SCALABLE NONLINEAR PROGRAMMING VIA EXACT DIFFERENTIABLE PENALTY FUNCTIONS AND TRUST-REGION NEWTON METHODS[*]

VICTOR M. ZAVALA AND MIHAI ANITESCU[†]

Abstract. We present an approach for nonlinear programming (NLP) based on the direct minimization of an exact differentiable penalty function using trust-region Newton techniques. As opposed to existing algorithmic approaches to NLP, the approach provides all the features required for scalability: it can efficiently detect and exploit directions of negative curvature, it is superlinearly convergent, and it enables the scalable computation of the Newton step through iterative linear algebra. Moreover, it presents features that are desirable for parametric optimization problems that have to be solved in a latency-limited environment, as is the case for model predictive control and mixed-integer nonlinear programming. These features are fast detection of activity, efficient warm-starting, and progress on a primal-dual merit function at every iteration. We derive general convergence results for our approach and demonstrate its behavior through numerical studies.

Key words. scalable, nonlinear programming, exact differentiable penalty, trust-region, Newton, iterative linear algebra

AMS subject classifications. 34B15, 34H05, 49N35, 49N90, 90C06, 90C30, 90C55, 90C59

1. Problem Definition and Previous Work.

We consider the general nonlinear programming (NLP) problem and its parametric version (pNLP),

$$\begin{aligned} &\min_x \quad f(x) && (1.1a) \\ &\text{s.t.} \quad h(x) = 0_m \quad (\lambda) && (1.1b) \\ &\qquad\quad x \geq 0_n \quad (\nu). && (1.1c) \end{aligned} \qquad \begin{aligned} &\min_x \quad f(x,t) && (1.2a) \\ &\text{s.t.} \quad h(x,t) = 0_m \quad (\lambda) && (1.2b) \\ &\qquad\quad x \geq 0_n \quad (\nu). && (1.2c) \end{aligned}$$

Here, $x \in \mathcal{R}^n$ are primal variables and $\lambda \in \mathcal{R}^m$, $\nu \in \mathcal{R}^n$ are multipliers for the equality constraints and bounds, respectively, and $t \in \mathcal{R}$ is a scalar parameter. We note that any NLP with general inequality constraints can be transformed into this form without destroying the properties of the approach derived in this work.

Our motivation for considering pNLP is model predictive control (MPC). In this area, we have recently shown that solving inexactly pNLP successively in t by using one major sequential quadratic programming (SQP) iteration on the attached NLP is asymptotically stable [42]. Here we will analyze exclusively NLP, but the application of the resulting algorithms to pNLP will motivate us seeking them to have certain properties. In particular, a critical factor in MPC is *latency:* the time before an improved decision (control) is computed. While in [42] that is the time it takes to do a QP solve, in this work we aim to reduce this even further for cases where the dimension of the underlying NLP is very large.

Our objectives in designing the algorithms for such problems are two fold. First, we seek properties that allow very large NLPs to be solved efficiently. To this end, we seek an algorithm with global convergence properties and the following features:

i) **Superlinear convergence** in the primal-dual space.
ii) **Scalable step computation**. It is matrix-free in that it does not require matrix factorizations, it enables use of iterative linear algebra, and it detects and exploits directions of negative curvature.

[*]Preprint ANL/MCS-P3014-0712.

[†] Mathematics and Computer Science Division, Argonne National Laboratory, 9700 South Cass Avenue, Argonne, IL 60439, USA (vzavala@mcs.anl.gov, anitescu@mcs.anl.gov)

Second, we seek good properties for pNLP (1.2) in latency-limited environments, such as MPC, where the problem at t may need to be solved incompletely because of low-latency requirements before new data comes in and the clock is advanced to $t + \Delta t$. Yet, even in this circumstance it is desirable to show improvement in a way that seamlessly transitions into good NLP convergence properties, so properties (i) and (ii) are still relevant. This occurs, for example, in MPC when responding to a parametric perturbation, where one restarts relatively far away from the new optimal manifold. In addition, the latency concern adds the following objectives

iii) **Asymptotic monotonicity in minor iterations**. Each minor iteration has small complexity and makes constant progress in a primal-dual function, at least sufficiently close to the solution.
iv) **Active set detection and warm-start**. The algorithm enables the efficient use and detection of activity.

The main algorithmic frameworks found to date can be broadly categorized as interior-point (IP), active set SQP, and active set augmented Lagrangian (AL). IP algorithms satisfy (i) [16]. Property (ii) is currently achieved by using incomplete symmetric indefinite factorizations for preconditioning the augmented system, detecting its inertia [35] or testing the resulting direction for descent [12], and modifying the Hessian matrix to achieve the correct inertia or other criteria and recalculating the search direction [39, 12]. Another approach consists in using a constraint-projected preconditioned conjugate gradient (PCG) scheme for the augmented system with constraint preconditioners [18] exploiting negative curvature through trust-region regularization [40]. While this does not require the factorization of the KKT matrix, however, it does require the factorization of the constraint preconditioner and thus does not satisfy (ii). Property (iii) is currently enforced indirectly by using filter mechanisms and/or primal merit functions [39, 9, 40]. To achieve (i) in this context, however, second-order corrections, are needed to avoid the Maratos effect [38]. As for property (iv), detection is efficient from cold-start. For warm-starting, reductions in number of iteration count are achieved by recovering centrality or crossing-over to SQP but this is cumbersome and computationally costly [21, 41], so IP algorithms do not generally fare well with respect to (iv). SQP algorithms largely mimic the features of IP with two exceptions. Emphasis on exact satisfaction of the linearized constraints generally requires a factorization, though perhaps only the one of a constraint preconditioner, so (ii) may not be satisfied. With respect to (iv), cold-start active set detection is inefficient but warm-start is possible. Detecting multiple changes in activity, however, is inefficient, so (iv) is not entirely satisfied. AL frameworks can achieve (ii) by using iterative linear algebra on the Hessian of the augmented Lagrangian [11, 42]. Property (iii) can be achieved by using the AL function directly as merit function [10]. Property (iv) can be achieved by using gradient projection methods for the bound-constrained AL subproblems [6, 42]. Property (i), however, is satisfied only when second-order multiplier updates or linearly-constrained formulations are used [4, 23]. This requires major linear algebra operations and thus prevents satisfaction of (ii). An alternative is to use primal-dual augmented Lagrangian functions [20] to compute steps on both primal and dual variables simultaneously, but it is still not clear how to ensure (ii), (iii), and (iv).

In this work, we build on an interesting class of exact differentiable penalty functions (EDPFs) that was proposed and analyzed by DiPillo and Grippo for equality-constrained NLPs [13]. These functions are primal-dual AL functions of the form in (3.4) having the distinctive feature that they incorporate a penalty term for the norm of the gradient of the Lagrangian. In addition to the good global convergence properties typical of exact penalty functions, DiPillo and Grippo's function has powerful properties that can be exploited to construct algorithms that directly minimize this merit function. This satisfies properties (i) and (iii). As for property (ii), the main limitation in designing algorithms based on direct minimization of DiPillo and Grippo's function is the appearance of complex third-order derivative terms. While these terms can be shown to vanish at a critical point of the NLP, they greatly complicate the search because they introduce nonconvexity. Consequently, capabilities to detect and exploit directions of negative curvature are essential in order to enable (ii) for such EDPFs. As for property (iv), DiPillo and Grippo proposed exact penalty functions to deal with inequality constraints but these are twice differentiable only in a neighborhood of the solution thus preventing (iii) and hindering (ii) [31, 14]. A reformulation, central to this work, that maintains the smoothness of the EDPF by enforcing bound constraints explicitly is presented by Bertsekas [4, pp.229-230]. The function has the form (3.10). The function is twice differentiable everywhere thus enabling (iii) and potentially (ii) if capabilities to exploit directions of negative curvature are devised. Moreover, to put the comparison with other methods on an equal footing, an efficient method based on such EDPFs should avoid evaluation of third derivatives without hindering (i) or global convergence.

In this work, we demonstrate that one can construct EDPF NLP algorithms capable of achieving all the properties (i)-(iv) while avoiding the computation of third derivatives and efficiently handling their nonconvexity. The latter two objectives are accomplished by using a trust-region/gradient-projection approach similar to [25] around the bound constrained EDPF formulation (3.13) with EDPF function (3.10). Our contributions are (1) We formalize the properties of the EDPF to establish conditions under which first- and second-order conditions for critical points of the NLP and of the bound constrained EDPF (3.13) coincide in the case of inequality constraints. This extends existing results for the case of equality constraints [4, 13]. (2) We derive formulas for the approximate Hessian and the reduced approximate Hessian for the EDPF and discuss their asymptotic convergence properties to the exact counterparts. (3) We demonstrate that superlinear and global convergence can be achieved using a trust-region Newton framework [25]. (4) We demonstrate that the framework satisfies properties (i)-(iv) and present numerical examples to illustrate this.

The paper is structured as follows. In Section 2 we introduce basic notation and properties of the NLP. In Section 3 we reformulate the NLP as a bound-constrained minimization problem using the EDPF reformulation. In Section 4 we present critical point identification properties of the reformulation. In Section 5 we present the trust-region Newton algorithm and convergence results. Numerical results are presented in Section 6. The last section presents concluding remarks and directions of future work.

2. Basic Properties and Notation.

We define the *partial* Lagrange function of the NLP (1.1):

$$\mathcal{L}(x,\lambda) = f(x) + \lambda^T h(x). \tag{2.1}$$

$$\tag{2.2}$$

Here, $f : \mathcal{R}^n \to \mathcal{R}$ and $h : \mathcal{R}^m \to \mathcal{R}^n$ are at least three times continuously differentiable. The first and second derivatives of the Lagrange function are

$$\nabla_x \mathcal{L}(x,\lambda) = \nabla_x f(x) + \nabla_x h(x)^T \lambda \tag{2.3a}$$

$$\nabla_\lambda \mathcal{L}(x,\lambda) = h(x) \tag{2.3b}$$

$$\nabla_{x,x} \mathcal{L}(x,\lambda) = \nabla_{x,x} f(x) + \nabla_x (\nabla_x h(x)^T \lambda) \tag{2.3c}$$

$$\nabla_{\lambda,x} \mathcal{L} = \nabla_x h(x) \tag{2.3d}$$

$$\nabla_{\lambda,\lambda} \mathcal{L} = 0_{m\times m}. \tag{2.3e}$$

Here, $\nabla_x h(x) \in \mathcal{R}^{m\times n}$. We also define the primal-dual vector $w^T := [x^T, \lambda^T]$ with $w \in \mathcal{R}^{n_w}$ and $n_w = n + m$. With this, we can write

$$\nabla_w \mathcal{L}(w) = \begin{bmatrix} \nabla_x \mathcal{L}(x,\lambda) \\ h(x) \end{bmatrix} \tag{2.4a}$$

$$\nabla_{w,w} \mathcal{L}(w) = \begin{bmatrix} \nabla_{x,x} \mathcal{L}(x,\lambda) & \nabla_x h(x)^T \\ \nabla_x h(x) & 0_{m\times m} \end{bmatrix}. \tag{2.4b}$$

We denote a solution of the NLP (1.1) using the pair x^*, λ^* or w^*. When convenient, we will use the dual vector μ^* for the bounds explicitly. The Karush-Kuhn-Tucker (KKT) conditions of the NLP (1.1) are given by

$$0_n = \nabla_x \mathcal{L}(x^*,\lambda^*) - \mu^* \tag{2.5a}$$

$$0_m = \nabla_\lambda \mathcal{L}(x^*,\lambda^*) \tag{2.5b}$$

$$0_n \le x^* \perp \mu^* \ge 0_n. \tag{2.5c}$$

In compact form,

$$0_n \le x^* \perp \nabla_x \mathcal{L}(x^*,\lambda^*) \ge 0_n \tag{2.6a}$$

$$0_m = \nabla_\lambda \mathcal{L}(x^*,\lambda^*). \tag{2.6b}$$

The first expression above also implies that $X^* \nabla_x \mathcal{L}(x^*,\lambda^*) = 0_n$ with $X^* := \text{diag}(x^*)$. We define the active set $\mathcal{A}(x^*) := \{\ i \in \{1..n\} \mid x^*_{(i)} = 0\ \}$ and the inactive set $\mathcal{I}(x^*)$ where $\mathcal{I}(x^*) \cup \mathcal{A}(x^*) = \{1..n\}$ and $\text{card}(\mathcal{I}(x^*)) := n^*_d$. We also define the inactive (free) vector $x^*_d \in \mathcal{R}^{n^*_d}$ and corresponding null-space or basis matrix *for the bound constraints* (1.1c) $N_x(x^*) \in \mathcal{R}^{n\times n^*_d}$ such that x^* can be written as $x^* = N_x(x^*)x^*_d$. In our case, the structure of $N_x(\cdot)$ is trivial. Using these definitions, we make the following assumptions for first-order necessary conditions (KKT) and sufficient second-order conditions (SSOC) for the NLP (1.1).

We first assume that any stationary point x^* of the NLP (1.1) satisfies the linear independence constraint qualification (LICQ) and strict complementarity (SC). This implies (Theorem 12.1 in [28]) that there exists a unique Lagrange multiplier λ^* satisfying the KKT conditions (2.6). Also, under LICQ and SC we can define the subspace of $\mathcal{R}^n$,

$$\mathcal{F}(x^*,\lambda^*) := \text{Null}\,[\nabla_x h(x^*)] \cap \mathcal{S}(x^*,\lambda^*), \tag{2.7}$$

where

$$\mathcal{S}(x^*,\lambda^*) := \left\{ x \in \mathcal{R}^n \mid x_{(i)} = 0,\ i \in \mathcal{A}(x^*) \right\}. \tag{2.8}$$

For SSOC, we assume that for a feasible point x^* there is a Lagrange multiplier λ^* satisfying the KKT conditions (2.6) and that

$$\nu^T \nabla_{x,x} \mathcal{L}(x^*,\lambda^*)\nu > 0, \quad \nu \in \mathcal{F}(x^*,\lambda^*), \quad \nu \ne 0. \tag{2.9}$$

This implies (Theorem 12.6 in [28]) that x^* is a strict local solution of the NLP (1.1).

We can also state the SSOC by first restricting ν to the subspace $\mathcal{S}(x^*,\lambda^*)$. Specifically, we write $\nu = N_x(x^*)\nu_x$, where $\nu_x \in \mathcal{R}^{n^*_d}$ and where the columns of $N_x(x^*)$ form a basis for the subspace $\mathcal{S}(x^*,\lambda^*)$. With this, the SSOC conditions can be stated for $\nu_x \in \mathcal{R}^{n^*_d}$ such that $N_x(x^*)\nu_x \ne 0$ and $\nabla_x h(x^*)N_x(x^*)\nu_x = 0_m$.

To summarize, we require the following assumptions for the NLP (1.1).

- A1. The functions $f(\cdot), h(\cdot)$ are at least three times continuously differentiable.
- A2. Any stationary point x^* of the NLP (1.1) satisfies LICQ and SC.
- A3. Any stationary point x^* of the NLP (1.1) satisfies the SSOC condition (2.9).

3. Exact Differentiable Penalty Formulation. To explain the derivation of the EDPF, we use the observation that one can eliminate the inequality constraints of the NLP (1.1) by using squared slack variables $z \in \mathcal{R}^n$ [22, 4]. This yields

$$\min_x \quad f(x) \quad \text{s.t.} \quad h(x) = 0_m, \quad x = z^2. \tag{3.1}$$

The full Lagrange function of this problem is given by

$$\bar{\mathcal{L}}(x, \lambda, z, \mu) = \mathcal{L}(x, \lambda) - \mu^T (x - z^2). \tag{3.2}$$

A solution $\tilde{x}^*, \tilde{\lambda}^*, \tilde{z}^*, \tilde{\mu}^*$ satisfies the KKT conditions,

$$0_n = \nabla_x \bar{\mathcal{L}}(\tilde{x}^*, \tilde{\lambda}^*, \tilde{z}^*, \tilde{\mu}^*) = \nabla_x \mathcal{L}(\tilde{x}^*, \tilde{\lambda}^*) - \tilde{\mu}^* \tag{3.3a}$$

$$0_n = \nabla_z \bar{\mathcal{L}}(\tilde{x}^*, \tilde{\lambda}^*, \tilde{z}^*, \tilde{\mu}^*) = 2 \cdot \tilde{\mu}^* \tilde{Z}^* \tag{3.3b}$$

$$0_m = \nabla_\lambda \bar{\mathcal{L}}(\tilde{x}^*, \tilde{\lambda}^*, \tilde{z}^*, \tilde{\mu}^*) = h(\tilde{x}^*) \tag{3.3c}$$

$$0_n = \nabla_\mu \bar{\mathcal{L}}(\tilde{x}^*, \tilde{\lambda}^*, \tilde{z}^*, \tilde{\mu}^*) \tilde{x}^* - (\tilde{z}^*)^2 = 0. \tag{3.3d}$$

where $Z = \text{diag}(z)$. We can see that at a solution $\tilde{x}^*$, we have $\tilde{z}^*_{(i)} = \sqrt{\tilde{x}^*_{(i)}}$, $i = 1, ..., n$, so that $\tilde{x}^*_{(i)} > 0$ implies $\sqrt{\tilde{x}^*_i} > 0$ and $\tilde{\mu}^*_{(i)} = 0$ and vice-versa. In other words, (3.3b) and (3.3d) represent the complementarity conditions between the gradient of the Lagrangian and the primal variables (2.6a).

It is not difficult to show that $\tilde{x}^*$ is a stationary point for the reformulated NLP (1.1) if and only if $\tilde{x}^*, \sqrt{\tilde{x}^*}$ is a stationary point of the original NLP (3.1). In addition, if the SSOC, LICQ, and SC hold for the NLP (1.1) at $\tilde{x}^*$ (A2,A3), then these properties are inherited by (3.1) (see Proposition 1.32 in [4]). Hence, we can solve the NLP (1.1) indirectly by solving the equality-constrained NLP (3.1).

The squared-slacks reformulation has been widely used particularly in the control community [29, 15, 27] as a way to eliminate inequality constraints. An important caveat of this reformulation, however, arises from a computational point of view because the introduction of squared slacks can lead to numerical instability and poor solver performance [42, 2, 19]. This is related primarily to the introduction of the complementarity condition (3.3b).

While not directly amenable for computation, Bertsekas noticed that reformulation (3.1) provides a setting to construct EDPFs. He considered the penalty function of Di Pillo and Grippo [31] for the equality-constrained NLP min$f(x)$ s.t. $h(x) = 0$,

$$P_{\alpha,\beta}(w) = \mathcal{L}(w) + \frac{1}{2} \alpha h(x)^T h(x) + \frac{1}{2} \beta \, \nabla_x \mathcal{L}(w)^T \nabla_x \mathcal{L}(w), \tag{3.4}$$

where $P_{\alpha,\beta} : \mathcal{R}^{n_w} \to \mathcal{R}$ and $\alpha, \beta \in \mathcal{R}_+$ are parameters. In condensed form,

$$P_{\alpha,\beta}(w) = \mathcal{L}(w) + \frac{1}{2} \nabla_w \mathcal{L}(w)^T K_{\alpha,\beta} \nabla_w \mathcal{L}(w), \tag{3.5}$$

where

$$K_{\alpha,\beta} = \begin{bmatrix} \beta \mathbb{I}_n & \\ & \alpha I_m \end{bmatrix}. \tag{3.6}$$

We note that the partial Lagrange function of the reformulated problem (3.1) can also be expressed as

$$\mathcal{L}(z^2, \lambda) = f(z^2) + \lambda^T h(z^2). \tag{3.7}$$

The stationarity condition of (3.1) with respect to z is

$$\nabla_z \mathcal{L}(z^2, \lambda) = 2Z \left(\nabla_x f(z^2) + \nabla_x h(z^2)^T \lambda \right). \tag{3.8}$$

Applying the function (3.4) to problem (3.1), we obtain

$$\tilde{P}_{\alpha,\beta}(z,\lambda) = \mathcal{L}(z,\lambda) + \frac{1}{2}\alpha h(z^2)^T h(z^2) + 2\beta \left(\nabla_x f(z^2) + \nabla_x h(z^2)^T \lambda \right)^T ZZ \left(\nabla_x f(z^2) + \nabla_x h(z^2)^T \lambda \right). \tag{3.9}$$

We can see that the last term includes the complementarity condition between the gradient of the Lagrangian and the primal variables. To prevent divergence of the squared slacks, we follow [4, pp.229-230] and use an equivalent formulation of the EDPF in terms of x and enforce $x \geq 0_n$. This leads to the EDPF,

$$P_{\alpha,\beta}(x,\lambda) = \mathcal{L}(x,\lambda) + \frac{1}{2}\alpha h(x)^T h(x) + 2\beta \nabla_x \mathcal{L}(x,\lambda)^T X \nabla_x \mathcal{L}(x,\lambda). \tag{3.10}$$

In compact form,

$$P_{\alpha,\beta}(x,\lambda) = \mathcal{L}(x,\lambda) + \frac{1}{2}\nabla_w \mathcal{L}(x,\lambda)^T K_{\alpha,\beta}(x) \nabla_w \mathcal{L}(x,\lambda), \tag{3.11}$$

with

$$K_{\alpha,\beta}(x,\lambda) = \begin{bmatrix} 4\beta X & \\ & \alpha I_m \end{bmatrix}. \tag{3.12}$$

We can thus, in principle, solve the original NLP (1.1) by solving the EDPF minimization problem,

$$\min_{x,\lambda} \quad P_{\alpha,\beta}(x,\lambda) \quad \text{s.t.} \quad x \geq 0_n. \tag{3.13}$$

The KKT conditions for a solution $x^*(\alpha,\beta), \lambda^*(\alpha,\beta)$ are

$$0_n \leq x^* \perp \nabla_x P_{\alpha,\beta}(x^*,\lambda^*) \geq 0_n \tag{3.14a}$$

$$0_m = \nabla_\lambda P_{\alpha,\beta}(x^*,\lambda^*). \tag{3.14b}$$

We define the active set at w^* as $\mathcal{A}_P(w^*) := \{ \, i \in 1..n_w \mid w^*_{(i)} = 0 \}$. We also define the inactive set $\mathcal{I}_P(w^*)$, where $\mathcal{I}_P(w^*) \cup \mathcal{A}(w^*) = \{1..n_w\}$ and $\text{card}(\mathcal{I}(w^*)) := d$, where d are the number of inactive (free) variables in the primal-dual vector w^*.

The SSOC for the EDPF problem can be stated as,

$$\nu^T \nabla_{w,w} P_{\alpha,\beta}(w^*) \nu > 0, \quad \nu \in \mathcal{S}_P(w^*), \tag{3.15}$$

where

$$\mathcal{S}_P(w^*) := \left\{ w \in \mathcal{R}^{n_w} \mid w_{(i)} = 0, \quad i \in \mathcal{B}_P(w^*) \right\} \tag{3.16a}$$

$$\mathcal{B}_P(w^*) := \left\{ i \in \mathcal{A}_P(w^*) \mid \nabla_w P(w^*)_{(i)} > 0 \right\}. \tag{3.16b}$$

Several key properties result from the EDPF formulation (3.13). First, the EDPF problem can be solved as an *NLP with box constraints*. As we see in Section 5, this enables the use of trust-region Newton techniques and gradient projection [25] in order to obtain global and superlinear convergence and fast identification of activity. Second, the objective of (3.13) is a *natural merit function*. Improvement of the objective value implies progress on the Lagrangian, primal infeasibility, or dual infeasibility. Consequently, similar to Fletcher's AL, direct minimization of the EDPF can in principle avoid the Maratos effect. To obtain these desirable features, we first analyze the properties of the derivatives of the EDPF and establish conditions for parameters α, β for which minimizers of (3.13) coincide with those of the NLP (1.1).

4. Properties of Exact Penalty Formulation.

Throughout this section we simplify notation as $P \leftarrow P_{\alpha,\beta}(w)$, $\nabla\mathcal{L} \leftarrow \nabla\mathcal{L}(w)$, $K \leftarrow K_{\alpha,\beta}(w)$ and write

$$P = \mathcal{L} + \frac{1}{2}\nabla\mathcal{L}^T K \nabla\mathcal{L}. \tag{4.1}$$

We also define $\nabla P = \nabla_w \mathcal{P}$, $\nabla^2 P = \nabla_{w,w} P$ and a diagonal matrix $\Gamma \in \mathcal{R}^{n_w \times n_w}$ with entries $\Gamma_{i,i} = 4\beta,\ i = 1, ..., n$ and $\Gamma_{i,i} = 0,\ i = n+1, ..., n_w$. Using these definitions, we have

$$\begin{aligned}\nabla P &= \nabla\mathcal{L} + \frac{1}{2}\nabla(K\nabla\mathcal{L})\nabla\mathcal{L} + \frac{1}{2}\nabla^2\mathcal{L}K\nabla\mathcal{L} \\ &= \nabla\mathcal{L} + \nabla^2\mathcal{L}K\nabla\mathcal{L} + \frac{1}{2}\Gamma\mathrm{diag}(\nabla\mathcal{L})\nabla\mathcal{L},\end{aligned} \tag{4.2}$$

where we use the property

$$\nabla(K\nabla\mathcal{L}) = \nabla^2\mathcal{L}K + \Gamma\mathrm{diag}(\nabla\mathcal{L}).$$

For notational convenience, we define the expanded null-space matrix $N := N(w) \in \mathcal{R}^{n_w \times n_d}$ corresponding to the subspace $\mathcal{S}_P(w)$ with structure,

$$N = \begin{bmatrix} N_x & \\ & \mathbb{I}_m \end{bmatrix}. \tag{4.3}$$

Here, the lower corner is the identity because the multipliers λ are free and $N_x = N_x(x)$ is the null-space matrix defined in Section 2.

We now prove that there exist parameters α, β for which if w^* is a KKT point of (3.13), then it is a KKT point of the NLP (1.1). We first need the following property.

PROPERTY 1. *At any KKT point w^* of the NLP (1.1) we have that*

$$\begin{aligned}&\nabla_x P(w^*) = \nabla_x\mathcal{L}(w^*) + 2\beta\, diag(\nabla_x\mathcal{L}(w^*))\nabla_x\mathcal{L}(w^*)\\ &\nabla_\lambda P(w^*) = 0_m.\end{aligned}$$

Proof: The result follows because $K_{\alpha,\beta}(w^*)\nabla_w\mathcal{L}(w^*) = 0_{n_w}$, since $4\beta X^*\nabla_x\mathcal{L}(w^*) = 0_n$ and $\nabla_\lambda\mathcal{L}(w^*) = h(x^*) = 0_m$. $\square$

THEOREM 4.1. *Assume x^* is a KKT point of the EDPF problem (3.13). In addition, assume that LICQ and SC (A2) hold at this point. There exists a scalar $\bar{\beta} > 0$ and, for each $\beta \in (0, \bar{\beta}]$, a scalar $\bar{\alpha}(\beta)$ such that for all $\alpha \geq \bar{\alpha}(\beta)$, x^* is also a KKT point for the NLP (1.1).*

Proof: To simplify notation we write $X = X^*$. From (3.14) we have that a KKT point for (3.13) satisfies $0 \leq x^* \perp \nabla_x P(x^*, \lambda^*) \geq 0$, which implies $\sqrt{X}\nabla_x P(x^*, \lambda^*) = 0_n$. We thus have

$$\begin{aligned}0_n &= \sqrt{X}\nabla_x P(w^*)\\ &= \sqrt{X}\nabla_x\mathcal{L}(w^*) + 4\beta\sqrt{X}\nabla_{x,x}\mathcal{L}(w^*)\sqrt{X}\sqrt{X}\nabla_x\mathcal{L}(w^*) + 2\beta\sqrt{X}\mathrm{diag}\left(\nabla_x\mathcal{L}(w^*)\right)\nabla_x\mathcal{L}(w^*)\\ 0_m &= \nabla_\lambda P(w^*) = h(x^*) + 4\beta\nabla_x h(x^*)\sqrt{X}\sqrt{X}\nabla_x\mathcal{L}(w^*).\end{aligned}$$

We need to show that these two conditions imply $h(x^*) = 0$ and $X\nabla_x\mathcal{L}(w^*) = 0_n$, the KKT conditions of the NLP. We write the above system as

$$\begin{bmatrix} \mathbb{I}_{n\times n} + 4\beta\sqrt{X}\nabla_{x,x}\mathcal{L}(w^*)\sqrt{X} + 2\beta\mathrm{diag}\left(\nabla_x\mathcal{L}(w^*)\right) & \alpha\sqrt{X}\nabla_x h(x^*)^T \\ 4\beta\nabla_x h(x^*)\sqrt{X} & \mathbb{I}_{m\times m}\end{bmatrix} \cdot$$

$$\cdot \begin{bmatrix} \sqrt{X}\nabla_x\mathcal{L}(w^*) \\ h(x^*) \end{bmatrix} = \begin{bmatrix} 0_n \\ 0_m \end{bmatrix}. \tag{4.4}$$

We define $H = \mathbb{I}_{n\times n} + 4\beta\sqrt{X}\nabla_{x,x}\mathcal{L}(w^*)\sqrt{X} + 2\beta\text{diag}\,(\nabla_x\mathcal{L}(w^*))$. The result follows if the matrix on the left-hand side is nonsingular. We let $\bar{\beta} > 0$ be such that for all $\beta \in (0, \bar{\beta}]$ the matrix in the upper left corner is positive definite. In this case, we can apply Schur elimination to obtain

$$\sqrt{X}\nabla_x\mathcal{L}(w^*) = -\alpha H^{-1}\sqrt{X}\nabla_x h(x^*)^T h(x^*), \tag{4.5}$$

and

$$\begin{aligned} 0_m &= h(x^*) - 4\alpha\,\beta\nabla_x h(x^*)\sqrt{X}H^{-1}\sqrt{X}\nabla_x h(x^*)^T h(x^*) \\ &= \left[4\alpha\beta\nabla_x h(x^*)\sqrt{X}H^{-1}\sqrt{X}\nabla_x h(x^*)^T - \mathbb{I}_{m\times m}\right] h(x^*). \end{aligned} \tag{4.6}$$

For all $\beta \in (0, \bar{\beta}]$ we can choose $\bar{\alpha}(\beta)$ such that for all $\alpha \geq \bar{\alpha}(\beta)$ we have that the matrix above on the left-hand side is positive definite. This implies $h(x^*) = 0$ and $X\nabla_x\mathcal{L}(w^*) = 0_n$. Furthermore, from Property 1 we have $\nabla_x P(w^*)_{(i)} = \nabla_x\mathcal{L}(w^*)_{(i)} + 2\beta(\nabla_x\mathcal{L}(w^*)_{(i)})^2$. We thus have $\nabla_x P(w^*)_{(i)} = 0$ for $x_{(i)} > 0$, which implies $\nabla_x\mathcal{L}(w^*)_{(i)} = 0$. For $x_{(i)} > 0$ we have $\nabla_x P(w^*)_{(i)} > 0$, which implies $\nabla_x\mathcal{L}(w^*)_{(i)} > 0$ for β sufficiently small. $\square$

Condition (4.4) indicates that sufficiently small β exists to make the matrix in the upper left corner positive definite. Condition (4.6) indicates that, for this β, a sufficiently large α exists to make the matrix on the left-hand side positive definite. Consequently, we can see that the critical points of the EDPF coincide with critical points of the NLP as $\alpha \to +\infty$ and $\beta \to 0$. Our analysis extends the results of DiPillo and Grippo [13] for the case in which the exact penalty includes inequalities. Our proof follows along the lines of Proposition 4.15 provided by Bertsekas [4] for equality-constrained problems. We note that the upper left matrix in the inequality case becomes $\mathbb{I}_{n\times n} + 4\beta\sqrt{X}\nabla_{x,x}\mathcal{L}(w^*)\sqrt{X} + 2\beta\text{diag}\,(\nabla_x\mathcal{L}(w^*))$, as opposed to $\mathbb{I}_{n\times n} + \beta\nabla_{x,x}\mathcal{L}(w^*)$ for the equality-constrained case. For the equality-constrained case Bertsekas observed that β can be any non-negative value if $\nabla_{x,x}\mathcal{L}(w^*)$ is positive definite. Interestingly, this situation also applies in our case because the terms $\sqrt{X}$ and $2\beta\text{diag}\,(\nabla_x\mathcal{L}(w^*))$ are positive. As opposed to the equality-constrained case, however, β needs to be chosen sufficiently small to enforce $\nabla_x\mathcal{L}(w^*)_{(i)} > 0$ for $x_{(i)} = 0$, which restricts β.

The above result also implies that the solution of the EDPF problem identifies the optimal active set of the NLP. Consequently, $\mathcal{A}(x^*) = \mathcal{A}_P(w^*)$ because λ^* are free. This also implies that the upper n_d^* rows of the null-space matrix $N(w^*)$ defined for $\mathcal{S}_P(w^*)$ correspond to the null-space matrix $N_x(x^*)$ defined for $\mathcal{S}(x^*)$. We also emphasize that given a solution w^* for the NLP and corresponding null-space matrix $N_x(x^*)$, we can construct an expanded null-space matrix $N(w^*)$ (4.3). This observation will be needed in the following analysis.

We now connect the SSOC for the NLP and for the EDPF problem. If $f(\cdot)$ and $h(\cdot)$ are three times continuously differentiable (A1), the Hessian of the EDPF exists. The Hessian of the EDPF involves a tensor because of the introduction of the penalty term on the gradient of the Lagrangian. To enable notation in the Hessian derivation, we use the fact that

$$\nabla^2 P \cdot u = \nabla(\nabla P^T \cdot u) \tag{4.7}$$

for a constant vector $u \in \mathcal{R}^{n_w}$. We have,

$$\nabla P^T \cdot u = \nabla\mathcal{L}^T \cdot u + \nabla\mathcal{L}^T K \nabla^2\mathcal{L} \cdot u + \frac{1}{2}\nabla\mathcal{L}^T \Gamma \text{diag}(\nabla\mathcal{L}) \cdot u, \tag{4.8}$$

and

$$\nabla^2 P \cdot u = \nabla^2\mathcal{L} \cdot u + \nabla(\nabla\mathcal{L}^T K \nabla^2\mathcal{L} \cdot u) + \frac{1}{2}\nabla(\nabla\mathcal{L}^T \Gamma \text{diag}(\nabla\mathcal{L}) \cdot u). \tag{4.9}$$

Expanding terms, we obtain

$$\begin{aligned}\nabla(\nabla\mathcal{L}^T K\nabla^2\mathcal{L}\cdot u) &= \nabla(K\nabla\mathcal{L})\nabla^2\mathcal{L}\cdot u + \nabla(\nabla^2\mathcal{L}\cdot u)K\nabla\mathcal{L}\\ &= \nabla^2\mathcal{L}K\nabla^2\mathcal{L}\cdot u + \Gamma\mathrm{diag}(\nabla\mathcal{L})\nabla^2\mathcal{L}\cdot u + \nabla(\nabla^2\mathcal{L}\cdot u)K\nabla\mathcal{L},\end{aligned}\tag{4.10}$$

and

$$\frac{1}{2}\nabla(\nabla\mathcal{L}^T\Gamma\mathrm{diag}(\nabla\mathcal{L})\cdot u) = \nabla^2\mathcal{L}\mathrm{diag}(\nabla\mathcal{L})\Gamma\cdot u.\tag{4.11}$$

Merging terms,

$$\begin{aligned}\nabla^2 P\cdot u = \nabla^2\mathcal{L}\cdot u + \nabla^2\mathcal{L}K\nabla^2\mathcal{L}\cdot u\\ + \nabla^2\mathcal{L}\mathrm{diag}(\nabla\mathcal{L})\Gamma\cdot u + \Gamma\mathrm{diag}(\nabla\mathcal{L})\nabla^2\mathcal{L}\cdot u + \nabla(\nabla^2\mathcal{L}\cdot u)K\nabla\mathcal{L}.\end{aligned}\tag{4.12}$$

From this expression we derive the following property of the Hessian and reduced Hessian matrices for the EDPF.

PROPERTY 2. *At any KKT point w^* of the NLP* (1.1) *with corresponding expanded null-space matrix $N := N(w^*)$ we have that*

$$\text{(i)}\quad \nabla^2_w P(w^*) = \nabla^2\mathcal{L} + \nabla^2\mathcal{L}K\nabla^2\mathcal{L} + \nabla^2\mathcal{L}\,diag(\nabla\mathcal{L})\Gamma + \Gamma\,diag(\nabla\mathcal{L})\nabla^2\mathcal{L}\tag{4.13a}$$

$$\text{(ii)}\quad N^T\nabla^2_w P(w^*)N = N^T\nabla^2\mathcal{L}N + N^T\nabla^2\mathcal{L}K\nabla^2\mathcal{L}N.\tag{4.13b}$$

Proof: (i) follows because $K_{\lambda,\beta}(w^*)\nabla_w\mathcal{L}(w^*) = 0_{n_w}$. (ii) follows because $N^T\Gamma\mathrm{diag}(\nabla\mathcal{L}) = 0_{n_d\times n}$ and $\Gamma\mathrm{diag}(\nabla\mathcal{L})N = 0_{n\times n_d}$. □

We highlight that the high-order term $\nabla(\nabla^2_w\mathcal{L}(w^*)\cdot u)K_{\alpha,\beta}(w^*)\nabla_w\mathcal{L}(w^*)$ vanishes at a KKT point w^* of the NLP, thus suggesting (4.13a) as a natural Hessian approximation which we define as

$$Q(w) = \nabla^2\mathcal{L}(w) + \nabla^2\mathcal{L}(w)K(w)\nabla^2(w)\mathcal{L} + \nabla^2\mathcal{L}(w)\mathrm{diag}(\nabla\mathcal{L})\Gamma + \Gamma\mathrm{diag}(\nabla\mathcal{L})\nabla^2\mathcal{L}(w).\tag{4.14}$$

The corresponding reduced Hessian is $N^TQ(w)N$. In Section 6 we explain how these properties can be exploited to enable matrix-free implementations.

Another crucial implication resulting from (4.12) and Property 2 is that the Hessian approximation $Q(w)$ satisfies the Dennis-Moré condition:

$$\begin{aligned}(Q(w) - \nabla^2 P(w))\cdot u &= \nabla(\nabla^2\mathcal{L}(w)\cdot u)K(w)\nabla\mathcal{L}(w)\\ &\overset{w\to w*}{=} 0_{n_w}.\end{aligned}\tag{4.15}$$

This follows from the fact that the third derivatives of $\mathcal{L}$ are uniformly bounded, whereas $K(w)\nabla\mathcal{L}(w)$ is continuously differentiable, and thus Lipschitz and satisfies $K_{\lambda,\beta}(w^*)\nabla_w\mathcal{L}(w^*) = 0_{n_w}$. We thus obtain that

$$\begin{aligned}\nabla(\nabla^2\mathcal{L}(w)\cdot u)K(w)\nabla\mathcal{L}(w) &= O(u)O(||w-w^*||)\\ &\overset{w\to w*}{=} 0_{n_w}.\end{aligned}\tag{4.16}$$

The result is stronger than the classical Dennis-Moré condition because convergence is independent of the direction u. This result will become important in establishing superlinear convergence results in Section 5. We now prove that there exist α,β such that the reduced

Hessian matrix of the EDPF is positive definite at w^* if SSOC are satisfied for the NLP at this point. We also show that there exist α, β for which positive definiteness of the reduced Hessian at a KKT point w^* of the EDPF problem (3.13) implies that this is a strict local minimum of the NLP. In this sense and under these condition, EDPF is exact, that is it has local minima where (1.1) does and it will not introduce any additional local minima.

THEOREM 4.2. *If w^* is a strict local minimum of the NLP* (1.1) *satisfying the SSOC conditions* (2.9) *(A3), then for every $\beta > 0$ there exists $\bar{\alpha}(\beta) > 0$ such that for all $\alpha \geq \bar{\alpha}(\beta)$, w^* is a strict local minimum of the EDPF problem* (3.13)*, and the reduced Hessian $N(w^*)^T \nabla^2 P_{\alpha,\beta}(w^*) N(w^*)$ is positive definite. Furthermore, assume w^* satisfies the KKT conditions for NLP* (1.1) *but not the SSOC* (2.9)*. Then, there exists $\bar{\beta} > 0$ such that for each $\beta \in (0, \bar{\beta}]$ and $\alpha > 0$, w^* is not a strict local minimum of* (3.13).

Proof: We define $H = \nabla_x^2 \mathcal{L}(w^*)$ and $A = \nabla_x h(x^*)$. We construct the projected Hessian $\nu^T N^T \nabla^2 P N \nu$ using $N = N(w^*)$ constructed from w^* and with $\nu^T = [\nu_x^T \; \nu_\lambda^T]$, where $\nu \in \mathcal{R}^{d^*}$ and $\nu_x \in \mathcal{R}^{n_d^*}$. Because w^* satisfies the SSOC (2.9), we have that $\nu_x^T N_x^T H N_x \nu_x > 0$ for all $N_x \nu_x \neq 0$ and $A N_x \nu_x = 0_m$. Using Property 2, we have that $\nu^T N^T \nabla^2 P N \nu$ has the following simplified form:

$$\begin{aligned}
&\nu^T N^T \nabla^2 P N \nu \\
&= \begin{bmatrix} \nu_x^T N_x^T & \nu_\lambda^T \end{bmatrix} \begin{bmatrix} H & A^T \\ A & \end{bmatrix} \begin{bmatrix} N_x \nu_x \\ \nu_\lambda \end{bmatrix} \\
&\quad + \begin{bmatrix} \nu_x^T N_x^T & \nu_\lambda^T \end{bmatrix} \begin{bmatrix} H & A^T \\ A & \end{bmatrix} \begin{bmatrix} 4\beta X & 0 \\ 0 & \alpha \mathbb{I}_m \end{bmatrix} \begin{bmatrix} H & A^T \\ A & \end{bmatrix} \begin{bmatrix} N_x \nu_x \\ \nu_\lambda \end{bmatrix}.
\end{aligned} \tag{4.17}$$

Expanding terms, we have

$$\begin{aligned}
&\nu^T N^T \nabla^2 P N \nu \\
&= \left[\nu_x^T N_x^T H N_x \nu_x + 2\nu_\lambda^T A N_x \nu_x + 4\beta (H N_x \nu_x + A^T \nu_\lambda)^T X (H N_x \nu_x + A^T \nu_\lambda)\right] + \alpha \nu_x^T N_x^T A^T A N_x \nu_x.
\end{aligned}$$

If the SSOC hold, the first term in brackets is positive for all $N_x \nu_x \neq 0$, and we have $\nu_x^T N_x^T A^T A N_x \nu_x = 0$. Under this condition and by Debreu's lemma [17] we have that there exists $\bar{\alpha}(\beta)$ for all $\beta \geq 0$ such that $N^T \nabla^2 P(w^*) N$ is positive definite. To establish the second result, let $N_x \nu_x$ satisfy $A N_x \nu_x = 0_m$ and $\nu_x^T N_x^T H N_x \nu_x < 0$. Setting $\nu_\lambda = 0$, we have

$$\nu^T N^T \nabla^2 P N \nu = \nu_x^T N_x^T H N_x \nu_x + 4\beta \nu_x^T N_x^T X H N_x < 0$$

for any $\beta < \frac{1}{4} \nu_x^T N_x^T H N_x \nu_x / \nu_x^T N_x^T X H N_x \nu_x$, which implies that w^* is not a local minimum. The proof is complete. □

The above result implies that a point w^* satisfying the SSOC conditions for the NLP (1.1) satisfies the SSOC conditions (3.15) for the EDPF problem (3.13) if α, β are chosen sufficiently large and sufficiently small, respectively. Our proof follows along the lines of Proposition 4.16 in [4], but this is extended to deal with inequality constraints. An important finding resulting from our analysis of the structure of the reduced Hessian (4.13b) is that the introduction of inequalities does not alter the positive curvature of the EDPF. Positive definiteness of the reduced Hessian is also important because this enables the use of PCG to compute truncated Newton steps.

5. Trust-Region Newton.

The results of the previous section indicate that the curvature of the EDPF is benign close to the solution. From the structure of the Hessian (4.12), however, it is clear that negative curvature is likely to occur along the search. Therefore, implementations based on line-search methods would be inefficient. We propose to apply a trust-region Newton framework to the EDPF problem (3.13) following the developments of Lin and Moré [25]. In our analysis, we will assume that α, β are chosen such that conditions and results of Theorems 4.1 and 4.2 hold. Consequently, we simplify the notation as $P(\cdot) \leftarrow P_{\alpha,\beta}(\cdot)$. Lin and Moré's convergence analysis relies on the exact Hessian of the objective function. We have extended their results to the particular case of the approximate Hessian (4.13a) and demonstrate that this approximation does not destroy convergence properties.

We write (3.13) as

$$\min_{w} \quad P(w) \quad \text{s.t.} \quad w \in \Omega, \tag{5.1}$$

where $P(\cdot) := P_{\alpha,\beta}(\cdot)$ and $\Omega := \{w \,|\, w \geq l\} \subseteq \mathcal{R}^{n_w}$, where $l_{(i)} = 0,\ i = 1, ..., n$ and $l_{(i)} = -\infty,\ i = n+1, ...n_w$. We have that $P(\cdot)$ is twice continuously differentiable if $f(\cdot)$ and $h(\cdot)$ in (1.1) are three times continuously differentiable. We also define the projection operator $\text{Proj} : \mathcal{R}^{n_w} \to \Omega$ and the projected gradient as $g_{\text{Proj}}(w)$. For formal definitions see Section 2 in [25].

Lin and Moré use the concept of exposed faces to characterize the active set. We have that the face exposed by the gradient $-\nabla_w P(w^*)$ is given by

$$E[-\nabla_w P(w^*)] = \left\{ w \in \Omega \,|\, w_{(i)} = 0 \quad \text{if} \quad \nabla_w P(w^*)_{(i)} > 0 \right\}. \tag{5.2}$$

Burke and Moré [8] showed that w^* is a stationary point of (5.1) if and only if $w^* \in E[-\nabla_w P(w^*)]$. We also have that, if SC holds, $w \in E[-\nabla_w P(w^*)] \iff \mathcal{A}_P(w^*) \subset \mathcal{A}_P(w)$. As shown in Theorem 4.2, there exist α, β such that a point w^* satisfying SSOC for the original NLP also satisfies the SSOC for (5.1). The SSOC for (5.1) can also be written as

$$\nu^T \nabla^2 P(w^*) \nu \geq \kappa \|\nu\|^2, \quad \nu \in \mathcal{S}_P(w^*),\ \kappa > 0. \tag{5.3}$$

In a trust-region framework, we have at each iteration $w^k \in \Omega$ a radius Δ^k and a quadratic *model* $\Psi : \mathcal{R}^{n_w} \to \mathcal{R}$ of the actual reduction $P(w^k + s) - P(w^k)$:

$$\Psi^k(s) = {g^k}^T s + \frac{1}{2} s^T Q^k s. \tag{5.4}$$

Here, Q^k is approximation of the exact penalty Hessian $\nabla^2_w P(w^k)$ given in (4.13a), $g^k := \nabla_w P(w^k)$ is the gradient, and g^k_{Proj} is the projected gradient. We also define $q^k(w) := \Psi^k(w - w^k)$.

Given a step s^k such that $w^k + s^k \in \Omega$ and $\Psi^k(s^k) < 0$, the trust-region bound is updated according to the reduction ratio

$$\rho^k = \frac{P(w^k + s^k) - P(w^k)}{\Psi^k(s^k)}. \tag{5.5}$$

Since the step is chosen such that $\rho^k > 0$, $\Psi^k(s^k)$ indicates a reduction of the actual function. The iterate is updated using ρ^k according to

$$w^{k+1} = \left\{ \begin{array}{ll} w^k + s^k & \text{if} \quad \rho^k > \eta_0 \\ w^k & \text{if} \quad \rho^k \leq \eta_0 \end{array} \right., \tag{5.6}$$

for $\eta_0 > 0$. The trust-region bound is updated according to the following rule:

$$\Delta^{k+1} \in \left\{ \begin{array}{ll} \left[\sigma_1 \cdot \min\{\|s^k\|, \Delta^k\}, \sigma_2 \Delta^k\right] & \text{if} \quad \rho^k \leq \eta_1 \\ \left[\sigma_1 \Delta^k, \sigma_3 \Delta^k\right] & \text{if} \quad \rho^k \in (\eta_1, \eta_2), \\ \left[\Delta^k, \sigma_3 \Delta^k\right] & \text{if} \quad \rho^k \geq \eta_2 \end{array} \right. \tag{5.7}$$

where $\eta_1 < \eta_2 < 1$ and $\sigma_1 < \sigma_2 < 1 < \sigma_3$.

To obtain the search step s^k, we first compute the Cauchy step s_c^k by performing projected line searches on the step size $\kappa_c \in [0,1]$,

$$s_c^k = \text{Proj}(w^k - \kappa_c \nabla_w P(w^k)) - w^k, \tag{5.8}$$

until the following conditions are satisfied,

$$\Psi^k(s_c^k) \le \mu_0 \nabla_w P(w^k)^T s_c^k, \qquad \|s_c^k\| \le \mu_1 \Delta^k, \tag{5.9}$$

with $\mu_0, \mu_1 > 0$. The projected search also gives the active set guess $\mathcal{A}_P^k := \mathcal{A}_P(w_c^k)$ with $w_c^k = w^k + s_c^k$. We also define the inactive set $\mathcal{I}^k = \mathcal{I}(w_c^k)$. An important property is that if any sequence w^k converges to a stationary point w^* and w^k lands on $E[-\nabla_w P(w^*)]$, then the Cauchy point $x_k + s_c^k$ remains on $E[-\nabla_w P(w^*)]$ (see Theorem 3.2 in [25]).

We now seek a step s^k that improves the Cauchy step in the sense that

$$\Psi^k(s^k) \le \mu_0 \Psi^k(s_c^k), \quad w^k + s^k \in \Omega. \tag{5.10}$$

Given this step, we check the reduction ratio (5.5), accept or reject the step (5.6), and update the trust-region radius (5.7). This gives rise to a basic trust-region method. Theorem 3.3 in [25] states that if the steps s^k generated by the trust-region method satisfy

$$\mathcal{A}_P(w_c^k) \subset \mathcal{A}_P(w^k + s^k) \tag{5.11}$$

for $k \ge 0$, then if $\{w^k\}$ converges to w^*, there is an index k_0 such that for $k \ge k_0$ we have $w^k \in E[-\nabla_w P(w^*)]$ and $w^k + s^k \in E[-\nabla_w P(w^*)]$. This property is essential to guarantee local convergence.

We refine the Cauchy point w_c^k while satisfying conditions (5.10) and (5.11) in order to preserve global convergence. Specifically, Lin and Moré suggest the following procedure. At each *major* iteration k we compute *minor* iterates $w_0^k, ..., w_{\ell+1}^k$ with $w_0^k := w_c^k$ and $w^k := w_{\ell+1}^k$. For each minor iterate $j = 0, ..., \ell+1$ we require that

$$w_j^k \in \Omega, \quad \mathcal{A}_P(w_c^k) \subset \mathcal{A}_P(w_j^k), \quad \|w_j^k - w^k\| \le \mu_1 \Delta^k, \tag{5.12}$$

and

$$q^k(w_{j+1}^k) \le q^k(w_j^k) + \mu_0 \cdot \min\left\{\nabla_w q^k(w^k)^T (w_{j+1}^k - w_j^k)\ ,\ 0\right\}. \tag{5.13}$$

To satisfy these conditions, we require a descent direction for $q^k(\cdot)$. To this end we use the trust-region quadratic program (QP),

$$\min_{s^k} {g^k}^T s^k + \frac{1}{2} {s^k}^T Q^k s^k \tag{5.14a}$$

$$\text{s.t. } s_{(i)}^k = 0,\ i \in \mathcal{A}_P(w_j^k) \tag{5.14b}$$

$$\|D^k s^k\| \le \Delta^k, \tag{5.14c}$$

where D^k is a preconditioning matrix for Q^k. Descent directions can be obtained by using Steihaug's PCG approach [36]. At each minor iterate j, we define the null space matrix $N_j^k \in \mathcal{R}^{n \times d_j^k}$ consistent with $\mathcal{A}_P(w_j^k)$ and the reduced-space step $s_d^k \in \mathcal{R}^{d_j^k}$ so that $s_j^k = N_j^k s_d^k$. The QP takes the form

$$\min_{s_d^k} {g_d^k}^T s_d^k + \frac{1}{2} {s_d^k}^T Q_d^k s_d^k \tag{5.15a}$$

$$\text{s.t. } \|D^k N_j^k s_d^k\| \le \Delta^k. \tag{5.15b}$$

Here $Q_d^k = {N_j^k}^T Q^k N_j^k$, $g_d^k = {N_j^k}^T g^k$ are the reduced approximate Hessian and reduced gradient, respectively. The PCG search iterates $i > 0$ are terminated by generating step $s_j^k := N_j^k s_{d,i}^k$ if

C-i) a descent step is found,

C-ii) the step hits the trust-region radius, or

C-iii) negative curvature is detected.

A final step $s_{\ell+1}^k$ is considered successful from a global convergence point of view if it satisfies (5.10). This condition can be enforced by performing projected searches along step each step s_j^k,

$$s_j^k \leftarrow \mathrm{Proj}(w^k + \kappa s_j^k) - w^k, \tag{5.16}$$

where $\kappa \in [0, 1]$ is the step length, and stopping when (5.12) and (5.13) are satisfied. The procedure terminates with a final step $s^k = s_{\ell+1}^k = w_{\ell+1}^k - w^k$ satisfying (5.10).

To enable superlinear convergence, we require the step $s^k := w_{\ell+1}^k - w^k$ to satisfy

$$\left\|(N_m^k)^T \left[g^k + Q^k s^k\right]\right\| \leq \xi^k \|(N_m^k)^T g^k\|, \quad w^k + s^k \in \Omega. \tag{5.17}$$

To satisfy this condition simultaneously with (5.12)-(5.13), Lin and Moré proposed the following approach. Close to the solution, the trust-region constraint is inactive and directions of negative curvature are not encountered. Consequently, we require that at each minor iterate w_j^k, $j = 0, ..., \ell + 1$ is obtained by terminating the PCG search at the minimizer of $q^k(\cdot)$. This amounts to assume that $\xi^k = 0$. This gives $s^k = w_{j+1}^k - w^k$. We perform a line-search along this step to stay inside Ω, in order to satisfy (5.13), and such that $\mathcal{A}_P(w_{j+1}^k)$ has at least one more active variable than $\mathcal{A}_P(w_j^k)$ such that

$$\mathcal{A}_P(w_j^k) \subset \mathcal{A}_P(w_{j+1}^k) \tag{5.18}$$

This procedure implicitly satisfies (5.12). In the worst case, this procedure terminates with all variables active and for which condition $w^k + s^k \in \Omega$ is satisfied trivially.

The above procedure gives the major step s^k. We next compute the reduction (5.5). If the first condition in (5.6) is satisfied, we accept the step $w^{k+1} = w^k + s^k$; otherwise it is rejected. Finally, we update the trust-region radius Δ^{k+1} following (5.7). The trust-region Newton (TRN) algorithm is summarized below.

Trust-Region Newton Algorithm

Assume given algorithmic parameters $\eta_0, \eta_1, \eta_2, \sigma_1, \sigma_2, \mu_0, \mu_1$, and ℓ.

1. **Initialization**. Start with w^0 and Δ^0 at $k = 0$. DO for $k > 0$:
2. **Major Step Test**. If w^k is a stationary point, STOP.
3. **Cauchy Search**. Compute Cauchy step s_k^c and active set $\mathcal{A}_P(w_c^k)$ from (5.8), and perform line-search until conditions (5.9) are satisfied. Define step $w_c^k \leftarrow w^k + s_c^k$.
4. **Refinement Search**. Start at $w_0^k := w_c^k$. DO for $j = 0, ..., \ell + 1$:
   4.1. *PCG Step.* At $\mathcal{A}_P(w_j^k)$ apply Steihaug's PCG search on (5.15), and terminate with s_d^k if either (C-i),(C-ii), or (C-iii) holds.
   4.2. *Minor Step Test.* Set $w_{j+1}^k \leftarrow w_j^k + N_j^k s_d^k$. If improvement over Cauchy (5.9) and (5.11) are satisfied, STOP refinement search, set $s^k = w_{j+1}^k - w^k$, and GO TO 5.
   4.3. *Update Minor Step.* Cut step to satisfy (5.12), (5.13), and (5.18), update w_{j+1}^k, set $j \leftarrow j + 1$, and RETURN TO 4.1.

5. **Update Step**. Compute reduction ratio (5.5) update step w^{k+1} according to (5.6), and update the trust-region radius Δ^{k+1} according to (5.7). Set $k \leftarrow k+1$, and RETURN TO 2.

We now establish convergence results for the trust-region Newton algorithm. We start by establishing global convergence. The following result is an adaptation of the result of Burke, Moré, and Toraldo [7] (see also Theorem 2.1 in [25]) to the EDPF problem.

Theorem 5.1. *Let sequence of approximate Hessians $\{Q^k\}$ of the quadratic model* (5.4) *be uniformly bounded. i) If w^* is a limit point of the sequence $\{w^k\}$ generated by the trust-region Newton algorithm then there is a subsequence $\{w^{k_i}\}$ of successful steps which converges to w^* with:*

$$\lim_{i\to\infty} \|g_{Proj}(w_c^{k_i})\| = 0. \tag{5.19}$$

In addition, $\{w_c^{k_i}\}$ also converges to w^*, and thus w^* is a stationary point for problem (5.1). *Moreover, ii) if the EDPF parameters α, β satisfy conditions of Theorems 4.1 and 4.2 then w^* is also stationary for the NLP* (1.1).

Proof: Result i) follows from the proof of Theorem 5.5 in [7]. Result ii) follows from Theorems 4.1 and 4.2 since we have that for any $\beta, \alpha \geq \bar{\alpha}(\beta)$ a stationary point w^* of the EDPF problem is also stationary for the NLP. □

The above result implies that every limit point of the sequence $\{w^k\}$ is a stationary point of problem (5.1).

To establish rate of convergence, we first show that if the approximate Hessian $Q(\cdot)$ is used, the limit point of the trust-region radius is bounded away from zero. Our analysis follows that of Lin and Moré [25].

From Theorems 4.1 and 4.2 we have that there exist α, β such that $Q(w)$ is bounded if $\nabla_{w,w}\mathcal{L}(w)$ is bounded. The point w^* satisfies the SSOC of the EDPF problem (5.3) if α, β satisfy conditions of Theorems 4.1 and 4.2. In this case we also have that the limit point w^* of the trust-region Newton method is a strict local minimum of the original NLP.

To establish rate of convergence we first show that if the approximate Hessian $Q(w)$ is used, the limit point of the trust-region radius is bounded away from zero.

Theorem 5.2. *Let $\{w^k\}$ be the sequence generated by the trust-region Newton method. Assume that the EDPF parameters α, β satisfy conditions of Theorems 4.1 and 4.2. Assume that $\{w^k\}$ converges to a limit point w^* that satisfies the SSOC* (5.3). *If the minor iterates satisfy* (5.12) *and* (5.13), *then there is an index k_0 such that all steps s^k with $k \geq k_0$ are successful and the trust-region bound Δ^k is bounded away from zero.*

Proof: The proof follows along the lines of proof of Theorem 5.3 in [25]. We extend this proof by accounting for the Hessian approximation error $\nabla^2 P - Q^k$. We need to prove that $\rho^k \to 1$ so that the trust-region update rules eventually accept all steps with Δ^k bounded away from zero. We have that

$$\rho^k - 1 = \frac{P(w^k+s^k) - P(w^k) - \Psi^k(s^k)}{\Psi^k(s^k)} \tag{5.20}$$

and

$$P(w^k+s^k) = P(w^k) + \nabla P(w^k)s^k + \frac{1}{2}{s^k}^T \nabla^2 P(w^k+\theta s^k)s^k, \quad \theta \in (0,1), \tag{5.21}$$

so that

$$\begin{aligned}
&P(w^k + s) - P(w^k) - \Psi^k(s^k) \\
&= \frac{1}{2}s^T\nabla^2 P(w^k+\theta s^k)s^k - \frac{1}{2}{s^k}^T Q(w^k)s^k \\
&= \frac{1}{2}{s^k}^T\nabla^2 P(w^k+\theta s^k)s^k - \frac{1}{2}{s^k}^T\nabla^2 P(w^k)s^k + \frac{1}{2}{s^k}^T\nabla^2 P(w^k)s^k - \frac{1}{2}{s^k}^T Q(w^k)s^k. \quad (5.22)
\end{aligned}$$

Bounding, we have

$$|P(w^k+s) - P(w^k) - \Psi^k(s^k)| = (\sigma_1^k + \sigma_2^k)\|s^k\|^2, \tag{5.23}$$

where

$$\sigma_1^k = \sup_{\theta\in(0,1)} \left\{\|\nabla^2 P(w^k+\theta s^k) - \nabla^2 P(w^k)\|\right\} \tag{5.24}$$

follows from Taylor's theorem and $\sigma_2^k = \|\nabla^2 P(w^k) - Q(w^k)\|$. Lemma 5.2 and Theorem 5.3 in [25] show that there exists κ_0 such that $|\rho^k - 1| \le (\sigma_1^k + \sigma_2^k)/\kappa^0$, so that the result is obtained if $\{\sigma_1^k\}, \{\sigma_2^k\}$ converge to zero. The sequence $\{\sigma_1^k\}$ converges to zero if $\{s^k\}$ converges to zero because $\{w^k\}$ converges to w^*. Proof of Theorem 5.3 shows that the sequence $\{s^k\}$ converges to zero. Consequently, the sequence $\{\sigma_1^k\}$ converges to zero. As for σ_2^k, we have that for α, β satisfying conditions of Theorems 4.1 and 4.2 the limit point w^* satisfies the KKT conditions of the NLP; and by Property 2 and condition (4.15) we have that $Q(w^k)$ converges to $\nabla^2 P(w^k)$ because $\{w^k\}$ converges to $\{w^*\}$. Consequently, $\{\sigma_2^k\}$ converges to zero. The proof is complete. □

We highlight that the convergence result of the Hessian approximation error σ_2^k is stronger than that obtained in a classical quasi-Newton setting because convergence is independent of the step s^k. We also emphasize that α, β need to satisfy conditions of Theorems 4.1 and 4.2 in order to guarantee convergence of $\{\sigma_2^k\}$ because this can be guaranteed only at a point w^* satisfying the KKT conditions of the original NLP.

The above result implies that there exists a bound $\mu^* < \mu_1$ such that $\|s^k\| \le \mu^*\Delta^k$. To establish superlinear convergence results, we also require that $\mathcal{A}_P(w^k) = \mathcal{A}_P(w^*)$ for k sufficiently large. We can now state the superlinear convergence result.

THEOREM 5.3. *Let $\{w^k\}$ be the sequence generated by the trust-region Newton method. Assume that the EDPF parameters α, β satisfy conditions of Theorems 4.1 and 4.2. Assume that $\{w^k\}$ converges to a solution w^* that satisfies the SSOC of the penalty problem* (5.3). *If the step satisfies $\|s^k\| \le \mu^*\Delta^k$, then the sequence $\{w^k\}$ converges Q-superlinearly to w^*.*

Proof: We follow the steps of Theorem 5.4 in [25]. The authors first showed that $\mathcal{A}_P(w^k) = \mathcal{A}_P(w^*)$ for k sufficiently large. We need the following estimate:

$$\begin{aligned}
&\|\mathbb{N}^k\nabla P(w^{k+1})\| \\
&\le \left\|\mathbb{N}^k\left[\nabla P(w^{k+1}) - \nabla P(w^k) - \nabla^2 P(w^k)s^k\right]\right\| + \left\|\mathbb{N}^k\left[\nabla^2 P(w^k)s^k - Q(w^k)s^k\right]\right\| \\
&\quad + \left\|\mathbb{N}^k\left[\nabla P(w^k) + Q(w^k)s^k\right]\right\| \\
&\le (\epsilon_1^k + \epsilon_2^k)\|s^k\|.
\end{aligned}$$

Here, $\mathbb{N}^k = N^k{N^k}^T$. The first bound on the right-hand side arises from Taylor's theorem, the second bound follows from Property 2 and the Dennis-Moré condition (4.15), and the third term is bounded above by zero because of the convergence condition (5.17) with $\xi^k = 0$. We also have that $\{\epsilon_1^k\}$ converges to zero. From Theorems 4.1 and 4.2 we have that the KKT conditions of the original NLP hold at w^* such that $\{\epsilon_2^k\}$ converges to zero. Theorem 5.3 in [25] shows that there exists $\nu_0 > 0$ such that $\|s^k\| \leq \nu_0 \|\mathbb{N}^* \nabla P(w^k)\|$ with $\mathbb{N}^* = N^* N^{*T}$. With this, we obtain

$$\frac{\|\mathbb{N}^k \nabla P(w^{k+1})\|}{\|\mathbb{N}^* \nabla P(w^k)\|} \leq (\epsilon_1^k + \epsilon_2^k)\nu_0 \tag{5.25}$$

and

$$\lim_{k\to\infty} \frac{\|\mathbb{N}^k \nabla P(w^{k+1})\|}{\|\mathbb{N}^* \nabla P(w^k)\|} \leq 0. \tag{5.26}$$

Lin and Moré show that there exists $\nu_1 > 0$ such that

$$\|\mathbb{N}^k P(w^{k+1})\| \geq (\nu_1 - \epsilon_k^1)\|w^{k+1} - w^*\| \tag{5.27}$$

and

$$\|\mathbb{N}^* P(w^k)\| \leq \nu_2 \|w^k - w^*\| + \epsilon_1^k \|w^k - w^*\|, \quad \nu_2 = \|\mathbb{N}^* \nabla^2 P(w^*) \mathbb{N}^*\|. \tag{5.28}$$

These estimates use w^* as fixed point and only rely on $\nabla^2 P(w^*)$ which, by Property 2 is equal to $Q(w^*)$. Consequently,

$$(\nu_1 - \epsilon_k^1)\|w^{k+1} - w^*\| \leq \nu_2 \|w^k - w^*\| + \epsilon_1^k \|w^k - w^*\|. \tag{5.29}$$

Since $\{\epsilon_1^k\}$ converges to zero we have

$$\lim_{k\to\infty} \frac{\|w^{k+1} - w^*\|}{\|w^k - w^*\|} \leq 0.$$

The proof is complete. □

If α, β are appropriately chosen, then superlinear convergence can be achieved. This implies the surprising result that ignoring the third-order term (last term in (4.12)) does not destroy superlinear convergence. For the equality-constrained case the result is perhaps less surprising because the gradient of the Lagrangian converges to zero.

6. Summary of Algorithm and Scalability.

In this section we provide the whole set of algorithmic components in order to highlight the main computational steps and discuss scalability issues.

6.1. EDPF-TRN Algorithm.

Assume given user-provided functions to compute $f(x)$, $c(x)$,$\nabla_x f(x)$,$\nabla_x h(x) \cdot u$, $\nabla_x h(x)^T \cdot u$ and $\nabla_{x,x} \mathcal{L}(x, \lambda) \cdot u$.

EDPF-TRN Algorithm

```
A) Start at primal-dual pair $w = [x, \lambda]$. Set $\alpha, \beta$ and tolerance $\tau_P$.
B) Call TRN algorithm at $w^0 \leftarrow w$ and $\Delta^0$.
    1. **Initialization.** Start at $w^0$ and $\Delta^0$.
    2. **Major Step Test.**
        2.1. Compute EDPF $P^k := P_{\alpha,\beta}(w^k)$ from (3.11).
        2.2. Compute gradient of EDPF $g^k := \nabla_w P_{\alpha,\beta}(w^k)$ from (4.2) and projected gradient
             $g_{\text{Proj}}(w^k)$.
        2.3. If $w^k$ is a stationary point set $w^*(\alpha, \beta) \leftarrow w^k$, and STOP.
    3. **Cauchy Search.** Compute Cauchy step $s_k^c$ from (5.8). At each line-search step
       compute approximate Hessian-vector product $Q(w^k) \cdot s_c^k$ using (4.14) and $\Psi^k(s_c^k)$
       from (5.4), and STOP when (5.11) is satisfied. Update $w_c^k \leftarrow w^k + s_c^k$.
```

```
4. **Refinement Search.**. Start at $w_0^k := w_c^k$. DO for $j = 0, ..., \ell + 1$:
   4.1. *PCG Step.* At $\mathcal{A}_P(w_j^k)$ construct $N \leftarrow N_j^k$, compute $g_d^k = N^T g^k$, and call
        Steihaug's PCG search for (5.15) with tolerance $\epsilon$:
      4.1.1. Starting at $i = 0$ with $r_i = g_d^k$, $d_i = -r_i$, $z_i = 0$, and apply preconditioner
             $y_i = D^{-1} \cdot r_i$. DO for $i > 0$:
      4.1.2. Form $\bar{d}_i = N^T d_i$ and $Q(w^k) \cdot \bar{d}_i$ using (4.14).
      4.1.3. If (C-ii) holds: $\bar{d}_i^T (Q(w^k) \cdot d_i) \leq 0$ find $\tau > 0$ such that $\|s_d^k\| = \Delta^k$ with
             $s_d^k = z_i + \tau d_i$, and STOP.
      4.1.4. Set $\gamma_i \leftarrow r_i^T y_i / \bar{d}_i^T (Q(w^k) \cdot \bar{d}_i)$.
      4.1.5. Set $z_{i+1} \leftarrow z_i + \gamma_i d_i$.
      4.1.6. If (C-iii) holds: $\|z_{i+1}\| \geq \Delta^k$ find $\tau > 0$ such that $\|s_d^k\| = \Delta^k$ with $s_d^k =$
             $z_i + \tau d_i$ and STOP.
      4.1.7. Set $r_{i+1} \leftarrow r_i + \gamma_i N^T (Q(w^k) \cdot \bar{d}_i)$, and apply preconditioner $y_{i+1} = D^{-1} r_{i+1}$.
      4.1.8. If (C-i) holds: $\|r_{i+1}\| \leq \epsilon$ set $s_k^d = z_{i+1}$ and STOP.
      4.1.9. Set $\delta_{i+1} \leftarrow r_{i+1}^T y_{i+1} / r_i^T y_i$ and $d_{i+1} \leftarrow -y_{i+1} + \delta_{i+1} d_i$ and RETURN TO
             4.1.2.
   4.2. *Minor Step Test.* Set $w_{j+1}^k \leftarrow w_j^k + N_j^k s_d^k$. If improvement over Cauchy test (5.9)
        and (5.11) are satisfied, STOP refinement search, set $s^k = w_{j+1}^k - w^k$, and GO
        TO 5.
   4.3. *Update Minor Step.* Cut step to satisfy (5.12),(5.13), and (5.18), update $w_{j+1}^k$,
        set $j \leftarrow j + 1$, and RETURN TO 4.1.
5. **Update Step**. Compute reduction ratio (5.5), update step $w^{k+1}$ according to (5.6),
   and update trust-region radius $\Delta^{k+1}$ according to (5.7). Set $k \leftarrow k+1$, and RETURN
   to 2.
```

C) Given $w^*(\alpha, \beta)$, check primal-dual infeasibility $\|w^*(\alpha,\beta) - \mathrm{Proj}(w^*(\alpha,\beta) - \nabla_w \mathcal{L}(w^*(\alpha,\beta)))\| \leq \tau_P$. If satisfactory, STOP. Otherwise, update α, β and RETURN TO B).

Relationship of the EPDF-TRN Algorithm to Objectives (i)-(iv): For fixed and appropriate α, β, the EPDF-TRN algorithm does not use third-order derivative information and exhibits global convergence, Theorem 5.1. In addition, it satisfies (i) from Theorem 5.3. By design, since it uses only PCG and projected gradient, it is matrix-free and uses iterative linear algebra, whereas the truncated trust-region allows it to accommodate negative curvature, so (ii) is achieved. Moreover, inertia detection through linear algebra is not necessary, nonconvexity being handled by the truncated trust-region approach. In the limit of the active set being correctly detected and being close to the optimal manifold and thus the trust-region being inactive, as one does a time shift for the parametric problem (1.2), one PCG iteration will reduce the EPDF for the equality-constrained problem, so monotonic progress will be achieved, and thus (iii). Moreover, inheriting from its bound-constrained active set philosophy, warm start is intrinsically achieved, whereas active set detection can be efficiently done by gradient projection so (iv) is achieved.

6.2. Choice of α, β.

The success of the EDPF formulation strongly relies on the choice of α, β. The results of Theorems 4.1 and 4.2 indicate that β should be small and α large in order to avoid the introduction of spurious critical points that are not minimizers of the original NLP. To detect a poorly chosen β during the search, Bertsekas suggests monitoring the curvature of the Hessian of the EDPF. A key advantage of our framework is that this can be done through the PCG procedure. In particular, if negative curvature is persistently observed (after a given number of iterations), we can terminate the current search and adjust α, β. Bertsekas argues that decreases in β should be accompanied by increases in α so as to keep the product $\alpha \cdot \beta$ constant. He provides a test to determine appropiate values. This follows from the analysis of the QP,

$$\min_{d} \ \frac{1}{2} d^T H d \tag{6.1}$$

$$\text{s.t. } Ad = 0. \tag{6.2}$$

Here, the gradient and right-hand side of the constraints can be dropped because the analysis is of second order. The EDPF of this QP is $P_{QP} = \frac{1}{2} d^T H d + \lambda^T A d + \frac{1}{2} \alpha d^T A^T A d + \frac{1}{2} \beta (Hd + A^T \lambda)^T (Hd + A^T \lambda)$. Bertsekas proved that a condition for $\nabla^2 P$ to be positive definite is that

$$\alpha\beta(AA^T)^2 - \alpha A H A^T - AA^T > 0_m. \tag{6.3}$$

This is equivalent to testing locally (neglecting third-order terms) for the curvature of the EDPF (3.4) for an equality-constrained NLP. In our setting, we can determine values for α, β using the reduced-space QP (5.15) by defining $H := N_x^T Q_{x,x} N_x$, $A := AN_x$. Here N_x is the null-space matrix at the current iterate, and

$$Q_{x,x} = \nabla_{x,x}\mathcal{L} + 4\beta\nabla_{x,x}\mathcal{L} X \nabla_{x,x}\mathcal{L} + 4\beta\nabla_{x,x}\mathcal{L}\text{diag}(\nabla_x\mathcal{L}) + 4\beta\text{diag}(\nabla_x\mathcal{L})\nabla_{x,x}\mathcal{L}, \tag{6.4}$$

is the x, x component of Q in (4.13a). Another way (post-optimization) of detecting inappropriate values of β is convergence to stationary points where some gradients $\nabla_x\mathcal{L}(w^*)_{(i)}$, $i \in \mathcal{A}(x^*)$ are negative [5]. If this case is encountered, we decrease β and increase α by a proportional ratio.

6.3. Derivatives. We can assemble the gradient and approximate Hessian of the EDPF (4.13a) times a vector using only vector products with the Hessian of the Lagrangian $H = \nabla_{x,x}\mathcal{L}(x, \lambda)$ and the Jacobian of the constraints $A = \nabla_x h(x)$. Evaluating the gradient of the EDPF (2.4) requires one Hessian vector product, one Jacobian vector product, and two Jacobian transpose vector products. We define an arbitrary vector $\nu^T = [\nu_x^T \ \nu_\lambda^T]$. We first note that,

$$\nabla^2\mathcal{L} \cdot \nu = \begin{bmatrix} H & A^T \\ A & \end{bmatrix} \begin{bmatrix} \nu_x \\ \nu_\lambda \end{bmatrix} = \begin{bmatrix} H \cdot \nu_x + A^T \cdot \nu_\lambda \\ A \cdot \nu_x \end{bmatrix}. \tag{6.5}$$

Hence, each product requires one Hessian vector product, one Jacobian vector product, and one Jacobian transpose vector product. Using the above relation, we can see from (4.13a) that three *unique* products are needed to form the Hessian approximation of the EDPF. This can be computed with automatic differentiation (AD) packages at a cost that is proportional to the evaluation of the objective function and constraints (i.e., $O(n)$ and $O(m)$, respectively) [28]. We also highlight that coloring only needs to be performed once because the structure of the Hessian of the EDPF does not change along the search. This is particularly beneficial when the NLP is solved repetitively as in (1.2). The computation of the reduced Hessian can proceed without altering the structure of the Hessian by defining $\nu_x = N \cdot \nu_d$ because the null-space matrix is trivial. These observations are of importance because evaluating the exact Hessian of the EDPF (i.e., by specifying the EDPF directly) requires a recursive application of AD. This results from the appearance gradient of the Lagrange function in the definition of the EDPF.

6.4. Step Computation. The Cauchy point makes progress in the EDPF (e.g., solution of the NLP) in $O(n_w)$ operations. More important, each PCG iteration makes progress in $O(n_w)$ operations with same order for storage requirements. The dominant complexity, as expected, is the application of the preconditioner $D^{-1} \cdot r$.

As revealed by Theorems 4.1 and 4.2 an existing caveat of the EDPF formulation is that a large value of α and a small value of β are typically required to enable identification of critical points of the original NLP. This requirement increases the spectrum of the approximate Hessian Q and of the reduced Hessian $N^T QN$. Interestingly, however, the spectrum does not grow as w^* is approached as in IP methods. General preconditioning techniques that can be used in the proposed approach include constraint preconditioning, incomplete Cholesky, and Bunch-Parlett factorization [3, 24, 34] for which parallel implementations exist. The fact that the PCG matrix is positive definite close to solution also opens the possibility to apply general algebraic multi-grid preconditioners, which scale as $O(n_w)$ [37]. We also highlight that inertia is detected externally through PCG so that linear algebra solvers do not need to provide inertia, as is currently required by IP methods [35]. This puts fewer restrictions on the range of applicable iterative linear algebra solvers.

6.5. Warm-Start and Early Termination. The ability to warm-start is a key property of the proposed approach. In particular, because it is based on gradient projection, no bound multiplier information and no centrality recovery are required, as in IP methods. In addition, multiple active set changes can be computed at each iteration through the Cauchy search. The ability to determine activity quickly is particularly relevant in applications that require early termination, such as model predictive control, state estimation, and rigid body simulation [15, 42, 1, 26].

7. Numerical Studies.

In this section we illustrate the behavior of the proposed framework, and we demonstrate its scalability properties.

7.1. Algorithmic Behavior. We explain the behavior of the algorithmic framework using the following example:

$$\min \quad (x_1-1)^2+(x_2-2)^2+(x_3-3)^2+x_1x_4 \tag{7.1a}$$
$$\text{s.t.} \quad x_1x_4+x_1x_2+x_3=4, \quad (\lambda) \tag{7.1b}$$
$$x_1,x_2,x_3,x_4 \geq 0. \tag{7.1c}$$

The solution of this problem is $x^*=[1.62\ 1.62\ 1.38\ 0]$, $\lambda^*=-7.64$, and $f(x^*)=0.91$. The gradient of the Lagrangian is given by

$$\nabla_x \mathcal{L}(x,\lambda)=\begin{bmatrix} 2(x_1-1)+x_4+\lambda x_2+\lambda x_4 \\ 2(x_2-2)+\lambda x_1 \\ 2(x_3-3)+\lambda \\ x_1+\lambda x_1 \end{bmatrix}. \tag{7.2}$$

We can see that nonlinear terms exist in λ and x_1, x_2 and x_4. This induces third-derivative terms in the Hessian of the EDPF. To solve this problem, we set $\alpha = 1\text{e}{+}2$ and $\beta = 1\text{e}{-}3$ and use tolerance for the projected gradient of 1e$-$5. We initialize the problem at $x^0=[1\ 1\ 1\ 1]$ and $\lambda^0 = 1$. We summarize the convergence history of the trust-region Newton algorithm in Table 7.1. Here, we define $H^k := \nabla^2 P(w^k)$ and $H_d^k := N^T\nabla^2 P(w^k)N$; $\underline{\lambda}(H)$ is the minimum eigenvalue of matrix H. We make the following observations:

- The step in $k=1$ is obtained from a direction of negative curvature and is accepted because it leads to $\rho^k > 1$. The trust-region radius Δ^k is increased. We can also observe a large error in the Hessian approximation.
- The step in $k=2$ is rejected because $\rho^k < 0$. The trust-region radius is decreased. The same behavior is observed until $k=6$, when Δ^k is sufficiently small to make $\rho^k > 0$. The step at this iteration results from a direction of negative curvature and is accepted but Δ^k is kept constant because ρ^k is not large enough.

TABLE 7.1
Convergence history for example problem.

k	P^k	g^k_{Proj}	ρ^k	$\\|s^k\\|$	$\\|\Delta^k\\|$	$\\|Q^k - H^k\\|$	$\underline{\lambda}(Q^k_d)$	$\underline{\lambda}(H^k_d)$	card$(\mathcal{A}^k_P)$
0	25.150	2.0e+2							0
1	3.449	5.9e+1	+3.26	2.5e−1	261.9	2.0e+2	-2.48	-22.67	0
2	3.449	5.9e+1	-0.70	0.0e+0	523.9	5.8e+1	-2.48	-22.67	0
3	3.449	5.9e+1	-0.62	0.0e+0	131.0	5.8e+1	-2.48	-22.67	0
4	3.449	5.9e+1	-0.33	0.0e+0	32.0	5.8e+1	-2.48	-22.67	0
5	3.449	5.9e+1	-0.28	0.0e+0	8.0	5.8e+1	-2.48	-22.67	0
6	1.533	2.5e+1	+0.37	2.0e+0	2.0	5.8e+1	-2.48	-22.67	0
7	0.945	1.6e+0	+0.52	1.9e−1	2.0	2.9e+1	+0.15	-0.39	0
8	0.944	4.9e−1	+0.48	2.6e−3	4.0	1.9e+0	+0.19	+0.37	0
9	0.943	4.5e−1	+0.93	1.4e−3	4.0	4.0e−1	+0.19	+0.25	0
10	0.909	2.3e−1	+0.94	1.8e−1	8.0	3.4e−1	+0.40	+0.40	1
11	0.908	1.7e−6	+0.99	8.7e−3	16.0	3.1e−6	+0.38	+0.38	1

- At iterations $k = 7, 8, 9$ the algorithm makes progress, and Δ^k keeps increasing. At $k = 7$ we note that the minimum eigenvalue of the approximate reduced Hessian is positive while that of the exact is negative. This illustrates how the third-derivative term can introduce negative curvature.
- At iterations $k = 10, 11$ the Cauchy step detects the active variable and superlinear convergence is observed. The error of the approximate Hessian converges to zero, and ρ^k converges to one. The Hessian is positive definite.

7.2. Scalability. To demonstrate scalability, we consider the following continuous-time optimal control problem (OCP):

$$\min \int_0^T \left(\alpha_c \cdot (c(\tau) - \bar{c})^2 + \alpha_t \cdot (t(\tau) - \bar{t})^2 + \alpha_u \cdot (u(t) - \bar{u})^2\right) d\tau \tag{7.3a}$$

$$\text{s.t.}\;\; \dot{c}(\tau) = \frac{1 - c(\tau)}{\theta} - p_k \cdot \exp\left(-\frac{p_E}{t(\tau)}\right) \cdot c(t) \tag{7.3b}$$

$$\dot{t}(\tau) = \frac{t_f - t(\tau)}{\theta} + p_k \cdot \exp\left(-\frac{p_E}{t(\tau)}\right) \cdot c(\tau) - p_\alpha \cdot u(\tau) \cdot (t(\tau) - t_c) \tag{7.3c}$$

$$c(\tau), t(\tau), u(\tau) \geq 0, \quad \tau \in [0, T] \tag{7.3d}$$

$$c(0) = c(\tau_{sys}), \quad t(0) = t(\tau_{sys}). \tag{7.3e}$$

The system is an unstable chemical reactor. The internal time is given by τ covering the prediction horizon $[0, T]$. The system states are concentration of reactant $c(\cdot)$ and temperature of reacting mixture $t(\cdot)$. The control is the cooling water flow $u(\cdot)$. The real time of the system is τ_{sys} and the system states at this time are $c(\tau_{sys}), t(\tau_{sys})$. Symbols $\alpha_c, \alpha_t, \alpha_u, p_\alpha, p_E, t_f, t_c, p_k$ are model parameters and can be found in [42]. The objective function is of Bolza type with the desired end points $\bar{c}, \bar{t}, \bar{u}$. We transform this problem into an NLP by direct transcription with implicit Euler discretization. We use a mesh with N points each of length $\Delta\tau$. To scale the problem, we increase the number of time steps in the horizon N. The resulting dimensions of the NLPs are presented in Table 7.2. Here, we present the density in percentage for the Hessian of the Lagrange function %dens$(\nabla^2\mathcal{L})$ and for the approximate Hessian of the EDPF %dens(Q). From these numbers we observe that the Hessian is very sparse, which is typical

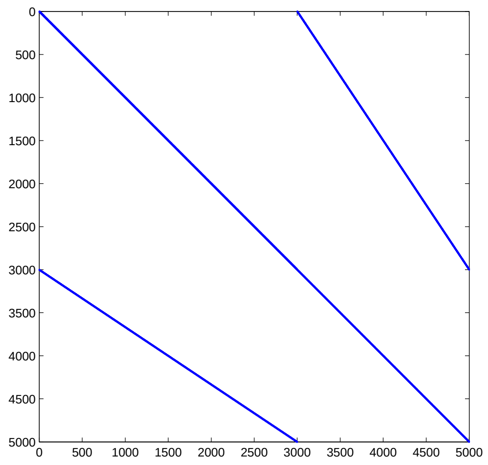

FIG. 7.1. *Sparsity structure of approximate Hessian Q for NLP with $n_w = 5,000$.*

TABLE 7.2
Dimensions of discretized OCP.

N	n	m	n_w	nnz($\nabla^2\mathcal{L}$)	nnz(Q)	%dens($\nabla^2\mathcal{L}$)	%dens(Q)
500	1,500	1,000	2,500	10,486	26,492	2.0e−1	4.0e−1
1,000	3,000	2,000	5,000	20,996	52,972	8.4e−2	2.0e−1
5,000	15,000	10,000	25,000	104,996	264,972	1.6e−2	4.0e−2
10,000	30,000	20,000	50,000	209,996	529,972	8.3e−3	2.1e−2

in direct transcription approaches for optimal control [43]. We also observe that, even if the number of nonzeros in the EDPF Hessian is larger by a factor of 2.5 compared to the Hessian of the Lagrangian (augmented system), sparsity is preserved. The sparsity structure for a problem with effective dimension $n_w = 5,000$ is presented in Figure 7.1, from where we can also appreciate a banded structure also typical in OCPs [32].

In this study, we test for scalability of the two dominant computational steps. The first step is the incomplete sparse Cholesky factorization of the approximate Hessian matrix to generate the preconditioner D for PCG. In our experiments we set the drop tolerance to 1e−4. We compare against the case in which a full sparse Cholesky factorization of the Hessian is performed. The second dominant step is the PCG search, which involves recursive backsolves with the factors of the preconditioner. In the case of full Cholesky factorization a single PCG iteration (backsolve) is needed. All computations were performed in Matlab.

To solve these problems, we used $\alpha = 1\text{e}+6$ and $\beta = 1\text{e}{-1}$, and we initialize them at a perturbed point from the optimal solution generated. This perturbation gives an initial error of the projected gradient of $O(10^3)$. We use a tolerance for the projected gradient of 1e−5. All problems were solved in four iterations.

Scalability results as a function of the effective dimension n_w are reported in Table 7.3. Here, we compare the average number of PCG iterations (it_{cg}) per Newton iteration, average time of PCG procedure ($\theta_{ichol,pcg}$) per Newton iteration, average time of incomplete factorization ($\theta_{ichol,fact}$), and average total time per iteration $\theta_{ichol,tot} = \theta_{ichol,pcg} + \theta_{ichol,fact}$. The same quantities are presented for the full Cholesky case and are denoted as $\theta_{chol,pcg}$, $\theta_{chol,fact}$, and $\theta_{chol,tot}$, respectively. The times are illustrated graphically in Figure 7.2. We make the following observations

TABLE 7.3
Average computational times for OCP per Newton iteration.

n_w	it_{pcg}	$\theta_{ichol,pcg}$	$\theta_{ichol,fact}$	$\theta_{ichol,tot}$	$\theta_{chol,pcg}$	$\theta_{chol,fact}$	$\theta_{chol,tot}$
1,250	17	8.5e−2	3.1e−2	1.1e−1	2.7e−2	3.3e−2	6.0e−2
2,500	24	4.9e−1	1.3e−1	6.2e−1	1.1e−1	1.5e−1	2.6e−1
5,000	29	1.7e+0	4.4e−1	2.2e+0	5.7e−1	8.5e−1	1.4e+0
12,500	31	9.0e+0	1.8e+0	1.1e+1	3.8e+0	8.4e+0	1.2e+1
25,000	31	1.8e+1	5.5e+0	2.4e+1	2.5e+1	5.4e+1	7.8e+1
50,000	31	3.7e+1	1.8e+1	5.5e+1	-	-	-
125,000	31	9.4e+1	1.1e+2	2.0e+2	-	-	-
250,000	31	1.9e+2	4.9e+2	6.8e+2	-	-	-

- The number of PCG iterations increases linearly with n_w and settles. This "settling" is a particular property of OCPs with Bolza objective because the end point $\bar{c}, \bar{t}, \bar{u}$ is reached after a given number of time steps N and the Krylov subspace remains constant.
- The total times for full Cholesky are not competitive. This result is clearly seen in the left panel of Figure 7.2. This illustrates the additional flexibility gained by enabling relaxation in the incomplete Cholesky factorization procedure.
- From the full Cholesky times we can see that, as expected, the complexity of performing a single backsolve $\theta_{chol,pcg}$ is similar to that performing the factorization $\theta_{chol,fact}$ particularly because of fill-in effects.
- From the incomplete Cholesky times and the number of PCG iterations we can see that the time per backsolve can be dramatically reduced. This is particularly evident in the very large problems where the number of PCG iterations is constant.
- The number of backsolves needed make PCG the dominant expense below the $n_w = 100,000$ threshold. Factorization becomes dominant above the threshold. This is illustrated in right panel of Figure 7.2.

To demonstrate the activity detection properties of the framework, we generated an NLP with $n_w = 2,500$. We consider two cases. In the first case we start the search at a point with $\mathcal{A}_P(w^0) = 44$ and with optimal solution $\mathcal{A}_P(w^*) = 173$. In the second case we reverse the process to generate an initial point with $\mathcal{A}_P(w^0) = 173$ and $\mathcal{A}_P(w^*) = 44$. The convergence history is summarized in Table 7.4. As can be seen, the Cauchy step can make large adjustments in the active set at each iteration. This is a key advantage over traditional SQP methods that allow only for one active set change per iteration. Moreover, the number of PCG iterations remains fairly stable and in fact reduces as the solution is approached (as the third order term vanishes and the active set settles). This is a key advantage over IP methods where preconditioner performance degrades as the solution is approached [30].

7.3. Warm-Starting and Early Termination. One of the crucial properties of the framework is that it enables warm-start and early termination. To demonstrate these capabilities, we consider a receding-horizon OCP framework. This approach is also called model predictive control (MPC) and is typically used to avoid solving problems with infinite horizons. The OCP (7.3) is solved recursively, as a pNLP (1.2) and the time horizon is shifted by a factor $\Delta\tau$ as $\tau_{sys} \leftarrow \tau_{sys} + \Delta\tau$. The initial states are updated using the time-evolving $c(\tau_{sys}), t(\tau_{sys})$ system states as $c(0) \leftarrow c(\tau_{sys})$, $t(0) \leftarrow t(\tau_{sys})$. This update is done until the system converges $c(\tau_{sys}) \to \bar{c}\ t(\tau_{sys}) \to \bar{t}$. The procedure generates continuous-time manifold $c^*(\tau), t^*(\tau), u^*(\tau),\ \tau \in [0, \tau_{sys}]$, where τ_{sys}.

The main obstacle preventing the use of MPC is the computational latency of the OCP

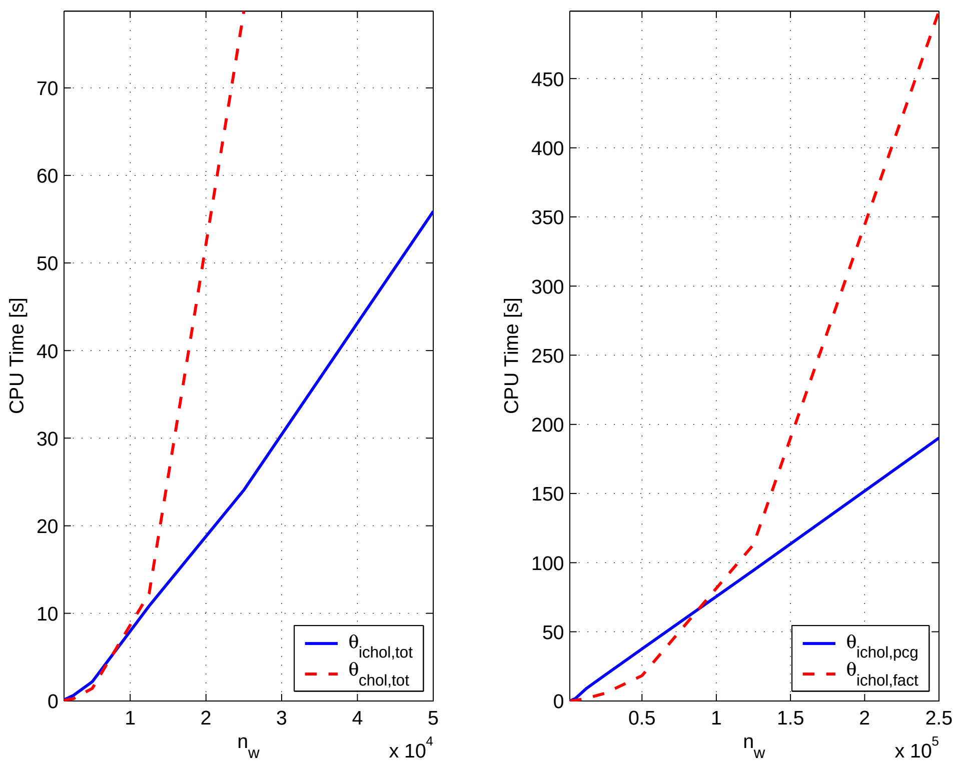

FIG. 7.2. *Average total time per iteration with Cholesky and incomplete Cholesky (left). Average PCG and incomplete Cholesky times per iteration (middle). Average number of PCG iterations (right).*

TABLE 7.4
active set identification histories for NLP with $n_w = 2,500$.

	Case 1				Case 2			
k	P^k	g^k_{Proj}	$\mathcal{A}_P(w^k)$	n^k_{PCG}	P^k	g^k_{Proj}	$\mathcal{A}_P(w^k)$	n_{PCG}
0	4.05e+3	4.52e+3	44	-	1.21e+4	2.43e+5	173	-
1	1.14e+2	4.70e+3	44	41	4.96e+2	5.76e+4	0	132
2	1.83e+1	3.72e+3	119	32	9.48e+1	1.86e+3	0	45
3	1.83e+1	1.55e+2	170	27	5.57e+0	3.27e+4	26	37
4	1.83e+1	5.59e−6	173	17	3.98e+0	1.11e+3	43	26
5	-	-	-		3.98e+0	8.50e−6	44	13

solution. It has been recently shown by the authors that one inexact QP solution (inexact Newton iteration in case of no inequalities) for the NLP can be sufficient to track the optimal manifold stably and to ultimately steer the system to the end point [42]. In the case described here, the use of iterative linear algebra provides significantly more flexibility than does direct linear algebra for early termination. This flexibility, combined with fast active set identification, enables the use of MPC in a much wider range of applications. Recently, we proposed an augmented Lagrangian NLP reformulation and a projected Gauss-Seidel scheme to solve the QP inexactly by terminating after a finite number iterations [42]. The approach, however, is limited in scalability because of the difficulty in preconditioning projected Gauss-Seidel, and it requires estimates of the multipliers. The EDPF approach proposed here overcomes these limitations.

To illustrate the flexibility gained with the ability to warm-start and terminate early, we performed MPC computations for a problem with $N = 100$. We compute optimal manifolds from different starting points by solving the NLPs with a tolerance of 1e−5. This process required, on average, 4 Newton iterations and 40 PCG iterations per Newton iteration. For comparison, we consider a suboptimal strategy in which we terminate after 2 Newton iterations and and 20 PCG iterations per Newton iteration. This *reduced the latency by a factor of 4*. The optimal and suboptimal manifolds are presented in Figure 7.3. As can be seen, errors are present but are stable and the trajectories converge in all cases to the desired end point.

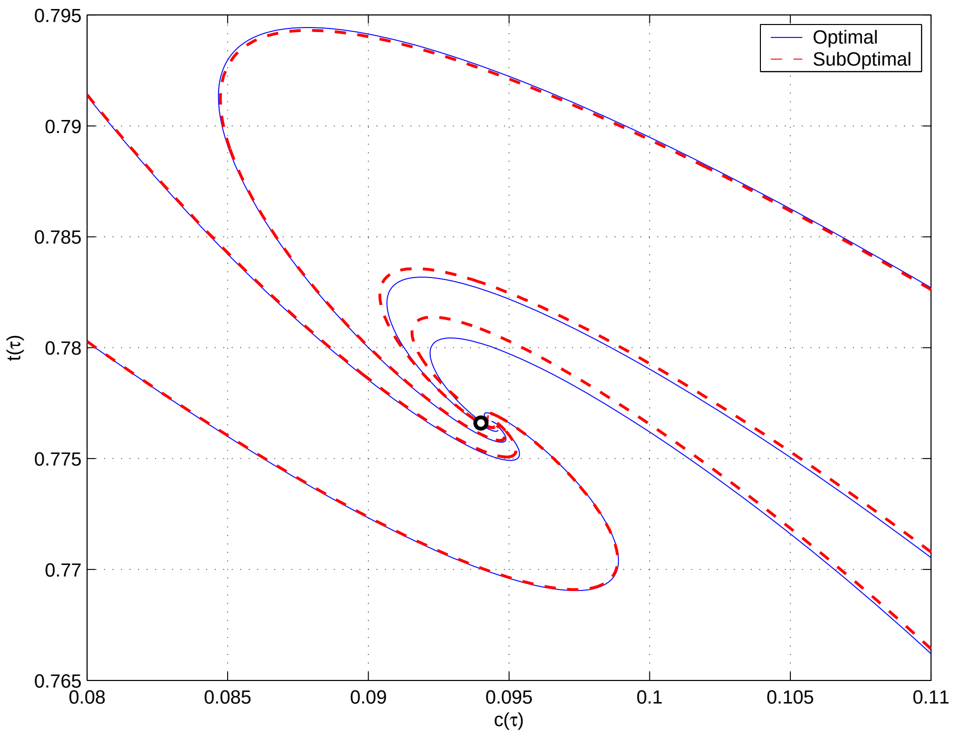

Fig. 7.3. *Phase plane manifolds with tight tolerance and with early termination.*

8. Conclusions and Future Work.

We have presented a general approach for nonlinear programming based on the direct minimization of an exact differentiable penalty function using a trust-region Newton setting. We demonstrate that the framework is scalable in the sense that (i) it is superlinearly convergent, (ii) it is matrix-free and exploits directions of negative curvature, (iii) it makes direct progress on the merit function in its minor iterations, and (iv) it enables efficient detection of activity through gradient projection and enables the use of activity information to warm-start.

As part of future work, we will develop a more general implementation of the framework allowing for automatic adjustment of the penalty function parameters. In addition, we will consider more general penalty functions requiring only the α parameter. This function has the form [4]:

$$P_\alpha(w) = \mathcal{L}(w) + \frac{1}{2}\alpha h(x)^T h(x) + 2\,\nabla_x \mathcal{L}(w)^T \sqrt{X} M(x) M(x)^T \sqrt{X} \nabla_x \mathcal{L}(w), \qquad (8.1)$$

where $M(x) \in \mathcal{R}^{n\times p}$, $m \leq p \leq n$ is a matrix that makes $M(x)\nabla_x h(x)$ nonsingular. A typical choice is $M(x) = \nabla_x h(x)^T$ with $p = m$. The use of this function will enable more algorithmic flexibility and will limit the spectrum of the Hessian matrix. However, derivative computations are more complicated and efficient ways for computing them using Hessian-vector and Jacobian-vector products need to be investigated. These extensions will require major modifications to existing convergence results because a more intrusive implementation of the trust-region Newton method and different step acceptance criteria are needed. We will also investigate different approaches to efficiently solve the trust-region QP [33]. Moreover, we will investigate efficient preconditioning approaches including of the multilevel type for the resulting linear systems, which is somewhat facilitated by the fact that we do not need to detect inertia as is the case for other NLP frameworks. All these additions will enable us to benchmark against existing NLP solvers.

Acknowledgments. Mihai Anitescu thanks Stephen Wright for pointing him to some key exact penalty function issues during the former's Wilkinson fellowship. This work was supported by the U.S. Department of Energy, under Contract No. DE-AC02-06CH11357.

REFERENCES

[1] M. Anitescu and F. A. Potra. Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems. *Nonlinear Dynamics*, 14:231–247, 1997.
[2] P. Armand and D. Orban. The squared slacks transformation in nonlinear programming. *http://www.optimization-online.org/DBFILE/2007/08/1762.pdf*, 2007.
[3] M. Benzi. Preconditioning techniques for large linear systems: A survey. *Journal of Computational Physics*, 182(2):418 – 477, 2002.
[4] D. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. Academic Press, 1982.
[5] D. Bertsekas. Enlarging the region of convergence of Newton's method for constrained optimization. *Journal of Optimization Theory and Applications*, 36:221–252, 1982.
[6] E. G. Birgin, R. A. Castillo, and J. M. Martnez. Numerical comparison of augmented lagrangian algorithms for nonconvex problems. *Computational Optimization and Applications*, 31:31–55, 2005.
[7] J. Burke, J. Moré, and G. Toraldo. Convergence properties of trust region methods for linear and convex constraints. *Mathematical Programming*, 47:305–336, 1990.
[8] James V. Burke and Jorge J. More. On the identification of active constraints. *SIAM Journal on Numerical Analysis*, 25(5):pp. 1197–1211, 1988.
[9] R. H. Byrd, J. Ch. Gilbert, and J. Nocedal. A trust-region method based on interior-point techniques for nonlinear programming. *Math. Programm.*, 89:149–185, 2000.
[10] A. Conn, N. Gould, and P. Toint. A globally convergent augmented lagrangian algorithm for optimization with general constraints and simple bounds. *SIAM Journal on Numerical Analysis*, 28(2):545–572, 1991.
[11] A. Conn, Nick Gould, and Ph. Toint. Numerical experiments with the lancelot package (release a) for large-scale nonlinear optimization. *Mathematical Programming*, 73:73–110, 1996.
[12] F. Curtis, O. Schenk, and A. Wvchter. An interior-point algorithm for large-scale nonlinear optimization with inexact step computations. *SIAM Journal on Scientific Computing*, 32(6):3447–3475, 2010.
[13] G. Di Pillo and L. Grippo. A new class of augmented lagrangians in nonlinear programming. *SIAM Journal on Control and Optimization*, 17(5):618–628, 1979.
[14] G. Di Pillo and L. Grippo. Exact penalty functions in constrained optimization. *SIAM Journal on Control and Optimization*, 27(6):1333–1360, 1989.
[15] M. Diehl, H. J. Ferreau, and N. Haverbeke. Efficient numerical methods for nonlinear MPC and moving horizon estimation. *In Nonlinear Model Predictive Control*, pages 391–417, 2009.
[16] A. Forsgren, P. E. Gill, and M. H. Wright. Interior methods for nonlinear optimization. *SIAM Review*, 44(4):pp. 525–597, 2002.
[17] A. Forsgren, P. E. Gill, and M. H. Wright. Interior methods for nonlinear optimization. *SIAM Review*, 44:525–597, 2002.

[18] A. Forsgren, P.E. Gill, and J.D. Griffin. Iterative solution of augmented systems arising in interior methods. *SIAM Journal on Optimization*, 18(2):666–690, 2008.

[19] P. Gill, W. Murray, and M. Wright. *Practical Optimization.* Academic Press, 1981.

[20] P. E. Gill and D. Robinson. A primal-dual augmented lagrangian. *Computational Optimization and Applications*, 51:1–25, 2012.

[21] J. Gondzio and A. Grothey. Reoptimization with the primal-dual interior point method. *SIAM Journal on Optimization*, 13(3):842–864, 2002.

[22] S. P. Han and O. L. Mangasarian. Exact penalty functions in nonlinear programming. *Mathematical Programming*, 17:251–269, 1979.

[23] A. Izmailov and M. Solodov. Inexact josephynewton framework forgeneralized equations anditsapplications tolocalanalysis ofnewtonianmethods forconstrained optimization. *Computational Optimization and Applications*, 46:347–368, 2010.

[24] C. Lin and J. Moré. Incomplete cholesky factorizations with limited memory. *SIAM Journal on Scientific Computing*, 21(1):24–45, 1999.

[25] C. Lin and J. Moré. Newton's method for large bound-constrained optimization problems. *SIAM Journal on Optimization*, 9(4):1100–1127, 1999.

[26] J. L. Morales, J. Nocedal, and M. Smelyanskiy. An algorithm for the fast solution of symmetric linear complementarity problems. *Numerische Mathematik*, 11:251–266, 2008.

[27] Stephen G. Nash. Sumt (revisited). *Operations Research*, 46(6):pp. 763–775, 1998.

[28] J. Nocedal and S. Wright. *Numerical Optimization.* Springer, New York, NY, 1999.

[29] T. Ohtsuka. A continuation/GMRES method for fast computation of non-linear receding horizon control. *Automatica*, 40:563–574, 2004.

[30] C. Petra and M. Anitescu. A preconditioning technique for schur complement systems arising in stochastic optimization. *Computational Optimization and Applications*, 52:315–344, 2012.

[31] G. Di Pillo and L. Grippo. A continuously differentiable exact penalty function for nonlinear programming problems with inequality constraints. *SIAM Journal on Control and Optimization*, 23(1):72–84, 1985.

[32] C. V. Rao, S. J. Wright, and J. B. Rawlings. Application of interior-point methods to model predictive control. *JOTA*, 99:723–757, 1998.

[33] M. Rojas, S. Santos, and D. Sorensen. A new matrix-free algorithm for the large-scale trust-region subproblem. *SIAM Journal on Optimization*, 11(3):611–646, 2001.

[34] O. Schenk, A. Wächter, and M. Hagemann. Matching-based preprocessing algorithms to the solution of saddle-point problems in large-scale nonconvex interior-point optimization. *J. Comp. Opt. and App.*, 36:321–341, 2007.

[35] O. Schenk, A. Wächter, and M. Weiser. Inertia-revealing preconditioning for large-scale nonconvex constrained optimization. *SIAM Journal on Scientific Computing*, 31(2):939–960, 2009.

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

[37] K. Stueben. A review of algebraic multigrid. *Journal of Computational and Applied Mathematics*, 128(1):281 – 309, 2001.

[38] A. Wächter and L. Biegler. Line search filter methods for nonlinear programming: Motivation and global convergence. *SIAM Journal on Optimization*, 16(1):1–31, 2005.

[39] A. Wächter and L. T. Biegler. On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106:25–57, 2006.

[40] R.A. Waltz, J.L. Morales, J. Nocedal, and D. Orban. An interior algorithm for nonlinear optimization that combines line search and trust region steps. *Mathematical Programming*, 107:391–408, 2006.

[41] V. M. Zavala. *Computational Strategies for the Operation of Large-Scale Chemical Processes, Ph. D. thesis.* Carnegie Mellon University, 2008.

[42] V. M. Zavala and M. Anitescu. Real-time nonlinear optimization as a generalized equation. *SIAM Journal on Control and Optimization*, 48(8):5444–5467, 2010.

[43] V. M. Zavala and L. T. Biegler. Nonlinear programming strategies for state estimation and model predictive control. *In Nonlinear Model Predictive Control*, pages 419–432, 2009.