Noname manuscript No.
(will be inserted by the editor)

LLNL IM Release number: LLNL-JRNL-755231

Large-Scale Quasi-Newton Trust-Region Methods With Low-Dimensional Linear Equality Constraints

Johannes J. Brust · Roummel F. Marcia · Cosmin G. Petra

Received: date / Accepted: date

Abstract We propose two limited memory BFGS (L-BFGS) trust-region methods for large-scale optimization with linear equality constraints. The methods are intended for problems where the number of equality constraints is small. By exploiting the structure of the quasi-Newton compact representation, both proposed methods solve the trust-region subproblems nearly exactly, even for large problems. We derive theoretical global convergence results of the proposed algorithms, and compare their numerical effectiveness and performance on a variety of large-scale problems.

Keywords Linear Equality Constraints · Quasi-Newton · L-BFGS · Trust-Region Algorithm · Compact Representation · Eigendecomposition · Shape-Changing Norm

1 Introduction

The minimization problem with linear equality constraints is

$$\underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimize}}\ f(\mathbf{x}) \quad \text{subject to} \quad \mathbf{A}\mathbf{x}=\mathbf{b}, \tag{1}$$

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. R. Marcia's research is partially supported by NSF Grant IIS 1741490. C. Petra also acknowledges support from the LDRD Program of Lawrence Livermore National Laboratory under projects 16-ERD-025 and 17-SI-005.

J. J. Brust
University of California Merced, Merced, CA (formerly), Argonne National Laboratory, Lemont, IL, E-mail: jbrust@ucmerced.edu, jbrust@anl.gov

R. F. Marcia
University of California Merced, Merced, CA E-mail: rmarcia@ucmerced.edu

C. G. Petra
Lawrence Livermore National Laboratory, Livermore, CA E-mail: petra1@llnl.gov

where the objective function $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and the equality constraints are defined by $\mathbf{A} \in \mathbb{R}^{m\times n}$ and $\mathbf{b} \in \mathbb{R}^m$. We assume that $m \ll n$ and that $\mathbf{A}$ has full row rank, i.e., $\text{rank}(\mathbf{A}) = m$. The minimization problem (1) arises in many applications, including optimal allocation with resource constraints, equality constrained centering problems, and network flow optimization (see [1]). We assume it is computationally infeasible to evaluate the Hessian of f because n is large and, for this reason, we focus on limited-memory quasi-Newton methods for (1). The approaches in this article are based on solves with $\mathbf{A}\mathbf{A}^T$. Such computations may be inexpensive when m is small even for large n. Thus, the proposed methods are expected to be most efficient for relatively small m, although large m can be handled as long as solves with $\mathbf{A}\mathbf{A}^T$ can be computed efficiently.

1.1 Existing methods

In the context of quasi-Newton methods for large-scale problems such as (1), the algorithm of Lalee, Nocedal, and Plantenga [19] uses the limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) quasi-Newton matrix to generate feasible iterates, $\mathbf{x}_k$. In order to update the iterates, subproblems involving quasi-Newton matrices are solved using preconditioned conjugate-gradient techniques. A popular solver is IPOPT [26], which is an implementation of an interior-point algorithm for large-scale optimization with the option of L-BFGS matrices. The FMINCON algorithm [12] also uses L-BFGS matrices, and applies either direct LDL^{T} factorizations or iterative methods [13] to systems with these matrices. Another algorithm based on quasi-Newton methods is RSQP (Reduced-Hessian Successive Quadratic Programming [29]), which uses an L-BFGS approximation to the Hessian of the Lagrangian objective function in a particular subspace. Finally, PDCO (Primal-Dual interior method for Convex Objectives [23]) by M. Saunders is an algorithm specifically targeting problems of the form (1) (and allowing general bounds $\mathbf{l} \leq \mathbf{x} \leq \mathbf{u}$). The method assumes convex objective functions and uses explicit information from the Hessian matrix of second derivatives of f.

1.2 Paper contribution

We propose solving large-scale optimization problems with linear equality constraints by combining limited-memory quasi-Newton methods with trust-region methods. Unlike other constrained trust-region methods [11,19,22,25], we concentrate on exploiting the representations of large-scale quasi-Newton matrices. In particular, we propose two algorithms that extend unconstrained quasi-Newton trust-region methods to incorporate linear equality constraints. The first proposed method uses an ℓ_2-norm inequality constraint to define the trust-region, while the second uses a shape-changing norm proposed in [5]. Both methods treat $\mathbf{A}$ as a dense matrix and generate iterates that consistently

satisfy the linear constraints. While recent methods have exploited the compact representation of quasi-Newton matrices to solve trust-region subproblems (see e.g., [2–5, 7]), in this work we compute the compact representation of the $(1, 1)$ block of the inverse Karush-Kuhn-Tucker matrix, which prescribes the conditions for optimality of the trust-region subproblem with equality constraints. The main benefit of the proposed compact representation is that it allows computing the trust-region step efficiently for large problems. In addition, we demonstrate that the search directions generated by both methods satisfy a sufficient decrease condition that allows us to prove that the iterates generated by both methods converge to critical points.

1.3 Outline

In Section 2, we review quasi-Newton methods and trust-region methods. We define the trust-region subproblem corresponding to (1) and discuss the general strategy of our two proposed methods. In Section 3, we describe a trial step that both methods compute and test for feasibility. In Section 4, we propose our first algorithm, which uses the ℓ_2-norm to define the trust-region subproblem. In Section 5, we describe our second algorithm, which uses a shape-changing norm. In Section 6, we prove the global convergence of the proposed methods. In Section 7, we present a variety of numerical experiments to demonstrate the effectiveness of the proposed methods. In Section 8, we discuss how the proposed methods can be extended to solve problems with higher-dimensional linear equality constraints. In Section 9, we summarize our main results. Appendix A lists the main notation used.

2 Background

2.1 Limited-memory quasi-Newton methods

Quasi-Newton methods maintain approximations $\mathbf{B}_k$ to the Hessian matrix of the objective function $\nabla^2 f(\mathbf{x}_k)$ throughout the numerical optimization process. A variety of quasi-Newton Hessian approximation strategies exist. The Broyden-Fletcher-Goldfarb-Shanno (BFGS) method is arguably the most widely-used quasi-Newton update for large-scale optimization; it is defined by the recursion formula

$$\mathbf{B}_k = \mathbf{B}_{k-1} - \frac{1}{\mathbf{s}_{k-1}^T \mathbf{B}_{k-1} \mathbf{s}_{k-1}} \mathbf{B}_{k-1} \mathbf{s}_{k-1} \mathbf{s}_{k-1}^T \mathbf{B}_{k-1} + \frac{1}{\mathbf{s}_{k-1}^T \mathbf{y}_{k-1}} \mathbf{y}_{k-1} \mathbf{y}_{k-1}^T, \quad (2)$$

where

$$\mathbf{s}_{k-1} \equiv \mathbf{x}_k - \mathbf{x}_{k-1} \quad \text{and} \quad \mathbf{y}_{k-1} \equiv \nabla f(\mathbf{x}_k) - \nabla f(\mathbf{x}_{k-1}). \quad (3)$$

This rank-two update to $\mathbf{B}_{k-1}$ preserves positive definiteness when $\mathbf{s}_{k-1}^T \mathbf{y}_{k-1} > 0$. For large-scale problems, only $l \ll n$ of the most recent updates $\{\mathbf{s}_i, \mathbf{y}_i\}$ with

$k-l \leq i \leq k-1$ are stored in the so-called limited-memory BFGS (L-BFGS) methods. (Typically, $l \in [3,7]$ (see, e.g., [10]).) One can recursively write

$$\mathbf{B}_k = \mathbf{B}_0^{(k)} + \sum_{i=k-l}^{k-1} \left(-\frac{1}{\mathbf{s}_i^T \mathbf{B}_{i-(k-l)}^{(k)} \mathbf{s}_i} \mathbf{B}_{i-(k-l)}^{(k)} \mathbf{s}_i \mathbf{s}_i^T \mathbf{B}_{i-(k-l)}^{(k)} + \frac{1}{\mathbf{s}_i^T \mathbf{y}_i} \mathbf{y}_i \mathbf{y}_i^T \right),$$

where $\mathbf{B}_0^{(k)} \in \mathbb{R}^{n \times n}$ is a suitably-chosen initial matrix. We assume that $\mathbf{B}_0^{(k)} = \gamma_k \mathbf{I}_n$, which is a typical choice for $\mathbf{B}_0$, where $\gamma_k = \mathbf{y}_{k-1}^T \mathbf{y}_{k-1} / \mathbf{y}_{k-1}^T \mathbf{s}_{k-1}$. Alternative choices of $\mathbf{B}_0$ are proposed in [3], and the approaches from this paper still apply to these alternatives to $\mathbf{B}_0^{(k)}$.

Using the l most recently computed pairs, we define the following matrices:

$$\mathbf{S}_k \equiv [\mathbf{s}_{k-l} \ \cdots \ \mathbf{s}_{k-1}] \quad \text{and} \quad \mathbf{Y}_k \equiv [\mathbf{y}_{k-l} \ \cdots \ \mathbf{y}_{k-1}].$$

We partition the matrix $\mathbf{S}_k^T \mathbf{Y}_k$ into the sum $\mathbf{S}_k^T \mathbf{Y}_k = \mathbf{L}_k + \mathbf{T}_k$, where $\mathbf{L}_k$ is the strictly lower triangular part of $\mathbf{S}_k^T \mathbf{Y}_k$ and $\mathbf{T}_k$ is its upper-triangular part. Let $\mathbf{D}_k$ denote the diagonal of $\mathbf{S}_k^T \mathbf{Y}_k$. Then the *compact representation* of the L-BFGS matrix with the initial matrix $\mathbf{B}_0^{(k)} = \gamma_k \mathbf{I}_n$ is

$$\mathbf{B}_k = \gamma_k \mathbf{I}_n + \widehat{\mathbf{\Psi}}_k \widehat{\mathbf{\Xi}}_k \widehat{\mathbf{\Psi}}_k^T, \tag{4}$$

where

$$\widehat{\mathbf{\Psi}}_k \equiv [\mathbf{S}_k \ \ \mathbf{Y}_k] \quad \text{and} \quad \widehat{\mathbf{\Xi}}_k \equiv -\gamma_k \begin{bmatrix} \mathbf{S}_k^T \mathbf{S}_k & \mathbf{L}_k \\ \mathbf{L}_k^T & -\gamma_k \mathbf{D}_k \end{bmatrix}^{-1},$$

with $\widehat{\mathbf{\Psi}}_k \in \mathbb{R}^{n \times 2l}$ and $\widehat{\mathbf{\Xi}}_k \in \mathbb{R}^{2l \times 2l}$ (see [10] for details). Note that the inverse $\widehat{\mathbf{\Xi}}_k$ exists and is uniquely defined provided $\mathbf{s}_i^T \mathbf{y}_i > 0$ for all i [10, Theorem 2.3]. From the Sherman-Morrison-Woodbury formula, the inverse quasi-Newton matrix $\mathbf{H}_k = \mathbf{B}_k^{-1}$, with the notation $\delta_k = \gamma_k^{-1}$, is given by

$$\mathbf{H}_k = \delta_k \mathbf{I}_n + \widehat{\mathbf{\Psi}}_k \widehat{\mathbf{M}}_k \widehat{\mathbf{\Psi}}_k^T, \tag{5}$$

where

$$\widehat{\mathbf{M}}_k \equiv -(\gamma_k^2 \widehat{\mathbf{\Xi}}_k^{-1} + \gamma_k \widehat{\mathbf{\Psi}}_k^T \widehat{\mathbf{\Psi}}_k)^{-1} = -\begin{bmatrix} \mathbf{0}_{l \times l} & \gamma_k \mathbf{T}_k \\ \gamma_k \mathbf{T}_k^T & \gamma_k^2 (\mathbf{D}_k + \delta_k \mathbf{Y}_k^T \mathbf{Y}_k) \end{bmatrix}^{-1}, \tag{6}$$

with $\widehat{\mathbf{M}}_k \in \mathbb{R}^{2l \times 2l}$. Note that the methods described in this article are applicable to any quasi-Newton matrix with a compact representation, not only the L-BFGS matrix. (For examples of compact representations of other quasi-Newton matrices, see [10,6,15].)

2.2 Trust-region methods

In unconstrained minimization, the trust-region algorithm is an iterative method that defines and solves at each iteration k a quadratic subproblem of the form

$$\mathbf{s}_k = \underset{\|\mathbf{s}\| \leq \Delta_k}{\arg\min}\, Q(\mathbf{s}) = \mathbf{s}^T \mathbf{g}_k + \frac{1}{2}\mathbf{s}^T \mathbf{B}_k \mathbf{s}, \tag{7}$$

where $\mathbf{g}_k = \nabla f(\mathbf{x}_k)$, $\mathbf{B}_k \approx \nabla^2 f(\mathbf{x}_k) \in \mathbb{R}^{n \times n}$ is an approximation to the Hessian, and $\Delta_k > 0$ is the trust-region radius (see [14]). Typically, the norm used in the constraint is the Euclidean norm $\|\cdot\| = \|\cdot\|_2$. The solution $\mathbf{s}_k$ is used to compute the next iterate $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{s}_k$. For this article, we use the L-BFGS matrix for $\mathbf{B}_k$.

To use trust-region methods to solve (1), we note that if the current iterate $\mathbf{x}_k$ is feasible, i.e., $\mathbf{A}\mathbf{x}_k = \mathbf{b}$, then the next iterate $\mathbf{x}_{k+1}$ is also feasible, i.e., $\mathbf{A}\mathbf{x}_{k+1} = \mathbf{b}$, only if $\mathbf{A}\mathbf{s}_k = \mathbf{0}$. Thus, the trust-region subproblem corresponding to (1) is given by

$$\underset{\|\mathbf{s}\| \leq \Delta_k}{\text{minimize}}\;\; Q(\mathbf{s}) = \mathbf{s}^T \mathbf{g}_k + \frac{1}{2}\mathbf{s}^T \mathbf{B}_k \mathbf{s} \quad \text{subject to} \;\; \mathbf{A}\mathbf{s} = \mathbf{0}. \tag{8}$$

Without the equality constraint in (8), the trust-region subproblem can be solved to high accuracy with a Euclidean norm (see e.g., [7,17]) or a shape-changing norm [5] by exploiting the compact representation of the Hessian of $Q(\mathbf{s})$, namely $\mathbf{B}_k$, to compute the minimizer. However, these methods cannot be used directly to solve (8) with the equality constraint because the optimality conditions are different. In other words, we simply cannot extend the same approach to the Lagrangian associated with (8) because the Hessian of the Lagrangian does not have a readily available compact representation. Instead, we compute a compact representation of part of the inverse Hessian of the Lagrangian, which allows us to solve (8) efficiently and accurately using either the ℓ_2 or shape-changing norm without explicitly computing the inverse Hessian of the Lagrangian.

Our overall strategy is as follows. First, we compute the solution to (8) without the inequality constraint (Sec. 3). If this solution satisfies $\|\mathbf{s}\|_2 \leq \Delta_k$ and yields a sufficient decrease in the objective function, we use it to define the next iterate. Otherwise, we solve (8) using an ℓ_2-norm (Sec. 4) or a shape-changing norm (Sec. 5).

3 Trust-Region Subproblem Solution without an TR Constraint

When the constraint $\|\mathbf{s}\| \leq \Delta_k$ is not present, the solution of (8) can be characterized as a stationary point of the Lagrangian objective function

$$\mathcal{L}(\boldsymbol{s}, \boldsymbol{\lambda}) = Q(\mathbf{s}) + \boldsymbol{\lambda}^T \mathbf{A}\mathbf{s} = \mathbf{s}^T \mathbf{g}_k + \frac{1}{2}\mathbf{s}^T \mathbf{B}_k \mathbf{s} + \boldsymbol{\lambda}^T \mathbf{A}\mathbf{s},$$

where $\boldsymbol{\lambda} \in \mathbb{R}^m$ is a vector of Lagrange multipliers (cf. [21, Section 18.1] with linear constraints). The stationary point $(\boldsymbol{s}_e, \boldsymbol{\lambda}_e)$ is obtained by setting the gradient of the Lagrangian equal to zero, i.e., $\nabla \mathcal{L}(\boldsymbol{s}_e, \boldsymbol{\lambda}_e) = \mathbf{0}_{n+m}$. This gives rise to the Karush-Kuhn-Tucker (KKT) system

$$\begin{bmatrix} \mathbf{B}_k & \mathbf{A}^T \\ \mathbf{A} & \mathbf{0}_{m\times m} \end{bmatrix} \begin{bmatrix} \boldsymbol{s}_e \\ \boldsymbol{\lambda}_e \end{bmatrix} = \begin{bmatrix} -\mathbf{g}_k \\ \mathbf{0}_m \end{bmatrix}. \tag{9}$$

Let $\mathbf{K} \in \mathbb{R}^{(n+m)\times(n+m)}$ define the KKT matrix in (9). With $\boldsymbol{\Omega}_k \equiv (\mathbf{A}\mathbf{B}_k^{-1}\mathbf{A}^T)^{-1} \in \mathbb{R}^{m\times m}$, the inverse of $\mathbf{K}$ is given by

$$\mathbf{K}^{-1} = \begin{bmatrix} \mathbf{B}_k^{-1} - \mathbf{B}_k^{-1}\mathbf{A}^T\boldsymbol{\Omega}_k\mathbf{A}\mathbf{B}_k^{-1} & \mathbf{B}_k^{-1}\mathbf{A}^T\boldsymbol{\Omega}_k \\ (\mathbf{B}_k^{-1}\mathbf{A}^T\boldsymbol{\Omega}_k)^T & -\boldsymbol{\Omega}_k \end{bmatrix} \equiv \begin{bmatrix} \mathbf{V}_k & \mathbf{W}_k \\ \mathbf{W}_k^T & -\boldsymbol{\Omega}_k \end{bmatrix} \tag{10}$$

(see e.g., [18]). The solution of (9) is then

$$\begin{bmatrix} \boldsymbol{s}_e \\ \boldsymbol{\lambda}_e \end{bmatrix} = \mathbf{K}^{-1} \begin{bmatrix} -\mathbf{g}_k \\ \mathbf{0}_m \end{bmatrix} = \begin{bmatrix} -\mathbf{V}_k\mathbf{g}_k \\ -\mathbf{W}_k^T\mathbf{g}_k \end{bmatrix}. \tag{11}$$

Note that the equality-constrained minimizer $\boldsymbol{s}_e$ depends only on the (1,1) block of $\mathbf{K}^{-1}$. Next, we compute the compact representation of $\mathbf{V}_k$ to make computing $\boldsymbol{s}_e$ more efficient. This compact representation enables us to compute the *partial eigendecomposition* of $\mathbf{V}_k$, which we combine with a shape-changing norm in order to compute an analytic solution to (8).

3.1 Compact representation of $\mathbf{V}_k$

Using the notation $\mathbf{H}_k = \mathbf{B}_k^{-1}$, we characterize the compact representation of $\mathbf{V}_k$ using the following lemma.

Lemma 1 *The (1,1) block of the inverse KKT matrix in* (10) *has the compact representation*

$$\mathbf{V}_k = \delta_k\mathbf{I}_n + \boldsymbol{\Psi}_k\mathbf{M}_k\boldsymbol{\Psi}_k^T, \tag{12}$$

where $\boldsymbol{\Psi}_k \equiv [\mathbf{A}^T \quad \widehat{\boldsymbol{\Psi}}_k] \in \mathbb{R}^{(2l+m)\times n}$, $\mathbf{C}_k \equiv \mathbf{A}\widehat{\boldsymbol{\Psi}}_k\widehat{\mathbf{M}}_k \in \mathbb{R}^{m\times 2l}$ *and*

$$\mathbf{M}_k \equiv \begin{bmatrix} -\delta_k^2\boldsymbol{\Omega}_k & -\delta_k\boldsymbol{\Omega}_k\mathbf{C}_k \\ -\delta_k\mathbf{C}_k^T\boldsymbol{\Omega}_k & \widehat{\mathbf{M}}_k - \mathbf{C}_k^T\boldsymbol{\Omega}_k\mathbf{C}_k \end{bmatrix}. \tag{13}$$

Proof From (5), observe that

$$\mathbf{A}\mathbf{H}_k = \delta_k\mathbf{A} + \mathbf{A}\widehat{\boldsymbol{\Psi}}_k\widehat{\mathbf{M}}_k\widehat{\boldsymbol{\Psi}}_k^T = [\delta_k\mathbf{I}_m \quad \mathbf{C}_k] \begin{bmatrix} \mathbf{A} \\ \widehat{\boldsymbol{\Psi}}_k^T \end{bmatrix}. \tag{14}$$

Now, using (14),

$$\mathbf{H}_k\mathbf{A}^T\boldsymbol{\Omega}_k\mathbf{A}\mathbf{H}_k = [\mathbf{A}^T \quad \widehat{\boldsymbol{\Psi}}_k] \begin{bmatrix} \delta_k^2\boldsymbol{\Omega}_k & \delta_k\boldsymbol{\Omega}_k\mathbf{C}_k \\ \delta_k\mathbf{C}_k^T\boldsymbol{\Omega}_k & \mathbf{C}_k^T\boldsymbol{\Omega}_k\mathbf{C}_k \end{bmatrix} \begin{bmatrix} \mathbf{A} \\ \widehat{\boldsymbol{\Psi}}_k^T \end{bmatrix}.$$

Then (5) can be expressed as

$$\mathbf{H}_k = \delta_k \mathbf{I}_n + \widehat{\mathbf{\Psi}}_k \widehat{\mathbf{M}}_k \widehat{\mathbf{\Psi}}_k^T = \delta_k \mathbf{I}_n + [\mathbf{A}^T \quad \widehat{\mathbf{\Psi}}_k] \begin{bmatrix} \mathbf{0}_{m\times m} & \\ & \widehat{\mathbf{M}}_k \end{bmatrix} \begin{bmatrix} \mathbf{A} \\ \widehat{\mathbf{\Psi}}_k^T \end{bmatrix}.$$

Combining these to express $\mathbf{V}_k$ in (10) yields

$$\begin{aligned} \mathbf{V}_k &= \mathbf{H}_k - \mathbf{H}_k \mathbf{A}^T \mathbf{\Omega}_k \mathbf{A} \mathbf{H}_k \\ &= \delta_k \mathbf{I}_n + [\mathbf{A}^T \quad \widehat{\mathbf{\Psi}}_k] \begin{bmatrix} -\delta_k^2 \mathbf{\Omega}_k & -\delta_k \mathbf{\Omega}_k \mathbf{C}_k \\ -\delta_k \mathbf{C}_k^T \mathbf{\Omega}_k & \widehat{\mathbf{M}}_k - \mathbf{C}_k^T \mathbf{\Omega}_k \mathbf{C}_k \end{bmatrix} \begin{bmatrix} \mathbf{A} \\ \widehat{\mathbf{\Psi}}_k^T \end{bmatrix}, \end{aligned}$$

which is the desired result. □

Thus, the minimizer of (8) without the inequality constraint is given by

$$\boldsymbol{s}_e = -\mathbf{V}_k \mathbf{g}_k = -(\delta_k \mathbf{I}_n + \mathbf{\Psi}_k \mathbf{M}_k \mathbf{\Psi}_k^T) \mathbf{g}_k. \tag{15}$$

3.2 Computational complexity

To estimate the computational complexity in computing $\boldsymbol{s}_e$, we concentrate on the dominant number of multiplications. In particular, since $m \ll n$ and $l \ll n$, this means that we concentrate on terms that involve n.

First, we analyze the computational complexity of the matrices $\widehat{\mathbf{M}}_k$ in (6) and $\mathbf{M}_k$ in (13). In order to compute $\widehat{\mathbf{M}}_k$ in (6), the upper triangular part $\mathbf{T}_k$ of $\mathbf{S}_k^T \mathbf{Y}_k$ and $\mathbf{Y}_k^T \mathbf{Y}_k$ must be formed. However, the leading $(l-1)\times(l-1)$ blocks of $\mathbf{S}_k^T \mathbf{Y}_k$ and $\mathbf{Y}_k^T \mathbf{Y}_k$ can be obtained from $\mathbf{S}_{k-1}^T \mathbf{Y}_{k-1}$ and $\mathbf{Y}_{k-1}^T \mathbf{Y}_{k-1}$. Since we only need the upper triangular part of $\mathbf{S}_k^T \mathbf{Y}_k$ and $\mathbf{Y}_k^T \mathbf{Y}_k$ is symmetric, only the last columns of $\mathbf{S}_k^T \mathbf{Y}_k$ and $\mathbf{Y}_k^T \mathbf{Y}_k$ have to be computed. Each requires $l \cdot n$ multiplications. Then $\widehat{\mathbf{M}}_k$ in (6) can be computed by expressing the inverse explicitly as

$$\widehat{\mathbf{M}}_k = \begin{bmatrix} \mathbf{T}_k^{-T}(\mathbf{D}_k + \delta_k \mathbf{Y}_k^T \mathbf{Y}_k)\mathbf{T}_k^{-1} & -\delta_k \mathbf{T}_k^{-T} \\ -\delta_k \mathbf{T}_k^{-1} & \mathbf{0}_{l\times l} \end{bmatrix},$$

which requires $\mathcal{O}(l^3)$ multiplications to form. Thus, the dominant cost of forming $\widehat{\mathbf{M}}_k$ comes from computing the last columns of $\mathbf{S}_k^T \mathbf{Y}_k$ and $\mathbf{Y}_k^T \mathbf{Y}_k$, which requires $l \cdot n$ multiplications each.

Next, we compute $\mathbf{M}_k$ in (13) by first computing $\mathbf{C}_k = \mathbf{A}\widehat{\mathbf{\Psi}}_k \widehat{\mathbf{M}}_k = [\mathbf{A}\mathbf{S}_k \ \mathbf{A}\mathbf{Y}_k]\widehat{\mathbf{M}}_k \in \mathbb{R}^{m\times 2l}$. Note that the previous search directions $\mathbf{s}_i$ are feasible directions, $\mathbf{A}\mathbf{s}_i = \mathbf{0}$. Therefore, $\mathbf{A}\mathbf{S}_k = \mathbf{0}$. Now, the first $l-1$ columns of $\mathbf{A}\mathbf{Y}_k$ are the last $l-1$ columns of $\mathbf{A}\mathbf{Y}_{k-1}$. Thus to form $\mathbf{A}\widehat{\mathbf{\Psi}}_k$, we only need to compute $\mathbf{A}\mathbf{y}_{k-1}$, which requires mn multiplications. Then, forming $\mathbf{C}_k$ from $\mathbf{A}\widehat{\mathbf{\Psi}}_k$ and $\widehat{\mathbf{M}}_k$, requires $4l^2 m$ multiplications. Finally, using the expression for $\mathbf{H}_k$ in (5), $\mathbf{\Omega}_k = (\mathbf{A}\mathbf{H}_k\mathbf{A}^T)^{-1} = (\delta_k \mathbf{A}\mathbf{A}^T + \mathbf{C}_k \widehat{\mathbf{M}}_k^{-1} \mathbf{C}_k^T)^{-1}$. Since $\mathbf{A}$ does not change at each iteration, $\mathbf{A}\mathbf{A}^T$ is not recomputed at each iteration but rather has a one-time computational complexity of $\mathcal{O}(m^2 n)$. Then since $\mathbf{\Omega}_k \in \mathbb{R}^{m\times m}$,

inverting $\mathbf{A}\mathbf{H}_k\mathbf{A}^T$ has a computational complexity of $\mathcal{O}(m^3)$. Thus the dominant cost for forming $\mathbf{M}_k$ comes from computing the last columns of $\mathbf{A}\mathbf{Y}_k$, which requires mn multiplications.

We compute the equality-constrained minimizer $\boldsymbol{s}_e$ as

$$\boldsymbol{s}_e = -\boldsymbol{\Psi}_k(\mathbf{M}_k(\boldsymbol{\Psi}_k^T\mathbf{g}_k)) - \delta_k\mathbf{g}_k. \tag{16}$$

where $\mathbf{M}_k$ is $(2l+m)\times(2l+m)$ and $\boldsymbol{\Psi}_k$ is $n\times(2l+m)$. First forming $\boldsymbol{\Psi}_k^T\mathbf{g}_k$, then pre-multiplying by $\mathbf{M}_k$ and then by $\boldsymbol{\Psi}_k$, and finally subtracting $\delta_k\mathbf{g}_k$ leads to a computational complexity of $\mathcal{O}((2(2l+m)+1)n)$ (ignoring terms that do not depend on n). This estimate is consistent with [10, Sec. 3.1].

4 Trust-Region Subproblem Solution with an ℓ_2 TR Constraint

In this section and the next, we assume that the equality-constrained solution $\boldsymbol{s}_e$ from (11) is not feasible with respect to the ℓ_2-norm inequality constraint. That is, we assume $\|\boldsymbol{s}_e\|_2 > \Delta_k$.

If the ℓ_2-norm is used in (8), the Lagrangian for (8) is

$$\mathcal{L}(\mathbf{s},\boldsymbol{\lambda},\sigma) \;=\; Q(\mathbf{s}) + \boldsymbol{\lambda}^T\mathbf{A}\mathbf{s} + \frac{\sigma}{2}\|\mathbf{s}\|_2^2 \;=\; \mathbf{s}^T\mathbf{g}_k + \frac{1}{2}\mathbf{s}^T\mathbf{B}_k\mathbf{s} + \boldsymbol{\lambda}^T\mathbf{A}\mathbf{s} + \frac{\sigma}{2}\|\mathbf{s}\|_2^2,$$

where $\boldsymbol{\lambda}\in\mathbb{R}^m$ and $\sigma\in\mathbb{R}$ are the Lagrange multipliers. A stationary point $(\mathbf{s}^*,\boldsymbol{\lambda}^*,\sigma^*)$ of $\mathcal{L}(\mathbf{s},\boldsymbol{\lambda},\sigma)$ must satisfy

$$\begin{bmatrix}(\mathbf{B}_k+\sigma\mathbf{I}_n) & \mathbf{A}^T\\ \mathbf{A} & \mathbf{0}_{m\times m}\end{bmatrix}\begin{bmatrix}\mathbf{s}\\ \boldsymbol{\lambda}\end{bmatrix} = \begin{bmatrix}-\mathbf{g}_k\\ \mathbf{0}_m\end{bmatrix}. \tag{17}$$

Letting $\mathbf{H}_k(\sigma) = (\mathbf{B}_k+\sigma\mathbf{I}_n)^{-1}$ and $\boldsymbol{\Omega}_k(\sigma) = (\mathbf{A}\mathbf{H}_k(\sigma)\mathbf{A}^T)^{-1}$, we see from calculations similar to (11) that the stationary point $(\mathbf{s}^*,\boldsymbol{\lambda}^*,\sigma^*)$ must satisfy

$$\begin{bmatrix}\mathbf{s}^*\\ \boldsymbol{\lambda}^*\end{bmatrix} = \begin{bmatrix}-\mathbf{V}_k(\sigma^*)\mathbf{g}_k\\ -\mathbf{W}_k^T(\sigma^*)\mathbf{g}_k\end{bmatrix},$$

where

$$\begin{aligned}\mathbf{V}_k(\sigma) &= \mathbf{H}_k(\sigma) - \mathbf{H}_k(\sigma)\mathbf{A}^T\boldsymbol{\Omega}_k(\sigma)\mathbf{A}\mathbf{H}_k(\sigma),\\ \mathbf{W}_k(\sigma) &= \mathbf{H}_k(\sigma)\mathbf{A}^T\boldsymbol{\Omega}_k(\sigma).\end{aligned} \tag{18}$$

Thus we can obtain the optimal primal and dual solutions, $\mathbf{s}^*$ and $\boldsymbol{\lambda}^*$ if we compute the optimal Lagrange multiplier σ^*.

4.1 Computing σ^*

By computing the inverse of the KKT matrix in (17), we observe from (18) that $\mathbf{s}$ depends on σ in the following manner:

$$\mathbf{s}(\sigma) = -\mathbf{H}_k(\sigma)\mathbf{\Phi}_k(\sigma)\mathbf{g}_k, \tag{19}$$

where $\mathbf{\Phi}_k(\sigma) = \mathbf{I}_n - \mathbf{A}^T\mathbf{\Omega}_k(\sigma)\mathbf{A}\mathbf{H}_k(\sigma)$.

Since $\|\boldsymbol{s}_e\|_2 > \Delta_k$, the solution to (8) must lie on the boundary, i.e., $\|\mathbf{s}^*\|_2 = \Delta_k$. Since $\mathbf{s}$ explicitly depends on σ (see (19)), the optimal Lagrange multiplier σ^* can be obtained by finding the zero of the function $\psi(\sigma) = \|\mathbf{s}(\sigma)\|_2 - \Delta_k$, or equivalently, the zero of

$$\phi(\sigma) \equiv \frac{1}{\|\mathbf{s}(\sigma)\|_2} - \frac{1}{\Delta_k}.$$

The following theorem guarantees that σ^* can be obtained using Newton's method.

Theorem 1 *Newton's method applied to $\phi(\sigma)$ with initial iterate $\sigma_0 = 0$ is guaranteed to converge to σ^* monotonically.*

Proof First note that $\phi(0) < 0$ because we assume $\|\mathbf{s}(0)\|_2 > \Delta_k$. To apply Newton's method to find the zero σ^* of $\phi(\sigma)$, we note that the derivative of $\phi(\sigma)$ is

$$\phi'(\sigma) = -\frac{\mathbf{s}(\sigma)^T\mathbf{s}'(\sigma)}{\|\mathbf{s}(\sigma)\|_2^3},$$

where $\mathbf{s}'(\sigma)$ represents the derivative of $\mathbf{s}(\sigma)$ [14, Lemma 7.3.1]. The vector $\mathbf{s}'(\sigma)$ is computed by differentiating system (17) with respect to σ and solving the resulting equations for $\mathbf{s}'(\sigma)$:

$$\begin{bmatrix} (\mathbf{B}_k + \sigma\mathbf{I}_n) & \mathbf{A}^T \\ \mathbf{A} & \mathbf{0}_{m\times m} \end{bmatrix} \begin{bmatrix} \mathbf{s}'(\sigma) \\ \boldsymbol{\lambda}'(\sigma) \end{bmatrix} = \begin{bmatrix} -\mathbf{s}(\sigma) \\ \mathbf{0}_m \end{bmatrix}. \tag{20}$$

Using the same calculations as obtaining (19) from (17), we have that

$$\mathbf{s}'(\sigma) = -\mathbf{H}_k(\sigma)\mathbf{\Phi}_k(\sigma)\mathbf{s}(\sigma). \tag{21}$$

Noting that $\mathbf{\Phi}_k(\sigma)^T\mathbf{H}_k(\sigma)\mathbf{\Phi}_k(\sigma) = \mathbf{H}_k(\sigma)\mathbf{\Phi}_k(\sigma)$, we observe that

$$-\mathbf{s}(\sigma)^T\mathbf{s}'(\sigma) = \mathbf{s}(\sigma)^T\mathbf{\Phi}_k(\sigma)^T\mathbf{H}_k(\sigma)\mathbf{\Phi}_k(\sigma)\mathbf{s}(\sigma) \geq 0,$$

because $\mathbf{H}_k(\sigma)$ is positive definite for $\sigma \geq 0$. Therefore, $\phi(\sigma)$ is non-decreasing in the interval $[0, \infty)$. Next, we show that $\phi(\sigma)$ is concave in $[0, \infty)$ by showing that $\phi''(\sigma) \leq 0$. We obtain an expression for $\phi''(\sigma)$ by differentiating the linear system in (20), which yields

$$\begin{bmatrix} (\mathbf{B}_k + \sigma\mathbf{I}_n) & \mathbf{A}^T \\ \mathbf{A} & \mathbf{0}_{m\times m} \end{bmatrix} \begin{bmatrix} \mathbf{s}''(\sigma) \\ \boldsymbol{\lambda}''(\sigma) \end{bmatrix} = \begin{bmatrix} -2\mathbf{s}'(\sigma) \\ \mathbf{0}_m \end{bmatrix}, \tag{22}$$

and hence

$$\mathbf{s}''(\sigma) = -2\mathbf{H}_k(\sigma)\mathbf{\Phi}_k(\sigma)\mathbf{s}'(\sigma),$$

so that

$$\begin{aligned}\mathbf{s}(\sigma)^T\mathbf{s}''(\sigma) &= -2\mathbf{s}(\sigma)^T\mathbf{H}_k(\sigma)\mathbf{\Phi}_k(\sigma)\mathbf{s}'(\sigma)\\ &= -2\mathbf{s}(\sigma)^T\mathbf{\Phi}_k(\sigma)^T\mathbf{H}_k(\sigma)\mathbf{\Phi}_k(\sigma)\mathbf{s}'(\sigma)\\ &= \ 2\mathbf{s}'(\sigma)^T\mathbf{s}'(\sigma).\end{aligned}$$

Thus,

$$\begin{aligned}\phi''(\sigma) &= -\frac{\|\mathbf{s}(\sigma)\|_2^3\big(\|\mathbf{s}'(\sigma)\|_2^2 + \mathbf{s}(\sigma)^T\mathbf{s}''(\sigma)\big) - 3(\mathbf{s}(\sigma)^T\mathbf{s}'(\sigma))^2\|\mathbf{s}(\sigma)\|_2}{\|\mathbf{s}(\sigma)\|_2^6}\\ &= 3\frac{(\mathbf{s}(\sigma)^T\mathbf{s}'(\sigma))^2 - \|\mathbf{s}(\sigma)\|_2^2\|\mathbf{s}'(\sigma)\|_2^2}{\|\mathbf{s}(\sigma)\|_2^5}\\ &\leq 0\end{aligned}$$

by the Cauchy-Schwartz inequality. Next, we show that $\phi(\sigma) \to \infty$ as $\sigma \to \infty$. Let $\lambda_{\max} > 0$ and $\lambda_{\min} > 0$ be the largest and smallest eigenvalues of $\mathbf{B}_k$, and let $\sigma_{\max}$ and $\sigma_{\min}$ be the largest and smallest singular values of $\mathbf{A}$. Note that

$$\begin{aligned}\|\mathbf{s}(\sigma)\|_2 &= \|\mathbf{H}_k(\sigma)\mathbf{\Phi}_k(\sigma)\mathbf{g}_k\|_2\\ &\leq \|\mathbf{H}_k(\sigma)\|_2\|\mathbf{\Phi}_k(\sigma)\|_2\|\mathbf{g}_k\|_2\\ &\leq \frac{1}{\lambda_{\min}+\sigma}\left(1 + \|\mathbf{A}^T\|_2\|\mathbf{\Omega}_k(\sigma)\|_2\|\mathbf{A}\|_2\|\mathbf{H}_k(\sigma)\|_2\right)\|\mathbf{g}_k\|_2\\ &\leq \frac{1}{\lambda_{\min}+\sigma}\left(1 + \sigma_{\max}^2\frac{\lambda_{\max}+\sigma}{\sigma_{\min}^2}\frac{1}{\lambda_{\min}+\sigma}\right)\|\mathbf{g}_k\|_2,\end{aligned}$$

which tends to 0 as $\sigma \to \infty$, which implies that $\phi(\sigma) \to \infty$ as $\sigma \to \infty$. Since $\phi''(\sigma) \leq 0$ in $[0, \infty)$, $\phi'(\sigma)$ must be non-increasing in $[0, \infty)$. If $\phi'(\hat{\sigma}) = 0$ for some $\hat{\sigma} > 0$, then $\phi'(\sigma) = 0$ for all $\sigma > \hat{\sigma}$ because $\phi'(\sigma)$ is non-increasing and $\phi'(\sigma) \geq 0$. However, this cannot happen because $\phi(\sigma) \to \infty$ as $\sigma \to \infty$. Thus, $\phi'(\sigma) > 0$ for all σ in $[0, \infty)$, and every iteration of Newton's method

$$\sigma_{j+1} = \sigma_j - \frac{\phi(\sigma_j)}{\phi'(\sigma_j)}$$

is well-defined. Finally, since $\phi(0) < 0$ and $\phi(\sigma)$ is strictly increasing and concave with $\phi(\sigma) \to \infty$ as $\sigma \to \infty$, Newton's method will converge monotonically to σ^* with the initial point $\sigma_0 = 0$. □

Thus, given the optimal Lagrange multiplier σ^*, the minimizer is given by

$$\mathbf{s}(\sigma^*) = -\mathbf{V}_k(\sigma^*)\mathbf{g}_k = -\mathbf{H}_k(\sigma^*)\mathbf{\Phi}_k(\sigma^*)\mathbf{g}_k. \tag{23}$$

As in Sec. 3.1 eqs. (12)–(13), we compute the compact representation of $\mathbf{V}_k(\sigma^*)$ to efficiently compute the solution $\mathbf{s}(\sigma^*)$.

4.2 Compact representation of $\mathbf{V}_k(\sigma)$

Like the compact representation of $\mathbf{H}_k$ in (5)–(6), the matrix $\mathbf{H}_k(\sigma)$ has the compact representation

$$\mathbf{H}_k(\sigma) = \frac{1}{\gamma_k + \sigma}\mathbf{I}_n + \widehat{\mathbf{\Psi}}_k \widehat{\mathbf{M}}_k(\sigma) \widehat{\mathbf{\Psi}}_k^T,$$

where

$$\widehat{\mathbf{M}}_k(\sigma) = -\big((\gamma_k + \sigma)^2 \widehat{\mathbf{\Xi}}_k^{-1} + (\gamma_k + \sigma)\widehat{\mathbf{\Psi}}_k^T \widehat{\mathbf{\Psi}}_k\big)^{-1}.$$

We can obtain the compact representation of $\mathbf{V}_k(\sigma)$ using the following corollary to Lemma 1.

Corollary 1 *The compact representation of* $\mathbf{V}_k(\sigma)$ *from* (18) *is given by*

$$\mathbf{V}_k(\sigma) = \tau_k \mathbf{I}_n + \mathbf{\Psi}_k \mathbf{M}_k(\sigma) \mathbf{\Psi}_k^T, \tag{24}$$

where $\tau_k = (\gamma_k + \sigma)^{-1}$, $\mathbf{\Psi}_k \equiv [\mathbf{A}^T \quad \widehat{\mathbf{\Psi}}_k]$, *and*

$$\mathbf{M}_k(\sigma) \equiv \begin{bmatrix} -\tau_k^2 \mathbf{\Omega}_k(\sigma) & -\tau_k \mathbf{\Omega}_k(\sigma)\mathbf{C}_k \\ -\tau_k \mathbf{C}_k^T \mathbf{\Omega}_k(\sigma) & \widehat{\mathbf{M}}_k(\sigma) - \mathbf{C}_k^T \mathbf{\Omega}_k(\sigma)\mathbf{C}_k \end{bmatrix}.$$

Given the compact representation of $\mathbf{V}_k(\sigma)$, we can efficiently compute the solution $\mathbf{s}_{\ell_2}^*$ to (8) with the ℓ_2-norm by

$$\mathbf{s}_{\ell_2}^* = -\mathbf{V}_k(\sigma^*)\mathbf{g}_k = -(\tau_k^* \mathbf{I}_n + \mathbf{\Psi}_k \mathbf{M}_k(\sigma^*)\mathbf{\Psi}_k^T)\mathbf{g}_k, \tag{25}$$

where $\tau_k^* = (\gamma_k + \sigma^*)^{-1}$.

Our approach using the ℓ_2-norm constraint is summarized in Algorithm 1.

4.3 Computational complexity

Algorithm 1 first computes the equality-constrained minimizer (line 2). The computational complexity of this step is described in Section 3.2.

When the trust-region constraint is active, $\mathbf{s}_k(\sigma)$ and $\mathbf{s}_k'(\sigma)$ are computed on lines 13 and 14 for new values of σ in Newton's method. Corollary 1 implies that $\mathbf{s}_k(\sigma)$ and $\mathbf{s}_k'(\sigma)$ are computed by the same computational complexity as the equality-constrained minimizer; however, now two vectors instead of one are computed. Thus the dominant cost for one iteration of Newton's method is $2\mathcal{O}(4(l+m)n + n)$. This computational complexity is still linear in n, and typically Newton's method converges in few iterations [4,7]. The main advantage of Algorithm 1 is that it computes trust-region subproblem solutions to high accuracy, even when n becomes large. Alternative methods that compute nearly exact solutions of the trust-region subproblem, such as [20], use direct factorizations of the matrix $\mathbf{B}_k$, which become prohibitively expensive for large n. On the other hand, for large-scale problems iterative techniques are used to solve the trust-region subproblem [13,24]; however, these only compute approximate solutions. In this regard the proposed method reveals its strength, because it is a high-accuracy method for large-scale problems.

Algorithm 1: LTRL2-LEC (Limited-Memory Trust-Region ℓ_2-norm for Linear Equality Constraints)

Initialize: $0 \leq c_1$, $0 < c_2, c_3, c_4, c_5, c_6 < 1 < c_7$, $0 < \varepsilon_1, \varepsilon_2$, $k = 0$, $3 \leq l \leq 7$, $\Delta_k = \|\mathbf{x}_k\|_2$, $\mathbf{g}_k = \nabla f(\mathbf{x}_k)$, $\delta_k = 1/\|\mathbf{\Phi}_k(0)\mathbf{g}_k\|_2$, $\gamma_k = \delta_k^{-1}$, $\mathbf{\Psi}_k = \mathbf{A}^T$, $\mathbf{M}_k = \delta_k(\mathbf{A}\mathbf{A}^T)^{-1}$, $0 < i_{\max}$

while $(\varepsilon_1 \leq \|\mathbf{g}_k - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A}\mathbf{g}_k\|_2 / \max(1, \|\mathbf{x}_k\|_2))$ **do**
  $\mathbf{s}_k = -\left(\delta_k\mathbf{I} + \mathbf{\Psi}_k\mathbf{M}_k\mathbf{\Psi}_k^T\right)\mathbf{g}_k$; `/* Equality constrained step */`
  $\rho_k = 0$;
  if $\|\mathbf{s}_k\|_2 \leq \Delta_k$ **then**
    $\rho_k = (f(\mathbf{x}_k) - f(\mathbf{x}_k + \mathbf{s}_k))/(Q(\mathbf{0}) - Q(\mathbf{s}_k))$;
  end
  if $\rho_k \leq c_1$ **then**
    repeat
      $\sigma = 0$, $i = 0$;
      $\mathbf{s}_k'(\sigma) = -(\delta_k\mathbf{I} + \mathbf{\Psi}_k\mathbf{M}_k\mathbf{\Psi}_k^T)\mathbf{s}_k$;
      while $\varepsilon < |\phi(\sigma)|$ **and** $i < i_{\max}$ **do** `/* Newton's method */`
        $\sigma = \sigma - \phi(\sigma)/\phi'(\sigma)$;
        $\mathbf{s}_k = \mathbf{s}_k(\sigma) = -((\gamma_k + \sigma)^{-1}\mathbf{I} + \mathbf{\Psi}_k\mathbf{M}_k(\sigma)\mathbf{\Psi}_k^T)\mathbf{g}_k$;
        $\mathbf{s}_k' = \mathbf{s}_k'(\sigma) = -((\gamma_k + \sigma)^{-1}\mathbf{I} + \mathbf{\Psi}_k\mathbf{M}_k(\sigma)\mathbf{\Psi}_k^T)\mathbf{s}_k$;
        $i = i + 1$;
      end
      $\rho_k = 0$;
      if $0 < (f(\mathbf{x}_k) - f(\mathbf{x}_k + \mathbf{s}_k))$ **then**
        $\rho_k = (f(\mathbf{x}_k) - f(\mathbf{x}_k + \mathbf{s}_k))/(Q(\mathbf{0}) - Q(\mathbf{s}_k))$;
      end
      if $\rho_k \leq c_2$ **then**
        $\Delta_k = \min(c_3\|\mathbf{s}_k\|_2, c_4\Delta_k)$;
      end
    until $c_1 < \rho_k$;
  end
  $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{s}_k$; `/* Accept step */`
  if $c_5\Delta_k \leq \|\mathbf{s}_k\|_2$ **and** $c_6 \leq \rho_k$ **then**
    $\Delta_k = c_7\Delta_k$;
  end
  $\mathbf{g}_{k+1} = \nabla f(\mathbf{x}_{k+1})$ and $\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$;
  $\delta_{k+1} = \mathbf{y}_k^T\mathbf{s}_k / \mathbf{s}_k^T\mathbf{s}_k$ and $\gamma_{k+1} = \delta_{k+1}^{-1}$;
  Update $\mathbf{\Psi}_k, \mathbf{M}_k$ from (12), $k = k + 1$;
end

5 Trust-Region Subproblem Solution with a Shape-Changing TR Constraint

In this section, we use an alternative to the Euclidean norm to define the trust-region in (8). The main benefit of this approach is that it provides analytical solutions to the TR subproblems that are numerically cheaper than the l_2 counterparts without affecting the convergence properties. More specifically, we employ the so-called *shape-changing* norm introduced in [5]. We describe how to transform (8) by a change of variables to solve it more easily using this shape-changing norm, which is based on a partial eigendecomposition of $\mathbf{V}_k$.

5.1 Partial eigendecomposition of $\mathbf{V}_k$

Our approach to computing the eigendecomposition of $\mathbf{V}_k = \delta_k \mathbf{I} + \mathbf{\Psi}_k \mathbf{M}_k \mathbf{\Psi}_k^T$ in (12) starts with factoring the low-rank matrix

$$\mathbf{U}_k \equiv -\mathbf{\Psi}_k \mathbf{M}_k \mathbf{\Psi}_k^T.$$

Note that since $\text{Rank}(\mathbf{U}_k) = 2l + m$, the matrix $\mathbf{U}_k$ has only $2l + m$ non-zero eigenvalues. Let $\mathbf{Q}_3 \in \mathbb{R}^{n \times (n-(2l+m))}$ be a matrix whose columns form an orthonormal basis for $\text{Null}(\mathbf{U}_k)$. In other words, $\mathbf{Q}_3$ satisfies $\mathbf{U}_k \mathbf{Q}_3 = \mathbf{0}$. Note that since $\mathbf{Q}_3$ is prohibitively expensive to compute for large n, we do not explicitly compute it in our approach. Next, observe that $\mathbf{0} = \mathbf{V}_k \mathbf{A}^T = (\delta_k \mathbf{I} - \mathbf{U}_k)\mathbf{A}^T$ so that $\mathbf{U}_k \mathbf{A}^T = \delta_k \mathbf{A}^T$. Thus, if $\mathbf{A}^T = \mathbf{Q}_1 \mathbf{R}_1$ is the "thin" QR decomposition of $\mathbf{A}^T$, the eigendecomposition of $\mathbf{U}_k$ is

$$\mathbf{U}_k = -[\mathbf{A}^T \ \widehat{\mathbf{\Psi}}_k] \mathbf{M}_k \begin{bmatrix} \mathbf{A} \\ \widehat{\mathbf{\Psi}}_k^T \end{bmatrix} = [\mathbf{Q}_1 \ \mathbf{Q}_2 \ \mathbf{Q}_3] \begin{bmatrix} \delta_k \mathbf{I}_m & & \\ & \widehat{\mathbf{\Lambda}}_k & \\ & & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{Q}_1^T \\ \mathbf{Q}_2^T \\ \mathbf{Q}_3^T \end{bmatrix}, \tag{26}$$

where $\widehat{\mathbf{\Lambda}}_k \in \mathbb{R}^{2l \times 2l}$ is a diagonal matrix of eigenvalues, and the columns of the matrices $\mathbf{Q}_1 \in \mathbb{R}^{n \times m}$, $\mathbf{Q}_2 \in \mathbb{R}^{n \times 2l}$, and $\mathbf{Q}_3 \in \mathbb{R}^{n \times (n-(2l+m))}$ represent mutually orthogonal eigenvectors. Note that even though we define the large orthogonal matrix $\mathbf{Q} \equiv [\mathbf{Q}_1 \ \mathbf{Q}_2 \ \mathbf{Q}_3]$, our approach only explicitly computes the matrix $\mathbf{Q}_2$.

Next we now describe how to compute the non-zero eigenvalues in $\widehat{\mathbf{\Lambda}}_k$ and its corresponding eigenvectors in $\mathbf{Q}_2$. Subsequently, we combine the results to arrive at the implicit eigendecomposition of (12). First, note that

$$\delta_k \mathbf{Q}_1 \mathbf{Q}_1^T = \delta_k \mathbf{A}^T \mathbf{R}_1^{-1} \mathbf{R}_1^{-T} \mathbf{A} = \delta_k \mathbf{A}^T (\mathbf{A}\mathbf{A}^T)^{-1} \mathbf{A}^T.$$

Using the expression for $\mathbf{M}_k$ in (13), we can write (26) as

$$[\mathbf{A}^T \ \widehat{\mathbf{\Psi}}_k] \begin{bmatrix} -\delta_k^2 \mathbf{\Omega}_k - \delta_k \left(\mathbf{A}\mathbf{A}^T\right)^{-1} & -\delta_k \mathbf{\Omega}_k \mathbf{C}_k \\ -\delta_k \mathbf{C}_k^T \mathbf{\Omega}_k & \widehat{\mathbf{M}}_k - \mathbf{C}_k^T \mathbf{\Omega}_k \mathbf{C}_k \end{bmatrix} \begin{bmatrix} \mathbf{A} \\ \widehat{\mathbf{\Psi}}_k^T \end{bmatrix} = \mathbf{Q}_2 \widehat{\mathbf{\Lambda}}_k \mathbf{Q}_2^T. \tag{27}$$

Pre- and post-multiplying both sides of (27) by the orthogonal projection $\mathbf{P} = \mathbf{I} - \mathbf{A}^T (\mathbf{A}\mathbf{A}^T)^{-1} \mathbf{A}$ yields

$$\mathbf{P} \widehat{\mathbf{\Psi}}_k \left(\widehat{\mathbf{M}}_k - \mathbf{C}_k^T \mathbf{\Omega}_k \mathbf{C}_k \right) \widehat{\mathbf{\Psi}}_k^T \mathbf{P}^T = \mathbf{Q}_2 \widehat{\mathbf{\Lambda}}_k \mathbf{Q}_2^T,$$

because $\mathbf{P}\mathbf{A}^T = \mathbf{0}$ and $\mathbf{A}\mathbf{Q}_2 = \mathbf{0}$. To compute the eigendecomposition of the left-hand side, we first compute the "thin" QR factorization $\mathbf{P}\widehat{\mathbf{\Psi}}_k = \widehat{\mathbf{Q}}_2 \widehat{\mathbf{R}}_2$, and then we compute the eigendecomposition $\widehat{\mathbf{R}}_2 (\widehat{\mathbf{M}}_k - \mathbf{C}_k^T \mathbf{\Omega}_k \mathbf{C}_k) \widehat{\mathbf{R}}_2^T = \widehat{\mathbf{V}}_2 \widehat{\mathbf{\Lambda}}_k \widehat{\mathbf{V}}_2^T$ so that

$$\begin{aligned} \mathbf{P} \widehat{\mathbf{\Psi}}_k \left(\widehat{\mathbf{M}}_k - \mathbf{C}_k^T \mathbf{\Omega}_k \mathbf{C}_k \right) \widehat{\mathbf{\Psi}}_k^T \mathbf{P}^T &= \widehat{\mathbf{Q}}_2 \widehat{\mathbf{R}}_2 \left(\widehat{\mathbf{M}}_k - \mathbf{C}_k^T \mathbf{\Omega}_k \mathbf{C}_k \right) \widehat{\mathbf{R}}_2^T \widehat{\mathbf{Q}}_2^T \\ &= \widehat{\mathbf{Q}}_2 \widehat{\mathbf{V}}_2 \widehat{\mathbf{\Lambda}}_k \widehat{\mathbf{V}}_2^T \widehat{\mathbf{Q}}_2^T. \end{aligned}$$

By letting $\mathbf{Q}_2 = \widehat{\mathbf{Q}}_2\widehat{\mathbf{V}}_2$, we obtain the eigendecomposition

$$\begin{aligned}\mathbf{V}_k &= \delta_k\mathbf{I} + [\mathbf{A}^T \ \widehat{\mathbf{\Psi}}_k]\,\mathbf{M}_k\begin{bmatrix}\mathbf{A}\\ \widehat{\mathbf{\Psi}}_k^T\end{bmatrix}\\ &= [\mathbf{Q}_1 \ \mathbf{Q}_2 \ \mathbf{Q}_3]\begin{bmatrix}\mathbf{0}_m & & \\ & \delta_k\mathbf{I}_{2l} - \widehat{\mathbf{\Lambda}}_k & \\ & & \delta_k\mathbf{I}_{n-(2l+m)}\end{bmatrix}\begin{bmatrix}\mathbf{Q}_1^T\\ \mathbf{Q}_2^T\\ \mathbf{Q}_3^T\end{bmatrix}\\ &= \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T. \end{aligned} \tag{28}$$

Observe that the eigendecomposition only requires the computation of a few eigenvalues ($2l$), corresponding to $(\delta_k\mathbf{I}_{2l} - \widehat{\mathbf{\Lambda}}_k)$ explicitly. Moreover, we do not compute $\mathbf{Q}_1$ and $\mathbf{Q}_3$ explicitly.

5.2 Transforming the trust-region subproblem

Given the eigendecomposition of $\mathbf{V}_k$, we now perform a change of variables that allows $Q(\mathbf{s})$ in (8) to become separable. Letting $\mathbf{s} = \mathbf{Q}\mathbf{z}$, where $\mathbf{z} = [\mathbf{z}_1^T \ \mathbf{z}_2^T \ \mathbf{z}_3^T]^T$ so that $\mathbf{s} = \mathbf{Q}_1\mathbf{z}_1 + \mathbf{Q}_2\mathbf{z}_2 + \mathbf{Q}_3\mathbf{z}_3$, the trust-region subproblem (8) has the equivalent expression

$$\underset{\|\mathbf{Q}\mathbf{z}\|\le\Delta_k}{\text{minimize}}\ Q(\mathbf{Q}\mathbf{z}) = (\mathbf{Q}\mathbf{z})^T\mathbf{g}_k + \frac{1}{2}(\mathbf{Q}\mathbf{z})^T\mathbf{B}_k\mathbf{Q}\mathbf{z} \quad \text{subject to} \quad \mathbf{A}\mathbf{Q}\mathbf{z} = \mathbf{0}. \tag{29}$$

Since $\mathbf{Q}_1, \mathbf{Q}_2$, and $\mathbf{Q}_3$ are mutually orthogonal and $\mathbf{A}^T = \mathbf{Q}_1\mathbf{R}_1$, the linear equality constraint $\mathbf{A}\mathbf{Q}\mathbf{z} = \mathbf{0}$ becomes

$$\mathbf{0} = \mathbf{A}\mathbf{Q}\mathbf{z} = [\mathbf{A}\mathbf{Q}_1 \quad \mathbf{A}\mathbf{Q}_2 \quad \mathbf{A}\mathbf{Q}_3]\,\mathbf{z} = \mathbf{A}\mathbf{Q}_1\mathbf{z}_1 = \mathbf{R}_1^T\mathbf{z}_1.$$

We assume here that $\mathbf{A}^T$ has full column rank, so that $\mathbf{R}_1$ must be nonsingular. Thus the equality constraint $\mathbf{A}\mathbf{s} = \mathbf{A}\mathbf{Q}\mathbf{z} = \mathbf{0}$ is equivalent to

$$\mathbf{z}_1 = \mathbf{0}. \tag{30}$$

With this constraint on $\mathbf{z}_1$, the quadratic term in the objective function $Q(\mathbf{Q}\mathbf{z})$ in (29) becomes

$$(\mathbf{Q}\mathbf{z})^T\mathbf{B}_k\mathbf{Q}\mathbf{z} = \begin{bmatrix}\mathbf{z}_2^T \ \mathbf{z}_3^T\end{bmatrix}\begin{bmatrix}\mathbf{Q}_2^T\\ \mathbf{Q}_3^T\end{bmatrix}\mathbf{B}_k\begin{bmatrix}\mathbf{Q}_2 \ \mathbf{Q}_3\end{bmatrix}\begin{bmatrix}\mathbf{z}_2\\ \mathbf{z}_3\end{bmatrix}.$$

The following lemma further simplifies $Q(\mathbf{Q}\mathbf{z})$.

Lemma 2 *The identity*

$$\begin{bmatrix}\mathbf{Q}_2^T\\ \mathbf{Q}_3^T\end{bmatrix}\mathbf{B}_k\begin{bmatrix}\mathbf{Q}_2 \ \mathbf{Q}_3\end{bmatrix} = \begin{bmatrix}\delta_k\mathbf{I}_{2l} - \widehat{\mathbf{\Lambda}}_k & \\ & \delta_k\mathbf{I}_{n-(2l+m)}\end{bmatrix}^{-1}$$

holds for the eigenvectors and eigenvalues in (28).

Proof From (10), note that $\mathbf{B}_k\mathbf{V}_k + \mathbf{A}^T\mathbf{W}_k^T = \mathbf{I}$. Then

$$\begin{aligned}\mathbf{I} &= \begin{bmatrix}\mathbf{Q}_2^T\\ \mathbf{Q}_3^T\end{bmatrix}\left(\mathbf{B}_k\mathbf{V}_k + \mathbf{A}^T\mathbf{W}_k^T\right)\begin{bmatrix}\mathbf{Q}_2 & \mathbf{Q}_3\end{bmatrix}\\ &= \left(\begin{bmatrix}\mathbf{Q}_2^T\\ \mathbf{Q}_3^T\end{bmatrix}\mathbf{B}_k\mathbf{V}_k + \begin{bmatrix}\mathbf{Q}_2^T\\ \mathbf{Q}_3^T\end{bmatrix}\mathbf{A}^T\mathbf{W}_k^T\right)\begin{bmatrix}\mathbf{Q}_2 & \mathbf{Q}_3\end{bmatrix}\\ &= \begin{bmatrix}\mathbf{Q}_2^T\\ \mathbf{Q}_3^T\end{bmatrix}\mathbf{B}_k\mathbf{V}_k\begin{bmatrix}\mathbf{Q}_2 & \mathbf{Q}_3\end{bmatrix}\end{aligned}$$

because $\mathbf{A}^T = \mathbf{Q}_1\mathbf{R}_1$ and therefore $\mathbf{Q}_2^T\mathbf{A}^T = \mathbf{0}$ and $\mathbf{Q}_3^T\mathbf{A}^T = \mathbf{0}$. Then using $\mathbf{V}_k\mathbf{Q} = \mathbf{Q}\boldsymbol{\Lambda}$ from (28), we obtain

$$\mathbf{I} = \begin{bmatrix}\mathbf{Q}_2^T\\ \mathbf{Q}_3^T\end{bmatrix}\mathbf{B}_k\begin{bmatrix}\mathbf{Q}_2 & \mathbf{Q}_3\end{bmatrix}\begin{bmatrix}\delta_k\mathbf{I}_{2l} - \widehat{\boldsymbol{\Lambda}}_k & \\ & \delta_k\mathbf{I}_{n-(2l+m)}\end{bmatrix},$$

which is equivalent to the desired result. □

Thus, the objective function $Q(\mathbf{Qz})$ can now be written as

$$q(\mathbf{z}_2, \mathbf{z}_3) = \mathbf{z}_2^T\mathbf{Q}_2^T\mathbf{g}_k + \frac{1}{2}\mathbf{z}_2^T(\delta_k\mathbf{I} - \widehat{\boldsymbol{\Lambda}}_k)^{-1}\mathbf{z}_2 + \mathbf{z}_3^T\mathbf{Q}_3^T\mathbf{g}_k + \frac{1}{2}\delta_k^{-1}\mathbf{z}_3^T\mathbf{z}_3,$$

which is separable in $\mathbf{z}_2$ and $\mathbf{z}_3$. Next, we describe a shape-changing norm that allows the trust-region subproblem in (8) to be separated into two subproblems that can be easily solved.

5.3 Shape-changing norm

Given the eigendecomposition $\mathbf{V}_k = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^T$ and the matrices $\mathbf{Q}_{\|} \equiv [\ \mathbf{Q}_1\ \ \mathbf{Q}_2\ ]$ and $\mathbf{Q}_{\perp} \equiv \mathbf{Q}_3$, we use the shape-changing norm given by

$$\|\mathbf{s}\|_{\mathbf{Q}} \equiv \max\left(\|\mathbf{Q}_{\|}^T\mathbf{s}\|_\infty, \|\mathbf{Q}_{\perp}^T\mathbf{s}\|_2\right), \tag{31}$$

which was one of two norms proposed in [5]. In our work, we focus only on the norm in (31) because it obtained the best results in the numerical experiments presented in [5, Section 9]. To be consistent with the notation in [5], we define

$$\mathbf{Q}^T\mathbf{s} = \begin{bmatrix}\mathbf{Q}_1^T\mathbf{s}\\ \mathbf{Q}_2^T\mathbf{s}\\ \mathbf{Q}_3^T\mathbf{s}\end{bmatrix} = \begin{bmatrix}\mathbf{z}_1\\ \mathbf{z}_2\\ \mathbf{z}_3\end{bmatrix} \equiv \begin{bmatrix}\mathbf{0}\\ \mathbf{z}_{\|}\\ \mathbf{z}_{\perp}\end{bmatrix}.$$

Therefore,

$$\|\mathbf{s}\|_{\mathbf{Q}} \equiv \max\left(\|\mathbf{z}_{\|}\|_\infty, \|\mathbf{z}_{\perp}\|_2\right), \tag{32}$$

and the inequality constraint $\|\mathbf{s}\|_{\mathbf{Q}} \leq \Delta_k$ is equivalent to requiring both

$$\|\mathbf{z}_{\|}\|_\infty \leq \Delta_k \quad \text{and} \quad \|\mathbf{z}_{\perp}\|_2 \leq \Delta_k.$$

Using this shape-changing norm, we can now express the original trust-region subproblem as

$$\underset{\|\mathbf{z}_\parallel\|_\infty \le \Delta_k, \|\mathbf{z}_\perp\|_2 \le \Delta_k}{\text{minimize}} \; q(\mathbf{z}_\parallel, \mathbf{z}_\perp),$$

which separates into the following two subproblems:

$$\underset{\|\mathbf{z}_\parallel\|_\infty \le \Delta_k}{\text{minimize}} \quad q_\parallel(\mathbf{z}_\parallel) = \mathbf{z}_\parallel^T(\mathbf{Q}_2^T\mathbf{g}_k) + \frac{1}{2}\mathbf{z}_\parallel^T(\delta_k\mathbf{I} - \widehat{\mathbf{\Lambda}}_k)^{-1}\mathbf{z}_\parallel \tag{33}$$

and

$$\underset{\|\mathbf{z}_\perp\|_2 \le \Delta_k}{\text{minimize}} \; q_\perp(\mathbf{z}_\perp) = \mathbf{z}_\perp^T(\mathbf{Q}_3^T\mathbf{g}_k) + \frac{1}{2}\delta_k^{-1}\|\mathbf{z}_\perp\|_2^2. \tag{34}$$

The two separated minimization problems can be solved analytically.

5.4 Analytical solution

With $\mu_i = (\delta_k - [\widehat{\mathbf{\Lambda}}_k]_{ii})^{-1}$ and $\mathbf{g}_\parallel \equiv \mathbf{Q}_2^T\mathbf{g}_k$, the minimizer $\mathbf{z}_\parallel^*$ of $q_\parallel(\mathbf{z}_\parallel)$ in (33) is given coordinate-wise by

$$[\mathbf{z}_\parallel^*]_i = \theta_i[\mathbf{g}_\parallel]_i, \qquad \text{where} \;\; \theta_i = \begin{cases} -\dfrac{1}{\mu_i} & \text{if } \left|\frac{1}{\mu_i}[\mathbf{g}_\parallel]_i\right| \le \Delta_k, \\ -\dfrac{\Delta_k}{|[\mathbf{g}_\parallel]_i|} & \text{otherwise.} \end{cases} \tag{35}$$

Similarly, with $\mathbf{g}_\perp \equiv \mathbf{Q}_\perp^T\mathbf{g}_k = \mathbf{Q}_3^T\mathbf{g}_k$, the minimizer $\mathbf{z}_\perp^*$ of $q_\perp(\mathbf{z}_\perp)$ in (34) is

$$\mathbf{z}_\perp^* = \beta\mathbf{g}_\perp, \qquad \text{where} \;\; \beta = \begin{cases} -\delta_k & \text{if } \|\delta_k\mathbf{g}_\perp\|_2 \le \Delta_k, \\ -\dfrac{\Delta_k}{\|\mathbf{g}_\perp\|_2} & \text{otherwise.} \end{cases} \tag{36}$$

The solution $\mathbf{s}_{\text{SC}}^*$ to (8) with the shape-changing norm in (31) is $\mathbf{s}_{\text{SC}}^* = \mathbf{Q}\mathbf{z}^*$, where the components $\mathbf{z}_1^*$, $\mathbf{z}_\parallel^*$, and $\mathbf{z}_\perp^*$ of $\mathbf{z}^*$ are given by (30), (35), and (36), respectively. Next, we demonstrate how $\mathbf{Q}\mathbf{z}^*$ can be computed without explicitly forming the large matrix $\mathbf{Q}$.

5.5 Computing the shape-changing-norm solution $\mathbf{s}_{\text{SC}}^*$

Recall that since $\mathbf{Q}$ is orthogonal, $\mathbf{I} = \mathbf{Q}\mathbf{Q}^T = \mathbf{Q}_1\mathbf{Q}_1^T + \mathbf{Q}_2\mathbf{Q}_2^T + \mathbf{Q}_3\mathbf{Q}_3^T$. Because $\mathbf{Q}_1\mathbf{Q}_1^T = \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A}$, the optimal solution $\mathbf{s}_{\text{SC}}^* = \mathbf{Q}\mathbf{z}^*$ to the

trust-region subproblem using a shape-changing norm is

$$\begin{aligned}
\mathbf{s}_{\mathrm{SC}}^{*} &= [\mathbf{Q}_1\ \mathbf{Q}_2\ \mathbf{Q}_3] \begin{bmatrix} \mathbf{0} \\ \mathbf{z}_{\|}^{*} \\ \mathbf{z}_{\perp}^{*} \end{bmatrix} \\
&= \mathbf{Q}_2 \mathbf{z}_{\|}^{*} + \mathbf{Q}_3 \mathbf{z}_{\perp}^{*} \\
&= \mathbf{Q}_2 \mathbf{z}_{\|}^{*} + \beta \mathbf{Q}_3 \mathbf{Q}_3^T \mathbf{g}_k \\
&= \mathbf{Q}_2 \mathbf{z}_{\|}^{*} + \beta \left(\mathbf{I}_n - \mathbf{A}^T (\mathbf{A}\mathbf{A}^T)^{-1} \mathbf{A} - \mathbf{Q}_2 \mathbf{Q}_2^T \right) \mathbf{g}_k. \\
&= \mathbf{Q}_2 \left(\mathbf{z}_{\|}^{*} - \beta \mathbf{Q}_2^T \mathbf{g}_k \right) + \beta \left(\mathbf{I}_n - \mathbf{A}^T (\mathbf{A}\mathbf{A}^T)^{-1} \mathbf{A} \right) \mathbf{g}_k, \qquad (37)
\end{aligned}$$

with $\mathbf{Q}_2 = \left(\mathbf{I} - \mathbf{A}^T (\mathbf{A}\mathbf{A}^T)^{-1} \mathbf{A} \right) \widehat{\mathbf{\Psi}}_k \widehat{\mathbf{R}}_2^{-1} \widehat{\mathbf{V}}_2$ from Sec. 5.1. Observe that the expression (37) is a direct formula for computing a search direction (using the shape-changing norm), and it only requires the small sub-matrix $\mathbf{Q}_2$ of the large orthogonal matrix $\mathbf{Q}$. The minimization using the shape-changing norm is summarized in Algorithm 2.

5.6 Computational complexity

Like Algorithm 1, Algorithm 2 first computes the equality-constrained minimizer (line 2). The computational complexity of this step is described in Section 3.2. The trust-region step is computed on line 15 and is of the form

$$\mathbf{s}_k = \left(\mathbf{I}_n - \mathbf{A}^T (\mathbf{A}\mathbf{A}^T)^{-1} \mathbf{A} \right) \left(\widehat{\mathbf{\Psi}}_k \widehat{\mathbf{R}}_2^{-1} \widehat{\mathbf{V}}_2 (\mathbf{z}_{\|} - \beta \mathbf{g}_{\|}) + \beta \mathbf{g}_k \right).$$

To obtain an estimate of the computational complexity of $\mathbf{s}_k$, we first focus on

$$\mathbf{g}_{\|} = \mathbf{Q}_2^T \mathbf{g}_k = \widehat{\mathbf{V}}_2^T \widehat{\mathbf{R}}_2^{-T} \left(\widehat{\mathbf{\Psi}}_k^T \mathbf{g}_k - \widehat{\mathbf{\Psi}}_k^T \mathbf{A}^T (\mathbf{A}\mathbf{A}^T)^{-1} \mathbf{A} \mathbf{g}_k \right).$$

In this representation, $\widehat{\mathbf{\Psi}}_k^T \mathbf{g}_k$ and $\mathbf{A}\mathbf{g}_k$ (from $\mathbf{\Psi}_k^T \mathbf{g}_k$) and $\widehat{\mathbf{\Psi}}_k^T \mathbf{A}^T$ (from computing $\mathbf{M}_k$) are available as a result of generating the equality-constrained minimizer first (line 2). Moreover, since $\widehat{\mathbf{V}}_2$ and $\widehat{\mathbf{R}}_2$ are $\mathbb{R}^{2l \times 2l}$ matrices, the most complex calculation of $\mathbf{g}_{\|}$ is of order $\max\left(\mathcal{O}(m^3), \mathcal{O}(4l^2) \right)$. This depends on whether solving the symmetric linear system $\mathbf{A}\mathbf{A}^T$ or the triangular system $\widehat{\mathbf{R}}_2^T$ requires more effort. However, these terms do not depend on the large variable n, and are expected to be inexpensive. Based on this, and because $\mathbf{z}_{\|}$ is computed from $\mathbf{g}_{\|}$ by (35), the dominant number of multiplications in forming the vector

$$\boldsymbol{\xi}_k \equiv \widehat{\mathbf{\Psi}}_k \widehat{\mathbf{R}}_2^{-1} \widehat{\mathbf{V}}_2 (\mathbf{z}_{\|} - \beta \mathbf{g}_{\|}) + \beta \mathbf{g}_k \qquad (38)$$

is $\mathcal{O}(2ln + n)$. Subsequently, $\mathbf{s}_k = \left(\mathbf{I}_n - \mathbf{A}^T (\mathbf{A}\mathbf{A}^T)^{-1} \mathbf{A} \right) \boldsymbol{\xi}_k$ is obtained with approximately $\mathcal{O}(2mn)$ multiplications. The latter estimates were based on the assumption that $\widehat{\mathbf{R}}_2$ and $\widehat{\mathbf{V}}_2$ are available. These matrices are computed

Algorithm 2: LTRSC-LEC (Limited-Memory Trust-Region Shape-Changing Norm for Linear Equality Constraints)

Initialize: $0 \leq c_1$, $0 < c_2, c_3, c_4, c_5, c_6 < 1 < c_7$, $0 < \varepsilon_1$, $3 \leq l \leq 7$, $k = 0$, $\Delta_k = \|\mathbf{x}_k\|_2$, $\mathbf{g}_k = \nabla f(\mathbf{x}_k)$, $\delta_k = 1/\|\mathbf{\Phi}_k(0)\mathbf{g}_k\|_2$, $\gamma_k = \delta_k^{-1}$, $\mathbf{\Psi}_k = \mathbf{A}^T$, $\mathbf{M}_k = \delta_k(\mathbf{A}\mathbf{A}^T)^{-1}$

1 **while** $(\varepsilon_1 \leq \|\mathbf{g}_k - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A}\mathbf{g}_k\|_2 / \max(1, \|\mathbf{x}_k\|_2))$ **do**
2 $\quad \mathbf{s}_k = -\left(\delta_k \mathbf{I} + \mathbf{\Psi}_k \mathbf{M}_k \mathbf{\Psi}_k^T\right)\mathbf{g}_k$; `/* Equality constrained step */`
3 $\quad \rho_k = 0$;
4 $\quad$ **if** $\|\mathbf{s}_k\|_2 \leq \Delta_k$ **then**
5 $\quad\quad \rho_k = (f(\mathbf{x}_k) - f(\mathbf{x}_k + \mathbf{s}_k))/(Q(\mathbf{0}) - Q(\mathbf{s}_k))$;
6 $\quad$ **end**
7 $\quad$ **if** $\rho_k \leq c_1$ **then**
8 $\quad\quad \widehat{\mathbf{R}}_2^T \widehat{\mathbf{R}}_2 = \widehat{\mathbf{\Psi}}_k^T \widehat{\mathbf{\Psi}}_k - \widehat{\mathbf{\Psi}}_k^T \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A}\widehat{\mathbf{\Psi}}_k$; `/* Cholesky factorization */`
9 $\quad\quad \widehat{\mathbf{V}}_2 \widehat{\mathbf{\Lambda}}_k \widehat{\mathbf{V}}_2^T = \widehat{\mathbf{R}}_2 (\mathbf{M}_k)_{22} \widehat{\mathbf{R}}_2^T$; `/* Eigendecomposition */`
10 $\quad\quad \mathbf{g}_{\|} = \widehat{\mathbf{V}}_2^T \widehat{\mathbf{R}}_2^{-T}(\widehat{\mathbf{\Psi}}_k^T \mathbf{g}_k - \widehat{\mathbf{\Psi}}_k^T \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A}\mathbf{g}_k)$;
11 $\quad\quad \|\mathbf{g}_{\perp}\|_2 = (\|\mathbf{g}_k\|_2^2 - \mathbf{g}_k^T \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A}\mathbf{g}_k - \|\mathbf{g}_{\|}\|_2^2)^{\frac{1}{2}}$;
12 $\quad\quad$ **repeat**
13 $\quad\quad\quad$ Set $\mathbf{z}_{\|}$ from (35) ;
14 $\quad\quad\quad$ Set β from (36);
15 $\quad\quad\quad \mathbf{s}_k = (\mathbf{I}_n - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A})(\widehat{\mathbf{\Psi}}_k \widehat{\mathbf{R}}_2^{-1} \widehat{\mathbf{V}}_2(\mathbf{z}_{\|} - \beta \mathbf{g}_{\|}) + \beta \mathbf{g}_k)$;
16 $\quad\quad\quad \rho_k = 0$;
17 $\quad\quad\quad$ **if** $0 < (f(\mathbf{x}_k) - f(\mathbf{x}_k + \mathbf{s}_k))$ **then**
18 $\quad\quad\quad\quad \rho_k = (f(\mathbf{x}_k) - f(\mathbf{x}_k + \mathbf{s}_k))/(Q(\mathbf{0}) - Q(\mathbf{s}_k))$;
19 $\quad\quad\quad$ **end**
20 $\quad\quad\quad$ **if** $\rho_k \leq c_2$ **then**
21 $\quad\quad\quad\quad \Delta_k = \min(c_3 \|\mathbf{s}_k\|_{\mathbf{Q}}, c_4 \Delta_k)$;
22 $\quad\quad\quad$ **end**
23 $\quad\quad$ **until** $c_1 < \rho_k$;
24 $\quad$ **end**
25 $\quad \mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{s}_k$; `/* Accept step */`
26 $\quad$ **if** $c_5 \Delta_k \leq \|\mathbf{s}_k\|_{\mathbf{Q},\infty}$ **and** $c_6 \leq \rho_k$ **then**
27 $\quad\quad \Delta_k = c_7 \Delta_k$;
28 $\quad$ **end**
29 $\quad \mathbf{g}_{k+1} = \nabla f(\mathbf{x}_{k+1})$ and $\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$;
30 $\quad \delta_{k+1} = \mathbf{y}_k^T \mathbf{s}_k / \mathbf{s}_k^T \mathbf{s}_k$ and $\gamma_{k+1} = \delta_{k+1}^{-1}$;
31 $\quad$ Update $\mathbf{\Psi}_k, \mathbf{M}_k$ from (12), $k = k + 1$;
32 **end**

in lines 8 and 9, respectively. The upper triangular $\widehat{\mathbf{R}}_2$ is defined by an implicit QR factorization

$$\mathbf{P}\widehat{\mathbf{\Psi}}_k = \left(\mathbf{I}_n - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A}\right)\widehat{\mathbf{\Psi}}_k = \widehat{\mathbf{Q}}_2 \widehat{\mathbf{R}}_2,$$

and, equivalently, by the explicit relation

$$\widehat{\mathbf{R}}_2^T \widehat{\mathbf{Q}}_2^T \widehat{\mathbf{Q}}_2 \widehat{\mathbf{R}}_2 = \widehat{\mathbf{R}}_2^T \widehat{\mathbf{R}}_2 = \widehat{\mathbf{\Psi}}_k^T \widehat{\mathbf{\Psi}}_k - \widehat{\mathbf{\Psi}}_k^T \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A}\widehat{\mathbf{\Psi}}_k.$$

The last identity is due to the idempotency of orthogonal projection matrices: $\mathbf{P} = \mathbf{P}^2$. Since $\mathbf{A}\widehat{\mathbf{\Psi}}_k$ is available, the dominant cost in computing the small matrix $\widehat{\mathbf{R}}_2$ via a Cholesky factorization comes from updating $\widehat{\mathbf{\Psi}}_k^T \widehat{\mathbf{\Psi}}_k$. This matrix is updated by the vectors $\widehat{\mathbf{\Psi}}_{k-1}^T \mathbf{s}_{k-1}$ and $\widehat{\mathbf{\Psi}}_{k-1}^T \mathbf{y}_{k-1}$ at complexity

$\mathcal{O}(4l \cdot n)$, which is expected to be larger than the approximate $\mathcal{O}(8l^3)$ from a Cholesky factorization. The orthogonal matrix $\widehat{\mathbf{V}}_2$ is computed from the eigendecomposition $\widehat{\mathbf{R}}_2(\widehat{\mathbf{M}}_k - \mathbf{C}_k^T\boldsymbol{\Omega}_k\mathbf{C}_k)\widehat{\mathbf{R}}_2^T = \widehat{\mathbf{V}}_2\widehat{\boldsymbol{\Lambda}}_k\widehat{\mathbf{V}}_2^T \in \mathbb{R}^{2l\times 2l}$. Note that this eigendecomposition does not depend on n. In sum, the dominant number of multiplications for computing $\mathbf{s}_k$ are $\mathcal{O}((6l+2m)n+n)$. However after $\widehat{\mathbf{R}}_2$ has been formed once, the subsequent costs of computing the trust-region step $\mathbf{s}_k$ are $\mathcal{O}(2(l+m)n+n)$. This is advantageous when the trust-region constraint is active and $\mathbf{s}_k$ is recomputed, repeatedly. Moreover, from the analysis of Section 3.2, the computational complexity of the step in line 2 is comparable with the estimate in [10, Section 3.1], and therefore also with Algorithm 1. However, when the trust-region subproblem is required to be recomputed, we expect computational efficiencies from Algorithm 2.

6 Convergence Analysis

This section analyzes the convergence properties of our two proposed methods (Algorithms 1 and 2). The analysis is based on the *sufficient decrease* principle for trust-region methods [14]. This principle requires that a computed minimizer to the trust-region subproblem reduces the quadratic approximation $Q(\mathbf{s})$ by a satisfactory amount. Specifically, in [14, Sec. 12.2] the sufficient decrease condition for the trust-region subproblem (8) is formulated as

$$Q(\mathbf{0}) - Q(\mathbf{s}_k) \geq c\pi_k \min\left(\pi_k/\|\mathbf{B}_k\|_2, \Delta_k\right), \tag{39}$$

where $0 < c < 1$, and $\pi_k = \|(\mathbf{I}_n - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A})\mathbf{g}_k\|_2$. Note that π_k may be equivalently expressed as

$$\pi_k = \underset{\boldsymbol{\rho}\in\mathbb{R}^m}{\text{minimize}}\ \|\mathbf{g}_k - \mathbf{A}^T\boldsymbol{\rho}\|_2.$$

If the steps in a trust-region algorithm satisfy condition (39), then global convergence of the algorithm is deduced, as the method is guaranteed a sufficient improvement at each accepted trial step [14, 28]. In this section, we prove that the equality-constrained solution, $\boldsymbol{s}_e$ from (15), and the solutions of the trust-region subproblems from Algorithms 1 and 2 ((23) and (37), respectively) all satisfy the sufficient decrease condition (39).

6.1 Sufficient decrease with the equality-constrained minimizer $\mathbf{s}_e$

Observe that both proposed algorithms first test the equality-constrained minimizer $\boldsymbol{s}_e = -\mathbf{V}_k\mathbf{g}_k$. If the length of this direction exceeds the trust-region constraint, i.e., $\|\boldsymbol{s}_e\|_2 > \Delta_k$, or it does not satisfactorily reduce the objective function, then the steps computed by the two methods will be different. The following lemma demonstrates that $\boldsymbol{s}_e$ satisfies (39).

Lemma 3 *Provided* $\|\boldsymbol{s}_e\|_2 \leq \Delta_k$*, the equality-constrained minimizer*

$$\boldsymbol{s}_e = -\mathbf{V}_k\mathbf{g}_k = -\mathbf{H}_k\boldsymbol{\Phi}_k(0)\mathbf{g}_k$$

of the trust-region subproblem in (7) *satisfies the sufficient decrease condition* (39)*.*

Proof Recall from Sec. 4 that $\mathbf{H}_k\boldsymbol{\Phi}_k = \boldsymbol{\Phi}_k{}^T\mathbf{H}_k\boldsymbol{\Phi}_k$, where we let $\boldsymbol{\Phi}_k = \boldsymbol{\Phi}_k(0)$. Then substituting the equality-constrained minimizer $\boldsymbol{s}_e$ into the left-hand side of (39), we obtain

$$\begin{aligned} Q(\mathbf{0}) - Q(\boldsymbol{s}_e) &= \mathbf{g}_k^T\mathbf{H}_k\boldsymbol{\Phi}_k\mathbf{g}_k - \tfrac{1}{2}\mathbf{g}_k^T\boldsymbol{\Phi}_k{}^T\mathbf{H}_k\boldsymbol{\Phi}_k\mathbf{g}_k \\ &= \tfrac{1}{2}\mathbf{g}_k^T\boldsymbol{\Phi}_k{}^T\mathbf{H}_k\boldsymbol{\Phi}_k\mathbf{g}_k \\ &\geq \tfrac{1}{2\hat{\lambda}_{\max}}\mathbf{g}_k^T\boldsymbol{\Phi}_k{}^T\boldsymbol{\Phi}_k\boldsymbol{\Phi}_k\mathbf{g}_k \\ &= \tfrac{1}{2}\|\boldsymbol{\Phi}_k\mathbf{g}_k\|_2 \cdot \frac{\|\boldsymbol{\Phi}_k\mathbf{g}_k\|_2}{\|\mathbf{B}_k\|_2} \\ &\geq \tfrac{1}{2}\pi_k \cdot \min(\pi_k/\|\mathbf{B}_k\|_2, \Delta_k), \end{aligned} \tag{40}$$

where $\hat{\lambda}_{\max}$ is the largest eigenvalue of $\mathbf{B}_k$ and where $\|\boldsymbol{\Phi}_k\mathbf{g}_k\|_2 = \|\mathbf{g}_k + \mathbf{A}^T\boldsymbol{\rho}_e\|_2 = \|\mathbf{g}_k - \mathbf{A}^T(-\boldsymbol{\rho}_e)\|_2 \geq \pi_k$ with $\boldsymbol{\rho}_e$ from (11). Comparing the final inequality from (40) with (39), we conclude that the unconstrained minimizer satisfies the sufficient decrease condition. □

6.2 Sufficient decrease with the ℓ_2-norm minimizer $\mathbf{s}_{\ell_2}^*$

When the equality-constrained minimizer $\boldsymbol{s}_e$ is not accepted, Algorithm 1 computes the ℓ_2-norm inequality constraint minimizer $\mathbf{s}_{\ell_2}^*$ in (23). We prove $\mathbf{s}_{\ell_2}^*$ satisfies the sufficient decrease condition in the following lemma.

Lemma 4 *The solution* $\mathbf{s}_{\ell_2}^* = -\mathbf{H}_k(\sigma^*)\boldsymbol{\Phi}_k(\sigma^*)\mathbf{g}_k$ *of the trust-region subproblem in* (7) *defined by the* ℓ_2*-norm, where* $\sigma^* \geq 0$*, satisfies the sufficient decrease condition* (39)*.*

Proof Note that $\mathbf{s}_{\ell_2}^*$ lies on the boundary, i.e., $\|\mathbf{s}_{\ell_2}^*\|_2 = \Delta_k$. Then

$$(\mathbf{s}_{\ell_2}^*)^T(\mathbf{B}_k + \sigma^*\mathbf{I})\mathbf{s}_{\ell_2}^* = (\mathbf{s}_{\ell_2}^*)^T\mathbf{B}_k\mathbf{s}_{\ell_2}^* + \sigma^*\Delta_k^2.$$

In addition, note from (23) that

$$(\mathbf{s}_{\ell_2}^*)^T(\mathbf{B}_k + \sigma^*\mathbf{I})\mathbf{s}_{\ell_2}^* = -(\mathbf{s}_{\ell_2}^*)^T\boldsymbol{\Phi}_k^*\mathbf{g}_k = \mathbf{g}_k^T(\boldsymbol{\Phi}_k^*)^T\mathbf{H}_k^*\boldsymbol{\Phi}_k^*\mathbf{g}_k,$$

where $\boldsymbol{\Phi}_k^* = \boldsymbol{\Phi}_k(\sigma^*)$ and $\mathbf{H}_k^* = \mathbf{H}_k(\sigma^*)$. Moreover, from the definition of $\boldsymbol{\Phi}_k^*$, it holds that $(\boldsymbol{\Phi}_k^*)^T\mathbf{H}_k^*\boldsymbol{\Phi}_k^* = \mathbf{H}_k^*\boldsymbol{\Phi}_k^*$. Therefore

$$\begin{aligned} Q(\mathbf{0}) - Q(\mathbf{s}_{\ell_2}^*) &= -(\mathbf{g}_k^T\mathbf{s}_{\ell_2}^* + \tfrac{1}{2}(\mathbf{s}_{\ell_2}^*)^T\mathbf{B}_k\mathbf{s}_{\ell_2}^*) \\ &= -\left(-\mathbf{g}_k^T\mathbf{H}_k^*\boldsymbol{\Phi}_k^*\mathbf{g}_k + \tfrac{1}{2}(\mathbf{g}_k^T(\boldsymbol{\Phi}_k^*)^T\mathbf{H}_k^*\boldsymbol{\Phi}_k^*\mathbf{g}_k - \sigma^*\Delta_k^2)\right) \\ &= \tfrac{1}{2}\mathbf{g}_k^T(\boldsymbol{\Phi}_k^*)^T\mathbf{H}_k^*\boldsymbol{\Phi}_k^*\mathbf{g}_k + \tfrac{1}{2}\sigma^*\Delta_k^2 \\ &\geq \tfrac{1}{2}\mathbf{g}_k^T(\boldsymbol{\Phi}_k^*)^T\mathbf{H}_k^*\boldsymbol{\Phi}_k^*\mathbf{g}_k. \end{aligned}$$

Because $\mathbf{H}_k^* = (\mathbf{B}_k + \sigma^*\mathbf{I})^{-1}$, thus $\|\mathbf{\Phi}_k^*\mathbf{g}_k\|_2^2/(\hat{\lambda}_{\max} + \sigma^*) \leq \mathbf{g}_k^T(\mathbf{\Phi}_k^*)^T\mathbf{H}_k^*\mathbf{\Phi}_k^*\mathbf{g}_k \leq \|\mathbf{\Phi}_k^*\mathbf{g}_k\|_2^2/(\hat{\lambda}_{\min} + \sigma^*)$, where $\hat{\lambda}_{\max}$ and $\hat{\lambda}_{\min}$ are the largest and smallest eigenvalues of $\mathbf{B}_k$, respectively. Moreover, because

$$\Delta_k^2 = \|\mathbf{s}_{\ell_2}^*\|_2^2 = \mathbf{g}_k^T(\mathbf{\Phi}_k^*)^T(\mathbf{H}_k^*)^2\mathbf{\Phi}_k^*\mathbf{g}_k \leq \|\mathbf{\Phi}_k^*\mathbf{g}_k\|_2^2/(\hat{\lambda}_{\min} + \sigma^*)^2,$$

therefore $\sigma^* \leq \|\mathbf{\Phi}_k^*\mathbf{g}_k\|_2/\Delta_k - \hat{\lambda}_{\min} \leq \|\mathbf{\Phi}_k^*\mathbf{g}_k\|_2/\Delta_k$ because $\mathbf{B}_k$ is an L-BFGS matrix and therefore $\hat{\lambda}_{\min} > 0$. Then the following inequalities hold:

$$\begin{aligned}
Q(\mathbf{0}) - Q(\mathbf{s}_{\ell_2}^*) &\geq \tfrac{1}{2}\|\mathbf{\Phi}_k^*\mathbf{g}_k\|_2^2/(\hat{\lambda}_{\max} + \sigma^*) \\
&\geq \tfrac{1}{2}\left(\frac{\|\mathbf{\Phi}_k^*\mathbf{g}_k\|_2^2}{\|\mathbf{B}_k\|_2 + \|\mathbf{\Phi}_k^*\mathbf{g}_k\|_2/\Delta_k}\right) \\
&\geq \begin{cases} \tfrac{1}{4}\|\mathbf{\Phi}_k^*\mathbf{g}_k\|_2^2/\|\mathbf{B}_k\|_2 & \text{if } \|\mathbf{B}_k\|_2 > \|\mathbf{\Phi}_k^*\mathbf{g}_k\|_2/\Delta_k, \\ \tfrac{1}{4}\|\mathbf{\Phi}_k^*\mathbf{g}_k\|_2\Delta_k & \text{otherwise} \end{cases} \\
&\geq \tfrac{1}{4}\pi_k \cdot \min(\pi_k/\|\mathbf{B}_k\|_2, \Delta_k)
\end{aligned}$$

because $\|\mathbf{\Phi}^*\mathbf{g}_k\|_2 = \|\mathbf{g}_k + \mathbf{A}^T\boldsymbol{\rho}(\sigma^*)\|_2 = \|\mathbf{g}_k - \mathbf{A}^T(-\boldsymbol{\rho}(\sigma^*))\|_2 \geq \pi_k$, where $\boldsymbol{\rho}(\sigma^*)$ is specified by system (17). □

6.3 Sufficient decrease with the shape-changing-norm minimizer $\mathbf{s}_{\text{SC}}^*$

Note from Sec. 5.2 that the quadratic objective function has the equivalent representation $Q(\mathbf{s}) = q(\mathbf{z}_\|, \mathbf{z}_\perp)$. Using the closed-form expressions for the solutions $\mathbf{z}_\|^*$ and $\mathbf{z}_\perp^*$ allows us to show that the corresponding solution $\mathbf{s}_{\text{SC}}^*$ satisfies (39).

Lemma 5 *The solution* $\mathbf{s}_{SC}^* = \mathbf{Q}\mathbf{z}^*$ *in* (37) *to the trust-region subproblem in* (8) *with shape-changing norm* (31) *satisfies the sufficient decrease condition* (39).

Proof The expressions for $\mathbf{z}_\|^*$ in (35) and $\mathbf{z}_\perp^*$ in (36) give

$$\begin{aligned}
Q(\mathbf{0}) - Q(\mathbf{s}_{\text{SC}}^*) &= -q(\mathbf{z}_\|^*, \mathbf{z}_\perp^*) \\
&= -\Big[(\mathbf{z}_\|^*)^T\mathbf{Q}_2^T\mathbf{g}_k + \tfrac{1}{2}(\mathbf{z}_\|^*)^T(\delta_k\mathbf{I} - \widehat{\mathbf{\Lambda}}_k)^{-1}\mathbf{z}_\|^* + \\
&\qquad (\mathbf{z}_\perp^*)^T\mathbf{Q}_\perp^T\mathbf{g}_k + \tfrac{1}{2}\delta_k^{-1}(\mathbf{z}_\perp^*)^T\mathbf{z}_\perp^*\Big] \\
&= -\left[\sum_{i=1}^{2l}\left\{\theta_i[\mathbf{g}_\|]_i^2 + \tfrac{1}{2}\theta_i^2\mu_i[\mathbf{g}_\|]_i^2\right\} + \beta\|\mathbf{g}_\perp\|_2^2 + \tfrac{1}{2}\frac{\beta^2}{\delta_k}\|\mathbf{g}_\perp\|_2^2\right] \\
&= \sum_{i=1}^{2l}\left\{\left(-\theta_i - \tfrac{1}{2}\theta_i^2\mu_i\right)[\mathbf{g}_\|]_i^2\right\} + \left(-\beta - \tfrac{1}{2}\frac{\beta^2}{\delta_k}\right)\|\mathbf{g}_\perp\|_2^2.
\end{aligned}$$

Now from (35), if $|[\mathbf{g}_\parallel]_i/\mu_i| \leq \Delta_k$, then $\theta_i = -1/\mu_i$ and

$$\left(-\theta_i - \tfrac{1}{2}\theta_i^2\mu_i\right) = \frac{1}{\mu_i} - \tfrac{1}{2}\left(\frac{1}{\mu_i}\right)^2\mu_i = \frac{1}{2\mu_i} = -\tfrac{1}{2}\theta_i.$$

Otherwise, $|[\mathbf{g}_\parallel]_i/\mu_i| > \Delta_k$ and therefore $\theta_i = -\Delta_k/|[\mathbf{g}_\parallel]_i|$, which both imply $1 > \Delta_k|\mu_i|/|[\mathbf{g}_\parallel]_i| = |\theta_i\mu_i|$. Thus

$$\left(-\theta_i - \tfrac{1}{2}\theta_i^2\mu_i\right) = \frac{\Delta_k}{|[\mathbf{g}_\parallel]_i|}\left(1 + \tfrac{1}{2}\theta_i\mu_i\right) > \tfrac{1}{2}\frac{\Delta_k}{|[\mathbf{g}_\parallel]_i|} = -\tfrac{1}{2}\theta_i.$$

Consequently,

$$\begin{aligned}
\sum_{i=1}^{2l}\left\{\left(-\theta_i - \tfrac{1}{2}\theta_i^2\mu_i\right)[\mathbf{g}_\parallel]_i^2\right\} &\geq \tfrac{1}{2}\sum_{i=1}^{2l}\left\{\left(-\theta_i\right)[\mathbf{g}_\parallel]_i^2\right\} \\
&\geq \tfrac{1}{2}\min_{1\leq i\leq 2l}\{-\theta_i\}\sum_{i=1}^{2l}\left\{[\mathbf{g}_\parallel]_i^2\right\} \\
&\geq \tfrac{1}{2}\min_{1\leq i\leq 2l}\left\{\frac{1}{\mu_i}, \frac{\Delta_k}{|[\mathbf{g}_\parallel]_i|}\right\}\|\mathbf{g}_\parallel\|_2^2 \\
&\geq \tfrac{1}{2}\min\left\{\frac{1}{\|(\delta_k\mathbf{I} - \widehat{\mathbf{\Lambda}}_k)^{-1}\|_2}, \frac{\Delta_k}{\|\mathbf{g}_\parallel\|_2}\right\}\|\mathbf{g}_\parallel\|_2^2.
\end{aligned}$$

Since ${\mathbf{Q}_2}^T\mathbf{B}_k\mathbf{Q}_2 = (\delta_k\mathbf{I} - \widehat{\mathbf{\Lambda}}_k)^{-1}$, $\|\mathbf{B}_k\|_2 \geq \|(\delta_k\mathbf{I} - \widehat{\mathbf{\Lambda}}_k)^{-1}\|_2$ and therefore

$$\sum_{i=1}^{2l}\left\{\left(-\theta_i - \tfrac{1}{2}\theta_i^2\mu_i\right)[\mathbf{g}_\parallel]_i^2\right\} \geq \tfrac{1}{2}\min\left\{\frac{1}{\|\mathbf{B}_k\|_2}, \frac{\Delta_k}{\|\mathbf{g}_\parallel\|_2}\right\}\|\mathbf{g}_\parallel\|_2^2.$$

Similarly, if $\|\delta_k\mathbf{g}_\perp\|_2 \leq \Delta_k$, then $\beta = -\delta_k$ and

$$\left(-\beta - \tfrac{1}{2}\frac{\beta^2}{\delta_k}\right) = \delta_k - \tfrac{1}{2}\frac{\delta_k^2}{\delta_k} = \tfrac{1}{2}\delta_k = -\tfrac{1}{2}\beta.$$

Otherwise, $\|\delta_k\mathbf{g}_\perp\|_2 > \Delta_k$ and therefore $\beta = -\Delta_k/\|\mathbf{g}_\perp\|_2$, which both imply $1 > \Delta_k/\|\delta_k\mathbf{g}_\perp\|_2 = |\beta/\delta_k|$. Thus

$$\left(-\beta - \tfrac{1}{2}\frac{\beta^2}{\delta_k}\right) = \frac{\Delta_k}{\|\mathbf{g}_\perp\|_2}\left(1 - \tfrac{1}{2}\frac{|\beta|}{|\delta_k|}\right) > \tfrac{1}{2}\frac{\Delta_k}{\|\mathbf{g}_\perp\|_2} = -\tfrac{1}{2}\beta.$$

Consequently,

$$\left(-\beta - \tfrac{1}{2}\frac{\beta^2}{\delta_k}\right)\|\mathbf{g}_\perp\|_2^2 \;\geq\; -\tfrac{1}{2}\beta\|\mathbf{g}_\perp\|_2^2 \;\geq\; \tfrac{1}{2}\min\left\{\delta_k, \frac{\Delta_k}{\|\mathbf{g}_\perp\|_2}\right\}\|\mathbf{g}_\perp\|_2^2.$$

Because $\delta_k^{-1} = \gamma_k$ is an eigenvalue of $\mathbf{B}_k$, $\delta_k \geq 1/\|\mathbf{B}_k\|_2$ and therefore

$$\left(-\beta - \tfrac{1}{2}\frac{\beta^2}{\delta_k}\right)\|\mathbf{g}_\perp\|_2^2 \geq \tfrac{1}{2}\min\left\{\frac{1}{\|\mathbf{B}_k\|_2}, \frac{\Delta_k}{\|\mathbf{g}_\perp\|_2}\right\}\|\mathbf{g}_\perp\|_2^2.$$

Combining these results, we obtain

$$Q(\mathbf{0}) - Q(\mathbf{s}^*_{\mathrm{SC}}) \geq \tfrac{1}{2}\min\left\{\frac{1}{\|\mathbf{B}_k\|_2}, \frac{\Delta_k}{\|\mathbf{g}_\parallel\|_2}\right\}\|\mathbf{g}_\parallel\|_2^2 \; + \\ \tfrac{1}{2}\min\left\{\frac{1}{\|\mathbf{B}_k\|_2}, \frac{\Delta_k}{\|\mathbf{g}_\perp\|_2}\right\}\|\mathbf{g}_\perp\|_2^2.$$

Finally, note that

$$\begin{aligned}\|\mathbf{g}_\parallel\|_2^2 + \|\mathbf{g}_\perp\|_2^2 &= \mathbf{g}_k^T\mathbf{Q}_2\mathbf{Q}_2^T\mathbf{g}_k + \mathbf{g}_k^T\mathbf{Q}_\perp\mathbf{Q}_\perp^T\mathbf{g}_k \\ &= \mathbf{g}_k^T(\mathbf{I} - \mathbf{Q}_1\mathbf{Q}_1^T)\mathbf{g}_k \\ &= \mathbf{g}_k^T(\mathbf{I} - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A})\mathbf{g}_k \\ &= \|(\mathbf{I} - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A})\mathbf{g}_k\|_2^2 \\ &= \pi_k^2.\end{aligned}$$

Since $\|\mathbf{g}_\parallel\|_2 \leq \pi_k$ and $\|\mathbf{g}_\perp\|_2 \leq \pi_k$,

$$\begin{aligned}Q(\mathbf{0}) - Q(\mathbf{s}^*_{\mathrm{SC}}) &\geq \tfrac{1}{2}\min\left\{\frac{1}{\|\mathbf{B}_k\|_2}, \frac{\Delta_k}{\pi_k}\right\}\|\mathbf{g}_\parallel\|_2^2 \; + \tfrac{1}{2}\min\left\{\frac{1}{\|\mathbf{B}_k\|_2}, \frac{\Delta_k}{\pi_k}\right\}\|\mathbf{g}_\perp\|_2^2 \\ &= \tfrac{1}{2}\min\left\{\frac{1}{\|\mathbf{B}_k\|_2}, \frac{\Delta_k}{\pi_k}\right\}\pi_k^2 \\ &= \tfrac{1}{2}\pi_k\min\left\{\frac{\pi_k}{\|\mathbf{B}_k\|_2}, \Delta_k\right\}.\end{aligned}$$

□

6.4 Convergence

We establish the convergence of Algorithms 1 and 2 in a theorem that invokes the theory developed by Conn et al. [14]. To be consistent with [14], our result is based on the following assumptions:

A. The objective function $f(\mathbf{x})$ is twice continuously differentiable and bounded below ($f(\mathbf{x}) \geq k_-$), and the Hessian is bounded above ($\nabla^2 f(\mathbf{x}) \leq k_+$), for two constants k_-, k_+.
B. The constraints are twice continuously differentiable and consistent.
C. A first-order constraint qualification holds at a stationary point $\mathbf{x}^*$.
D. The quadratic approximation $Q(\mathbf{s})$ is twice continuously differentiable
E. The quasi-Newton matrix $\mathbf{B}_k$ is nonsingular for all k, i.e., the lowest eigenvalue $\hat{\lambda}_{\min}$ is bounded from 0, and the largest eigenvalue $\hat{\lambda}_{\max}$ is bounded from infinity.

These properties are shown for the L-BFGS matrix in [5]. Finally we note that

$$\frac{1}{\sqrt{2l+m}}\|\mathbf{s}\|_2 \leq \|\mathbf{s}\|_{\mathbf{Q}} \leq \sqrt{2l+m}\|\mathbf{s}\|_2,$$

which relates the shape-changing norm to the ℓ_2-norm, and ensures a measure of 'closeness' to the ℓ_2-norm . We thus propose the following theorem to show the convergence of our two proposed algorithms.

Theorem 2 *Suppose that the eigenvalues of* $\mathbf{B}_k$ *are bounded, i.e.,* $0 < c_l \leq \hat{\lambda}_{\min} \leq \hat{\lambda}_{\max} < c_u$ *for some constants* c_l *and* c_u*. Then every limit point of the sequence of iterates* $\{\mathbf{x}_k\}$ *generated by Algorithm 1 and by Algorithm 2 is first-order critical.*

Proof Algorithms 1 and 2 have the same form as Algorithm 12.2.1 in [14], which is included here as Algorithm 3 for completeness. We make slight adaptations to be consistent with the problem formulation in this article. Algorithm 3 converges to a first-order critical point, as long as the steps $\mathbf{s}_k$ satisfy the sufficient decrease condition

$$Q(\mathbf{0}) - Q(\mathbf{s}_k) \geq c\pi_k \min\left(\pi_k / \|\mathbf{B}_k\|_2, \Delta_k\right).$$

From Lemmas 3–5, all steps used in Algorithms 1 and 2 satisfy the sufficient decrease condition. Therefore we conclude that both algorithms generate iterates that converge to critical points. □

Algorithm 3 (Algorithm 12.2.1 in [14])

Step 0: Initialization. An initial feasible point $\mathbf{x}_0$ and an initial trust-region radius Δ_0 are given. The constants $0 < \varepsilon_1 \leq \varepsilon_2 < 1$ and $0 < \gamma_1 \leq \gamma_2 < 1$ are also given. Compute $f(\mathbf{x}_0)$ and set $k = 0$.

Step 1: Model definition. Define a model $Q(\mathbf{s})$ subject to $\mathbf{A}\mathbf{s} = \mathbf{0}, \|\mathbf{s}\| \leq \Delta_k$.

Step 2: Step calculation. Compute a step $\mathbf{s}_k$ that sufficiently reduces the model $Q(\mathbf{s})$ in the sense of (39) while satisfying the constraints from Step 1;

Step 3: Acceptance of the trial point. Compute $f(\mathbf{x}_k + \mathbf{s}_k)$ and define the ratio

$$\rho_k = \frac{f(\mathbf{x}_k) - f(\mathbf{x}_k + \mathbf{s}_k)}{Q(\mathbf{0}) - Q(\mathbf{s}_k)}.$$

If $\rho_k \geq \varepsilon_1$, then define $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{s}_k$; otherwise define $\mathbf{x}_{k+1} = \mathbf{x}_k$.

Step 4: Trust-region radius update. Set

$$\Delta_{k+1} \in \begin{cases} [\Delta_k, \infty) & \text{if } \rho_k \geq \varepsilon_2, \\ [\gamma_2 \Delta_k, \Delta_k] & \text{if } \rho_k \in [\varepsilon_1, \varepsilon_2), \\ [\gamma_1 \Delta_k, \gamma_2 \Delta_k] & \text{if } \rho_k < \varepsilon_1. \end{cases}$$

Increment k by 1 and go to Step 1.

7 Numerical Experiments

This section describes four types of numerical experiments that benchmark implementations of our proposed algorithms. The codes are developed using MATLAB R2014a (64 bit) and are available at

`https://github.com/johannesbrust/LTR_LEC`

All numerical experiments are carried out on a Dell Precision T1700 desktop computer with Intel i5-4590 CPU @ 3.30GHz x 4 processors, 8 GB RAM, and Linux Ubuntu 14.04, 64-bit. The goal of this section is to demonstrate the effectiveness of the proposed methods for problems of the form (1). Experiment I compares our implementations of Algorithm 1 (labeled TR1) and Algorithm 2 (labeled TR2) with three alternative solvers. In Experiment II, the proposed methods are applied to large-scale quadratic problems, where the number of variables may be as large as 10^7. Experiment III compares the proposed methods on problems where the linear constraints may be degenerate and $\mathbf{A}$ may be large and sparse. In Experiment IV, we compare the methods on a collection of large-scale objective functions from the standard CUTEst library. Performance profiles [16] are provided. We compare the number of iterations (`iter`) and the time (`time`) for each solver on the test set of problems. The performance metric $\rho_s(\tau)$ with a given number of test problems n_p is

$$\rho_s(\tau) = \frac{\text{card}\,\{p : \pi_{p,s} \leq \tau\}}{n_p} \quad \text{and} \quad \pi_{p,s} = \frac{t_{p,s}}{\min\limits_{1\leq i\leq S} t_{p,i}},$$

where $t_{p,s}$ is the "output" (i.e., `time` or `iter`) of "solver" s on problem p. Here S denotes the total number of solvers for a given comparison. This metric measures the proportion of how close a given solver is to the best result. The parameters in Algorithms 1 and 2 are set to $c_1 = 0$, $c_2 = 0.75$, $c_3 = 0.5$, $c_4 = 0.25$, $c_5 = 0.8$, $c_6 = 0.25$, $c_7 = 2$, and $l = 5$, $i_{\max} = 10$.

7.1 Experiment I

This experiment compares TR1 and TR2 to three alternative solvers: FMINCON–LDL [12], FMINCON–CG [12] and IPOPT [26]. FMINCON represents an interior-point solver that implements a L-BFGS quasi-Newton approximation of the Hessian of the Lagrangian, and uses one of two approaches to solve a sequence of linear-equality-constrained subproblems. FMINCON–LDL solves the subproblems by a direct LDL^{T} factorization of the corresponding KKT matrices, while FMINCON-CG uses the projected conjugate gradient method [13] to solve equality-constrained trust-region subproblems. The strategy of FMINCON is based on [8,9,27]. IPOPT [26] implements an interior-point algorithm for large-scale optimization with the option of L-BFGS Hessians. IPOPT and FMINCON handle general constraints in addition to linear equality constraints.

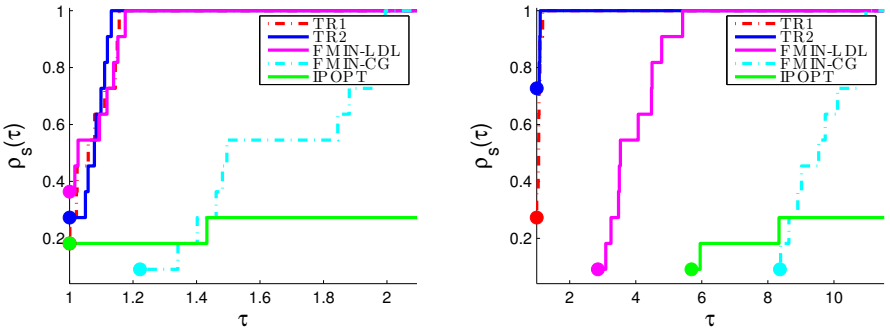

Fig. 1 Performance profiles comparing `iter` (left) and `time` (right) of various solvers on convex quadratic problems with dimensions $n_i = 1000 + 100i, 0 \leq i \leq 10$.

In this experiment we define the objective function as a convex quadratic function

$$f(\mathbf{x}) = \mathbf{c}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{G}\mathbf{x},$$

where $\mathbf{c} \in \mathbb{R}^n$ and $\mathbf{G} \in \mathbb{R}^{n\times n}$ is dense and positive semi-definite. With this definition the solution to problem (1) can be analytically computed. To compare the computed solutions of the different solvers, a solver is determined to have found a solution if the computed vector $\widehat{\mathbf{x}}$ satisfies

$$|f(\widehat{\mathbf{x}}) - f(\mathbf{x}^*)|/|f(\mathbf{x}^*)| \leq 10^{-6} \quad \text{and} \quad \|\mathbf{A}\widehat{\mathbf{x}} - \mathbf{b}\|_2 \leq 10^{-9}, \tag{41}$$

where $\mathbf{x}^*$ is the analytic solution. The problem dimensions are moderately sized with $n_i = 1000 + 100i, 0 \leq i \leq 10$. We expect all solvers to converge for at least some of the problems. In order to set-up the methods, we use the default options of all solvers, with the following exceptions. FMINCON–LDL and FMINCON–CG: `MaxIter = 1e6`, `MaxFunEvals = 1e6`, `TolX = 1e-10`, `Hessian = {'lbfgs',5}`, Ipopt: `jac_c_constant = 'yes'`, `hessian_approximation = 'limited-memory'`, `mu_strategy = 'adaptive'`, `tol = 1.e-7`. A matrix $\tilde{\mathbf{G}} \in \mathbb{R}^{n\times n}$ is used to define $\mathbf{G} = \tilde{\mathbf{G}}^T\tilde{\mathbf{G}}$, and the problem data $\mathbf{c}, \mathbf{b}, \tilde{\mathbf{G}}$, and $\mathbf{A}$ are generated as samples from a standard normal distribution, with $m = 10$. The initial point is $\mathbf{x}_0 = \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A}\mathbf{b}$. Observe in Figure 1 that TR1, TR2 and FMINCON–LDL do best in terms of iterations (left plot) overall. FMINCON–LDL used the fewest iterations for the largest fraction of problems, as indicated by the highest circle on the y-axis. However, TR1 and TR2 tend to have consistent low iteration numbers across all problems, as indicated by the crossings of the dashed-red and solid-blue lines with the magenta line. The right plot in Figure 1 displays that the computational times of TR1 and TR2 were significantly lower than of any other solver. Note that the low profiles of IPOPT do not mean that it did not converge to its tolerances. Rather, they indicate that the computed solutions did not fulfill the combined criteria in (41).

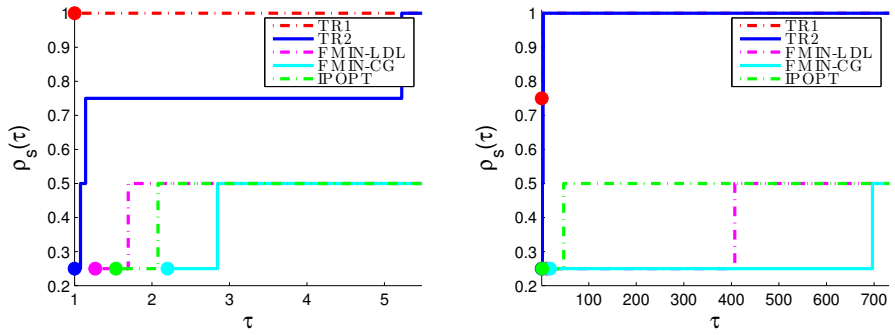

Fig. 2 Performance profiles comparing `iter` (left) and `time` (right) of various solvers on convex quadratic problems with large-scale dimensions $n = 10^4, 10^5, 10^6, 10^7$. The differences in computational times are large, as can be deduced from the relatively large values of τ on the x-axis in the right plot.

7.2 Experiment II

In this experiment the quadratic minimization problems are large-scale, with $10^4 \leq n \leq 10^7$. For large values of n, dense Hessians $\mathbf{G} \in \mathbb{R}^{n \times n}$ cannot be stored. Therefore for this experiment we define the convex quadratic objective function as

$$f(\mathbf{x}) = \mathbf{c}^T\mathbf{x} + \frac{1}{2}(\phi\mathbf{x}^T\mathbf{x} + \mathbf{x}^T\mathbf{G}\mathbf{G}^T\mathbf{x}),$$

where $\mathbf{G} \in \mathbb{R}^{n \times r}$ and $\phi > 0$. The problem data $\phi, \mathbf{c}, \mathbf{b}, \mathbf{G}$, and $\mathbf{A}$ are randomly generated from a standard normal distribution) with $m = 10$ and $r = 5$. Numerical experiments with TR1, TR2, FMINCON–LDL, FMINCON–CG and IPOPT are reported. Beyond $10^5 \leq n$ we observe that FMINCON–LDL, FMINCON–CG and IPOPT take excessively long times, and are therefore not applied to the largest problems. Observe in Figure 2 that TR1 and TR2 obtain the best results in terms of the number of iterations and computational time. We note that both algorithms computed solutions within a few seconds on all problem sizes.

7.3 Experiment III

This experiment uses problems with sparse and possibly low-rank $\mathbf{A} \in \mathbb{R}^{m \times n}$. The objective function is the *Rosenbrock* function

$$f(\mathbf{x}) = \sum_{i=1}^{n/2} (\mathbf{x}_{2i} - \mathbf{x}_{2i-1})^2 + (1 - \mathbf{x}_{2i-1})^2,$$

where n is an even integer. The matrices $\mathbf{A} \in \mathbb{R}^{m \times n}$ are obtained from netlib (see `http://www.netlib.org/lp/data/`). Specific properties of the matrices from this experiment are listed in Table 1.

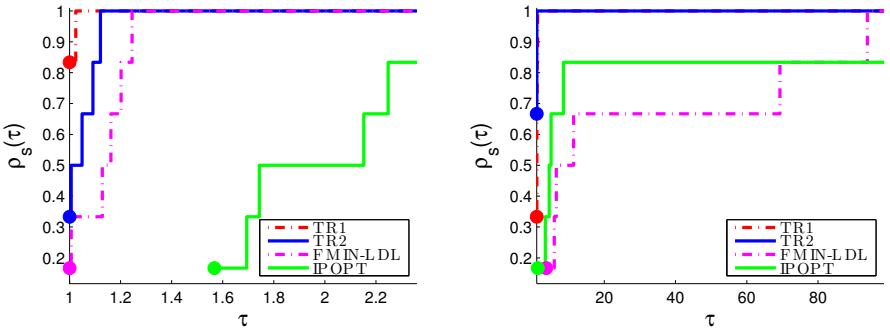

Fig. 3 Performance profiles comparing `iter` (left) and `time` (right) of solvers applied to the Rosenbrock function for various matrices $\mathbf{A}$ from netlib.

Table 1 Properties of the constraint matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ for each problem. We report the number of rows (m), the number of columns (n), the number of nonzeros (nnz) and the rank.

Problem	m	n	nnz	rank
FIT1D	25	1026	14430	25
FIT2D	26	10500	138018	26
D6CUBE	416	6184	43888	405
SCSD1	78	760	3148	78
SCSD6	148	1350	5666	148
SCSD8	398	2750	11334	398

For the Rosenbrock function and the various matrices $\mathbf{A}$, there is in general no analytic formula of the solution. Therefore we compare the computed solutions from the different solvers by checking if the following conditions are satisfied:

$$\|\widehat{\mathbf{g}}_k - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^\dagger \mathbf{A}\widehat{\mathbf{g}}_k\|_2 / \max(1, \|\widehat{\mathbf{x}}_k\|_2) \leq 10^{-5} \quad \text{and} \quad \|\mathbf{A}\widehat{\mathbf{x}}_k - \mathbf{b}\|_2 \leq \epsilon_0,$$

where $\epsilon_0 \leq 10^{-9}$, and $\widehat{\mathbf{x}}_k, \widehat{\mathbf{g}}_k$ are computed solution and gradient, with $(.)^\dagger$ representing the pseudoinverse of a matrix. FMINCON–CG required longer than 9 hours to compute solutions, which is why it is omitted in the comparison. Overall, Figure 3 indicates that the proposed methods obtain desirable results when $\mathbf{A}$ is large, sparse and possibly of low-rank.

7.4 Experiment IV

This experiment benchmarks our algorithms on a set of large nonlinear and possibly non-convex objective functions. We use 62 large-scale unconstrained CUTEst problems and add 10 randomly generated linear equality constraints to each. Each minimization problem is defined by the CUTEst objective function with linear constraints, which are generated using the command `A = randn(m,n)/norm(x0);` where `x0` is the initial vector from the CUTEst problem and the random number generator is initialized by `rng(090317)`. Table 2

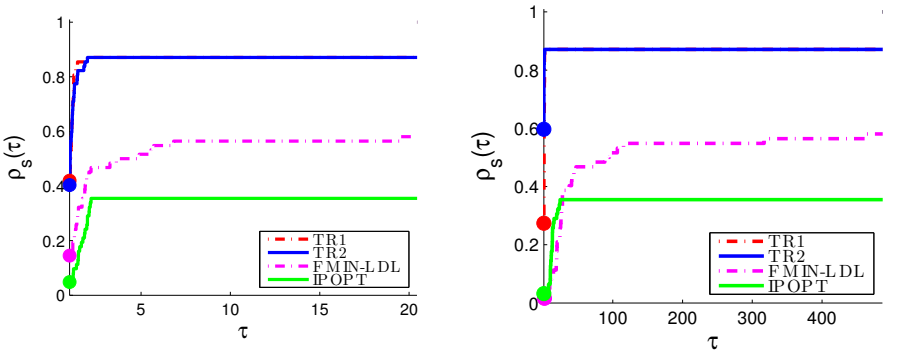

Fig. 4 Performance profiles comparing `iter` (left) and `time` (right) of applying TR1 and TR2 and 2 alternative solvers on large-scale CUTEst problems with added linear equality constraints.

lists the CUTEst objective functions. The results of running four methods are summarized in Figure 4. FMINCON–CG is omitted in the comparisons because it took exceedingly long on some problems. Moreover, we set a time limit of 20 minutes per problem. We observe that TR2 performs better than TR1 in terms of execution time even though the two approaches need similar numbers of iterations. This performance difference is mainly because the trust-region steps of TR2 are computed using an analytical formula that has lower computational cost than the numerical root-finding step of TR1.

Table 2 Unconstrained CUTEst problems used in Experiment IV.

Problem	n				
ARWHEAD	5000	DIXON3DQ	10000	POWER	10000
BDQRTIC	5000	DQDRTIC	5000	QUARTC	5000
BOX	10000	DQRTIC	5000	SCHMVETT	5000
BROYDN7D	5000	EDENSCH	2000	SINQUAD	5000
BRYBND	5000	EG2	1000	SPARSQUR	10000
COSINE	10000	ENGVAL1	5000	SPMSRTLS	4999
CRAGGLVY	5000	EXTROSNB	1000	SROSENBR	5000
CURLY10	10000	FLETCHCR	1000	TOINTGSS	5000
CURLY20	10000	FMINSRF2	5625	TQUARTIC	5000
CURLY30	10000	FREUROTH	5000	TRIDIA	5000
DIXMAANA	3000	GENHUMPS	5000	VAREIGVL	50
DIXMAANB	3000	LIARWHD	5000	WOODS	4000
DIXMAANC	3000	MOREBV	5000	SPARSINE	5000
DIXMAAND	3000	MSQRTALS	1024	TESTQUAD	5000
DIXMAANE	3000	MSQRTBLS	1024	JIMACK	3549
DIXMAANF	3000	NCB20	5010	NCB20B	5000
DIXMAANG	3000	NONCVXU2	5000	EIGENALS	2550
DIXMAANH	3000	NONCVXUN	5000	EIGENBLS	2550
DIXMAANI	3000	NONDIA	5000		
DIXMAANJ	3000	NONDQUAR	5000		
DIXMAANK	3000	PENALTY1	1000		
DIXMAANL	3000	POWELLSG	5000		

Figure 5 compares TR1 and TR2 in more detail.

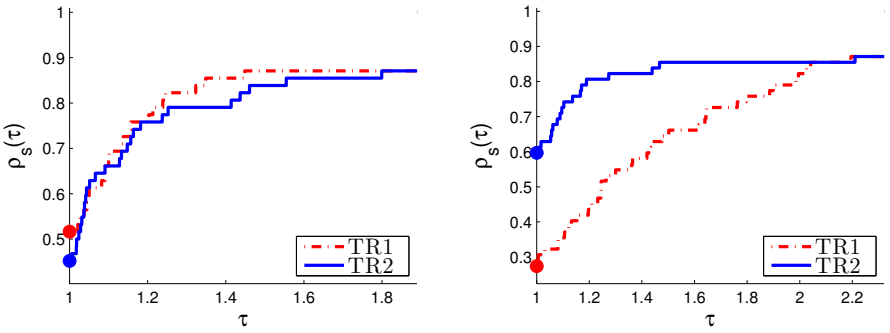

Fig. 5 Performance profiles comparing `iter` (left) and `time` (right) of applying TR1 and TR2 on large-scale CUTEst problems with added linear equality constraints.

8 Conclusion

We developed two limited-memory quasi-Newton trust-region methods for optimization problems with linear constraints $\mathbf{Ax} = \mathbf{b}$. The methods differ in the norm that defines the trust-region subproblem. An advantage of both proposed methods is that they efficiently compute nearly exact trust-region subproblem solutions for large-scale problems. Moreover, one of the proposed methods computes search directions by using an analytic formula. Numerical experiments indicate that the proposed methods perform well on large-scale problems.

Appendix A. Notation

Section 2: Background

$$\begin{aligned} \mathbf{s}_{k-1} &= \mathbf{x}_k - \mathbf{x}_{k-1} & \mathbf{S}_k &= [\mathbf{s}_{k-l} \ \cdots \ \mathbf{s}_{k-1}] \\ \mathbf{y}_{k-1} &= \nabla f(\mathbf{x}_k) - \nabla f(\mathbf{x}_{k-1}) & \mathbf{Y}_k &= [\mathbf{y}_{k-l} \ \cdots \ \mathbf{y}_{k-1}] \\ \mathbf{S}_k^T\mathbf{Y}_k &= \mathbf{L}_k + \mathbf{T}_k & \mathbf{D}_k &= \operatorname{diag}(\mathbf{S}_k^T\mathbf{Y}_k) \end{aligned}$$

$$\begin{aligned} \mathbf{B}_0^{(k)} &= \gamma_k \mathbf{I}_n & \mathbf{H}_k &= \mathbf{B}_k^{-1} \\ \gamma_k &= \mathbf{y}_{k-1}^T\mathbf{y}_{k-1}/\mathbf{y}_{k-1}^T\mathbf{s}_{k-1} & \delta_k &= 1/\gamma_k \end{aligned}$$

$$\begin{aligned} \mathbf{B}_k &= \gamma_k \mathbf{I}_n + \widehat{\mathbf{\Psi}}_k \widehat{\mathbf{\Xi}}_k \widehat{\mathbf{\Psi}}_k^T & \widehat{\mathbf{\Psi}}_k &= [\mathbf{S}_k \ \ \gamma_k \mathbf{Y}_k] \\ \mathbf{H}_k &= \delta_k \mathbf{I}_n + \widehat{\mathbf{\Psi}}_k \widehat{\mathbf{M}}_k \widehat{\mathbf{\Psi}}_k^T \\ \widehat{\mathbf{\Xi}}_k &= \gamma_k \begin{bmatrix} -\mathbf{S}_k^T\mathbf{S}_k & -\mathbf{L}_k \\ -\mathbf{L}_k^T & \gamma_k \mathbf{D}_k \end{bmatrix}^{-1} \\ \widehat{\mathbf{M}}_k &= -(\gamma_k^2 \widehat{\mathbf{\Xi}}_k^{-1} + \gamma_k \widehat{\mathbf{\Psi}}_k^T \widehat{\mathbf{\Psi}}_k)^{-1} \end{aligned}$$

Section 3: Trust-Region Subproblem Solution without an Inequality Constraint

$$\mathbf{K} = \begin{bmatrix} \mathbf{B}_k & \mathbf{A}^T \\ \mathbf{A} & \mathbf{0} \end{bmatrix} \qquad \begin{aligned} \boldsymbol{\Omega}_k &= (\mathbf{A}\mathbf{B}_k^{-1}\mathbf{A}^T)^{-1} \\ \boldsymbol{\Psi}_k &= [\mathbf{A}^T \quad \widehat{\boldsymbol{\Psi}}_k] \end{aligned}$$

$$\mathbf{K}^{-1} = \begin{bmatrix} \mathbf{B}_k^{-1} - \mathbf{B}_k^{-1}\mathbf{A}^T\boldsymbol{\Omega}_k\mathbf{A}\mathbf{B}_k^{-1} & \mathbf{B}_k^{-1}\mathbf{A}^T\boldsymbol{\Omega}_k \\ (\mathbf{B}_k^{-1}\mathbf{A}^T\boldsymbol{\Omega}_k)^T & -\boldsymbol{\Omega}_k \end{bmatrix}$$

$$\begin{aligned} \mathbf{V}_k &= \mathbf{B}_k^{-1} - \mathbf{B}_k^{-1}\mathbf{A}^T\boldsymbol{\Omega}_k\mathbf{A}\mathbf{B}_k^{-1} \\ \mathbf{V}_k &= \delta_k\mathbf{I}_n + \boldsymbol{\Psi}_k\mathbf{M}_k\boldsymbol{\Psi}_k^T \end{aligned}$$

$$\mathbf{W}_k = \mathbf{B}_k^{-1}\mathbf{A}^T\boldsymbol{\Omega}_k$$

$$\mathbf{M}_k = \begin{bmatrix} -\delta_k^2\boldsymbol{\Omega}_k & -\delta_k\boldsymbol{\Omega}_k\mathbf{C}_k \\ -\delta_k\mathbf{C}_k^T\boldsymbol{\Omega}_k & \widehat{\mathbf{M}}_k - \mathbf{C}_k^T\boldsymbol{\Omega}_k\mathbf{C}_k \end{bmatrix}$$

$$\mathbf{C}_k = \mathbf{A}\widehat{\boldsymbol{\Psi}}_k\widehat{\mathbf{M}}_k$$

Section 4: Trust-Region Subproblem Solution with an ℓ_2-Norm Inequality Constraint

$$\begin{aligned} \mathbf{H}_k(\sigma) &= (\mathbf{B}_k + \sigma\mathbf{I})^{-1} & \mathbf{H}_k &= \mathbf{H}_k(0) \\ \boldsymbol{\Phi}_k(\sigma) &= \mathbf{I}_n - \mathbf{A}^T\boldsymbol{\Omega}_k(\sigma)\mathbf{A}\mathbf{H}_k(\sigma) & \boldsymbol{\Phi}_k &= \boldsymbol{\Phi}_k(0) \\ \mathbf{H}_k(\sigma) &= \tfrac{1}{\gamma_k+\sigma}\mathbf{I}_n + \widehat{\boldsymbol{\Psi}}_k\widehat{\mathbf{M}}_k(\sigma)\widehat{\boldsymbol{\Psi}}_k^T \\ \boldsymbol{\Omega}_k(\sigma) &= (\mathbf{A}\mathbf{H}_k(\sigma)\mathbf{A}^T)^{-1} \\ \widehat{\mathbf{M}}_k(\sigma) &= -\big((\gamma_k+\sigma)^2\widehat{\boldsymbol{\Xi}}_k^{-1} + (\gamma_k+\sigma)\widehat{\boldsymbol{\Psi}}_k^T\widehat{\boldsymbol{\Psi}}_k\big)^{-1} \\ \mathbf{V}_k(\sigma) &= \mathbf{H}_k(\sigma) - \mathbf{H}_k(\sigma)\mathbf{A}^T\boldsymbol{\Omega}_k(\sigma)\mathbf{A}\mathbf{H}_k(\sigma) \\ \mathbf{V}_k(\sigma) &= \mathbf{H}_k(\sigma)\boldsymbol{\Phi}_k(\sigma) \end{aligned}$$

$$\begin{aligned} \mathbf{s}(\sigma) &= -\mathbf{H}_k(\sigma)\boldsymbol{\Phi}_k(\sigma)\mathbf{g}_k \\ \mathbf{s}'(\sigma) &= -\mathbf{H}_k(\sigma)\boldsymbol{\Phi}_k(\sigma)\mathbf{s}(\sigma) \end{aligned}$$

Section 5: Trust-Region Subproblem Solution with a Shape-Changing Norm Inequality Constraint

$$\begin{aligned} \mathbf{U}_k &= -\boldsymbol{\Psi}_k\mathbf{M}_k\boldsymbol{\Psi}_k^T \\ \mathbf{A}^T &= \mathbf{Q}_1\mathbf{R}_1 & \mathbf{Q}_1\mathbf{Q}_1^T &= \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A} \\ \mathbf{P} &= \mathbf{I}_n - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A} & \mathbf{P}\widehat{\boldsymbol{\Psi}}_k &= \widehat{\mathbf{Q}}_2\widehat{\mathbf{R}}_2 \\ \widehat{\mathbf{V}}_2\widehat{\boldsymbol{\Lambda}}_k\widehat{\mathbf{V}}_2^T &= \widehat{\mathbf{R}}_2(\widehat{\mathbf{M}}_k - \mathbf{C}_k^T\boldsymbol{\Omega}_k\mathbf{C}_k)\widehat{\mathbf{R}}_2^T \\ \mathbf{Q}_2 &= \widehat{\mathbf{Q}}_2\widehat{\mathbf{V}}_2 \\ \mathbf{Q} &= [\mathbf{Q}_1\ \mathbf{Q}_2\ \mathbf{Q}_3] \\ \mathbf{Q}_\parallel &= [\mathbf{Q}_1\ \mathbf{Q}_2] & \mathbf{Q}_\perp &= \mathbf{Q}_3 \\ \mathbf{z} &= \begin{bmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \\ \mathbf{z}_3 \end{bmatrix} & \mathbf{s} &= \mathbf{Q}\mathbf{z} \\ \mathbf{z}_\parallel &= \mathbf{z}_2 = \mathbf{Q}_2^T\mathbf{s} & \mathbf{z}_\perp &= \mathbf{z}_3 = \mathbf{Q}_3^T\mathbf{s} \\ \mathbf{g}_\parallel &= \mathbf{Q}_2^T\mathbf{g}_k & \mathbf{g}_\perp &= \mathbf{Q}_\perp^T\mathbf{g}_k \end{aligned}$$

$$\mathbf{V}_k = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^T = [\mathbf{Q}_1\ \mathbf{Q}_2\ \mathbf{Q}_3] \begin{bmatrix} \mathbf{0} & & \\ & \delta_k\mathbf{I} - \widehat{\boldsymbol{\Lambda}}_k & \\ & & \delta_k\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{Q}_1^T \\ \mathbf{Q}_2^T \\ \mathbf{Q}_3^T \end{bmatrix}$$

References

1. S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK, 2007.
2. J. J. Brust, O. Burdakov, J. B. Erway, R. F. Marcia, and Y.-X. Yuan. Shape-changing L-SR1 trust-region methods. Technical Report 2016-2, Department of Mathematics, Wake Forest University, 2016.
3. J. J. Brust, O. P. Burdakov, J. B. Erway, and R. F. Marcia. Dense initializations for limited-memory quasi-Newton methods. Technical Report 2017-2, Department of Applied Mathematics, UC Merced, 2017.
4. J. J. Brust, J. B. Erway, and R. F. Marcia. On solving L-SR1 trust-region subproblems. *Computational Optimization and Applications*, 66(2):245–266, 2017.
5. O. Burdakov, L. Gong, Y.-X. Yuan, and S. Zikrin. On efficiently combining limited memory and trust-region techniques. *Mathematical Programming Computation*, 9:101–134, 2016.
6. O. Burdakov, J. Martinez, and E. Pilotta. A limited-memory multipoint symmetric secant method for bound constrained optimization. *Annals of Operations Research*, 117:51–70, 2002.
7. J. V. Burke, A. Wiegmann, and L. Xu. Limited memory BFGS updating in a trust-region framework. Technical Report, University of Washington, 1996.
8. R. H. Byrd, J. C. Gilbert, and J. Nocedal. A trust region method based on interior point techniques for nonlinear programming. *Math. Program., Ser. A*, 89:149–185, 2000.
9. R. H. Byrd, M. Hribar, and J. Nocedal. An interior point algorithm for large-scale nonlinear programming. *SIAM J. Optim.*, 9:877–900, 1999.
10. R. H. Byrd, J. Nocedal, and R. B. Schnabel. Representations of quasi-Newton matrices and their use in limited-memory methods. *Math. Program.*, 63:129–156, 1994.
11. M. Celis, J. Dennis Jr., and R. Tapia. A trust region strategy for equality constrained optimization. Technical Report 84-1, Mathematical Sciences Department, Rice University, 1984.
12. T. Coleman, M. A. Branch, and A. Grace. Optimization toolbox for use with MATLAB. MathWorks: Natick, MA, 1999.
13. T. Coleman and A. Verma. A preconditioned conjugate gradient approach to linear equality constrained minimization. *Computational Optimization and Applications*, 20:61–72, 2001.
14. A. R. Conn, N. I. M. Gould, and P. L. Toint. *Trust-Region Methods*. SIAM, Philadelphia, PA, 2000.
15. O. DeGuchy, J. B. Erway, and R. F. Marcia. Compact representation of the full Broyden class of quasi-Newton updates. *Numerical Linear Algebra with Applications*, 25(5):e2186, 2018.
16. E. Dolan and J. Moré. Benchmarking optimization software with performance profiles. *Math. Program.*, 91:201–213, 2002.
17. J. B. Erway and R. F. Marcia. Algorithm 943: MSS: MATLAB software for L-BFGS trust-region subproblems for large-scale optimization. *ACM Trans. Math. Softw.*, 40(4):28:1–28:12, June 2014.
18. W. W. Hager. Updating the inverse of a matrix. *SIAM Review*, 31(2):221–239, 1989.
19. M. Lalee, J. Nocedal, and T. Plantenga. On the implementation of an algorithm for large-scale equality constrained optimization. *SIAM J. Optim.*, 8(3):682–706, 1998.
20. J. J. Moré and D. C. Sorensen. Computing a trust region step. *SIAM J. Sci. Statist. Comput.*, 4:553–572, 1983.
21. J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer-Verlag, New York, 2 edition, 2006.
22. M. Powell and Y. Yuan. A trust region algorithm for equality constrained optimization. *Math. Program.*, 49:189–211, 1991.
23. M. A. Saunders. PDCO: Primal-dual interior method for convex objectives. http://www.stanford.edu/group/SOL/software/pdco.html, 2002–2015.
24. T. Steihaug. The conjugate gradient method and trust regions in large scale optimization. *SIAM J. Numer. Anal.*, 20:626–637, 1983.

25. A. Vardi. A trust region algorithm for equality constrained minimization: Convergence properties and implementation. *SIAM J. Numer. Anal.*, 22(3), 1985.
26. A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Math. Program.*, 106:25–57, 2006.
27. R. Waltz, J. Morales, J. Nocedal, and D. Orban. An interior algorithm for nonlinear optimization that combines line search and trust region steps. *SIAM. J. Optim.*, 9:877–900, 1999.
28. Y.-X. Yuan. Trust region algorithms for constrained optimization. Technical report, State Key Laboratory of Scientific and Engineering Computing, Beijing.
29. S. Zhijiang. RSQP toolbox for MATLAB. https://www.mathworks.com/matlabcentral/fileexchange/13046-rsqp-toolbox-for-matlab, 2006.