RC24736 (W0901-097) January 26, 2009
Mathematics

IBM Research Report

An Interior-Point Algorithm for Large-Scale Nonlinear Optimization with Inexact Step Computations

Frank E. Curtis
Courant Institute of Mathematical Sciences
New York University
New York, NY
USA

Olaf Schenk
Department of Computer Science
University of Basel
Basel, Switzerland

Andreas Wächter
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598
USA

Research Division
Almaden - Austin - Beijing - Cambridge - Haifa - India - T. J. Watson - Tokyo - Zurich

AN INTERIOR-POINT ALGORITHM FOR LARGE-SCALE NONLINEAR OPTIMIZATION WITH INEXACT STEP COMPUTATIONS

FRANK E. CURTIS*, OLAF SCHENK†, AND ANDREAS WÄCHTER‡

Abstract. We present a line-search interior-point algorithm for large-scale continuous optimization. The algorithm is matrix-free in that it does not require the factorization of derivative matrices and instead uses iterative linear system solvers. Inexact step computations are supported in order to save computational expense during each iteration. The algorithm is an interior-point approach derived from the inexact Newton method for equality constrained optimization described by Curtis, Nocedal, and Wächter in [9], with additional functionality for handling inequality constraints. The algorithm is shown to be globally convergent under standard assumptions. Numerical results are presented on partial differential equation constrained model problems.

Key words. large-scale optimization, constrained optimization, interior-point methods, nonconvex programming, trust regions, inexact linear system solvers, Krylov subspace methods

AMS subject classifications. 49M05, 49M37, 65K05, 90C06, 90C26, 90C30, 90C51

1. Introduction.

We consider nonlinear optimization problems of the form

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \; & f(x) \\ \text{s.t. } & c_{\mathcal{E}}(x) = 0 \\ & c_{\mathcal{I}}(x) \geq 0, \end{aligned} \tag{1.1}$$

where the objective $f : \mathbb{R}^n \to \mathbb{R}$ and the constraints $c_{\mathcal{E}} : \mathbb{R}^n \to \mathbb{R}^p$ and $c_{\mathcal{I}} : \mathbb{R}^n \to \mathbb{R}^q$ are sufficiently smooth. We are particularly interested in large-scale problems such as those where the equality constraints are obtained by discretizing partial differential equations (PDEs) and the inequality constraints are, for example, restrictions on a set of control and/or state variables. However, our techniques can be applied to problems with general nonlinear equality and inequality constraints of the form (1.1).

Contemporary optimization methods often require the use of derivatives throughout the solution process. Fast convergence from remote starting points requires first-order derivatives of the objective and the constraints, and local convergence in the neighborhood of solution points can be greatly enhanced with second-order derivatives in the form of the Hessian of the Lagrangian of problem (1.1). For the solution of small to moderately large problems the storage and computational requirements of factoring explicit derivative matrices are reasonable, which allows for the use of very sophisticated and efficient techniques. For this reason, the use of contemporary optimization methods has spread and been successful throughout a number of scientific communities.

In many application areas, however, there is great interest in solving optimization problems of extremely large sizes. For example, if the constraints of the problem correspond to a discretized PDE, then the accuracy of a solution with respect to this infinite-dimensional problem is directly related to size of the largest discrete approximate problem that can be solved. The storage and factorization of explicit

*Courant Institute of Mathematical Sciences, New York University, New York, NY, USA. This author was supported by National Science Foundation grant DMS 0602235.

†Department of Computer Science, University of Basel, Basel, Switzerland.

‡IBM T.J. Watson Research Center, Yorktown Heights, NY, USA.

derivative matrices for such *large-scale* applications is intractable, so researchers and practitioners are often forced to seek alternatives to the better available optimization techniques.

One possible alternative for large-scale optimization is to reduce the original problem into one of a smaller size through a process of *nonlinear elimination* [11, 35]. This process involves an iteration for determining an optimal set of *control variables*. For each set of controls, the equality constraints in (1.1) are solved for the remaining *state variables*, and an auxiliary system may be solved for the sensitivities of the state variables with respect to the controls. The remainder of the iteration involves only the computation of a displacement in the controls. Algorithms of this type, however, suffer from a number of setbacks. For example, if a large number of iterations are required to find an optimal set of control variables, then such a procedure requires a large number of exact solutions of the equality constraints (a PDE) and the adjoint equations (another set of PDEs).

The challenge is thus to design a constrained optimization algorithm that emulates an efficient nonlinear programming approach. The algorithm may utilize matrix-vector products with the constraint Jacobian, its transpose, and the Hessian of the Lagrangian of problem (1.1) together with appropriate preconditioners — quantities that are computable for many large-scale applications of interest — but must overcome the fact that exact factorizations of derivative matrices are impractical to obtain. Iterative linear system solvers present a viable alternative to direct factorization methods, but the benefits of these techniques are only realized if inexact step computations are controlled appropriately in order to guarantee global convergence of the algorithm.

A line of efficient and robust algorithms have been developed that meet these requirements for equality constrained problems; see [4, 5, 9]. These line-search methods illustrate how inexactness in the iterative step computations can be controlled to ensure global convergence to a solution point; first in the case of a convex sequential quadratic programming (SQP) framework, then when handling nonconvexity of the problem functions, and finally when the constraint functions may be ill-conditioned or even inconsistent. A competing trust-region framework, developed in [15, 26] with a close relationship to [6, 7], has also been developed and analyzed. However, embedded in this approach are computations requiring approximate projections of vectors onto the null space of the constraint Jacobian multiple times during each iteration, which can be expensive for large-scale applications. Our interest, therefore, is the extension of the aforementioned line-search methods to the solution of generally constrained problems of the form (1.1).

In this paper, we propose and analyze such an algorithm for large-scale continuous optimization problems and investigate its practical performance. We show that with appropriate scaling matrices, the method developed in [9] is readily extendable to problems where inequality constraints are present. The resulting method is matrix-free, allows for inexact step computations, and is globally convergent to first-order optimal points of (1.1), or at least to stationary points of the feasibility problem

$$\min_{x\in\mathbb{R}^n} \ \tfrac{1}{2}\|c_{\mathcal{E}}(x)\|_2^2 + \tfrac{1}{2}\|c_{\mathcal{I}}(x)^-\|_2^2 \tag{1.2}$$

that yield a nonzero objective value; i.e., infeasible stationary points of problem (1.1). (Here, for a vector z we define $z^- = \max\{0, -z\}$, where the max is taken elementwise.) In addition, the method yields encouraging numerical results on a pair of model PDE-constrained optimization problems, implying that it has much potential

for other large-scale applications.

We note that our method has much in common with the algorithms in [1, 2, 14, 25] as we follow a *full-space*, or *all-at-once*, approach for PDE-constrained problems. The major difference, however, is that we present conditions that guarantee the global convergence of the algorithm. Other related techniques take the form of reduced-space SQP methods [17], more general step decomposition approaches [16, 31], and algorithms that reformulate the problem as a mixed complementarity problem [18]. We believe that our framework has an advantage over these, however, in terms of practical applicability for general purpose problems.

We organize the paper as follows. In §2 we present our matrix-free method, at the heart of which are conditions dictating when a given trial search direction should be considered sufficient to ensure global convergence of the algorithm. Section 3 contains analysis of the global behavior of the approach, while §4 illustrates the practical behavior of a preliminary implementation of our algorithm. Finally, in §5 we present closing remarks.

Notation.. All norms are considered ℓ_2 unless otherwise indicated, though much of our analysis would apply for any vector-based norm. We drop function dependencies once values are clear from the context and use the expression $M_1 \succ M_2$ to indicate that the matrix $M_1 - M_2$ is positive definite. Parenthesized superscripts are used to indicate the component of a vector and subscripts are used to indicate the current iteration number in an algorithm.

2. An Interior-point Algorithm. In this section we present our algorithm. We begin by describing an interior-point framework in an environment where steps can be computed exactly from subproblems and linear systems, and then introduce algorithmic components when inexact step computations are introduced. The resulting algorithm is a line-search interior-point method with safeguards for situations when the problem is nonconvex and when the constraint Jacobians are ill-conditioned or rank deficient. We note that the components of our interior-point framework are defined in a nonstandard fashion (as explained below), but that through our formulation we may more directly extend the techniques presented in [9].

2.1. Interior-point Framework. Our algorithm follows a standard interior-point strategy in that problem (1.1) is solved via the solution of a sequence of barrier subproblems of the form

$$\begin{aligned} \min\ & f(x) - \mu \sum_{i=1}^{q} \ln s^{(i)} \\ \text{s.t. } & c_{\mathcal{E}}(x) = 0 \\ & c_{\mathcal{I}}(x) - s = 0 \end{aligned} \tag{2.1}$$

for decreasing values of the barrier parameter $\mu > 0$. The Lagrangian for (2.1), with multipliers $\lambda_{\mathcal{E}} \in \mathbb{R}^p$ and $\lambda_{\mathcal{I}} \in \mathbb{R}^q$, is given by

$$\mathcal{L}(x, s, \lambda_{\mathcal{E}}, \lambda_{\mathcal{I}}; \mu) \triangleq f(x) - \mu \sum_{i=1}^{q} \ln s^{(i)} + \lambda_{\mathcal{E}}^T c_{\mathcal{E}}(x) + \lambda_{\mathcal{I}}^T (c_{\mathcal{I}}(x) - s), \tag{2.2}$$

so if f, $c_{\mathcal{E}}$, and $c_{\mathcal{I}}$ are sufficiently smooth, then the first-order optimality conditions for (2.1) are

$$\begin{aligned} \nabla f(x) + (\nabla c_{\mathcal{E}}(x))\lambda_{\mathcal{E}} + (\nabla c_{\mathcal{I}}(x))\lambda_{\mathcal{I}} &= 0 \\ -\mu S^{-1}e - \lambda_{\mathcal{I}} &= 0 \\ c_{\mathcal{E}}(x) &= 0 \\ c_{\mathcal{I}}(x) - s &= 0, \end{aligned} \tag{2.3}$$

along with $s > 0$. Here, we have defined $S = \operatorname{diag}(s)$ and $e \in \mathbb{R}^q$ as a vector of ones.

If problem (2.1) is infeasible, then the algorithm is designed to converge toward a first-order optimal solution of the feasibility problem (1.2) so that a justified declaration of infeasibility can be made. Noting that $\|c_{\mathcal{I}}(x)^-\|^2$ is differentiable with $\nabla\left(\|c_{\mathcal{I}}(x)^-\|^2\right) = -2\nabla c_{\mathcal{I}}(x)c_{\mathcal{I}}(x)^-$, such a point can be characterized as a solution to the nonlinear system of equations

$$(\nabla c_{\mathcal{E}}(x))c_{\mathcal{E}}(x) - (\nabla c_{\mathcal{I}}(x))c_{\mathcal{I}}(x)^- = 0.$$

If $s \geq 0$ and $c_{\mathcal{I}}(x) - s \leq 0$, then this is equivalent to

$$\begin{aligned} (\nabla c_{\mathcal{E}}(x))c_{\mathcal{E}}(x) + (\nabla c_{\mathcal{I}}(x))(c_{\mathcal{I}}(x) - s) &= 0 \\ -S(c_{\mathcal{I}}(x) - s) &= 0. \end{aligned} \tag{2.4}$$

Let j be the outer iteration counter for solving the nonlinear program (1.1) and let k be the inner iteration counter for solving the barrier subproblem (2.1). Defining the primal and dual iterates as

$$z = \begin{bmatrix} x \\ s \end{bmatrix} \quad \text{and} \quad \lambda = \begin{bmatrix} \lambda_{\mathcal{E}} \\ \lambda_{\mathcal{I}} \end{bmatrix},$$

respectively, the barrier objective and constraints as

$$\varphi(z;\mu) = f(x) - \mu \sum_{i=1}^{q} \ln s^{(i)} \quad \text{and} \quad c(z) = \begin{bmatrix} c_{\mathcal{E}}(x) \\ c_{\mathcal{I}}(x) - s \end{bmatrix}$$

with corresponding *scaled* first derivatives

$$\gamma(z;\mu) = \begin{bmatrix} \nabla f(x) \\ -\mu e \end{bmatrix} \quad \text{and} \quad A(z) = \begin{bmatrix} \nabla c_{\mathcal{E}}(x)^T & 0 \\ \nabla c_{\mathcal{I}}(x)^T & -S \end{bmatrix}, \tag{2.5}$$

respectively, and the *scaled* Hessian of the Lagrangian as

$$W(z,\lambda;\mu) \triangleq \begin{bmatrix} \nabla^2_{xx} f & 0 \\ 0 & \mu I \end{bmatrix} + \sum_{i=1}^{p} \lambda_{\mathcal{E}}^{(i)} \begin{bmatrix} \nabla^2_{xx} c_{\mathcal{E}}^{(i)} & 0 \\ 0 & 0 \end{bmatrix} + \sum_{i=1}^{q} \lambda_{\mathcal{I}}^{(i)} \begin{bmatrix} \nabla^2_{xx} c_{\mathcal{I}}^{(i)} & 0 \\ 0 & 0 \end{bmatrix}, \tag{2.6}$$

a Newton iteration for (2.1) amounts to the solution of the linear system

$$\begin{bmatrix} W_k & A_k^T \\ A_k & 0 \end{bmatrix} \begin{bmatrix} d_k \\ \delta_k \end{bmatrix} = -\begin{bmatrix} \gamma_k + A_k^T \lambda_k \\ c_k \end{bmatrix}. \tag{2.7}$$

(In our algorithm, W_k represents a bounded symmetric approximation to (2.6) that is sufficient for ensuring global convergence.)

Notice that our definition of the primal search direction d_k is nonstandard; i.e., with the formulation above, the primal iterate is to be updated with a line-search coefficient $\alpha_k \in (0, 1]$ as

$$z_{k+1} \leftarrow z_k + \alpha_k \widetilde{d}_k \quad \text{with} \quad \widetilde{d}_k = \begin{bmatrix} d_k^x \\ S_k d_k^s \end{bmatrix}, \tag{2.8}$$

where d_k^x and d_k^s are the components of d_k corresponding to the x and s variables, respectively. This update follows from our use of *scaled* first and second derivatives; see [6] for an example of another algorithm that uses such a scaling for the slack variables. The scaling is crucial for allowing us to directly apply many of the theoretical results from [9], which discusses the algorithm below applied to problems with only equality constraints.

Line-search interior-point methods that compute search directions directly via (2.7) have been shown to fail to converge from remote starting points; see [32]. Moreover, in situations where the matrix A_k is (nearly) rank deficient, it is necessary to safeguard the step computation to avoid long, unproductive search directions, or to handle situations when the system is inconsistent and the Newton step is thus undefined. Following the algorithm in [9], we avoid these difficulties by replacing d_k in (2.7) by a concatenation of a *normal* step v_k and a *tangential* step u_k.

The normal step v_k is designed as a move toward the satisfaction of a linear model of the constraints within a trust region, and is defined via the subproblem

$$\begin{aligned} \min_{v \in \mathbb{R}^n} \ & \tfrac{1}{2}\|c_k + A_k v\|^2 \\ \text{s.t.} \ & \|v\| \leq \omega \|A_k^T c_k\|, \end{aligned} \tag{2.9}$$

where $\omega > 0$ is a given constant. Note that the radius of the trust region in this problem is related to the conditions (2.4), which are equivalent to $A_k^T c_k = 0$, so that, for example, at a stationary point of the feasibility measure we have $v_k = 0$. We prefer this form of a trust-region constraint as it simplifies our analysis in §3; see [30] and references therein for similar approaches used by other authors.

Our *tangential* step u_k is intended as a move toward optimality that does not mar the progress toward feasibility attained by the normal component. It is defined implicitly via the perturbed Newton system

$$\begin{bmatrix} W_k & A_k^T \\ A_k & 0 \end{bmatrix} \begin{bmatrix} d_k \\ \delta_k \end{bmatrix} = - \begin{bmatrix} \gamma_k + A_k^T \lambda_k \\ -A_k v_k \end{bmatrix}, \tag{2.10}$$

which yields

$$u_k \triangleq d_k - v_k. \tag{2.11}$$

Note that (2.10) has a consistent second block of equations even if A_k is rank deficient. Moreover, if the matrix W_k is positive definite in the null space of A_k, then a solution to (2.10) corresponds to a solution to the quadratic program

$$\begin{aligned} \min_{u \in \mathbb{R}^n} \ & (\gamma_k + W_k v_k)^T u + \tfrac{1}{2} u^T W_k u \\ \text{s.t.} \ & A_k u = 0. \end{aligned} \tag{2.12}$$

However, if W_k is not sufficiently positive definite, then this matrix must be modified or replaced by a sufficiently positive definite approximation matrix in order to ensure

that the resulting solution to (2.10) is an appropriate search direction. If a factorization is performed, then the need for such a modification can be verified by observing the inertia of the primal-dual matrix (see [21] and the algorithm in [28]), but for our purposes we leave consideration of this issue until a precise procedure is outlined in our algorithm with inexact step computations below.

With a search direction $d_k = u_k + v_k$ computed via (2.9) and (2.10), we perform our line search by first determining the maximum stepsize $\alpha_k^{\max} \in (0, 1]$ satisfying the fraction-to-the-boundary rule

$$s_k + \alpha_k^{\max} S_k d_k^s \geq (1 - \eta_1) s_k \tag{2.13}$$

for a constant $\eta_1 \in (0, 1)$. Then, an appropriate stepsize $\alpha_k \in (0, \alpha_k^{\max}]$ is determined yielding progress in the penalty function

$$\phi(z; \mu, \pi) = \varphi(z; \mu) + \pi \|c(z)\|, \tag{2.14}$$

where $\pi > 0$ is a penalty parameter. Denoting $D\phi(\widetilde{d}; \mu, \pi)$ as the directional derivative of ϕ at z along $\widetilde{d}$ (see (2.8)) and setting π_k so that $D\phi_k(\widetilde{d}_k; \mu_j, \pi_k) < 0$, an appropriate sufficient decrease requirement for $\alpha_k \in (0, 1]$ is given by the Armijo condition

$$\phi(z_k + \alpha_k \widetilde{d}_k; \mu_j, \pi_k) \leq \phi(z_k; \mu_j, \pi_k) + \eta_2 \alpha_k D\phi_k(\widetilde{d}_k; \mu_j, \pi_k) \tag{2.15}$$

for a constant $\eta_2 \in (0, 1)$.

A framework that assumes exact solutions of subproblems and linear systems is summarized as Algorithm 2.1. A number of modifications to this framework are possible, and further details are necessary in order to guarantee global convergence. However, Algorithm 2.1 is suitable for our purposes of setting up the details that follow. Notice that the fraction-to-the-boundary rule (2.13) ensures $s_k \geq 0$ and that the slack reset, performed after the iterate has been updated in the inner `for` loop, ensures

$$c_{\mathcal{I}}(x_{k+1}) - s_{k+1} = c_{\mathcal{I}}(x_{k+1}) - \max\{s_k + \alpha_k S_k d_k^s, c_{\mathcal{I}}(x_{k+1})\} \leq 0 \tag{2.16}$$

for all k, so an iterate z_k yielding

$$A_k^T c_k = 0, \tag{2.17}$$

(equivalently, (2.4)), is a stationary point of problem (1.2).

2.2. An Interior-Point Method with Inexact Step Computations. In the remainder of this section we present techniques for applying Algorithm 2.1 when the matrices A_k and W_k do not need to be explicitly stored or factored, meaning that exact solutions of (2.9) and (2.10) do not need to be computed. The algorithm is very closely related to the method for equality constrained problems presented in [9].

Our algorithm requires that the normal component v_k satisfies the following condition.

NORMAL COMPONENT CONDITION. *The normal component v_k must be feasible for problem* (2.9) *and satisfy the Cauchy decrease condition*

$$\|c_k\| - \|c_k + A_k v_k\| \geq \epsilon_v(\|c_k\| - \|c_k + \bar{\alpha}_k A_k \bar{v}_k\|) \tag{2.18}$$

Algorithm 2.1 Interior-Point Framework

Choose parameters $0 < \eta_1, \eta_2 < 1$ and initialize $\mu_0 > 0$
for $j = 0, 1, 2, \ldots,$ until termination criteria for (1.1) or for (1.2) is satisfied **do**
  if $j = 0$ **then**
    Initialize (z_0, λ_0) with $s_0 > 0$ and $s_0 \geq c_{\mathcal{I}}(x_0)$
  else
    Set $(z_0, \lambda_0) \leftarrow (z_k, \lambda_k)$ from the solution of the last barrier subproblem
  end if
  Initialize $\pi_{-1} > 0$
  for $k = 0, 1, 2, \ldots,$ until termination criteria for (2.1) or for (1.2) is satisfied **do**
    Compute $d_k = u_k + v_k$ and δ_k via (2.9) and (2.10)
    Set $\widetilde{d}_k \leftarrow (d_k^x, S_k d_k^s)$
    Set $\pi_k \geq \pi_{k-1}$ so that $D\phi_k(\widetilde{d}_k; \mu_j, \pi_k) < 0$
    Choose $\alpha_k \in (0, 1]$ satisfying (2.13) and (2.15)
    Update $z_{k+1} \leftarrow z_k + \alpha_k \widetilde{d}_k$ and choose λ_{k+1}
    Set $s_{k+1} \leftarrow \max\{s_{k+1}, c_{\mathcal{I}}(x_{k+1})\}$
  end for
  Choose μ_{j+1} (so that $\{\mu_j\} \to 0$)
end for

for some constant $\epsilon_v \in (0, 1)$, where $\bar{v}_k = -A_k^T c_k$ is the steepest descent direction for the objective of problem (2.9) *at $v = 0$ and $\bar{\alpha}_k$ is chosen as the solution to the one-dimensional problem*

$$\begin{aligned} &\min_{\bar{\alpha} \in \mathbb{R}} \ \tfrac{1}{2}\|c + \bar{\alpha} A_k \bar{v}_k\|^2 \\ &\text{s.t. } \bar{\alpha} \leq \omega. \end{aligned} \tag{2.19}$$

A number of iterative techniques have been developed and well-studied for the inexact solution of (2.9) with solutions satisfying this condition, including the conjugate gradient or LSQR [22] algorithm with Steihaug stopping tests [29]. In our implementation described in §4, we prefer a type of inexact dogleg approach [23, 24], which we found to outperform these methods, especially in cases where A_k is (nearly) rank deficient.

Given v_k satisfying the normal component condition, we next compute a tangential component and a displacement for the Lagrange multipliers by applying an iterative linear system solver to the primal-dual system (2.10). During each iteration, this process yields the residual vector

$$\begin{bmatrix} \rho_k \\ r_k \end{bmatrix} \triangleq \begin{bmatrix} W_k & A_k^T \\ A_k & 0 \end{bmatrix} \begin{bmatrix} d_k \\ \delta_k \end{bmatrix} + \begin{bmatrix} \gamma_k + A_k^T \lambda_k \\ -A_k v_k \end{bmatrix}. \tag{2.20}$$

Unlike common inexact Newton techniques (see [10]) that accept or reject a search direction simply based on values of this corresponding residual vector, however, we need to be sure that the resulting direction is one that ensures sufficient progress toward a solution of the current barrier subproblem (2.1). Moreover, the primal-dual matrix may need to be modified within the solution process to guarantee descent for our penalty function.

Following the terminology developed in [4, 5, 9], we present three termination criteria for the primal-dual step computation as sufficient merit function approximation

reduction termination tests (SMART tests, for short). The central tenet of these criteria is that a nonzero primal search direction is acceptable if and only if it provides a sufficient reduction in the local model

$$m(d;\mu,\pi) = \varphi + \gamma^T d + \pi\|c + Ad\|$$

of the penalty function ϕ from a given iterate z_k for an appropriate value of π_k. The reduction in m_k attained by d_k is defined as

$$\begin{aligned}\Delta m_k(d_k;\mu_j,\pi_k) = &\ m_k(0;\mu_j,\pi_k) - m_k(d_k;\mu_j,\pi_k)\\ = &\ -\gamma_k^T d_k + \pi_k(\|c_k\| - \|c_k + A_k d_k\|),\end{aligned}$$

and can be computed easily for any given d_k. We show later on (see Lemma 3.5) that d_k corresponds to a search direction of sufficient descent in ϕ when Δm_k is sufficiently large, so the following condition plays a crucial role in our termination conditions for the primal-dual step computation.

Model reduction condition. *A search direction $d_k = u_k + v_k \neq 0$ must satisfy*

$$\Delta m_k(d_k;\mu_j,\pi_k) \geq \max\{\tfrac{1}{2}u_k^T W_k u_k, \theta\|u_k\|^2\} + \sigma\pi_k(\|c_k\| - \|c_k + A_k v_k\|) \tag{2.21}$$

for $\pi_k > 0$, where $\sigma \in (0,1)$ and $\theta > 0$ are given constants.

Although the model reduction condition (2.21) forms the center of the termination conditions for the step computation of our algorithm, a couple auxiliary requirements are also necessary in order to ensure global convergence. We present these two conditions before presenting our termination tests below.

First, the following condition ensures that either the tangential component u_k is proportional in norm to the normal component v_k, or that it satisfies a certain optimality property with respect to the quadratic program (2.12).

Tangential component condition. *A tangential component u_k yielding $d_k = u_k + v_k \neq 0$ must satisfy*

$$\|u_k\| \leq \psi\|v_k\| \tag{2.22}$$

or

$$\tfrac{1}{2}u_k^T W_k u_k \geq \theta\|u_k\|^2 \tag{2.23a}$$

$$\text{and} \quad (\gamma_k + W_k v_k)^T u_k + \tfrac{1}{2}u_k^T W_k u_k \leq \zeta\|v_k\|, \tag{2.23b}$$

where $\psi \geq 0$ and $\zeta \geq 0$ are given constants and θ is given in (2.21).

Specifically, (2.23) ensures that the tangential component u_k yields a sufficiently low objective value for problem (2.12) and is a direction of sufficiently positive curvature.

The second auxiliary condition resembles a common inexact Newton condition [10]. In the analysis of our algorithm, this condition essentially ensures that dual feasibility is attained as the algorithm converges to a primal solution point.

Dual residual condition. *The dual residual vector ρ_k must satisfy*

$$\|\rho_k\| \leq \kappa \min\left\{\left\|\begin{bmatrix}\gamma_k + A_k^T\lambda_k\\ -A_k v_k\end{bmatrix}\right\|, \left\|\begin{bmatrix}\gamma_{k-1} + A_{k-1}^T\lambda_k\\ -A_{k-1}v_{k-1}\end{bmatrix}\right\|\right\} \tag{2.24}$$

for a given $\kappa \in (0,1)$.

We note that the norm of the residual r_k in (2.20) is implicitly controlled by the tangential component condition, so it does not need to be controlled explicitly in the algorithm.

We are now ready to present our three termination tests for the iterative primal-dual step computation. The first termination test is simply a compilation of the above conditions.

TERMINATION TEST 1. *A search direction (d_k, δ_k) is acceptable if for $\pi_k = \pi_{k-1}$ the model reduction condition* (2.21) *is satisfied, the tangential component condition* (2.22) *or* (2.23) *is satisfied, and if the dual residual condition* (2.24) *holds.*

The second termination test amounts to an adjustment of the multiplier estimates where we temporarily suspend movement in the primal space. Inclusion of this test may be necessary for situations when z_k is a stationary point for the feasibility problem (1.2), but $\gamma_k + A_k^T \lambda_k \neq 0$, as in these situations a (near) exact solution to (2.10) may be required to produce an acceptable step. The test may also save computational expense when z_k is in the neighborhood of such points. It is important to note that for search directions satisfying only this test, the primal step components are reset to zero vectors; i.e., we set $(v_k, u_k, d_k) \leftarrow (0, 0, 0)$, $\pi_k \leftarrow \pi_{k-1}$, and choose $\alpha_k \leftarrow 1$.

TERMINATION TEST 2. *If for a given constant $\epsilon_2 > 0$ we have*

$$\|A_k^T c_k\| \leq \epsilon_2 \|\gamma_k + A_k^T \lambda_k\|, \tag{2.25}$$

then a search direction $(d_k, \delta_k) \leftarrow (0, \delta_k)$ is acceptable if the dual residual condition (2.24) *holds; i.e.,*

$$\|\gamma_k + A_k^T(\lambda_k + \delta_k)\| \leq \kappa \min \left\{ \|\gamma_k + A_k^T \lambda_k\|, \left\| \begin{bmatrix} \gamma_{k-1} + A_{k-1}^T \lambda_k \\ -A_{k-1} v_{k-1} \end{bmatrix} \right\| \right\}. \tag{2.26}$$

for the given $\kappa \in (0, 1)$.

Condition (2.25) implicitly ensures that the algorithm will allow only a finite number of iterations where only termination test 2 is satisfied before requiring an iteration to satisfy one of the other two tests; i.e., it ensures that the algorithm does not only focus on reducing dual infeasibility.

The third termination test is necessary for situations where the model reduction condition cannot be satisfied without an accompanying increase in the penalty parameter. Such an increase may be needed, for example, if the current primal iterate is a stationary point for $\phi(\cdot; \mu_j, \pi_{k-1})$ that is not a solution of the current barrier subproblem (2.1) or the feasibility problem (1.2). The test requires a reduction in the local linear model of the constraints, so for this test to be considered during iteration k, we require $\|c_k\| - \|c_k + A_k v_k\| > 0$.

TERMINATION TEST 3. *A search direction (d_k, δ_k) is acceptable if the tangential component condition* (2.22) *or* (2.23) *is satisfied, the dual residual condition* (2.24) *holds, and if*

$$\|c_k\| - \|c_k + A_k d_k\| \geq \epsilon_3 (\|c_k\| - \|c_k + A_k v_k\|) > 0. \tag{2.27}$$

for some constant $\epsilon_3 \in (0, 1)$.

For steps satisfying only termination test 3, we require

$$\pi_k \geq \frac{\gamma_k^T d_k + \max\left\{ \frac{1}{2} u_k^T W_k u_k, \theta \|u_k\|^2 \right\}}{(1 - \tau)(\|c_k\| - \|c_k + A_k d_k\|)} \triangleq \pi_k^{trial}, \tag{2.28}$$

for a given constant $\tau \in (0,1)$. Along with (2.27) this bound yields

$$\begin{aligned}\Delta m_k(d_k;\mu_j,\pi_k) &\geq \max\{\tfrac{1}{2}u_k^T W_k u_k, \theta\|u_k\|^2\} + \tau\pi_k(\|c_k\| - \|c_k + A_k d_k\|) \\ &\geq \max\{\tfrac{1}{2}u_k^T W_k u_k, \theta\|u_k\|^2\} + \tau\epsilon_3\pi_k(\|c_k\| - \|c_k + A_k v_k\|),\end{aligned}$$

so (2.21) is satisfied for $\sigma = \tau\epsilon_3$. From now on we assume that the constants τ, σ, and ϵ_3 are chosen to satisfy this relationship for consistency between termination tests 1 and 3.

As previously mentioned, W_k may need to be modified or replaced by a sufficiently positive definite matrix to ensure that an appropriate search direction satisfying the tangential component condition (2.22) or (2.23) will eventually be computed. During the iterative primal-dual step computation, we call for such a modification by implementing the following rule.

HESSIAN MODIFICATION STRATEGY. *Let W_k be the current scaled Hessian approximation and let a trial step (d_k, δ_k) be given. If $u_k = d_k - v_k$ satisfies* (2.22) *or* (2.23a)*, then maintain the current W_k; otherwise, modify W_k to increase its smallest eigenvalue.*

We do not require that the modifications take on a specific form, though in our implementation we employ the common technique of adding a multiple of a positive definite diagonal matrix to W_k to increase all of its eigenvalues. For theoretical purposes, our only stipulation is that after a finite number of modifications the matrix W_k is sufficiently positive definite and uniformly bounded; see Assumptions 3.1 and 3.3 below.

Finally, once a suitable search direction has been computed, we perform a backtracking line search to compute $\alpha_k \in (0,1]$ so that the fraction-to-the-boundary rule (2.13) is satisfied and sufficient decrease of the penalty function ϕ is made. As the directional derivative $D\phi_k(\widetilde{d}_k;\mu_j,\pi_k)$ is not easily computed for an arbitrary search direction, in place of (2.15) we observe the related condition

$$\phi(z_k + \alpha_k\widetilde{d}_k;\mu_j,\pi_k) \leq \phi(z_k;\mu_j,\pi_k) - \eta_2\alpha_k\Delta m_k(d_k;\mu_j,\pi_k). \tag{2.29}$$

This sufficient decrease condition is justified by Lemma 3.5 below. The new dual iterate λ_{k+1} is required to satisfy

$$\|\gamma_k + A_k^T\lambda_{k+1}\| \leq \|\gamma_k + A_k^T(\lambda_k + \delta_k)\|. \tag{2.30}$$

Using the dual space direction δ_k to obtain $\lambda_{k+1} = \lambda_k + \beta_k\delta_k$ with a steplength coefficient $\beta_k \in [0,1]$, this inequality may be satisfied by simply setting $\beta_k \leftarrow 1$, or by performing a one-dimensional minimization of the dual feasibility measure along δ_k. A third viable option is described with our implementation in §4.

The details of our algorithm are specified as Algorithm 2.2.

3. Algorithm Analysis. In this section we analyze the global behavior of Algorithm 2.2. As the formulation of our approach was intentionally constructed to resemble the method presented in [9], our analysis is facilitated by referring directly to results in that work. In particular, the quantities in the linear system (2.7) have the same properties as those in the primal-dual system presented in [9], so all results pertaining to the (inexact) solution of this system and its solution carry over. The major difference, however, is that with slack variables, a logarithmic barrier term in the objective of (2.1), and the fraction-to-the-boundary rule (2.13), results effected by the line search must be reanalyzed.

Algorithm 2.2 Interior-Point Algorithm with SMART Tests

Choose parameters $\psi, \zeta \geq 0,\ 0 < \eta_1, \eta_2, \epsilon_v, \kappa, \epsilon_3, \tau < 1,\ 0 < \omega, \theta, \epsilon_2, \delta_\pi$ and set $\sigma \leftarrow \tau\epsilon_3$
Initialize $\mu_0 > 0$
for $j = 0, 1, 2, \ldots,$ until termination criteria for (1.1) or for (1.2) is satisfied **do**
  if $j = 0$ **then**
    Initialize (z_0, λ_0) with $s_0 > 0$ and $s_0 \geq c_{\mathcal{I}}(x_0)$
  else
    Set $(z_0, \lambda_0) \leftarrow (z_k, \lambda_k)$ from the solution of the last barrier subproblem
  end if
  Initialize $\pi_{-1} > 0$
  for $k = 0, 1, 2, \ldots,$ until termination criteria for (2.1) or for (1.2) is satisfied **do**
    Compute v_k satisfying the normal component condition
    Compute an approximate solution to (2.10) with the unmodified W_k
    while (d_k, δ_k) does not satisfy any of termination tests 1, 2, or 3 **do**
      Run the Hessian modification strategy to modify W_k, if necessary
      Compute an improved approximate solution to (2.10) with the current W_k
    end while
    if (d_k, δ_k) satisfies termination test 3 and (2.28) does not hold **then**
      Set $\pi_k \leftarrow \pi_k^{trial} + \delta_\pi$
    end if
    if (d_k, δ_k) satisfies only termination test 2 **then**
      Set $d_k \leftarrow 0$
    end if
    Set $\widetilde{d}_k \leftarrow (d_k^x, S_k d_k^s)$
    if $\widetilde{d}_k \neq 0$ **then**
      Compute the maximum $\alpha_k^{\max} \in (0, 1]$ satisfying (2.13)
      Compute the smallest $l \in \mathbb{N}_0$ such that $\alpha_k \leftarrow 2^{-l}\alpha_k^{\max}$ satisfies (2.29)
    else
      Set $\alpha_k \leftarrow 1$
    end if
    Update $z_{k+1} \leftarrow z_k + \alpha_k \widetilde{d}_k$ and choose λ_{k+1} satisfying (2.30)
    Set $s_{k+1} \leftarrow \max\{s_{k+1}, c_{\mathcal{I}}(x_{k+1})\}$
  end for
  Choose μ_{j+1} (so that $\{\mu_j\} \to 0$)
end for

We begin with the following assumption concerning a particular iteration in the solution of a given barrier subproblem (2.1).

ASSUMPTION 3.1. *After a finite number of perturbations of W_k made via the Hessian modification strategy, we have $W_k \succ 2\theta I$ for the constant $\theta > 0$ defined in the model reduction condition* (2.21)*. Moreover, the iterative linear system solver can solve* (2.10) *to an arbitrary accuracy for each W_k.*

This assumption is reasonable for a number of iterative linear system solvers, even in situations where the primal-dual matrix in (2.10) is singular. We note that in practical situations for a given W_k the primal-dual system may be inconsistent, but as this can only be due to singularity of W_k, it can be remedied by further modifications of W_k as in the Hessian modification strategy.

The following lemma, proved in [9] under the same conditions as presented here, states that Algorithm 2.2 is well-posed.

LEMMA 3.2. *During iteration k of the inner* `for` *loop in Algorithm 2.2, one of the following holds:*

(a) The primal-dual pair (z_{k-1}, λ_k) is first-order optimal for (2.1) or (1.2), i.e.,

$$A_{k-1}^T c_{k-1} = 0 \quad \text{and} \quad \gamma_{k-1} + A_{k-1}^T \lambda_k = 0; \tag{3.1}$$

(b) The primal-dual pair (z_k, λ_k) is first-order optimal for (2.1) or (1.2), i.e.,

$$A_k^T c_k = 0 \quad \text{and} \quad \gamma_k + A_k^T \lambda_k = 0; \tag{3.2}$$

(c) The `while` *loop terminates finitely with a primal-dual search direction (d_k, δ_k) satisfying at least one of termination tests 1, 2, or 3.*

In cases (a) and (b) in Lemma 3.2 we state that the inner iteration has terminated finitely, so the algorithm can proceed by updating the barrier parameter and moving on to solving the next barrier subproblem (2.1) or by terminating at an infeasible stationary point of the nonlinear program (1.1). Case (c), on the other hand, guarantees that when we are not at a solution point of (2.1) or (1.2), the algorithm will produce a suitable search direction.

We split the remainder of our analysis into two subsections; in the first we consider the convergence behavior for a given barrier subproblem assuming the method does not terminate finitely, and in the second we consider the convergence behavior for the outer `for` loop of Algorithm 2.2. For convenience in the first subsection, we define T_1, T_2, and T_3 as the sets of iteration indices during which termination test 1, 2, and 3 are satisfied, respectively.

3.1. Global convergence for a barrier subproblem. We make the following assumption for the solution of a given barrier subproblem (2.1). Here, and throughout the remainder of our analysis, W_k refers exclusively to the value of this matrix used to compute the search direction (d_k, δ_k). In light of Lemma 3.2, we define the termination criteria for the inner `for` loop of Algorithm 2.2 to be satisfied only if (3.1) or (3.2) holds.

ASSUMPTION 3.3. *The infinite sequence $\{(z_k, \lambda_k)\}$ generated by Algorithm 2.2 is contained in a convex set over which the functions f, $c_{\mathcal{E}}$, and $c_{\mathcal{I}}$ and their first derivatives are bounded and Lipschitz continuous. The sequence $\{W_k\}$ is also bounded over all k.*

The theorem we prove is the following.

THEOREM 3.4. *If all limit points of $\{A_k\}$ have full row rank, then $\{\pi_k\}$ is bounded and*

$$\lim_{k \to \infty} \left\| \begin{bmatrix} \gamma_k + A_k^T \lambda_{k+1} \\ c_k \end{bmatrix} \right\| = 0. \tag{3.3}$$

Otherwise,

$$\lim_{k \to \infty} \| A_k^T c_k \| = 0, \tag{3.4}$$

and if $\{\pi_k\}$ is bounded then

$$\lim_{k \to \infty} \left\| \begin{bmatrix} \gamma_k + A_k^T \lambda_{k+1} \\ A_k^T c_k \end{bmatrix} \right\| = 0. \tag{3.5}$$

This theorem states that all limit points of the algorithm are either feasible or stationary points of the feasibility measure; see (2.17). If all limit points are feasible and satisfy the linear independence constraint qualification (LICQ) for (1.1), then the limit points of $\{A_k\}$ have full row rank and (3.3) holds.

Our first result illustrates that our local model m of the penalty function ϕ is sufficiently accurate in a neighborhood of a given iterate along directions of the type that are computed in the algorithm.

LEMMA 3.5. *There exists $\xi_1 > 0$ such that for all $d \in \mathbb{R}^n$ and $\alpha \in (0,1]$ with $\alpha d^s \geq -\eta_1$ we have*

$$\phi(z + \alpha\widetilde{d}; \mu, \pi) - \phi(z; \mu, \pi) \leq -\alpha \Delta m(d; \mu, \pi) + \xi_1 \pi \alpha^2 \|d\|^2. \tag{3.6}$$

Proof. For any scalars ξ and ξ' satisfying $\xi > 0$ and $\xi' \geq -\eta_1 \xi$, we have

$$\left| \ln(\xi + \xi') - \ln \xi - \frac{\xi'}{\xi} \right| \leq \sup_{\xi'' \in [\xi, \xi+\xi']} \left| \frac{\xi'}{\xi''} - \frac{\xi'}{\xi} \right| = \frac{\xi}{\xi + \xi'} \left(\frac{\xi'}{\xi} \right)^2 \leq \frac{1}{1 - \eta_1} \left(\frac{\xi'}{\xi} \right)^2.$$

Under Assumption 3.3, Lipschitz continuity of $\nabla f(x)$ and $A(z)$ then implies that for some constant $\xi_1 > 0$ we have

$$\begin{aligned}
& \phi(z + \alpha\widetilde{d}; \mu, \pi) - \phi(z; \mu, \pi) \\
= \;& f(x + \alpha d^x) - f(x) - \mu \sum_{i=1}^{n} \ln(s + \alpha S d^s)^{(i)} + \mu \sum_{i=1}^{n} \ln s^{(i)} \\
& + \pi \left(\left\| \begin{bmatrix} c_{\mathcal{E}}(x + \alpha d^x) \\ c_{\mathcal{I}}(x + \alpha d^x) - (s + \alpha S d^s) \end{bmatrix} \right\| - \left\| \begin{bmatrix} c_{\mathcal{E}}(x) \\ c_{\mathcal{I}}(x) - s \end{bmatrix} \right\| \right) \\
\leq \;& \alpha \nabla f(x)^T d^x - \alpha \mu d^s \\
& + \pi \left(\left\| \begin{bmatrix} c_{\mathcal{E}}(x) + \alpha \nabla c_{\mathcal{E}}(x)^T d^x \\ c_{\mathcal{I}}(x) + \alpha \nabla c_{\mathcal{I}}(x)^T d^x - (s + \alpha S d^s) \end{bmatrix} \right\| - \left\| \begin{bmatrix} c_{\mathcal{E}}(x) \\ c_{\mathcal{I}}(x) - s \end{bmatrix} \right\| \right) + \xi_1 \pi \alpha^2 \|d\|^2 \\
= \;& \alpha \gamma^T d + \pi(\|c(z) + \alpha A(z) d\| - \|c(z)\|) + \xi_1 \pi \alpha^2 \|d\|^2 \\
= \;& \alpha \gamma^T d + \pi(\|(1 - \alpha) c(z) + \alpha (c(z) + A(z) d)\| - \|c(z)\|) + \xi_1 \pi \alpha^2 \|d\|^2 \\
\leq \;& \alpha \left(\gamma^T d - \pi(\|c(z)\| - \|c(z) + A(z) d\|) \right) + \xi_1 \pi \alpha^2 \|d\|^2,
\end{aligned}$$

which is (3.6). ∎

Although the algorithm considers the penalty function ϕ, it will be convenient in part of our analysis to work with the scaled and shifted penalty function

$$\hat{\phi}(z; \mu, \pi) \triangleq \tfrac{1}{\pi}(\varphi(z; \mu) - \chi) + \|c(z)\|, \tag{3.7}$$

where χ is a given constant. A useful property of the function $\hat{\phi}$ for a particular value of χ is the subject of the next lemma.

LEMMA 3.6. *The sequence $\{s_k\}$ is bounded, and so $\{\phi_k\}$ is bounded below. Moreover, with χ in (3.7) set as the infimum of φ over all k, we find that for each k we have*

$$\hat{\phi}(z_k; \mu_j, \pi_k) \leq \hat{\phi}(z_{k-1}; \mu_j, \pi_{k-1}) - \tfrac{1}{\pi_{k-1}} \eta_2 \alpha_{k-1} \Delta m_{k-1}(d_{k-1}; \mu_j, \pi_{k-1}), \tag{3.8}$$

so $\{\hat{\phi}(z_k; \mu_j, \pi_k)\}$ is monotonically decreasing.

Proof. By (2.29) and the fact that the slack reset (see (2.16)) only decreases ϕ, it follows that

$$\hat{\phi}(z_k;\mu_j,\pi_{k-1}) \leq \hat{\phi}(z_{k-1};\mu_j,\pi_{k-1}) - \tfrac{1}{\pi_{k-1}}\eta_2\alpha_{k-1}\Delta m_{k-1}(d_{k-1};\mu_j,\pi_{k-1}),$$

which implies

$$\begin{aligned}\hat{\phi}(z_k;\mu_j,\pi_k) \leq\ & \hat{\phi}(z_{k-1};\mu_j,\pi_{k-1}) \\ & + \left(\tfrac{1}{\pi_k} - \tfrac{1}{\pi_{k-1}}\right)(\varphi(z_k;\mu_j) - \chi) - \tfrac{1}{\pi_{k-1}}\eta_2\alpha_{k-1}\Delta m_{k-1}(d_{k-1};\mu_j,\pi_{k-1}). \quad (3.9)\end{aligned}$$

Let ξ be an upper bound for $-f$ and $\|c_{\mathcal{I}}\|$, the existence of which follows from Assumption 3.3. By (3.9), the fact that $\{\pi_k\}$ is monotonically non-decreasing, the inequalities

$$\sum_{i=1}^{q} \ln s_k^{(i)} \leq q \ln \|s_k\|_\infty \leq q \ln \|s_k\|, \quad (3.10)$$

and the non-negativity of Δm_k, we then have that

$$\hat{\phi}(z_k;\mu_j,\pi_k) \leq \hat{\phi}(z_0;\mu_j,\pi_0) + \left(\tfrac{1}{\pi_0} - \tfrac{1}{\pi_k}\right)(\xi + \chi + \mu_j q \max_{0\leq l\leq k} \ln \|s_l\|). \quad (3.11)$$

On the other hand, from the definition of $\hat{\phi}$ and (3.10) we have that for any k,

$$\hat{\phi}(z_k;\mu_j,\pi_k) \geq -\tfrac{1}{\pi_k}(\xi + \chi + \mu_j q \ln \|s_k\|) + \|s_k\| - \xi, \quad (3.12)$$

since $\|c(z_k)\| \geq \|c_{\mathcal{I}}(x_k) - s_k\| \geq \|s_k\| - \xi$. Consider the indices l_i such that $\|s_{l_i}\| = \max_{k\leq l_i} \|s_k\|$. Combining (3.11) and (3.12) for k given by any such l_i, we then obtain

$$-\tfrac{1}{\pi_{l_i}}(\xi+\chi+\mu_j q \ln \|s_{l_i}\|) + \|s_{l_i}\| - \xi \leq \hat{\phi}(z_0;\mu_j,\pi_0) + \left(\tfrac{1}{\pi_0} - \tfrac{1}{\pi_{l_i}}\right)(\xi+\chi+\mu_j q \ln \|s_{l_i}\|),$$

which implies

$$\|s_{l_i}\| \leq \hat{\phi}(z_0;\mu_j,\pi_0) + \xi + \tfrac{1}{\pi_0}(\xi + \chi + \mu_j q \ln \|s_{l_i}\|).$$

Since the ratio $(\ln \|s\|)/\|s\|$ tends to zero as $\|s\| \to \infty$, this relation implies that $\{s_{l_i}\}$ must be bounded. By definition of the indices l_i we conclude that the entire sequence $\{s_k\}$ is bounded.

For the second part of the lemma, we first note that if $k \in T_2$, then $d_k = 0$, the model reduction is $\Delta m_k(d_k;\mu_j,\pi_k) = 0$, and $\hat{\phi}(z_{k+1};\mu_j,\pi_{k+1}) = \hat{\phi}(z_k;\mu_j,\pi_k)$, and so (3.8) follows trivially. Otherwise, with χ chosen as the infimum of φ over all k, it follows from (3.9) and the fact that π_k is non-decreasing that (3.8) holds. □

We now state two results, proved as Lemmas 3.6–3.7 and 3.9–3.10 in [9] under the same conditions as considered here, that will be useful in our analysis. The first relates to the norms of the normal and tangential components computed in the algorithm.

LEMMA 3.7. *There exists $\xi_2 > 0$ such that, for all k,*

$$\xi_2 \|A_k^T c_k\|^2 \leq \|v_k\| \leq \omega \|A_k^T c_k\|, \quad (3.13)$$

and hence v_k is bounded in norm over all k. Moreover, u_k is bounded in norm over all k, so together these results imply that d_k is bounded in norm over all k.

The second result provides related bounds for the model reduction Δm_k and the primal step component.

LEMMA 3.8. *There exists $\xi_3 > 0$ such that, for all $k \notin T_2$, we have*

$$\Delta m_k(d_k; \mu_j, \pi_k) \geq \xi_3 \left(\|u_k\|^2 + \pi_k \|A_k^T c_k\|^2 \right).$$

Similarly, there exists $\xi_4 > 0$ such that, for all k, we have

$$\|d_k\|^2 \leq \xi_4 \left(\|u_k\|^2 + \max\{1, \pi_k\} \|A_k^T c_k\|^2 \right). \tag{3.14}$$

The limit (3.4) now follows from the above results.

LEMMA 3.9. *The sequence $\{z_k\}$ yields*

$$\lim_{k \to \infty} \|A_k^T c_k\| = 0.$$

Proof. Consider an arbitrary constant $\xi > 0$ and define the set

$$\Sigma = \{z : \xi \leq \|A(z)^T c(z)\|\}. \tag{3.15}$$

We prove the result by showing that there can only be a finite number of iterates $z_k \in \Sigma$. Since ξ is chosen arbitrarily, this will prove the result.

We first show that if the algorithm computes an iterate in the set Σ, then we eventually have an iteration not in the set T_2. By contradiction, suppose that there exists an iteration number $k' \geq 0$ such that $z_{k'} \in \Sigma$ and for all $k \geq k'$ we have $k \in T_2$. Then, since $d_k = 0$ for $k \in T_2$, we have $z_k = z_{k'}$ for all $k \geq k'$ and the inequalities (2.26) and (2.30) yield

$$\|\gamma_{k+1} + A_{k+1}^T \lambda_{k+1}\| \leq \|\gamma_k + A_k^T (\lambda_k + \delta_k)\| \leq \kappa \|\gamma_k + A_k^T \lambda_k\|.$$

Thus, since $\kappa \in (0, 1)$ we have the limit

$$\|\gamma_k + A_k^T \lambda_k\| \to 0. \tag{3.16}$$

On the other hand, by the conditions of termination test 2 and the fact that $z_{k'} \in \Sigma$, we have that for all $k \geq k'$

$$0 < \xi \leq \|A_{k'}^T c_{k'}\| = \|A_k^T c_k\| \leq \epsilon_2 \|\gamma_k + A_k^T \lambda_k\|,$$

which contradicts (3.16). Therefore, there must exist $k'' \geq k'$ such that $z_{k''} \in \Sigma$ and $k'' \notin T_2$.

Now consider $z_k \in \Sigma$ with $k \notin T_2$. By Lemma 3.7 we have

$$\|v_k\| \geq \xi_2 \|A_k^T c_k\|^2 \geq \xi_2 \xi^2$$

and we may define

$$u_{\sup}^{\Sigma} \triangleq \sup\{\|u_k\| : z_k \in \Sigma\} < \infty,$$

so together we have

$$\|u_k\| \leq \left(u_{\sup}^{\Sigma} / (\xi_2 \xi^2) \right) \|v_k\|. \tag{3.17}$$

Lemmas 3.7 and 3.8 then imply that there exists a constant $\xi' > 0$ yielding

$$\begin{aligned} \Delta m_k(d_k; \mu_j, \pi_k) &\geq \xi_3 \pi_k \|A_k^T c_k\|^2 \qquad (3.18) \\ &\geq \xi_3 \pi_k \frac{1}{\omega^2} \|v_k\|^2 \\ &\geq \xi' \pi_k \|d_k\|^2, \qquad (3.19) \end{aligned}$$

where the last inequality follows from (3.17). The steplength coefficient α_k can now be bounded in the following manner. First, we note that the fraction-to-the-boundary rule (2.13) is satisfied for any $\alpha_k \leq \eta_1 / \|d_k^s\| \leq \eta_1 / d_{\sup}$, where $\|d_k^s\| \leq d_{\sup}$ for some constant $d_{\sup} > 0$ independent of k whose existence follows from Lemma 3.7. Moreover, if the line search condition (2.29) does not hold for some $\bar{\alpha} \in (0, \eta_1 / d_{\sup})$, then

$$\phi(z_k + \bar{\alpha} \widetilde{d}_k; \mu_j, \pi_k) - \phi(z_k; \mu_j, \pi_k) > -\eta_2 \bar{\alpha} \Delta m_k(d_k; \mu_j, \pi_k), \qquad (3.20)$$

which, along with Lemma 3.5, yields

$$(1 - \eta_2) \Delta m_k(d_k; \mu_j, \pi_k) < \xi_1 \bar{\alpha} \pi_k \|d_k\|^2. \qquad (3.21)$$

This inequality and (3.19) implies

$$(1 - \eta_2) \xi' \pi_k \|d_k\|^2 < \xi_1 \bar{\alpha} \pi_k \|d_k\|^2,$$

so $\alpha_k \geq \alpha_{\inf}^{\Sigma} > 0$ where $\alpha_{\inf}^{\Sigma} \triangleq (1/2) \min\{\eta_1 / d_{\sup}, (1 - \eta_2) \xi' / \xi_1\}$ is a constant independent of k.

Suppose that there is an infinite number of iterations with $z_k \in \Sigma$. For those $z_k \in \Sigma$, Lemma 3.6, (3.15), and (3.18) imply that for each iterate with this property we have

$$\begin{aligned} \hat{\phi}(z_{k+1}; \mu_j, \pi_{k+1}) &\leq \hat{\phi}(z_k; \mu_j, \pi_k) - \tfrac{1}{\pi_k} \eta_2 \alpha_k \Delta m_k(d_k; \mu_j, \pi_k) \\ &\leq \hat{\phi}(z_k; \mu_j, \pi_k) - \eta_2 \alpha_{\inf}^{\Sigma} \xi' \|A_k^T c_k\|^2 \\ &\leq \hat{\phi}(z_k; \mu_j, \pi_k) - \eta_2 \alpha_{\inf}^{\Sigma} \xi' \xi^2, \qquad (3.22) \end{aligned}$$

so $\hat{\phi}$ is reduced by at least a positive constant amount. However, this contradicts the fact that by Lemma 3.6, $\hat{\phi}$ is bounded below and monotonically decreasing, which means that there can only be a finite number of iterates with $z_k \in \Sigma$. □

The first statement in Theorem 3.4 inspires special consideration for situations where all limit points of the sequence $\{A_k\}$ produced by Algorithm 2.2 for a given μ_j have full row rank. This next result, proved in [9] under these conditions, highlights important instances for which we can expect the penalty parameter to remain bounded.

LEMMA 3.10. *Suppose that there exists $k_A \geq 0$ such that the smallest singular values of $\{A_k\}_{k \geq k_A}$ are bounded away from zero. Then, the sequence $\{z_k\}$ yields*

$$\lim_{k \to \infty} \|c_k\| = 0 \qquad (3.23)$$

and $\pi_k = \bar{\pi}$ for all $k \geq \bar{k}$ for some $\bar{k} \geq k_A$ and $\bar{\pi} < \infty$.

There may be other situations, however, when the sequence of penalty parameter values remains constant after a given iteration. In general, this behavior implies that the sequence of steplength coefficients will remain bounded away from zero, as seen in the following lemma.

LEMMA 3.11. *If $\pi_k = \bar{\pi}$ for all $k \geq \bar{k}$ for some $\bar{k} \geq 0$ and $\bar{\pi} < \infty$, then the sequence $\{\alpha_k\}$ is bounded away from zero.*

Proof. First, for $k \in T_2$, Algorithm 2.2 sets $\alpha_k = 1$, so we need only consider cases where $k \notin T_2$.

For $k \notin T_2$, the fraction-to-the-boundary rule (2.13) is satisfied for all $\alpha_k \leq \eta_1/\|d_k^s\| \leq \eta_1/d_{\sup}$, where $\|d_k^s\| \leq d_{\sup}$ for some constant $d_{\sup} > 0$ independent of k whose existence follows from Lemma 3.7. Then, as in the proof of Lemma 3.9 (see (3.20)-(3.21)), we have that if (2.29) fails for $\bar{\alpha} \in (0, \eta_1/d_{\sup})$, then

$$(1-\eta_2)\Delta m_k(d_k;\mu_j,\pi_k) < \xi_1\bar{\alpha}\pi_k\|d_k\|^2.$$

Lemma 3.8 then yields

$$(1-\eta_2)\xi_3\left(\|u_k\|^2 + \pi_k\|A_k^T c_k\|^2\right) < \xi_1\xi_4\bar{\alpha}\pi_k\left(\|u_k\|^2 + \max\{1,\pi_k\}\|A_k^T c_k\|^2\right),$$

so

$$\bar{\alpha} > \frac{(1-\eta_2)\xi_3\left(\|u_k\|^2 + \pi_k\|A_k^T c_k\|^2\right)}{\xi_1\xi_4\pi_k\left(\|u_k\|^2 + \max\{1,\pi_k\}\|A_k^T c_k\|^2\right)} \geq \alpha_{\inf},$$

where $\alpha_{\inf} > 0$ is some constant bounded away from zero whose existence follows from the fact that $0 < \pi_{-1} \leq \pi_k \leq \bar{\pi}$. Thus, if $\pi_k = \bar{\pi}$ for all $k \geq \bar{k}$ for some $\bar{k} \geq 0$ and $\bar{\pi} < \infty$, then α_k for $k \notin T_2$ need never be set below $(1/2)\min\{\eta_1/d_{\sup}, \alpha_{\inf}\}$ for (2.29) to be satisfied. □

This last result ensures that sufficient progress toward a solution point will be made along each nonzero search direction d_k computed in the algorithm. As a result, the following lemma, proved in [9] under these same conditions, states that if the penalty function settles with a particular value of the penalty parameter, the primal step components vanish and dual feasibility is guaranteed.

LEMMA 3.12. *If $\pi_k = \bar{\pi}$ for all $k \geq \bar{k}$ for some $\bar{k} \geq 0$ and $\bar{\pi} < \infty$, then*

$$\lim_{k\to\infty} \|d_k\| = 0$$

and

$$\lim_{k\to\infty} \|\gamma_k + A_k^T\lambda_{k+1}\| = 0.$$

We are now ready to prove the main result stated at the beginning of this section.

Proof. (Theorem 3.4) If all limit points of $\{A_k\}$ have full row rank, then there exists $k_A \geq 0$ such that the smallest singular values of $\{A_k\}_{k\geq k_A}$ are bounded away from zero. By Lemma 3.10 we then have that the penalty parameter is constant for all k sufficiently large, which means $\{\pi_k\}$ is bounded, and so by Lemma 3.10 and Lemma 3.12 we have the limit (3.3). On the other hand, if a limit point of $\{A_k\}$ does not have full row rank, then we have the limit (3.4) by Lemma 3.9. Moreover, if $\{\pi_k\}$ is bounded, then the fact that if Algorithm 2.2 increases π_k it does so by at least δ_π implies that the penalty parameter is in fact constant for all k sufficiently large. This implies with Lemma 3.12 that we have the limit (3.5). □

We reiterate here some key features of our algorithm that have allowed us to extend so many of the results presented in [9] for equality constrained optimization in order to prove Theorem 3.4. First, the inclusion of a slack reset to ensure $s_k \geq 0$ and $c_{\mathcal{I}}(x_k) - s_k \leq 0$ have allowed us to require only $A_k^T c_k \to 0$ to state that the algorithm

converges to a feasible point, or at least to an infeasible stationary point of problem (1.1). Without such a slack reset, the algorithm may stall at a point with $s_k \geq 0$, $A_k^T c_k = 0$, and $c_{\mathcal{I}}^{(i)}(x_k) - s_k^{(i)} > 0$ for some i, which is not a stationary point for the feasibility measure in (1.2). The second critical component of our approach was our use of scaled derivatives in our definition of the primal-dual system (2.10). Naturally, in an implementation an equivalent system can be defined without such a scaling, but it is important to note that our termination tests *must* be enforced with this scaling in order for our global convergence results to apply.

3.2. Global convergence for the nonlinear program. We close this section by proving a result related to the overall global convergence of Algorithm 2.2. In the following theorem, we suppose that for a given j, the inner `for` loop terminates when

$$\|\nabla f(x_k) + \nabla c_{\mathcal{E}}(x_k)\lambda_{\mathcal{E},k+1} + \nabla c_{\mathcal{I}}(x_k)\lambda_{\mathcal{I},k+1}\|_\infty \leq \varepsilon\mu_j, \tag{3.24a}$$

$$\|S_k\lambda_{\mathcal{I},k+1} + \mu_j e\|_\infty \leq \varepsilon\mu_j, \tag{3.24b}$$

$$\text{and} \quad \|c_k\|_\infty \leq \varepsilon\mu_j, \tag{3.24c}$$

for some constant $\varepsilon \in (0,1)$, yielding the outer iterate $\{(z_j, \lambda_j)\}$. Note that if the algorithm terminates finitely with (3.2) satisfied, we set $\lambda_{k+1} \leftarrow \lambda_k$ so that (3.24) holds.

THEOREM 3.13. *Suppose that Assumptions 3.1 and 3.3 hold and define $\{\mu_j\}$ be a sequence of positive constants such that $\{\mu_j\} \to 0$. Algorithm 2.2 then yields one of the following outcomes:*

(i) During outer iteration j, (3.24c) *is never satisfied, in which case the stationarity condition* (2.4) *for the infeasibility problem* (1.2) *is satisfied in the limit.*

(ii) During outer iteration j, there exists an infinite subsequence of inner iterates where (3.24c) *is satisfied but* (3.24a) *or* (3.24b) *(or both) is not, in which case the stationarity condition* (2.4) *for the feasibility problem* (1.2) *is satisfied in the limit and $\{\pi_k\} \to \infty$.*

(iii) Each outer iteration results in an iterate $\{(z_j, \lambda_j)\}$ satisfying (3.24)*, in which case all limit points of $\{x_j\}$ are feasible, and if a limit point $\bar{x}$ of $\{x_j\}$ satisfies the linear independence constraint qualification (LICQ), the first-order optimality conditions of* (1.1) *hold at $\bar{x}$; i.e., there exists $\bar{\lambda}$ such that*

$$\nabla f(\bar{x}) + (\nabla c_{\mathcal{E}}(\bar{x}))\bar{\lambda}_{\mathcal{E}} + (\nabla c_{\mathcal{I}}(\bar{x}))\bar{\lambda}_{\mathcal{I}} = 0 \tag{3.25a}$$

$$c_{\mathcal{E}}(\bar{x}) = 0 \tag{3.25b}$$

$$c_{\mathcal{I}}(\bar{x}) \geq 0 \tag{3.25c}$$

$$\bar{\lambda}_{\mathcal{I}} \leq 0 \tag{3.25d}$$

$$\bar{\lambda}^{(i)} c_{\mathcal{I}}^{(i)}(\bar{x}) = 0, \quad i = 1, \ldots, q. \tag{3.25e}$$

Proof. If (3.24c) is never satisfied during outer iteration j, then we know by Theorem 3.4 that the limit (3.4) holds, which means that (2.4) is satisfied in the limit. This corresponds to situation (i). Further, by Theorem 3.4, we have that if (3.24a) or (3.24b) (or both) is not satisfied for an infinite subsequence of iterates, then $\{\pi_k\} \to \infty$, which along with (3.4) completes the proof of situation (ii).

The remaining possibility is that (3.24) is eventually satisfied during each outer iteration $j \geq 0$. Define an infinite subsequence of indices j_l such that $x_{j_l} \to \bar{x}$ as

$l \to \infty$. Since $s_{j_l} \geq 0$ and $c(z_{j_l}) = (c_{\mathcal{E}}(x_{j_l}), c_{\mathcal{I}}(x_{j_l}) - s_{j_l}) \to 0$, we have $c_{\mathcal{E}}(\bar{x}) = 0$, $s_{j_l} \to \bar{s} = c_{\mathcal{I}}(\bar{x})$ as $l \to \infty$, and $c_{\mathcal{I}}(\bar{x}) \geq 0$. Therefore, all limit points of $\{x_j\}$ are feasible, and in particular (3.25b) and (3.25c) hold.

Now define $\widehat{\mathcal{I}} = \{i : c_{\mathcal{I}}^{(i)}(\bar{x}) = 0\}$. By (3.24b), we have $S_j \lambda_{\mathcal{I},j} \to 0$, which means that $\lambda_{\mathcal{I},j_l}^{(i)} \to 0$ for $i \notin \widehat{\mathcal{I}}$. Along with (3.24a), this implies

$$\nabla f(x_{j_l}) + \nabla c_{\mathcal{E}}(x_{j_l}) \lambda_{\mathcal{E},j_l} + \sum_{i \in \widehat{\mathcal{I}}} \lambda_{\mathcal{I},j_l}^{(i)} \nabla c_{\mathcal{I}}^{(i)}(x_{j_l}) \to 0. \tag{3.26}$$

The inequality (3.24b) also yields

$$s_j^{(i)} \lambda_{\mathcal{I},j}^{(i)} \leq (\varepsilon - 1)\mu_j < 0 \tag{3.27}$$

for $i = 1, \ldots, q$, which along with $s_j \geq 0$ implies that $\lambda_{\mathcal{I},j} \leq 0$ for all j. By LICQ, the columns of $\nabla c_{\mathcal{E}}(\bar{x})$ and the vectors $\nabla c_{\mathcal{I}}^{(i)}(\bar{x})$ for $i \in \widehat{\mathcal{I}}$ form a linearly independent set, so (3.26) and (3.27) imply that the sequence $\{\lambda_{j_l}\}$ converges to some value $\bar{\lambda}$ satisfying (3.25a) and (3.25d). □

4. Numerical Experiments. We included an implementation of our algorithm in the `Ipopt` optimization package [34, 33] version 3.5.5, tied with the iterative linear system solver and preconditioners implemented in the `PARDISO` software package [27] version 3.4. In this section we describe some details of our code and illustrate its performance on a pair of model PDE-constrained test problems.

4.1. Implementation details. The default `Ipopt` code implements the algorithm described in [33]. In order to test the method described in this paper, the following modifications were made.

We augmented the `Ipopt` code to include the option to obtain the search direction by means of inexact normal step and primal-dual step computations. Rather than scale the derivative matrices in order to set up the primal-dual system (2.10), however, we solve an equivalent system with unscaled derivatives (as already implemented in `Ipopt`) and scale the search direction for the slack variables in order to apply our termination tests. Note also that although the algorithm in this paper defines $\widetilde{d}_k^s = S_k d_k^s$ (see (2.8)), all of our theoretical results still hold if the matrix S_k in this definition is replaced by $\hat{S}_k = \mathrm{diag}(\hat{s}_k)$, where $\hat{s}_k^{(i)} = \min\{100, s_k\}$; i.e., only the small elements in the slack variable vector need to be scaled. The advantage of this modified scaling, as it is used in our implementation, is that the algorithm does not encounter inefficiencies or numerical problems if some slacks become very large.

If the (2,2)-block μI in (2.6) were unscaled, it would correspond to the (2,2)-block of the primal Hessian μS_k^{-2}. We use this form in the discussion of our algorithm for simplicity, as it results from applying a Newton iteration to (2.3). An equivalent system of primal-dual equations, however, would result in the primal-dual Hessian $S_k^{-1} Y_k$, where y_k are additional dual iterates corresponding to slack bounds. The original `Ipopt` implementation and our implementation of Algorithm 2.2 use this primal-dual Hessian. We argue that all of our theoretical results still hold with this choice, since y_k are set to ensure that $S_k^{-1} Y_k$ does not deviate too much from μS_k^{-2} and hence W_k remains bounded; see [33]. We initialize $\lambda_0 \leftarrow 0$ and $y_0 \leftarrow 10^{-4}$.

Finally, `Ipopt` 's default filter line-search procedure is replaced by the backtracking line-search method described in Algorithm 2.2 using the exact penalty function (2.14), including the slack reset. Moreover, since we observed in some cases that the

penalty parameter was set to a high value in the early stages of the optimization that hampered progress later on, we employ the flexible penalty method described in [8] since it proved to be more efficient in our tests.

Next we describe some of the details of the normal and primal-dual step computations of Algorithm 2.2. Each is performed by applying the symmetric quasi-minimum residual (`SQMR`) method [12, 13] as implemented in `PARDISO` to a large symmetric indefinite linear system.

As previously mentioned, a conjugate gradient or the LSQR method can be used to compute the normal component v_k. However, we found this approach to perform poorly for ill-conditioned Jacobians. Furthermore, it is not clear how to precondition the underdetermined problem (2.9). Therefore, we chose to apply an inexact dogleg approach; see [23, 24]. We begin by computing the Cauchy point $v_k^C = \bar{\alpha}_k \bar{v}_k$ (see the normal component condition (2.18)) and then (approximately) solve the *augmented system* (e.g., see [7])

$$\begin{bmatrix} I & A_k^T \\ A_k & 0 \end{bmatrix} \begin{bmatrix} v_k^N \\ \delta_k^N \end{bmatrix} = - \begin{bmatrix} 0 \\ c_k \end{bmatrix}, \tag{4.1}$$

which for an inexact solution yields the residual vector

$$\begin{bmatrix} \rho_k^N \\ r_k^N \end{bmatrix} = \begin{bmatrix} I & A_k^T \\ A_k & 0 \end{bmatrix} \begin{bmatrix} v_k^N \\ \delta_k^N \end{bmatrix} + \begin{bmatrix} 0 \\ c_k \end{bmatrix}. \tag{4.2}$$

Note that an exact solution of (4.1) gives the least-norm solution of (2.9) with $\omega = \infty$. The inexact dogleg step is then defined as a point along the line segment between the Cauchy point and v_k^N that is feasible for problem (2.9) and satisfies the Cauchy decrease condition (2.18); i.e., it satisfies our normal component condition. We tailor this approach into an implementation that has worked well in our tests, which means, for example, that we consider the fraction-to-the-boundary rule (2.13) when choosing between the Cauchy and inexact dogleg steps. A detailed description of the normal step computation is provided as Algorithm 4.1.

Algorithm 4.1 Normal Step Computation

Given parameters $0 < \tilde{\epsilon}_v, \kappa_v < 1$ and $l^{\mathrm{n}}_{\max}, \omega > 0$.
Compute the Cauchy point $v_k^C = \bar{\alpha}_k \bar{v}_k$, where $\bar{v}_k = -A_k^T c_k$ and $\bar{\alpha}_k$ solves (2.19)
Initialize $v_k \leftarrow v_k^C$
for $l = 0, 1, 2, \ldots, l^{\mathrm{n}}_{\max}$ **do**
  Perform a `SQMR` iteration on (4.1) to compute $(v^{N,l}, \delta^{N,l})$
  if $\|(\rho^{N,l}, r^{N,l}\|) \leq \kappa_v \|c_k\|$ and $\|c_k + A_k v^{N,l}\| \leq \|c_k + A_k v_k^C\|$ **then**
    break
  end if
end for
Set $v_k^N \leftarrow v^{N,l}$
Set $v_k^D = (1-\alpha) v_k^C + \alpha v_k^N$ where $\alpha \in [0,1]$ is the largest value such that v_k^D is feasible for (2.9)
Set α_k^C and α_k^D as the largest values in $[0,1]$ satisfying (2.13) along v_k^C and v_k^D, respectively
if $(\|c_k\| - \|c_k + \alpha_k^D A_k v_k^D\|) \geq \tilde{\epsilon}_v (\|c_k\| - \|c_k + \alpha_k^C A_k v_k^C\|)$ **then**
  Set $v_k \leftarrow v_k^D$
end if

Algorithm 4.2 provides details of the primal-dual step computation. We run the SQMR algorithm until either an acceptable search direction has been computed (i.e., one satisfying at least one of our termination tests) or the Hessian modification strategy indicates that a perturbation to W_k is appropriate. In the latter case, we restart SQMR with the zero vector and continue this process until an acceptable search direction is computed. The form of the Hessian modification is the same as in the default Ipopt algorithm and described in [33]. We write the algorithm with an iteration limit $l_{\max}^{\mathrm{pd}}$ to indicate that due to numerical errors a solver may fail to satisfy our tests in a finite number of iterations, in which case one may simply accept the last inexact solution that is computed. Note, however, that this upper limit was reached in our experiments only once and the resulting step did not adversely affect progress of the algorithm.

To avoid an unnecessarily large number of outer iterations k and to promote fast local convergence, Algorithm 4.2 aims to produce reasonably accurate solutions that satisfy the residual bound

$$\left\| \begin{bmatrix} \rho_k \\ r_k \end{bmatrix} \right\| \leq \kappa' \left\| \begin{bmatrix} \gamma_k + A_k^T \lambda_k \\ A_k v_k \end{bmatrix} \right\| \tag{4.3}$$

for a given $\kappa' \in (0, 1)$ before consideration of our termination tests. However, if after a fixed number of iterations, $l'_{\max}$, this condition is not satisfied, then it is dropped and the algorithm requires only that one of termination test 1, 2, or 3 is satisfied.

Algorithm 4.2 Primal-dual Step Computation

Choose $\kappa', \tilde{l}_{\max}^{\mathrm{pd}}, l_{\max}^{\mathrm{pd}} > 0$
Set $l \leftarrow 0$, $(d^0, \delta^0) \leftarrow 0$, and W_k via (2.6)
for $l = 1, 2, \ldots, l_{\max}^{\mathrm{pd}}$ **do**
  Perform a SQMR iteration on (2.10) to compute (d^l, δ^l)
  if $l \geq \tilde{l}_{\max}^{\mathrm{pd}}$ or (4.3) holds **then**
    if termination test 1 or 3 is satisfied **then**
      break
    else if termination test 2 is satisfied **then**
      Set $d^l \leftarrow 0$ and **break**
    else if (2.22) and (2.23a) do not hold for d^l **then**
      Set $l \leftarrow 0$, $(d^0, \delta^0) \leftarrow 0$, and modify W_k to increase its smallest eigenvalue
    end if
  end if
end for
Set $(d_k, \delta_k) \leftarrow (d^l, \delta^l)$

As preconditioner, both for the augmented system (4.1) and the primal-dual system (2.7), we use the incomplete multilevel factorization solver in PARDISO with inverse-based pivoting [3], stabilized by symmetric-weighted matchings; see [28]. Here, we used the maximum inverse norm factor $\kappa_1 = 5$ and the dropping tolerances 10^{-3} and 10^{-4} for the Schur complement and the factor, respectively.

As for the input parameters used in our implementation, these quantities are summarized in Table 4.1. Otherwise, Ipopt 's default parameters were used, except that the termination tolerance for the nonlinear program (1.1) was set to 10^{-6}.

Finally, we note that we update $\lambda_{k+1} = \lambda_k + \beta_k \delta_k$ by choosing β_k as the smallest value in $[\alpha_k, 1]$ satisfying (2.30), as this choice yields good practical performance in

Para.	Value	Para.	Value	Para.	Value
ψ	0.1	κ	0.1	ϵ_2	1
ζ	0.1	ϵ_3	0.99	δ_π	10^{-4}
η_1	$\max\{0.99, 1-\mu_j\}$	τ	0.1	σ	$\tau\epsilon_3$
η_2	10^{-8}	ω	100	μ_0	0.1
ϵ_v	1	θ	10^{-12}	π_{-1}	10^{-6}
$\tilde{\epsilon}_v$	0.1	$l^{\mathrm{n}}_{\max}$	200	κ_v	10^{-3}
κ'	10^{-3}	$\tilde{l}^{\mathrm{pd}}_{\max}$	100	$l^{\mathrm{pd}}_{\max}$	500

TABLE 4.1
Parameter values used for Algorithm 2.2

our tests.

4.2. Numerical results on PDE-constrained model problems. We applied our implementation to two PDE-constrained optimal control problems to illustrate its performance on large-scale models implemented in `MATLAB`. Each problem is solved on the three-dimensional domain $\Omega = [0,1] \times [0,1] \times [0,1]$ over which we use equidistant Cartesian grids with N grid points in each spacial direction and a standard 7-point stencil for discretizing the differential operators. The goal is to find optimal values of the control u and state variables $y(x)$ to fit a target $y_t(x)$; i.e., the objective of each problem is to minimize

$$f(y(x)) = \tfrac{1}{2} \int_\Omega (y(x) - y_t(x))^2 \, dx \tag{4.4}$$

The results were obtained on a 2GHz Xeon machine running RedHat Linux.

Our first model is a boundary control problem, inspired by Example 5.1 in [19].

EXAMPLE 4.1. *Let $u(x)$ be defined on the boundary $\partial\Omega$. Then, minimize* (4.4) *for*

$$y_t(x) = 3 + 10x^{(1)}(x^{(1)} - 1)x^{(2)}(x^{(2)} - 1)\sin(2\pi x^{(3)}),$$

subject to $y(x) = u(x)$ on $\partial\Omega$, the differential equation

$$-\nabla(e^{y(x)} \cdot \nabla y(x)) = 20 \quad \text{in } \Omega,$$

and the bounds

$$2.5 \le u(x) \le 3.5 \quad \text{on } \partial\Omega.$$

We use $y_0(x) = 3$ and $u_0(x) = 3$ as starting point. Numerical results for different discretization levels are presented in Table 4.2. Here, the first column shows the number of grid points per spacial dimension, followed by the number of variables, equality constraints, and inequality constraints, respectively. The column headed "# nnz" lists the number of nonzero elements in the primal-dual matrix in (2.10). The next column indicates if the inexact Algorithm 2.2 was used ("I") or, for the sake of comparison, the default `Ipopt` algorithm with the direct linear solver in `PARDISO` was employed ("D"). The last entries show the number of iterations, the required computation time in CPU seconds, and the final value of the objective function.

We find that our implementation of Algorithm 2.2 solves all instances of Example 4.1 in a small number of iterations. The gain in efficiency compared to the original

N	n	p	q	# nnz	Alg	# iter	CPU sec	f^*
20	8000	5832	4336	95561	I	12	33.4	1.3368e-2
30	27000	21952	10096	339871	I	12	139.4	1.3039e-2
40	64000	54872	18256	827181	I	12	406.0	1.2924e-2
50	125000	110592	28816	1641491	I	12	935.6	1.2871e-2
60	216000	195112	41776	2866801	I	13	1987.2	1.2843e-2
70	343000	314432	57136	4587111	I	13	3504.6	1.2826e-2
30	27000	21952	10096	339871	D	10	500.0	1.3039e-2
40	64000	54872	18256	827181	D	10	3196.3	1.2924e-2

TABLE 4.2
Numerical results for Example 4.1

`Ipopt` algorithm that uses a direct linear solver is apparent, particularly in the case of $N = 40$. Even though the original algorithm required fewer iterations and solves only one linear system per iteration instead of both (4.1) and (2.10), the computation time is about 8 times higher. This discrepancy will most likely be more pronounced for larger N.

Example 4.1 is nonlinear and nonconvex, but it was solved in few iterations without ever requiring modifications to the Hessian during the primal-dual step computation. Thus, we now turn to a more challenging problem to illustrate further the efficiency of our method.

Our second example is a simplified hyperthermia cancer treatment model; see [20, 28]. Regional hyperthermia is a cancer therapy that aims at heating large and deeply seated tumors by means of radio wave adsorption, as heating tumors above a temperature of about 41°C results in preferential killing of tumor cells and makes them more susceptible to an accompanying radio or chemotherapy. For designing an optimal therapy, amplitudes and phases of the antennas have to be selected such that the tumor temperature is maximized up to a target therapeutical temperature y_t of 43°C.

EXAMPLE 4.2. *Let the control $u^{(j)} = a^{(j)}e^{i\phi^{(j)}}$ be a complex vector of amplitudes $a \in \mathbb{R}^{10}$ and phases $\phi \in \mathbb{R}^{10}$ of 10 antennas, let $M(x)$ be a 10×10 matrix with $M^{(j,k)}(x) =< E^{(j)}(x), E^{(k)}(x) >$ where $E^{(j)}(x) = sin\left(j \cdot \pi \cdot x^{(1)}x^{(2)}x^{(3)}\right)$, and define the "tumor" to be the central region $\Omega_0 = [3/8, 5/8] \times [3/8, 5/8] \times [3/8, 5/8]$. Then, minimize* (4.4) *for*

$$y_t(x) = \begin{cases} 37 & in\ \Omega\backslash\Omega_0 \\ 43 & in\ \Omega_0 \end{cases}$$

subject to the differential equation

$$-\Delta y(x) - 10(y(x) - 37) - u^* M(x)u = 0 \quad in\ \Omega, \tag{4.5}$$

the state variable bounds

$$37.0 \le y(x) \le 37.5 \quad on\ \partial\Omega$$
$$42.0 \le y(x) \le 44.0 \quad in\ \Omega_0$$

and the control variable bounds

$$-10 \le a \le 10$$
$$-4\pi \le \phi \le 4\pi.$$

The PDE (4.5) describes heat dissipation in the tissue, removal of heat via 37°C blood flow, and absorbed energy produced by the 10 microwave antennas. Note that the algorithm uses a and ϕ as control variables (that form the complex numbers u), and that the $u^*M(x)u$ term in (4.5) is real-valued. As starting point we selected $y(x) = 40$, $a = (1, 0, \ldots, 0)^T$ and $\phi = 0$.

Results for Example 4.2 are presented in Table 4.3. Again, the proposed algorithm is able to solve the problem in a number of iterations comparable to the original algorithm, while savings in computation time become apparent as N increases. The higher iteration count shows that this problem is more difficult to solve than Example 4.1, and both algorithms encountered a significant number of iterations in which the Hessian had to be modified. Also, the problem has several local solutions which explains the difference in the final objective function values.

N	n	p	q	# nnz	Alg	# iter	CPU sec	f^*
10	1020	512	1070	20701	I	40	15.0	2.3037
20	8020	5832	4626	212411	I	62	564.7	2.3619
30	27020	21952	10822	779121	I	146	4716.5	2.3843
40	64020	54872	20958	1924831	I	83	9579.7	2.6460
20	8020	5832	4626	212411	D	87	667.6	2.3571
30	27020	21952	10822	779121	D	91	10952.4	2.3719

TABLE 4.3
Numerical results for Example 4.2

For illustrative purposes, we end by providing plots of the target function y_t and the solution y obtained by Algorithm 2.2 for $N = 40$; see Figures 4.1 and 4.2.

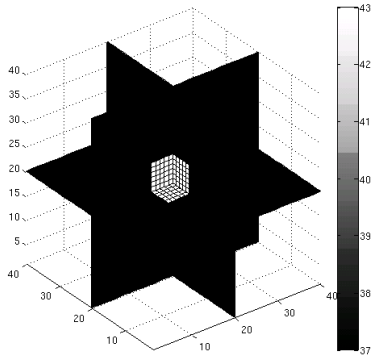

FIG. 4.1. *Target function $y_t(x)$ for Example 4.2 with $N = 40$*

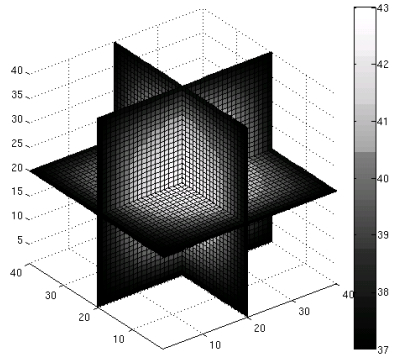

FIG. 4.2. *Solution $y(x)$ obtained by Algorithm 2.2 for Example 4.2 with $N = 40$*

5. Conclusion and Final Remarks. We have presented and analyzed an interior-point algorithm for large-scale nonlinear nonconvex optimization. By extending the results from [9] to problems with inequality constraints, we were able to show that the proposed method is globally convergent under standard assumptions.

The novel aspects of the approach include our definition of termination tests for iterative linear system solvers. These conditions allow for inexact step computations and therefore offer flexibility to the user in terms of computational expense while

providing a theoretical convergence guarantee. Furthermore, the use of iterative linear solvers avoids the requirement to store large derivative matrices explicitly.

In addition, the normal step computation, incorporating a trust region, allows the algorithm to deal with situations where the active constraint gradients are not linearly independent at infeasible limit points. As a consequence, this line-search algorithm can be shown to avoid the convergence problem discussed in [32] and to generate limit points that are stationary points for an infeasibility measure if feasibility is not achieved asymptotically.

An implementation of our algorithm was described and numerical results were presented on two large-scale PDE-constrained problems. These experiments illustrate the computational advantages of our algorithm, especially as the problem size increases. We showed that our method is comparable with respect to iteration count and solution quality with a state-of-the-art continuous optimization algorithm and outperforms the conventional approach in terms of storage and CPU time for the larger problem instances in our tests.

Acknowledgements. We thank Johannes Huber (University of Basel) for writing the Matlab problem formulation for our first numerical example and Peter Carbonetto (University of British Columbia) for contributing his Matlab interface to the `Ipopt` open source project.

REFERENCES

[1] G. Biros and O. Ghattas. Parallel Lagrange-Newton-Krylov-Schur methods for PDE-constrained optimization. Part I: The Krylov-Schur solver. *SIAM Journal on Scientific Computing*, 27(2):687–713, 2005.

[2] G. Biros and O. Ghattas. Parallel Lagrange-Newton-Krylov-Schur methods for PDE-constrained optimization. Part II: The Lagrange-Newton solver and its application to optimal control of steady viscous flows. *SIAM Journal on Scientific Computing*, 27(2):714–739, 2005.

[3] M. Bollhöfer and Y. Saad. Multilevel preconditioners constructed from inverse-based ILUs. *SIAM Journal on Scientific Computing*, 27(5):1627–1650, 2006.

[4] R. H. Byrd, F. E. Curtis, and J. Nocedal. An inexact SQP method for equality constrained optimization. *SIAM Journal on Optimization*, 19(1):351–369, 2008.

[5] R. H. Byrd, F. E. Curtis, and J. Nocedal. An inexact Newton method for nonconvex equality constrained optimization. *Mathematical Programming, Series A*, 2009. (To appear).

[6] R. H. Byrd, J.-Ch. Gilbert, and J. Nocedal. A trust region method based on interior point techniques for nonlinear programming. *Mathematical Programming*, 89(1):149–185, 2000.

[7] R. H. Byrd, M. E. Hribar, and J. Nocedal. An interior point algorithm for large scale nonlinear programming. *SIAM Journal on Optimization*, 9(4):877–900, 1999.

[8] F. E. Curtis and J. Nocedal. Flexible penalty functions for nonlinear constrained optimization. *IMA Journal of Numerical Analysis*, 28(4):749–769.

[9] F. E. Curtis, J. Nocedal, and A. Wächter. A matrix-free algorithm for equality constrained optimization problems with rank deficient jacobians. *SIAM Journal on Optimization*, 2008. (Submitted).

[10] R. S. Dembo, S. C. Eisenstat, and T. Steihaug. Inexact Newton methods. *SIAM Journal on Numerical Analysis*, 19(2):400–408, 1982.

[11] M. Fisher, J. Nocedal, Y. Trémolet, and S. J. Wright. Data assimilation in weather forecasting: A case study in PDE-constrained optimization. *Optimization and Engineering*, 2008. DOI: 10.1007/s11081-008-9051-5.

[12] R. W. Freund. Preconditioning of symmetric, but highly indefinite linear systems. In *15th IMACS World Congress on Scientific Computation, Modelling and Applied Mathematics, Vol. 2, Numerical Mathematics*, pages 551–556. Wissenschaft und Technik Verlag, 1997.

[13] R. W. Freund and F. Jarre. A QMR-based interior-point algorithm for solving linear programs. *Mathematical Programming, Series B*, 76:183–210, 1996.

[14] E. Haber and U. M. Ascher. Preconditioned all-at-once methods for large, sparse parameter estimation problems. *Inverse Problems*, 17(6):1847–1864, 2001.

[15] M. Heinkenschloss and D. Ridzal. An inexact trust-region SQP methods with applications to PDE-constrained optimization. In O. Steinbach and G. Of, editors, *Numerical Mathematics and Advanced Applications: Proceedings of Enumath 2007, the 7th European Conference on Numerical Mathematics and Advanced Applications, Graz, Austria*, Heidelberg, Germany, 2007. Springer-Verlag. (submitted).

[16] M. Heinkenschloss and L. N. Vicente. Analysis of inexact trust-region SQP algorithms. *SIAM Journal on Optimization*, 12(2):283–302, 2001.

[17] H. Jäger and E. W. Sachs. Global convergence of inexact reduced SQP methods. *Optimization Methods and Software*, 7(2):83–110, 1996.

[18] F. Leibfritz and E. W. Sachs. Inexact SQP interior point methods and large scale optimal control problems. *SIAM Journal on Control and Optimization*, 38(1):272–293, 1999.

[19] H. Maurer and H. D. Mittelmann. Optimization techniques for solving elliptic control problems with control and state constraints: Part 1. Boundary control. *Computational Optimization and Applications*, 16(1):29–55, 2000.

[20] E. Neufeld. *High Resolution Hyperthermia Treatment Planning*. PhD thesis, Swiss Federal Institute of Technology Zürich, Zürich, Switzerland, 2008.

[21] J. Nocedal and S. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer, Second edition, 2006.

[22] C. C. Paige and M. A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software*, 8(1):43–71, 1982.

[23] R. P. Pawlowski, J. P. Simonis, H. F. Walker, and J. N. Shadid. Inexact newton dogleg methods. *SIAM Journal on Numerical Analysis*, 46(4):2112–2132, 2008.

[24] M. J. D. Powell. A hybrid method for nonlinear equations. In P. Rabinowitz, editor, *Numerical Methods for Nonlinear Algebraic Equations*, pages 87–114, London, England, 1970. Gordon and Breach.

[25] E. E. Prudencio, R. H. Byrd, and X.-C. Cai. Parallel full space SQP Lagrange-Newton-Krylov-Schwarz algorithms for PDE-constrained optimization problems. *SIAM Journal on Scientific Computing*, 27(4):1305–1328, 2006.

[26] D. Ridzal. *Trust Region SQP Methods with Inexact Linear System Solves for Large-Scale Optimization*. PhD thesis, Rice University, Houston, TX, USA, 2006.

[27] O. Schenk and K. Gärtner. On fast factorization pivoting methods for sparse symmetric indefinite systems. *Electronic Transactions on Numerical Analysis*, 23:158–179, 2006.

[28] O. Schenk, A. Wächter, and M. Weiser. Inertia revealing preconditioning for large-scale nonconvex constrained optimization. *SIAM Journal on Scientific Computing*, 31(2):939–960, 2008.

[29] T. Steihaug. The conjugate gradient method and trust regions in large scale optimization. *SIAM Journal on Numerical Analysis*, 20(3):626–637, 1983.

[30] Ph. L. Toint. Nonlinear stepsize control, trust regions and regularizations for unconstrained optimization. Technical Report TR08-15, Department of Mathematics, FUNDP, University of Namur, Namur, Belgium, 2008.

[31] S. Ulbrich. Generalized SQP-methods with "parareal" time-domain decomposition for time-dependent PDE-constrained optimization. In L. Biegler, O. Ghattas, M. Heinkenschloss, D. Keyes, and B. van Bloemen Waanders, editors, *Real-Time PDE-Constrained Optimization*, Philadelphia, PA, USA, 2008. SIAM. (to appear).

[32] A. Wächter and L. T. Biegler. Failure of global convergence for a class of interior point methods for nonlinear programming. *Mathematical Programming*, 88(3):565–574, 2000.

[33] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming, Series A and B*, 106(1):25–57, 2006.

[34] Ipopt web page. `http://www.coin-or.org/Ipopt`.

[35] D. P. Young, W. P. Huffman, R. G. Melvin, C. L. Hilmes, and F. T. Johnson. Nonlinear elimination in aerodynamic analysis and design optimization. In L. T. Biegler, O. Ghattas, M. Heinkenschloss, and B. v. Bloemen-Waanders, editors, *Large-Scale PDE-Constrained Optimization*, pages 17–44, New York, NY, USA, 2003. Springer.