	3.2	7.52E-05	5.20E-03
hs054	0.01	1.0	10.0	3.8	7.51E-04	1.65E-02
hs055	0.0001	1.2	9.0	12.0	7.24E-08	1.42E-01
hs055	0.001	1.2	9.0	12.4	5.85E-08	2.09E-01
hs055	0.01	1.2	9.0	12.6	4.97E-08	2.85E-01
hs062	0.0001	1.0	12.0	14.8	8.22E-06	9.95E-06
hs062	0.001	1.0	12.0	14.0	8.24E-05	9.94E-05
hs062	0.01	1.0	12.0	15.4	8.30E-04	9.94E-04
hs086	0.0001	1.8	13.0	7.0	1.41E-06	1.13E-04
hs086	0.001	1.8	13.0	7.0	1.14E-06	1.95E-04
hs086	0.01	1.8	13.0	7.4	1.83E-06	1.47E-03
hs112	0.0001	1.4	15.0	11.0	3.89E-05	3.66E-04
hs112	0.001	1.4	15.0	11.0	3.98E-04	1.19E-03
hs112	0.01	1.4	15.0	9.2	4.09E-03	4.34E-03
hs119	0.0001	5.4	21.0	21.0	9.29E-06	4.83E-05
hs119	0.001	5.4	21.0	21.0	9.28E-05	1.77E-04
hs119	0.01	5.4	21.0	21.0	9.27E-04	1.05E-03
hubfit	0.0001	1.0	9.0	4.0	6.01E-06	2.65E-03
hubfit	0.001	1.0	9.0	4.2	6.00E-05	8.40E-03
hubfit	0.01	1.0	9.0	6.0	5.99E-04	2.66E-02
water	0.0001	5.2	23.0	21.4	3.44E-02	4.22E+00
water	0.001	5.2	23.0	22.0	2.62E-01	3.18E+00
water	0.01	5.2	22.8	33.6	7.98E-01	2.90E+00

TABLE 19. Numerical performance of LOQO on NLPs from the CUTEr test suite when warmstarting a problem whose constraint coefficients, A, have been modified. All problems considered have linear constraints. 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 2 columns are the scaled Euclidean norm of the distance between the optimal solutions and the Lagrange multipliers of the initial and the perturbed problem.

Problem	δ	#Pert	ColdIters	WarmIters	$\|x^I - x^P\|$	$\|y^I - y^P\|$
dtoc1l	0.0001	18.6	6.0	2.0	8.16E-06	1.09E-03
dtoc1l	0.001	18.6	6.0	3.0	8.14E-05	3.45E-03
dtoc1l	0.01	18.6	6.0	4.0	8.14E-04	1.11E-02
expfita	0.0001	2.2	19.0	13.0	2.27E-04	6.00E-02
expfita	0.001	2.2	19.0	13.6	2.32E-03	1.92E-01
expfita	0.01	2.2	18.4	14.2	1.44E-02	4.79E-01
expfitb	0.0001	10.2	21.8	19.0	7.16E-05	3.07E-02
expfitb	0.001	10.2	25.2	19.4	8.10E-04	1.01E-01
expfitb	0.01	10.2	24.6	24.6	9.75E-03	3.55E-01
expfitc	0.0001	23.0	30.8	32.4	5.38E-05	2.60E-02
expfitc	0.001	23.0	34.2	33.2	3.29E-04	6.53E-02
expfitc	0.01	23.0	38.0	35.2	7.02E-03	2.94E-01
himmelbi	0.0001	1.2	22.0	1.2	3.17E-01	8.37E+01
himmelbi	0.001	1.2	22.0	9.0	3.16E-01	8.34E+01
himmelbi	0.01	1.2	22.0	8.0	3.30E-01	8.53E+01
himmelbj	0.0001	1.8	65.8	8.2	1.09E-03	2.13E-01
himmelbj	0.001	1.8	65.8	13.6	2.30E-03	2.08E-01
himmelbj	0.01	1.8	60.2	26.6	2.86E-03	3.10E-01
hong	0.0001	1.0	23.2	40.0	1.23E-05	3.62E-01
hong	0.001	1.0	21.6	40.6	1.24E-04	3.63E-01
hong	0.01	1.0	19.6	38.8	1.25E-03	3.73E-01
hs009	0.0001	1.0	5.0	2.0	9.14E-08	1.07E+00
hs009	0.001	1.0	5.0	3.0	9.13E-08	1.07E+00
hs009	0.01	1.0	5.0	4.4	9.14E-08	1.07E+00
hs024	0.0001	1.0	12.0	4.0	2.99E-05	5.07E-03
hs024	0.001	1.0	12.0	4.8	2.99E-04	1.60E-02
hs024	0.01	1.0	12.0	5.4	2.99E-03	5.09E-02
hs036	0.0001	1.0	13.0	4.0	3.17E-05	2.61E-04
hs036	0.001	1.0	13.8	4.2	3.17E-04	8.26E-04
hs036	0.01	1.0	15.0	5.0	3.17E-03	2.61E-03
hs037	0.0001	1.0	10.0	3.0	2.41E-05	1.88E-04
hs037	0.001	1.0	10.0	3.2	2.41E-04	7.61E-04
hs037	0.01	1.0	10.0	5.0	2.41E-03	5.28E-03
hs041	0.0001	1.0	8.0	5.0	3.79E-06	3.08E-03
hs041	0.001	1.0	8.0	5.0	3.26E-05	8.88E-03
hs041	0.01	1.0	8.0	5.0	3.22E-04	2.78E-02
hs049	0.0001	1.0	16.0	1.8	4.08E-03	1.15E-01
hs049	0.001	1.0	16.0	2.8	5.17E-03	1.30E-01
hs049	0.01	1.0	16.0	3.8	4.58E-03	1.18E-01
hs050	0.0001	1.0	14.0	1.2	1.90E-05	7.36E-03
hs050	0.001	1.0	14.0	2.2	1.95E-04	2.37E-02
hs050	0.01	1.0	14.0	3.4	1.95E-03	7.57E-02
hs054	0.0001	1.0	10.0	1.0	6.84E-06	1.62E-03
hs054	0.001	1.0	10.0	2.0	7.18E-05	5.24E-03
hs054	0.01	1.0	10.0	3.0	7.20E-04	1.66E-02
hs062	0.0001	1.0	12.0	15.0	1.16E-05	2.62E-05
hs062	0.001	1.0	12.0	15.0	1.16E-04	2.62E-04
hs062	0.01	1.0	12.0	15.0	1.16E-03	2.62E-03
hs086	0.0001	1.0	13.0	7.0	8.32E-06	2.63E-04
hs086	0.001	1.0	13.0	7.0	7.95E-05	8.19E-04
hs086	0.01	1.0	13.0	7.2	7.92E-04	2.84E-03
hs112	0.0001	1.0	15.0	11.0	9.37E-06	1.79E-04
hs112	0.001	1.0	15.0	11.0	8.62E-05	5.35E-04
hs112	0.01	1.0	15.0	9.6	8.55E-04	1.70E-03
hs119	0.0001	1.2	21.0	21.0	1.92E-05	9.06E-05
hs119	0.001	1.2	21.0	21.0	1.92E-04	5.97E-04
hs119	0.01	1.2	20.4	21.0	1.92E-03	5.46E-03
hubfit	0.0001	1.0	9.0	4.0	2.00E-05	5.00E-03
hubfit	0.001	1.0	9.0	4.4	2.02E-04	1.59E-02
hubfit	0.01	1.0	9.0	4.8	2.02E-03	5.03E-02
lin	0.0001	1.0	6.0	2.0	1.12E-05	3.58E-03
lin	0.001	1.0	6.2	2.8	1.14E-04	1.15E-02
lin	0.01	1.0	6.4	3.8	1.14E-03	3.63E-02

TABLE 20. Numerical performance of LOQO on NLPs from the CUTEr test suite when warmstarting a problem whose constraint right-hand sides, b, have been modified. All problems considered have linear constraints. 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 2 columns are the scaled Euclidean norm of the distance between the optimal solutions and the Lagrange multipliers of the initial and the perturbed problem.

Problem	δ	#Pert	ColdIters	WarmIters	$\|x^I - x^P\|$	$\|y^I - y^P\|$
bt10	0.0001	1.0	9.0	2.0	9.53E-08	2.13E-04
bt10	0.001	1.0	9.0	2.0	9.54E-08	6.57E-04
bt10	0.01	1.0	9.0	2.6	8.21E-08	5.97E-03
byrdsphr	0.0001	1.0	55.0	1.0	8.85E-06	2.96E-03
byrdsphr	0.001	1.0	106.2	2.0	9.27E-05	9.56E-03
byrdsphr	0.01	1.0	103.0	2.8	9.30E-04	3.04E-02
cantilvr	0.0001	1.0	15.0	5.0	3.22E-06	3.94E-03
cantilvr	0.001	1.0	15.0	5.0	3.22E-05	1.24E-02
cantilvr	0.01	1.0	15.0	5.0	3.23E-04	3.94E-02
cb2	0.0001	1.0	14.0	4.0	1.35E-06	9.89E-04
cb2	0.001	1.0	13.6	4.0	1.34E-05	3.04E-03
cb2	0.01	1.0	11.8	4.0	1.34E-04	9.71E-03
cb3	0.0001	1.0	11.0	5.0	6.73E-09	9.42E-05
cb3	0.001	1.0	11.0	5.0	6.73E-09	1.19E-04
cb3	0.01	1.0	11.0	5.0	6.73E-09	6.72E-04
chaconn1	0.0001	1.0	9.0	4.0	1.36E-06	1.01E-03
chaconn1	0.001	1.0	9.0	4.0	1.35E-05	3.05E-03
chaconn1	0.01	1.0	9.0	4.0	1.34E-04	9.72E-03
chaconn2	0.0001	1.0	10.0	5.0	1.24E-08	1.28E-04
chaconn2	0.001	1.0	10.0	5.0	1.24E-08	1.48E-04
chaconn2	0.01	1.0	10.0	5.0	1.24E-08	6.80E-04
congigmz	0.0001	1.0	27.0	11.0	7.35E-06	1.64E-03
congigmz	0.001	1.0	27.0	11.0	9.18E-06	1.83E-03
congigmz	0.01	1.0	27.0	10.8	1.63E-05	2.86E-03
csfi1	0.0001	1.0	40.4	18.2	3.93E-02	7.13E-02
csfi1	0.001	1.0	43.6	16.8	3.94E-02	7.13E-02
csfi1	0.01	1.0	41.0	14.8	3.94E-02	7.14E-02
csfi2	0.0001	1.0	82.0	15.2	2.76E-02	4.95E-02
csfi2	0.001	1.0	81.2	14.6	2.76E-02	4.95E-02
csfi2	0.01	1.0	75.4	14.4	2.76E-02	4.95E-02
demymalo	0.0001	1.0	24.0	4.0	1.33E-08	1.68E-04
demymalo	0.001	1.0	24.0	4.0	1.33E-08	1.86E-04
demymalo	0.01	1.0	23.6	4.0	1.33E-08	6.87E-04
disc2	0.0001	3.2	44.4	19.0	1.53E-01	1.41E+00
disc2	0.001	3.2	52.2	17.4	1.53E-01	1.41E+00
disc2	0.01	3.2	42.0	29.0	1.38E-01	1.31E+00
eigmaxb	0.0001	9.2	12.0	3.0	2.12E-08	4.09E-04
eigmaxb	0.001	9.2	12.0	3.0	2.25E-08	3.93E-03
eigmaxb	0.01	9.2	12.4	3.6	3.01E-08	3.92E-02
eigmaxc	0.0001	9.2	101.0	21.0	3.75E-09	3.89E-05
eigmaxc	0.001	9.2	101.8	20.8	3.98E-09	9.45E-05
eigmaxc	0.01	9.2	103.6	20.6	4.94E-09	7.19E-04
eigminb	0.0001	9.2	9.0	3.0	1.56E-07	2.26E-03
eigminb	0.001	9.2	9.0	3.0	1.58E-07	2.22E-02
eigminb	0.01	9.2	9.0	3.8	4.24E-07	2.22E-01
eigminc	0.0001	9.2	13.0	4.0	9.24E-10	2.36E-05
eigminc	0.001	9.2	13.0	4.0	9.23E-10	7.84E-05
eigminc	0.01	9.2	13.0	4.0	9.19E-10	7.13E-04
gigomez1	0.0001	1.0	19.0	4.0	5.19E-08	3.31E-04
gigomez1	0.001	1.0	19.0	4.0	5.19E-08	3.41E-04
gigomez1	0.01	1.0	19.0	4.0	5.19E-08	7.69E-04
gigomez2	0.0001	1.0	11.0	4.0	1.36E-06	1.03E-03
gigomez2	0.001	1.0	11.0	4.0	1.34E-05	3.07E-03
gigomez2	0.01	1.0	11.0	4.0	1.34E-04	9.72E-03
gigomez3	0.0001	1.0	11.0	5.0	3.39E-08	2.11E-04
gigomez3	0.001	1.0	11.0	5.0	3.39E-08	2.24E-04
gigomez3	0.01	1.0	11.0	5.0	3.39E-08	7.08E-04
hadamard	0.0001	6.0	8.0	5.0	1.00E-08	1.49E+01
hadamard	0.001	6.0	8.0	5.0	1.01E-08	1.49E+01
hadamard	0.01	6.0	8.0	5.0	1.13E-08	1.49E+01
het	z-0.0001	1.0	44.2	37.0	3.97E-04	7.31E-02
het	z-0.001	1.0	40.0	35.6	4.00E-04	7.06E-02
het	z-0.01	1.0	38.0	40.8	4.00E-04	1.95E-01

TABLE 21. Numerical performance of LOQO on NLPs from the CUTEr test suite when warmstarting a problem whose objective function coefficients, c, have been modified. All problems considered have linear objective functions. 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 2 columns are the scaled Euclidean norm of the distance between the optimal solutions and the Lagrange multipliers of the initial and the perturbed problem.

Problem	δ	#Pert	ColdIters	WarmIters	$\|x^I - x^P\|$	$\|y^I - y^P\|$
hs010	0.0001	1.0	49.4	4.0	1.71E-05	3.66E-03
hs010	0.001	1.0	51.2	4.0	1.71E-04	1.16E-02
hs010	0.01	1.0	48.4	4.0	1.70E-03	3.66E-02
hs034	0.0001	1.0	14.0	11.0	7.96E-10	6.69E-05
hs034	0.001	1.0	14.0	11.0	7.97E-10	1.16E-04
hs034	0.01	1.0	14.0	11.0	8.03E-10	9.03E-04
hs039	0.0001	1.0	16.6	8.0	2.17E-06	7.41E-04
hs039	0.001	1.0	19.4	8.0	2.16E-05	2.30E-03
hs039	0.01	1.0	37.4	8.0	2.16E-04	7.76E-03
hs066	0.0001	1.0	13.0	17.0	1.49E-05	4.48E-03
hs066	0.001	1.0	13.0	17.0	1.49E-04	1.42E-02
hs066	0.01	1.0	13.0	16.8	1.49E-03	4.48E-02
hs072	0.0001	1.0	21.0	20.0	5.32E-06	8.47E-06
hs072	0.001	1.0	21.0	20.0	5.32E-05	8.22E-05
hs072	0.01	1.0	21.0	19.4	5.33E-04	8.19E-04
hs073	0.0001	1.0	16.0	14.0	9.76E-07	7.70E-05
hs073	0.001	1.0	16.0	14.0	7.56E-07	3.23E-04
hs073	0.01	1.0	16.4	14.0	5.86E-07	3.08E-03
hs106	0.0001	1.2	69.8	9.0	8.64E-06	2.96E-05
hs106	0.001	1.2	71.4	9.0	8.71E-05	1.79E-04
hs106	0.01	1.2	65.8	9.0	8.73E-04	1.60E-03
madsen	0.0001	1.0	20.8	5.0	5.87E-06	1.39E-03
madsen	0.001	1.0	20.0	5.0	5.83E-05	4.17E-03
madsen	0.01	1.0	18.4	5.0	5.84E-04	1.35E-02
madsschj	0.0001	8.2	73.4	12.0	4.62E-07	1.68E-02
madsschj	0.001	8.2	72.2	12.0	3.58E-06	4.40E-02
madsschj	0.01	8.2	72.4	12.0	3.55E-05	1.38E-01
makela1	0.0001	1.0	18.2	4.0	4.01E-06	1.47E-03
makela1	0.001	1.0	20.6	4.0	3.99E-05	4.48E-03
makela1	0.01	1.0	24.0	4.0	3.98E-04	1.42E-02
makela2	0.0001	1.0	18.0	6.0	8.10E-07	1.22E-03
makela2	0.001	1.0	18.0	6.0	8.03E-06	3.64E-03
makela2	0.01	1.0	18.0	6.0	8.02E-05	1.15E-02
mifflin1	0.0001	1.0	60.8	4.0	5.17E-06	1.02E-03
mifflin1	0.001	1.0	49.2	4.0	5.16E-05	3.04E-03
mifflin1	0.01	1.0	59.6	4.0	5.16E-04	9.92E-03
mifflin2	0.0001	1.0	58.8	5.0	1.37E-08	9.46E-05
mifflin2	0.001	1.0	49.6	5.0	1.37E-08	1.31E-04
mifflin2	0.01	1.0	50.6	5.0	1.37E-08	8.72E-04
minmaxrb	0.0001	1.0	9.0	7.0	5.24E-09	4.68E-01
minmaxrb	0.001	1.0	9.0	7.0	5.24E-09	4.68E-01
minmaxrb	0.01	1.0	9.0	7.0	5.28E-09	4.67E-01
polak1	0.0001	1.0	37.6	4.0	1.39E-04	6.13E-03
polak1	0.001	1.0	25.4	4.0	1.39E-03	1.90E-02
polak1	0.01	1.0	29.0	4.0	1.39E-02	6.01E-02
polak4	0.0001	1.0	9.0	8.0	4.16E-06	3.00E-01
polak4	0.001	1.0	12.0	8.8	1.62E-05	3.80E-01
polak4	0.01	1.0	12.2	9.2	1.56E-04	3.81E-01
polak6	0.0001	1.0	20.0	5.0	1.28E-07	1.35E-03
polak6	0.001	1.0	20.0	5.0	1.10E-06	3.72E-03
polak6	0.01	1.0	20.0	5.0	1.09E-05	1.15E-02
prodpl0	0.0001	7.0	17.0	16.0	1.34E-06	1.36E-04
prodpl0	0.001	7.0	17.0	16.0	1.36E-06	2.02E-04
prodpl0	0.01	7.0	17.0	16.2	1.25E-06	1.38E-03
prodpl1	0.0001	6.6	17.0	16.0	2.15E-07	1.03E-04
prodpl1	0.001	6.6	17.0	16.0	2.16E-07	1.84E-04
prodpl1	0.01	6.6	17.0	16.0	2.22E-07	1.47E-03
rosenmmx	0.0001	1.0	44.6	5.0	1.23E-07	1.30E-03
rosenmmx	0.001	1.0	44.6	5.0	1.10E-06	3.69E-03
rosenmmx	0.01	1.0	43.4	5.0	1.09E-05	1.15E-02
smmpsf	0.0001	20.4	124.8	113.0	5.71E-06	3.14E+00
smmpsf	0.001	20.4	129.8	124.4	3.89E-06	2.93E+00
smmpsf	0.01	20.4	113.0	94.2	5.48E-06	3.29E+00

TABLE 22. Numerical performance of LOQO on NLPs from the CUTEr test suite when warmstarting a problem whose objective function coefficients, c, have been modified. All problems considered have linear objective functions. 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 2 columns are the scaled Euclidean norm of the distance between the optimal solutions and the Lagrange multipliers of the initial and the perturbed problem.

Problem	δ	#Pert	ColdIters	WarmIters	$\|x^I - x^P\|$	$\|y^I - y^P\|$
spiral	0.0001	1.0	45.2	12.0	1.69E+02	2.76E+02
spiral	0.001	1.0	26.8	12.0	1.69E+02	2.76E+02
spiral	0.01	1.0	102.2	12.0	1.69E+02	2.76E+02
swopf	0.0001	8.4	18.0	13.0	5.49E-04	1.25E-02
swopf	0.001	8.4	19.4	13.6	5.50E-03	3.97E-02
swopf	0.01	8.4	17.8	14.0	7.62E-02	1.46E-01
tenbars1	0.0001	2.0	21.0	6.0	4.42E-06	8.58E-04
tenbars1	0.001	2.0	21.0	6.0	4.41E-05	2.71E-03
tenbars1	0.01	2.0	21.0	6.4	4.41E-04	8.66E-03
tenbars2	0.0001	2.0	19.0	6.0	4.41E-06	8.58E-04
tenbars2	0.001	2.0	19.0	6.0	4.41E-05	2.71E-03
tenbars2	0.01	2.0	19.0	6.0	4.41E-04	8.66E-03
tenbars3	0.0001	2.0	20.0	8.0	4.54E-06	6.11E-03
tenbars3	0.001	2.0	20.0	8.0	4.54E-05	1.93E-02
tenbars3	0.01	2.0	20.0	8.0	4.53E-04	6.12E-02
tenbars4	0.0001	1.6	15.0	5.0	3.39E-06	9.08E-04
tenbars4	0.001	1.6	15.0	5.0	3.17E-05	2.66E-03
tenbars4	0.01	1.6	15.0	5.0	3.16E-04	8.42E-03
truspyr1	0.0001	1.6	15.8	7.8	8.41E-02	2.52E-02
truspyr1	0.001	1.6	15.4	7.8	9.29E-02	2.85E-02
truspyr1	0.01	1.6	15.4	7.2	8.57E-02	2.59E-02
truspyr2	0.0001	1.6	15.8	7.8	3.02E-01	2.86E-01
truspyr2	0.001	1.6	15.8	7.4	3.02E-01	2.82E-01
truspyr2	0.01	1.6	16.4	7.6	3.02E-01	2.85E-01
ubh5	0.0001	17.8	7.0	2.0	6.42E-02	1.49E+00
ubh5	0.001	17.8	7.0	3.6	6.44E-01	4.71E+00
ubh5	0.01	17.8	8.0	5.2	6.45E+00	1.49E+01
womflet	0.0001	1.0	24.0	5.0	2.24E-06	1.67E-03
womflet	0.001	1.0	24.2	5.0	2.18E-05	4.92E-03
womflet	0.01	1.0	25.2	5.0	2.18E-04	1.54E-02

TABLE 23. Numerical performance of LOQO on NLPs from the CUTEr test suite when warmstarting a problem whose objective function coefficients, c, have been modified. All problems considered have linear objective functions. 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 2 columns are the scaled Euclidean norm of the distance between the optimal solutions and the Lagrange multipliers of the initial and the perturbed problem.

HANDE Y. BENSON, DREXEL UNIVERSITY, PHILADELPHIA, PA

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