IMPLEMENTING A SMOOTH EXACT PENALTY FUNCTION FOR EQUALITY-CONSTRAINED NONLINEAR OPTIMIZATION[*]

RON ESTRIN[†], MICHAEL P. FRIEDLANDER[‡], DOMINIQUE ORBAN[§], AND MICHAEL A. SAUNDERS[¶]

Dedicated to Roger Fletcher

Abstract. We develop a general equality-constrained nonlinear optimization algorithm based on a smooth penalty function proposed by Fletcher (1970). Although it was historically considered to be computationally prohibitive in practice, we demonstrate that the computational kernels required are no more expensive than other widely accepted methods for nonlinear optimization. The main kernel required to evaluate the penalty function and its derivatives is solving a structured linear system. We show how to solve this system efficiently by storing a single factorization each iteration when the matrices are available explicitly. We further show how to adapt the penalty function to the class of factorization-free algorithms by solving the linear system iteratively. The penalty function therefore has promise when the linear system can be solved efficiently, e.g., for PDE-constrained optimization problems where efficient preconditioners exist. We discuss extensions including handling simple constraints explicitly, regularizing the penalty function, and inexact evaluation of the penalty function and its gradients. We demonstrate the merits of the approach and its various features on some nonlinear programs from a standard test set, and some PDE-constrained optimization problems.

1. Introduction.

We consider a penalty-function approach for solving general equality-constrained nonlinear optimization problems

$$\text{(NP)} \qquad \underset{x\in\mathbb{R}^n}{\text{minimize}} \;\; f(x) \;\; \text{subject to} \;\; c(x) = 0 \;:\; y,$$

where $f : \mathbb{R}^n \to \mathbb{R}$ and $c : \mathbb{R}^n \to \mathbb{R}^m$ are smooth functions ($m \le n$), and $y \in \mathbb{R}^m$ is the dual variable (the Lagrange multipliers). A smooth exact penalty function ϕ_σ is used to eliminate the constraints $c(x) = 0$. The penalty function is the Lagrangian $L(x, y) = f(x) - c(x)^T y$ with the vector $y = y_\sigma(x)$ treated as a function of x depending on a parameter $\sigma > 0$. Hence, the penalty function depends only on the primal variables x. It was first proposed by Fletcher (1970) for (NP). A long-held view is that Fletcher's penalty function is not practical because it is costly to compute (Bertsekas, 1975; Conn et al., 2000; Nocedal and Wright, 2006). In particular, Nocedal and Wright (2006, p.436) warn that "although this merit function has some interesting theoretical properties, it has practical limitations…". Our aim is to challenge that notion and to demonstrate that the computational kernels are no more expensive than other widely accepted methods for nonlinear optimization, such as sequential quadratic programming.

The penalty function is *exact* because local minimizers of (NP) are minimizers of the penalty function for all values of σ larger than a finite threshold σ^*. The

[*]Version of December 25, 2018

[†]Institute for Computational and Mathematical Engineering, Stanford University, Stanford, CA. (E-mail: restrin@stanford.edu.)

[‡]Department of Computer Science, University of British Columbia, Vancouver V6T 1Z4, BC, Canada (E-mail: mpf@cs.ubc.ca). The work of this author was supported by NSERC Discovery Grant 312104.

[§]GERAD and Department of Mathematics and Industrial Engineering, École Polytechnique, Montréal, QC, Canada (E-mail: dominique.orban@gerad.ca). The work of this author was supported by NSERC Discovery Grant 299010-04.

[¶]Systems Optimization Laboratory, Department of Management Science and Engineering, Stanford University, Stanford, CA. E-mail: E-mail: saunders@stanford.edu. Research partially supported by the National Institute of General Medical Sciences of the National Institutes of Health [award U01GM102098].

main computational kernel for evaluating the penalty function and its derivatives is the solution of a certain structured linear system. If the system matrix is available explicitly, we show how to factorize it once and re-use the factors to evaluate the penalty function and its derivatives. We also adapt the penalty function for *factorization-free* optimization by solving the linear system iteratively. This makes the penalty function particularly applicable for certain problem classes, such as PDE-constrained optimization problems when good preconditioners exist; see section 9.

1.1. Related work on penalty functions. Penalty functions have long been used to solve constrained problems by transforming them into unconstrained problems that penalize violations of feasibility. We provide a brief overview of common penalty methods and their relation to Fletcher's penalty $\phi_\sigma(x)$. More detail is given by di Pillo and Grippo (1984), Conn et al. (2000), and Nocedal and Wright (2006).

The simplest example is the quadratic penalty function (Courant, 1943), which removes the nonlinear constraints by adding $\frac{1}{2}\rho\|c(x)\|^2$ to the objective (for some $\rho > 0$). There are two main drawbacks: a sequence of optimization subproblems must be solved with increasing ρ, and a feasible solution is obtained only when $\rho \to \infty$. The subproblems become increasingly difficult to solve.

An alternative to smooth non-exact penalty functions is an exact non-smooth function such as the 1-norm penalty $\rho\|c(x)\|_1$ (Nocedal and Wright, 2006, §17.2). However, only non-smooth optimization methods apply, which typically exhibit slower convergence. Maratos (1978) further noted that non-smooth merit functions may reject steps and prevent quadratic convergence.

Another distinct approach is the class of augmented Lagrangian methods, independently introduced by Hestenes (1969) and Powell (1969). These minimize a sequence of augmented Lagrangians, $L_{\rho_k}(x, y_k) = L(x, y_k) \; + \; \frac{1}{2}\rho_k\|c(x)\|^2$. When y_k is optimal, $L_{\rho_k}(x, y_k)$ is exact for sufficiently large ρ_k, thus avoiding the stability issues of the quadratic penalty. However, a sequence of subproblems must be solved to drive y_k to optimality.

Although these penalty functions have often been successful in practice, in light of their drawbacks, a class of smooth exact penalty functions has been explored (di Pillo and Grippo, 1984; Zavala and Anitescu, 2014). With smooth exact penalty functions, constrained optimization problems such as (NP) can be replaced by a single smooth unconstrained optimization problem (provided the penalty parameter is sufficiently large). Approximate second-order methods can be applied to obtain at least superlinear local convergence. These methods are variations of minimizing the augmented Lagrangian, where either the multipliers are parametrized in terms of x, or they are kept independent and the gradient of the Lagrangian is penalized. The price for smoothness (as we find for ϕ_σ) is that a derivative of the penalty function requires a higher-order derivative from the original problem data. That is, evaluating ϕ_σ requires ∇f and ∇c; $\nabla\phi_\sigma$ requires $\nabla^2 f$ and $\nabla^2 c_i$; and so on. The third derivative terms are typically discarded during computation, but it can be shown that superlinear convergence is retained.

Fletcher (1970) introduced the class of smooth exact penalty functions from which ϕ_σ originates. Extensions and variations of this class have been explored by several authors, whose contributions are described by Conn et al. (2000, §14.6). However, Fletcher (1970) envisioned his method being applied to small problems, and assumed "the matrices in the problem are non-sparse". Further, most developments surrounding this method focused on linesearch schemes that require computing an explicit Hessian approximation and using it to compute a Newton direction. One of our goals is to show how to adapt the method to large-scale problems by taking advantage of computational advances made since Fletcher's proposal. Improved sparse matrix factorizations and iterative methods for solving linear systems, and modern Newton-CG trust-region methods (Ph. L. Toint, 1981; Steihaug, 1983), play a key role in the efficient implementation of his penalty function. We also show how regularization can be used to accommodate constraint dependency, and explain how to take advantage of inexact evaluations of functions and gradients.

1.2. Outline. We introduce the penalty function in section 2 and describe its properties and derivatives in section 3. In section 4 we discuss options for efficiently evaluating the penalty function and its derivatives. We then discuss modifications of the penalty function in section 5–6 to take advantage of linear constraints and to regularize the penalty function if the constraint Jacobian is rank-deficient. In some applications, it may be necessary to solve large linear systems inexactly, and we show in section 7 how the resulting imprecision can be accommodated. Other practical matters are described in section 8. We apply the penalty function to several optimization problems in section 9, and conclude with future research directions in section 10.

2. The penalty function for equality constraints.

For (NP), Fletcher's penalty function is

$$\phi_\sigma(x) := f(x) - c(x)^T y_\sigma(x), \tag{1}$$

where $y_\sigma(x)$ are Lagrange multiplier estimates defined with other items as

$$y_\sigma(x) := \arg\min_y \ \tfrac{1}{2}\|A(x)y - g(x)\|_2^2 + \sigma c(x)^T y, \qquad g(x) := \nabla f(x), \tag{2a}$$

$$A(x) := \nabla c(x) = \begin{bmatrix} g_1(x) & \dots & g_m(x) \end{bmatrix}, \qquad g_i(x) := \nabla c_i(x), \tag{2b}$$

$$Y_\sigma(x) := \nabla y_\sigma(x). \tag{2c}$$

Note that A and Y_σ are n-by-m matrices. The form of $y_\sigma(x)$ is reminiscent of the variable-projection algorithm of Golub and Pereyra (1973) for separable nonlinear least-squares problems.

We assume that (NP) satisfies some variation of the following assumptions:

- (A1) f and c are twice continuously differentiable and have Lipschitz second-derivatives (A1a), or are three-times continuously differentiable (A1b).
- (A2) The linear independence constraint qualification (LICQ) is satisfied at stationary points of (NP) (A2a), or for all n-vectors x (A2b). (LICQ is satisfied at a point x if the vectors $\{\nabla c_i(x)\}_{i=1}^m$ are linearly independent.)
- (A3) The problem is feasible, i.e., there exists x such that $c(x) = 0$.

Assumption (A1b) ensures that ϕ_σ has two continuous derivatives and is typical for smooth exact penalty functions (Bertsekas, 1982, Proposition 4.16). However, we use at most two derivatives of f and c throughout. We typically assume (A1b) to simplify the discussion, but this assumption can often be weakened to (A1a). We also initially assume that (NP) satisfies (A2b) so that $Y_\sigma(x)$ and $y_\sigma(x)$ are uniquely defined. We relax this assumption to (A2a) in section 6.

2.1. Notation. Denote x^* as a local minimizer of (NP), with corresponding dual solution y^*. Let $H(x) = \nabla^2 f(x)$, $H_i(x) = \nabla^2 c_i(x)$, and define

$$g_L(x,y) := g(x) - A(x)y, \qquad g_\sigma(x) := g_L(x, y_\sigma(x)), \tag{3a}$$

$$H_L(x,y) := H(x) - \sum_{i=1}^m y_i H_i(x), \qquad H_\sigma(x) := H_L(x, y_\sigma(x)) \tag{3b}$$

as the gradient and Hessian of $L(x,y)$ evaluated at x and y, or evaluated at $y_\sigma(x)$. We also define the matrix operators

$$S(x,v) := \nabla_x[A(x)^T v] = \nabla_x \begin{bmatrix} g_1(x)^T v \\ \vdots \\ g_m(x)^T v \end{bmatrix} = \begin{bmatrix} v^T H_1(x) \\ \vdots \\ v^T H_m(x) \end{bmatrix},$$

$$T(x,w) := \nabla_x[A(x)w] = \nabla_x \left[\sum_{i=1}^m w_i g_i(x)\right] = \sum_{i=1}^m w_i H_i(x),$$

where $v \in \mathbb{R}^n$, $w \in \mathbb{R}^m$, and $T(x,w)$ is a symmetric matrix. The operation of multiplying the adjoint of S with a vector w is described by

$$S(x,v)^T w = \left[\sum_{i=1}^m w_i H_i(x)\right] v = T(x,w)v = T(x,w)^T v\,.$$

If $A(x)$ has full rank m, the operators

$$P(x) := A(x)\big(A(x)^T A(x)\big)^{-1} A(x)^T \quad \text{and} \quad \bar{P}(x) := I - P(x) \tag{4}$$

define, respectively, orthogonal projectors onto range$(A(x))$ and its complement. More generally, for a matrix M, respectively define P_M and $\bar{P}_M$ as the orthogonal projectors onto range(M) and ker(M). We define $M^\dagger$ as the Moore-Penrose pseudoinverse, where $M^\dagger = (M^T M)^{-1} M^T$ if M has full column-rank.

Unless otherwise indicated, $\|\cdot\|$ is the 2-norm for vectors and matrices. For M positive definite, $\|u\|_M^2 = u^T M u$ is the energy norm. Define $\mathbb{1}$ as the vector of all ones.

3. Properties of the penalty function.

We show how the penalty function $\phi_\sigma(x)$ naturally expresses the optimality conditions of (NP). We also give explicit expressions for the threshold value of the penalty parameter σ.

3.1. Derivatives of the penalty function.

The gradient and Hessian of ϕ_σ may be written as

$$\nabla\phi_\sigma(x) = g_\sigma(x) - Y_\sigma(x)c(x), \tag{5a}$$

$$\nabla^2\phi_\sigma(x) = H_\sigma(x) - A(x)Y_\sigma(x)^T - Y_\sigma(x)A(x)^T - \nabla_x\left[Y_\sigma(x)c\right], \tag{5b}$$

where the last term $\nabla_x[Y_\sigma(x)c]$ purposely drops the argument on c to emphasize that this gradient is made on the product $Y_\sigma(x)c$ with $c := c(x)$ held fixed. This term involves third derivatives of f and c, and as we shall see, it is convenient and computationally efficient to ignore it. We leave it unexpanded.

3.2. Optimality conditions.

The penalty function ϕ_σ is closely related to the Lagrangian $L(x,y)$ associated with (NP). To make this connection clear, we define the Karush-Kuhn-Tucker (KKT) optimality conditions for (NP) in terms of formulas related to ϕ_σ and its derivatives. From the definition of ϕ_σ and y_σ and the derivatives (5), the following definitions are equivalent to the KKT conditions.

DEFINITION 1 (First-order KKT point). *A point x^* is a first-order KKT point of (NP) if for any $\sigma \ge 0$ the following hold:*

$$c(x^*) = 0, \tag{6a}$$

$$\nabla\phi_\sigma(x^*) = 0. \tag{6b}$$

The Lagrange multipliers associated with x^* are $y^* := y_\sigma(x^*)$.

DEFINITION 2 (Second-order KKT point). *The first-order KKT point x^* satisfies the second-order necessary KKT condition for* (NP) *if for any $\sigma \geq 0$,*

$$p^T \nabla^2 \phi_\sigma(x^*) p \geq 0 \quad \text{for all } p \text{ such that } A(x^*)^T p = 0, \tag{7}$$

i.e., $\bar{P}(x^*)\nabla^2\phi_\sigma(x^*)\bar{P}(x^*) \succeq 0$. The condition is sufficient if the inequality is strict.

The second-order KKT condition says that at x^*, ϕ_σ has nonnegative curvature along directions in the tangent space of the constraints. However, at x^*, increasing σ will increase curvature along the normal cone of the feasible set. We derive a threshold value for σ that causes ϕ_σ to have nonnegative curvature at a second-order KKT point x^*, as well as a condition on σ that ensures stationary points of ϕ_σ are primal feasible. For a given first- or second-order KKT pair (x^*, y^*) of (NP), we define

$$\sigma^* := \tfrac{1}{2}\lambda_{\max}\left(P(x^*)H_L(x^*, y^*)P(x^*)\right), \tag{8}$$

where $\lambda_{\max}(\cdot)$ is the maximum of the largest eigenvalue and zero.

LEMMA 3. *If $c(x) \in \operatorname{range}(A(x)^T)$, then $y_\sigma(x)$ satisfies*

$$A(x)^T A(x) y_\sigma(x) = A(x)^T g(x) - \sigma c(x). \tag{9}$$

Further, if $A(x)$ has full rank, then

$$A(x)^T A(x) Y_\sigma(x)^T = A(x)^T [H_\sigma(x) - \sigma I] + S(x, g_\sigma(x)). \tag{10}$$

Proof. For any x, the necessary and sufficient optimality conditions for (2a) give (9). By differentiating both sides of (9), we obtain

$$S(x, A(x)y_\sigma(x)) + A(x)^T\left[T(x, y_\sigma(x)) + A(x)Y_\sigma(x)^T\right] = S(x, g(x)) + A(x)^T[H(x) - \sigma I].$$

From definitions (3), we obtain (10). □

THEOREM 4 (Threshold penalty value).
Suppose $\nabla\phi_\sigma(\bar{x}) = 0$ for some $\bar{x}$, and let x_1^* and x_2^* be a first-order and a second-order necessary KKT point, respectively, for (NP). *Let σ^* be defined as in* (8). *Then*

$$\sigma > \|A(\bar{x})^T Y_\sigma(\bar{x})\| \implies g(\bar{x}) = A(\bar{x})y_\sigma(\bar{x}), \quad c(\bar{x}) = 0; \tag{11a}$$

$$\sigma \geq \|A(x_1^*)Y_\sigma(x_1^*)^T\| \implies \sigma \geq \sigma^*; \tag{11b}$$

$$\nabla^2\phi_\sigma(x_2^*) \succeq 0 \iff \sigma \geq \sigma^*. \tag{11c}$$

If x_2^* is second-order sufficient, then the inequalities in (11c) *hold strictly.*

Proof. We prove (11a), (11c), and (11b) in order.

Proof of (11a): The condition $\nabla\phi_\sigma(\bar{x}) = 0$ implies that

$$g(\bar{x}) = A(\bar{x})y_\sigma(\bar{x}) + Y_\sigma(\bar{x})c(\bar{x}).$$

Substituting (9) evaluated at $\bar{x}$ into this equation yields, after simplifying,

$$A(\bar{x})^T Y_\sigma(\bar{x}) c(\bar{x}) = \sigma c(\bar{x}).$$

Taking norms of both sides and using the triangle inequality gives the inequality $\sigma\|c(\bar{x})\| \leq \|A(\bar{x})^T Y_\sigma(\bar{x})\|\,\|c(\bar{x})\|$, which immediately implies that $c(\bar{x}) = 0$. The condition $\nabla\phi_\sigma(\bar{x}) = 0$ then becomes $g_\sigma(\bar{x}) = 0$.

Proof of (11c): Because x_2^* satisfies (6), we have $g_\sigma(x_2^*) = 0$ and $y^* = y_\sigma(x)$, independently of σ. It follows from (10), $H_L(x_2^*, y^*) = H_\sigma(x_2^*)$, and the definition of the projector $P = P(x_2^*)$ that

$$A(x_2^*)Y_\sigma(x_2^*)^T = P(H_L(x_2^*, y^*) - \sigma I). \tag{12}$$

We substitute this equation into (5b) and use the relation $P + \bar{P} = I$ to obtain

$$\nabla^2\phi_\sigma(x_2^*) = \bar{P}H_L(x_2^*, y^*)\bar{P} - PH_L(x_2^*, y^*)P + 2\sigma P. \tag{13}$$

Note that $\bar{P}H_L(x_2^*, y^*)\bar{P} \succeq 0$ because x_2^* is a second-order KKT point, so σ needs to be sufficiently large that $2\sigma P - PH_L(x_2^*, y^*)P \succeq 0$, which is equivalent to $\sigma \geq \sigma^*$.

Proof of (11b): With x_1^* in (12), $y^* = y_\sigma(x_1^*)$, and the properties of P, we have

$$\begin{aligned}
\sigma \geq \|A(x_1^*)Y_\sigma(x_1^*)^T\| &= \|P(H_L(x_1^*, y^*) - \sigma I)\| \\
&\geq \|P(H_L(x_1^*, y^*) - \sigma I)P\| \\
&\geq \|PH_L(x_1^*, y^*)P\| - \sigma\|P\| \\
&\geq 2\sigma^* - \sigma.
\end{aligned}$$

Thus, $\sigma \geq \sigma^*$ as required. ∎

According to (11c), if x^* is a second-order KKT point, there exists a threshold value σ^* such that ϕ_σ has nonnegative curvature for $\sigma \geq \sigma^*$. Unfortunately, as for many exact penalty functions, Theorem 4 does not discount the possibility of stationary points of $\phi_\sigma(x)$ that are not feasible points of (NP); for an example, see Appendix A.1. However, we rarely encounter this in practice, and usually minimizers of $\phi_\sigma(x)$ correspond to feasible (and therefore optimal) points of (NP).

3.3. Additional quadratic penalty.

In light of Theorem 4, it is somewhat unsatisfying that local minimizers of $\phi_\sigma(x)$ might not be local minimizers of (NP). We may add a quadratic penalty term to promote feasibility, and under mild conditions ensure that minimizers of ϕ_σ are KKT points of (NP). Like Fletcher (1970), we define

$$\phi_{\sigma,\rho}(x) := \phi_\sigma(x) + \tfrac{1}{2}\rho\|c(x)\|^2 = f(x) - [y_\sigma(x) - \tfrac{1}{2}\rho c(x)]^T c(x). \tag{14}$$

The multiplier estimates are now shifted by the constraint violation, similar to an augmented Lagrangian. All expressions for the derivatives follow as before with an additional term from the quadratic penalty.

Theorem 5 (Threshold penalty value for quadratic penalty).
Let $\mathcal{S} \subset \mathbb{R}^n$ be a compact set, and suppose that $\sigma_{\min}(A(x)) \geq \lambda > 0$ for all $x \in \mathcal{S}$. Then for any $\sigma \geq 0$ there exists $\rho^*(\sigma) > 0$ such that for all $\rho > \rho^*(\sigma)$, if $\nabla\phi_{\sigma,\rho}(\bar{x}) = 0$ and $\bar{x} \in \mathcal{S}$, then $\bar{x}$ is a first-order KKT point for (NP).

Proof. The condition $\nabla\phi_{\sigma,\rho}(\bar{x}) = 0$ implies that

$$g(\bar{x}) - A(\bar{x})y_\sigma(\bar{x}) - Y_\sigma(\bar{x})c(\bar{x}) = \rho A(\bar{x})c(\bar{x}).$$

We premultiply with $A(\bar{x})^T$ and use (9) to obtain

$$\left(\sigma I - A(\bar{x})^T Y_\sigma(\bar{x})\right) c(\bar{x}) = \rho A(\bar{x})^T A(\bar{x}) c(\bar{x}). \tag{15}$$

The left-hand side of (15) is a continuous matrix function with finite supremum $R(\sigma) := \sup_{x\in\mathcal{S}} \|\sigma I - A(x)^T Y(x)\|$ defined over the compact set $\mathcal{S}$. We now define $\rho^*(\sigma) := R(\sigma)/\lambda^2$, so that for $\rho > \rho^*(\sigma)$, if $c(\bar{x}) \neq 0$,

$$\begin{aligned} R(\sigma)\|c(\bar{x})\| &\geq \|\sigma I - A(\bar{x})^T Y_\sigma(\bar{x})\| \cdot \|c(\bar{x})\| \\ &\geq \| \left(\sigma I - A(\bar{x})^T Y_\sigma(\bar{x})\right) c(\bar{x})\| \\ &= \rho \|A(\bar{x})^T A(\bar{x}) c(\bar{x})\| \\ &\geq \rho\lambda^2 \|c(\bar{x})\| > R(\sigma)\|c(\bar{x})\|, \end{aligned}$$

which is a contradiction, implying $\|c(\bar{x})\| = 0$, so that $\bar{x}$ is feasible for (NP). Because $c(\bar{x}) = 0$ and $\nabla\phi_\sigma(x) = \nabla\phi_{\sigma,\rho}(\bar{x}) = 0$, $\bar{x}$ is a first-order KKT point. ∎

We briefly consider the case $\sigma = 0$ and $\rho > 0$. The threshold value to ensure positive semidefiniteness of $\nabla^2\phi_{\sigma,\rho}$ at a second-order KKT pair (x^*, y^*) to (NP) is

$$\rho^* = \lambda_{\max}\left(A(x^*)^\dagger H_L(x^*, y^*) A(x^*)^\dagger\right).$$

This threshold parameter is more difficult to interpret in terms of the original problem data compared to σ^* because of the pseudoinverse. We give a theorem analogous to Theorem 4, but omit the proof as it is nearly identical.

THEOREM 6. *Suppose $\sigma = 0$ and $\rho \geq 0$. Let $\nabla\phi_{\sigma,\rho}(\bar{x}) = 0$ for some $\bar{x}$, and let x^* be a second-order necessary KKT point for* (NP). *Then*

$$\rho > \|A(\bar{x})^\dagger Y_\sigma(\bar{x})\| \implies g(\bar{x}) = A(\bar{x}) y_\sigma(\bar{x}), \quad c(\bar{x}) = 0; \tag{16a}$$

$$\nabla^2\phi_\sigma(x^*) \succeq 0 \iff \rho \geq \rho^*. \tag{16b}$$

If x^* is second-order sufficient, the inequalities in (16b) *hold strictly.*

Using $\rho > 0$ can help cases where attaining feasibility is problematic for moderate values of σ. For simplicity we let $\rho = 0$ from now on, because it is trivial to evaluate $\phi_{\sigma,\rho}$ and its derivatives if one can compute ϕ_σ.

3.4. Scale invariance. Note that ϕ_σ is invariant under diagonal scaling of the constraints, i.e., if $c(x)$ is replaced by $Dc(x)$ for some diagonal matrix D, then ϕ_σ is unchanged. It is an attractive property for ϕ_σ and σ^* to be independent of some choices in model formulation, like the Lagrangian. However, $\phi_{\sigma,\rho}$ with $\rho > 0$ is not scale invariant, like the augmented Lagrangian, because of the quadratic term. Therefore, constraint scaling is an important consideration if $\phi_{\sigma,\rho}$ is to be used.

4. Evaluating the penalty function.

4. Evaluating the penalty function. The main challenge in evaluating ϕ_σ and its gradient is solving the shifted least-squares problem (2a) in order to compute $y_\sigma(x)$, and computing the gradient $Y_\sigma(x)$. Below we show it is possible to compute products $Y_\sigma(x)v$ and $Y_\sigma(x)^T u$ by solving structured linear systems involving the matrix used to compute $y_\sigma(x)$. If direct methods are used, a single factorization that gives the solution (2a) is sufficient for all products.

For this section, it is convenient to drop the arguments on the various functions and assume they are all evaluated at a point x for some parameter σ. For example, $y_\sigma = y_\sigma(x)$, $A = A(x)$, $Y_\sigma = Y_\sigma(x)$, $H_\sigma = H_\sigma(x)$, $S_\sigma = S(x, g_\sigma(x))$, etc. We also express (10) using the shorthand notation

$$A^T A Y_\sigma^T = A^T [H_\sigma - \sigma I] + S_\sigma. \tag{17}$$

We first describe how to compute products $Y_\sigma u$ and $Y_\sigma^T v$, then describe how to put those pieces together to evaluate the penalty function and its derivatives.

4.1. Computing the product $\mathbf{Y_\sigma u}$. For a given $u \in \mathbb{R}^m$, we premultiply (17) by $u^T(A^TA)^{-1}$ to obtain

$$\begin{aligned} Y_\sigma u &= [H_\sigma - \sigma I]A(A^TA)^{-1}u + S_\sigma^T(A^TA)^{-1}u \\ &= [H_\sigma - \sigma I]v - S_\sigma^T w, \end{aligned}$$

where we define $w = -(A^TA)^{-1}u$ and $v = -Aw$. Observe that w and v solve the system

$$\begin{bmatrix} I & A \\ A^T & \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix} = \begin{bmatrix} 0 \\ u \end{bmatrix}. \tag{18}$$

Algorithm 1 formalizes the process.

Algorithm 1 Computing the matrix-vector product $Y_\sigma u$

```
1: $(v, w) \leftarrow$ solution of (18)
2: return $[H_\sigma - \sigma I]v - S_\sigma^T w$
```

4.2. Computing the product $\mathbf{Y_\sigma^T v}$. Multiplying both sides of (17) on the right by v gives

$$A^TA(Y_\sigma^T v) = A^T([H_\sigma - \sigma I]v) + (S_\sigma v).$$

The required product $u = Y_\sigma^T v$ is in the solution of the system

$$\begin{bmatrix} I & A \\ A^T & \end{bmatrix} \begin{bmatrix} r \\ u \end{bmatrix} = \begin{bmatrix} [H_\sigma - \sigma I]v \\ -S_\sigma v \end{bmatrix}. \tag{19}$$

Algorithm 2 formalizes the process.

Algorithm 2 Computing the matrix-vector product $Y_\sigma^T v$

```
1: Evaluate $[H_\sigma - \sigma I]v$ and $S_\sigma v$
2: $(r, u) \leftarrow$ solution of (19)
3: return $u$
```

4.3. Computing multipliers and first derivatives. The multiplier estimates y_σ can be obtained from the optimality conditions for (2a):

$$\begin{bmatrix} I & A \\ A^T & \end{bmatrix} \begin{bmatrix} g_\sigma \\ y_\sigma \end{bmatrix} = \begin{bmatrix} g \\ \sigma c \end{bmatrix}, \tag{20}$$

which also gives g_σ. Algorithm 1 then gives $Y_\sigma c$ and hence $\nabla\phi_\sigma$ in (5a).

Observe that we can re-order operations to take advantage of specialized solvers. Consider the pair of systems

$$\begin{bmatrix} I & A \\ A^T & \end{bmatrix} \begin{bmatrix} d \\ y \end{bmatrix} = \begin{bmatrix} g \\ 0 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} I & A \\ A^T & \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix} = \begin{bmatrix} 0 \\ c \end{bmatrix}. \tag{21}$$

We have $g_\sigma = d + \sigma v$ and $y_\sigma = y + \sigma w$, while the computation of $Y_\sigma c$ is unchanged. The systems in (21) correspond to pure least-squares and least-norm problems respectively. Specially tailored solvers may be used to improve efficiency or accuracy. This is further explored in section 4.5.

4.4. Computing second derivatives. We can approximate $\nabla^2\phi_\sigma$ using (5b) and (10) in two ways according to

$$\text{(22a)}\quad \nabla^2\phi_\sigma \approx B_1 := H_\sigma - AY_\sigma^T - Y_\sigma A^T$$
$$= H_\sigma - \widetilde{P}H_\sigma - H_\sigma\widetilde{P} + 2\sigma\widetilde{P} - A(A^TA)^{-1}S_\sigma - S_\sigma^T(A^TA)^{-1}A$$
$$\text{(22b)}\quad \nabla^2\phi_\sigma \approx B_2 := H_\sigma - \widetilde{P}H_\sigma - H_\sigma\widetilde{P} + 2\sigma\widetilde{P},$$

where $\widetilde{P} = A(A^TA)^{-1}A^T$. Note that $\widetilde{P} = P_A$ here, but this changes when regularization is used; see section 6. The first approximation ignores $\nabla[Y_\sigma(x)c]$ in (5b), while the second ignores $S_\sigma = S(x, g_\sigma(x))$. Because we expect $c(x) \to 0$ and $g_\sigma(x) \to 0$, B_1 and B_2 are similar to Gauss-Newton approximations to $\nabla^2\phi_\sigma(x)$, and as Fletcher (1973, Theorem 2) shows, using them in a Newton-like scheme is sufficient for quadratic convergence if (A1a) is satisfied.

Because $\widetilde{P}$ is a projection on range(A), we can compute products $\widetilde{P}u$ by solving

$$\text{(23)}\qquad \begin{bmatrix} I & A \\ A^T & \end{bmatrix}\begin{bmatrix} p \\ q \end{bmatrix} = \begin{bmatrix} u \\ 0 \end{bmatrix}$$

and setting $\widetilde{P}u \leftarrow u - p$. Note that with regularization, the $(2,2)$ block of this system is modified and $\widetilde{P}$ is no longer a projection; see section 6.

The approximations (22a) and (22b) trade Hessian accuracy for computational efficiency. If the operator $S(x, v)$ is not immediately available (or not efficiently implemented), it may be avoided. Using B_2 requires only least-square solves, which allows us to apply specialized solvers (e.g., LSQR (Paige and Saunders, 1982)), which cannot be done when products with Y_σ^T are required.

4.5. Solving the augmented linear system. We discuss some approaches to solving linear systems of the form

$$\text{(24)}\qquad \mathcal{K}\begin{bmatrix} p \\ q \end{bmatrix} = \begin{bmatrix} w \\ z \end{bmatrix}, \qquad \mathcal{K} := \begin{bmatrix} I & A \\ A^T & -\delta^2 I \end{bmatrix},$$

which have appeared repeatedly in this section. Although $\delta = 0$ so far, we now look ahead to regularized systems because they require only minor modification. Let (p^*, q^*) solve (24). Define $A_\delta := \begin{bmatrix} A^T & \delta I \end{bmatrix}^T$ when $\delta > 0$; otherwise $A_\delta := A$.

Conceptually it is not important how this system is solved as long as it is with sufficient accuracy. However, this is the most computationally intensive part of using ϕ_σ. Different solution methods have different advantages and limitations, depending on the size and sparsity of A, whether A is available explicitly, and the prescribed solution accuracy.

One option is to use direct methods: factorize $\mathcal{K}$ once per iteration and use the factors to solve with each right-hand side. Several factorization-based approaches can be employed with various advantages and drawbacks; see Appendix A.2.

In line with the goal of creating a factorization-free solver for minimizing ϕ_σ, we discuss iterative methods for solving (24), particularly Krylov subspace solvers. This approach has two potential advantages: if a good preconditioner $\mathcal{P} \approx A^TA$ is available, then solving (24) could be much more efficient than with direct methods, and we can take advantage of solvers using inexact function values, gradients or Hessian products by solving (24) approximately.

When $z = 0$, (24) is a (regularized) least-squares problem: $\min_q \|A_\delta q - w_\delta\|$. We use LSQR (Paige and Saunders, 1982), which ensures that the error in iterates p_k and

q_k decreases monotonically at every iteration. (Hestenes and Stiefel (1952) show this for CG, and LSQR is equivalent to CG on the normal equations.) Furthermore, Estrin et al. (2017) provide a way to compute an upper bound on $\|p^* - p_k\|$ and $\|q^* - q_k\|$ via LSLQ when given an understimate of $\sigma_{\min}(A\mathcal{P}^{-1/2})$. (Note that the error norm for q depends on the preconditioner.) Further discussion is in section 9.

When $w = 0$, (24) is a least-norm problem: $\min_p \|p\|$ s.t. $A_\delta^T p = z$. We then use CRAIG (Craig, 1955) because it minimizes the error in each Krylov subspace. Given the same underestimate of $\sigma_{\min}(A\mathcal{P}^{-1/2})$, Arioli (2013) and Estrin et al. (2018) give a way to bound the error norms for p and q.

Recall that ϕ_σ and $\nabla\phi_\sigma$ can be computed by solving only least-squares and least-norm problems (only one of w and z is nonzero at a time). Furthermore, if (22b) is used, the remaining solves with $\mathcal{K}$ are least-squares solves. If both w and z are nonzero (for products with Y_σ^T), we can shift the right-hand side of (24) and solve the system

$$\mathcal{K}\begin{bmatrix}\bar{p}\\ q\end{bmatrix} = \begin{bmatrix}0\\ z - A^T w\end{bmatrix}, \qquad p = \bar{p} + w.$$

Thus, (24) can be solved by CRAIG or LNLQ (Arioli, 2013; Estrin et al., 2018) or other least-norm solvers.

Although $\mathcal{K}$ is symmetric indefinite, we do not recommend methods such as MINRES or SYMMLQ (Paige and Saunders, 1975). Orban and Arioli (2017) show that if full-space methods are applied directly to $\mathcal{K}$ then every other iteration of the solver makes little progress. However, if solves with $\mathcal{P}$ can only be performed approximately, it may be necessary to apply flexible variants of nonsymmetric full-space methods to $\mathcal{K}$, such as GMRES (Saad and Schultz, 1986).

5. Maintaining explicit constraints.

We consider a variation of (NP) where some of the constraints $c(x)$ are easy to maintain explicitly; for example, some are linear. We can then maintain feasibility for a subset of the constraints, the contours of the ϕ_σ are simplified, and as we show soon, the threshold penalty parameter σ^* is decreased. We discuss the case where some of the constraints are linear, but it is possible to extend the theory to any type of constraint.

Consider the problem

$$\text{(NP-EXP)} \qquad \underset{x\in\mathbb{R}^n}{\text{minimize}} \;\; f(x) \;\; \text{subject to} \;\; c(x) = 0, \;\; B^T x = d,$$

where we have nonlinear constraints $c(x) \in \mathbb{R}^{m_1}$ and linear constraints $B^T x = d$ with $B \in \mathbb{R}^{n\times m_2}$, so that $m_1 + m_2 = m$. We assume that (NP-EXP) at least satisfies (A2a), so that B has full column rank. We define the penalty function problem to be

$$(25)\qquad \begin{aligned} &\underset{x\in\mathbb{R}^n}{\text{minimize}} \quad \phi_\sigma(x) := f(x) - c(x)^T y_\sigma(x) \quad \text{subject to} \quad B^T x = d,\\ &\begin{bmatrix} y_\sigma(x)\\ w_\sigma(x)\end{bmatrix} := \underset{y,w}{\arg\min}\, \tfrac{1}{2}\|A(x)y + Bw - g(x)\|^2 + \sigma \begin{bmatrix} c(x)\\ B^T x - d\end{bmatrix}^T \begin{bmatrix} y\\ w\end{bmatrix}, \end{aligned}$$

which is similar to (1) except the linear constraints are not penalized in $\phi_\sigma(x)$, and the penalty function is minimized subject to the linear constraints. A possibility is to penalize the linear constraints as well, while keeping the linear constraints explicit; however, penalizing the linear constraints in $\phi_\sigma(x)$ introduces additional nonlinearity, and if all constraints are linear, it makes sense that the penalty function reduces to (NP-EXP).

For a given first- or second-order KKT solution (x^*, y^*), the threshold penalty parameter becomes

$$\sigma^* := \tfrac{1}{2}\lambda_{\max}\left(\bar{P}_B P_C H_L(x^*, y^*) P_C \bar{P}_B\right) \le \tfrac{1}{2}\lambda_{\max}\left(P_C H_L(x^*, y^*) P_C\right), \tag{26}$$

where $C(x) = \begin{bmatrix} A(x) & B \end{bmatrix}$ is the Jacobian for all constraints. Inequality (26) holds because $\bar{P}_B$ is an orthogonal projector. If the linear constraints were not explicit, the threshold value would be the right-most term in (26). Intuitively, the threshold penalty value decreases by the amount of the top eigenspace of the Lagrangian Hessian that lies in the range of B^T, because positive semidefiniteness of $\nabla^2\phi_\sigma(x^*)$ along that space is guaranteed by the underlying solver.

The following result is analogous to Theorem 4 with the smaller threshold value.

THEOREM 7 (Threshold penalty value with explicit constraints). *Suppose $\bar{x}$ is a first-order necessary KKT point for (25):*

$$B^T\bar{x} = d,$$
$$\nabla\phi_\sigma(\bar{x}) = Bw^*,$$

and let x_1^* and x_2^* be first- and second-order necessary KKT points respectively for (NP-EXP). Then for all $p \neq 0$ such that $B^Tp = 0$,

$$\sigma > \|A(\bar{x})^T\bar{P}_B Y_\sigma(\bar{x})\| \implies g(\bar{x}) = A(\bar{x})y_\sigma(\bar{x}) + Bw_\sigma(\bar{x}), \quad c(\bar{x}) = 0; \tag{27a}$$
$$\sigma \ge \|\bar{P}_B A(x_1^*) Y_\sigma(x_1^*)^T\| \implies \sigma \ge \sigma^*; \tag{27b}$$
$$p^T\nabla^2\phi_\sigma(x_2^*)p \succeq 0 \iff \sigma \ge \sigma^*. \tag{27c}$$

If x_2^* is second-order sufficient, the inequalities in (27c) hold strictly.

The proof of the theorem, and details of evaluating the penalty function with explicit constraints, are given in Appendix A.3.

6. Regularization. Even if $A(x^*)$ has full column rank, $A(x)$ might have low column rank or small singular values away from the solution. If $A(x)$ is rank-deficient and $c(x)$ is not in the range of $A(x)^T$, then $y_\sigma(x)$ and $\phi_\sigma(x)$ are undefined. Even if $A(x)$ has full column rank but is close to rank-deficiency, the linear systems (18)–(20) and (23) are ill-conditioned, threatening inaccurate solutions and impeded convergence.

We modify ϕ_σ by changing the definition of the multiplier estimates in (2a) to solve a regularized shifted least-squares problem with regularization parameter $\delta > 0$:

$$\phi_\sigma(x;\delta) := f(x) - c(x)^T y_\sigma(x;\delta) \tag{28}$$
$$y_\sigma(x;\delta) := \operatorname*{arg\,min}_y \ \tfrac{1}{2}\|A(x)y - g(x)\|_2^2 + \sigma c(x)^T y + \tfrac{1}{2}\delta^2\|y\|_2^2. \tag{29}$$

This modification is similar to the exact penalty function of di Pillo and Grippo (1986). The regularization term $\tfrac{1}{2}\delta^2\|y\|_2^2$ ensures that the multiplier estimate $y_\sigma(x;\delta)$ is always defined even when $A(x)$ is rank-deficient. The only computational change is that the $(2,2)$ block of the matrices in (18)–(20) and (23) is now $-\delta^2 I$.

Besides improving $\operatorname{cond}(\mathcal{K})$, $\delta > 0$ has the advantage of making $\mathcal{K}$ symmetric quasi-definite. Vanderbei (1995) shows that any symmetric permutation of such a matrix possesses an LDL^T factorization with L unit lower triangular and D diagonal indefinite. Result 2 of Gill et al. (1996) implies that the factorization is stable as long as δ is sufficiently far from zero. Various authors propose regularized matrices of this type to stabilize optimization methods in the presence of degeneracy. In particular, Wright (1998) accompanies his discussion with an update scheme for δ that guarantees fast asymptotic convergence.

We continue to assume that (NP) satisfies (A1b), but we now replace (A2b) by (A2a). For a given isolated local minimum x^* of (NP), σ sufficiently large, and δ sufficiently small, we define

$$x(\delta) := \arg\min_x \|x - x^*\| \ \text{ such that } x \text{ is a local-min of } \phi_\sigma(x;\delta)$$

for use as an analytical tool in the upcoming discussion.

Note that for $\delta > 0$, we would not expect that $x(\delta) = x^*$, but we want to ensure that $x(\delta) \to x^*$ as $\delta \to 0$. Note that for x such that $y_\sigma(x)$ is defined,

$$\begin{aligned} y_\sigma(x;\delta) &= (A(x)^T A(x) + \delta^2 I)^{-1} A(x)^T A(x) y_\sigma(x) \\ &= y_\sigma(x) - \delta^2 (A(x)^T A(x) + \delta^2 I)^{-1} y_\sigma(x). \end{aligned}$$

Therefore for x such that $\phi_\sigma(x)$ is defined, we can write the regularized penalty function as a perturbation of the unregularized one:

$$\begin{aligned} \phi_\sigma(x;\delta) &= f(x) - c(x)^T y_\sigma(x;\delta) \\ &= f(x) - c(x)^T y_\sigma(x) + \delta^2 c(x)^T (A(x)^T A(x) + \delta^2 I)^{-1} y_\sigma(x) \\ &= \phi_\sigma(x) + \delta^2 P_\delta(x), \end{aligned} \tag{30}$$

where $P_\delta(x) = c(x)^T (A(x)^T A(x) + \delta^2 I)^{-1} y_\sigma(x)$. By (A1b), P_δ is bounded and has at least two continuous derivatives in a neighbourhood of x^*.

Theorem 8. *Suppose (A1b) and (A2a) are satisfied, x^* is a second-order KKT point for (NP), and $\nabla^2 \phi_\sigma(x^*) \succ 0$. Then there exists $\bar{\delta} > 0$ such that $x(\delta)$ is a $\mathcal{C}_1$ function for $0 \le \delta < \bar{\delta}$. In particular, $\|x(\delta) - x^*\| = O(\delta)$.*

Proof. The theorem follows from the Implicit Function Theorem (Ortega and Rheinboldt, 2000, Theorem 5.2.4) applied to $\nabla\phi_\sigma(x;\delta) = 0$. ∎

An option to recover x^* using $\phi_\sigma(x;\delta)$ is to minimize a sequence of problems defined by $x_{k+1} = \arg\min_x \phi_\sigma(x;\delta_k)$ with $\delta_k \to 0$, using x_k to warm-start the next subproblem. However, we show that it is possible to decrease δ within a single problem, while retaining fast local convergence.

To keep results independent of the minimization algorithm being used, for a family of functions $\mathcal{F}$ we define $G : \mathcal{F} \times \mathbb{R}^n \to \mathbb{R}^n$ such that for $f \in \mathcal{F}$ and an iterate x, $G(f,x)$ computes an update direction. For example, if $\mathcal{F} = \mathcal{C}_2$, we can represent Newton's method with $G(f,x) = -H(x)^{-1} g(x)$, where $g(x) = \nabla f(x)$ and $H(x) = \nabla^2 f(x)$. Define $\nu(\delta)$ as a function such that for repeated applications, $\nu^k(\delta) \to 0$ as $k \to \infty$ at a chosen rate; for example, for a quadratic rate, we let $\nu(\delta) = \delta^2$.

Algorithm 3 describes how to adaptively update δ each iteration.

In order to analyze Algorithm 3, we formalize the notions of rates of convergence using definitions equivalent to those of Ortega and Rheinboldt (2000, §9).

Definition 9. *We say that $x_k \to x^*$ with order at least $\tau > 1$ if there exists $M > 0$ such that, for all sufficiently large k, $\|x_{k+1} - x^*\| \le M\|x_k - x^*\|^\tau$. We say that $x_k \to x^*$ with R-order at least $\tau > 1$ if there exists a sequence α_k such that, for all sufficiently large k,*

$$\|x_k - x^*\| \le \alpha_k, \qquad \alpha_k \to 0 \textit{ with order at least } \tau.$$

Algorithm 3 Minimization of the regularized penalty function $\phi_\sigma(x, \delta)$ with $\delta \to 0$

```
1: Choose $x_1$, $\delta_0$
2: for $k = 1, 2, \ldots$ do
3:     Set
       (31)    $\delta_k \leftarrow \max\{\min\{\|\nabla\phi_\sigma(x_k; \delta_{k-1})\|, \delta_{k-1}\}, \nu(\delta_{k-1})\}$
4:     $p_k \leftarrow G(\phi_\sigma(\cdot, \delta_k), x_k)$
5:     $x_{k+1} \leftarrow x_k + p_k$
6: end for
```

We first show that any minimization algorithm achieving a certain local rate of convergence can be regarded as *inexact Newton* (Dembo et al., 1982).

Lemma 10. *Let $f(x)$ be a $\mathcal{C}_2$ function with local minimum x^* and $H(x^*) \succ 0$. Suppose we minimize f according to*

$$x_{k+1} = x_k + p_k, \qquad p_k = G(f, x_k), \tag{32}$$

such that $x_k \to x^*$ with order at least $\tau \in (1, 2]$. Then in some neighborhood of x^*, the update procedure $G(f, x)$ is equivalent to the inexact-Newton iteration

$$x_{k+1} \leftarrow x_k + p_k, \qquad H(x_k)p_k = -g(x_k) + r_k, \qquad \|r_k\| = O(\|g(x_k)\|^\tau). \tag{33}$$

Proof. There exists a neighborhood $N_N(x^*)$ such that for any $x_0^N \in N_N(x^*)$, the Newton update $x_{k+1}^N = x_k^N + p_k^N$ with $H(x_k^N)p_k^N = -g(x_k^N)$ is quadratically convergent:

$$\|x^* - x_{k+1}^N\| \le M_1\|x^* - x_k^N\|^2, \qquad x_k \in N_N(x^*).$$

Similarly let $N_G(x^*)$ be the neighborhood where order τ convergence is obtained for (32) with constant M_2. Let $B_\epsilon(x^*) = \{x \mid \|x^* - x\| \le \epsilon\}$. Choose $\epsilon \le \min\{M_2^{-1}, 1\}$ such that $B_\epsilon(x^*) \subseteq N_N \cap N_G$, and observe that if $x_0 \in B_\epsilon(x^*)$, then $x_k \in B_\epsilon(x^*)$ for all k because $\|x^* - x_k\|$ is monotonically decreasing. By continuity of $H(x)$, there exists $M_3 > 0$ such that $\|H(x)\| \le M_3$ for all $B_\epsilon(x^*)$. Then for $x_k \in B_\epsilon(x^*)$,

$$\begin{aligned}
\|r_k\| = \|H(x_k)p_k + g(x_k)\| &= \|H(x_k)(x_{k+1} - x_k - p_k^N)\| \\
&\le \|H(x_k)\|\|x_{k+1} - x^* + x^* - x_{k+1}^N\| \\
&\le M_3(\|x_{k+1} - x^*\| + \|x_{k+1}^N - x^*\|) \\
&\le M_3(M_1 + M_2)\|x_k - x^*\|^\tau,
\end{aligned}$$

Because $f \in \mathcal{C}_2$, there exists a constant M_4 such that $\|x_k - x^*\| \le M_4\|g(x_k)\|$ for $x_k \in N_G(x^*) \cap N_N(x^*)$. Therefore $\|r_k\| \le M_4M_3(M_1 + M_2)\|g(x_k)\|^\tau$, which is the inexact-Newton method, convergent with order τ. ∎

Note that Lemma 10 can be modified to accommodate any form of superlinear convergence, as long as $\|r_k\|$ converges at the same rate as $x_k \to x^*$.

Theorem 11. *Suppose that (A1b) and (A2a) are satisfied, x^* is a second-order KKT point for (NP), $\nabla^2\phi_\sigma(x^*) \succ 0$, and there exists $\bar{\delta}$ and an open set $B(x^*)$ containing x^* such that for $\widetilde{x}_0 \in B(x^*)$ and $\delta \le \bar{\delta}$, the sequence defined by $\widetilde{x}_{k+1} = \widetilde{x}_k + G(\phi_\sigma(\cdot; \delta), \widetilde{x}_k)$ converges quadratically to $x(\delta)$:*

$$\|x(\delta) - \widetilde{x}_{k+1}\| \le M_\delta\|x(\delta) - \widetilde{x}_k\|^2.$$

Further suppose that for $\delta \le \bar{\delta}$, $M_\delta \le M$ is uniformly bounded. Then there exists an open set, $B'(x^*)$ that contains x^*, and $\delta' > 0$ such that if $x \in B'(x^*)$, $\delta \le \delta'$ and $x_k \to x^*$ for x_k defined by Algorithm 3 (with $\nu(\delta) = \delta^2$), then $x_k \to x^*$ R-quadratically.

The proof is in Appendix A.4. Although there are many technical assumptions, the takeaway message is that we need only minimize $\phi_\sigma(\cdot;\delta_k)$ until $\|\nabla\phi_\sigma\| = O(\delta_k)$, because under typical smoothness assumptions we have that $\|x(\delta) - x^*\| = O(\delta)$ for δ sufficiently small. Decreasing δ at the same rate as the local convergence rate of the method on a fixed problem should not perturb $\phi_\sigma(x;\delta)$ too much, therefore allowing for significant progress on the perturbed problem in few steps. Within the basin of convergence and for a fixed $\delta > 0$, an optimization method would achieve the same local convergence rate that it would have with $\delta = 0$ fixed.

Theorem 11 can be generalized to superlinear rates of convergence using a similar proof. As long as $\nu(\cdot)$ drives $\delta \to 0$ as fast as the underlying algorithm would locally converge for fixed δ, local convergence of the entire regularized algorithm is unchanged.

7. Inexact evaluation of the penalty function.

We discuss the effects of solving (24) approximately, and thus evaluating ϕ_σ and its derivatives inexactly. Various optimization solvers can utilize inexact function values and derivatives while ensuring global convergence and certain local convergence rates, provided the user can compute relevant quantities to a prescribed accuracy. For example, Conn et al. (2000, §8–9) describe conditions on the inexactness of model and gradient evaluations to ensure convergence, and Heinkenschloss and Ridzal (2014) describe an inexact trust-region SQP solver for PDE-constrained optimization using inexact function values and gradients. We focus on inexactness within trust-region methods for optimizing ϕ_σ.

The accuracy and computational cost in the evaluation of ϕ_σ and its derivatives depends on the accuracy of the solves of (24). If the cost to solve (24) depends on solution accuracy (e.g., with iterative linear solvers), it is advantageous to consider optimization solvers that use inexact computations, especially for large-scale problems.

Let $\mathcal{S} \subseteq \mathbb{R}^n$ be a compact set. In this section, we use $\widetilde{\phi}_\sigma(x)$, $\nabla\widetilde{\phi}_\sigma(x)$, etc. to distinguish the inexact quantities from their exact counterparts. We also drop the arguments from operators as in section 4. We consider three quantities that are computed inexactly: g_σ, ϕ_σ and $\nabla\phi_\sigma$. For given error tolerances η_i, we are interested in exploring termination criteria for solving (24) to ensure that the following conditions hold for all $x \in \mathcal{S}$:

$$\left|\phi_\sigma - \widetilde{\phi}_\sigma\right| \le M\eta_1, \tag{34a}$$

$$\left\|\nabla\phi_\sigma - \nabla\widetilde{\phi}_\sigma\right\| \le M\eta_2, \tag{34b}$$

$$\|g_\sigma - \widetilde{g}_\sigma\| \le M\eta_3, \tag{34c}$$

where $M > 0$ is some fixed constant (which may or may not be known). Kouri et al. (2014) give a trust-region method using inexact objective value and gradient information that guarantees global convergence provided (34a)–(34b) hold without requiring that M be known a priori. We may compare this to the conditions of Conn et al. (2000, §8.4, §10.6), which require more stringent conditions on (34a)–(34b). They require that $\eta_2 = \|\nabla\widetilde{\phi}_\sigma\|$ and that M be known and fixed according to parameters in the trust-region method.

This leads us to the following proposition, which allows us to bound the residuals of (18) and (20) to ensure (34).

PROPOSITION 12. *Let $\mathcal{S}$ be a compact set, and suppose that $\sigma_{\min}(A(x)) \geq \lambda > 0$ for all $x \in \mathcal{S}$. Then for $x \in \mathcal{S}$, if*

$$\|r_1\| = \left\| \mathcal{K} \begin{bmatrix} \widetilde{g}_\sigma \\ \widetilde{y}_\sigma \end{bmatrix} - \begin{bmatrix} g \\ \sigma c \end{bmatrix} \right\| \leq \min\{1, \|c\|^{-1}\} \cdot \min\{\eta_1, \eta_3\}, \tag{35}$$

then (34a) and (34c) hold for some constant M. Also, if

$$\|r_1\| \leq \eta_2 \quad \text{and} \quad \|r_2\| = \left\| \mathcal{K} \begin{bmatrix} \widetilde{v} \\ \widetilde{w} \end{bmatrix} - \begin{bmatrix} 0 \\ c \end{bmatrix} \right\| \leq \min\{1, \eta_2\}, \tag{36}$$

then (34b) holds for some (perhaps different) constant M.

Proof. Because $\mathcal{S}$ is compact and $\lambda > 0$, there exists $\bar{\lambda} > 0$ such that $\|\mathcal{K}\|$, $\|\mathcal{K}^{-1}\| \leq \bar{\lambda}$ for all $x \in \mathcal{S}$. Thus, (34c) follows directly from (20) and (35) with $M = \bar{\lambda}$. Similarly,

$$\left|\phi_\sigma - \widetilde{\phi}_\sigma\right| = \left|c^T \left(y_\sigma - \widetilde{y_\sigma}\right)\right| \leq \|c\| \, \|y_\sigma - \widetilde{y_\sigma}\| \leq \bar{\lambda}\eta_1,$$

and (34a) holds with $M = \bar{\lambda}$. We apply a similar analysis to ensure that (34b) holds. Define the vector $h \in \mathbb{R}^m$ such that $h_i = \|H_i\|$. Define v, w as the solutions to (36) for $r_2 = 0$, so that from (36) we have

$$\begin{aligned}
\|\nabla\phi_\sigma - \nabla\widetilde{\phi}_\sigma\| &\leq \|g_\sigma - \widetilde{g}_\sigma\| + \|Y_\sigma c - \widetilde{Y}_\sigma c\| \\
&\leq \bar{\lambda}\eta_2 + \|(H_\sigma - \sigma I)v - S_\sigma^T w - (\widetilde{H}_\sigma - \sigma I)\widetilde{v} + \widetilde{S}_\sigma^T \widetilde{w}\| \\
&\leq \bar{\lambda}\eta_2 + \sigma\|v - \widetilde{v}\| + \|H_\sigma v - \widetilde{H}_\sigma \widetilde{v}\| + \|S_\sigma^T w - \widetilde{S}_\sigma^T \widetilde{w}\| \\
&\leq \left(\bar{\lambda} + \sigma\bar{\lambda}\right)\eta_2 + \|H_\sigma(v - \widetilde{v}) + (H_\sigma - \widetilde{H}_\sigma)\widetilde{v}\| \\
&\quad + \|S_\sigma^T(w - \widetilde{w}) + (S_\sigma - \widetilde{S}_\sigma)^T \widetilde{w}\| \\
&\leq \left(\bar{\lambda} + \sigma\bar{\lambda}\right)\eta_2 + \|H_\sigma\|\|v - \widetilde{v}\| + \left\|\sum_{i=1}^m \left((y_\sigma)_i - (\widetilde{y}_\sigma)_i\right) H_i\right\| \|\widetilde{v}\| \\
&\quad + \|S_\sigma\|\|w - \widetilde{w}\| + \left\|\sum_{i=1}^m \widetilde{w}_i H_i\right\| \|g_\sigma - \widetilde{g}_\sigma\| \\
&\leq \left(\bar{\lambda} + \sigma\bar{\lambda} + \|H_\sigma\|\bar{\lambda} + \bar{\lambda}\|\widetilde{v}\|\|h\| + \|S_\sigma\|\bar{\lambda} + \|\widetilde{w}\|\|h\|\bar{\lambda}\right)\eta_2.
\end{aligned}$$

Note that $\|H_\sigma\|$, $\|h\|$, $\|S_\sigma\|$, $\|\widetilde{w}\|$ and $\|\widetilde{v}\|$ are bounded uniformly in $\mathcal{S}$. ∎

In the absence of additional information, using (34) with unknown M may be the only way to take advantage of inexact computations, because computing exact constants (such as the norms $\mathcal{K}$ or the various operators above) is not practical. In some cases the bounds (34) are relative, e.g., $\eta_2 = \min\{\|\nabla\widetilde{\phi}_\sigma\|, \Delta\}$ for a certain $\Delta > 0$. It may then be necessary to compute $\|\nabla\widetilde{\phi}_\sigma\|$ and refine the solutions of (20) and (18) until they satisfy (35)–(36). However, given the expense of applying these operators, it may be more practical to use a nominal relative tolerance, as in the numerical experiments of section 9.

We include a (trivial) improvement to Proposition 12 that satisfies (34a) and (34c) with $M = 1$, given additional spectral information about A. If we solve (20) by solving

$$\begin{bmatrix} I & A \\ A^T & 0 \end{bmatrix} \begin{bmatrix} \Delta g_\sigma \\ y_\sigma \end{bmatrix} = \begin{bmatrix} 0 \\ \sigma c - A^T g \end{bmatrix}, \qquad g_\sigma = g + \Delta g_\sigma,$$

we can use LNLQ (Estrin et al., 2018), a Krylov subspace method for such systems, which ensures that $\|\Delta g_\sigma - \Delta\widetilde{g}_\sigma^{(j)}\|$ and $\|y_\sigma - \widetilde{y}_\sigma^{(j)}\|$ are monotonic, where $\Delta\widetilde{g}_\sigma^{(j)}$, $\widetilde{y}_\sigma^{(j)}$ are the jth LNLQ iterates. Given $\lambda > 0$ such that $\sigma_{\min}(A) \ge \lambda$, LNLQ can compute cheap upper-bounds on $\|\Delta g_\sigma - \Delta\widetilde{g}_\sigma^{(j)}\|$ and $\|y_\sigma - \widetilde{y}_\sigma^{(j)}\|$, allowing us to terminate the solve when $\|\Delta g_\sigma - \widetilde{\Delta g_\sigma}\| \le \eta_2\|\widetilde{\Delta g_\sigma} + g\| = \eta_2\|\widetilde{g}_\sigma\|$ and $\|y_\sigma - \widetilde{y}_\sigma\| \le \min\{1, \|c\|^{-1}\}\eta_1$. Typically, termination criteria for the optimization solver will include a condition that $\|g_\sigma\| \le \epsilon_d$ to determine approximate local minimizers to (NP). For such cases, we can instead require that $\|\widetilde{g}_\sigma\| \le \frac{1}{1+\eta_2}\epsilon_d$, because then

$$\|g_\sigma\| \le \|g_\sigma - \widetilde{g}_\sigma\| + \|\widetilde{g}_\sigma\| \le (1+\eta_2)\|\widetilde{g}_\sigma\| \le \epsilon_d.$$

Similarly, we have

$$\left|\phi_\sigma - \widetilde{\phi}_\sigma\right| \le \|c\|\|y_\sigma - \widetilde{y}_\sigma\| \le \eta_1,$$

which now satisfies (34a) with $M = 1$.

Although finding suitable λ may be difficult in general, it is trivially available in some cases because of the way $\mathcal{K}$ is preconditioned (for an example, see section 9). However, a complication is that if LNLQ is used with a right-preconditioner $\mathcal{P} \approx A^TA$, then $\|y_\sigma - \widetilde{y}_\sigma\|_{\mathcal{P}}$ is monotonic and LNLQ provides bounds on the preconditioned norm instead of the Euclidean norm. If $\|\mathcal{P}^{-1}\|$ can be bounded, then the bound $\|y_\sigma - \widetilde{y}_\sigma\| \le \|y_\sigma - \widetilde{y}_\sigma\|_{\mathcal{P}}\|\mathcal{P}^{-1}\|$ can be used.

8. Practical considerations.

We discuss some matters related to the use of ϕ_σ in practice. In principle, nearly any smooth unconstrained solver can be used to find a local minimum of ϕ_σ because it has at least one continuous derivative, and a continuous Hessian approximation if (A1a) is satisfied. However, the structure of ϕ_σ lends itself more readily to certain optimization methods than to others, especially when the goal of creating a factorization-free solver is kept in mind.

Fletcher (1973) originally envisioned a Newton-type procedure

$$x_{k+1} \leftarrow x_k - \alpha_k B_i^{-1}(x_k)\nabla\phi_\sigma(x_k), \qquad i = 1 \quad \text{or} \quad 2,$$

where B_1, B_2 are the Hessian approximations from (22a)–(22b) and $\alpha_k > 0$ is a step size. Fletcher (1973, Theorem 2) further proved that superlinear convergence is achieved, or quadratic convergence if the second derivatives of f and c are Lipschitz continuous. However, for large problems it is expensive to compute B_i explicitly and solve the dense system $B_i s_k = -\nabla\phi_\sigma(x_k)$.

We instead propose using a Steihaug (1983) Newton-CG type trust-region solver to minimize ϕ_σ. First, trust-region methods are preferable to linesearch methods (Nocedal and Wright, 2006, §3–4) for objectives with expensive evaluations; it is costly to evaluate ϕ_σ repeatedly to determine a step-size every iteration as this requires solving a linear system. Further, $\nabla^2\phi_\sigma$ is often indefinite and trust-region methods can take advantage of directions of negative curvature. Computing B_i explicitly is not practical, but products are reasonable as they only require solving two additional linear systems with the same matrix, thus motivating the use of a Newton-CG type trust-region solver. In particular, solvers such as TRON (Lin and Moré, 1999a) and KNITRO (Byrd et al., 2006) are suitable for minimizing ϕ_σ. KNITRO has the additional advantage of handling explicit linear constraints.

We have not yet addressed choosing σ. Although we can provide an a posteriori threshold value for σ^*, it is difficult to know this threshold ahead of time. Mukai and

Polak (1975) give a scheme for updating ρ with $\phi_{\sigma,\rho}$ and $\sigma = 0$; however, they were using a Newton-like scheme that required a solve with $B_1(x)$. Further, σ^* ensures only *local* convexity, and that a local minimizer of (NP) is a local minimizer of ϕ_σ—but as with other penalty functions, ϕ_σ may be unbounded below in general for any σ. A heuristic that we employ is to ensure that the primal and dual feasibility, $\|c(x)\|$ and $\|g_L(x, y_\sigma(x))\|$, are within some factor of each other (e.g., 100) to encourage them to decrease at approximately the same rate. If primal feasibility decreases too quickly and small steps are taken, it is indicative of σ being too large, and similarly if primal feasibility is too large then σ should be increased; this can be done with a multiplicative factor or by estimating $\|P_A(x)H_\sigma(x)P_A(x)\|$ via the power method. Although this heuristic is often effective in our experience, in situations where the penalty function begins moving toward negative infinity, we require a different recovery strategy, which is the subject of future work.

In practice, regularization (section 6) is used only if necessary. For well-behaved problems, using $\delta = 0$ typically requires fewer outer iterations than using $\delta > 0$. However, when convergence is slow and/or the Jacobians are ill-conditioned, initializing with $\delta > 0$ is often vital and can improve performance significantly.

9. Numerical experiments.

We investigate the performance of Fletcher's penalty function on several PDE-constrained optimization problems and some standard test problems. For each test we use the stopping criterion

$$\begin{array}{c} \|c(x_k)\| \le \epsilon_p \\ \|g_\sigma(x_k)\| \le \epsilon_d \end{array} \qquad \text{or} \qquad \|\nabla\phi_\sigma(x_k)\| \le \epsilon_d, \tag{37}$$

with $\epsilon_p := \epsilon\,(1 + \|x_k\|_\infty + \|c(x_0)\|_\infty)$ and $\epsilon_d := \epsilon\,(1 + \|y_k\|_\infty + \|g_\sigma(x_0)\|_\infty)$, where $\epsilon > 0$ is a tolerance, e.g., $\epsilon = 10^{-8}$. We also keep σ fixed for each experiment.

Depending on the problem, the augmented systems (24) are solved by either direct or iterative methods. For direct methods, we use the corrected semi-normal equations (Appendix A.2.2). For iterative solves, we use CRAIG (Arioli, 2013) with preconditioner $\mathcal{P}$ and two possible termination criteria:

$$\left\| \mathcal{K}\begin{bmatrix} p^{(k)} \\ q^{(k)} \end{bmatrix} - \begin{bmatrix} u \\ v \end{bmatrix} \right\|_{\bar{\mathcal{P}}^{-1}} \le \eta \left\| \begin{bmatrix} u \\ v \end{bmatrix} \right\|_{\bar{\mathcal{P}}^{-1}}, \qquad \bar{\mathcal{P}} := \begin{bmatrix} I & \\ & \mathcal{P} \end{bmatrix} \tag{38a}$$

$$\left\| \begin{bmatrix} p^* \\ q^* \end{bmatrix} - \begin{bmatrix} p^{(k)} \\ q^{(k)} \end{bmatrix} \right\|_{\bar{\mathcal{P}}} \le \eta \left\| \begin{bmatrix} p^{(k)} \\ q^{(k)} \end{bmatrix} \right\|_{\bar{\mathcal{P}}}, \tag{38b}$$

which are respectively based on the relative residual and the relative error (obtained via LNLQ). We can use (38b) when a lower bound on $\sigma_{\min}(A\mathcal{P}^{-1/2})$ is available (e.g., for the PDE-constrained optimization problems).

Because we are using trust-region methods to minimize ϕ_σ, the objective and gradient of ϕ_σ (and therefore of f) are evaluated once per iteration. We use KNITRO (Byrd et al., 2006) and a Matlab implementation of TRON (Lin and Moré, 1999b). Our implementation of TRON does not require explicit Hessians (only Hessian-vector products) and is unpreconditioned. We use $B_1(x)$ (22a) when efficient products with $S(u, x)$ are available, otherwise we use $B_2(x)$ (22b). When ϕ_σ is evaluated approximately (for coarse η), we use the solvers without modification, thus pretending that the function and gradient are evaluated exactly.

Table 1: Results of solving (39) using TRON to minimize ϕ_σ, with (38b) (left) and (38a) (right) to terminate the augmented linear system solves for various η. We record the number of Lagrangian Hessian (#Hv), Jacobian (#Av), and transposed Jacobian (#A^Tv) products.

η	itns	#Hv	#Av	#A^Tv	itns	#Hv	#Av	#A^Tv
10^{-2}	37	19112	56797	50553	35	7275	29453	27148
10^{-4}	34	6758	35559	33423	35	7185	36757	34482
10^{-6}	35	7182	45893	43619	35	7194	47999	45721
10^{-8}	35	7176	53296	51204	35	7176	54025	51753
10^{-10}	35	7176	59802	57530	35	7176	59310	57038

error-based termination residual-based termination

9.1. 1D Burger's problem. We solve the following one-dimensional ODE-constrained control problem:

$$\begin{aligned} &\underset{u,z}{\text{minimize}} \;\; \tfrac{1}{2}\int_\Omega (u(x)-u_d(x))^2\,dx + \tfrac{1}{2}\alpha\int_\Omega z(x)^2 dx \\ &\text{subject to} \quad -\nu u_{xx} + u u_x = z + h \quad \text{in } \Omega, \\ &\qquad\qquad u(0)=0,\;\; u(1)=-1, \end{aligned} \tag{39}$$

where the constraint is a 1D Burger's equation over $\Omega = (0,1)$, with $h(x) = 2\left(\nu + x^3\right)$ and $\nu = 0.08$. The first objective term measures deviation from the data $u_d(x)$, while the second term regularizes the control with $\alpha = 10^{-2}$. We discretize (39) by segmenting Ω into $n_c = 512$ equal-sized cells, and approximate u and z with piecewise linear elements. This results in a nonlinearly constrained optimization problem with $n = 2n_c = 1024$ variables and $m = n_c - 1$ constraints.

We optimize u, z by minimizing ϕ_σ with $\sigma = 10^3$, using $B_1(x)$ (22a) as Hessian approximation and $u_0 = \mathbb{1}$, $z_0 = \mathbb{1}$ as the initial point. We use TRON to optimize ϕ_σ and LNLQ to (approximately) solve (24). We partition the Jacobian of the discretized constraints into $A(x)^T = \begin{bmatrix} A_u(x)^T & A_z(x)^T \end{bmatrix}$, where $A_u(x) \in \mathbb{R}^{n\times n}$ and $A_z(x) \in \mathbb{R}^{m\times m}$ are the Jacobians for u and z. We use the preconditioner $\mathcal{P}(x) = A_u(x)^T A_u(x)$, which amounts to performing two solves of Burger's equation with a given source. For this preconditioner, $\sigma_{\min}(A\mathcal{P}^{-1/2}) \geq 1$, allowing us to bound the error via LNLQ and to use both (38b) and (38a) to terminate LNLQ. The maximum number of inner-CG iterations (for solving the trust-region subproblem) is n.

We choose $\epsilon = 10^{-8}$ in the stopping conditions (37). Table 1 records the number of Hessian- and Jacobian-vector products as we vary the accuracy of the linear system solves via η in (38).

TRON required a moderate number of trust-region iterations. However, evaluating ϕ_σ and its derivatives can require many Jacobian and Hessian products, because for every product with the approximate Hessian we need to solve an augmented linear system. On the other hand, the linear systems did not have to be solved to full precision. As η increased from 10^{-10} to 10^{-2}, the number of Hessian-vector products stayed relatively constant, but the number of Jacobian-vector products dropped substantially, and the average number of LNLQ iterations required per solve dropped from about 9 to 5, except when $\eta = 10^{-2}$ in (38b), the linear solves were too inaccurate and the

Table 2: Results of solving (40) using TRON to minimize ϕ_σ with various η in (38b) (left) and (38a) (right) to terminate the linear system solves. We record the number of Lagrangian Hessian (#Hv), Jacobian (#(Av) and adjoint Jacobian (#A^Tv) products.

η	Iter.	$\#Hv$	$\#Av$	$\#A^Tv$	Iter.	$\#Hv$	$\#Av$	$\#A^Tv$
10^{-2}	29	874	1794	2608	27	850	1772	2562
10^{-4}	27	830	1950	2728	25	668	1649	2265
10^{-6}	27	866	2317	3129	27	868	2356	3168
10^{-8}	27	866	2673	3485	27	866	2784	3596
10^{-10}	27	866	3145	3957	27	866	3251	4063
	error-based termination				residual-based termination			

number of CG iterations per trust-region subproblem increased dramatically near the solution (requiring more linear solves). Using (38a) tended to perform more products with the Lagrangian Hessian and Jacobian, except when the linear solves were nearly exact, or extremely inexact.

9.2. 2D Inverse Poisson problem. Let $\Omega = (0,1)^2$ denote the physical domain and $H^1(\Omega)$ denote the Sobolev space of functions in $L^2(\Omega)$, whose weak derivatives are also in $L^2(\Omega)$. Let $H_0^1(\Omega) \subset H^1(\Omega)$ be the Hilbert space of functions whose value on the boundary $\partial\Omega$ is zero. We solve the following 2D PDE-constrained control problem:

$$\begin{aligned} &\underset{u\in H_0^1(\Omega),\, z\in L^2(\Omega)}{\text{minimize}} && \tfrac{1}{2}\int_\Omega (u-u_d)^2\,dx + \tfrac{1}{2}\alpha\int_\Omega z^2 dx \\ &\text{subject to} && -\nabla\cdot(z\nabla u) = h \quad \text{in } \Omega, \\ & && u = 0 \quad \text{in } \partial\Omega. \end{aligned} \tag{40}$$

Let $c = (0.2, 0.2)$ and define $S_1 = \{x \mid \|x-c\|_2 \le 0.3\}$ and $S_2 = \{x \mid \|x-c\|_1 \le 0.6\}$. The target state u_d is generated as the solution of the PDE with $z_*(x) = 1 + 0.5 \cdot I_{S_1}(x) + 0.5 \cdot I_{S_2}(x)$, where for any set C, $I_C(x) = 1$ if $x \in C$ and 0 otherwise.

The force term here is $h(x_1,x_2) = -\sin(\omega x_1)\sin(\omega x_2)$, with $\omega = \pi - \frac{1}{8}$. The control variable z represents the diffusion coefficients for the Poisson problem that we are trying to recover based on the observed state u_d. We set $\alpha = 10^{-4}$ as regularization parameter. We discretize (40) using P_1 finite elements on a uniform mesh of 1089 triangular elements and employ an identical discretization for the optimization variables $z \in L^2(D)$, obtaining a problem with $n_z = 1089$ controls and $n_u = 961$ states, so that $n = n_u + n_z$. There are $m = n_u$ constraints, as we must solve the PDE on every interior grid point. The initial point is $u_0 = \mathbb{1}$, $z_0 = \mathbb{1}$.

We use $\sigma = 10^{-2}$ as penalty parameter, and $B_2(x)$ as Hessian approximation. We again use LNLQ for the linear solves, with the same preconditioning strategy as in section 9.1. The results are given in Table 2. We see a similar trend to that of Table 1, as larger η allows TRON to converge within nearly the same number of outer iterations and Lagrangian Hessian-vector products (even when $\eta = 10^{-2}$), while significantly decreasing the number of Jacobian-vector products. We see again that using (38b) to terminate LNLQ tends to need less work than with (38a). The exception is using (38a) with $\eta = 10^{-4}$. The solver terminates two iterations sooner, resulting in a sharp drop in Jacobian-vector products but little change in solution quality. Note that if $\epsilon = 10^{-9}$

Table 3: Results of solving (41) using TRON to minimize ϕ_σ with various η in (38b) (left) and (38a) (right) to terminate the linear system solves. We record the number of Lagrangian Hessian ($\#Hv$), Jacobian ($\#(Av)$ and adjoint Jacobian ($\#A^Tv$) products.

η	Iter.	$\#Hv$	$\#Av$	$\#A^Tv$	Iter.	$\#Hv$	$\#Av$	$\#A^Tv$
10^{-2}	29	822	1582	2342	29	822	1636	2396
10^{-4}	29	816	1635	2389	29	816	1801	2555
10^{-6}	29	816	1800	2554	29	816	2029	2783
10^{-8}	29	816	2077	2831	29	816	2301	3055
10^{-10}	29	816	2351	3105	29	816	2637	3391
	error-based termination				residual-based termination			

were used for the experiment, the runs would appear more similar to one another.

9.3. 2D Poisson-Boltzmann problem. We now solve a control problem where the constraint is a 2D Poisson-Boltzmann equation:

$$\begin{aligned} &\underset{u\in H_0^1(\Omega), z\in L^2(\Omega)}{\text{minimize}} && \tfrac{1}{2}\int_\Omega (u-u_d)^2\,dx + \tfrac{1}{2}\alpha\int_\Omega z^2 dx \\ &\text{subject to} && -\Delta u + \sinh(u) = h + z \quad \text{in } \Omega, \\ & && u = 0 \quad \text{in } \partial\Omega. \end{aligned} \tag{41}$$

We use the same notation and Ω as in section 9.2, with forcing term $h(x_1,x_2) = -\sin(\omega x_1)\sin(\omega x_2)$, $\omega = \pi - \frac{1}{8}$, and target state

$$u_d(x) = \begin{cases} 10 & \text{if } x \in [0.25, 0.75]^2 \\ 5 & \text{otherwise.} \end{cases}$$

We discretize (41) using P_1 finite elements on a uniform mesh with 1089 triangular elements, resulting in a problem with $n = 2050$ variables and $m = 961$ constraints. We use $u_0 = \mathbb{1}$, $z_0 = \mathbb{1}$ as the initial point.

We perform the experiment described in section 9.2 using $\sigma = 10^{-1}$, and record the results in Table 3. The results are similar to Table 2, where the number of Jacobian products decrease with η, while the number of outer iterations and Lagrangian-Hessian products stay fairly constant. We see that with stopping criterion (38a), the LNLQ iterations increase somewhat compared to (38b), as it is a tighter criterion.

9.4. Explicit linear constraints. We investigate the effect of maintaining the linear constraints explicitly (section 5), using problems from the CUTEst test set (Gould et al., 2003) that have linear constraints. We use KNITRO to minimize ϕ_σ with and without linear constraints, because it can handle them explicitly. We use the corrected semi-normal equations to perform linear solves, and Hessian approximation $B_1(x)$ (22a). The threshold penalty parameters (8) and (26) are computed from earlier optimal solutions when the linear constraints were kept implicit (σ^*_{impl}) and explicit (σ^*_{expl}) respectively. The results are recorded in Table 4.

We observe that maintaining the linear constraints explicitly decreases the penalty parameter for the `Chain` problems, and that KNITRO finds ϕ_σ to be unbounded when

Table 4: Results for problems with linear constraints. m_{lin} and m_{nln} are the number of linear and nonlinear constraints; σ^*_{impl} and σ^*_{expl} are the threshold penalty parameters when the linear constraints are handled implicitly and explicitly; σ is the penalty parameter. The last two columns give the number of iterations before convergence; the symbol "$*$" indicates that unboundedness was detected, and "-" that 100 iterations were performed without converging. The solver exits when unboundedness is detected or an iterate satisfies (37) with $\epsilon = 10^{-8}$.

Problem	n	m_{lin}	m_{nln}	σ^*_{impl}	σ^*_{expl}	σ	Impl.	Expl.
Chain100	202	102	1	0.0047	0	10^{-3}	$*$	13
						0.005	8	10
Chain200	402	202	1	0.0024	0	10^{-3}	$*$	11
						0.003	7	10
Chain400	802	402	1	0.0012	0	10^{-3}	$*$	10
						0.002	7	10
Channel100	800	400	400	0	0	10^{-3}	–	5
						1	–	5
Channel200	1600	800	800	0	0	10^{-3}	–	5
						1	–	5
Channel400	1600	800	800	0	0	10^{-3}	–	5
						1	–	5

$\sigma < \sigma^*_{\text{impl}}$ and the linear constraints are implicit. If σ is large enough, both versions converge (with and without explicit linear constraints), and keeping the constraints implicit saves a few iterations.

For the `Channel` problems, the threshold parameter is zero in both cases. However, KNITRO converges quickly when the linear constraints are kept explicit, but otherwise fails to converge in a reasonable number of iterations. This phenomenon for the `Channel` problems appears to be independent of σ (more values were investigated than are reported here). Again it appears beneficial to maintain some of the constraints explicitly.

9.5. Regularization. We next solve problems where $A(x)$ is rank-deficient for some iterates. Such problems require that ϕ_σ be regularized (section 6). We again use the corrected semi-normal equations to solve the linear systems, with $B_2(x)$ as the Hessian approximation.

For problem `hs061` ($n = 3$ variables, $m = 3$ constraints) from the CUTEst test set (Gould et al., 2003) we use $x_0 = 0$, $\sigma = 10^2$, $\delta_0 = 10^{-1}$. For problem `mss1` ($n = 90$, $m = 73$) we use $x_0 = 0$, $\sigma = 10^3$, $\delta_0 = 10^{-2}$. In both cases we decrease δ according to $\nu(\delta) = \delta^2$ to retain local quadratic convergence. For both problems, $A(x_0)$ is rank-deficient and ϕ_σ is undefined, so the trust-region solvers terminate immediately. We therefore regularize ϕ_σ and record the iteration at which δ changed. For `mss1`, we set $\delta_{\min} = 10^{-7}$ to avoid ill-conditioning. The results are in Table 5.

The regularized problems converge with few iterations between δ updates, showing evidence of quadratic convergence. Note that a large δ can perturb $\phi_\sigma(\cdot;\delta)$ substantially, so that δ_0 may need to be chosen judiciously. We use $\delta_0 = 10^{-2}$ because neither TRON nor KNITRO would converge for `mss1` when $\delta_0 = 10^{-1}$.

Table 5: Convergence results for `hs061` (left) and `mss1` (right) when TRON and KNITRO minimize $\phi_\sigma(\cdot;\delta)$. The first rows show the iteration at which δ is updated, and the last two rows record the final primal and dual infeasibilities.

δ	TRON	KNITRO
10^{-1}	22	12
10^{-2}	23	13
10^{-4}	24	14
$\|c(\bar{x})\|$	10^{-10}	10^{-9}
$\|g_\sigma(\bar{x})\|$	10^{-7}	10^{-8}

δ	TRON	KNITRO
10^{-2}	46	33
10^{-4}	52	36
10^{-7}	53	37
$\|c(\bar{x})\|$	10^{-12}	10^{-14}
$\|g_\sigma(\bar{x})\|$	10^{-8}	10^{-9}

10. Discussion and concluding remarks.

The smooth exact penalty approach is promising for nonlinearly constrained optimization particularly when the augmented linear systems (24) can be solved efficiently. However, several potential drawbacks remain as avenues for future work.

One property of ϕ_σ observed from our experiments is that it is highly nonlinear and nonconvex. Even though we can show superlinear or quadratic local convergence, the high nonlinearity potentially results in more globalization iterations and smaller step-sizes than for other constrained methods. Further, ϕ_σ is usually nonconvex, even if (NP) is convex.

An aspect not yet discussed is perhaps more problem-dependent. Preconditioning the trust-region subproblems for this penalty function is particularly non-trivial because the (approximate) Hessian is available only as an operator. Traditional approaches based on incomplete factorizations (Lin and Moré, 1999a) are not applicable. One possibility is to use a preconditioner for the Lagrangian Hessian H_σ as a preconditioner for the approximate penalty Hessian B_i (22a)–(22b). This may be effective when $m \ll n$ because H_σ and B_i would differ only by a low-rank update; otherwise H_σ can be a poor approximation to B_i. Preconditioning is vital for trust-region solvers, and is a direction for future work.

Products with B_i (22a)–(22b) are generally more expensive than typical Hessian-vector products, as they require solving a linear system. Products with a quasi-Newton approximation would be significantly faster. Also, exact curvature away from the solution may be less important than near the solution for choosing good directions; therefore a hybrid scheme that begins with quasi-Newton and switches to B_i may be effective. Another strategy, similar to Morales and Nocedal (2000), is to use quasi-Newton approximations to precondition the trust-region subproblems involving B_i: the approximation for iteration k can be updated with every $B_i(x_{k-1})$ product, or with every update step $x_k - x_{k-1}$.

Further improvements that would make our approach applicable to a wider range of problems include: developing a robust update for the penalty parameter; termination criteria on the augmented systems in order to integrate ϕ_σ fully into a Heinkenschloss and Ridzal (2014)-style solver; and relaxing (A2a) to weaker constraint qualifications. Even if $\sigma \geq \sigma^*$ it is possible for ϕ_σ to be unbounded, because σ only guarantees local convexity. Diverging iterates must be detected sufficiently early because by the time unboundedness is detected, it may be too late to update σ and we would need to backtrack several iterates. To relax (A2a), it may be possible to combine our regularization (28) with a dual proximal-point approach.

The next obvious extension is to handle inequality constraints—the subject of a future paper. Fletcher (1973) proposed a non-smooth extension to ϕ_σ for this purpose, but it is less readily applicable to most solvers.

Our Matlab implementation can be found at https://github.com/optimizers/FletcherPenalty. To highlight the flexibility of Fletcher's approach, we implemented several options for applying various solvers to the penalty function and for solving the augmented systems, and other options discussed along the way.

Appendix A. Technical details.

We provide proofs and technical details that were omitted earlier.

A.1. Example problem with a spurious local minimum.

Consider the feasibility problem (NP) with $f(x) = 0$ and $c(x) = x^3 + x - 2$. The only minimizer is $x^* = 1$. The penalty function

$$\phi_\sigma(x) = \sigma \frac{(x^3 + x - 2)^2}{(9x^2 + 1)^2}$$

is defined everywhere and has local minimizers at $x_1 = 1$ (the solution) and $x_2 \approx -1.56$. Because the stationary points are independent of σ in this case, ϕ_σ always has the spurious local minimizer x_2.

A.2. Direct methods for solving the augmented system.

We describe various direct methods for solving (24) required to evaluate ϕ_σ and its derivatives. Recall that $A_\delta = \begin{bmatrix} A^T & \delta I \end{bmatrix}^T$ when $\delta > 0$; otherwise $A_\delta = A$. For this section, given a matrix R and vector b, the shorthand notation $x \leftarrow R \setminus b$ means that x solves the system $Rx = b$ (typically via forward or backward substitution).

A.2.1. QR factorization.

Algorithm 4 computes (p, q) using the thin QR factorization of $A_\delta = QR$, with Q orthogonal and $R \in \mathbb{R}^{m \times m}$.

Algorithm 4 Solving (24) using the QR factorization.

```
1: $Q,\ R \leftarrow \mathrm{qr}(A_\delta)$
2: $\bar{w} \leftarrow Q_{1:n,:}^T w$
3: $\bar{z} \leftarrow R^T \setminus z$
4: $p \leftarrow w - Q_{1:n,:}(\bar{w} - \bar{z})$
5: $q \leftarrow R \setminus (\bar{w} - \bar{z})$
6: return $(p, q)$
```

The advantage is that A_δ is factorized instead of $\mathcal{K}$, and this method is backward stable for both p and q (Golub and Van Loan, 2013, §5.3.6). If A_δ is sparse, R is likely to be sparse (for some column permutation of A_δ) but unfortunately Q is not. For large problems, it may not be practical to store Q in order to solve (24).

A.2.2. Corrected semi-normal equations.

The R factor from $A_\delta = QR$ can be computed without storing Q. We can then solve the semi-normal equations $R^T Rq = A^T w - z$ and set $p = w - Aq$. Björck and Paige (1994) show that this is not acceptable-error stable for p, possibly giving large error in p, particularly when $\|p\| \ll \|w\|$. Note that $p = g_\sigma$ in (20) means we may obtain large errors in the gradient near the solution if care is not taken. Fortunately, Björck and Paige (1994) show that one step of iterative refinement ensures p is acceptable-error stable; see Algorithm 5.

Algorithm 5 Solving (24) using the semi-normal equations.

```
1: $R \leftarrow \mathrm{qr}(A_\delta)$
2: $q \leftarrow (R^TR) \setminus (A^Tw - z)$      $p \leftarrow w - Aq$
3: $\Delta q \leftarrow (R^TR) \setminus (A^Tp - \delta^2 q - z)$      ▷ Iterative refinement
4: $q \leftarrow q + \Delta q$      $p \leftarrow p - A\Delta q$
5: **return** $(p, q)$
```

A.2.3. LDL and Bunch-Kaufman factorization. When it is not practical to store Q from the QR factors of A, or the semi-normal equations do not provide sufficient accuracy, it may be possible to compute the LDL or Bunch-Kaufman factorization of $\mathcal{K}$ directly. Although an $(n+m) \times (n+m)$ matrix is factorized (rather than an $n \times m$ matrix), the entire factorization is likely to be sparse, and the solution is typically more accurate than with the semi-normal equations.

Björck (1967) and Saunders (1995) discuss scaling of the $(1,1)$ identity block to improve the condition number of $\mathcal{K}$. Saunders (1995) also considers the case where $\mathcal{K}$ is regularized with $-\delta^2 I$ in the $(2,2)$ block.

A.3. Maintaining explicit constraints. We discuss the technical details about the penalty function when a subset of the constraints are linear and maintained explicitly, as defined in (25). We define $W_\sigma(x) = \nabla w_\sigma(x) \in \mathbb{R}^{n \times m_2}$, and $C(x) = \begin{bmatrix} A(x) & B \end{bmatrix}$ as the Jacobian of all constraints. The operators $g_\sigma(x)$, $H_\sigma(x)$, $S(x,v)$ and $T(x,w)$ are still defined over all constraints, not just the nonlinear ones, and so they act on $C(x)$ instead of $A(x)$. Define

$$g^y_\sigma(x) = g(x) - A(x) y_\sigma(x) \tag{42}$$

as the gradient of the partial Lagrangian with respect to the nonlinear constraints $c(x)$ only (note that the linear constraints do not affect H_σ). The gradient and Hessian of the penalty function become

$$\nabla \phi_\sigma(x) = g^y_\sigma(x) - Y_\sigma(x) c(x), \tag{43a}$$

$$\nabla^2 \phi_\sigma(x) = H_\sigma(x) - A(x) Y_\sigma(x)^T - Y_\sigma(x) A(x)^T - \nabla_x \left[Y_\sigma(x) c \right]. \tag{43b}$$

We restate the optimality conditions for (NP-EXP) in terms of the penalty function.

Definition 13 (First-order KKT point). *A point x^* is a first-order KKT point of (NP-EXP) if there exists a Lagrange multiplier $w^* \in \mathbb{R}^{m_2}$ for the linear constraints such that for any $\sigma \ge 0$ the following hold:*

$$c(x^*) = 0, \tag{44a}$$

$$B^T x^* = d, \tag{44b}$$

$$\nabla \phi_\sigma(x^*) = B w^*. \tag{44c}$$

The elements of $y^* := y_\sigma(x^*)$ and $w^* := w_\sigma(x^*)$ comprise the Lagrange multipliers of (NP-EXP) associated with x^*.

Definition 14 (Second-order KKT point). *The first-order KKT point x^* satisfies the second-order necessary KKT condition for (NP-EXP) if for any $\sigma \ge 0$,*

$$p^T \nabla^2 \phi_\sigma(x^*) p \ge 0 \text{ for all } p \text{ such that } C(x^*)^T p = 0. \tag{45}$$

The condition is sufficient if the above inequality is strict.

A.3.1. Proof of Theorem 7. We prove (27a), (27c) and (27b) in order. Observe that the multiplier estimates $y_\sigma(x)$ and $w_\sigma(x)$ satisfy

$$C(x)^T C(x) \begin{bmatrix} y_\sigma(x) \\ w_\sigma(x) \end{bmatrix} = C(x)^T g(x) - \sigma \begin{bmatrix} c(x) \\ B^T x - d \end{bmatrix}. \tag{46}$$

Proof of (27a): If $\bar{x}$ is a first-order KKT point for (25), then

$$Bw^* = g(\bar{x}) - A(\bar{x}) y_\sigma(\bar{x}) - Y_\sigma(\bar{x}) c(\bar{x}),$$

and by multiplying both sides by $C(\bar{x})^T$ and using (46) we have

$$\begin{bmatrix} A(\bar{x})^T B w^* \\ B^T B w^* \end{bmatrix} = \sigma \begin{bmatrix} c(\bar{x}) \\ 0 \end{bmatrix} + \begin{bmatrix} A(\bar{x})^T B w_\sigma(\bar{x}) \\ B^T B w_\sigma(\bar{x}) \end{bmatrix} - \begin{bmatrix} A(\bar{x})^T Y_\sigma(\bar{x}) c(\bar{x}) \\ B^T Y_\sigma(\bar{x}) c(\bar{x}) \end{bmatrix},$$

so that $w_\sigma(\bar{x}) = w^* + (B^T B)^{-1} B^T Y_\sigma(\bar{x}) c(\bar{x})$. Substituting $w_\sigma(\bar{x})$ into the first block of equations and rearranging gives

$$A(\bar{x}) \bar{P}_B Y_\sigma(\bar{x}) c(\bar{x}) = \sigma c(\bar{x}).$$

The triangle inequality gives $\sigma\|c(\bar{x})\| \le \|A(\bar{x})^T \bar{P}_B Y_\sigma(\bar{x})\| \|c(\bar{x})\|$, implying $c(\bar{x}) = 0$. Then $w_\sigma(\bar{x}) = w^*$ and $g_\sigma(\bar{x}) = 0$, so $\bar{x}$ is a first-order KKT point for (NP-EXP).

Proof of (27c): As in the proof of (11c), we differentiate (46) to obtain

$$C(x)^T C(x) \begin{bmatrix} Y_\sigma(x)^T \\ W_\sigma(x)^T \end{bmatrix} = C(x)^T [H_\sigma(x) - \sigma I] + S(x, g_\sigma(x)). \tag{47}$$

Because x_2^* satisfies first-order conditions (44), $g_\sigma(x) = 0$ and so for $P_C = P_C(x)$,

$$A(x_2^*) Y_\sigma(x_2^*)^T + B W_\sigma(x_2^*)^T = P_C [H_\sigma(x_2^*) - \sigma I]. \tag{48}$$

Substituting into (43b) and using $H_\sigma(x_2^*) = H_L(x_2^*, y^*)$ and $P_C + \bar{P}_C = I$ gives

$$\nabla^2 \phi_\sigma(x_2^*) = \bar{P}_C H_L(x_2^*, y^*) \bar{P}_C - P_C H_L(x_2^*, y^*) P_C + 2\sigma P_C - B W_\sigma(x)^T - W_\sigma(x) B^T.$$

Because $B^T p = 0$, we can write $p = \bar{P}_B \bar{p}$ and hence

$$\begin{aligned} & 0 \le p^T \nabla^2 \phi_\sigma(x_2^*) p \\ \iff & 0 \preceq \bar{P}_B \bar{P}_C H_L(x_2^*, y^*) \bar{P}_C \bar{P}_B - \bar{P}_B P_C H_L(x_2^*, y^*) P_C \bar{P}_B + 2\sigma \bar{P}_B P_C \bar{P}_B, \end{aligned}$$

which is equivalent to $\sigma \ge \sigma^*$.

Proof of (27b). With x_1^* in (48) and again using properties of P_C and $\bar{P}_B$,

$$\begin{aligned} \sigma \ge \|\bar{P}_B A(x_1^*) Y_\sigma(\bar{x}_1)\| &= \|\bar{P}_B P_C (H_L(x_1^*, y^*) - \sigma I)\| \\ &\ge \|\bar{P}_B P_C (H_L(x_1^*, y^*) - \sigma I) P_C \bar{P}_B\| \\ &\ge \|\bar{P}_B P_C H_L(x_1^*, y^*) P_C \bar{P}_B\| - \|\sigma P_C \bar{P}_B\| \\ &\ge 2\sigma^* - \sigma. \end{aligned}$$

Thus $\sigma \ge \sigma^*$ as required.

A.3.2. Evaluating the penalty function and derivatives. We again drop the arguments on functions and assume they are evaluated at a point x for some σ:

$$y = y_\sigma(x), \quad A = A(x), \quad Y_\sigma = Y_\sigma(x), \quad H_\sigma = H_\sigma(x), \quad S_\sigma = S_\sigma(x, g_\sigma(x)), \quad \text{etc.}$$

The multipliers for evaluating the penalty function are obtained by solving

$$\begin{bmatrix} I & A & B \\ A^T & & \\ B^T & & \end{bmatrix} \begin{bmatrix} g_\sigma \\ y_\sigma \\ w_\sigma \end{bmatrix} = \begin{bmatrix} g \\ \sigma c \\ \sigma(Bx-d) \end{bmatrix}. \tag{49}$$

To compute the gradient and Hessian products, we use the identity

$$C^T C \begin{bmatrix} Y_\sigma^T \\ W_\sigma^T \end{bmatrix} = C^T[H_\sigma - \sigma I] + S_\sigma \tag{50}$$

to obtain the necessary products with Y_σ and Y_σ^T. Observe that

$$Y_\sigma u = \begin{bmatrix} Y_\sigma & W_\sigma \end{bmatrix} \begin{bmatrix} u \\ 0 \end{bmatrix}, \qquad Y_\sigma^T v = \begin{bmatrix} I & 0 \end{bmatrix} \begin{bmatrix} Y_\sigma^T \\ W_\sigma^T \end{bmatrix} v,$$

so that Algorithm 1 and Algorithm 2 can be applied.

Note that to compute the gradient in (43a), g_σ^y is not available directly from the solution to (49) and must be computed explicitly using (42).

Approximate products with $\nabla^2\phi_\sigma$ can be computed via

$$\begin{aligned} \nabla^2\phi_\sigma \approx B_1 &:= H_\sigma - AY^T - YA^T \qquad (51)\\ &= H_\sigma - \begin{bmatrix} A & 0 \end{bmatrix} (C^TC)^{-1}C^T(H_\sigma - \sigma I) - \begin{bmatrix} A & 0 \end{bmatrix} (C^TC)^{-1}S_\sigma \\ &\quad - (H_\sigma - \sigma I)C(C^TC)^{-1}\begin{bmatrix} A^T \\ 0 \end{bmatrix} - S_\sigma(C^TC)^{-1}\begin{bmatrix} A^T \\ 0 \end{bmatrix} \\ &\approx B_2 := H_\sigma - \begin{bmatrix} A & 0 \end{bmatrix} C^\dagger(H_\sigma - \sigma I) - (H_\sigma - \sigma I)\left(C^\dagger\right)^T \begin{bmatrix} A^T \\ 0 \end{bmatrix}. \qquad (52) \end{aligned}$$

For products with the pseudoinverse and its transpose, we can compute

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = (C^TC)^{-1}C^Tv, \qquad v = C(C^TC)^{-1}\begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

by solving the respective block systems

$$\begin{bmatrix} I & A & B \\ A^T & & \\ B^T & & \end{bmatrix} \begin{bmatrix} t \\ u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} v \\ 0 \\ 0 \end{bmatrix}, \qquad \begin{bmatrix} I & A & B \\ A^T & & \\ B^T & & \end{bmatrix} \begin{bmatrix} v \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} 0 \\ -u_1 \\ -u_2 \end{bmatrix}. \tag{53}$$

Thus we can obtain the same types of Hessian approximations with two augmented system solves, except in B_2 they now correspond to a least-squares and least-norm solve instead of only projections as in section 4.4.

A.4. Proof of Theorem 11. We repeat the assumptions of Theorem 11:

(B1) (NP) satisfies (A1b) and (A2a).

(B2) x^* is a second-order KKT point for (NP) satisfying $\nabla^2\phi_\sigma(x^*) \succ 0$.

(B3) There exist $\bar\delta > 0$ and an open set $B(x^*)$ containing x^* such that if $\widetilde{x}_0 \in B(x^*)$ and $\delta \le \bar\delta$, the sequence $\widetilde{x}_{k+1} = \widetilde{x}_k + G(\phi_\sigma(\cdot;\delta), \widetilde{x}_k)$ converges quadratically to $x(\delta)$ with constant M independent of δ.

LEMMA 15. *Under the assumptions of Theorem 11:*

1. *$\phi_\sigma(\cdot;\delta)$ has two continuous derivatives for $\delta > 0$ and $x \in \mathbb{R}^n$ by (B1).*
2. *There exists an open set $B_1(x^*)$ containing x^* such that $\phi_\sigma(x)$ is well-defined and has two continuous derivatives for all $x \in B_1(x^*)$ by (B1).*
3. *$\nabla^2\phi_\sigma(x^*) = \nabla^2\phi_\sigma(x^*;0) \succ 0$ and $\phi_\sigma(x;\delta)$ is second-order smooth in both x and δ, so by assumption (B2) there exists an open set $B_2(x^*)$ containing x^* and $\widetilde{\delta} > 0$ such that $\nabla^2\phi_\sigma(x;\delta) \succ 0$ for $(x,\delta) \in B_2(x^*) \times [0,\widetilde{\delta}]$.*
4. *By Theorem 8, there exists $\hat{\delta}$ such that for $\delta \le \hat{\delta}$, $x(\delta)$ is continuous in δ. Therefore there exists an open set $B_3(x^*)$ such that $x(\delta) \in B_3(x^*)$ for $\delta \le \hat{\delta}$.*
5. *There exists a neighborhood $B_4(x^*)$ where Newton's method is quadratically convergent (with factor N) on $\phi_\sigma(x)$ by (B2).*
6. *Given $\delta_0 \le \bar{\delta}$, where $\bar{\delta}$ is defined in (B3), there exists a neighborhood $B_5(x^*)$ such that $\|\nabla\phi_\sigma(x;\delta_0)\| \le \delta_0$ for all $x \in B_5(x^*)$.*

We define

$$B'(x^*) := B(x^*) \cap \left(\bigcap_{i=1}^{5} B_i(x^*)\right) \qquad \text{and} \qquad \delta' := \min\{\bar{\delta}, \widetilde{\delta}, \hat{\delta}, 1\},$$

and note that x_k defined by Algorithm 3 satisfies $x_k \in B'(x^*)$ for all k by (B3). Because $\phi_\sigma(x;\delta)$ is a $\mathcal{C}_2$ function in $B'(x^*) \times [0,\delta']$, there exist positive constants $K_1,\ldots,K_5$ such that

7. *$\|\nabla\phi_\sigma(x;\delta)\| \le K_1$, $\|\nabla^2\phi_\sigma(x;\delta)^{-1}\| \le K_2$ for $x \in B'(x^*)$ and $\delta \le \delta'$;*
8. *$\|\nabla P_\delta(x)\| \le K_3$, $\|\nabla^2 P_\delta(x)\| \le K_4$ for $x \in B'(x^*)$ and $\delta \le \delta'$;*
9. *$\|x_k - x^*\| \le K_5\|\nabla\phi_\sigma(x_k)\|$.*

Proof. Statements 1–2 follow from (B1). Statements 3 and 5 follow from (B2). Statement 4 follows from Theorem 8.

Now consider Statement 6. For a given δ_0, we have $\nabla\phi_\sigma(x(\delta_0);\delta_0) = 0$ and so there exists a neighbourhood $\widetilde{B}$ around $x(\delta_0)$ such that $\|\nabla\phi_\sigma(x;\delta_0)\| \le \delta_0$ for all $x \in \widetilde{B}$. Further, $x(\delta_0) \in B_3(x^*)$, so let $B_5(x^*) = \widetilde{B} \cap B_3(x^*)$. ∎

We first give some technical results. All assume that $x_k \in B'(x^*)$ and $\delta_k \le \delta'$.

LEMMA 16. *Assume $\delta_{k-1} \le \delta_0 \le \delta'$. For δ_k defined according to* (31),

$$\|\nabla\phi_\sigma(x_k;\delta_k)\| = O(\delta_k).$$

Proof. The result holds for $k = 1$ in view of observation 6 of Lemma 15.

Because $x_k \in B'(x^*)$ and δ_k, $\delta_{k-1} \le \delta'$, observation 8 of Lemma 15 gives $\|\nabla P_{\delta_{k-1}}(x_k)\| \le K_3$ and $\|\nabla P_{\delta_k}(x_k)\| \le K_3$. Using (30), we have

$$\begin{aligned}
\|\nabla\phi_\sigma(x_k;\delta_k)\| &= \|\nabla\phi_\sigma(x_k;\delta_{k-1}) - \delta_{k-1}^2\nabla P_{\delta_{k-1}}(x_k) + \delta_k^2\nabla P_{\delta_k}(x_k)\| \\
&\le \|\nabla\phi_\sigma(x_k;\delta_{k-1})\| + \delta_{k-1}^2\|\nabla P_{\delta_{k-1}}(x_k)\| + \delta_k^2\|\nabla P_{\delta_k}(x_k)\| \\
&\le \|\nabla\phi_\sigma(x_k;\delta_{k-1})\| + (\delta_{k-1}^2 + \delta_k^2)K_3 \\
&= \|\nabla\phi_\sigma(x_k;\delta_{k-1})\| + O(\delta_k), \qquad (54)
\end{aligned}$$

where the last inequality follows from $\delta_k^2 \le \delta_k$ and (31), which implies that $\delta_k \ge \nu(\delta_{k-1}) = \delta_{k-1}^2$.

We consider two cases. If $\|\nabla\phi_\sigma(x_k;\delta_{k-1})\| \le \delta_{k-1}$, (31) implies that

$$\delta_k = \max(\|\nabla\phi_\sigma(x_k;\delta_{k-1})\|,\ \delta_{k-1}^2) \ge \|\nabla\phi_\sigma(x_k;\delta_{k-1})\|,$$

and therefore (54) gives $\|\nabla\phi_\sigma(x_k;\delta_k)\| = O(\delta_k)$.

Otherwise, (31) yields $\delta_k = \max(\delta_{k-1}, \delta_{k-1}^2) = \delta_{k-1}$, and there exists $\ell \le k-1$ such that $\delta_k = \delta_{k-1} = \cdots = \delta_\ell < \delta_{\ell-1}$, or $\ell = 1$. If $\delta_\ell < \delta_{\ell-1}$, step 3 of Algorithm 3 implies that $\|\nabla\phi_\sigma(x_\ell;\delta_{\ell-1})\| < \delta_{\ell-1}$, which by the above sequence of inequalities implies that $\|\nabla\phi_\sigma(x_\ell;\delta_\ell)\| = O(\delta_\ell)$. Then, because $\delta_k = \delta_\ell$,

$$\|\nabla\phi_\sigma(x_\ell,\delta_k)\| = \|\nabla\phi_\sigma(x_\ell,\delta_\ell)\| = O(\delta_\ell) = O(\delta_k).$$

Define the sequence $\{\widetilde{x}_j\}$ with $\widetilde{x}_0 = x_\ell$ and $\widetilde{x}_{j+1} = \widetilde{x}_j + G(\phi_\sigma(\cdot;\delta_\ell),\widetilde{x}_j)$. By (B3), $\widetilde{x}_j \to x(\delta_\ell)$ quadratically, so after j iterations of this procedure, for some $\widetilde{M}$, we have

$$\|\nabla\phi_\sigma(\widetilde{x}_j;\delta_k)\| \le \widetilde{M}^j\|\nabla\phi_\sigma(\widetilde{x}_0;\delta_k)\|^{2^j} \le \widetilde{M}^j K_1^{2^j-1}\|\nabla\phi_\sigma(\widetilde{x}_0;\delta_k)\|^2.$$

Then after $j = O(1)$ iterations of this procedure (depending only on $\widetilde{M}$ and K_1), we have $\widetilde{M}^j K_1^{2^j-1} \le 1$, so that

$$\|\nabla\phi_\sigma(\widetilde{x}_j;\delta_k)\| \le \|\nabla\phi_\sigma(\widetilde{x}_0;\delta_k)\|^2 = \|\nabla\phi_\sigma(x_\ell;\delta_k)\|^2 = O(\delta_k^2) < \delta_k.$$

Therefore, $k-\ell \le j = O(1)$, and by (B3),

$$\|\nabla\phi_\sigma(x_k;\delta_{k-1})\| = O\left(M^{k-\ell}\|\nabla\phi_\sigma(x_\ell;\delta_{k-1})\|^{2^{k-\ell}}\right) = O\left(M^{k-\ell}\delta_k^{2^{k-\ell}}\right) = O(\delta_k). \qquad \square$$

LEMMA 17. *For p_k defined by step 4 of Algorithm 3, $\|p_k\| = O(\delta_k)$.*

Proof. According to (B3), we may apply Lemma 10 with $\tau = 2$ and view step 4 of Algorithm 3 as an inexact-Newton step, i.e., there exists a constant $N_2 > 0$ such that

$$\nabla^2\phi_\sigma(x_k;\delta_k)p_k = -\nabla\phi_\sigma(x_k;\delta_k) + r_k, \qquad \|r_k\| \le N_2\|\nabla\phi_\sigma(x_k;\delta_k)\|^2.$$

Therefore by Lemma 16,

$$\begin{aligned}\|p_k\| &= \|\nabla^2\phi_\sigma(x_k;\delta_k)^{-1}(-\nabla\phi_\sigma(x_k;\delta_k) + r_k)\| \\ &\le \|\nabla^2\phi_\sigma(x_k;\delta_k)^{-1}\|\left(\|\nabla\phi_\sigma(x_k;\delta_k)\| + \|r_k\|\right) \\ &\le K_2\left(\|\nabla\phi_\sigma(x_k;\delta_k)\| + N_2\|\nabla\phi_\sigma(x_k;\delta_k)\|^2\right) \\ &\le K_2\left(O(\delta_k) + O(\delta_k^2)\right) = O(\delta_k). \qquad \square\end{aligned}$$

LEMMA 18. *Let p_k be defined by step 4 of Algorithm 3 and q_k be the Newton direction for the unregularized penalty function defined by $\nabla^2\phi_\sigma(x_k)q_k = -\nabla\phi_\sigma(x_k)$. Then $\|p_k - q_k\| \in O(\delta_k^2)$.*

Proof. According to (B3), we may apply Lemma 10 with $\tau = 2$ and view step 4 of Algorithm 3 as an inexact-Newton step, i.e.,

$$\nabla^2\phi_\sigma(x_k;\delta_k)p_k = -\nabla\phi_\sigma(x_k;\delta_k) + r_k, \tag{55a}$$

$$\|r_k\| = O(\|\nabla\phi_\sigma(x_k;\delta_k)\|^2). \tag{55b}$$

We premultiply (55a) by $\nabla^2\phi_\sigma(x_k)^{-1}$ and use (30) to obtain

$$p_k + \delta_k^2\nabla^2\phi_\sigma(x_k)^{-1}\nabla^2 P_{\delta_k}(x_k)p_k = q_k + \delta_k^2\nabla^2\phi_\sigma(x_k)^{-1}\nabla P_{\delta_k}(x_k) + \nabla^2\phi_\sigma(x_k)^{-1}r_k.$$

Lemma 16, Lemma 17 and the triangle inequality then yield

$$\begin{aligned}\|p_k - q_k\| &= \left\|\delta_k^2\nabla^2\phi_\sigma(x_k)^{-1}\left(\nabla P_{\delta_k}(x_k) - \nabla^2 P_{\delta_k}(x_k)p_k\right) + \nabla^2\phi_\sigma(x_k)^{-1}r_k\right\| \\ &\le \delta_k^2\|\nabla^2\phi_\sigma(x_k)^{-1}\|\left(\|\nabla P_{\delta_k}(x_k)\| + \|\nabla^2 P_{\delta_k}(x_k)p_k\|\right) + \|\nabla^2\phi_\sigma(x_k)^{-1}r_k\| \\ &\le \delta_k^2 K_2\left(K_3 + O(\delta_k)\right) + O(\delta_k^2) = O(\delta_k^2). \qquad \square\end{aligned}$$

Using the previous technical results, we are in position to establish our main result.

Proof of Theorem 11. We show that for $x_0 \in B'(x^*)$ we achieve R-quadratic convergence, by showing that $\|x_k - x^*\| = O(\delta_k)$ and that $\delta_k \to 0$ quadratically. By observation 9 of Lemma 15, (30), the triangle inequality, Lemma 16, and observation 8 of Lemma 15, we have

$$\begin{aligned}\|x_k - x^*\| &\le K_5\|\nabla\phi_\sigma(x_k)\| \\ &= K_5\|\nabla\phi_\sigma(x_k;\delta_k) - \delta_k^2\nabla P_{\delta_k}(x_k)\| \\ &\le K_5(\|\nabla\phi_\sigma(x_k;\delta_k)\| + \delta_k^2\|\nabla P_{\delta_k}(x_k)\|) \\ &\le K_5\left(O(\delta_k) + \delta_k^2 K_3\right) = O(\delta_k).\end{aligned}$$

Let q_k be the Newton direction defined in Lemma 18. There exists a constant $N > 0$ such that

$$\begin{aligned}\|x_{k+1} - x^*\| &= \|x_k + p_k - x^*\| \\ &\le \|x_k + q_k - x^*\| + \|p_k - q_k\| \\ &\le N\|x_k - x^*\|^2 + \|p_k - q_k\| = O(\delta_k^2).\end{aligned}$$

It remains to show that δ_k decreases quadratically. If $\|\nabla\phi_\sigma(x_{k+1},\delta_k)\| \le \delta_k^2$,

$$\delta_{k+1} = \max\{\min\{\|\nabla\phi_\sigma(x_{k+1},\delta_k)\|,\delta_k\},\delta_k^2\} \le \max\{\|\nabla\phi_\sigma(x_{k+1},\delta_k)\|,\delta_k^2\} = \delta_k^2.$$

Assume now that $\|\nabla\phi_\sigma(x_{k+1},\delta_k)\| > \delta_k^2$. We have from (30) and observations 7–8 of Lemma 15 that

$$\begin{aligned}\delta_{k+1} &= \max\{\min\{\|\nabla\phi_\sigma(x_{k+1},\delta_k)\|,\delta_k\},\delta_k^2\} \\ &\le \|\nabla\phi_\sigma(x_{k+1},\delta_k)\| \\ &\le \|\nabla\phi_\sigma(x_{k+1})\| + \delta_k^2\|\nabla P_{\delta_k}^2(x_{k+1})\| \\ &\le K_2^{-1}\|x_{k+1} - x^*\| + \delta_k^2 K_3 = O(\delta_k^2).\end{aligned}$$

Thus we have $\|x_k - x^*\| = O(\delta_k)$ and $\delta_{k+1} = O(\delta_k^2)$, which means that $x_k \to x^*$ R-quadratically. ◻

Acknowledgements. We would like to express our deep gratitude to Drew Kouri for supplying PDE-constrained optimization problems in Matlab, for helpful discussions throughout this project, and for hosting the first author for two weeks at Sandia National Laboratories.

References.

M. Arioli. Generalized Golub-Kahan bidiagonalization and stopping criteria. *SIAM J. Matrix Anal. Appl.*, 34(2):571–592, 2013. DOI: 10.1137/120866543.

D. P. Bertsekas. Necessary and sufficient conditions for a penalty method to be exact. *Math. Program.*, 9:87–99, 1975.

D. P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods.* Academic Press, New York, 1982.

A. Björck. Iterative refinement of linear least squares solutions. I. *Nordisk Tidskr. Informations-Behandling (BIT)*, 7:257–278, 1967.

A. Björck and C. C. Paige. Solution of augmented linear systems using orthogonal factorizations. *BIT*, 34(1):1–24, 1994. DOI: 10.1007/BF01935013.

R. H. Byrd, J. Nocedal, and R. A. Waltz. KNITRO: An integrated package for nonlinear optimization. In G. di Pillo and M. Roma, editors, *Large-Scale Nonlinear Optimization*, pages 35–59. Springer-Verlag, New York, 2006.

A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *Trust-Region Methods*. MPS-SIAM Series on Optimization. SIAM, Philadelphia, 2000.

R. Courant. Variational methods for the solution of problems with equilibrium and variation. *Bull. Amer. Math. Soc.*, 49:1–23, 1943.

J. E. Craig. The N-step iteration procedures. *Journal of Mathematics and Physics*, 34 (1):64–73, 1955.

R. S. Dembo, S. C. Eisenstat, and T. Steihaug. Inexact Newton methods. *SIAM J. Numer. Anal.*, 19(2):400–408, 1982.

G. di Pillo and L. Grippo. A class of continuously differentiable exact penalty function algorithms for nonlinear programming problems. In E. P. Toft-Christensen, editor, *System Modelling and Optimization*, pages 246–256. Springer-Verlag, Berlin, 1984.

G. di Pillo and L. Grippo. An exact penalty function method with global convergence properties for nonlinear programming problems. *Math. Programming*, 36(1):1–18, 1986. DOI: 10.1007/BF02591986.

R. Estrin, D. Orban, and M. A. Saunders. LSLQ: An iterative method for linear least-squares with an error minimization property. Cahier du GERAD G-2017-05, GERAD, Montréal, QC, Canada, 2017. To appear in SIMAX.

R. Estrin, D. Orban, and M. A. Saunders. LNLQ: An iterative method for least-norm problems with an error minimization property. Cahier du GERAD G-2018-40, GERAD, Montréal, QC, Canada, 2018.

R. Fletcher. A class of methods for nonlinear programming with termination and convergence properties. In J. Abadie, editor, *Integer and Nonlinear Programming*, pages 157–175. North-Holland, Amsterdam, 1970.

R. Fletcher. A class of methods for nonlinear programming: III. rates of convergence. In F. A. Lootsma, editor, *Numerical Methods for Nonlinear Optimization*. Academic Press, New York, 1973.

P. E. Gill, M. A. Saunders, and J. R. Shinnerl. On the stability of Cholesky factorization for symmetric quasidefinite systems. *SIAM J. Matrix Anal. Appl.*, 17(1):35–46, January 1996.

G. H. Golub and V. Pereyra. The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate. *SIAM J. Numer. Anal.*, 10:413–432, 1973. DOI: 10.1137/0710036. Collection of articles dedicated to the memory of George E. Forsythe.

G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, fourth edition, 2013. ISBN 978-1-4214-0794-4; 1-4214-0794-9; 978-1-4214-0859-0.

N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEr and SifDec: A constrained and unconstrained testing environment, revisited. *ACM Trans. Math. Softw.*, 29(4): 373–394, Dec. 2003.

M. Heinkenschloss and D. Ridzal. A matrix-free trust-region SQP method for equality constrained optimization. *SIAM J. Optim.*, 24(3):1507–1541, 2014. DOI: 10.1137/130921738.

M. R. Hestenes. Multiplier and gradient methods. *J. Optim. Theory Appl.*, 4(5): 303–320, 1969.

M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Research Nat. Bur. Standards*, 49:409–436 (1953), 1952.

D. P. Kouri, M. Heinkenschloss, D. Ridzal, and B. G. van Bloemen Waanders. Inexact objective function evaluations in a trust-region algorithm for PDE-constrained optimization under uncertainty. *SIAM J. Sci. Comput.*, 36(6):A3011–A3029, 2014. DOI: 10.1137/140955665.

C.-J. Lin and J. J. Moré. Incomplete cholesky factorizations with limited memory. *SIAM J. Comput.*, 21(1):24–45, 1999a.

C.-J. Lin and J. J. Moré. Newton's method for large bound-constrained optimization problems. *SIAM J. Optim.*, 9(4):1100–1127, 1999b.

N. Maratos. *Exact Penalty Function Algorithms for Finite Dimensional and Optimization Problems*. PhD thesis, Imperial College of Science and Technology, London, UK, 1978.

Ph. L. Toint. Towards an efficient sparsity exploiting Newton method for minimization. In I. S. Duff, editor, *Sparse Matrices and Their Uses*, pages 57–88, London, 1981. Academic Press.

J. L. Morales and J. Nocedal. Automatic preconditioning by limited memory quasi-Newton updating. *SIAM J. Optim.*, 10(4):1079–1096, 2000.

H. Mukai and E. Polak. A quadratically convergent primal-dual algorithm with global convergence properties for solving optimization problems with equality constraints. *Math. Programming*, 9(3):336–349, 1975.

J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, second edition, 2006.

D. Orban and M. Arioli. *Iterative Solution of Symmetric Quasi-definite Linear Systems*, volume 3 of *SIAM Spotlights*. SIAM, Philadelphia, PA, 2017. ISBN 978-1-611974-72-0.

J. M. Ortega and W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. SIAM, Philadelphia, 2000.

C. C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM J. Numer. Anal.*, 12:617–629, 1975.

C. C. Paige and M. A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Softw.*, 8:43–71, 1982.

M. J. D. Powell. A method for nonlinear constraints in minimization problems. In R. Fletcher, editor, *Optimization*, chapter 19. Academic Press, London and New York, 1969.

Y. Saad and M. H. Schultz. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 7(3):856–869, 1986. DOI: 10.1137/0907058.

M. A. Saunders. Solution of sparse rectangular systems using LSQR and Craig. *BIT*, 35(4):588–604, 1995. DOI: 10.1007/BF01739829.

T. Steihaug. The conjugate gradient method and trust regions in large scale optimization. *SIAM J. Numer. Anal.*, 20(3):626–637, 1983. DOI: 10.1137/0720042.

R. J. Vanderbei. Symmetric quasi-definite matrices. *SIAM J. Optim.*, 5(1):100–113, 1995.

S. J. Wright. Superlinear convergence of a stabilized SQP method to a degenerate solution. *Comput. Optim. Appl.*, 11(3):253–275, 1998.

V. M. Zavala and M. Anitescu. Scalable nonlinear programming via exact differentiable penalty functions and trust-region Newton methods. *SIAM J. Optim.*, 24(1):528–558, 2014.