

INTERIOR-POINT METHODS FOR NONCONVEX NONLINEAR PROGRAMMING: CONVERGENCE ANALYSIS AND COMPUTATIONAL PERFORMANCE

HANDE Y. BENSON, ARUN SEN, AND DAVID F. SHANNO

ABSTRACT. In this paper, we present global and local convergence results for an interior-point method for nonlinear programming and analyze the computational performance of its implementation. The algorithm uses an ℓ_1 penalty approach to relax all constraints, to provide regularization, and to bound the Lagrange multipliers. The penalty problems are solved using a simplified version of Chen and Goldfarb's strictly feasible interior-point method [12]. The global convergence of the algorithm is proved under mild assumptions, and local analysis shows that it converges Q-quadratically for a large class of problems. The proposed approach is the first to simultaneously have all of the following properties while solving a general nonconvex nonlinear programming problem: (1) the convergence analysis does not assume boundedness of dual iterates, (2) local convergence does not require the Linear Independence Constraint Qualification, (3) the solution of the penalty problem is shown to locally converge to optima that may not satisfy the Karush-Kuhn-Tucker conditions, and (4) the algorithm is applicable to mathematical programs with equilibrium constraints. Numerical testing on a set of general nonlinear programming problems, including degenerate problems and infeasible problems, confirm the theoretical results. We also provide comparisons to a highly-efficient nonlinear solver and thoroughly analyze the effects of enforcing theoretical convergence guarantees on the computational performance of the algorithm.

1. INTRODUCTION

Computational studies including [7], [8], and [26] show that there are a number of good nonlinear programming (NLP) codes that work well and solve most problems in standard test sets such as CUTEr [14]. However, codes like IPOPT [35], KNITRO [27], LOQO [33], MINOS [25], and SNOPT [15] do sometimes fail, and there is a great deal of computational work focused on characterizing and overcoming these failures. Recent work includes the solution of mathematical programs with equilibrium constraints (MPECs), especially using interior-point solvers, as documented in [1], [3], [21], and [28]. There is also work on general NLPs that do not satisfy certain constraint qualifications at the optimal solution, and work on such problems includes [24] and [34]. Identification of infeasible problems has also been a persistent and difficult problem in NLP, as discussed in [3] and [15]. Many of these papers have dealt with modifications to a standard algorithm to alleviate the

Date: June 25, 2009.

Key words and phrases. interior-point methods, nonlinear programming, mathematical programs with equilibrium constraints.

Research of the first author is sponsored by ONR grant N00014-04-1-0145 and NSF grant CCF-0725692. Research of the second author is supported by NSF grant DMS-9870317 and ONR grant N00014-98-1-0036. Research of the third author is supported by NSF grant DMS-0107450.

problems. As documented in our work [3] and [7], and those of others, including [1] and [16], one technique that has proved particularly effective on all the issues raised above is constraint relaxation. Much of the work in this area has been to describe modifications to the basic algorithm and some also include numerical results.

Separate from pure algorithmic studies has been a long history of convergence proofs often for purely theoretical algorithms. The purpose of this paper is to provide a convergence proof for an algorithm that is as close to efficient implemented algorithms as possible. The proof contains many features previously explored in proofs of alternate algorithms, yet to our knowledge, it is the first to simultaneously have all of the following properties while solving a general nonconvex NLP: (1) the convergence analysis does not assume boundedness of dual iterates, (2) local convergence does not require the Linear Independence Constraint Qualification, (3) the solution of the penalty problem is shown to locally converge to optima that may not satisfy the Karush-Kuhn-Tucker conditions, and (4) the algorithm is applicable to MPECs. Therefore, we present a more complete approach than the results of [23], [29], [2], and [16], all of which use constraint relaxation as well. In [23], Mayne and Polak present a penalty approach for handling problems with equality and inequality constraints, however, they only relax the equality constraints and only on one side. Doing so requires them to assume that the Linear Independence Constraint Qualification holds for the inequality constraints and that the dual iterates remain bounded. Similarly, in [29], Tits et.al. only relax the equality constraints and have a regularity assumption on the inequality constraints. In [2], Armand presents a quasi-Newton method for handling convex NLPs with only inequality constraints. Due to the assumption of convexity, a brief mention of including linear equality constraints is made, but no results are proven. [16] is perhaps the paper most closely related to our work, but provides no discussion of solving MPECs. In addition, none of the papers provide as detailed an analysis of convergence to optima that do not satisfy Karush-Kuhn-Tucker conditions.

As stated, the convergence proof provided is for an algorithm that closely approximates existing efficient codes. In fact, we have implemented this algorithm and report numerical results on a standard test suite to show that it does solve some particularly difficult problems on which standard algorithms fail. However, to strictly implement a theoretically convergent algorithm can at times greatly reduce the numerical efficiency of the resulting code. This paper will analyze the reasons for this lack of efficiency. The paper will also examine various types of particularly difficult problems and show what conditions can lead to poor performance or failure without constraint relaxation and what conditions seem not to adversely affect practical performance. Test results will be given in comparison with an algorithm that does not use constraint relaxation and has not been proved convergent. In those cases where the convergent algorithm is clearly superior, we suggest a hybrid approach that can inherit the best features of both algorithms in terms of both robustness and efficiency.

In the next section, we propose an interior-point method that uses an ℓ_1 penalty approach. This approach will relax all the constraints, which makes it trivial to identify a feasible solution. Therefore, an assumption of a nonempty feasible region will not be necessary, and moreover, a certificate of infeasibility can be determined even when using this quasi-feasible approach. It also ensures that the dual iterates remain bounded, removing the need for the assumption of the boundedness of the

Hessian of the Lagrangian in [12]. In Section 3, we will start by showing various characteristics of the algorithm that will be useful for developing a convergence proof. With these results in hand, we will provide a global convergence proof using results from [12], showing that our algorithm will either converge to a first-order point or an infeasible point that minimizes an ℓ_1 measure of infeasibility. In Section 4, we will show that locally, our algorithm has a Q-quadratic rate of convergence under mild conditions and without resorting to the traditional Linear Independence Constraint Qualification (LICQ) or the weaker Mangasarian-Fromowitz Constraint Qualification (MFCQ). In Section 5, we discuss further properties of some problems which do not satisfy all of the assumptions of the proposed algorithm, including a result that shows that the KKT solutions of the penalty problem locally converge to optima for the original problem, even if the said optima do not satisfy the KKT conditions. Finally, in Section 6, we present extensive numerical results on the Hock and Schittkowski test suite [18], degenerate or infeasible problems from [24], and several other test problems that appear in literature. We provide a comparison to LOQO and thoroughly analyze any strengths in robustness and weaknesses in efficiency exhibited by the convergent code. The proposed reformulation and algorithm will be applicable to Mathematical Programs with Equilibrium Constraints (MPECs), as well. In short, this approach will strengthen the assumptions mentioned in literature and share characteristics with efficiently-implemented interior-point methods.

2. A PENALTY INTERIOR-POINT METHOD FOR NONCONVEX NLP

One of the goals of this paper is to use a penalty method approach in order to propose an interior-point algorithm whose convergence results use rather mild assumptions on the underlying problem. Traditional proofs of convergence for interior-point methods, both with and without penalty approaches, generally make assumptions on the boundedness of primal and dual iterates and the existence of strictly feasible primal and dual solutions, among others. If used, the penalty approach is generally employed to handle the equality constraints. Two recent works that propose and prove the convergence of interior-point methods with a penalty approach are [22] and [12]. In [22], Liu and Sun show that their interior-point algorithm converges globally under the standard assumptions of bounded primal and dual iterates. However, they also propose using a steepest descent approach whenever the Newton direction fails to be a descent direction. Doing so guarantees the convergence theoretically, but would greatly increase the iteration count within an implementation. In [12], Goldfarb and Chen analyze a Newton-method based interior-point method which is similar to the underlying algorithms in LOQO and IPOPT, two highly efficient interior-point codes. Their approach incorporates an ℓ_2 penalty function for use with NLPs that have inequality and equality constraints. The authors prove that the algorithm either (1) finds a first-order point, (2) approaches a minimizer of infeasibility, or (3) finds an optimal solution that does not satisfy the Karush-Kuhn-Tucker (KKT) conditions by letting the penalty parameter tend to infinity. They assume that the primal iterates and the Hessians calculated at each iterate remain bounded. For our proposed algorithm and the subsequent convergence proof, we will make use of the algorithm and results of the latter paper.

An NLP in standard form can be written as

$$(1) \quad \begin{array}{ll} \min_{y,z} & f(y, z) \\ \text{s.t.} & g(y, z) \geq 0 \\ & h(y, z) = 0 \\ & y \geq 0, \end{array}$$

where $y \in \mathbb{R}^p$, $z \in \mathbb{R}^l$, $f : \mathbb{R}^{p+l} \rightarrow \mathbb{R}$, $g : \mathbb{R}^{p+l} \rightarrow \mathbb{R}^q$, and $h : \mathbb{R}^{p+l} \rightarrow \mathbb{R}^k$. To simplify our notation, we will convert our problem into

$$(2) \quad \begin{array}{ll} \min_x & f(x) \\ \text{s.t.} & r(x) \geq 0, \end{array}$$

where

$$x = \begin{bmatrix} y \\ z \end{bmatrix}, \text{ and } r(x) = \begin{bmatrix} g(y, z) \\ h(y, z) \\ -h(y, z) \\ y \end{bmatrix}$$

We will let $n = p + l$ and $m = q + 2k + p$, so we have that $x \in \mathbb{R}^n$ and $r : \mathbb{R}^n \rightarrow \mathbb{R}^m$.

In order to establish convergence results for an interior-point method solving (2), we need to focus on two important features of this problem. The first is the presence of the equality constraints. We chose to split the equality constraint into two inequalities, which are incorporated separately into the penalty approach. Theoretically, it is essential to do this, because relaxing an equality constraint means introducing both lower and upper bounds. As later will be discussed, doing so does not violate linear independence of the constraint gradients since a separate relaxation variable is added to each constraint. Further, as demonstrated in [33], this does not have to increase the problem size in practice as one of the two inequalities can always be expressed solely in terms of slack variables and reduced. The second important feature of (1) is that we make no assumptions on the boundedness of the optimal set of dual solutions. It is a well-known fact (see, for instance [1]) that the set of optimal Lagrange multipliers for a large class of problems, mathematical programs with equilibrium constraints, is always unbounded. Equivalently, it can be stated that the Mangasarian-Fromovitz Constraint Qualification (MFCQ) cannot hold at a solution, or that the solution is not regular. However, assumptions of regularity, the satisfaction of MFCQ, or the boundedness of the optimal set of Lagrange multipliers must appear in standard proofs of convergence for primal-dual interior-point methods, because such methods approach the analytic center of the face of optimal primal and dual solutions, and unboundedness implies that the algorithm will fail to converge.

In [3], we proposed the use of a penalty method in order to resolve the issue of having unbounded Lagrange multipliers within the context of an interior-point method. In fact, other papers such as [1], also discuss the use of a penalty approach for bounding the Lagrange multipliers. In [3], however, we applied an ℓ_∞ penalty method by relaxing only the complementarity conditions. Here, we will consider an ℓ_1 penalty method that relaxes all of the constraints in the problem.

We first reformulate the problem using an ℓ_1 penalty approach on the constraints. Thus, problem (2) is first converted to

$$(3) \quad \begin{aligned} \min_{x, \xi} \quad & f(x) + d^T \xi \\ \text{s.t.} \quad & r(x) + \xi \geq 0 \\ & \xi \geq 0, \end{aligned}$$

where $\xi \in \mathbb{R}^m$ are the relaxation variables and $d \in \mathbb{R}^m$ are the corresponding (positive) penalty parameters. Note that the penalty approach used to form (3) corresponds to a smooth version of the ℓ_1 penalty method due to the introduction of the relaxation variables. Also, the use of vectors of penalty parameters serves to accommodate scale differences in the problem and allows for a more stable implementation.

Now, we can apply either of the interior-point methods presented in [12] to solve (3). The first is a quasi-feasible approach, where the constraints are always satisfied, and the second is an infeasible interior-point method, where all constraints are converted into equalities using slack variables, and the resulting equalities are handled with an ℓ_2 penalty method. While the proofs of convergence differ slightly, the convergence results for the two approaches are almost identical. The differences are: (1) The objective and constraint functions need to be twice continuously differentiable everywhere for the infeasible approach, whereas such differentiability is only needed for the interior of the feasible region for the quasi-feasible approach, and (2) The quasi-feasible approach requires a nonempty interior for the feasible region, whereas the infeasible approach does not. The first is more restrictive for the infeasible approach, while the second is more restrictive for the quasi-feasible approach. While we could also easily use the infeasible approach, in this paper, we focus on the combination of the quasi-feasible approach of [12] with the ℓ_1 penalty method. Doing so simplifies the notation as we will not need slack variables, but it is still easy to readily locate a feasible solution for the problem. We will assume that twice continuous differentiability will be required everywhere, but as we will discuss later, a less restrictive differentiability requirement may suffice.

2.1. The quasi-feasible interior-point method for (3). The barrier problem associated with (3) for the quasi-feasible approach is

$$(4) \quad \begin{aligned} \min_{x, \xi} \quad & \rho(x, \xi; \mu) = f(x) + d^T \xi - \mu \sum_{i=1}^m \log(r_i(x) + \xi_i) - \mu \sum_{i=1}^m \log(\xi_i) \\ \text{s.t.} \quad & r(x) + \xi > 0 \\ & \xi > 0, \end{aligned}$$

where $\mu > 0$ is the barrier parameter. Using this approach, therefore, requires that there exist a strictly feasible solution to (3). Such a solution may be hard to obtain without the ℓ_1 penalty on the inequality constraints, but simply setting $\xi_i > \max\{-r_i(x), 0\}$ will ensure that a strictly feasible solution for any x can be found readily.

The first-order conditions for this system are

$$(5) \quad \begin{aligned} \nabla f(x) - A(x)^T u &= 0 \\ d - u - \psi &= 0 \\ U(r(x) + \xi) - \mu e &= 0 \\ \Psi \xi - \mu e &= 0, \end{aligned}$$

where $A(x)$ is the transpose of the Jacobian of $r(x)$ and $u, \psi > 0$ are the Lagrange multipliers. By convention, U and Ψ are diagonal matrices with entries from the vectors u and ψ , respectively. In (5) and throughout the paper, e denotes a vector of ones of appropriate dimension.

Applying Newton's Method, at each iteration k we have the following system to solve:

$$(6) \quad \begin{aligned} H(x^k, u^k) \Delta x^k - A(x^k)^T v^k &= -\nabla f(x^k) \\ v^k + \lambda^k &= d \\ (R(x^k) + \Xi^k) v^k + U^k (A(x^k) \Delta x^k + \Delta \xi^k) &= \mu e \\ \Psi^k \Delta \xi^k + \Xi^k \lambda^k &= \mu e, \end{aligned}$$

where $R(x^k)$ and Ξ^k are diagonal matrices with entries from the vectors $r(x^k)$ and ξ^k , respectively,

$$H(x^k, u^k) = \nabla^2 f(x^k) - \sum_{i=1}^m u_i^k \nabla^2 r_i(x^k),$$

$v^k = u^k + \Delta u^k$, and $\lambda^k = \psi^k + \Delta \psi^k$. Then, we let

$$(7) \quad \begin{aligned} x^{k+1} &= x^k + \alpha^k \Delta x^k \\ \xi^{k+1} &= \xi^k + \alpha^k \Delta \xi^k \end{aligned}$$

where $\alpha^k \in (0, 1)$ is the steplength, such that

$$(8) \quad \begin{aligned} r(x^{k+1}) + \xi^{k+1} &> 0 \\ \xi^{k+1} &> 0 \\ \rho(x^{k+1}, \xi^{k+1}; \mu) - \rho(x^k, \xi^k; \mu) &< \tau \alpha^k (\nabla_x \rho(x^k, \xi^k; \mu)^T \Delta x^k + \nabla_\xi \rho(x^k, \xi^k; \mu)^T \Delta \xi^k) \end{aligned}$$

for some $\tau > 0$, where the last inequality is a standard Armijo condition on the decrease of the barrier objective function of (4). We update the Lagrange multipliers as follows:

$$(9) \quad \begin{aligned} u_i^{k+1} &= \begin{cases} v_i^k, & \text{if } \min\{\theta u_i^k, \frac{\mu}{r_i(x^k) + \xi_i^k}\} \leq v_i^k \leq \frac{\mu \gamma}{r_i(x^k) + \xi_i^k} \\ \min\{\theta u_i^k, \frac{\mu}{r_i(x^k) + \xi_i^k}\}, & \text{if } v_i^k < \min\{\theta u_i^k, \frac{\mu}{r_i(x^k) + \xi_i^k}\} \\ \frac{\mu \gamma}{r_i(x^k) + \xi_i^k}, & \text{if } v_i^k > \frac{\mu \gamma}{r_i(x^k) + \xi_i^k} \end{cases} \\ \psi_i^{k+1} &= \begin{cases} \lambda_i^k, & \text{if } \min\{\theta \psi_i^k, \frac{\mu}{\xi_i^k}\} \leq \lambda_i^k \leq \frac{\mu \gamma}{\xi_i^k} \\ \min\{\theta \psi_i^k, \frac{\mu}{\xi_i^k}\}, & \text{if } \lambda_i^k < \min\{\theta \psi_i^k, \frac{\mu}{\xi_i^k}\} \\ \frac{\mu \gamma}{\xi_i^k}, & \text{if } \lambda_i^k > \frac{\mu \gamma}{\xi_i^k} \end{cases}, \end{aligned}$$

where $\theta \in (0, 1)$ and $\gamma > 1$. The iterations continue until the first-order conditions (5) are satisfied to within a tolerance ϵ_μ for the current value of the barrier parameter, that is

$$(10) \quad \text{res}(x, \xi, u, \psi; \mu) = \left\| \begin{bmatrix} \nabla f(x) - A(x)^T u \\ d - u - \psi \\ U(r(x) + \xi) - \mu e \\ \Psi \xi - \mu e \end{bmatrix} \right\| < \epsilon_\mu.$$

Note that v and λ can be eliminated from the system (6) to obtain

$$(11) \quad \mathcal{M}^k \begin{pmatrix} \Delta x^k \\ \Delta \xi^k \end{pmatrix} = \begin{pmatrix} -\nabla f(x^k) + \mu A(x^k)^T (R(x^k) + \Xi^k)^{-1} e \\ \mu (R(x^k) + \Xi^k)^{-1} e + \mu (\Xi^k)^{-1} e - d \end{pmatrix},$$

Set $\gamma > 1$, $\theta \in (0, 1)$.
 Let $k = 0$.
while $res(x^k, \xi^k, u^k, \psi^k; \mu) > \epsilon_\mu$ **do**
 if (13) *does not hold* **then** modify $H(x^k, u^k)$ such that it is positive definite.
 Compute $(\Delta x^k, \Delta \xi^k, v^k, \lambda^k)$ from (6).
 Compute steplength α^k such that (8) is satisfied.
 Set u^{k+1} and ψ^{k+1} as in (9) and x^{k+1} and ξ^{k+1} as in (7).
 Set $k \leftarrow k + 1$.
end

Algorithm 1: A description of the solution algorithm for (4) for a fixed barrier parameter μ .

where

$$(12) \quad \begin{aligned} \mathcal{M}^k &= \begin{bmatrix} \hat{H}(x^k, \xi^k, u^k) & A(x^k)^T (R(x^k) + \Xi^k)^{-1} U^k \\ (R(x^k) + \Xi^k)^{-1} U^k A(x^k) & (R(x^k) + \Xi^k)^{-1} U^k + (\Xi^k)^{-1} \Psi^k \end{bmatrix} \\ \hat{H}(x^k, \xi^k, u^k) &= H(x^k, u^k) + A(x^k)^T (R(x^k) + \Xi^k)^{-1} U^k A(x^k). \end{aligned}$$

In order to use the convergence results of [12], we need to ensure that, for each iteration, the matrix \mathcal{M}^k is sufficiently positive definite. Since we have that

$$(R(x^k) + \Xi^k)^{-1} U^k + (\Xi^k)^{-1} \Psi^k \succ 0,$$

we need

$$\begin{aligned} &\hat{H}(x^k, \xi^k, u^k) - A(x^k)^T (R(x^k) + \Xi^k)^{-1} U^k \\ &((R(x^k) + \Xi^k)^{-1} U^k + (\Xi^k)^{-1} \Psi^k)^{-1} (R(x^k) + \Xi^k)^{-1} U^k A(x^k) \succ 0 \end{aligned}$$

or, equivalently,

$$(13) \quad \begin{aligned} &H(x^k, u^k) + A(x^k)^T (R(x^k) + \Xi^k)^{-1} U^k \\ &((R(x^k) + \Xi^k)^{-1} U^k + (\Xi^k)^{-1} \Psi^k)^{-1} (\Xi^k)^{-1} \Psi^k A(x^k) \succ 0 \end{aligned}$$

at each iteration. However, this may not always be the case. Then, we will modify $H(x^k, u^k)$ by adding a term of the form δI where δ is chosen to ensure that (13) holds. While any value of δ above a certain threshold will work in theory, we would also like to keep it as small as possible in order to make this algorithm work in practice, as larger values of δ will make the algorithm behave as in steepest descent, which is not desirable.

The algorithm provided above, Algorithm 1, is a simplified version of Algorithm I presented in [12]. There are several steps of Algorithm I that are not needed, such as the penalty parameter update as we no longer have equality constraints and the termination due to MFCQ failure as (3) always satisfies MFCQ. We will give a formal proof of convergence for this algorithm in the next section.

This algorithm solves (4) for a fixed value of the barrier parameter μ . In order to solve (3), we will need to solve a series of problems of the form (4) for decreasing values of μ . In order to do so, we will apply Algorithm II from [12]. A formal statement of this algorithm is provided below as Algorithm 2.

In the next section, we will show that Algorithm 2 will always terminate finitely and find a point that satisfies the first-order optimality conditions (5) of the penalty problem (3) for fixed values of the penalty parameters d . If $\|\xi\|$ is sufficiently close

Set $\mu^0 > 0$, $\epsilon_{\mu^0} > 0$, and $\beta \in (0, 1)$.
Initialize $u^{(0)} > 0$, $\psi^{(0)} > 0$, and $\xi_i^{(0)} > \max\{-r_i(x^{(0)}), 0\}$, $i = 1, \dots, m$.
Let $j = 0$.
while $\text{res}(x^j, \xi^j, u^j, \psi^j; 0) > \hat{\epsilon}$ **do**
 Starting from $(x^j, \xi^j, u^j, \psi^j)$, apply Algorithm 1 to solve (4) with barrier parameter μ^j and stopping tolerance ϵ_{μ^j} . Let the solution be $(x^{j+1}, \xi^{j+1}, u^{j+1}, \psi^{j+1})$.
 Set $\mu^{j+1} \leftarrow \beta\mu^j$ and $\epsilon_{\mu^{j+1}} \leftarrow \beta\epsilon_{\mu^j}$.
 Set $j \leftarrow j + 1$.
end

Algorithm 2: A description of the solution algorithm for (3) for fixed penalty parameters d .

Initialize $x^{(0)} \in \mathbb{R}^n$.
Set $d^{(0)} > 0$, $\nu > 1$, $\hat{i} > 1$, and $\hat{\epsilon} > 0$.
Let $i = 0$.
repeat
 if $(i = \hat{i})$ **then**
 Starting from x^i , apply Algorithm 2 to solve (3) with $f(x) \equiv 0$ and penalty parameters $d \equiv e$.
 Let the solution be $(x^{i+1}, \xi^{i+1}, u^{i+1}, \psi^{i+1})$.
 if $\|\xi^{i+1}\| \geq \hat{\epsilon}$ **then**
 STOP. Problem (1) is infeasible.
 else
 Set $i \leftarrow i + 1$.
 end
 end
 Starting from x^i , apply Algorithm 2 to solve (3) with penalty parameters d^i .
 Let the solution be $(x^{i+1}, \xi^{i+1}, u^{i+1}, \psi^{i+1})$.
 Set $d^{i+1} \leftarrow \nu d^i$.
 Set $i \leftarrow i + 1$.
until $\|\xi^i\| \leq \hat{\epsilon}$;

Algorithm 3: A description of the solution algorithm for (2).

to 0 at this point, we will declare it as a first-order point for (2). Otherwise, we will increase the penalty parameters and solve the problem again. After a finite number of updates, an infeasibility detection phase is entered, where we drop the original objective function and the penalty parameters and instead minimize the ℓ_1 norm of the infeasibility. If the norm is sufficiently larger than 0, the algorithm stops with a certificate of (local) infeasibility. Under an assumption on the exactness of the penalty approach, the resulting algorithm will be shown to either converge to a first-order point of (2) or to a minimizer of the ℓ_1 norm of the infeasibility after a finite number of penalty parameter updates. A formal description of the algorithm is given as Algorithm 3.

2.2. Definitions and other Preliminaries. Before establishing global and local convergence results for our algorithm, we need to present some of the definitions and other preliminaries that will be necessary in the following discussion. We start with a discussion of several constraint qualifications.

Definition 1. *A feasible solution for an NLP satisfies the linear independence constraint qualification (LICQ) for that NLP if the active constraint gradients at the solution are linearly independent.*

LICQ is a standard assumption for proving fast local convergence of algorithms and for establishing certain desirable properties of the penalty problem since it implies the uniqueness of the Lagrange multipliers at each feasible point. A less restrictive constraint qualification is the following:

Definition 2. *Given a point x feasible for (2), we say that the Mangasarian-Fromowitz Constraint Qualification (MFCQ) holds at x if either x is strictly feasible or there exists a vector $s \in \mathbb{R}^n$ such that*

$$\nabla r_i(x)^T s > 0, \quad \text{if } r_i(x) = 0, \quad i = 1, \dots, m$$

This constraint qualification implies that the set of Lagrange multipliers at the solution remains bounded. The definition of MFCQ can also be modified for the penalty problem, that is, given a point (x, ξ) feasible for (3), we say that MFCQ for (3) holds at (x, ξ) if either (x, ξ) is strictly feasible or there exists vectors $s \in \mathbb{R}^n$ and $t \in \mathbb{R}^m$ such that

$$\begin{aligned} \nabla r_i(x)^T s + t_i &> 0, & \text{if } r_i(x) + \xi_i = 0, & i = 1, \dots, m \\ t_i &> 0, & \text{if } \xi_i = 0, & i = 1, \dots, m. \end{aligned}$$

We will show in the next section that MFCQ always holds for (3).

Given the three problems that are being solved by the algorithms just described, we now provide some terminology to describe their solutions. We start with the barrier problem:

Definition 3. *The point (x, ξ) is a Karush-Kuhn-Tucker (KKT) point of (4) if it is a feasible point of (3) and there exist $u \geq 0$ and $\psi \geq 0$ such that $\text{res}(x, \xi, u, \psi; \mu) = 0$.*

Since Algorithm 1 solves each barrier problem only approximately, we need the following definition as well:

Definition 4. *The point (x, ξ) is an approximate KKT point of (4) if it is a feasible point of (3) and there exist $u \geq 0$ and $\psi \geq 0$ such that $\text{res}(x, \xi, u, \psi; \mu) < \chi$ for some $\chi > 0$.*

For the penalty problem (3), we have the following:

Definition 5. *A point (x, ξ, u, ψ) is a first-order point of the penalty problem (3) if (x, ξ) is feasible for (3), $u \geq 0$, $\psi \geq 0$, and conditions (5) are satisfied with $\mu = 0$.*

Finally, for (2), we have

Definition 6. *A point (x, u) is a first-order point of the problem (2) if*

$$\begin{aligned} r(x) &\geq 0 \\ \nabla f(x) - A(x)^T u &= 0 \\ Ur(x) &= 0 \\ u &\geq 0. \end{aligned}$$

The set of first-order points for (2) will be denoted by \mathcal{S} , and the set of vectors x for which an ξ exists such that (x, ξ) is a first-order point for (3) with penalty parameter d will be denoted by \mathcal{S}_d . If (2) is infeasible or if the set of Lagrange multipliers corresponding to (in the sense of a first-order point) the unique minimum of (2) is empty, then $\mathcal{S} = \emptyset$. We refer to the latter case as *nonKKT optima* and discuss it further in Section 5.

To establish our convergence results, we will need the penalty method to be exact:

Definition 7. *Let $\mathcal{S} \neq \emptyset$. Then, the penalty problem (3) is said to be exact for (2) if there exists \hat{d} such that $d > \hat{d}$ implies $\mathcal{S}_d \subseteq \mathcal{S}$.*

Exactness implies that, for penalty parameter values above a certain threshold, every first-order point of the penalty problem is a first-order point of the original problem. We will prove the nonemptiness and boundedness of \mathcal{S}_d in the next section. In Section 4, we will introduce some additional assumptions to analyze the local convergence of our approach. Under these additional assumptions, it is shown in [17] that if MFCQ holds for (2) then (3) is exact, so exactness is a weaker assumption than MFCQ and thereby LICQ. Additionally, the ℓ_1 penalty approach was shown in [1] to be exact for MPECs, even though this class of problems does not satisfy MFCQ. Therefore, the convergence results of this paper apply directly to solving MPECs as well.

As discussed, there is an infeasibility detection phase in Algorithm 3 after a finite number of updates to the penalty parameters. The goal of this phase is to simply locate a feasible solution to (3). The description of Algorithm 3 states that we drop $f(x)$ from the objective function, set each of the penalty parameters equal to 1 during this phase, and only retain a focus on reducing the constraint violations. Thus, in the infeasibility detection phase, we solve the following problem:

Definition 8. *The infeasibility problem corresponding to (2) is*

$$\begin{aligned} \min_{x, \xi} \quad & e^T \xi \\ \text{s.t.} \quad & r(x) + \xi \geq 0 \\ & \xi \geq 0. \end{aligned}$$

As before, we can define a first-order point for this problem:

Definition 9. *A point (x, ξ, u, ψ) is a first-order point of the infeasibility problem corresponding to (2) if*

$$\begin{aligned} r(x) + \xi & \geq 0 \\ \xi & \geq 0 \\ A(x)^T u & = 0 \\ e - u - \psi & = 0 \\ U(r(x) + \xi) & = 0 \\ \Psi \xi & = 0 \\ u & \geq 0 \\ \psi & \geq 0. \end{aligned}$$

We now provide two more definitions which will be used for the presentation of fast local convergence results.

Definition 10. For a strict local minimum x^* of (2), strict complementarity holds if there exists a corresponding Lagrange multiplier u^* satisfying the first-order conditions for (2) with

$$u_i^* + r_i(x^*) > 0, \quad i = 1, 2, \dots, m.$$

Definition 11. For a strict local minimum x^* of (2), the second-order sufficient condition (SOSC) holds if there exists a $\phi > 0$ such that

$$s^T H(x^*, u^*) s \geq \phi \|s\|^2,$$

for each of its corresponding Lagrange multipliers u^* and for all $s \in \mathbb{R}^n$ with $s \neq 0$ and $\nabla r_i(x)^T s = 0$ for each active constraint i .

3. GLOBAL CONVERGENCE RESULTS

We now present a formal convergence proof for Algorithm 3. We start by stating our assumptions and proving some intermediate results. Lemmas 5 and 6 will show convergence of Algorithms 1 and 2, respectively, by using the convergence results of [12]. Theorem 1 will establish the convergence of Algorithm 3.

Assumption 1. f and r are twice continuously differentiable everywhere.

This assumption also implies that the functions g and h of (1) are twice continuously differentiable. As stated before, this assumption is more restrictive than actually needed. It is sufficient to assume that f and r are twice continuously differentiable in a region that can be defined by bounds on x or on ξ . Another option is to use the primal-dual penalty approach proposed in [7] which naturally provides upper bounds for the primal variables.

Assumption 2. While applying Algorithm 1, the sequence $\{x^k\}$ lies in a bounded set.

Lemma 1. Let Assumptions 1 and 2 hold. Then, the sequence $\{\xi^k\}$ will also remain bounded.

Proof. If the sequence $\{x^k\}$ generated by Algorithm 1 remains bounded, then by Assumption 1, we have that $\{\Delta x^k\}$, $\{r(x^k)\}$, and $\{A(x^k)\Delta x^k\}$ will also remain bounded.

Assume now that $\xi_i^k \rightarrow \infty$ for some i as $k \rightarrow \infty$. By the above observations and multiplier updates defined by (9), for a fixed μ , we have that $\psi_i^k \rightarrow 0$ and $u_i^k \rightarrow 0$. Also by (9), $\psi_i^k > 0$ and $u_i^k > 0$ for all k .

The above observations also imply that $r_i(x^k) + \xi_i^k \rightarrow \infty$ and $\sum_{j=1}^m A_{ij}(x^k)\Delta x^k + \xi_i^k \rightarrow \infty$ as $k \rightarrow \infty$. Also, as $\xi_i^k \rightarrow \infty$, we must have that the corresponding step directions will have a subsequence such that $\Delta \xi_i^k > 0$ for each k in the subsequence.

Putting all of these together, the last two equations of (6) for a fixed value of μ imply that there exists a subsequence $\{(v_i^k, \lambda_i^k)\}$ such that the multiplier estimates are either negative or approaching 0. However, any combination of such values would violate the second equation in (6). Therefore, $\{\xi^k\}$ must remain bounded. \square

Assumption 1 is a standard assumption for all algorithms using Newton's Method. As stated, Assumption 2, along with Lemma 1, implies that the primal iterates generated by this feasible algorithm remain bounded, which is also a standard assumption for convergence proofs. In fact, they are listed as assumptions A2 and A3 in [12]. These are rather mild assumptions, and, in fact, they are the only

assumptions we will need to prove convergence of our approach. The remaining assumptions made in [12] will always be satisfied as a result of using the ℓ_1 penalty as shown below in Lemmas 2 and 3.

Lemma 2. *The feasible region of (3) has a nonempty interior.*

Proof. Let $\hat{x} \in \mathbb{R}^n$ and let $\hat{\xi}_i > \max\{-r_i(\hat{x}), 0\}$, $i = 1, \dots, m$. Doing so ensures that

$$\begin{aligned} r_i(\hat{x}) + \hat{\xi} &> 0 \\ \hat{\xi} &> 0. \end{aligned}$$

Therefore, $(\hat{x}, \hat{\xi})$ lies in the interior of the feasible region of (3). This proves that (3) has a nonempty interior. \square

Lemma 3. *Let Assumptions 1 and 2 hold. The sequence of modified Hessians, $\{H^k\}$, is bounded.*

Proof. The update rules (9) ensure that the dual iterates will always remain bounded above. By this fact, Assumptions 1 and 2, and Lemma 1, the sequence of Hessians is bounded. This means that there is a finite threshold $\bar{\delta}$ such that $\{H^k + \delta I\}$ is positive definite for $\delta \geq \bar{\delta}$. Since $(r(x^k) + \xi^k)$ and ξ^k are kept bounded away from 0, the second term in (13) is positive definite. Thus, the required Hessian modification to satisfy (13) is bounded above by $\bar{\delta}I$, and the sequence of modified Hessians is also bounded. \square

As stated, Lemmas 2 and 3 correspond to assumptions A1 and A4 of [12], respectively. Therefore, the ℓ_1 penalty ensures that we will need fewer assumptions in order to prove global convergence of the interior-point method and strengthens the results of [12]. Additionally, the infeasibility detection phase of Algorithm 3 ensures that the penalty parameter does not need to approach infinity as in [12] when (2) is infeasible. We will make one additional assumption in this paper to ensure that the penalty parameters remain finite when a feasible solution exists:

Assumption 3. *If $S = \emptyset$, then (1) is infeasible. Otherwise, (3) is exact for (1).*

This assumption is not necessary to obtain the same results as [12], which allows for approaching Fritz John points of (2) that do not satisfy MFCQ as $d \rightarrow \infty$. However, we include Assumption 3 to keep the penalty parameters finite for Algorithm 3 and to provide a unified framework for global and local convergence results. We will discuss the cases where this assumption may not hold separately in Section 5.

We will now focus on showing that the proposed Algorithms 1 and 2 are equivalent to Algorithms I and II for the quasi-feasible approach of [12]. The ℓ_1 penalty will, in fact, allow us to simplify the algorithms by eliminating certain steps. We start by presenting some preliminaries and then establish the convergence of Algorithms 1 and 2 in Lemmas 5 and 6.

Lemma 4. *MFCQ holds on the feasible region of (3).*

Proof. Let (x, ξ) lie in the feasible region of (3). The existence of (x, ξ) is guaranteed by Lemma 2. If (x, ξ) is strictly feasible, then MFCQ holds at (x, ξ) . Otherwise, let $s = 0$ and $t = e$. Therefore,

$$\begin{aligned} \nabla r_i(x)^T s + t_i &= 1, & \text{if } r_i(x) + \xi_i &= 0, & i &= 1, \dots, m \\ t_i &= 1, & \text{if } \xi_i &= 0, & i &= 1, \dots, m, \end{aligned}$$

and MFCQ holds at (x, ξ) . Thus, MFCQ holds on the feasible region of (3). \square

Lemma 5. *Let Assumptions 1 and 2 hold. Then, Algorithm 1 will always find an approximate KKT point of (4) satisfying termination criteria (10) in a finite number of iterations.*

Proof. Along with Assumptions 1 and 2, Lemmas 2 and 3 show that assumptions A1-A4 of [12] hold. Therefore, we can use Theorem 3.12 of [12]. By Lemma 4, MFCQ always holds, so the second part of Step 1 of Algorithm I of [12] is not needed and, therefore, not included in Algorithm 1. Also, there is no need for an ℓ_2 penalty term because (3) does not have equality constraints. Therefore, there is no ℓ_2 penalty parameter that could be updated indefinitely. As such, a penalty parameter update step is not included in Algorithm 1. Thus, our Algorithm 1 is identical to Algorithm I of [12] for solving (4). By Theorem 3.12 of [12], we will always identify an approximate KKT point of the barrier problem, satisfying the termination criteria (10) in a finite number of iterations. \square

Thus, Lemma 5 shows that Algorithm 1 will always succeed in finding an approximate KKT point for the barrier problem for a fixed value of the barrier parameter μ . Since Algorithm 1 only requires an approximate KKT point, Lemma 5 is sufficient for our purposes. We need to now show that we can solve (3) for a fixed value of the penalty parameter, which means showing that the sequence of approximate solutions found for the barrier problem for decreasing values of μ converges to a first-order point of the penalty problem. Again, we will do so by showing that our proposed algorithm, Algorithm 2, is equivalent to Algorithm II of [12], allowing us to use its convergence results.

Lemma 6. *Let Assumptions 1 and 2 hold. Ignoring the termination condition of Algorithm 2, any limit point of the sequence $\{(x^j, \xi^j, u^j, \psi^j)\}$ generated by Algorithm 2 is a first-order point of the penalty problem (3).*

Proof. Along with Assumptions 1 and 2, Lemmas 2 and 3 show that assumptions A1-A4 of [12] hold. By Lemma 4, we have that MFCQ holds on the feasible region of (3), and by Lemma 2, we have that (3) always has a local feasible solution. Therefore, we satisfy all the assumptions of Theorem 3.14 of [12]. Since there is no ℓ_2 penalty function, there is no penalty parameter to keep bounded. Because our Algorithm 2 is identical to Algorithm II of [12], Theorem 3.14 applies here as well. Therefore, ignoring the termination condition of Algorithm 2, any limit point of the sequence $\{(x^j, \xi^j, u^j, \psi^j)\}$ generated by Algorithm 2 is a first-order point of the penalty problem (3). \square

Lemma 6 shows that Algorithm 2 will always find a first-order point of the penalty problem (3) for fixed values of the penalty parameters d . We will now analyze Algorithm 3, which solves (3) for increasing values of d in order to find a first-order point for (2) if such a point exists. We begin with some preliminaries and then state our convergence result in Theorem 1.

Theorem 1. *Let Assumptions 1, 2, and 3 hold. Then, Algorithm 3 terminates successfully after a finite number of iterations, and one of the following outcomes occurs:*

- (1) *The limit point of any subsequence $\{(x^i, u^i)\}$ is a first-order point of (2).*
- (2) *A first-order point of the infeasibility problem is found.*

Proof. Case 1: If Algorithm 3 enters the infeasibility detection phase, it will solve the infeasibility problem using Algorithm 2. Since $f(x) \equiv 0$ and $d \equiv e$ in this phase, by Lemma 6, Algorithm 2 will find a first-order point of the infeasibility problem. Let this first-order point be $\{(\hat{x}, \hat{\xi}, \hat{u}, \hat{\psi})\}$. If $\|\hat{\xi}\| \geq \hat{\epsilon}$, then Algorithm 3 will terminate at this point.

Case 2: If Algorithm 3 never enters the infeasibility detection phase, the penalty parameters will get updated finitely many times (at most \hat{i} times) before the termination criterion $\|\xi^i\| \leq \hat{\epsilon}$ is satisfied for some iteration i . Therefore, for a sequence of solutions $\{(x^i, \xi^i, u^i, \psi^i)\}$ to Algorithm 2, we can assume that we have a finite subsequence such that $\{\|\xi^i\|\}$ approaches 0. The solutions to Algorithm 2 (approximately) satisfy the conditions

$$(14) \quad \begin{aligned} r(x^i) + \xi^i &\geq 0 \\ \xi^i &\geq 0 \\ \nabla f(x^i) - A(x^i)^T u^i &= 0 \\ d^i - u^i - \psi^i &= 0 \\ U^i(r(x^i) + \xi^i) &= 0 \\ \Psi^i \xi^i &= 0, \end{aligned}$$

which reduce to the first-order conditions of (2) as $\|\xi^i\|$ approaches 0.

Case 3: If Algorithm 3 enters the infeasibility detection phase, but finds a solution with $\|\hat{\xi}\| < \hat{\epsilon}$, then a feasible solution to (1) exists. Therefore, Algorithm 3 will continue solving (1) using larger values of d . By Assumption 3, there exists a finite threshold \hat{d} such that for $d^i > \hat{d}$, the first-order points located by Algorithm 2 for (3) are also first-order points of (2). Therefore, Algorithm 3 will terminate at such a point when $d^i > \hat{d}$ is satisfied after a finite number of updates. \square

4. LOCAL CONVERGENCE RESULTS

In the previous section, we showed that our proposed algorithm gets arbitrarily close to a solution of (2), if one exists. In this section, we examine the local convergence properties of the proposed algorithm, including its rate of convergence. Most local convergence proofs such as the one in [13] require the satisfaction of LICQ. However, the constraint Jacobian for (3) is given by

$$\begin{bmatrix} A(x) & I \\ 0 & I \end{bmatrix},$$

and if LICQ does not hold for the original problem, then it will not hold for the penalty problem, either. Thus, requiring LICQ limits the applicability of any algorithm, including ours, to large classes of problems which do not satisfy this qualification. Similarly, requiring MFCQ does not allow for our algorithm to be applicable to MPECs. Therefore, we will not include any such constraint qualifications in this paper. Instead, we will make some assumptions about the neighborhood of a local minimizer and employ the theory developed in [36] to show fast local convergence of Algorithm 2. Coupled with the assumption of exactness, this will allow us to show fast local convergence of Algorithm 3.

First, we need to make a modification to our algorithm, which will allow us to use the results of [36]. This modification is to the choice of the barrier parameter

at each iteration of Algorithm 2. Rather than setting $\mu^{j+1} \leftarrow \beta\mu^j$, we will update the barrier parameter as follows:

$$(15) \quad \mu^{j+1} \leftarrow \min\{\beta\sigma(x^j, \xi^j, u^j, \psi^j), \hat{\beta}\sigma(x^j, \xi^j, u^j, \psi^j)^2\},$$

where

$$\sigma(x, \xi, u, \psi) = \max\left\{\left\|\begin{bmatrix} \nabla f(x) - A(x)^T u \\ d - u - \psi \end{bmatrix}\right\|, \left\|\begin{bmatrix} U(r(x) + \xi) \\ \Psi\xi \end{bmatrix}\right\|\right\}$$

and $\hat{\beta} > 1$. Note that doing so ensures that $\mu^{j+1} \leftarrow \beta\mu^j$ still holds away from the solution, as $\sigma(x, \xi, u, \psi)$ will be dominated by the complementarity terms which are all close to μ^j . Close to the solution, however, the barrier parameter will be decreased at a quadratic rate. We set $\epsilon_\mu = \mu/10$, so that it also decreases at the same rate as the barrier parameter.

There are also two additional assumptions required for fast local convergence: strict complementarity and second-order sufficiency.

Assumption 4. *Strict complementarity holds for (2) at a strict local minimum x^* with Lagrange multiplier u^* .*

Assumption 5. *SOSC holds for (2) at a strict local minimum x^* .*

Lemma 7. *Under Assumption 4, strict complementarity holds for (3) with penalty parameter $d > u^*$ at $(x^*, 0)$.*

Proof. The condition

$$u_i + (r_i(x) + \xi_i) > 0, \quad i = 1, 2, \dots, m$$

reduces to

$$u_i + r_i(x) > 0, \quad i = 1, 2, \dots, m$$

when $\xi = 0$. Also, the first-order condition

$$d - u - \psi = 0$$

implies that $\psi^* = d - u^* > 0$. Therefore, there exists a Lagrange multiplier (u^*, ψ^*) such that strict complementarity holds at $(x^*, 0)$. \square

Lemma 8. *Under Assumption 5, SOSC holds for (3) at $(x^*, 0)$.*

Proof. At $(x^*, 0)$, we have that

$$\begin{aligned} r(x^*) + \xi^* &= r(x^*) \\ \xi^* &= 0. \end{aligned}$$

Thus, the set of active constraints for (3) at $(x^*, 0)$ is the set of active constraints for (2) at x^* along with the nonnegativity constraints on the relaxation variables.

Letting $(s, \hat{s}) \in \mathbb{R}^{n+m}$ with $(s, \hat{s}) \neq 0$ be in the null space of the active constraints of the penalty problem, we have that

$$\begin{bmatrix} 0 \\ e_i \end{bmatrix}^T \begin{bmatrix} s \\ \hat{s} \end{bmatrix} = 0$$

for each nonnegativity constraint i , with e_i denoting the vector of zeros except for a one in the i th place. This implies that $\hat{s} = 0$. Also, we have that

$$\begin{bmatrix} \nabla r_i(x) \\ e_i \end{bmatrix}^T \begin{bmatrix} s \\ \hat{s} \end{bmatrix} = 0$$

for each active inequality constraint i , which implies that $\nabla r_i(x)^T s = 0$. By Assumption 5, there exists a $\phi > 0$ such that $s^T H(x^*, u^*) s \geq \phi \|s\|^2$ for each corresponding u^* .

For (3), we have that

$$\begin{bmatrix} s \\ \hat{s} \end{bmatrix}^T \begin{bmatrix} H(x^*, u^*) & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} s \\ \hat{s} \end{bmatrix} = s^T H(x^*, u^*) s \geq \phi \|s\|^2.$$

Therefore, SOSC holds for (3) at $(x^*, 0)$. \square

Note that since we do not assume the uniqueness of the Lagrange multipliers corresponding to the x^* in the first-order conditions, the above definitions and lemmas accommodate the existence of multiple vectors. Therefore, the definition of strict complementarity only requires one among possibly many optimal vectors of Lagrange multipliers, and SOSC is defined to hold at every vector of optimal Lagrange multipliers.

The algorithm presented in [36] is a purely primal one. For each barrier parameter μ , instead of solving (6) and then using (9) to compute the dual variables, [36] solves the system (11) to obtain the primal step directions and sets the values of the dual variables as follows:

$$(16) \quad \begin{aligned} u_i^{k+1} &= \frac{\mu}{r_i(x^k) + \xi_i^k} \\ \psi_i^{k+1} &= \frac{\mu}{\xi_i^k}. \end{aligned}$$

Instead of making these changes, we will show that (9) implies that our dual variables are related to these quantities. The reason for the choice of the Lagrange multipliers in [36] is to be able to use the a critical lemma which shows that the choices in (16) for updating the multipliers remain bounded in the neighborhood of a first-order point. We now show that the same property holds for our choice of Lagrange multipliers given by (9).

Lemma 9. *Suppose that $(x^*, \xi^*, u^*, \psi^*)$ is a first-order point for (3). Then, there are positive constants r and ϵ such that the following property holds: For any $\zeta_1 \geq 0$, there exists a ζ_2 such that for all (x^k, ξ^k) and μ with*

$$\begin{aligned} \sqrt{\|x^k - x^*\|^2 + \|\xi^k - \xi^*\|^2} &\leq r, \\ (x^k, \xi^k) &\text{feasible for (3),} \\ \mu &\in (0, \epsilon], \text{ and} \\ \|\nabla \rho(x^k, \xi^k; \mu)\| &\leq \zeta_1, \end{aligned}$$

we have

$$\sqrt{\|u^k\|^2 + \|\psi^k\|^2} \leq \zeta_2.$$

Proof. By Lemma 4, we know that MFCQ holds for (3). Lemma 2 of [36] shows that under this condition, there exists a ζ such that

$$\sqrt{\|\mu(R(x^k) + \Xi^k)^{-1} e\|^2 + \|\mu(\Xi^k)^{-1} e\|^2} \leq \zeta.$$

By (9), we have that

$$\begin{aligned} 0 &< u_i^k &\leq \frac{\mu\gamma}{r_i(x^k) + \xi_i^k} \\ 0 &< \psi_i^k &\leq \frac{\mu\gamma}{\xi_i^k} \end{aligned}$$

for some constant $\gamma > 1$. Setting $\zeta_2 = \gamma\zeta$ proves the lemma. \square

As stated, the convergence results of [36] use the Lagrange multiplier updates defined by (16). Lemma 2 of [36], which shows the boundedness of the dual iterates, is key for proving convergence in [36]. Given the same set of assumptions on the problem, we have shown that Lemma 9 proves the same result. We now show that Lemma 3 of [36] also holds.

Lemma 10. *Suppose that $(x^*, \xi^*, u^*, \psi^*)$ is a first-order point for (3). Then, given any sequences $\{\mu^j\}$ and $\{(x^j, \xi^j)\}$ with $(x^j, \xi^j) \rightarrow (x^*, \xi^*)$, $\mu^j \downarrow 0$, and $\nabla \rho(x^j, \xi^j; \mu^j) \rightarrow 0$, we have that*

$$\text{dist}_{S_d}(u^j, \psi^j) = O(\sqrt{\|x^j - x^*\|^2 + \|\xi^j - \xi^*\|^2}) + O(\mu^j) + O(\|\nabla \rho(x^j, \xi^j; \mu^j)\|).$$

Proof. By Lemma 4, we know that MFCQ holds for (3). Lemma 3 of [36] shows that under this condition,

$$\begin{aligned} \text{dist}_{S_d}(\mu^j(R(x^j) + \Xi^j)^{-1}e, \mu^j(\Xi^j)^{-1}e) = \\ O(\sqrt{\|x^j - x^*\|^2 + \|\xi^j - \xi^*\|^2}) + O(\mu^j) + O(\|\nabla \rho(x^j, \xi^j; \mu^j)\|). \end{aligned}$$

Again, by (9), we have that

$$\text{dist}_{S_d}(u^j, \psi^j) \leq \gamma \text{dist}_{S_d}(\mu^j(R(x^j) + \Xi^j)^{-1}e, \mu^j(\Xi^j)^{-1}e)$$

for some fixed value $\gamma > 1$, which proves the lemma. \square

An additional assumption in [36] is that at least one of the constraints is active at the optimal solution. We now show that this assumption will always hold for (3).

Lemma 11. *For a strict local minimum (x^*, ξ^*) of (3), $r_i(x^*) + \xi_i^* = 0$ or $\xi_i^* = 0$ for some $i = 1, 2, \dots, m$.*

Proof. Suppose that for all $i = 1, 2, \dots, m$, we have that

$$\begin{aligned} r_i(x^*) + \xi_i^* &> 0 \\ \xi_i^* &> 0. \end{aligned}$$

Let

$$\hat{\xi}_i = \min\{r_i(x^*) + \xi_i^*, \xi_i^*\}.$$

Thus, we have that $\hat{\xi} > 0$, and, therefore, $\xi^* - \hat{\xi} < \xi^*$ componentwise. Note also that by the way we constructed $\hat{\xi}$, $(x^*, \xi^* - \hat{\xi})$ is feasible for (3). Since

$$f(x^*) + d^T(\xi^* - \hat{\xi}) < f(x^*) + d^T \xi^*,$$

(x^*, ξ^*) cannot be the strict local minimum, and we have a contradiction. \square

Lemma 12. *Under Assumptions 1, 2, 4, and 5, with the above modifications for fast local convergence, Algorithm 2 will always find a strict local optimum (x^*, ξ^*) of the penalty problem (3) and do so at a Q-quadratic rate.*

Proof. By Lemma 4, (3) always satisfies MFCQ. Therefore, the assumptions of Theorem 8 of [36] are satisfied, which means that a full Newton step for Algorithm 1 is well-defined in a neighborhood of a local minimizer of (3) and that the iterates of Algorithm 2 converge Q-quadratically to this minimizer. \square

Lemma 13. *Under Assumptions 1, 2, 4, and 5, with the above modifications for fast local convergence, the iterates $(x^j, \xi^j, u^j, \psi^j)$ generated by Algorithm 2 satisfy $(u^j, \psi^j) \rightarrow (\mu^j(R(x^j) + \Xi^j)^{-1}e, \mu^j(\Xi^j)^{-1}e)$.*

Proof. By the previous lemma, under the same assumptions, Algorithm 2 will converge to a strict local optimum of (3). As such, Δx^k and $\Delta \xi^k$ computed by Algorithm 1 will approach 0 at each iteration of Algorithm 2. Since $\epsilon_{\mu^j} \downarrow 0$ as $\mu^j \downarrow 0$, the complementarity conditions of (14) will satisfy $(u^j, \psi^j) \rightarrow (\mu^j(R(x^j) + \Xi^j)^{-1}e, \mu^j(\Xi^j)^{-1}e)$. \square

Lemma 14. *Under Assumptions 1, 2, 4, and 5, with the above modifications for fast local convergence, the iterates $(x^j, \xi^j, u^j, \psi^j)$ generated by Algorithm 2 converge to a first-order point of (3) denoted by $(x^*, \xi^*, u^*, \psi^*)$, where (x^*, ξ^*) is a strict local optimum of the penalty problem (3).*

Proof. By Lemma 4, (3) always satisfies MFCQ. Therefore, the assumptions of Theorem 11 of [36] are satisfied, and the theorem implies that the sequence $\{(\mu^j(R(x^j) + \Xi^j)^{-1}e, \mu^j(\Xi^j)^{-1}e)\}$ converges to the analytic center of the face of dual solutions that satisfy the first-order conditions of (3) for (x^*, ξ^*) . We let (u^*, ψ^*) denote the analytic center, which is guaranteed to exist and is finite by MFCQ. Then, by the previous two lemmas, we have that $(x^j, \xi^j, u^j, \psi^j) \rightarrow (x^*, \xi^*, u^*, \psi^*)$. \square

We now show that with the modifications made above, Algorithm 3 converges locally to a strict local minimum of (2) and does so at a quadratic rate.

Theorem 2. *If Assumptions 1-5 hold for (2) at a first-order point (x^*, u^*) , then for sufficiently large but finite d^* , a solution $(x, \xi, u, \psi) = (x^*, 0, u^*, d^* - u^*)$ exists for (3). If Algorithm 3 is applied with the above modifications at a point $(x^i, \xi^i, u^i, \psi^i)$ sufficiently close to this solution with the penalty parameter set to d^* , then it converges to x^* with no further update of the penalty parameter needed, and the rate of convergence is Q -quadratic.*

Proof. The first-order conditions (5) for (3) reduce to those of (2) at the solution $(x^*, 0, u^*, d^* - u^*)$ for $d^* > u^*$, so it is a first-order point for (3). This means that when the penalty parameter is set to d^* and we start sufficiently close, we only need one iteration of Algorithm 3, which means invoking Algorithm 2 once, to find the solution. Thus, by Lemmas 12-14, Algorithm 3 converges to the solution and does so quadratically. \square

5. BEHAVIOR OF THE PENALTY PROBLEM NEAR AN ISOLATED MINIMIZER

Note that both our global and local convergence results assume that we have converged to a KKT solution whenever the problem is not infeasible. However, we can still prove some results for a strict local minimum, x^* , that may not necessarily be a KKT solution. We proved a similar result in Theorem 3 of [3], but that was done with some assumptions, which we now drop.

In this section, we will establish that there is a sequence of solutions to the penalized problem (2) which will converge to x^* (even if x^* is not a KKT solution). Again, we are not able to assert that the our algorithm will *necessarily* follow such a sequence. However, absent truly pathological behavior, the algorithm often will be able to track such a path if a starting point is chosen sufficiently close to x^* for sufficiently large d . What we have provided is an explanation for behavior that has in fact been seen in practice (see [3]), not a guarantee that such behavior will always occur.

In the following results, the notation $B_\kappa(x)$ refers to a ball of positive radius κ centered at the point x .

Lemma 15. *For all $B_\kappa(x^*)$ there exists a \bar{d} such that $x^i \in B_\kappa(x^*)$ for all $d^i > \bar{d}$.*

Proof. We choose κ to be small enough such that $f(x^*) < f(x)$ for all $x \in B_\kappa(x^*)$ with x feasible for (2). Suppose $B_\kappa(x^*)$ does not contain x^i for some d^i . The function $f(x) + (d^i)^T \xi$ attains a global minimum over the set of solutions satisfying

$$\begin{aligned} r(x) + \xi &\geq 0 \\ \xi &\geq 0 \\ x &\in B_\kappa(x^*), \end{aligned}$$

since x is restricted to a closed and bounded set. Call this global minimum $(\hat{x}, \hat{\xi})$. By assumption this global minimum must lie on the boundary of $B_\kappa(x^*)$ and must be better than any point $(\bar{x}, \bar{\xi})$, where \bar{x} is in the interior of $B_\kappa(x^*)$ and $\bar{\xi}$ is any (nonnegative) value of the relaxation variables, that is,

$$(17) \quad f(\hat{x}) + (d^i)^T \hat{\xi} < f(\bar{x}) + (d^i)^T \bar{\xi}$$

Furthermore, it must be true that $\|\hat{\xi}\| > 0$, since otherwise \hat{x} would be feasible for (2) with $f(\hat{x}) < f(x^*)$ which cannot happen. This can be verified by letting $\bar{x} = x^*$ and $\bar{\xi} = \hat{\xi}$. Now let \mathcal{D} be the set of all such d just described (i.e. those for which the penalty problem has no solutions in $B_\kappa(x^*)$). We assume for the moment that \mathcal{D} is unbounded. If we have that

$$(18) \quad \exists \epsilon > 0 \text{ s.t. } \|\hat{\xi}\| > \epsilon \forall d \in \mathcal{D},$$

then it is not hard to see that, since f is bounded on $B_\kappa(x^*)$, we must have for sufficiently large d in \mathcal{D} ,

$$f(\hat{x}) + d^T \hat{\xi} > f(x^*),$$

which of course contradicts (17) with $\bar{x} = x^*$ and $\bar{\xi} = 0$. So suppose (18) does not hold. Then we have that, for some sequence $\{d^l\}$ in \mathcal{D} , $\lim_{l \rightarrow \infty} d^l = \tilde{d}$ which must be feasible for (2) and lie in $B_\kappa(x^*)$ (since a closed set must contain its limit points). It must be true that $f(\tilde{x}) > f(x^*)$, and yet $\lim_{l \rightarrow \infty} f(\hat{x}^l) + (d^l)^T \hat{\xi}^l = f(\tilde{x})$, and $f(\hat{x}^l) + (d^l)^T \hat{\xi}^l < f(x^*)$ for all l . Obviously this cannot happen since the limit of a sequence cannot be strictly greater than a number if all the elements of the sequence are strictly less than it. Therefore \mathcal{D} must be bounded and the lemma is proved. \square

The proof of the claim now follows:

Theorem 3. *There is a sequence of solutions (x^i, ξ^i) which converges to $(x^*, 0)$ as $d^i \rightarrow \infty$.*

Proof. We simply apply Lemma 15, and the result is obvious. In fact there are uncountably many such sequences. \square

6. NUMERICAL RESULTS

We implemented the algorithm described in Section 2 and a variant including the changes (15) of Section 4 for fast local convergence and present details of their numerical performance here. The code was written in C, using modified versions of the linear algebra routines and the AMPL interface of the nonlinear solver LOQO.

The modification to the linear solver was necessary because the matrix defined by (12) needs to be kept positive definite, and LOQO expects its corresponding matrix to be quasi-definite. As analyzed in [31], a quasi-definite matrix is strongly factorizable, so the change made to the linear solver was simply to the signs of the matrix entries, which is trivial and does not impact the performance.

The default settings for the algorithm parameters were chosen as follows:

$$\begin{array}{ll} \nu & = 10 & \beta & = 0.1 \\ \gamma & = 1000 & \theta & = 0.1 \\ \alpha & = 0.95 & \hat{\beta} & = 1000. \end{array}$$

All penalty parameters were initialized to 100, and for each value of the penalty parameter, the initial value of the barrier parameter μ was 100. We let $\epsilon_{\mu^j} = \mu^j$, and $\hat{\epsilon} = 10^{-7}$. Numerical testing was conducted on a laptop running Fedora Core 8 with 3GB of main memory and dual CPUs with 2.4GHz clock speeds. We did not specifically take advantage of the dual CPUs in our numerical testing.

Tables 1-5 also provide a comparison to the code LOQO, Version 6.06. LOQO implements a primal-dual interior-point method for nonconvex NLP as described in [33], with the modification of using a filter-based linesearch for steplength calculations, as described in [9]. It updates the barrier parameter at every iteration, based on the uniformness of the complementarity products, and even allows the barrier parameter to increase from one iteration to the next. The version tested is the official release which does not use constraint relaxation, does not incorporate the primal-dual penalty approach of [7] and does not guarantee that the Lagrange multipliers will remain bounded. It currently has no convergence proof, but, as numerical studies including [7] and [8] show, it is a highly efficient, state-of-the-art NLP code.

Numerical results on an implementation of the algorithm of Chen and Goldfarb [12] are available and reported in [11]. These results were obtained by modifying IPOPT to implement the key features of the solution algorithm, such as the penalty method and the barrier parameter updates. However, it retains many of the other algorithmic features and efficiency improvements of IPOPT, and as such compares very favorably to existing codes. In our numerical work, we have chosen to implement strictly the algorithm as described here and provide a thorough analysis of many features that can have an effect on the overall performance.

The numerical testing was conducted on a wide range of problems from literature. Tables 1-3 present the testing results on the Hock and Schittkowski [18] test suite, for which the AMPL models were obtained from [30]. We also tested our algorithm on a set of degenerate problems described in [24] and present the results in Table 4. We made AMPL models of the problems, and they can be obtained from [6]. Problems *DEGEN1*, *DEGEN4*, and *DEGEN8* are identical to *HS113*, *HS13*, and *HS32*, respectively, but we included them again in Table 4 for completeness. Problems whose names end in “f” are feasibility problems, and the first twelve are obtained by replacing the objective function of the corresponding problem with 0. Problems *DEGEN13f*, *DEGEN13fb*, and *DEGEN14f* are infeasible problems. Finally, in Table 5, we have presented results on some MPECs gathered from the MacMPEC test suite [20] along with a few created from MPEC representations of large integer programming problems. Results on the full suite [20] were reported in [3] for LOQO. Results on the whole CUTer [14] test set for LOQO are also available from [4]. We have not included a discussion of the full test sets here as the results

are long and Tables 1-5 proved adequate to make our points. The convergent algorithm was able to handle almost all of the problems successfully, and we will now discuss some of the interesting cases.

6.1. Problems with nonKKT optima. There are several problems in the test set, including *HS13*, *DEGEN4*, *ex9.2.2*, *qpec2*, *ralph1*, and *scholtes4*, for which no feasible Lagrange multipliers exist at the optimal solution. Problem 13 of the Hock and Schittkowski test suite is

$$(19) \quad \begin{aligned} \min \quad & (x_1 - 2)^2 + x_2^2 \\ \text{s.t.} \quad & (1 - x_1)^3 - x_2 \geq 0 \\ & x_1 \geq 0 \\ & x_2 \geq 0. \end{aligned}$$

It is easy to see that the optimal solution of this problem is $(x_1, x_2) = (1, 0)$. The optimality conditions of (19) are

$$\begin{aligned} 2(x_1 - 2) + 3(1 - x_1)^2 y_1 - y_2 &= 0 & y_1 \left((1 - x_1)^3 - x_2 \right) &= 0 \\ 2x_2 + y_1 - y_3 &= 0 & y_2 x_1 &= 0 \\ y_1, y_2, y_3 &\geq 0 & y_3 x_2 &= 0. \end{aligned}$$

At the optimal solution, the first condition reduces to

$$y_2 = -2$$

which violates the nonnegativity of y_2 . Therefore, (19) has an optimal solution that is not a KKT point. A primal-dual interior-point method, such as LOQO, cannot solve this problem.

The penalty problem associated with (19) is

$$(20) \quad \begin{aligned} \min \quad & (x_1 - 2)^2 + x_2^2 + d^T \xi \\ \text{s.t.} \quad & (1 - x_1)^3 - x_2 + \xi_1 \geq 0 \\ & x_1 + \xi_2 \geq 0 \\ & x_2 + \xi_3 \geq 0 \\ & \xi_1, \xi_2, \xi_3 \geq 0. \end{aligned}$$

The optimality conditions of (20) are

$$\begin{aligned} 2(x_1 - 2) + 3(1 - x_1)^2 y_1 - y_2 &= 0 & y_1 \left((1 - x_1)^3 - x_2 + \xi_1 \right) &= 0 \\ 2x_2 + y_1 - y_3 &= 0 & y_2 (x_1 + \xi_2) &= 0 \\ y_1, y_2, y_3 &\geq 0 & y_3 (x_2 + \xi_3) &= 0 \\ d_1 - y_1 - \psi_1 &= 0 & \psi_1 \xi_1 &= 0 \\ d_2 - y_2 - \psi_2 &= 0 & \psi_2 \xi_2 &= 0 \\ d_3 - y_3 - \psi_3 &= 0 & \psi_3 \xi_3 &= 0 \\ \xi_1, \xi_2, \xi_3 &\geq 0. \end{aligned}$$

Provided $d_3 \geq d_1$, the penalty problem (20) has a KKT point for each value of d satisfying

$$\begin{aligned} (x_1, x_2) &= \left(1 + \frac{2}{1 + \sqrt{6d_1 + 1}}, 0 \right) \\ (y_1, y_2, y_3) &= (d_1, 0, d_1) \\ (\xi_1, \xi_2, \xi_3) &= \left(\left(\frac{2}{1 + \sqrt{6d_1 + 1}} \right)^3, 0, 0 \right) \\ (\psi_1, \psi_2, \psi_3) &= (0, d_2, d_3 - d_1). \end{aligned}$$

As $d \rightarrow \infty$, the primal variables approach the nonKKT optimum of (19) and $y_1, y_3 \rightarrow \infty$. Table 1 shows that our implementation was able to solve this problem, and the code increased the penalty parameters from 10^2 to 10^4 , stopping at $(x_1, x_2) = (1.00813, -2.69002 \times 10^{-7})$. Of the remaining problems with nonKKT optima, *DEGEN4* is identical to *HS13*, and the MPECs *ex9.2.2*, *qpec2*, *ralph1*, and *scholtes4* are each solved after several updates to the penalty parameter values, while LOQO failed on all of them.

6.2. Infeasibility Detection. As stated, we also tested the codes on the infeasible problems *DEGEN13f*, *DEGEN13fb*, and *DEGEN14f*. The convergent code was able to successfully enter “elastic mode” for infeasibility detection on these problems, while LOQO ran to its iteration limit without reporting a result. The behavior of LOQO on these problems is not surprising, since it is based on an infeasible interior-point method which only enforces feasibility at the optimal solution. Also, unlike linear programming and its use of theorems of the alternative and the homogeneous self-dual embedding, no readily available infeasibility indicators exist for nonlinear programming problems.

One interesting point about the infeasible problems relates to the choice of infeasibility tolerance in both codes. *DEGEN13f* and *DEGEN13fb* were obtained by modifying the constraints of *DEGEN12f* so that the minimum value of $\text{res}(x, \xi, u, \psi; 0)$ are 10^{-4} and 10^{-7} , respectively. Since the latter value is within the default infeasibility tolerance, both the proposed convergent algorithm and LOQO originally claimed to find an optimal solution to this problem. However, if $\hat{\epsilon}$ is decreased to 10^{-8} , the convergent algorithm enters elastic mode and detects infeasibility, while LOQO goes to its iteration limit.

6.3. Unbounded Penalty Problems. Constraint relaxation can lead to a problem becoming unbounded for some or all values of the penalty parameter. We have identified four such problems in our test suite: *HS3* which becomes unbounded for penalty parameter values of 1 or less, *HS24* which becomes unbounded for all penalty parameter values but the optimal solution of the original problem is a local minimum located by the convergent code, and *HS36* and *HS37* both of which become unbounded for all penalty parameter values and the convergent code cannot find a local minimum. The failure of the algorithm is not inconsistent with our theoretical results, since Assumption 2 states that the results apply when the primal iterates remain bounded. As can be seen from these four problems, assuming boundedness of the primal iterates is not a particularly strong assumption but can be an issue, and it can be remedied by related approaches such as the primal-dual penalty method outlined in [7].

Problem 3 of the Hock and Schittkowski test suite is

$$(21) \quad \begin{array}{ll} \min & x_2 + 0.00001(x_2 - x_1)^2 \\ \text{s.t.} & x_2 \geq 0. \end{array}$$

It is clear to see that the optimal solution for (21) is $(x_1, x_2) = (0, 0)$, with $y = 1$. The penalty problem for (21) is

$$(22) \quad \begin{array}{ll} \min & x_2 + 0.00001(x_2 - x_1)^2 + d\xi \\ \text{s.t.} & x_2 + \xi \geq 0 \\ & \xi \geq 0. \end{array}$$

The optimality conditions for a barrier problem associated with (22) are

$$(23) \quad \begin{aligned} -0.00002(x_2 - x_1) &= 0 \\ 1 + 0.00002(x_2 - x_1) - y &= 0 \\ d - y - \psi &= 0 \\ y(x_2 + \xi) &= \mu \\ \psi\xi &= \mu. \end{aligned}$$

The first condition in (23) means that $x_1 = x_2$. In fact, just this observation is enough to conclude that as long as $x_1 = x_2$ in (22), x_2 can be decreased arbitrarily for $d < 1$. Nevertheless, let us analyze what happens to the iterates for a fixed value of $\mu > 0$. The second condition implies that $y = 1$. The third condition ensures that $\psi = d - 1$, the fourth condition implies that $x_2 = \mu - \xi$, and the final condition ensures that $\xi = \frac{\mu}{d-1}$. If we set an initial value of the penalty parameter $d \leq 1$, the following occur:

- ψ approaches 0: For $d \leq 1$, ψ will need to take on nonpositive values. However, even if $\lambda \leq 0$, the updating formula (9) for ψ would ensure that ψ approaches 0 while remaining strictly positive.
- ξ becomes arbitrarily large: As ψ approaches 0 with $\mu > 0$, the last equation of (23) can only be satisfied with $\xi \rightarrow \infty$. This allows for $x_2 \rightarrow -\infty$.

Therefore, for $d \leq 1$, the penalty problem (22) associated with Problem 3 of the Hock and Schittkowski test suite (21) does not satisfy Assumption 2 that the primal iterates will remain bounded. As stated above, our default initial value for the penalty parameter is 100 in the implementation, so the numerical results shown in Table 1 reflect that the algorithm was able to solve this problem.

Problem 24 of the Hock and Schittkowski test suite is

$$(24) \quad \begin{aligned} \min \quad & \frac{(x_1-3)^2 x_2^3}{9} - x_2^3 \\ \text{s.t.} \quad & x_1 - \sqrt{3}x_2 \geq 0 \\ & 0 \leq x_1 + \sqrt{3}x_2 \leq 6 \\ & x_1, x_2 \geq 0. \end{aligned}$$

The optimal solution of this problem is $(x_1, x_2) = (3, \sqrt{3})$. The penalty problem associated with (24) can be expressed as

$$(25) \quad \begin{aligned} \min \quad & \frac{(x_1-3)^2 x_2^3}{9} - x_2^3 + d^T \xi \\ \text{s.t.} \quad & x_1 - \sqrt{3}x_2 + \xi_1 \geq 0 \\ & 6 - x_1 - \sqrt{3}x_2 + \xi_2 \geq 0 \\ & x_1 + \sqrt{3}x_2 + \xi_3 \geq 0 \\ & x_1 + \xi_4 \geq 0 \\ & x_2 + \xi_5 \geq 0 \\ & \xi \geq 0. \end{aligned}$$

Letting $x_1 = 3$, x_2 can get arbitrarily large. In this case, $\xi_3 = \xi_4 = \xi_5 = 0$, and ξ_1 and ξ_2 will increase at the same rate as x_2 . The penalty problem (25) is unbounded for any value of d since x_2^3 will increase faster than $d_1 \xi_1 + d_2 \xi_2$. The optimal solution of (24), however, is a local minimum for (25), so our solver locates this solution when starting from the initial value of (1, 0.5), provided in [18].

Problem 36 of the Hock and Schittkowski test suite becomes similarly unbounded. The model is

$$(26) \quad \begin{aligned} \min \quad & -x_1 x_2 x_3 \\ \text{s.t.} \quad & x_1 + 2x_2 + 2x_3 \leq 72 \\ & 0 \leq x_1 \leq 20 \\ & 0 \leq x_2 \leq 11 \\ & 0 \leq x_3 \leq 42. \end{aligned}$$

The optimal solution of this problem is $(x_1, x_2, x_3) = (20, 11, 15)$. The penalty problem associated with (26) can be expressed as

$$(27) \quad \begin{aligned} \min \quad & -x_1 x_2 x_3 + d^T \xi \\ \text{s.t.} \quad & 72 - x_1 - 2x_2 - 2x_3 + \xi_1 \geq 0 \\ & x_1 + \xi_2 \geq 0 \\ & 20 - x_1 + \xi_3 \geq 0 \\ & x_2 + \xi_4 \geq 0 \\ & 11 - x_2 + \xi_5 \geq 0 \\ & x_3 + \xi_6 \geq 0 \\ & 42 - x_3 + \xi_7 \geq 0 \\ & \xi \geq 0. \end{aligned}$$

For arbitrarily large values of x_1 , x_2 , and x_3 , we have that $\xi_2 = \xi_4 = \xi_6 = 0$ and

$$\begin{aligned} \xi_1 &= x_1 + 2x_2 + 2x_3 - 72 \\ \xi_3 &= x_1 - 20 \\ \xi_5 &= x_2 - 11 \\ \xi_7 &= x_3 - 42. \end{aligned}$$

It is clear to see that there are sufficiently large values of x_1 , x_2 , and x_3 , above which the objective function of (27) tends to $-\infty$. Thus, the penalty problem is unbounded for all values of the penalty parameters. However, if the values of the penalty parameters are sufficiently large, for example, 10^4 , the optimal solution of the original problem (26) is a local minimum of the penalty problem (27), and the proposed algorithm can locate this optimum when starting from a nearby solution such as $(x_1, x_2, x_3) = (10, 10, 10)$. Similar behavior is noted on Problem 37, which differs from Problem 36 by the constraint

$$x_1 + 2x_2 + 2x_3 \geq 0$$

which remains satisfied for large values of x_1 , x_2 , and x_3 .

6.4. Scaling Issues. As stated, we implemented the algorithm described in Section 2 without any modification. This also means that we did not scale the infeasibility measures. On the other hand, LOQO uses the following stopping criteria:

$$\begin{aligned} \|r(x) + w\| / (1.0 + \|Ax\|) &\leq 10^{-7} \\ \|\nabla f(x) - A^T y + w\| / (1.0 + \|\nabla f(x) - Hx\|) &\leq 10^{-7} \\ \|Yw\| / (1.0 + |f(x)|) &\leq 10^{-7}, \end{aligned}$$

where $w \in \mathbb{R}^m$ are the slack variables used by the infeasible interior-point method. Other interior-point solvers such as IPOPT [35] and KNITRO [27] use similar scalings. The lack of scaling can cause issues with convergence for some problems. In our numerical testing, we observed that Problem 99 of the Hock and Schittkowski test suite has all elements of $\nabla f(x)$ on the order of 10^8 and that the unscaled residual did not become sufficiently small for $\mu < 0.01$. However, when we rescaled the

model by dividing the objective function by 10^4 , we were able to solve the problem. Similarly, we needed to rescale *HS84* and *HS107* by 10^4 , and *HS67*, *HS114*, *ex9.2.2* by 10^2 .

6.5. Unbounded Sets of Optimal Lagrange Multipliers. Degenerate problems highlight the fact that without safeguards, a primal-dual interior-point code may encounter numerical problems while searching for the analytic center of the face of optimal Lagrange multipliers. If the optimal set is unbounded, then the dual iterates can approach infinity. This behavior is not observed on any of the problems in Table 4, either because the optimal set of Lagrange multipliers is not unbounded or the primal-dual iterate for LOQO attains the required accuracy level before the multipliers get too large. However, a modified version of Problem *DEGEN10* can be a potential source of issues:

$$\begin{aligned}
 (28) \quad & \min \quad 0 \\
 & \text{s.t.} \quad -(x_1 - 1)^2 + 2(x_2 - 1)^2 \geq 0 \\
 & \quad \quad 0.5(x_1 - 1)^2 + 0.5(x_2 - 1)^2 - 3(x_1 - 1)(x_2 - 1) \geq 0 \\
 & \quad \quad 2(x_1 - 1)^2 - (x_2 - 1)^2 \geq 0 \\
 & \quad \quad 0.5(x_1 - 1)^2 + 0.5(x_2 - 1)^2 + 3(x_1 - 1)(x_2 - 1) \geq 0
 \end{aligned}$$

To obtain this problem, we have removed the ℓ_∞ penalty term from *DEGEN10* and list it as *DEGEN10u* in Table 5. The Lagrange multipliers must satisfy the following conditions:

$$\begin{aligned}
 (29) \quad & (x_1 - 1)(-2y_1 + y_2 + 4y_3 + y_4) - (x_2 - 1)(3y_2 - 3y_4) = 0 \\
 & (x_2 - 1)(4y_1 + y_2 - 2y_3 + y_4) - (x_1 - 1)(3y_2 - 3y_4) = 0
 \end{aligned}$$

The only feasible solution $(x_1, x_2) = (1, 1)$ is also the optimal solution to this problem. At this point, both conditions in (29) reduce to $0 = 0$, that is, all four Lagrange multipliers can take on any nonnegative value. When the multipliers are all initialized to 1, LOQO can find the optimal solution in 17 iterations. However, when the multipliers are initialized to 100, LOQO runs into numerical difficulties and cannot sufficiently reduce complementarity and dual infeasibility to the desired tolerance within 500 iterations as the multipliers get larger. The convergent code, on the other hand, keeps the multipliers bounded and is able to solve the problem easily with either initialization, taking 30 iterations when all multipliers are initialized to 1 and 37 iterations when they are initialized to 100. In fact, even when the multipliers are initialized to 10^4 , the convergent code is able to locate the optimum in 34 iterations. In all three initialization schemes, the starting penalty parameter value of 100 is sufficient.

Similar behavior is reported on MPECs in [3] for LOQO. MPECs are a class of problems that are shown to have unbounded sets of optimal Lagrange multipliers, and LOQO solves most of the problems in the MACMPEC [20] test suite without a safeguard. In fact, of the problems that are not solved by LOQO, many are due to other reasons, such as having nonKKT optima as mentioned before. In practice, for many small-to-medium sized MPECs, LOQO is able to locate a primal-dual solution satisfying the convergence criteria within a reasonable number of iterations before the Lagrange multipliers become too large and cause numerical issues. For example, the problem *scholtes5* can tolerate very large initial values for the multipliers. The

model is

$$\begin{aligned} \min \quad & (x_1 - 1)^2 + (x_2 - 2)^2 + (x_3 + 1)^2 \\ \text{s.t.} \quad & -x_1x_3 \geq 0 \\ & -x_2x_3 \geq 0 \\ & x_1, x_2, x_3 \geq 0. \end{aligned}$$

The constraints of the problem are NLP representations of the equilibrium conditions $0 \leq x_1 \perp x_3 \geq 0$ and $0 \leq x_2 \perp x_3 \geq 0$. The optimality conditions for this problem are

$$\begin{aligned} 2(x_1 - 1) + x_3y_1 - y_3 &= 0 & y_1(-x_1x_3) &= 0 \\ 2(x_2 - 2) + x_3y_2 - y_4 &= 0 & y_2(-x_2x_3) &= 0 \\ 2(x_3 + 1) + x_1y_1 + x_2y_2 - y_5 &= 0 & y_3x_1 &= 0 \\ y_1, y_2, y_3, y_4, y_5 &\geq 0 & y_4x_2 &= 0 \\ & & y_5x_3 &= 0. \end{aligned}$$

At the optimal solution $(x_1, x_2, x_3) = (1, 2, 0)$, we have that $y_3 = y_4 = 0$, and the optimality conditions reduce to

$$\begin{aligned} y_5 - y_1 - 2y_2 &= 2 \\ y_1, y_2, y_5 &\geq 0. \end{aligned}$$

It is clear that the set of optimal Lagrange multipliers is unbounded, but starting LOQO at initial values up to 10^{25} for the Lagrange multipliers does not prevent it from reaching the optimal solution. In fact, after initializing the multipliers to 10^{25} , LOQO is able to find the optimal solution within 208 iterations and the final values of the multipliers are $(y_1, y_2, y_3, y_4, y_5) = (3.15E + 026, 1.35E + 025, 1.44E - 009, 7.97E - 009, 3.42E + 026)$. Therefore, one can conclude that while unbounded Lagrange multipliers may cause theoretical difficulties, it is not always the case that the dual iterates will approach infinity and cause numerical issues before finding a KKT point and a set of corresponding Lagrange multipliers. This explains some of the success of LOQO without any safeguards on the MPECs from the MacMPEC test suite.

In our testing, we also found that larger problems were more susceptible to failure due to the unboundedness of Lagrange multipliers. We modified the problem *egout* from the MIPLIB test suite [10] by replacing the binary requirement on each variable x_j with the conditions

$$x_j \geq 0, \quad 1 - x_j \geq 0, \quad -x_j(1 - x_j) \geq 0,$$

converting it into an MPEC. Starting with the default multiplier initialization of 1, LOQO fails on this problem, reaching the iteration limit as the multipliers grow to above 10^{10} . The convergent code, on the other hand, reaches a solution in 30 iterations, keeping all the Lagrange multipliers bounded by the penalty parameter values of 100. The model for *egout* has 141 variables and 153 constraints after preprocessing by AMPL. As would be expected, this is not the optimal solution for the mixed-integer linear programming problem, but a quick way to obtain a good feasible solution that can be helpful to reduce the number of linear subproblems that need to be solved. We similarly modified the mixed-integer nonlinear programming problem *indextrack* from the MacMINLP test suite [19], and the convergent code solved this problem in 37 iterations, while LOQO reached its iteration limit. It should be noted for both *egout* and *indextrack* that it is very hard for either algorithm to approach or find a KKT point from the default initializations of the primal and

dual variables. For many of the other problems in each test suite, both algorithms approach a local minimizer of infeasibility. However, as shown in [5], the MPEC reformulation, combined with a penalty approach, can warmstart from the solution of the continuous relaxation and successfully locate an integer feasible solution for most problems.

6.6. Barrier Parameter Updates. Examination of the tables shows that the convergent algorithm can take a significantly larger number of iterations than LOQO on some problems. In general, this phenomenon is a result of the barrier parameter: the initial choice of μ , the requirement for (approximately) solving the barrier problem for a fixed μ before reducing μ , the choice of β for updating μ , and the level of convergence ϵ_μ that is required of each barrier problem. Interior-point linear programming codes reduce μ at every iteration based on the current iterate, but the special structure of these problems allows for this in a way that can be shown to be convergent [32]. Unfortunately, no such similar results are known for general nonconvex NLPs. LOQO changes μ at every iteration using the following formula given in [33]:

$$\mu = \beta \min \left((1 - \eta) \frac{1 - \iota}{\iota}, 2 \right)^3 \frac{w^T y}{m},$$

where $w \in \mathbb{R}^m$ are the slack variables for the constraints $r(x) \geq 0$, $0 < \eta < 1$ is a parameter which defaults to 0.95, and ι is a measure of uniformity computed as

$$\iota = \min_i \frac{w_i y_i}{w^T y / m}.$$

As such, when the complementarity products are decreasing at a uniform rate, the fraction $\frac{1-\iota}{\iota}$ will be small. When they are far from uniformity, a large μ will promote uniformity for the next iteration. Using this updating formula, μ can actually increase from one iteration to the next, especially in the earlier iterations. Allowing for an increase has proved efficient in numerical testing, but it can cause problems for proving convergence.

Let us now examine the following problems for which the convergent code takes more than 1000 iterations to solve: *HS25*, *HS89*, *HS99*, *HS103*, *HS109*, and *HS116*. Among these, *HS25* can be solved in 46 iterations if β is changed to 0.5, *HS89* and *116* can be solved in 63 and 195 iterations, respectively, if μ is initialized to 1, and *HS99* can be solved in 93 iterations by using $\beta\sigma$ as the initial value of the barrier parameter. The problems *HS103* and *HS109* perform most of their iterations in the first barrier problem, trying to get the sufficient level of accuracy. If the barrier parameter is updated dynamically at each iteration, as in LOQO, the solutions can be attained quickly.

6.7. Fast Local Convergence. In Tables 1-5, we also provide numerical results for a variant of the convergent algorithm with the barrier parameter updating rule (15) to ensure fast local convergence. As can be observed from these tables, there are a few cases where such modifications are beneficial, but for many of the problems, we see either no difference or a slight worsening of the iteration counts. There are several reasons for this behavior. First, the final value of μ is generally significantly smaller (10^{-12}) with fast local convergence than without (10^{-8}) for most problems. Several additional iterations may be necessary to find an accurate enough solution. In fact, on 69 of the Hock and Schittkowski problems, the fast local convergence results in Tables 1-3 are 1-3 iterations worse.

Second, it is crucial to change into this updating mode at the right time. From the updating formula, we can easily determine that we will start using this formula as soon as $\sigma < \frac{\beta}{\hat{\beta}}$, but depending on the problem, we may not be close enough to the solution yet. If we reduce μ too fast, the algorithm may stall. One example is *HS49*. The code takes 990 iterations for $\hat{\beta} = 1$ and 20 iterations for $\hat{\beta} = 10$. In the former case, it starts reducing μ to $\hat{\beta}\sigma^2$ when $\mu < 10^{-2}$, whereas in the latter case, it does so when $\mu < 10^{-3}$. We found that a value of $\hat{\beta} = 1000$ gave the best numerical performance over all the problems in our tests.

In practice, it seems that the fast local convergence formula is rarely useful and can even hurt algorithm performance with a naïve choice of the parameter $\hat{\beta}$. It significantly improves performance on only 5 problems from the Hock and Schittkowski set: *HS90*, *HS92*, *HS98*, *HS99*, and *HS103*. On the other hand, it greatly worsens the code's performance on *HS91*, *HS97*, and *HS116*.

7. CONCLUSION.

In this paper, we have proved convergence of an interior-point method under the mild assumptions of twice continuous differentiability and the boundedness of the primal iterates. By providing simple bounds on the variables, we can further relax these assumptions. Therefore, incorporating the ℓ_1 penalty approach has allowed us to strengthen the convergence results of [12].

Note that we started by attempting to solve a general problem given by (1). As mentioned, this problem had drawbacks, including the use of equality constraints. Again, the use of the ℓ_1 penalty approach allowed us to incorporate all of these constraints into the same framework while allowing for a strictly feasible interior for the feasible region and bounded Lagrange multipliers. The convergence results provided apply not only to NLP but to mathematical programs with equilibrium constraints (MPECs), a wider class of problems whose optimal Lagrange multipliers are always unbounded.

Numerical testing confirmed that the proposed algorithm is able to solve or appropriately identify infeasibility for all problems that satisfy the assumptions stated in Sections 3 and 4. The algorithm was able to solve degenerate problems, as well. Given the theoretical analysis, these results are to be expected, but we further analyzed the computational performance and compared it to the highly-efficient code LOQO to make the following observations:

- Certain types of “hard” problems, including those with nonKKT optima, cannot be solved without a safeguard such as constraint relaxation. Similarly, constraint relaxation and elastic mode are essential for infeasibility identification in an interior-point framework. These contributions toward the robustness of the code constitute the biggest benefit of using the convergent approach.
- There are times when constraint relaxation can hurt the robustness of the algorithm, as well. This is especially true if relaxation leads to the penalty problems becoming unbounded for some or all values of the penalty parameters. While we assumed that the primal iterates are to remain bounded, this phenomenon is not uncommon.
- Numerical issues can arise even within the context of a convergent algorithm. Proper scaling of stopping criteria is crucial for good computational

performance, but is generally not considered within algorithm descriptions for theoretical results.

- Unbounded sets of optimal Lagrange multipliers, while theoretically an issue, do not seem to cause much problem for interior-point methods without safeguards on many instances. In general, the algorithm is able to attain the desired accuracy level and conclude the solution is optimal before encountering numerical problems. This is true of the degenerate problems considered here and many MPECs reported in [3]. Nevertheless, most difficult NLPs we have encountered are relatively small, and unbounded sets of optimal multipliers can cause more problems as the problem size grows.
- For updating the barrier parameter, adaptive approaches invoked at each iteration work better than approximately solving each and every barrier problem. In most instances where the convergent code takes a significantly larger number of iterations, it is due to numerical issues involved in approximately solving the problem for one particular value of μ . LOQO, on the other hand, generally will not spend more than a single iteration on any value of μ .
- Changing the barrier parameter update formula for fast local convergence does not seem to significantly improve the performance of the code. In fact, in most cases, the performance stays the same or slightly worsens. While a naïve approach as the one described here is enough to prove fast local convergence, numerical issues, including the identification of a neighborhood to properly invoke local mode and the inability to attain desired accuracy levels for very small values of μ without scaling, prevent this benefit from being realized. Reducing μ linearly is not optimal, but for problems with numerical difficulties, it can be the best approach.

In general, setting values for algorithmic constants that are not problem dependent can be quite challenging in practice and have a major impact on computational performance. One of the main purposes of this paper is to illustrate this impact by comparing the implementation of a rigorously designed algorithm to one that is built for efficiency. LOQO tries hard to adjust all constants to data and update them frequently, and while doing so leads to increased efficiency, it does not create desirable properties such as monotonically decreasing barrier parameter values.

It is clear from the numerical results and these observations that a convergent code is useful for solving very difficult problems with particular characteristics, but it will never be as efficient as codes such as LOQO, IPOPT, and KNITRO. Stricter requirements that enforce convergence will hurt performance on problems without any issues. As such, a convergent code should only be invoked when it appears that the efficient code is beginning to experience difficulty. In [7], where we presented a primal-dual penalty framework without theoretical analysis and received similar efficiency results, we proposed the use of a hybrid approach, that would default to the code of LOQO and only invoke the safeguards of the penalty framework as necessary. This hybrid approach resulted in an increase in the number of problems solved within a comparable number of total iterations over the CUTER test set. A similar hybrid approach is recommended here. The switch into the convergent algorithm can occur after a predetermined number of iterations or by defining switching conditions. It is also important to note that unlike other hybrid algorithms, including the one described in [22] which switches into steepest descent

under certain conditions, all modes should be Newton-based to retain the desirable convergence properties.

REFERENCES

- [1] M. Anitescu, P. Tseng, and S.J. Wright. Elastic-mode algorithms for mathematical programs with equilibrium constraints. *Mathematical Programming*, 110(2):337–371, 2007.
- [2] P. Armand. A quasi-newton penalty barrier method for convex minimization problems. *Computational Optimization and Applications*, 26:5–34, 2003.
- [3] H. Y. Benson, A. Sen, D.F. Shanno, and R. J. Vanderbei. Interior point algorithms, penalty methods and equilibrium problems. *Computational Optimization and Applications*, 34(2):155–182, June 2006.
- [4] H.Y. Benson. Numerical testing results for the primal-dual penalty approach. <http://www.pages.drexel.edu/~hvb22/penaltyweb>.
- [5] H.Y. Benson. Mixed-integer nonlinear programming using interior-point methods. Technical report, Submitted to *Optimization Methods and Software*, November 2007.
- [6] H.Y. Benson, A. Sen, and D.F. Shanno. AMPL models and numerical testing results for a globally convergent interior-point method. <http://www.pages.drexel.edu/~hvb22/Conlib>.
- [7] H.Y. Benson and D.F. Shanno. Interior-point methods for nonconvex nonlinear programming: Regularization and warmstarts. *Computational Optimization and Applications*, 40(2):143–189, June 2008.
- [8] 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.
- [9] 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.
- [10] R. E. Bixby, S. Ceria, C. M. McZeal, and M. W. P Savelsbergh. An updated mixed integer programming library: MIPLIB 3.0. *Optima*, (58):12–15, June 1998.
- [11] L. Chen. Interior-point penalty methods for nonlinear programming. PhD. dissertation. Dept. of Industrial Engineering and Operations Research, Columbia University, New York, NY, USA, 2008.
- [12] L. Chen and D. Goldfarb. Interior-point ℓ_2 penalty methods for nonlinear programming with strong global convergence properties. *Mathematical Programming*, 108:1–36, 2006.
- [13] L. Chen and D. Goldfarb. On the fast local convergence of interior-point ℓ_2 penalty methods for nonlinear programming. Technical report, Working Paper, July 2006.
- [14] A.R. Conn, N. Gould, and Ph.L. Toint. Constrained and unconstrained testing environment. <http://www.dci.clrc.ac.uk/Activity.asp?CUTE>.
- [15] 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.
- [16] N.I.M. Gould, D. Orban, and Ph.L. Toint. An interior-point ℓ_1 -penalty method for nonlinear optimization. Technical Report RAL-TR-2003-022, Rutherford Appleton Laboratory Chilton, Oxfordshire, UK, November 2003.
- [17] S.P. Han and O.L. Mangasarian. Exact penalty functions for nonlinear programming. *Mathematical Programming*, 17(3):251–269, 1979.
- [18] W. Hock and K. Schittkowski. *Test examples for nonlinear programming codes*. Lecture Notes in Economics and Mathematical Systems 187. Springer Verlag, Heidelberg, 1981.
- [19] S. Leyffer. MacMINLP : ampl collection of mixed integer nonlinear programs. <http://www-unix.mcs.anl.gov/~leyffer/MacMINLP>.
- [20] S. Leyffer. MacMPEC: ampl collection of mathematical programs with equilibrium constraints. <http://www-unix.mcs.anl.gov/~leyffer/MacMPEC>.
- [21] 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.
- [22] X. Liu and J. Sun. Global convergence analysis of line search interior-point methods for nonlinear programming without regularity assumptions. *Journal of Optimization Theory and Applications*, 125(3):609–628, 2005.

- [23] D.Q. Mayne and E. Polak. Feasible directions algorithms for optimization problems with equality and inequality constraints. *Mathematical Programming*, 11:67–80, 1976.
- [24] E.M.E. Mostafa, L.M. Vicente, and S.J. Wright. Numerical behavior of a stabilized SQP method for degenerate NLP problems. In *Global Optimization and Constraint Satisfaction, Lecture Notes in Computer Science*, volume 2861, pages 123–141. Springer Berlin/Heidelberg, 2003.
- [25] B.A. Murtagh and M.A. Saunders. MINOS 5.4 user’s guide. Technical Report SOL 83-20R, Systems Optimization Laboratory, Stanford University, 1983. Revised February 1995.
- [26] J. Nocedal, J.L. Morales, R. Waltz, G. Liu, and J.P. Goux. Assessing the potential of interior-point methods for nonlinear optimization. In *Large-Scale PDE-Constrained Optimization, Lecture Notes in Computational Science and Engineering*, volume 30, pages 167–183, 2003.
- [27] J. Nocedal and R. A. Waltz. Knitro 2.0 user’s manual. Technical Report OTC 02-2002, Optimization Technology Center, Northwestern University, January 2002.
- [28] Arvind U. Raghunathan and Lorenz T. Biegler. Mathematical programs with equilibrium constraints (MPECs) in process engineering. *Computers and Chemical Engineering*, 27:1381–1392, 2003.
- [29] A.L. Tits, A. Wächter, S. Bakhtiari, T.J. Urban, and C.T. Lawrence. A primal-dual interior-point method for nonlinear programming with strong global and local convergence properties. *SIAM Journal on Optimization*, 14(1):173–199, 2003.
- [30] R.J. Vanderbei. AMPL models. <http://orfe.princeton.edu/~rvdb/ampl/nlmodels>.
- [31] R.J. Vanderbei. Symmetric quasi-definite matrices. *SIAM Journal on Optimization*, 5(1):100–113, 1995.
- [32] R.J. Vanderbei. *Linear Programming: Foundations and Extensions*. Kluwer Academic Publishers, 1997.
- [33] R.J. Vanderbei and D.F. Shanno. An interior-point algorithm for nonconvex nonlinear programming. *Computational Optimization and Applications*, 13:231–252, 1999.
- [34] 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.
- [35] 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.
- [36] S.J. Wright and D. Orban. Properties of the log-barrier function on degenerate nonlinear programs. *Mathematics of Operations Research*, 27(3):585–613, 2002.

HANDE Y. BENSON, DREXEL UNIVERSITY, PHILADELPHIA, PA

ARUN SEN, NERA CORPORATION, NEW YORK, NY

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

Problem	LOQO		Convergent Algorithm			Fast Local Updates		
	Iter	$f(x^*)$	Iter	i	$f(x^*)$	Iter	i	$f(x^*)$
HS1	30	4.3537E-011	39	1	1.6040E-015	39	1	6.4193E-015
HS2	19	4.9412E+000	29	1	4.9412E+000	29	1	4.9412E+000
HS3	11	6.0949E-009	10	1	9.8990E-008	10	1	1.9798E-007
HS4	8	2.6667E+000	14	1	2.6667E+000	14	1	2.6667E+000
HS5	10	-1.9132E+000	13	1	-1.9132E+000	13	1	-1.9132E+000
HS6	10	1.0380E-017	75	1	5.5331E-027	76	1	2.5406E-027
HS7	12	-1.7321E+000	23	1	-1.7321E+000	23	1	-1.7321E+000
HS8	8	-1.0000E+000	11	1	-1.0000E+000	11	1	-1.0000E+000
HS9	8	-5.0000E-001	10	1	-5.0000E-001	10	1	-5.0000E-001
HS10	15	-1.0000E+000	26	1	-1.0000E+000	26	1	-1.0000E+000
HS11	12	-8.4985E+000	13	1	-8.4985E+000	13	1	-8.4985E+000
HS12	14	-3.0000E+001	11	1	-3.0000E+001	11	1	-3.0000E+001
HS13		(IL)	105	3	9.8380E-001	102	3	9.8380E-001
HS14	11	1.3935E+000	13	1	1.3935E+000	14	1	1.3935E+000
HS15	31	3.0650E+002	62	3	3.0650E+002	65	3	3.0650E+002
HS16	18	2.5000E-001	29	1	2.5000E-001	30	1	2.5000E-001
HS17	29	1.0000E+000	28	1	1.0000E+000	29	1	1.0000E+000
HS18	15	5.0000E+000	18	1	5.0000E+000	19	1	5.0000E+000
HS19	18	-6.9618E+003	259	3	-6.9618E+003	261	3	-6.9618E+003
HS20	24	4.0199E+001	61	2	4.0199E+001	64	2	4.0199E+001
HS21	11	-9.9960E+001	15	1	-9.9960E+001	16	1	-9.9960E+001
HS22	9	1.0000E+000	15	1	1.0000E+000	15	1	1.0000E+000
HS23	14	2.0000E+000	21	1	2.0000E+000	22	1	2.0000E+000
HS24	24	-4.6765E+001	14	1	-4.6765E+001	14	1	-4.6765E+001
HS25	22	7.4305E-008	4284	1	1.3579E-012	4284	1	1.9555E-014
HS26	14	1.1439E-009	16	1	1.1980E-014	17	1	1.2283E-014
HS27	15	4.0000E-002	17	1	4.0000E-002	17	1	4.0000E-002
HS28	2	1.9906E-019	10	1	7.2230E-029	10	1	5.0006E-029
HS29	13	-2.2627E+001	24	1	-2.2627E+001	24	1	-2.2627E+001
HS30	9	1.0000E+000	12	1	1.0000E+000	13	1	1.0000E+000
HS31	15	6.0000E+000	15	1	6.0000E+000	16	1	6.0000E+000
HS32	23	1.0000E+000	21	1	1.0000E+000	22	1	1.0000E+000
HS33	11	-4.5858E+000	29	1	2.0000E+000	30	1	2.0000E+000
HS34	15	-8.3403E-001	16	1	-8.3403E-001	17	1	-8.3403E-001
HS35	10	1.1111E-001	16	1	1.1111E-001	17	1	1.1111E-001
HS36	16	-3.3000E+003			(IL)			(IL)
HS37	11	-3.4560E+003			(IL)			(IL)
HS38	12	7.8690E+000	43	1	2.3171E-019	43	1	5.7929E-020
HS39	8	-1.0000E+000	10	1	-1.0000E+000	11	1	-1.0000E+000
HS40	8	-2.5000E-001	11	1	-2.5000E-001	12	1	-2.5000E-001
HS41	16	1.9259E+000	18	1	1.9259E+000	19	1	1.9259E+000

TABLE 1. Numerical performance of LOQO, the proposed algorithm, and the proposed algorithm with fast local convergence modifications on the Hock and Schittkowski test suite [18]. *Iter* is the number of iterations, i is the number of penalty parameter updates, and $f(x^*)$ is the final objective function value reported by the algorithm. (IL) indicates that the algorithm reached its iteration limit.

Problem	LOQO		Convergent Algorithm			Fast Local Updates		
	Iter	$f(x^*)$	Iter	i	$f(x^*)$	Iter	i	$f(x^*)$
HS42	9	1.3858E+001	13	1	1.3858E+001	14	1	1.3858E+001
HS43	17	-4.4000E+001	14	1	-4.4000E+001	16	1	-4.4000E+001
HS44	16	-1.5000E+001	18	1	-1.5000E+001	19	1	-1.5000E+001
HS45	23	1.0000E+000	19	1	1.0000E+000	20	1	1.0000E+000
HS46	18	2.8053E-010	31	1	9.9696E-016	31	1	3.3060E-015
HS47	21	3.7027E-011	29	1	9.4427E-015	30	1	8.4540E-015
HS48	2	6.6200E-020	10	1	5.0536E-031	11	1	2.3296E-030
HS49	20	5.3563E-009	19	1	1.1408E-011	20	1	4.1230E-011
HS50	16	7.6569E-015	14	1	1.5098E-028	15	1	8.4852E-029
HS51	2	1.8692E-020	10	1	1.1710E-030	11	1	4.2525E-030
HS52	2	5.3266E+000	12	1	5.3266E+000	13	1	5.3266E+000
HS53	11	4.0930E+000	12	1	4.0930E+000	13	1	4.0930E+000
HS54	12	1.9286E-001	15	1	1.9286E-001	16	1	1.9286E-001
HS55	11	6.3333E+000	16	1	6.3333E+000	17	1	6.3333E+000
HS56	12	-3.4560E+000	14	1	-3.4560E+000	15	1	-3.4560E+000
HS57	16	2.8460E-002	25	1	3.0648E-002	26	1	3.0648E-002
HS59	23	-7.8028E+000	295	1	-6.7495E+000	295	1	-6.7495E+000
HS60	9	3.2568E-002	15	1	3.2568E-002	16	1	3.2568E-002
HS61	10	-1.4365E+002	17	1	-1.4365E+002	18	1	-1.4365E+002
HS62	13	-2.6273E+004	53	3	-2.6273E+004	56	3	-2.6273E+004
HS63	8	9.6172E+002	40	1	9.6172E+002	41	1	9.6172E+002
HS64	26	6.2998E+003	108	3	6.2998E+003	109	3	6.2998E+003
HS65	19	9.5353E-001	16	1	9.5353E-001	17	1	9.5353E-001
HS66	15	5.1816E-001	17	1	5.1816E-001	18	1	5.1816E-001
HS67*	18	-1.1332E+001	234	1	-1.1620E+001	234	1	-1.1620E+001
HS70	21	9.4020E-003	252	1	1.4321E-001	253	1	1.4321E-001
HS71	13	1.7014E+001	62	1	1.7014E+001	63	1	1.7014E+001
HS72	20	7.2768E+002	63	4	7.2768E+002	65	4	7.2768E+002
HS73	19	2.9894E+001	16	1	2.9894E+001	17	1	2.9894E+001
HS74	16	5.1265E+003	34	1	5.1265E+003	35	1	5.1265E+003
HS75	18	5.1744E+003	97	3	5.1744E+003	94	3	5.1744E+003
HS76	11	-4.6818E+000	15	1	-4.6818E+000	16	1	-4.6818E+000
HS77	12	2.4151E-001	19	1	2.4151E-001	20	1	2.4151E-001
HS78	7	-2.9197E+000	40	1	-2.9197E+000	41	1	-2.9197E+000
HS79	7	7.8777E-002	11	1	7.8777E-002	12	1	7.8777E-002
HS80	10	5.3950E-002	488	1	5.3950E-002	489	1	5.3950E-002
HS81	16	5.3950E-002	142	1	5.3950E-002	143	1	5.3950E-002
HS83	14	-3.0666E+004	33	2	-3.0666E+004	34	2	-3.0666E+004
HS84*	47	-5.2803E+002	16	1	-5.2803E+002	17	1	-5.2803E+002
HS85	30	-1.9052E+000	26	1	-1.9052E+000	26	1	-1.9052E+000
HS86	13	-3.2349E+001	18	1	-3.2349E+001	19	1	-3.2349E+001

TABLE 2. Numerical performance of LOQO, the proposed algorithm, and the proposed algorithm with fast local convergence modifications on the Hock and Schittkowski test suite [18]. *Iter* is the number of iterations, *i* is the number of penalty parameter updates, and $f(x^*)$ is the final objective function value reported by the algorithm. The objective functions of problems marked with a “*” were scaled.

Problem	LOQO		Convergent Algorithm			Fast Local Updates		
	Iter	$f(x^*)$	Iter	i	$f(x^*)$	Iter	i	$f(x^*)$
HS87	25	8.8276E+003	48	1	8.8276E+003	48	1	8.8276E+003
HS88	27	1.3627E+000	64	3	1.3627E+000	61	3	1.3627E+000
HS89	27	1.3627E+000	1353	3	1.3627E+000	1353	3	1.3627E+000
HS90	28	1.3627E+000	471	3	1.3627E+000	92	3	1.3627E+000
HS91	28	1.3627E+000	777	3	1.3627E+000	1356	3	1.3627E+000
HS92	22	1.3627E+000	864	3	1.3627E+000	304	3	1.3627E+000
HS93	13	1.3508E+002	13	1	1.3508E+002	14	1	1.3508E+002
HS95	16	1.5620E-002	172	1	1.5620E-002	173	1	1.5620E-002
HS96	19	1.5620E-002	134	1	1.5620E-002	135	1	1.5620E-002
HS97	18	4.0712E+000	420	2	4.0712E+000	1099	2	4.0712E+000
HS98	45	3.1358E+000	842	2	4.0712E+000	667	2	4.0712E+000
HS99*	20	-8.3108E+004	3206	1	-8.3108E+004	2812	1	-8.3108E+004
HS100	13	6.8063E+002	16	1	6.8063E+002	17	1	6.8063E+002
HS101	37	1.8098E+003	216	3	1.8098E+003	219	3	1.8098E+003
HS102	55	9.1188E+002	231	3	9.1188E+002	230	3	9.1188E+002
HS103	47	5.4367E+002	1380	3	5.4367E+002	309	3	5.4367E+002
HS104	14	3.9512E+000	20	1	3.9512E+000	21	1	3.9512E+000
HS105	17	1.1364E+003	24	1	1.1364E+003	25	1	1.1364E+003
HS106	25	7.0492E+003	108	3	7.0492E+003	111	3	7.0492E+003
HS107*	21	5.0550E-001	429	1	5.0550E-001	429	1	5.0550E-001
HS108	20	-8.6603E-001	20	1	-6.7498E-001	21	1	-6.7498E-001
HS109	33	5.3269E+003	1174	1	5.3269E+003	1175	1	5.3269E+003
HS110	8	-4.5778E+001	18	1	-4.5778E+001	18	1	-4.5778E+001
HS111	14	-4.7761E+001	41	1	-4.7761E+001	41	1	-4.7761E+001
HS112	17	-4.7761E+001	32	1	-4.7761E+001	27	1	-4.7761E+001
HS113	19	2.4306E+001	15	1	2.4306E+001	16	1	2.4306E+001
HS114*	26	-1.7688E+001	106	1	-1.7688E+001	106	1	-1.7688E+001
HS116	24	9.7588E+001	8668	3	9.7588E+001	8995	3	9.7588E+001
HS117	21	3.2349E+001	25	1	3.2349E+001	26	1	3.2349E+001
HS118	15	6.6482E+002	21	1	6.6482E+002	22	1	6.6482E+002
HS119	29	2.4490E+002	22	1	2.4490E+002	22	1	2.4490E+002

TABLE 3. Numerical performance of LOQO, the proposed algorithm, and the proposed algorithm with fast local convergence modifications on the Hock and Schittkowsky test suite [18]. *Iter* is the number of iterations, *i* is the number of penalty parameter updates, and $f(x^*)$ is the final objective function value reported by the algorithm. (IL) indicates that the algorithm reached its iteration limit. The objective functions of problems marked with “*” were scaled.

Problem	LOQO		Convergent Algorithm			Fast Local Updates		
	Iter	$f(x^*)$	Iter	i	$f(x^*)$	Iter	i	$f(x^*)$
DEGEN01	19	2.4306E+001	15	1	2.4306E+001	16	1	2.4306E+001
DEGEN02	18	5.7373E-010	42	1	2.1089E-002	44	1	2.1089E-002
DEGEN03	18	-4.4000E+001	16	1	-4.4000E+001	17	1	-4.4000E+001
DEGEN04		(IL)	105	3	9.8380E-001	102	3	9.8380E-001
DEGEN05	13	6.8063E+002	15	1	6.8063E+002	17	1	6.8063E+002
DEGEN06	9	6.8104E-009	13	1	1.9981E-007	13	1	8.3531E-007
DEGEN07	11	1.8000E+000	15	1	1.8000E+000	16	1	1.8000E+000
DEGEN08	23	1.0000E+000	21	1	1.0000E+000	22	1	1.0000E+000
DEGEN09	12	-2.5000E-001	19	1	-2.5000E-001	20	1	-2.5000E-001
DEGEN10	64	3.6362E-012	7560	1	3.9900E-007	6097	1	2.0429E-008
DEGEN11	12	-4.4000E+001	23	1	-4.4000E+001	25	1	-4.4000E+001
DEGEN12	9	-1.9662E-010	19	1	-2.3178E-010	19	1	-9.2829E-010
DEGEN01f	12	0.0000E+000	26	1	0.0000E+000	26	1	0.0000E+000
DEGEN02f	8	0.0000E+000	10966	1	0.0000E+000	10968	1	0.0000E+000
DEGEN03f	8	0.0000E+000	12	1	0.0000E+000	12	1	0.0000E+000
DEGEN04f	20	0.0000E+000	20	1	0.0000E+000	20	1	0.0000E+000
DEGEN05f	12	0.0000E+000	15	1	0.0000E+000	15	1	0.0000E+000
DEGEN06f	8	0.0000E+000	12	1	0.0000E+000	11	1	0.0000E+000
DEGEN07f	12	0.0000E+000	13	1	0.0000E+000	13	1	0.0000E+000
DEGEN08f	11	0.0000E+000	16	1	0.0000E+000	17	1	0.0000E+000
DEGEN09f	11	0.0000E+000	18	1	0.0000E+000	19	1	0.0000E+000
DEGEN10f	11	0.0000E+000	154	1	0.0000E+000	143	1	0.0000E+000
DEGEN11f	13	0.0000E+000	47	1	0.0000E+000	48	1	0.0000E+000
DEGEN12f	8	0.0000E+000	19	1	0.0000E+000	19	1	0.0000E+000
DEGEN13f		(IL)			(INF)			(INF)
DEGEN13fb**		(IL)			(INF)			(INF)
DEGEN14f		(IL)			(INF)			(INF)

TABLE 4. Numerical performance of the proposed algorithm and LOQO on the degenerate problems from [24]. *Iter* is the number of iterations, *i* is the number of penalty parameter updates, and $f(x^*)$ is the final objective function value reported by the algorithm. (INF) indicates that the algorithm identified the problem as infeasible, and (IL) indicates that the algorithm reached its iteration limit. The infeasibility tolerance was reduced to 10^{-8} for the problem marked with “**”.

Problem	LOQO		Convergent Algorithm			Fast Local Updates		
	Iter	$f(x^*)$	Iter	i	$f(x^*)$	Iter	i	$f(x^*)$
DEGEN10u†		(IL)	30	1	0.0000E+000	52	1	0.0000E+000
egout		(IL)	30	1	8.5083E+002	30	1	8.5083E+002
ex9.2.2*		(IL)	4842	1	9.9998E-001	4841	1	1.0000E+000
indextrack		(IL)	37	1	-1.7727E-004	45	1	-1.7729E-004
qpec2		(IL)	120	3	4.4992E+001	131	3	4.4992E+001
ralph1		(IL)	88	2	-5.0030E-004	89	2	-5.0002E-004
scholtes4		(IL)	300	3	-2.0020E-004	177	3	-2.0002E-004

TABLE 5. Numerical performance of the proposed algorithm and LOQO on additional problems. (IL) indicates that the algorithm reached its iteration limit. The objective functions of problems marked with “*” were scaled. † indicates that the multiplier initialization was changed.