Les Cahiers du GERAD ISSN: 0711–2440

A regularized factorization-free method for equality-constrained optimization

S. Arreckx
D. Orban

G–2016–65

August 2016
Revised: January 2018

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2018
– Bibliothèque et Archives Canada, 2018

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2018
– Library and Archives Canada, 2018

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

A regularized factorization-free method for equality-constrained optimization

Sylvain Arreckx

Dominique Orban

GERAD & Department of Mathematics and Industrial Engineering, Polytechnique Montréal, Montréal (Québec) Canada

sylvain.arreckx@gerad.ca

dominique.orban@gerad.ca

August 2016

Revised: January 2018

Les Cahiers du GERAD

G–2016–65

Abstract: We propose a method for equality-constrained optimization based on a problem in which all constraints are systematically regularized. The regularization is equivalent to applying an augmented Lagrangian method but the linear system used to compute a search direction is reminiscent of regularized sequential quadratic programming (SQP). A limited-memory BFGS approximation to second derivatives allows us to employ iterative methods for linear least squares to compute steps, resulting in a factorization-free implementation. We establish global and fast local convergence under weak assumptions. In particular, we do not require the LICQ and our method is suitable for degenerate problems. Preliminary numerical experiments show that a factorization-based implementation of our method exhibits significant robustness while a factorization-free implementation, though not as robust, is promising. We briefly discuss generalizing our framework to other classes of methods and to problems with inequality constraints.

Keywords: Sequential quadratic programming, regularization, augmented Lagrangian, limited-memory BFGS, factorization-free method

Résumé : Nous proposons une méthode d'optimisation sans factorisation pour les problèmes avec contraintes d'égalité pour lesquels toutes les contraintes sont systématiquement régularisées. Cette régularisation est équivalente à l'application d'une méthode de lagrangien augmenté dans laquelle les systèmes linéaires utilisés pour calculer une direction de recherche sont similaires à ceux des méthodes de programmation quadratique séquentielle (SQP). Grâce à l'emploi d'approximations BFGS à mémoire limitée des dérivées secondes, des méthodes itératives pour les moindres carrés linéaires peuvent être utilisées afin de calculer par étapes, faisant de la méthode proposée une méthode sans factorisation. Nous établissons rapidement une convergence globale et locale sous de faibles hypothèses. En particulier, la LICQ n'est pas requise et notre méthode est adaptée pour la résolution de problèmes dégénérés. Les résultats numériques préliminaires montrent qu'une implémentation avec factorisation de notre méthode est significativement robuste alors qu'une implémentation sans factorisation, bien que moins robuste, est prometteuse. Une brève discussion est incluse sur la généralisation de notre approche à d'autres classes de méthodes ainsi qu'aux problèmes avec inégalités.

Mots clés : Programmation quadratique séquentielle, régularisation, méthode de lagrangien augmenté, BFGS à mémoire limitée, méthode sans factorisation

Acknowledgments: Research partially supported by an NSERC Discovery Grant.

1 Introduction

We consider the general equality-constrained optimization problem

$$\underset{x\in\mathbb{R}^n}{\text{minimize}}\ f(x) \quad \text{subject to}\ c(x)=0. \tag{1}$$

The main objective of this paper is to devise an implementable factorization-free algorithm for (1) in the large-scale case that is somewhat resilient to constraint degeneracy and does not require matrix-vector products with potentially ill-conditioned operators such as $J(x)^T J(x)$, where J is the Jacobian of c, as is the case with a standard augmented Lagrangian method. We propose a framework inspired by that of [3] in which all constraints are systematically regularized. We show that the regularization can be interpreted as a proximal-point Hestenes-Powell augmented Lagrangian method applied to (1) in the same vein as [40]. Our method uses the proximal augmented Lagrangian as merit function to promote global convergence and asymptotically blends into a stabilized SQP method possessing fast local convergence properties. Thanks to appropriate limited-memory BFGS approximations of the Hessian of the Lagrangian, the linear system encountered at each iteration is symmetric and quasi-definite (SQD) [41], permitting inexact solves and an entirely factorization-free implementation suggested by methods described in [36]. We assume that $f:\mathbb{R}^n\to\mathbb{R}$ and $c:\mathbb{R}^n\to\mathbb{R}^m$ are twice continuously differentiable, although we only use exact second derivatives as an instrument in the analysis and in numerical illustration. In practice, only first derivatives are required when employing L-BFGS approximations.

The Karush-Kuhn-Tucker (KKT) conditions for (1) are only necessary for optimality when a constraint qualification holds. The most widely used constraint qualification condition, the Linear Independence Constraint Qualification (LICQ), requires that all constraint gradients be linearly independent at a stationary point. When such a constraint qualification fails to hold, the KKT conditions cease to be reliable for optimality. When it fails to hold at intermediate iterates, computational difficulties also arise. In particular, the linear systems used to compute search directions may become singular. Our regularization scheme is designed so as to overcome such complications.

We show that our method possesses global convergence properties similar to those of augmented Lagrangian methods [6, 7]. In addition, we show that whenever the sequence of iterates converges to an isolated minimizer, the algorithm reduces asymptotically to pure stabilized SQP iterations and converges superlinearly. The convergence rate is quadratic if second-derivative approximations and steps are sufficiently accurate. Our numerical experiments show that the proposed scheme is efficient and robust.

Related Work

Sequential quadratic programming (SQP) methods [8, 43] are among the most successful methods for the solution of (1). They compute steps via a sequence of subproblems in which a quadratic model of the Lagrangian is minimized subject to linearized constraints. Convergence is enforced by requiring an improvement in a merit function at each step. Each iteration of an SQP method requires the solution of a linear system that involves the constraint Jacobian and its transpose. Most convergence analyses for SQP and most SQP implementations require that those linear systems be solved exactly. In many large-scale applications, constraint Jacobians are only available as linear operators. In such cases, systems must be solved iteratively and inexactly, and it is crucial to account for this inexactness in the design and convergence analysis. An inexact trust-region SQP algorithm for equality constrained optimization is introduced in [28]. A composite-step approach is described in which the step in decomposed into a quasi-normal and a tangential step. A set of stopping criteria designed for controlling the inexactness of substep computations is given and ensures global convergence of their algorithm. The authors of [11] propose an inexact line-search SQP method for (1) where steps are computed from an inexact solution of a KKT system. A perturbation of the Hessian of the Lagrangian is employed to deal with nonconvexity. The perturbation is determined iteratively and may require repeated KKT solves per step computation. Two distinct termination tests control the level of inexactness in the step computation procedure.

Stabilized SQP methods were designed to remedy the numerical and theoretical difficulties associated with degenerate problems [15, 27, 46]. The term *stabilized* refers to the calming effect on multiplier estimates for degenerate problems [27, 45]. Stabilized SQP promises superlinear local convergence under certain assumptions, but not global convergence to a stationary point. The literature is sparse on globally-convergent methods that reduce to a sequence of stabilized SQP steps in the local regime. In [16], the authors combine stabilized SQP with the inexact restoration method to ensure convergence from an arbitrary starting point. The authors of [21] establish connections between stabilized SQP and augmented Lagrangian methods, including primal-dual variants of the augmented Lagrangian. In the equality-constrained case, the method described in the present paper is related to that in [31], where stabilized SQP is combined with the usual augmented Lagrangian algorithm when inequalities are present, although we emphasize algorithmic choices lending themselves to an efficient factorization-free implementation. The method of [31] also applies to problems with inequality constraints, which are penalized in the usual way by a primal augmented Lagrangian. However, their algorithm doesn't allow quasi-Newton approximations to second-order derivatives and linear systems must be solved exactly. In the context of equality-constrained problems, [32] propose outer iterations globalized by a merit function that combines the primal augmented Lagrangian and a quadratic penalty on dual feasibility. Several references focus on the primal-dual augmented Lagrangian as primary merit function. Global and local convergence properties of an active-set method for problems with equality constraints and bounds are presented in [23, 24]. The authors of [2] propose a primal-dual augmented Lagrangian approach to solve equality constrained optimization problems that is quadratically convergent. However, local convergence assumes the LICQ, exact linear system solves and exact second derivatives.

Notation

Throughout the paper, $\|\cdot\|$ denotes the Euclidean norm and I denotes the identity matrix of appropriate size. For any symmetric and positive definite matrix H, the H-norm is defined as $\|u\|_H^2 := u^T H u$. We use $\lambda_{\min}(M)$ and $\lambda_{\max}(M)$ to denote the smallest and largest eigenvalue of any symmetric matrix M. Similarly, $\sigma_{\min}(A)$ and $\sigma_{\max}(A)$ denote the smallest and largest singular values of any matrix A. For two non-negative scalar sequences a_k and b_k converging to zero, we use the Landau symbols $a_k = o(b_k)$ if $\lim_{k\to+\infty} a_k/b_k = 0$ and $a_k = \omega(b_k)$ if $b_k = o(a_k)$. We write $a_k = O(b_k)$ if there exists a constant $C > 0$, such that $a_k \le C b_k$ for large k and $a_k = \Theta(b_k)$ if $a_k = O(b_k)$ and $b_k = O(a_k)$.

The rest of the paper is organized as follows. Section 2 summarizes the connection between augmented Lagrangians and regularized SQP methods. In Section 3, we describe our algorithm in detail. Its global convergence properties are given in Section 4. Local convergence is analyzed in Section 5. We describe our implementations and report on numerical experience in Section 6. We conclude and discuss extensions to our framework in Section 7.

2 A primal-dual regularization and regularized SQP methods

The Lagrangian for (1) is defined as

$$L(x, y) := f(x) - c(x)^T y, \tag{2}$$

where $y \in \mathbb{R}^m$ is the vector of Lagrange multipliers associated to the equality constraints. If x^* is a local minimizer of (1), the KKT conditions require that there exist y^* such that

$$g(x^*) - J(x^*)^T y^* = 0, \quad c(x^*) = 0, \tag{3}$$

where $g(x) := \nabla f(x)$ and $J(x)$ is the Jacobian of $c(x)$. Existence of such a y^* is only guaranteed provided a constraint qualification condition holds at x^*. Should constraint qualifications fail to hold at x^*, there may exist no y^* satisfying (3) or there may exist an unbounded set of them [20]. In either case, numerical methods, such as SQP methods, may be confronted with degenerate direction-finding subproblems.

For the purposes of this paper, we say that (1) is *degenerate* at a feasible x if the LICQ fails to hold at x, i.e., the vectors $\nabla c_i(x)$, $i = 1, \dots, m$, are linearly dependent.

Consider applying an augmented Lagrangian method to (1). If we denote y_k the current approximation of the Lagrange multipliers, the k-th subproblem has the form

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \;\; L(x, y_k) + \tfrac{1}{2}\delta_k^{-1}\|c(x)\|^2, \tag{4}$$

where $\delta_k > 0$ is a penalty parameter. Following the procedure outlined in [19], it is not difficult to see that (4) may equivalently be written as

$$\underset{x\in\mathbb{R}^n, u\in\mathbb{R}^m}{\text{minimize}} \;\; f(x) + \tfrac{1}{2}\delta_k\|u + y_k\|^2 \quad \text{subject to} \;\; c(x) + \delta_k u = 0, \tag{5}$$

for some new variables u. The problem (5) provides an interpretation of the augmented Lagrangian method as an adaptive constraint regularization process. Since the regularization acts on the constraints and adds a term to the objective involving the multipliers, we term it *dual*. Of paramount importance is the fact that the LICQ is satisfied at every feasible point of (5).

In addition to the dual regularization term, we follow [19] and add primal regularization in the form of a proximal-point term, so that the k-th subproblem takes the form

$$\underset{x\in\mathbb{R}^n, u\in\mathbb{R}^m}{\text{minimize}} \;\; f(x) + \tfrac{1}{2}\rho_k\|x - x_k\|^2 + \tfrac{1}{2}\delta_k\|u + y_k\|^2 \quad \text{subject to} \;\; c(x) + \delta_k u = 0, \tag{6}$$

for a primal regularization parameter $\rho_k \geq 0$, where x_k is the current primal iterate.

The KKT conditions for (6),

$$\begin{bmatrix} g(x) + \rho_k(x - x_k) - J(x)^T y \\ \delta_k(u + y_k) - \delta_k y \\ c(x) + \delta_k u \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{7}$$

are therefore unconditionally necessary for optimality for any fixed value of $\rho_k \geq 0$ and $\delta_k > 0$. The following relationship between KKT points of (1) and those of (6) illustrates the fact that the primal-dual regularization is *exact*.

Theorem 1 *Suppose (x_k, u_k, y_k) is a KKT point of (6) for some $\rho_k \geq 0$ and $\delta_k > 0$. Then (x_k, y_k) is a KKT point of (1).*

Alternatively, suppose $\rho_k = 0$ and $(\bar{x}, \bar{u}, \bar{y})$ is a KKT point of (6) for some $\delta_k > 0$ and suppose $\bar{x}$ is feasible for (1). Then $\bar{u} = 0$, $\bar{y} = y_k$ and $(\bar{x}, \bar{y})$ is a KKT point of (1).

Conversely, suppose (x^*, y^*) is a KKT point of (1). Then $(x_k, u_k, y_k) := (x^*, 0, y^*)$ is a KKT point of (6) for any $\rho \geq 0$ and $\delta > 0$.

Proof. Immediate, by direct comparison of (3) and (7). □

Sequential quadratic programming methods for (6) may be interpreted as applying Newton's method to (7). A Newton-like step for (7) from (x_k, u_k, y_k) solves the linear system

$$\begin{bmatrix} H_k + \rho_k I & & -J_k^T \\ & \delta_k I & -\delta_k I \\ J_k & \delta_k I & \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta u \\ \Delta y \end{bmatrix} = - \begin{bmatrix} g_k - J_k^T y_k \\ \delta_k u_k \\ c_k + \delta_k u_k \end{bmatrix}, \tag{8}$$

where $g_k := g(x_k)$, $c_k := c(x_k)$, $J_k := J(x_k)$ and H_k is a symmetric approximation of $\nabla_{xx} L(x_k, y_k)$. The elimination of $\Delta u = -u_k + \Delta y$ yields the reduced system

$$\begin{bmatrix} H_k + \rho_k I & J_k^T \\ J_k & -\delta_k I \end{bmatrix} \begin{bmatrix} \Delta x \\ -\Delta y \end{bmatrix} = - \begin{bmatrix} g_k - J_k^T y_k \\ c_k \end{bmatrix}, \tag{9}$$

which is the familiar system encountered in stabilized SQP methods [45], while the system used in classical SQP methods corresponds to $\delta_k = 0$. For simplicity in the rest of this paper, the coefficient matrix of (9) is referred as K_k. The system (9) may be interpreted as the KKT conditions of the quadratic subproblem

$$\begin{aligned} &\underset{\Delta x, \Delta u}{\text{minimize}} && \nabla_x L(x_k, y_k)^T \Delta x + \tfrac{1}{2}\Delta x^T (H_k + \rho_k I)\Delta x + \tfrac{1}{2}\delta_k\|u_k + \Delta u\|^2 \\ &\text{subject to} && c_k + J_k \Delta x + \delta_k(u_k + \Delta u) = 0. \end{aligned} \tag{10}$$

Note that (10) itself always satisfies the LICQ and therefore infeasible subproblems never occur. The primal regularization term $\rho_k I$ may be interpreted as a convexifying term that encourages descent in an appropriate merit function.

Given a fixed $\bar{x} \in \mathbb{R}^n$, we define

$$\phi(x, y; \bar{x}, \rho, \delta) := f(x) - c(x)^T y + \tfrac{1}{2}\rho\|x - \bar{x}\|^2 + \tfrac{1}{2}\delta^{-1}\|c(x)\|^2. \tag{11}$$

For future reference, we note that

$$\nabla_x \phi(\bar{x}, y; \bar{x}, \rho, \delta) = \nabla_x L(\bar{x}, y) + \delta^{-1} J(\bar{x})^T c(\bar{x}) = g(\bar{x}) - J(\bar{x})^T (y - \delta^{-1} c(\bar{x})). \tag{12}$$

For simplicity of exposition, we write $\phi(x, y; \rho, \delta)$ instead of $\phi(x, y; x, \rho, \delta)$. We also let $w := (x, y)$ and define $F : \mathbb{R}^{n+m} \to \mathbb{R}^{n+m}$ as $F(w) := (\nabla_x L(w), c(x))$.

3 Main algorithm

Algorithm 1 is a simplification of [3, Algorithm 1] that ignores inequality constraints and allows symmetric Hessian approximations. In the description, we make use of the norm $\|F(w)\|_* := \|\nabla_x L(w)\| + \|c(x)\|$.

Algorithm 1 Outer iteration

1: Choose $\alpha \in (0, 1)$, $\theta \in (0, 1)$, and $\epsilon > 0$. Set $k = 0$.
2: If $\|F(w_k)\| < \epsilon$, terminate with final iterate w_k.
3: Choose a symmetric matrix H_k, $\rho_k \geq 0$, and $\delta_k \in [\min(\|F(w_k)\|, \alpha\delta_{k-1}), \delta_{k-1}]$. Compute a trial iterate w_k^+ as an approximate solution of

$$K_k(w_k^+ - w_k) + F(w_k) = 0. \tag{13}$$

4: Choose $\epsilon_k > 0$. If

$$\|F(w_k^+)\|_* \leq \theta\|F(w_k)\|_* + \epsilon_k, \tag{14}$$

then set $w_{k+1} = w_k^+$. Otherwise perform a sequence of inner iterations in order to find a new iterate w_{k+1} such that

$$\|F(w_{k+1})\|_* \leq \theta\|F(w_k)\|_* + \epsilon_k. \tag{15}$$

Increment k by one and return to Step 2.

The main idea of Algorithm 1 is to start each outer iteration with an extrapolation step (13). The extrapolation step is accepted if it achieves sufficient improvement in the first-order optimality residual. In the negative, an inner iteration procedure is started in order to identify an improved iterate.

A certain amount of flexibility is allowed in choosing the parameters ρ_k and δ_k at the beginning of each outer iteration. An important feature for global convergence is that it is allowed to keep δ_k fixed, at least after a certain number of iterations. An important feature for fast local convergence is that it is allowed to select $\delta_k = \|F(w_k)\|$ when close to an isolated minimizer.

Algorithm 2 describes the linesearch procedure used as the inner iteration, which is essentially a standard augmented Lagrangian subproblem solve in which steps are computed using the augmented form (16) followed by a Wolfe linesearch on the proximal augmented Lagrangian. During the inner iterations, y_k is kept fixed. The inner primal iterates and regularization parameter corresponding to the k-th outer iteration are denoted $x_{k,j}$, $\rho_{k,j}$ and $\delta_{k,j}$ for $j \geq 0$. The inner iterations stop as soon as the first-order optimality residual of (1) has improved in the sense of (14). As in the standard augmented Lagrangian, the penalty parameter is decreased when dual feasibility improved but primal feasibility lags behind.

In Step 3 of Algorithm 1 and Step 3 of Algorithm 2, the linear system could be solved exactly, but there is flexibility to compute an inexact solution. The inexactness in the outer iteration is a departure from the framework of [3].

In Algorithm 2, steplengths are computed so as to satisfy the Wolfe conditions. The reason for this requirement is that the global convergence analysis is based on a simple application of Zoutendijk's theorem [34, Theorem 3.2], which requires the Wolfe conditions. We believe it is possible to develop a global convergence analysis based solely on the Armijo condition, in the vein of [2].

Algorithm 2 Inner iteration

1: Set j to 0. Choose an initial guess $w_{k,0}$ and $\delta_{k,0} > 0$. Choose c_1 and c_2 such that $0 < c_1 < c_2 < 1$.

2: If $\|\nabla_x L(x_{k,j}, y_k - \delta_{k,j}^{-1} c(x_{k,j}))\| \le \theta \|\nabla_x L(x_{k,0}, y_k)\| + \frac{1}{2}\epsilon_k$, then

 if $\|c(x_{k,j})\| \le \theta \|c(x_{k,0})\| + \frac{1}{2}\epsilon_k$, stop with $w_{k+1} = (x_{k,j}, y_k - \delta_{k,j}^{-1} c(x_{k,j}))$,

 otherwise, set $\delta_{k,j} = \delta_{k,j-1}/10$.

 Go to Step 3.

3: Choose a symmetric matrix $H_{k,j}$ and $\rho_{k,j} \ge 0$. Compute Δx_j as an approximate solution to

$$\begin{bmatrix} H_{k,j} + \rho_{k,j} I & J_{k,j}^T \\ J_{k,j} & -\delta_{k,j} I \end{bmatrix} \begin{bmatrix} \Delta x_j \\ -\Delta y_j \end{bmatrix} = - \begin{bmatrix} g_{k,j} - J_{k,j}^T y_k \\ c_{k,j} \end{bmatrix}. \tag{16}$$

4: Set $x_{k,j+1} = x_{k,j} + \alpha_j \Delta x_j$, where α_j is obtained using a line search and satisfies the Wolfe conditions:

$$\phi(x_{k,j+1}, y_k; x_{k,j}, \rho_{k,j}, \delta_{k,j}) \le \phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j}) + c_1 \alpha_j \nabla \phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j})^T \Delta x_j$$
$$\nabla \phi(x_{k,j+1}, y_k; x_{k,j}, \rho_{k,j}, \delta_{k,j})^T \Delta x_j \ge c_2 \nabla \phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j})^T \Delta x_j.$$

 Increment j by one and return to Step 2.

The next section examines the global convergence properties of Algorithms 1 and 2 and, in particular, what conditions should be imposed on inexact steps.

4 Global convergence

In this section, we examine in turn the convergence of Algorithms 1 and 2.

4.1 Convergence of the inner iterations

The convergence of the inner iterations is divided into two parts depending on whether linear systems are solved exactly or not. In Section 4.1.1, we assume that the linear systems of Algorithms 1 and 2 are solved exactly. That situation covers the case where the systems are solved via a factorization, whether exact second derivatives are used or not. It also covers the case where the linear systems are solved iteratively but with an extremely tight tolerance, although that is not realistic in practice. In Section 4.1.2, we assume that systems are solved inexactly and we describe the iterative procedure that we use. The latter relies on H_k being positive definite and such that linear systems with coefficient H_k can be solved easily and cheaply. Such a situation occurs when H_k is a limited-memory BFGS approximation to the second derivatives, and that is our selection of choice. With that choice, H_k itself is never really needed and we simply maintain its inverse implicitly [33]. Other choices are of course possible, including $H_k = I$, or a positive-definite diagonal approximation of $\nabla^2_{xx} L(x_k, y_k)$.

In this section, k denotes the outer iteration index and appears everywhere to avoid ambiguity. We refer to the coefficient of the linear system at outer iteration k and inner iteration j in Step 3 of Algorithm 2 as $K_{k,j}$.

Our main working assumption is as follows.

Assumption 4.1 *The gradients $g_{k,j}$, the matrices $H_{k,j}$ and the matrices $J_{k,j}$ are uniformly bounded for all $j \in \mathbb{N}$. Moreover, $\rho_{k,j}$ is bounded for all $j \in \mathbb{N}$.*

Section 4.1.1 examines the situation where (16) is solved exactly. Because $H_{k,j}$ may be indefinite, the role of $\rho_{k,j}$ is to convexify locally so that $H_{k,j} + \rho_{k,j} I$ is positive definite on the nullspace of $J_{k,j}$. Our assumption that $\rho_{k,j}$ is bounded is reasonable in the sense that it amounts to assuming that the most negative eigenvalue of the reduction of $H_{k,j}$ to the nullspace of $J_{k,j}$ is bounded below.

4.1.1 Exact system solves

When $H_{k,j}$ is not positive definite, which is typically the case when exact second-derivatives are used, $\rho_{k,j}$ must be sufficiently large to ensure that Δx_j is a descent direction for the proximal augmented Lagrangian. Linear systems may be solved using a symmetric indefinite factorization such as the multifrontal implementation MA57 [14]. Because this factorization reveals the inertia of $K_{k,j}$, the regularization parameter $\rho_{k,j}$ can be increased until the correct inertia is detected [25]. Such a procedure is similar to that used in IPOPT [42]. Those observations motivate the following assumption.

Assumption 4.2 *The matrices $H_{k,j}+\rho_{k,j}I+\delta_{k,j}^{-1}J_{k,j}^TJ_{k,j}$ are uniformly positive definite and uniformly bounded for all $j \in \mathbb{N}$, i.e., there exist constants $\overline{\sigma} \geq \underline{\sigma} > 0$ such that for all $j \in \mathbb{N}$ and all $d \in \mathbb{R}^n$,*

$$\underline{\sigma}\|d\|^2 \leq d^T(H_{k,j} + \rho_{k,j}I + \delta_{k,j}^{-1}J_{k,j}^TJ_{k,j})d \leq \overline{\sigma}\|d\|^2.$$

Assumption 4.2 implies that $\{H_{k,j} + \rho_{k,j}I\}$ is uniformly positive definite over the nullspace of $J_{k,j}$ for all $j \in \mathbb{N}$.

Theorem 2 (Inner iteration, exact solves) *Suppose that Assumption 4.2 holds and that $\phi(\cdot, y_k; \rho_{k,j}, \delta_{k,j})$ is bounded below for all j. Then Algorithm 2 generates a sequence of iterates $x_{k,j}$ such that*

$$\lim_{j\to+\infty} \|\nabla_x\phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j})\| = 0.$$

Proof. We eliminate Δy_j from (16) and use (12) to obtain

$$\left(H_{k,j} + \rho_{k,j}I + \delta_{k,j}^{-1}J_{k,j}^TJ_{k,j}\right)\Delta x_j = -\nabla_x\phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j}). \tag{17}$$

We take the inner product of both sides of (17) with Δx_j and use Assumption 4.2, and obtain

$$-\nabla_x\phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j})^T\Delta x_j \geq \underline{\sigma}\|\Delta x_j\|^2. \tag{18}$$

Assumption 4.2 and (17) yield

$$\begin{aligned}\|\Delta x_j\| &\geq \lambda_{\min}\left((H_{k,j} + \rho_{k,j}I + \delta_{k,j}^{-1}J_{k,j}^TJ_{k,j})^{-1}\right)\|\nabla_x\phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j})\| \\ &= \left(\lambda_{\max}(H_{k,j} + \rho_{k,j}I + \delta_{k,j}^{-1}J_{k,j}^TJ_{k,j})\right)^{-1}\|\nabla_x\phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j})\| \\ &\geq \overline{\sigma}^{-1}\|\nabla_x\phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j})\|.\end{aligned}$$

Therefore

$$-\frac{\nabla_x\phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j})^T\Delta x_j}{\|\nabla_x\phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j})\|\,\|\Delta x\|} \geq \underline{\sigma}/\overline{\sigma} > 0.$$

Zoutendijk's theorem ensures that

$$\lim_{j\to\infty}\left(-\frac{\nabla_x\phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j})^T\Delta x}{\|\nabla\phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j})\|\|\Delta x\|}\right)^2\|\nabla\phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j})\|^2 = 0,$$

from which the desired result follows immediately. □

Theorem 2 implies that the first stopping condition of Algorithm 1 is satisfied after a finite number of iterations because (12) can also be written

$$\nabla_x\phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j}) = \nabla_x L(x_{k,j}, y_k - \delta_{k,j}^{-1}c(x_{k,j})).$$

In order to determine when the second stopping condition holds, we consider the following cases.

If $\{\delta_{k,j}\}_{j\in\mathbb{N}}$ is bounded away from zero, the mechanism of Algorithm 2 guarantees that the stopping condition $\|c(x_{k,j})\| \leq \theta\|c(x_{k,0})\| + \frac{1}{2}\epsilon_k$ is eventually satisfied.

If, on the contrary, there is an index set $\mathcal{J}$ such that $\lim_{j\in\mathcal{J}} \delta_{k,j} = 0$, the inner iterations may not terminate if $\|c(x_{k,j})\| > \theta\|c(x_{k,0})\| + \frac{1}{2}\epsilon_k$ for all j. Because of Theorem 2, we have

$$\begin{aligned} 0 &= \lim_{j\in\mathcal{J}} \delta_{k,j} \nabla_x \phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j}) \\ &= \lim_{j\in\mathcal{J}} \delta_{k,j} \left(g(x_{k,j}) - J(x_{k,j})^T y_k \right) + J(x_{k,j})^T c(x_{k,j}). \end{aligned}$$

Under Assumption 4.1, the first term of the previous left-hand side converges to zero, and therefore

$$\lim_{j\in\mathcal{J}} J(x_{k,j})^T c(x_{k,j}) = 0.$$

In other words, any limit point $\bar{x}_k$ of $\{x_{k,j}\}_{j\in\mathcal{J}}$ is stationary for the (typically underdetermined) least-squares problem

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \;\; \tfrac{1}{2}\|c(x)\|^2.$$

If $\bar{x}_k$ is feasible, the k-th set of inner iterations eventually terminates. If $\bar{x}_k$ is infeasible, we say that it is an infeasible stationary point for the infeasibility measure $\|c(x)\|$. Convergence to such a stationary point occurs, among other situations, when (1) is infeasible or locally infeasible.

4.1.2 Inexact system solves: a quasi-Newton strategy

A difficulty associated with the iterative solution of systems of the form (16), e.g., using MINRES [37], is the need to balance residuals associated to each block equation, as in [11]. The framework detailed in this section solves one of the block equations exactly while controlling the residual associated to the other. That is done by transforming (16) into the first-order optimality conditions of a preconditioned linear least-squares problem. The preconditioner used, $H_{k,j}^{-1}$ in the present case, must be applied at each iteration of an iterative method for least-squares problems. It is therefore crucial that $H_{k,j}^{-1}$ be positive definite and cheaply applicable.

In this section, we make the following assumption.

Assumption 4.3 *The matrices $H_{k,j}$ are uniformly positive definite and uniformly bounded for all $j \in \mathbb{N}$.*

For the above reasons, and in order to preserve hope for fast local convergence when close to an isolated minimizer, we set $H_{k,j}$ to a limited-memory BFGS approximation of the Hessian of the Lagrangian [33] that is possibly modified to take into account the fact that the Hessian of the Lagrangian cannot be expected to be positive definite. Details are not relevant here and will be given in Section 6. By construction, L-BFGS approximations are always positive definite. In addition, their inverse can be implicitly maintained and updated along the iterations, and can be applied to a vector cheaply using either the two-loop recursion or the compact storage format [10], or by maintaining the approximation in factored form [13, Algorithm A9.4.2].

With such a choice, $K_{k,j}$ is SQD, and therefore nonsingular irrespective of the rank of $J_{k,j}$. We always set $\rho_{k,j} = 0$ when $H_{k,j}$ is an L-BFGS approximation.

We first cast (16) as a least-square problem. We introduce $\Delta\bar{y} := \Delta y + \delta_{k,j}^{-1} c_{k,j}$, and rewrite (16) equivalently as

$$\begin{bmatrix} H_{k,j} & J_{k,j}^T \\ J_{k,j} & -\delta_{k,j} I \end{bmatrix} \begin{bmatrix} \Delta x \\ -\Delta\bar{y} \end{bmatrix} = \begin{bmatrix} b_{k,j} \\ 0 \end{bmatrix} \tag{19}$$

where $b_{k,j} = -g_{k,j} + J_{k,j}^T(y_k - \delta_{k,j}^{-1} c_{k,j}) = -\nabla_x \phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j})$. In [7] a similar system is obtained from linear algebra transformations of the Newton equations $\nabla_{xx}^2 \phi(x, y; \rho) d = -\nabla_x \phi(x, y; \rho)$.

The shifted system (19) can be seen as the necessary and sufficient optimality conditions of the regularized and preconditioned least-squares problem

$$\underset{\Delta\bar{y}}{\text{minimize}} \;\; \tfrac{1}{2}\|J_{k,j}^T \Delta\bar{y} + b_{k,j}\|_{H_{k,j}^{-1}}^2 + \tfrac{1}{2}\|\Delta\bar{y}\|_{\delta_{k,j} I}^2. \tag{20}$$

The latter can be solved approximately using a stopping criterion based exclusively on the residual of the second block equation of (19), i.e., the optimality residual of (20):

$$r_{k,j}(\Delta\bar{y}) := J_{k,j}\Delta x + \delta_{k,j}\Delta\bar{y}, \tag{21}$$

where $\Delta x := H_{k,j}^{-1}(J_{k,j}^T\Delta\bar{y} + b_{k,j})$ is the least-squares residual.

Our choice is to solve (20) with the LSMR method [17] modified as recommended in [36] to accommodate non-Euclidean norms. This choice is motivated by the fact that we wish to reduce the residual of (19), or equivalently, of (16), below a certain threshold. The least-squares interpretation guarantees that the first block equation is always satisfied exactly, by definition of the least-squares residual. The residual of the second block equation is precisely $r_{k,j}(\Delta\bar{y})$. An important property of LSMR, that is not shared by other methods such as LSQR [38], is that it decreases an appropriate norm of the optimality residual of (20), i.e., of $r_{k,j}(\Delta\bar{y})$, monotonically. Finally, using LSMR as described above can save half of the iterations as compared to using MINRES with the natural preconditioner $\text{blkdiag}(H_{k,j},\, \delta_{k,j}I)$ applied to (19) [36].

Our implementation requires that two termination tests be satisfied. The first guarantees sufficient descent.

Termination Test 1 *A step $(\Delta x, \Delta\bar{y})$ is an acceptable inexact solution of (19) if*

$$\|r_{k,j}(\Delta\bar{y})\|^2_{\delta_{k,j}^{-1}I} + \gamma_j\|b_{k,j}\|^2_{H_{k,j}^{-1}} \le \|J_{k,j}^T\Delta\bar{y} + b_{k,j}\|^2_{H_{k,j}^{-1}} + \|\Delta\bar{y}\|^2_{\delta_{k,j}I}, \tag{22}$$

for some $\gamma_j > 0$.

We defer comments on Termination Test 1 until the end of this section.

Lemma 1 *Let Assumptions 4.1 and 4.3 be satisfied. Suppose that Termination Test 1 is satisfied. Then*

$$-\nabla_x\phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j})^T\Delta x \ge \bar{\gamma}_j\,\|\nabla_x\phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j})\|^2,$$

where $\bar{\gamma}_j := \frac{1}{2}\gamma_j/\lambda_{\max}(H_{k,j})$.

Proof. Let us denote $\phi_{k,j} := \phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j})$ and $M_{k,j} := H_{k,j} + \delta_{k,j}^{-1}J_{k,j}^T J_{k,j}$ for conciseness.

The first block equation of (19) and (21) yield

$$\begin{aligned} -\nabla_x\phi_{k,j}^T\Delta x &= \Delta x^T H_{k,j}\Delta x - \Delta\bar{y}^T J_{k,j}\Delta x \\ &= \Delta x^T H_{k,j}\Delta x + \delta_{k,j}\Delta\bar{y}^T\Delta\bar{y} - \Delta\bar{y}^T r_{k,j}(\Delta\bar{y}). \end{aligned} \tag{23}$$

Isolating $\Delta\bar{y}$ in the second block equation of (19), substituting it into the first one and using (21) gives

$$-\nabla_x\phi_{k,j} = M_{k,j}\Delta x - \delta_{k,j}^{-1}J_{k,j}^T r_{k,j}(\Delta\bar{y}). \tag{24}$$

Thus

$$-\nabla_x\phi_{k,j}^T\Delta x = \Delta x^T M_{k,j}\Delta x - \delta_{k,j}^{-1}\Delta x^T J_{k,j}^T r_{k,j}(\Delta\bar{y}). \tag{25}$$

Adding (23) and (25) together, dividing by 2 and noting that the norm of the residual $r_{k,j}(\Delta\bar{y})$ can be expressed as

$$\|J_{k,j}\Delta x + \delta_{k,j}\Delta\bar{y}\|^2 := r_{k,j}(\Delta\bar{y})^T r_{k,j}(\Delta\bar{y}) = \Delta x^T J_{k,j}^T r_{k,j}(\Delta\bar{y}) + \delta_{k,j}\Delta\bar{y}^T r_{k,j}(\Delta\bar{y}), \tag{26}$$

leads to

$$\begin{aligned} -\nabla_x\phi_{k,j}^T\Delta x &= \tfrac{1}{2}\Delta x^T M_{k,j}\Delta x + \tfrac{1}{2}\Delta x^T H_{k,j}\Delta x + \tfrac{1}{2}\|\Delta\bar{y}\|^2_{\delta_{k,j}I} - \tfrac{1}{2}\|r_{k,j}\|^2_{\delta_{k,j}^{-1}I} \\ &= \tfrac{1}{2}\Delta x^T M_{k,j}\Delta x + \tfrac{1}{2}\|J_{k,j}^T\Delta\bar{y} + b_{k,j}\|^2_{H_{k,j}^{-1}} + \tfrac{1}{2}\|\Delta\bar{y}\|^2_{\delta_{k,j}I} - \tfrac{1}{2}\|r_{k,j}\|^2_{\delta_{k,j}^{-1}I} \\ &\ge \tfrac{1}{2}\|J_{k,j}^T\Delta\bar{y} + b_{k,j}\|^2_{H_{k,j}^{-1}} + \tfrac{1}{2}\|\Delta\bar{y}\|^2_{\delta_{k,j}I} - \tfrac{1}{2}\|r_{k,j}\|^2_{\delta_{k,j}^{-1}I} \\ &\ge \tfrac{1}{2}\|J_{k,j}^T\Delta\bar{y} + b_{k,j}\|^2_{H_{k,j}^{-1}} + \tfrac{1}{2}\|\Delta\bar{y}\|^2_{\delta_{k,j}I} - \tfrac{1}{2}\|r_{k,j}\|^2_{\delta_{k,j}^{-1}I}, \end{aligned}$$

where we denoted $r_{k,j} := r_{k,j}(\Delta\bar{y})$ for brevity. Using (22) yields $-\nabla_x\phi_{k,j}^T\Delta x \ge \frac{1}{2}\gamma_j\,\|\nabla_x\phi_{k,j}\|^2_{H_{k,j}^{-1}} \ge \bar{\gamma}_j\,\|\nabla_x\phi_{k,j}\|^2$. □

The second termination test requires sufficient decrease in the optimality residual of (20) with a specific form of the relative tolerance.

Termination Test 2 *Let $\mu > 0$ and $0 \le \beta_2 \le 1$ be given constants. A step $(\Delta x, \Delta \bar{y})$ is an acceptable inexact solution of (19) if*

$$\|r_{k,j}(\Delta \bar{y})\|_{\delta_{k,j}^{-1} I} \le \mu \min(1, \delta_{k,j}^{\beta_2}) \|b_{k,j}\|_{H_{k,j}^{-1}}. \tag{27}$$

Termination Test 2 leads directly to convergence of the inner iterations.

Theorem 3 (Inner iteration, inexact solves) *Suppose that Assumptions 4.1 and 4.3 hold, that Termination Tests 1 and 2 are satisfied, and that there exists $\gamma > 0$ such that $\gamma_j \ge \gamma$ for all $j \in \mathbb{N}$ in Termination Test 2. Then Algorithm 2 generates a sequence of iterates $x_{k,j}$ such that*

$$\lim_{j \to \infty} \|\nabla_x \phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j})\| = 0.$$

Proof. We use the shorthands $\phi_{k,j} := \phi(x_{k,j}, y_k; \rho_{k,j}, \delta_{k,j})$ and $M_{k,j} := H_{k,j} + \delta_{k,j}^{-1} J_{k,j}^T J_{k,j}$ for conciseness. Assumption 4.3 implies that there exists $\kappa_H > 0$ such that $\|b_{k,j}\|_{H_{k,j}^{-1}} \le \kappa_H \|b_{k,j}\|$ and $\kappa_M > 0$ such that $\|M_{k,j}^{-1}\| \le \kappa_M$ for all j. From (24) and Termination Test 2, we have

$$\begin{aligned}
\|\Delta x\| &\le \|M_{k,j}^{-1}\| \, \| - \nabla_x \phi_{k,j} + \delta_{k,j}^{-1} J_{k,j}^T r_{k,j}(\Delta \bar{y})\| \\
&\le \kappa_M \left(\|\nabla_x \phi_{k,j}\| + \delta_{k,j}^{-1} \|J_{k,j}^T r_{k,j}(\Delta \bar{y})\| \right) \\
&\le \kappa_M \left(\|\nabla_x \phi_{k,j}\| + \|J_{k,j}^T\| \, \|r_{k,j}(\Delta \bar{y})\|_{\delta_{k,j}^{-1} I} \right) \\
&\le \kappa_M \left(\|\nabla_x \phi_{k,j}\| + \mu \min(1, \delta_{k,j}^{\beta_2}) \sigma_{\max}(J_{k,j}) \kappa_H \|\nabla_x \phi_{k,j}\| \right) \\
&\le \kappa \, \|\nabla_x \phi_{k,j}\|
\end{aligned} \tag{28}$$

with $\kappa := \sup_j \, \kappa_M (1 + \mu \sigma_{\max}(J_{k,j}) \kappa_H) > 0$.

Lemma 1 and (28) together imply

$$-\frac{\nabla_x \phi_{k,j}^T \Delta x}{\|\nabla_x \phi_{k,j}\| \, \|\Delta x\|} \ge \frac{\bar{\gamma}_j}{\kappa} = \frac{\gamma_j}{\kappa \, \lambda_{\max}(H_{k,j})} \ge \frac{\gamma}{\kappa \, \sup_j \lambda_{\max}(H_{k,j})} > 0,$$

where the last inequality follows from Assumption 4.3. At this point, we are in position to apply Zoutendijk's theorem and conclude as in the proof of Theorem 2. □

In view of Theorem 3, it is easier to provide an interpretation of Termination Test 1. At outer iteration k and inner iteration j, the right-hand side of (22) is the objective value of (20), which is positive unless $b_{k,j} = 0$, i.e., unless $x_{k,j}$ is first-order stationary for the augmented Lagrangian. If $b_{k,j} = 0$, the only solution of (19) is $\Delta x = 0$ and $\Delta \bar{y} = 0$, which is the initial LSMR iterate. In that case, (21) yields $r_{k,j} = 0$, so that (22) is satisfied at the first iteration. Let us consider now the case $b_{k,j} \ne 0$. The first term in the left-hand side of (22) is the norm of (21), i.e., the optimality residual of (20), which decreases monotonically to zero along the LSMR iterations. The role of the sequence $\{\gamma_j\}$ is to allow sufficient room for satisfaction of (22) even when the value of the right-hand side is small. In particular, if $\Delta \bar{y}$ were the exact solution of (20), $r_{k,j}(\Delta \bar{y}) = 0$ and γ_j should be chosen so that

$$\gamma_j \le \frac{\|J_{k,j}^T \Delta \bar{y} + b_{k,j}\|_{H_{k,j}^{-1}}^2 + \|\Delta \bar{y}\|_{\delta_{k,j} I}^2}{\|b_{k,j}\|_{H_{k,j}^{-1}}^2} \le 1.$$

The discussion following Theorem 2 also applies in the context of the present section.

4.2 Convergence of the outer iterations

We now analyze the global convergence of the outer iterations. In this section, we assume that Algorithm 2 succeeds in computing a new iterate w_{k+1} that satisfies (15) each time it is called at Step 4 of Algorithm 1. The following corollary results immediately from Theorems 2 and 3.

Corollary 1 *Assume Algorithm 2 succeeds each time it is called at Step 4 of Algorithm 1. Then Algorithm 1 generates iterates w_k such that*

$$\|F(w_{k+1})\|_* \leq \theta\|F(w_k)\|_* + \epsilon_k.$$

The following result, which is a direct consequence of [3, Theorem 3.3], describes the behavior of the sequence of outer iterates.

Theorem 4 (Outer iteration) *Assume Algorithm 2 succeeds each time it is called at Step 4 of Algorithm 1 and that the sequence $\{\epsilon_k\}$ converges to zero. Then Algorithm 1 generates a sequence of iterates w_k such that $\{F(w_k)\}$ converges to zero.*

Proof. Let $\ell := \limsup_{k\to\infty} \|F(w_k)\|_*$. Taking the limit superior in (15) yields

$$\ell \leq \theta\ell + \limsup_{k\to+\infty} \epsilon_k.$$

Because $\lim_{k\to\infty} \epsilon_k = \limsup_{k\to+\infty} \epsilon_k = 0$, we have $\ell \leq \theta\ell$, i.e., $\ell = 0$, which means that $\{F(w_k)\}$ converges to zero. □

5 Local convergence

In this section, we analyze the asymptotic behavior of $\{w_k\}$ under the assumption that it converges to a stationary point satisfying certain assumptions. We establish that the rate of convergence of $\{w_k\}$ is Q-superlinear provided δ_k asymptotically approaches zero sufficiently fast, which is ensured by selecting $\delta_k = \|F(w_k)\|$ in Algorithm 1. This last requirement is a departure from standard augmented Lagrangian methods and ensures the transition to a stabilized SQP method in the local regime.

The analysis in this section broadly follows that of [3] and [45]. The results differ in two important respects. Firstly, we allow quasi-Newton approximations to second-order derivatives. Secondly, we also allow inexact solutions to the extrapolation linear system (13).

Assume x^* is a stationary point of (1) that satisfies (3). We use $\mathcal{Y}$ to denote the set of Lagrange multipliers associated to x^*, i.e.,

$$\mathcal{Y} = \{y^* \in \mathbb{R}^m \mid (x^*, y^*) \text{ satisfies the KKT conditions (3)}\}.$$

We also use the notation $\mathcal{S} := \{x^*\} \times \mathcal{Y}$.

Because $\nabla_x L(x^*, \cdot)$ is linear, $\mathcal{Y}$ is a closed convex set. It is well known that $\mathcal{Y}$ is a singleton under the assumption that $J(x^*)$ has full row rank and may be unbounded or empty if that assumption fails to hold [20].

Our working assumptions for this section are as follows.

Assumption 5.1 *The sequence $\{w_k\}$ generated by Algorithm 1 converges to $w^* = (x^*, y^*)$ for a certain $y^* \in \mathcal{Y}$.*

Assumption 5.2 *The functions f and c are twice continuously differentiable with locally Lipschitz second derivatives on $\mathbb{R}^n$.*

Assumption 5.3 *δ_k is chosen as $\|F(w_k)\|$ at Step 3 of Algorithm 1.*

Assumption 5.4 *H_k is uniformly bounded, i.e., there exists $\kappa > 0$ such that $\|H_k\| \leq \kappa$ for all $k = 0, 1, \ldots$.*

When (13) is solved inexactly, Assumption 4.3 from the global regime is sufficient for our purposes. Whether (13) is solved exactly or not, we establish fast local convergence under the following, weaker, assumption.

Assumption 5.5 *The approximation H_k is sufficiently positive definite on the nullspace of $J(x^*)$ for all sufficiently large k, i.e., there exists $\eta > 0$, such that $z^T H_k z \geq \eta \|z\|^2$ for all $z \in \mathrm{Null}(J(x^*))$.*

Assumption 5.5 implies that we may set $\rho_k = 0$ for all sufficiently large k.

The following assumption states that H_k is an increasingly accurate approximation of $\nabla^2_{xx} L(x^*, y^*)$ along Δx.

Assumption 5.6 *There exists $0 < \beta_1 \leq 1$ such that*

$$\|(H_k - \nabla^2_{xx} L(x^*, y^*))\Delta x\| = O(\|\Delta x\|^{1+\beta_1})$$

for all sufficiently large k, where Δx is computed from (13).

Assumption 5.6 is reminiscent of the Dennis-Moré condition [12] in unconstrained optimization but is more demanding. Though it is unlikely that any quasi-Newton approximation satisfies Assumption 5.6 in the strong form given, the analysis below illustrates how the assumption allows us to specify the precise convergence rate of the sequence of iterates. It is more likely that a quasi-Newton approximation satisfies

$$(H_k - \nabla^2_{xx} L(x^*, y^*))\Delta x = o(\Delta x) \tag{29}$$

instead. As we comment at the end of the present section, the local convergence analysis holds under that weaker assumption, except that the exact convergence rate cannot be specified.

The global convergence analysis does not depend on whether, how, or how accurately we solve (13). The local analysis, however, depends on (13) crucially. In this section, we specify how accurate the step computation should be. Note that (13) can be shifted to least-squares form exactly as (19):

$$\begin{bmatrix} H_k & J_k^T \\ J_k & -\delta_k I \end{bmatrix} \begin{bmatrix} \Delta x \\ -\Delta \bar{y} \end{bmatrix} = \begin{bmatrix} b_k \\ 0 \end{bmatrix}, \tag{30}$$

where $\Delta \bar{y} := \Delta y + \delta_{k,j}^{-1} c_{k,j}$ and $b_k = -\nabla_x \phi(x_k, y_k; 0, \delta_k)$. We use Termination Test 2 as our stopping condition. We repeat it here without mention of the index j.

Termination Test 3 *Let $\mu > 0$ and $0 \leq \beta_2 \leq 1$ be given constants. A step $(\Delta x, \Delta \bar{y})$ computed in an inexact solve of (30) at Step 3 of Algorithm 1 is acceptable if*

$$\|r_k\|_{\delta_k^{-1} I} \leq \mu \min(1, \delta_k^{\beta_2}) \|b_k\|_{H_k^{-1}}. \tag{31}$$

According to Theorem 4 and Assumption 5.3, $\delta_k \to 0$ and for all sufficiently large k, the term $\delta_k^{\beta_2}$ achieves the minimum appearing in (31).

For any $\epsilon > 0$, we define

$$\mathcal{N}(\epsilon) := \{(x, y) \mid \text{there exists } \bar{y} \in \mathcal{Y} \text{ such that } \|(x, y) - (x^*, \bar{y})\| \leq \epsilon\}.$$

We denote by P the projection onto $\mathcal{Y}$, i.e.,

$$P(y) := \arg\min\{\|y - \bar{y}\| \mid \bar{y} \in \mathcal{Y}\},$$

which is well defined because $\mathcal{Y}$ is closed and convex. Finally, the Euclidean distance from (x, y) to $\mathcal{S}$ is denoted

$$\mathrm{dist}((x, y), \mathcal{S}) := \inf\{\|(x, y) - (x^*, \bar{y})\| \mid \bar{y} \in \mathcal{Y}\} = \|(x, y) - (x^*, P(y))\|.$$

For any $w = (x, y) \in \mathcal{N}(\epsilon)$, we use the notation δ to denote $\|F(w)\|$, in accordance with Assumption 5.3.

Lemma 2 *Suppose that Assumptions 5.2 and 5.3 hold. Then there exists a constant* $\epsilon > 0$ *such that for all* $(x, y) \in \mathcal{N}(\epsilon)$ *we have* $\operatorname{dist}((x, y), \mathcal{S}) = \Omega(\delta)$.

Proof. Let $\epsilon > 0$ be arbitrary and $(x, y) \in \mathcal{N}(\epsilon)$. It follows from (3) and Assumption 5.2 that

$$\|\nabla_x L(x, y)\| = \|\nabla_x L(x, y) - \nabla_x L(x^*, P(y))\| = O(\operatorname{dist}((x, y), \mathcal{S})).$$

Similarly,

$$\|c(x)\| = \|c(x) - c(x^*)\| = O(\|x - x^*\|) = O(\operatorname{dist}((x, y), \mathcal{S})).$$

Thus $\delta = O(\operatorname{dist}((x, y), \mathcal{S}))$. □

Wright [45] establishes the converse of Lemma 2 under the Mangasarian and Fromovitz constraint qualification condition, which, in the case of equality constraints, amounts to the linear independence constraint qualification condition. The authors of [30] establish a similar result without assuming a constraint qualification but by restricting attention to a neighborhood of (x^*, y^*). We include the proof for completeness. We denote $\mathcal{B}_\epsilon(x^*, y^*)$ the ball centered at (x^*, y^*) of radius $\epsilon > 0$.

Lemma 3 *Suppose that Assumptions 5.2, 5.3 and 5.5 hold. Then there exists* $\epsilon > 0$ *such that for all* $(x, y) \in \mathcal{B}_\epsilon(x^*, y^*)$*, we have* $\operatorname{dist}((x, y), \mathcal{S}) = O(\delta)$.

Proof. By contradiction, suppose that for any $\epsilon > 0$, there exists $(x, y) \in \mathcal{B}_\epsilon(x^*, y^*)$ such that $\delta = o(\|x - x^*\|)$ and $\delta = o(\|y - y^*\|)$. By selecting a sequence $\{\epsilon_k\} \to 0$, we determine sequences $\{x_k\} \to x^*$ and $\{y_k\} \to y^*$ such that $\delta_k = o(\|x_k - x^*\|)$ and $\delta_k = o(\|y_k - y^*\|)$.

With the purpose of deriving a contradiction with Assumptions 5.2 and 5.5, we use (3) and the fact that $\nabla_x L(x_k, y_k) = O(\delta_k) = o(\|x_k - x^*\|)$ by assumption to deduce

$$\begin{aligned}
\nabla^2_{xx} L(x^*, y^*)(x_k - x^*) &= \nabla_x L(x_k, y^*) - \nabla_x L(x^*, y^*) + o(\|x_k - x^*\|) \\
&= \nabla_x L(x_k, y_k) - J(x_k)^T (y_k - y^*) + o(\|x_k - x^*\|) \\
&= -J(x_k)^T (y_k - y^*) + o(\|x_k - x^*\|) \\
&= -J(x^*)^T (y_k - y^*) \\
&\quad - (J(x_k) - J(x^*))^T (y_k - y^*) + o(\|x_k - x^*\|) \\
&= -J(x^*)^T (y_k - y^*) + o(\|x_k - x^*\|).
\end{aligned} \tag{32}$$

Similarly, our contradiction assumption gives

$$J(x^*)(x_k - x^*) = c(x_k) - c(x^*) + o(\|x_k - x^*\|) = o(\|x_k - x^*\|). \tag{33}$$

Reducing to a subsequence if necessary, there exists a vector z with $\|z\| = 1$ such that $\{(x_k - x^*)/\|x_k - x^*\|\} \to z$. We take limits in (32) and (33), and obtain

$$\nabla^2_{xx} L(x^*, y^*) z \in \operatorname{Range}(J(x^*)^T) \quad \text{and} \quad z \in \operatorname{Null}(J(x^*)),$$

which contradicts Assumption 5.5. Thus there exists $\epsilon > 0$ such that for all $(x, y) \in \mathcal{B}_\epsilon(x^*, y^*)$, we have $\|x - x^*\| = O(\delta)$.

Let $(x, y) \in \mathcal{B}_\epsilon(x^*, y^*)$. The linear system $J(x^*)^T (y - \tilde{y}) = \nabla_x L(x^*, y)$ in the unknown $\tilde{y}$ possesses at least the solution y^*, and all solutions $\tilde{y}$ are in $\mathcal{Y}$. In particular, Hoffman's lemma (see, e.g., [44, Lemma A.3]), implies that there exists a solution $\tilde{y} \in \mathcal{Y}$ such that $y - \tilde{y} = O(\|\nabla_x L(x^*, y)\|)$. Thus,

$$\begin{aligned}
\|y - P(y)\| &\leq \|y - \tilde{y}\| \\
&= O(\|\nabla_x L(x^*, y)\|) \\
&= O(\|\nabla_x L(x, y)\|) + O(\|\nabla_x L(x, y) - \nabla_x L(x^*, y)\|) \\
&= O(\delta) + O(\|x - x^*\|) \\
&= O(\delta),
\end{aligned}$$

where we used the first part of the proof. Finally, we have $\|x - x^*\| = O(\delta)$ and $\|y - P(y)\| = O(\delta)$, which concludes the proof. □

Lemmas 2 and 3 combine with Assumption 5.1 to yield the following corollary.

Corollary 2 *Suppose that Assumptions 5.1 to 5.3 and 5.5 hold. Then, for all sufficiently large k,* $\operatorname{dist}((x_k, y_k), \mathcal{S}) = \Theta(\delta_k)$.

Let $\bar{m}$ be the rank of $J(x^*)^T$, with $0 \le \bar{m} \le m$. The singular value decomposition of $J(x^*)^T$ may be written as

$$J(x^*)^T = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} \Sigma & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix}, \tag{34}$$

where Σ is a diagonal matrix containing the $\bar{m}$ nonzero singular values, U_1 is $n \times \bar{m}$, U_2 is $n \times (n - \bar{m})$, V_1 is $m \times \bar{m}$ and V_2 is $m \times (m - \bar{m})$. Note that $\begin{bmatrix} U_1 & U_2 \end{bmatrix}$ and $\begin{bmatrix} V_1 & V_2 \end{bmatrix}$ are orthogonal and that the columns of U_2 constitute an orthonormal basis for the nullspace of $J(x^*)$. The next result follows [45, Theorem 3.2].

Theorem 5 *Suppose that Assumptions 5.1 to 5.5 hold. Suppose that the approximate solution $(\Delta x, -\Delta y)$ of (13) satisfies Termination Test 3. Then, for all sufficiently large k,*

$$\Delta x = O(\delta_k) \quad \text{and} \quad \Delta y = O(\delta_k^{\beta_2}).$$

Proof. If we decompose $\Delta x = U_1 \tilde{x}_{U_1} + U_2 \tilde{x}_{U_2}$ and $\Delta y = V_1 \tilde{y}_{V_1} + V_2 \tilde{y}_{V_2}$, we may rewrite (13) as

$$\begin{bmatrix} U_1^T H_k U_1 & U_1^T H_k U_2 & U_1^T J_k^T V_1 & U_1^T J_k^T V_2 \\ U_2^T H_k U_1 & U_2^T H_k U_2 & U_2^T J_k^T V_1 & U_2^T J_k^T V_2 \\ V_1^T J_k U_1 & V_1^T J_k U_2 & -\delta_k I & 0 \\ V_2^T J_k U_1 & V_2^T J_k U_2 & 0 & -\delta_k I \end{bmatrix} \begin{bmatrix} \tilde{x}_{U_1} \\ \tilde{x}_{U_2} \\ -\tilde{y}_{V_1} \\ -\tilde{y}_{V_2} \end{bmatrix} = - \begin{bmatrix} s_{U_1} \\ s_{U_2} \\ s_{V_1} \\ s_{V_2} \end{bmatrix}, \tag{35}$$

where

$$\begin{bmatrix} s_{U_1} \\ s_{U_2} \\ s_{V_1} \\ s_{V_2} \end{bmatrix} = \begin{bmatrix} U_1^T (g_k - J_k^T y_k) \\ U_2^T (g_K - J_k^T y_k) \\ V_1^T (c_k - r_k) \\ V_2^T (c_k - r_k), \end{bmatrix}$$

and r_k satisfies (31). According to Assumption 5.2 and Lemma 3, $J_k - J(x^*) = O(\|x_k - x^*\|) = O(\delta_k)$, so that (34) yields $U_1^T J_k^T V_1 = \Sigma + O(\delta_k)$, $U_1^T J_k^T V_2 = O(\delta_k)$, $U_2^T J_k^T V_1 = O(\delta_k)$, and $U_2^T J_k^T V_2 = O(\delta_k)$. We substitute those estimates into (35) and obtain

$$\begin{bmatrix} U_1^T H_k U_1 & U_1^T H_k U_2 & \Sigma + O(\delta_k) & O(\delta_k) \\ U_2^T H_k U_1 & U_2^T H_k U_2 & O(\delta_k) & O(\delta_k) \\ \Sigma + O(\delta_k) & O(\delta_k) & -\delta_k I & 0 \\ O(\delta_k) & O(\delta_k) & 0 & -\delta_k I \end{bmatrix} \begin{bmatrix} \tilde{x}_{U_1} \\ \tilde{x}_{U_2} \\ -\tilde{y}_{V_1} \\ -\tilde{y}_{V_2} \end{bmatrix} = - \begin{bmatrix} s_{U_1} \\ s_{U_2} \\ s_{V_1} \\ s_{V_2} \end{bmatrix}.$$

After eliminating $\tilde{y}_{V_2} = -\delta_k^{-1} s_{V_2} + O(\|\tilde{x}_{U_1}\|) + O(\|\tilde{x}_{U_2}\|)$, there remains

$$(M_k + O(\delta_k)) \begin{bmatrix} -\tilde{y}_{V_1} \\ \tilde{x}_{U_2} \\ \tilde{x}_{U_1} \end{bmatrix} = - \begin{bmatrix} s_{U_1} + O(\|s_{V_2}\|) \\ s_{U_2} + O(\|s_{V_2}\|) \\ s_{V_1} \end{bmatrix}, \tag{36}$$

where

$$M_k := \begin{bmatrix} \Sigma & U_1^T H_k U_2 & U_1^T H_k U_1 \\ 0 & U_2^T H_k U_2 & U_2^T H_k U_1 \\ 0 & 0 & \Sigma \end{bmatrix}.$$

Assumption 5.4 and the orthogonality of U ensure that the blocks involving H_k are bounded above by κ. Thus, M_k is uniformly bounded. In addition, M_k is uniformly nonsingular because Σ is nonsingular and Assumption 5.5 ensures that $U_2^T H_k U_2$ is uniformly positive definite for all sufficiently large k. Thus for all sufficiently large k, $M_k + O(\delta_k)$ is also uniformly nonsingular. Then according to (36)

$$\|(\tilde{y}_{V_1}, \tilde{x}_{U_2}, \tilde{x}_{U_1})\| = O(\|(s_{U_1}, s_{U_2}, s_{V_1}, s_{V_2})\|),$$

and

$$\|\tilde{y}_{V_2}\| = O(\delta_k^{-1})\|s_{V_2}\| + O(\|(s_{U_1}, s_{U_2}, s_{V_1}, s_{V_2})\|).$$

From the right-hand side of (36), we have

$$\|(s_{U_1}, s_{U_2})\| = \|g_k - J_k^T y_k\| = \|\nabla_x L(x_k, y_k)\| = O(\delta_k),$$

and Termination Test 3 implies that

$$\begin{aligned}\|s_{V_1}\| &= \|V_1^T(c_k - r_k)\| \le \|c_k - r_k\| \le \|c(x_k) - c(x^*)\| + \|r_k\| \\ &= O(\|x_k - x^*\|) + O(\delta_k^{1+\beta_2}) = O(\delta_k).\end{aligned}$$

In addition,

$$s_{V_2} = V_2^T(c_k - r_k) = V_2^T\left(c(x^*) + J(x^*)(x_k - x^*) + O(\|x_k - x^*\|^2) - r_k\right).$$

Because $c(x^*) = 0$ and $V_2^T J(x^*) = 0$, there remains

$$\|s_{V_2}\| = O(\|x_k - x^*\|^2) + \|r_k\| = O(\delta_k^2) + O(\delta_k^{1+\beta_2}) = O(\delta_k^{1+\beta_2}).$$

Therefore, $\tilde{y}_{V_2} = O(\delta_k^{\beta_2})$. We have established that $\Delta x = U_1\tilde{x}_{U_1} + U_2\tilde{x}_{U_2} = O(\delta_k)$ and $\Delta y = V_1\tilde{y}_{V_1} + V_2\tilde{y}_{V_2} = O(\delta_k^{\beta_2})$. □

Note that Theorem 5 holds even if $\beta_2 = 0$ in Termination Test 3. However, in order to establish superlinear convergence, we need to be more demanding on the accuracy of the step computation.

Theorem 6 *Suppose that Assumptions 5.1 to 5.6 hold. Suppose that the approximate solution* $(\Delta x, -\Delta y)$ *of* (9) *satisfies Termination Test 3 with* $\beta_2 > 0$. *Then, for all sufficiently large* k,

$$\delta_k^+ := \delta(x_k^+, y_k^+) = \delta(x_k + \Delta x, y_k + \Delta y) = O(\delta_k^{1+\beta}).$$

where $0 < \beta = \min(\beta_1, \beta_2) \le 1$.

Proof. A straightforward Taylor expansion and the linearity of $\nabla_x L(x, \cdot)$ yield, for all sufficiently large k,

$$\begin{aligned}\nabla_x L(x_k + \Delta x, y_k + \Delta y) &= \nabla_x L(x_k, y_k) + \nabla_{xx}^2 L(x_k, y_k)\Delta x - J(x_k)^T\Delta y + O(\|\Delta x\|^2) \\ &= \left(\nabla_{xx}^2 L(x_k, y_k) - H_k\right)\Delta x \\ &\quad + H_k\Delta x + \nabla_x L(x_k, y_k) - J(x_k)^T\Delta y + O(\|\Delta x\|^2) \\ &= \left(\nabla_{xx}^2 L(x^*, y^*) - H_k\right)\Delta x + O(\delta_k^2) \\ &= O(\delta_k^{1+\beta_1}),\end{aligned}$$

where we used the first block equation of (13), Assumption 5.6 and Theorem 5.

Similarly, for all sufficiently large k,

$$\begin{aligned}c(x_k + \Delta x) &= c(x_k) + J(x_k)\Delta x + O(\|\Delta x\|^2) \\ &= r_k - \delta_k\Delta y + O(\|\Delta x\|^2) \\ &= O(\delta_k^{1+\beta_2}) + O(\delta_k^2) \\ &= O(\delta_k^{1+\beta_2}),\end{aligned}$$

where we used the second block equation of (13), Termination Test 3 and Theorem 5. The result holds with $\beta := \min(\beta_1, \beta_2) > 0$. □

The following corollary states that, asymptotically, no inner iterations are performed and thus only the extrapolation step of Algorithm 1 is employed.

Corollary 3 *Suppose that Assumptions 5.1 to 5.6 hold. Suppose that the approximate solution* $(\Delta x, -\Delta y)$ *of* (13) *satisfies Termination Test 3 with* $\beta_2 > 0$. *Assume that the sequence* $\{\epsilon_k\}$ *is chosen such that*

$$\epsilon_k = \omega(\delta_k^{1+\beta}),$$

where β *is as in Theorem 6. For sufficiently large* k, *the iterates computed at Step 4 of Algorithm 1 satisfy* $w_{k+1} = w_k^+$ *and* $\delta_k = \|F(w_k)\|$ *converges to zero at the rate* $1 + \beta$.

Proof. The result follows directly from Theorem 6 and the assumption on ϵ_k. □

Because $\beta := \min(\beta_1, \beta_2)$ and β_1 may be unknown, it is safe to set $\epsilon_k = \Theta(\delta_k)$ in Corollary 3. A consequence of Lemma 2 and Corollary 3 is that $\{w_k\} \to w^*$ R-superlinearly. The next result establishes Q-superlinear convergence to the set $\mathcal{S}$, which, though weaker than convergence to w^*, results from the fact that $\mathcal{Y}$ may be an unbounded set.

Theorem 7 *Under the assumptions of Corollary 3, the sequence* $\{\operatorname{dist}(w_k, \mathcal{S})\}$, *where* $\{w_k\}$ *is generated by Algorithm 1, converges Q-superlinearly to zero with rate* $1 + \beta$.

Proof. The result follows directly from Corollary 2 and Theorem 6. □

A more precise result follows when (9) is solved sufficiently accurately in the sense that $\beta_2 = 1$ in Termination Test 3. The next corollary follows [45, Corollary 4.2].

Corollary 4 *Under the assumptions of Corollary 3 with* $\beta_2 = 1$, *the sequence* $\{w_k\}$ *generated by Algorithm 1 converges Q-superlinearly to* w^* *with rate* $1 + \beta$, *and* $\|w_k - w^*\| = \Theta(\delta_k)$.

Proof. By Theorem 5, there exists a constant $C > 0$ such that $\|(\Delta x_j, \Delta y_j)\| \le C\delta_j$ for all sufficiently large j. By Theorem 6, $\{\delta_k\} \to 0$ superlinearly, and thus for all $\ell > k$ sufficiently large, we have

$$\|w_k - w_\ell\| \le \sum_{j=k}^{\ell-1} \|(\Delta x_j, \Delta y_j)\| \le C \sum_{j=k}^{\ell-1} \delta_j \le 2C\delta_k.$$

In the limit, we obtain $\|w_k - w^*\| = O(\delta_k)$. Conversely, Lemma 2 yields $\delta_k = O(\operatorname{dist}(w_k, \mathcal{S})) = O(\|w_k - w^*\|)$. □

We close this section by noting that if Hessian approximations are sufficiently accurate in the sense that $\beta_1 = 1$ in Assumption 5.6 and if (9) is solved sufficiently accurately in the sense that $\beta_2 = 1$ in Termination Test 3, Theorem 6 reveals that $\beta = 1$ and quadratic convergence takes place. In particular, such situation occurs if exact second derivatives are used and (9) is solved exactly, e.g., by way of a stable factorization.

It is possible to weaken Assumption 5.6 and only require $(H_k - \nabla_{xx}^2 L(x^*, y^*))\Delta x = o(\|\Delta x\|)$, which is closer to the original Dennis-Moré condition [12]. In that case, the conclusion of Theorem 6 changes to $\delta_k^+ = o(\delta_k)$. Corollary 3, Theorem 7 and Corollary 4 all remain valid except that the rate of superlinear convergence cannot be specified.

Our requirements on the quality of the Hessian approximation and on the accuracy of the step computation are substantially weaker than those of, e.g., [2], who require exact steps and the stringent bound $H_k - \nabla_{xx} L(x_k, y_k) = O(\delta_k)$. In view of Theorem 5, the latter is akin to requiring exact second derivatives.

6 Implementation and numerical results

In this section we examine the practical behavior of Algorithms 1 and 2 and specify the details of our implementation. Our implementation is written in Python with the help of the open-source package NLP.py [5], a programming environment for designing numerical optimization methods. A Python implementation of LSMR is available in the PyKrylov package [35], a library of Krylov methods in pure Python.

Initial Lagrange multipliers Given a user-defined starting point x_s, the vector of Lagrange multipliers y_s is obtained as least-square solutions of $\nabla L(x_s, y) = 0$, i.e., by solving the linear system

$$\begin{bmatrix} I & J_s^T \\ J_s & -\zeta I \end{bmatrix} \begin{bmatrix} v \\ -y \end{bmatrix} = - \begin{bmatrix} g_s \\ 0 \end{bmatrix},$$

using MA57 [14] and discarding v or, alternatively, using LSMR to solve

$$\underset{y \in \mathbb{R}^m}{\text{minimize}} \;\; \tfrac{1}{2}\|J_s^T y - g_s\|^2 + \tfrac{1}{2}\zeta\|y\|^2.$$

In our implementation, ζ is set to 10^{-8}.

Penalty parameter The initial penalty parameter is set to $\delta_0 = \min\{0.1, \|F(w_0)\|\}$. In Step 3 of Algorithm 1, δ_k is chosen as

$$\delta_k = \max\left\{\min\{\|F(w_k)\|,\ 0.9\delta_{k-1},\ \delta_{k-1}^{1.1}\},\ \delta_{\min}\right\}$$

where $\delta_{\min}$ is a lower bound imposed on the penalty parameter. Our implementation uses $\delta_{\min} = 10^{-8}$. This rule ensures that δ_k does not deviate much from the value selected by the augmented Lagrangian mechanism in the global regime, and is set to $\|F(w_k)\|$ asymptotically so that Assumption 5.3 of the local convergence analysis is satisfied.

When w_k^+ does not satisfy (14), a sequence of inner iterations using Algorithm 2 is started from the initial guess $w_{k,0} = w_k$.

Even though global convergence of Algorithm 2 is promoted by way of a Wolfe linesearch, our implementation uses a simple Armijo linesearch, which we have found to be nearly as effective. In the first Wolfe condition, we use $c_1 = 10^{-4}$. At the end of the inner iterations, the current value of the penalty parameter is returned to the outer iteration and is used as δ_k.

Stopping conditions Optimality is declared when the norm of the optimality conditions at w_k satisfies

$$\|F(w_k)\| \le \epsilon_a + \epsilon_r \|F(w_0)\|,$$

where $\epsilon_a = 10^{-6}$ and $\epsilon_r = 10^{-5}$. In (15), we set $\epsilon_k = 10^3\delta_k$ and $\theta = 0.99$.

The maximum number of iteration is set to 1000 for each problem.

Exact system solves We use the linear solver MA57 to solve (13) and (16). The primal regularization parameter ρ_k is updated until a correct inertia is detected according to [42, Algorithm IC], with the same values for the various constants.

Inexact system solves In Step 3 of Algorithm 1 and Step 3 of Algorithm 2, LSMR is used to solve the preconditioned linear least-squares problem (20). The preconditioner, $H_{k,j}^{-1}$, is obtained by maintaining a limited-memory BFGS approximation of the Hessian of the Lagrangian in inverse form. It is well known that a standard BFGS approximation is ineffective in the presence of constraints because the Hessian of the Lagrangian is typically indefinite at a solution—see, e.g., [9]. If $s_k = x_{k+1} - x_k$ and $t_k = \nabla L(x_{k+1}, y_{k+1}) - \nabla L(x_k, y_{k+1})$, the curvature condition $s_k^T t_k > 0$ is unlikely to hold asymptotically, and the pair (s_k, t_k) will be rejected.

Powell [39] suggests to use a damped BFGS update that compensates for the lack of positive definiteness in the Hessian at the solution. His approach is based on the Hessian of the Lagrangian, instead of its inverse. In the current work, we transpose Powell's modified update in terms of $B_k = H_k^{-1}$. Let $q_k := \theta_k s_k + (1-\theta_k)B_k t_k$ where $\theta_k \in (0, 1]$ is defined as

$$\theta_k = \begin{cases} 1 & \text{if } s_k^T t_k \ge \eta\, t_k^T B_k t_k \\ (1-\eta)\, t_k^T B_k t_k / (t_k^T B_k t_k - s_k^T t_k) & \text{otherwise,} \end{cases}$$

for some $\eta \in (0, 1)$. In our implementation, we set $\eta := 0.2$. A straightforward derivation similar to that of [39] shows that if B_k is positive definite, B_{k+1} is also positive definite. Thus starting this damped BFGS approximation by a scaled identity matrix $B_0 = \gamma_0 I$ with $\gamma_0 > 0$ ensures positive definiteness of all subsequent approximations. Note that the above update differs from the convergent updates of [1] but performed better in our limited experiments. In addition, only a small number of pairs (s_k, q_k) is stored so as to provide a limited-memory version of the damped BFGS method. Application of this approximation to a vector is performed using the two loop recursion.

Another way of ensuring the positive-definiteness of H_k is to maintain a BFGS approximation of the Hessian of the augmented Lagrangian ϕ [9]. This proposal is motivated by the fact that if δ is sufficiently large, the Hessian of $\phi(x, y; \rho, \delta)$ is positive definite for all (x, y) close to an isolated solution. However, in our numerical experiments, results were not as good as with Powell's damped BFGS. A reason for this is that when δ increases, convergence of the BFGS approximation is disrupted. Similar observations were also reported in [22].

In Termination Test 3, we set $\beta_2 = 0.5$ and $\mu = 0.2$. During the inner iterations, $\gamma_j = 10^{-4}$ for all j in Termination Test 1.

We perform a preliminary evaluation of our implementation of Algorithm 1, named RegSQP on equality-constrained problems from the Hock and Schittkowski [29] and CUTEst [26] collections. All models are formulated using the AMPL modeling language [18]. We do not apply any scaling procedure to the problems.

We first evaluate the algorithm using exact second derivatives. RegSQP uses MA57 [14] to compute steps. Table 1 summarizes the results. In the table, n is the number of variables, m is the number of equality constraints, "it" is the number of iterations, f is the final objective value, $\|c\|$ and $\|\nabla L\|$ are the final primal and dual feasibility in Euclidean norm, respectively, "$\#f$", "$\#g$", "$\#c$", "$\#J$" and "$\#H$" are the number of evaluations of the objective function, objective gradient, constraints, Jacobian and Hessian of the Lagrangian, respectively, and "time" is the solve time in seconds. A letter in the last colum indicates the final solver status in case of failure. The possible values of the final status are "i" when the maximum number of iterations is exceeded, and "x" for another kind of failure, typically a maximum number of backtracking steps during a linesearch.

Table 1: Performance of RegSQP with factorization on equality-constrained problems from CUTEst.

name	n	m	it	f	$\|c\|$	$\|\nabla L\|$	$\#f$	$\#g$	$\#c$	$\#J$	$\#H$	time	
aircrfta	5	5	2	0.0e+00	4.4e−06	5.2e−34	5	3	5	6	2	0.001	
argauss	3	15	1001	0.0e+00	1.1e−04	8.6e−10	2003	1002	2003	2004	1001	0.480	i
argtrig	100	100	3	0.0e+00	1.1e−08	7.8e−36	7	4	7	8	3	0.034	
artif	5000	5000	1002	0.0e+00	3.6e+02	1.2e+03	3091	1018	3095	2037	1012	24.376	i
aug2d	20192	9996	55	1.7e+06	5.7e−04	3.9e−12	111	56	111	112	55	3.440	
aug3d	3873	1000	6	5.5e+02	1.6e−07	5.0e−06	13	7	13	14	6	0.306	
aug3dc	3873	1000	31	7.7e+02	5.1e−04	8.5e−15	63	32	63	64	31	0.307	
bdvalue	5000	5000	1	0.0e+00	2.5e−08	1.2e−25	3	2	3	4	1	0.026	
booth	2	2	1	0.0e+00	8.9e−16	0.0e+00	3	2	3	4	1	0.000	
bratu2d	4900	4900	3	0.0e+00	3.5e−11	5.7e−29	7	4	7	8	3	0.270	
bratu2dt	4900	4900	31	0.0e+00	9.5e−02	0.0e+00	505	43	515	80	34	0.000	x
bratu3d	3375	3375	3	0.0e+00	3.2e−08	3.6e−25	7	4	7	8	3	0.516	
broydn3d	10000	10000	3	0.0e+00	3.1e−04	2.5e−31	7	4	7	8	3	0.113	
broydnbd	5000	5000	4	0.0e+00	4.5e−04	8.3e−32	9	5	9	10	4	0.169	
bt1	2	1	8	−1.0e+00	3.3e−06	1.7e−06	17	9	17	18	8	0.004	
bt10	2	2	9	−1.0e+00	7.9e−07	1.0e−06	19	10	19	20	9	0.004	
bt11	5	3	17	8.2e−01	6.5e−05	3.9e−05	35	18	35	36	17	0.009	
bt12	5	3	5	6.2e+00	4.9e−06	3.2e−08	11	6	11	12	5	0.002	
bt2	3	1	9	3.3e−02	1.7e−02	5.0e−02	19	10	19	20	9	0.004	
bt3	5	3	36	4.1e+00	1.4e−03	1.1e−15	73	37	73	74	36	0.022	
bt4	3	2	28	−3.7e+00	1.5e−09	9.5e−11	57	29	57	58	28	0.015	
bt5	3	2	7	9.6e+02	5.3e−08	2.0e−09	17	9	17	18	8	0.004	
bt6	5	2	11	2.8e−01	1.7e−05	2.6e−04	23	12	23	24	11	0.005	
bt7	5	3	161	3.0e+02	5.6e−03	2.8e−04	465	173	465	346	172	0.120	
bt8	5	2	37	1.0e+00	9.1e−11	3.1e−05	99	40	99	80	39	0.024	
bt9	4	2	12	−1.0e+00	3.6e−05	4.5e−05	25	13	25	26	12	0.006	
byrdsphr	3	2	16	−4.7e+00	7.9e−07	1.8e−08	35	18	35	36	17	0.008	
catena	32	11	51	−2.3e+04	4.2e−05	1.5e−04	103	52	103	104	51	0.025	
catenary	496	166	1002	−2.6e+06	1.3e+05	3.8e+02	20582	1024	20601	2032	1003	3.760	i

Continued on next page

Table 1—continued from previous page

name	n	m	it	f	$\|c\|$	$\|\nabla L\|$	$\#f$	$\#g$	$\#c$	$\#J$	$\#H$	time	
cbratu2d	882	882	1	0.0e+00	4.4e−07	3.1e−29	3	2	3	4	1	0.011	
cbratu3d	1024	1024	1	0.0e+00	8.6e−07	2.1e−27	3	2	3	4	1	0.037	
cluster	2	2	7	0.0e+00	7.4e−06	4.2e−08	15	8	15	16	7	0.003	
coolhan	9	9	7	0.0e+00	6.6e−04	6.9e−16	15	8	15	16	7	0.004	
dixchlng	10	5	61	2.5e+03	1.1e−03	1.3e+00	282	93	282	186	92	0.072	
drcavty1	10816	816	94	1.6e−11	4.1e−15	9.9e−07	375	129	375	258	128	51.076	
drcavty2	10816	816	495	6.0e−04	1.7e−15	1.4e−07	2016	549	2016	1098	548	209.313	
drcavty3	10816	816	729	2.0e−04	1.4e−15	1.7e−06	6497	3092	6497	6184	3091	1186.212	
dtoc1l	14985	9990	21	1.3e+02	1.8e−07	9.2e−08	43	22	43	44	21	1.359	
dtoc1na	1485	990	15	1.3e+01	3.8e−08	1.7e−08	31	16	31	32	15	0.377	
dtoc1nb	1485	990	16	1.6e+01	5.6e−08	1.9e−08	33	17	33	34	16	0.460	
dtoc1nc	1485	990	28	2.5e+01	3.8e−08	5.1e−08	57	29	57	58	28	0.852	
dtoc1nd	735	490	281	1.3e+01	3.3e−08	1.1e−05	2962	283	2962	566	282	4.856	
dtoc2	5994	3996	22	5.0e−01	9.2e−07	4.4e−06	45	23	45	46	22	0.585	
dtoc3	14997	9998	42	2.4e+02	6.5e−05	7.9e−14	85	43	85	86	42	1.522	
dtoc4	14997	9998	20	2.9e+00	9.4e−07	5.4e−08	41	21	41	42	20	0.802	
dtoc5	9998	4999	18	1.5e+00	2.8e−06	1.4e−07	37	19	37	38	18	0.379	
dtoc6	10000	5000	111	1.3e+05	1.9e−06	2.4e−05	310	124	310	248	123	2.849	
eigena2	110	55	12	8.4e−12	6.6e−04	1.4e−04	33	17	33	34	16	0.018	
eigenaco	110	55	64	8.5e−17	3.0e−09	1.7e−07	154	71	154	142	70	0.131	
eigenb2	110	55	1002	2.1e+00	3.4e−03	4.4e−01	12039	1009	12040	2018	1006	3.581	i
eigenbco	110	55	40	2.2e−01	2.2e−09	3.5e−05	81	41	81	82	40	0.271	
eigenc2	462	231	91	1.8e−12	5.5e−10	2.7e−06	434	112	434	224	111	3.818	
eigencco	30	15	26	5.7e−10	1.3e−05	5.8e−05	53	27	53	54	26	0.026	
genhs28	10	8	8	9.3e−01	7.2e−07	5.6e−16	17	9	17	18	8	0.004	
gottfr	2	2	7	0.0e+00	1.1e−09	6.6e−24	15	8	15	16	7	0.003	
gridnetb	13284	6724	50	1.4e+02	8.7e−05	1.6e−14	101	51	101	102	50	2.061	
hager1	10001	5001	6	8.8e−01	1.4e−06	9.3e−15	13	7	13	14	6	0.088	
hager2	10000	5000	2	4.3e−01	1.7e−07	3.6e−15	5	3	5	6	2	0.497	
hager3	10000	5000	1	1.4e−01	2.0e−02	1.5e−15	3	2	3	4	1	0.030	
hatfldf	3	3	7	0.0e+00	2.9e−06	2.0e−19	15	8	15	16	7	0.003	
hatfldg	25	25	35	0.0e+00	1.2e−08	5.5e−09	73	37	73	74	36	0.024	
heart6	6	6	606	0.0e+00	6.9e−06	4.6e−06	3121	611	3121	1222	610	0.577	
heart8	8	8	205	0.0e+00	4.8e−07	1.1e−06	844	215	844	430	214	0.249	
himmelba	2	2	1	0.0e+00	0.0e+00	0.0e+00	3	2	3	4	1	0.001	
himmelbc	2	2	5	0.0e+00	6.5e−08	1.1e−23	11	6	11	12	5	0.003	
himmelbd	2	2	1002	0.0e+00	3.1e+02	2.4e−02	19845	1030	19866	2050	1007	1.637	i
himmelbe	3	3	2	0.0e+00	2.2e-16	0.0e+00	5	3	5	6	2	0.002	
hs006	2	1	7	0.0e+00	0.0e+00	0.0e+00	17	9	17	18	8	0.004	
hs007	2	1	8	−1.7e+00	2.5e−06	7.6e−07	17	9	17	18	8	0.004	
hs008	2	2	5	−1.0e+00	1.4e−10	0.0e+00	11	6	11	12	5	0.002	
hs009	2	1	10	−5.0e−01	4.5e−13	1.6e−13	21	11	21	22	10	0.004	
hs026	3	1	16	3.7e−10	1.1e−05	4.8e−07	33	17	33	34	16	0.006	
hs027	3	1	20	4.0e−02	1.7e−09	6.9e−10	111	51	111	102	50	0.023	
hs028	3	1	2	1.2e−32	7.8e−16	2.8e−16	5	3	5	6	2	0.001	
hs039	4	2	13	−1.0e+00	3.2e−08	4.1e−08	27	14	27	28	13	0.005	
hs040	4	3	5	−2.5e−01	3.4e−09	1.7e−09	11	6	11	12	5	0.002	
hs046	5	2	23	7.2e−10	6.6e−06	5.5e−07	49	25	49	50	24	0.010	
hs047	5	3	47	7.4e−09	3.9e−06	1.6e−05	95	48	95	96	47	0.021	
hs048	5	2	2	1.2e−32	9.9e−15	1.4e−15	5	3	5	6	2	0.001	
hs049	5	2	25	1.4e−13	4.7e−05	3.8e−06	53	27	53	54	26	0.013	
hs050	5	3	25	1.1e−07	5.1e−04	3.2e−06	51	26	51	52	25	0.010	
hs051	5	3	6	1.1e−16	6.4e−09	2.1e−16	13	7	13	14	6	0.002	
hs052	5	3	38	5.3e+00	1.4e−07	1.4e−15	77	39	77	78	38	0.016	
hs061	3	2	12	−1.4e+02	4.0e−09	7.2e−10	25	13	25	26	12	0.005	
hs077	5	2	10	2.4e−01	5.9e−09	6.1e−08	21	11	21	22	10	0.004	
hs078	5	3	21	−8.2e−01	2.2e−11	8.6e−12	43	22	43	44	21	0.012	
hs079	5	3	8	7.9e−02	7.9e−08	1.5e−07	17	9	17	18	8	0.004	
hs100lnp	7	2	8	6.8e+02	8.5e−11	2.7e−10	17	9	17	18	8	0.004	
hs111lnp	10	3	13	−4.8e+01	4.8e−07	7.1e−07	27	14	27	28	13	0.007	
hypcir	2	2	4	0.0e+00	1.3e−05	1.2e−20	9	5	9	10	4	0.002	
integreq	100	100	2	0.0e+00	1.1e−06	1.6e−36	5	3	5	6	2	0.028	
lch	600	1	1002	−2.1e−31	1.0e+00	3.7e−06	6352	1158	6427	2390	1081	4.675	i
marato	2	1	3	−1.0e+00	5.3e−07	2.7e−07	7	4	7	8	3	0.001	
msqrta	1024	1024	5	0.0e+00	8.6e−04	3.3e−26	11	6	11	12	5	0.997	

Continued on next page

Table 1—continued from previous page

name	n	m	it	f	$\|c\|$	$\|\nabla L\|$	$\#f$	$\#g$	$\#c$	$\#J$	$\#H$	time
msqrtb	1024	1024	5	0.0e+00	8.6e−04	1.2e−28	11	6	11	12	5	1.010
mwright	5	3	26	1.3e+00	7.6e−07	5.8e−06	53	27	53	54	26	0.014
orthrdm2	4003	2000	6	1.6e+02	3.2e−06	2.7e−07	20	8	20	16	7	0.490
orthrds2	203	100	33	3.1e+01	2.9e−05	1.0e−02	83	38	83	76	37	0.070
orthrega	517	256	439	1.4e+03	1.6e−05	1.2e−02	1574	459	1574	918	458	3.072
orthregb	27	6	2	4.6e−15	1.1e−07	1.4e−07	5	3	5	6	2	0.002
orthregc	10005	5000	25	1.9e+02	3.3e−05	3.6e−05	51	26	51	52	25	24.355
orthregd	10003	5000	6	1.5e+03	8.0e−04	1.8e−04	20	8	20	16	7	2.711
orthrgdm	10003	5000	11	1.5e+03	3.1e−05	6.4e−06	30	13	30	26	12	13.896
orthrgd	10003	5000	53	1.6e+03	9.0e−04	1.6e−01	132	59	132	118	58	75.270
porous1	4900	4900	9	0.0e+00	2.8e−01	1.4e−26	30	11	30	22	10	1.075
porous2	4900	4900	6	0.0e+00	4.0e−02	6.3e−24	24	8	24	16	7	0.693
powellb	2	2	11	0.0e+00	7.8e−07	3.0e−21	23	12	23	24	11	0.006
powellsq	2	2	22	0.0e+00	1.1e−07	1.1e−08	45	23	45	46	22	0.010
recipe	3	3	2	0.0e+00	0.0e+00	0.0e+00	5	3	5	6	2	0.001
zangwil3	3	3	1	0.0e+00	4.4e−14	0.0e+00	3	2	3	4	1	0.001

We note that the direct implementation fails on only 7 problems out of 110. Regarding the failures, we note that IPOPT 3.12.8 reports that argauss has too few degrees of freedom, solves bratu2dt to "acceptable levels" only, reports that himmelbd may be infeasible, and finds an optimal solution to the other problems.

Next, we evaluate our method using L-BFGS approximations with 6 pairs in the history. Table 2 summarizes the results. The table headers are as before, with the exception of "#jprod", which represents the number of operator-vector products with the Jacobian or its transpose. It is apparent from Table 2 that our preliminary implementation of the matrix-free implementation is not as robust as the direct version; 67 out of 110 problems are solved to optimality and another 7 are suboptimal in the sense that somewhat looser, but acceptable, tolerances were attained: argauss, drcavty1, drcavty2, drcavty3, dtoc2, dtoc4, and eigenbco. Note that drcavty3 was interrupted after 500 iterations due to a long run time for higher values of the iteration limit. In all other instances, the final status "xfail" indicates a linesearch failure in our primitive implementation of the inner iteration. Such failures occur with both a simple Armijo backtracking and our strong Wolfe linesearch. They are never due to failure of the iterative solver for (19).

Table 2: Performance of RegSQP without factorization on equality-constrained problems from CUTEst.

name	n	m	it	f	$\|c\|$	$\|\nabla L\|$	$\#f$	$\#g$	$\#c$	#jprod	time	
aircrfta	5	5	12	0.0e+00	2.8e+00	0.0e+00	18	15	18	41	0.000	x
argauss	3	15	1001	0.0e+00	1.1e−04	2.9e−10	1002	1002	1002	3003	1.045	i
argtrig	100	100	20	0.0e+00	5.7e+00	0.0e+00	22	22	22	63	0.000	
artif	5000	5000	1002	0.0e+00	1.4e−01	3.1e−03	17980	1015	17981	2103	200.235	i
aug2d	20192	9996	873	1.7e+06	3.4e−04	1.2e−03	1722	882	1722	21828	603.021	
aug3d	3873	1000	47	5.5e+02	1.1e−09	4.9e−04	48	48	48	2142	13.621	
aug3dc	3873	1000	57	7.7e+02	3.2e−08	1.1e−04	61	59	61	2175	16.348	
bdvalue	5000	5000	0	0.0e+00	3.2e−06	0.0e+00	1	1	1	1	0.000	
booth	2	2	59	0.0e+00	8.3e−11	5.5e−06	68	63	68	187	0.055	
bratu2d	4900	4900	42	0.0e+00	2.2e−10	9.5e−07	66	44	66	109	51.779	
bratu2dt	4900	4900	47	0.0e+00	9.5e−02	0.0e+00	86	50	86	123	0.000	
bratu3d	3375	3375	60	0.0e+00	1.5e+00	0.0e+00	99	63	99	168	0.000	
broydn3d	10000	10000	1002	0.0e+00	2.0e+01	2.5e+03	1971	1005	1972	2039	201.235	i
broydnbd	5000	5000	649	0.0e+00	4.2e+02	0.0e+00	5314	700	5343	1428	0.000	x
bt1	2	1	1002	−9.3e+01	9.3e−01	1.1e+02	1979	1005	1980	2034	1.057	i
bt10	2	2	23	−1.0e+00	2.1e−08	1.4e−06	24	24	24	74	0.018	
bt11	5	3	29	8.2e−01	7.3e−08	1.1e−04	30	30	30	94	0.030	
bt12	5	3	35	6.2e+00	7.7e−09	3.2e−05	36	36	36	112	0.037	
bt2	3	1	34	3.3e−02	6.3e−04	8.3e−02	35	35	35	105	0.029	
bt3	5	3	35	4.1e+00	1.2e−06	1.1e−03	65	59	65	543	0.065	
bt4	3	2	1002	1.1e−02	2.5e+01	1.2e+03	2009	1008	2010	2060	1.017	i
bt5	3	2	57	9.6e+02	7.1e−09	1.3e−04	69	61	69	185	0.061	
bt6	5	2	196	2.8e−01	8.5e−09	1.4e−04	2719	239	2719	1392	0.376	
bt7	5	3	56	3.1e+02	1.6e−08	1.2e−03	197	114	197	2008	0.121	
bt8	5	2	26	1.0e+00	1.4e−09	3.5e−05	27	27	27	83	0.026	

Continued on next page

Table 2—continued from previous page

name	n	m	it	f	$\lVert c\rVert$	$\lVert\nabla L\rVert$	$\#f$	$\#g$	$\#c$	#jprod	time	
bt9	4	2	20	−1.0e+00	7.7e−08	2.1e−05	21	21	21	65	0.021	
byrdsphr	3	2	52	−4.7e+00	7.9e−08	2.3e−05	56	54	56	164	0.053	
catena	32	11	59	−2.3e+04	2.2e−10	7.9e−03	215	117	215	2555	0.202	
catenary	496	166	1002	−5.7e+05	4.3e+01	2.0e+03	7992	1041	7999	2174	3.328	i
cbratu2d	882	882	48	0.0e+00	6.4e−08	2.0e−07	90	50	90	109	4.148	
cbratu3d	1024	1024	14	0.0e+00	3.1e−06	5.0e−07	17	16	17	46	2.801	
cluster	2	2	48	0.0e+00	1.8e−08	3.7e−08	72	54	72	155	0.050	
coolhan	9	9	164	0.0e+00	9.5e+02	0.0e+00	301	177	301	431	0.000	x
dixchlng	10	5	35	2.5e+03	2.6e−07	1.1e+00	184	90	184	1194	0.098	
drcavty1	10816	816	1002	4.6e−05	1.7e−12	6.9e−03	2111	1066	2112	2401	67.716	i
drcavty2	10816	816	1002	2.2e−05	4.1e−12	5.4e−03	2100	1050	2102	2307	59.321	i
drcavty3	10816	816	502	2.0e−04	2.1e−13	8.3e−04	1078	552	1079	1345	43.114	i
dtoc1l	14985	9990	92	1.3e+02	2.0e−10	2.7e−09	140	94	140	20225	492.947	
dtoc1na	1485	990	84	1.3e+01	4.3e−11	7.1e−10	133	86	133	2191	10.957	
dtoc1nb	1485	990	94	1.6e+01	6.9e−11	1.8e−09	144	96	144	2220	13.345	
dtoc1nc	1485	990	103	2.5e+01	1.4e−06	1.2e−05	267	105	267	2242	14.411	
dtoc1nd	735	490	1002	1.3e+01	2.1e−04	4.6e−01	11154	1006	11155	3035	20.866	i
dtoc2	5994	3996	502	4.9e−01	2.6e−04	7.4e−03	981	510	986	1035	103.566	i
dtoc3	14997	9998	86	2.4e+02	2.1e−07	1.3e−04	155	90	155	240	521.957	
dtoc4	14997	9998	502	2.9e+00	1.1e−05	1.2e−02	980	507	981	1050	3288.848	i
dtoc5	9998	4999	45	2.0e−04	1.0e+00	0.0e+00	63	48	63	131	0.000	x
dtoc6	10000	5000	502	1.3e+05	4.6e−03	1.8e+00	1224	549	1228	11881	396.894	i
eigena2	110	55	6	1.1e−09	1.1e−06	2.3e−03	85	63	85	2728	0.283	
eigenaco	110	55	44	1.2e−09	2.8e−12	2.1e−04	137	96	137	2445	0.490	
eigenb2	110	55	1002	1.8e+01	8.3e−08	1.7e−01	17489	1060	17490	4622	2.913	i
eigenbco	110	55	1002	8.2e−01	8.7e−07	2.9e−02	18451	1039	18452	2707	3.620	i
eigenc2	462	231	1002	4.1e+01	3.6e−04	1.2e+02	24764	1051	24765	4466	39.692	i
eigencco	30	15	80	1.1e−10	7.3e−12	5.4e−05	266	131	266	1334	0.260	
genhs28	10	8	54	9.3e−01	1.9e−10	3.9e−07	64	56	64	179	0.099	
gottfr	2	2	21	0.0e+00	2.4e−09	1.4e−05	28	25	28	75	0.029	
gridnetb	13284	6724	438	1.4e+02	2.2e−09	6.7e−07	821	441	821	944	310.679	
hager1	10001	5001	21	0.0e+00	1.0e+00	0.0e+00	27	25	27	76	0.000	x
hager2	10000	5000	52	3.3e−05	5.0e+03	6.7e−09	55	54	55	10159	0.000	x
hager3	10000	5000	49	1.6e−05	5.0e+03	2.5e−05	52	51	52	10150	0.000	x
hatfldf	3	3	61	0.0e+00	2.8e−01	0.0e+00	244	64	244	146	0.000	x
hatfldg	25	25	66	0.0e+00	5.2e+00	0.0e+00	94	69	94	182	0.000	x
heart6	6	6	525	0.0e+00	2.4e+01	0.0e+00	1871	534	1871	1131	0.000	x
heart8	8	8	99	0.0e+00	1.3e+01	0.0e+00	225	104	225	263	0.000	x
himmelba	2	2	54	0.0e+00	1.4e−08	8.2e−05	55	55	55	162	0.052	
himmelbc	2	2	42	0.0e+00	1.1e−05	3.2e−05	46	44	46	127	0.042	
himmelbd	2	2	67	0.0e+00	1.8e+03	0.0e+00	330	75	333	187	0.000	x
himmelbe	3	3	53	0.0e+00	3.0e+00	0.0e+00	72	56	72	151	0.000	x
hs006	2	1	13	3.3e−15	3.1e−08	3.0e−05	14	14	14	42	0.011	
hs007	2	1	20	−1.7e+00	1.5e−07	2.5e−05	21	21	21	63	0.018	
hs008	2	2	50	−1.0e+00	8.5e−09	7.4e−07	53	52	53	154	0.046	
hs009	2	1	19	−5.0e−01	3.0e−11	3.3e−07	41	21	41	60	0.017	
hs026	3	1	121	2.1e+01	0.0e+00	1.2e+01	482	162	482	842	0.000	x
hs027	3	1	35	4.0e+00	7.0e+00	1.6e+01	310	88	310	1738	0.000	x
hs028	3	1	27	1.5e−13	2.5e−08	1.6e−05	28	28	28	84	0.024	
hs039	4	2	20	−1.0e+00	7.7e−08	2.1e−05	21	21	21	65	0.019	
hs040	4	3	30	−2.5e−01	9.6e−12	6.5e−07	34	32	34	100	0.032	
hs046	5	2	57	3.3e+00	2.2e−16	7.6e+00	597	112	597	3090	0.000	x
hs047	5	3	69	−6.6e−07	3.5e−08	1.8e−04	625	125	625	3024	0.154	
hs048	5	2	37	1.9e−13	1.1e−08	7.3e−05	38	38	38	116	0.036	
hs049	5	2	19	1.6e−06	2.1e−12	9.7e−04	93	76	93	2829	0.059	
hs050	5	3	16	1.7e−10	8.3e−09	4.9e−03	88	67	88	2412	0.052	
hs051	5	3	37	1.7e−13	2.0e−09	6.0e−05	38	38	38	118	0.038	
hs052	5	3	50	5.3e+00	2.4e−07	1.5e−04	51	51	51	157	0.049	
hs061	3	2	28	−1.4e+02	9.2e−06	6.9e−05	29	29	29	89	0.024	
hs077	5	2	110	5.5e+00	8.5e−12	1.1e−06	515	155	515	1088	0.158	
hs078	5	3	27	−2.9e+00	2.1e−08	2.3e−05	28	28	28	88	0.027	
hs079	5	3	26	7.9e−02	5.3e−09	5.4e−05	27	27	27	85	0.028	
hs100lnp	7	2	48	6.8e+02	9.7e−10	1.9e−04	119	79	119	651	0.080	
hs111lnp	10	3	49	−2.1e+01	1.4e+00	7.2e−01	61	52	61	155	0.000	x
hypcir	2	2	13	0.0e+00	1.5e−06	6.4e−06	14	14	14	39	0.012	

Continued on next page

Table 2—continued from previous page

name	n	m	it	f	$\|c\|$	$\|\nabla L\|$	$\#f$	$\#g$	$\#c$	#jprod	time	
integreq	100	100	35	0.0e+00	4.7e−09	1.7e−05	39	37	39	108	0.861	
lch	600	1	1002	−3.4e+00	8.6e−06	3.0e+00	11677	1060	11678	3234	2.552	i
marato	2	1	7	−1.0e+00	8.2e−11	4.2e−06	8	8	8	24	0.006	
msqrta	1024	1024	165	0.0e+00	8.9e+01	0.0e+00	292	172	292	408	0.000	x
msqrtb	1024	1024	261	0.0e+00	8.9e+01	0.0e+00	478	268	478	602	0.000	x
mwright	5	3	21	2.5e+01	8.2e−09	1.0e−04	93	69	93	1742	0.058	
orthrdm2	4003	2000	24	1.6e+02	9.6e−04	4.9e−02	90	26	90	57	3.209	
orthrds2	203	100	346	3.8e+02	4.5e−04	9.7e−03	902	366	902	826	1.478	
orthrega	517	256	1002	1.7e+03	3.8e−06	5.2e+00	2035	1040	2038	2211	6.065	i
orthregb	27	6	7	7.7e−07	1.5e−07	1.8e−03	8	8	8	21	0.011	
orthregc	10005	5000	524	1.9e+02	1.1e−09	8.7e−05	1713	536	1713	1152	180.110	
orthregd	10003	5000	345	7.4e+03	1.8e−03	2.6e−02	835	362	835	810	186.537	
orthrgdm	10003	5000	1002	1.2e+04	2.4e−01	2.4e+02	15970	1037	15971	2184	220.255	i
orthrgd	10003	5000	291	1.6e+03	1.3e−03	7.8e−02	1837	302	1837	10678	164.767	
porous1	4900	4900	1	0.0e+00	5.8e+04	0.0e+00	9	3	9	5	0.000	x
porous2	4900	4900	1	0.0e+00	5.4e+04	0.0e+00	10	3	10	5	0.000	x
powellb	2	2	26	0.0e+00	1.1e+00	0.0e+00	31	28	31	78	0.000	x
powellsq	2	2	1002	0.0e+00	1.1e+01	4.8e+00	31904	1013	31911	2068	2.695	i
recipe	3	3	52	0.0e+00	8.5e−09	1.7e−05	59	55	59	162	0.060	
zangwil3	3	3	30	0.0e+00	1.7e+02	0.0e+00	33	32	33	92	0.000	x

7 Discussion

The main contribution of this paper is the formulation and analysis of an algorithm for equality-constrained optimization that combines the favorable global properties of augmented Lagrangian methods and local properties of stabilized SQP methods. The use of positive-definite limited-memory approximations to the Hessian of the Lagrangian presents the significant advantage that the linear system encountered at each iteration is always SQD. An appropriate interpretation of that system in terms of a linear least-squares problem permits efficient inexact system solves and an entirely factorization-free implementation. The analysis of Sections 4 and 5 does not rely on the LICQ. The numerical results of Section 6 indicate that the proposed method is promising but call for a more robust implementation of the inner iteration. A performance evaluation on a set of degenerate problems is the subject on ongoing work. In order for our implementation to become more generally useful, it is necessary to supplement it with an infeasibility-detection mechanism as well as penalty and multiplier update rules that allow us to guarantee convergence towards a point when the constraint qualification fails when the penalty parameter diverges and the iterates approach the feasible set.

The use of our BFGS-LSMR strategy for solving (13) or (16) inexactly is not restricted to the specific scope of the present research. It could also be employed when applying a quadratic penalty method to (1) in the same vein as described in [4]. Following the procedure of Section 2 leads to the fully regularized subproblem

$$\underset{x\in\mathbb{R}^n, r\in\mathbb{R}^m}{\text{minimize}} \quad f(x) + \tfrac{1}{2}\rho_k\|x - x_k\|^2 + \tfrac{1}{2}\delta_k\|r\|^2 \quad \text{subject to} \;\; c(x) + \delta_k r = 0, \tag{37}$$

for $\delta_k > 0$, $\rho_k \geq 0$ and for some new variables r. Applying Newton's method to the KKT conditions of (37) yields

$$\begin{bmatrix} H_k + \rho_k I & J_k^T \\ J_k & -\delta_k I \end{bmatrix} \begin{bmatrix} \Delta x \\ -\Delta y \end{bmatrix} = - \begin{bmatrix} g_k - J_k^T y_k \\ c_k + \delta_k y_k \end{bmatrix}. \tag{38}$$

Note that the coefficient of (38) is the same as that of (16), only the right-hand side differs.

The local convergence properties of Section 5 assume that quasi-Newton approximations converge superlinearly to the exact Hessian at the solution along the primal steps. The authors of [9] establish that fast local convergence of a SQP method can take place provided H_k is defined as the BFGS approximation of the Hessian of the *augmented* Lagrangian. It may be possible to transpose their results to the present context because (9) is precisely a step on an augmented Lagrangian. Unfortunately, the performance of the augmented Lagrangian approximation is poor compared to that of the damped BFGS update of [39]. Local convergence properties of the damped update remains an open question. Powell proves that if convergence occurs, it does so at a R-superlinear rate. Further exploration of those considerations is beyond the scope of the present paper and is left for future research.

We briefly mention potential avenues to extend our method to problems with inequality constraints. A first approach is to use an augmented Lagrangian function that takes inequalities into account, such as that of [7], as in [31]. The advantage of such an approach is that we expect its convergence analysis to be similar to that developed in the present paper. Another approach is to add slack variables and treat bounds via a logarithmic barrier. Such an approach could continue to build upon the interior-point framework of [3] and its transition between the global and local regimes, and would continue to bear some resemblance with IPOPT.

Finally, allowing the penalty parameter to increase during the inner iterations as proposed in [2] might further improve performance.

References

[1] M. Al-Baali. Damped techniques for enforcing convergence of quasi-Newton methods. Optimization Methods and Software, 29(5):919–936, 2014. DOI: 10.1080/10556788.2014.891030.

[2] P. Armand and R. Omheni. A globally and quadratically convergent primal-dual augmented Lagrangian algorithm for equality constrained optimization. Optimization Methods and Software, 2015. DOI: 10.1080/10556788.2015.1025401. Online First.

[3] P. Armand, J. Benoist, and D. Orban. From global to local convergence of interior methods for nonlinear optimization. Optimization Methods and Software, 28(5):1051–1080, 2012. DOI: 10.1080/10556788.2012.668905.

[4] P. Armand, J. Benoist, R. Omheni, and V. Pateloup. Study of a primal-dual algorithm for equality constrained minimization. Computational Optimization and Applications, 59(3):405–433, July 2014. DOI: 10.1007/s10589-014-9679-3.

[5] S. Arreckx, D. Orban, and N. van Omme. NLP.py — a large-scale optimization toolkit in Python. Cahier du GERAD G–2016–42, GERAD, Montréal, QC, Canada, 2016.

[6] D. P. Bertsekas. Constrained optimization and Lagrange multiplier methods. Academic press, 1996. DOI: 10.1016/b978-0-12-093480-5.50005-2.

[7] E. G. Birgin and J. M. Martínez. Practical augmented Lagrangian methods for constrained optimization. SIAM, May 2014. DOI: 10.1137/1.9781611973365.

[8] P. T. Boggs and J. W. Tolle. Sequential quadratic programming. Acta Numerica, 4:1–51, January 1995. DOI: 10.1017/S0962492900002518.

[9] R. H. Byrd, R. A. Tapia, and Y. Zhang. An SQP augmented lagrangian BFGS algorithm for constrained optimization. SIAM Journal on Optimization, 2(2):210–241, 1992.

[10] R. H. Byrd, J. Nocedal, and R. B. Schnabel. Representations of quasi-Newton matrices and their use in limited memory methods. Mathematical Programming, 63(1):129–156, 1994. DOI: 10.1007/BF01582063.

[11] R. H. Byrd, F. E. Curtis, and J. Nocedal. An inexact Newton method for nonconvex equality constrained optimization. Mathematical Programming, 122(2):273–299, September 2009. DOI: 10.1007/s10107-008-0248-3.

[12] J. E. Dennis and Jorge J. Moré. Quasi-Newton methods, motivation and theory. SIAM Review, 19(1):46–89, January 1977. DOI: 10.1137/1019005.

[13] J. E. Jr Dennis and R. B. Schnabel. Numerical methods for unconstrained optimization and nonlinear equations, volume 16. SIAM, January 1996. DOI: 10.1137/1.9781611971200.

[14] I. S. Duff. MA57—a code for the solution of sparse symmetric definite and indefinite systems. ACM Transactions on Mathematical Software, 30(2):118–144, June 2004. DOI: 10.1145/992200.992202.

[15] D. Fernàndez and M. Solodov. Stabilized sequential quadratic programming for optimization and a stabilized Newton-type method for variational problems. Mathematical Programming, 125(1):47–73, September 2010. DOI: 10.1007/s10107-008-0255-4.

[16] D. Fernández, E. A. Pilotta, and G. A. Torres. An inexact restoration strategy for the globalization of the sSQP method. Computational Optimization and Applications, 54(3):595–617, 2012. DOI: 10.1007/s10589-012-9502-y.

[17] D. C.-L. Fong and M. A. Saunders. LSMR: An iterative algorithm for sparse least-squares problems. SIAM Journal on Scientific Computing, 33(5):2950–2971, 2011. DOI: 10.1137/10079687X.

[18] R. Fourer, D. M. Gay, and B. W. Kernighan. AMPL: A Modeling Language for Mathematical Programming. Duxbury Press / Brooks/Cole Publishing Company, second edition, 2003.

[19] M. P. Friedlander and D. Orban. A primal-dual regularized interior-point method for convex quadratic programs. Mathematical Programming Computation, 4(1):71–107, 2012. DOI: s12532-012-0035-2.

[20] J. Gauvin. A necessary and sufficient regularity condition to have bounded multipliers in nonconvex programming. Mathematical Programming, Series B, 12:136–138, 1977.

[21] P. Gill and D. Robinson. A globally convergent stabilized SQP method. SIAM Journal on Optimization, 23(4): 1983–2010, January 2013. DOI: 10.1137/120882913.

[22] P. E. Gill and E. Wong. Sequential quadratic programming methods. In Jon Lee and Sven Leyffer, editors, Mixed Integer Nonlinear Programming, volume 154 of The IMA Volumes in Mathematics and its Applications, pages 147–224. Springer New York, November 2011. DOI: 10.1007/978-1-4614-1927-3“6.

[23] P. E. Gill, V. Kungurtsev, and D. P. Robinson. A stabilized SQP method: global convergence. IMA Journal of Numerical Analysis, 37(1):407–443, 2017. DOI: 10.1093/imanum/drw004.

[24] P. E. Gill, V. Kungurtsev, and D. P. Robinson. A stabilized SQP method: superlinear convergence. Mathematical Programming, 163(1):369–410, 2017. DOI: 10.1007/s10107-016-1066-7.

[25] N. I. M. Gould. On practical conditions for the existence and uniqueness of solutions to the general equality quadratic programming problem. Mathematical Programming, 32(1):90–99, May 1985. DOI: 10.1007/BF01585660.

[26] N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEst: a Constrained and Unconstrained Testing Environment with safe threads. Computational Optimization and Applications, 60(3):545–557, 2015. DOI: 10.1007/s10589-014-9687-3.

[27] W. W. Hager. Stabilized sequential quadratic programming. Computational Optimization and Applications, 12(1):253–273, 1999. DOI: 10.1023/A:1008640419184.

[28] M. Heinkenschloss and D. Ridzal. A matrix-free trust-region sqp method for equality constrained optimization. SIAM Journal on Optimization, 24(3):1507–1541, 2014.

[29] W. Hock and K. Schittkowski. Test Examples for Nonlinear Programming Codes, volume 187, chapter Lectures Notes in Economics and Mathematical Systems. Springer Verlag, Berlin, 1981.

[30] A. F. Izmailov and M. V. Solodov. Stabilized SQP Revisited. Mathematical Programming, 133(1–2):93–120, June 2012. DOI: 10.1007/s10107-010-0413-3.

[31] A. F. Izmailov, M. V. Solodov, and E. I. Uskov. Combining stabilized SQP with the augmented Lagrangian algorithm. Computational Optimization and Applications, 62(2):405–429, 2015. DOI: 10.1007/s10589-015-9744-6.

[32] A. F. Izmailov, M. V. Solodov, and E. I. Uskov. Globalizing stabilized sequential quadratic programming method by smooth primal-dual exact penalty function. Journal of Optimization Theory and Applications, 169(1):148–178, 2016. DOI: 10.1007/s10957-016-0889-y.

[33] D. C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. Mathematical Programming, 45(1):503–528, August 1989. DOI: 10.1007/BF01589116.

[34] J. Nocedal and S. J. Wright. Numerical Optimization. Springer, New York, second edition, 2006. DOI: 10.1007/b98874.

[35] D. Orban. PyKrylov: Krylov subspace methods in pure Python. `github.com/PythonOptimizers/pykrylov`, July 2009.

[36] D. Orban and M. Arioli. Iterative Solution of Symmetric Quasi-Definite Linear Systems, volume 3. SIAM, 2017. DOI: 10.1137/1.9781611974737.

[37] C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. SIAM Journal on Numerical Analysis, 12(4):617–629, September 1975. DOI: 10.1137/0712047.

[38] C. C. Paige and M. A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. ACM Transactions on Mathematical Software, 8(1):43–71, 1982. DOI: 10.1145/355984.355989.

[39] M. J. D. Powell. A fast algorithm for nonlinearly constrained optimization calculations. In Lecture Notes in Mathematics, pages 144–157. Springer Science + Business Media, 1978. DOI: 10.1007/bfb0067703.

[40] R. T. Rockafellar. Augmented Lagrangians and applications of the proximal point algorithm in convex programming. Mathematics of Operations Research, 1:97–116, 1976. DOI: 10.1287/moor.1.2.97.

[41] R. J. Vanderbei. Symmetric quasi-definite matrices. SIAM Journal on Optimization, 5(1):100–113, 1995. DOI: 10.1137/0805005.

[42] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Mathematical Programming, 106:25–57, 2006. DOI: 10.1007/s10107-004-0559-y.

[43] R. B. Wilson. A Simplicial Method for Convex Programming. PhD thesis, Harvard University, Boston, USA, 1963.

[44] S. J. Wright. Primal-Dual Interior-Point Methods. SIAM, 1997. DOI: 10.1137/1.9781611971453.

[45] S. J. Wright. Superlinear Convergence of a Stabilized SQP Method to a Degenerate Solution. Computational Optimization and Applications, 11(3):253–275, 1998. DOI: 10.1023/A:1018665102534.

[46] S. J. Wright. An algorithm for degenerate nonlinear programming with rapid local convergence. SIAM Journal on Optimization, 15(3):673–696, 2005. DOI: 10.1137/030601235.