ALGORITHM XXX: SC-SR1: MATLAB SOFTWARE FOR SOLVING SHAPE-CHANGING L-SR1 TRUST-REGION SUBPROBLEMS

JOHANNES BRUST, OLEG BURDAKOV, JENNIFER B. ERWAY, ROUMMEL F. MARCIA, AND YA-XIANG YUAN

ABSTRACT. We present a MATLAB implementation of the shape-changing symmetric rank-one (SC-SR1) method that solves trust-region subproblems when a limited-memory symmetric rank-one (L-SR1) matrix is used in place of the true Hessian matrix. The method takes advantage of two shape-changing norms [4, 3] to decompose the trust-region subproblem into two separate problems. Using one of the proposed shape-changing norms, the resulting subproblems then have closed-form solutions. In the other proposed norm, one of the resulting subproblems has a closed-form solution while the other is easily solvable using techniques that exploit the structure of L-SR1 matrices. Numerical results suggest that the SC-SR1 method is able to solve trust-region subproblems to high accuracy even in the so-called "hard case".

1. INTRODUCTION

At each iteration of a trust-region method for minimizing a general nonconvex function $f(x)$, the so-called *trust-region subproblem* must be solved to obtain a step direction:

$$\underset{\mathbf{p}\in\mathbb{R}^n}{\text{minimize}} \;\; \mathcal{Q}(\mathbf{p}) \triangleq \mathbf{g}^T\mathbf{p} + \frac{1}{2}\mathbf{p}^T\mathbf{B}\mathbf{p} \quad \text{subject to} \quad \|\mathbf{p}\| \leq \delta, \tag{1}$$

where $\mathbf{g} \triangleq \nabla f(\mathbf{x}_k)$, $\mathbf{B}$ is an approximation to $\nabla^2 f(\mathbf{x}_k)$, δ is a positive constant, and $\|\cdot\|$ is a given norm. In this article, we describe a MATLAB implementation for solving the trust-region subproblem (1) when B is a limited-memory symmetric rank-one (L-SR1) matrix approximation of $\nabla^2 f(x_k)$. In large-scale optimization, solving (1) represents the bulk of the computational effort in trust-region methods. The norm used in (1) not only defines the trust region shape but also determines the difficulty of solving each subproblem.

The most widely-used norm chosen to define the trust-region subproblem is the two-norm. One reason for this choice of norm is that the necessary and sufficient conditions for a global solution to the subproblem defined by the two-norm are well-known [12, 20, 24]; many methods exploit these conditions to compute high-accuracy solutions to the trust-region subproblem (see e.g., [8, 9, 10, 15, 2, 20]). The infinity-norm is sometimes used to define the subproblem; however, when $\mathbf{B}$

Date: February 17, 2018.

Key words and phrases. Large-scale unconstrained optimization, nonconvex optimization, trust-region methods, limited-memory quasi-Newton methods, symmetric rank-one update, shape-changing norm.

J. B. Erway is supported in part by National Science Foundation grant CMMI-1334042 and IIS-1741264.

R. F. Marcia is supported in part by National Science Foundation grant CMMI-1333326 and IIS-1741490.

is indefinite, as can be the case when $\mathbf{B}$ is a L-SR1 matrix, the subproblem is NP-hard [21, 27]. For more discussion on norms other than the infinity-norm we refer the reader to [7].

In this article, we consider the trust-region subproblems defined by *shape-changing* norms originally proposed in [4]. Generally speaking, shape-changing norms are norms that depend on $\mathbf{B}$; thus, in the quasi-Newton setting where the quasi-Newton matrix $\mathbf{B}$ is updated each iteration, the shape of the trust region changes each iteration. One of the earliest references to shape-changing norms is found in [14] where a norm is implicitly defined by the product of a permutation matrix and a unit lower triangular matrix that arise from a symmetric indefinite factorization of $\mathbf{B}$. Perhaps the most widely-used shape-changing norm is the so-called "elliptic norm" given by $\|\mathbf{x}\|_A \triangleq \mathbf{x}^\mathbf{T}\mathbf{A}\mathbf{x}$, where $\mathbf{A}$ is a positive-definite matrix (see, e.g., [7]). A well-known use of this norm is found in the Steihaug method [25], and, more generally, truncated preconditioned conjugate-gradients (CG) [7]; these methods reformulate a two-norm trust-region subproblem using an elliptic norm to maintain the property that the iterates from preconditioned CG are increasing in norm. Other examples of shape-changing norms include those defined by vectors in the span of $\mathbf{B}$ (see, e.g., [7]).

The shape-changing norms proposed in [4] have the advantage of breaking the trust-region subproblem into two separate subproblems. Using one of the proposed shape-changing norms, the solution of the subproblem then has a closed-form solution. In the other proposed norm, one of the subproblems has a closed-form solution while the other is easily solvable. The recently-published LMTR algorithm [3] solves trust-region subproblems defined using these shape-changing norms when B in (1) is produced using limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) updates. To our knowledge, there are no other implementations for solving trust-region subproblems defined by these shape-changing norms.

1.1. **Overview of the proposed method.** In this paper, we describe a MATLAB implementation for solving trust-region subproblems defined by the same two shape-changing norms proposed in [4] when L-SR1 approximations to the Hessian are used instead of L-BFGS approximations. The proposed method, called the shape-changing SR1 method (SC-SR1), is based on the LMTR algorithm [3] and the OBS method [2], which solves two-norm trust-region subproblems that are defined using L-SR1 matrices. Unlike LMTR, SC-SR1 must take additional care when B is not positive definite, especially in the so-called "hard case". This work can be viewed as an extension of [3] in the case when L-SR1 matrices are used to define the trust-region subproblem, allowing high-accuracy subproblem solutions to be computed by exploiting the structure of L-SR1 matrices.

This paper is organized as follows: In Section 2, we review L-SR1 matrices, including the compact representation for these matrices and a method to efficiently compute their eigenvalues and a partial eigenbasis. In Section 3, we demonstrate how the shape-changing norms decouple the original trust-region subproblem into two problems and describe the proposed solver for each subproblem. Finally, we show how to construct a global solution to (1) from the solutions of the two decoupled subproblems. Optimality conditions are presented for each of these decoupled subproblems in Section 4. In Section 5, we demonstrate the accuracy of the proposed solver, and concluding remarks can be found in Section 6.

1.2. **Notation.** In this article, the identity matrix of dimension d is denoted by $\mathbf{I}_d = [\mathbf{e}_1 | \cdots | \mathbf{e}_d]$, and depending on the context the subscript d may be suppressed. Finally, we assume that all L-SR1 updates are computed so that the L-SR1 matrix is well defined.

2. L-SR1 matrices

Suppose $f : \mathbb{R}^n \to \mathbb{R}$ is a smooth objective function and $\{x_i\}$, $i = 0, \ldots k$, is a sequence of iterates, then the symmetric rank-one (SR1) matrix is defined using pairs (s_i, y_i) where

$$\mathbf{s}_i \triangleq \mathbf{x}_{i+1} - \mathbf{x}_i \quad \text{and} \quad \mathbf{y}_i \triangleq \nabla f(\mathbf{x}_{i+1}) - \nabla f(\mathbf{x}_i),$$

and ∇f denotes the gradient of f. Specifically, given an initial matrix $\mathbf{B}_0$, $\mathbf{B}_{k+1}$ is defined recursively as

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

provided $(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)^T\mathbf{s}_k \neq 0$. In practice, $\mathbf{B}_0$ is often taken to be a scalar multiple of the identity matrix; for the duration of this article we assume that $\mathbf{B}_0 = \gamma_k\mathbf{I}$, $\gamma_k \in \mathbb{R}$. *Limited-memory* symmetric rank-one matrices (L-SR1) store and make use of only the m most-recently computed pairs $\{(\mathbf{s}_i, \mathbf{y}_i)\}$, where $m \ll n$ (for example, Byrd et al. [5] suggest $m \in [3, 7]$). For simplicity of notation, we assume that the current iteration number k is less than the number of allowed stored limited-memory pairs m.

The SR1 update is a member of the Broyden class of updates (see, e.g., [23]). Unlike widely-used updates such as the Broyden-Fletcher-Goldfarb-Shanno (BFGS) and the Davidon-Fletcher-Powell (DFP) updates, this update can yield indefinite matrices; that is, SR1 matrices can incorporate negative curvature information. In fact, the SR1 update has convergence properties superior to other widely-used positive-definite quasi-Newton matrices such as BFGS; in particular, [6] give conditions under which the SR1 update formula generates a sequence of matrices that converge to the true Hessian. (For more background on the SR1 update formula, see, e.g., [16, 17, 18, 23, 26, 28].)

2.1. **Compact representation.** The compact representation of SR1 matrices can be used to compute the eigenvalues and a partial eigenbasis of these matrices. In this section, we review the compact formulation of SR1 matrices.

To begin, we define the following matrices:

$$\begin{aligned} \mathbf{S}_k &\triangleq [\, \mathbf{s}_0 \;\; \mathbf{s}_1 \;\; \mathbf{s}_2 \;\; \cdots \;\; \mathbf{s}_k \,] \in \mathbb{R}^{n \times (k+1)}, \\ \mathbf{Y}_k &\triangleq [\, \mathbf{y}_0 \;\; \mathbf{y}_1 \;\; \mathbf{y}_2 \;\; \cdots \;\; \mathbf{y}_k \,] \in \mathbb{R}^{n \times (k+1)}. \end{aligned}$$

The matrix $\mathbf{S}_k^T\mathbf{Y}_k \in \mathbb{R}^{(k+1)\times(k+1)}$ can be written as the sum of the following three matrices:

$$\mathbf{S}_k^T\mathbf{Y}_k = \mathbf{L}_k + \mathbf{D}_k + \mathbf{R}_k,$$

where $\mathbf{L}_k$ is strictly lower triangular, $\mathbf{D}_k$ is diagonal, and $\mathbf{R}_k$ is strictly upper triangular. Then, $\mathbf{B}_{k+1}$ can be written as

$$\mathbf{B}_{k+1} = \gamma_k\mathbf{I} + \mathbf{\Psi}_k\mathbf{M}_k\mathbf{\Psi}_k^T, \tag{3}$$

where $\mathbf{\Psi}_k \in \mathbb{R}^{n\times(k+1)}$ and $\mathbf{M}_k \in \mathbb{R}^{(k+1)\times(k+1)}$. In particular, $\mathbf{\Psi}_k$ and $\mathbf{M}_k$ are given by

$$\mathbf{\Psi}_k \ = \ \mathbf{Y}_k - \gamma_k \mathbf{S}_k \quad \text{and} \quad \mathbf{M}_k \ = \ (\mathbf{D}_k + \mathbf{L}_k + \mathbf{L}_k^T - \gamma_k \mathbf{S}_k^T \mathbf{S}_k)^{-1}.$$

The right side of equation (3) is the *compact representation* of $\mathbf{B}_{k+1}$; this representation is due to Byrd et al. [5, Theorem 5.1]. For the duration of this paper, we assume that updates are only accepted when both the next SR1 matrix $\mathbf{B_{k+1}}$ is well-defined and $\mathbf{M}_k$ exists [5, Theorem 5.1]. For notational simplicity, we assume $\mathbf{\Psi}_k$ has full column rank; when $\mathbf{\Psi}_k$ does not have full column rank, we refer to reader to [3] for the modifications needed for computing the eigenvalues reviewed in Section 2.2. Notice that the computation of $\mathbf{M}_k$ is computationally admissible since it is a very small square matrix.

2.2. **Eigenvalues.** In this subsection, we demonstrate how the eigenvalues and a partial eigenbasis can be computed for SR1 matrices. In general, this derivation can be done for any limited-memory quasi-Newton matrix that admits a compact representation; in particular, it can be done for any member of the Broyden convex class [11]. This discussion is based on [3].

Consider the problem of computing the eigenvalues of $\mathbf{B}_{k+1}$, which is assumed to be an L-SR1 matrix, obtained from performing $(k+1)$ rank-one updates to $\mathbf{B}_0 = \gamma \mathbf{I}$. For notational simplicity, we drop subscripts and consider the compact representation of $\mathbf{B}$:

$$\mathbf{B} = \gamma \mathbf{I} + \mathbf{\Psi} \mathbf{M} \mathbf{\Psi}^T, \tag{4}$$

The "thin" QR factorization of $\mathbf{\Psi}$ can be written as $\mathbf{\Psi} = \mathbf{QR}$ where $\mathbf{Q} \in \mathbb{R}^{n\times(k+1)}$ and $\mathbf{R} \in \mathbb{R}^{(k+1)\times(k+1)}$ is invertible because, as it was assumed above, $\mathbf{\Psi}$ has full column rank. Then,

$$\mathbf{B} = \gamma \mathbf{I} + \mathbf{QRMR}^T \mathbf{Q}^T. \tag{5}$$

The matrix $\mathbf{RMR^T} \in \mathbb{R}^{(k+1)\times(k+1)}$ is of a relatively small size, and thus, it is computationally inexpensive to compute its spectral decomposition. We define the spectral decomposition of $\mathbf{RMR}^T$ as $\mathbf{U}\hat{\Lambda}\mathbf{U}^T$, where $\mathbf{U} \in \mathbb{R}^{(k+1)\times(k+1)}$ is an orthogonal matrix whose columns are made up of eigenvectors of $\mathbf{RMR}^T$ and $\hat{\Lambda} = \text{diag}(\hat{\lambda}_1, \ldots, \hat{\lambda}_{k+1})$ is a diagonal matrix whose entries are the associated eigenvalues.

Thus,

$$\mathbf{B} = \gamma \mathbf{I} + \mathbf{QU}\hat{\Lambda}\mathbf{U}^T \mathbf{Q}^T.$$

Since both $\mathbf{Q}$ and $\mathbf{U}$ have orthonormal columns, $\mathbf{P}_\parallel \triangleq \mathbf{QU} \in \mathbb{R}^{n\times(k+1)}$ also has orthonormal columns. Let $P_\perp$ denote the matrix whose columns form an orthonormal basis for $\left(P_\parallel\right)^\perp$. Thus, the spectral decomposition of $\mathbf{B}$ is defined as $\mathbf{B} = \mathbf{P}\Lambda_\gamma \mathbf{P}^T$, where

$$\mathbf{P} \triangleq \begin{bmatrix} \mathbf{P}_\parallel & \mathbf{P}_\perp \end{bmatrix} \quad \text{and} \quad \Lambda_\gamma \triangleq \begin{bmatrix} \Lambda & 0 \\ 0 & \gamma \mathbf{I}_{n-(k+1)} \end{bmatrix}, \tag{6}$$

with $\Lambda_\gamma = \text{diag}(\lambda_1, \ldots, \lambda_n)$ and $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_{k+1}) = \hat{\Lambda} + \gamma \mathbf{I} \in \mathbb{R}^{(k+1)\times(k+1)}$.

We emphasize three important properties of the eigendecomposition. First, all eigenvalues of $\mathbf{B}$ are explicitly obtained and represented by Λ_γ. Second, only the first $(k+1)$ eigenvectors of $\mathbf{B}$ can be explicitly computed, if needed; they are represented by $\mathbf{P}_\parallel$. In particular, since $\mathbf{\Psi} = \mathbf{QR}$, then

$$\mathbf{P}_\parallel = \mathbf{QU} = \mathbf{\Psi}\mathbf{R}^{-1}\mathbf{U}. \tag{7}$$

If $\mathbf{P}_\parallel$ needs to only be available to compute matrix-vector products then one can avoid explicitly forming $\mathbf{P}_\parallel$ by storing $\mathbf{\Psi}$, $\mathbf{R}$, and $\mathbf{U}$. Third, the eigenvalues given by the parameter γ can be interpreted as an estimate of the curvature of f in the space spanned by the columns of $\mathbf{P}_\perp$. While there is no reason to assume the function f has negative curvature throughout the entire subspace $\mathbf{P}_\perp$, in this paper, we consider the case $\gamma \leq 0$ for the sake of completeness.

An alternative approach to computing the eigenvalues of $\mathbf{B}$ is presented in [19]. This method replaces the QR factorization of $\mathbf{\Psi}$ with the SVD and an eigendecomposition of a $(k+1)\times(k+1)$ matrix and $t\times t$ matrix, respectively, where $t \leq (k+1)$. For more details, see [19].

For the duration of this article, we assume the first $(k+1)$ eigenvalues in Λ_γ are ordered in increasing values, i.e., $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_{k+1})$ where $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_{k+1}$ and that r is the multiplicity of λ_1, i.e., $\lambda_1 = \lambda_2 = \cdots = \lambda_r < \lambda_{r+1}$. For details on updating this partial spectral decomposition when a new quasi-Newton pair is computed, see [11].

3. Proposed method

The proposed method is able to solve the L-SR1 trust-region subproblem to high accuracy, even when $\mathbf{B}$ is indefinite. The method makes use of the eigenvalues of $\mathbf{B}$ and the factors of $\mathbf{P}_\parallel$. To describe the method, we first transform the trust-region subproblem (1) so that the quadratic objective function becomes separable. Then, we describe the shape-changing norms proposed in [4, 3] that decouples the separable problem into two minimization problems, one of which has a closed-form solution while the other can be solved very efficiently. Finally, we show how these solutions can be used to construct a solution to the original trust-region subproblem.

3.1. Transforming the Trust-Region Subproblem.

Let $\mathbf{B} = \mathbf{P}\Lambda_\gamma\mathbf{P}^T$ be the eigendecomposition of $\mathbf{B}$ described in Section 2.2. Letting $\mathbf{v} = \mathbf{P}^T\mathbf{p}$ and $\mathbf{g}_\mathbf{P} = \mathbf{P}^T\mathbf{g}$, the objective function $\mathcal{Q}(\mathbf{p})$ in (1) can be written as a function of v:

$$\mathcal{Q}(\mathbf{p}) = \mathbf{g}^T\mathbf{p} + \frac{1}{2}\mathbf{p}^T\mathbf{B}\mathbf{p} = \mathbf{g}_\mathbf{P}^T\mathbf{v} + \frac{1}{2}\mathbf{v}^T\Lambda_\gamma\mathbf{v} \triangleq q(\mathbf{v}).$$

With $\mathbf{P} = \begin{bmatrix}\mathbf{P}_\parallel & \mathbf{P}_\perp\end{bmatrix}$, we partition $\mathbf{v}$ and $\mathbf{g}_\mathbf{P}$ as follows:

$$\mathbf{v} = \mathbf{P}^T\mathbf{p} = \begin{bmatrix}\mathbf{P}_\parallel^T\mathbf{p} \\ \mathbf{P}_\perp^T\mathbf{p}\end{bmatrix} = \begin{bmatrix}\mathbf{v}_\parallel \\ \mathbf{v}_\perp\end{bmatrix} \quad \text{and} \quad \mathbf{g}_\mathbf{P} = \begin{bmatrix}\mathbf{P}_\parallel^T\mathbf{g} \\ \mathbf{P}_\perp^T\mathbf{g}\end{bmatrix} = \begin{bmatrix}\mathbf{g}_\parallel \\ \mathbf{g}_\perp\end{bmatrix},$$

where $\mathbf{v}_\parallel, \mathbf{g}_\parallel \in \mathbb{R}^{(k+1)}$ and $\mathbf{v}_\perp, \mathbf{g}_\perp \in \mathbb{R}^{n-(k+1)}$. Then,

$$\begin{aligned} q(\mathbf{v}) &= \begin{bmatrix}\mathbf{g}_\parallel^T & \mathbf{g}_\perp^T\end{bmatrix}\begin{bmatrix}\mathbf{v}_\parallel \\ \mathbf{v}_\perp\end{bmatrix} + \frac{1}{2}\begin{bmatrix}\mathbf{v}_\parallel^T & \mathbf{v}_\perp^T\end{bmatrix}\begin{bmatrix}\Lambda & \\ & \gamma\mathbf{I}_{n-(k+1)}\end{bmatrix}\begin{bmatrix}\mathbf{v}_\parallel \\ \mathbf{v}_\perp\end{bmatrix} \\ &= \mathbf{g}_\parallel^T\mathbf{v}_\parallel + \mathbf{g}_\perp^T\mathbf{v}_\perp + \frac{1}{2}\left(\mathbf{v}_\parallel^T\Lambda\mathbf{v}_\parallel + \gamma\|\mathbf{v}_\perp\|^2\right) \\ &= q_\parallel\left(\mathbf{v}_\parallel\right) + q_\perp\left(\mathbf{v}_\perp\right), \end{aligned} \tag{8}$$

where

$$q_\parallel\left(\mathbf{v}_\parallel\right) \triangleq \mathbf{g}_\parallel^T\mathbf{v}_\parallel + \frac{1}{2}\mathbf{v}_\parallel^T\Lambda\mathbf{v}_\parallel \quad \text{and} \quad q_\perp\left(\mathbf{v}_\perp\right) \triangleq \mathbf{g}_\perp^T\mathbf{v}_\perp + \frac{\gamma}{2}\|\mathbf{v}_\perp\|^2.$$

Thus, the trust-region subproblem (1) can be expressed as

$$\underset{\|\mathbf{P}\mathbf{v}\|\leq\delta}{\text{minimize}} \; q(\mathbf{v}) = q_\parallel\left(\mathbf{v}_\parallel\right) + q_\perp\left(\mathbf{v}_\perp\right). \tag{9}$$

Note that the function $q(\mathbf{v})$ is now separable in $\mathbf{v}_\parallel$ and $\mathbf{v}_\perp$. To completely decouple (9) into two minimization problems, we use a shape-changing norm so that the norm constraint $\|\mathbf{Pv}\| \leq \delta$ decouples into separate constraints, one involving $\mathbf{v}_\parallel$ and the other involving $\mathbf{v}_\perp$.

3.2. **Shape-Changing Norms.** Consider the following shape-changing norms proposed in [4, 3]:

$$\|\mathbf{p}\|_{\mathbf{P},2} \triangleq \max\left(\|\mathbf{P}_\parallel^T\mathbf{p}\|_2, \|\mathbf{P}_\perp^T\mathbf{p}\|_2\right) = \max\left(\|\mathbf{v}_\parallel\|_2, \|\mathbf{v}_\perp\|_2\right), \tag{10}$$

$$\|\mathbf{p}\|_{\mathbf{P},\infty} \triangleq \max\left(\|\mathbf{P}_\parallel^T\mathbf{p}\|_\infty, \|\mathbf{P}_\perp^T\mathbf{p}\|_2\right) = \max\left(\|\mathbf{v}_\parallel\|_\infty, \|\mathbf{v}_\perp\|_2\right). \tag{11}$$

We refer to them as the $(\mathbf{P},2)$ and the $(\mathbf{P},\infty)$ norms, respectively. Since $\mathbf{p} = \mathbf{Pv}$, the trust-region constraint in (9) can be expressed in these norms as

$$\begin{aligned} \|\mathbf{Pv}\|_{\mathbf{P},2} \leq \delta \quad &\text{if and only if} \quad \|\mathbf{v}_\parallel\|_2 \leq \delta \text{ and } \|\mathbf{v}_\perp\|_2 \leq \delta, \\ \|\mathbf{Pv}\|_{\mathbf{P},\infty} \leq \delta \quad &\text{if and only if} \quad \|\mathbf{v}_\parallel\|_\infty \leq \delta \text{ and } \|\mathbf{v}_\perp\|_2 \leq \delta. \end{aligned}$$

Thus, from (9), the trust-region subproblem is given for the $(\mathbf{P},2)$ norm by

$$\underset{\|\mathbf{Pv}\|_{\mathbf{P},2}\leq\delta}{\text{minimize}}\ q(\mathbf{v}) = \underset{\|\mathbf{v}_\parallel\|_2\leq\delta}{\text{minimize}}\ q_\parallel\left(\mathbf{v}_\parallel\right) + \underset{\|\mathbf{v}_\perp\|_2\leq\delta}{\text{minimize}}\ q_\perp\left(\mathbf{v}_\perp\right), \tag{12}$$

and using the $(\mathbf{P},\infty)$ norm it is given by

$$\underset{\|\mathbf{Pv}\|_{\mathbf{P},\infty}\leq\delta}{\text{minimize}}\ q(\mathbf{v}) = \underset{\|\mathbf{v}_\parallel\|_\infty\leq\delta}{\text{minimize}}\ q_\parallel\left(\mathbf{v}_\parallel\right) + \underset{\|\mathbf{v}_\perp\|_2\leq\delta}{\text{minimize}}\ q_\perp\left(\mathbf{v}_\perp\right). \tag{13}$$

As shown in [3], these norms are equivalent to the two-norm, i.e.,

$$\begin{aligned} \frac{1}{\sqrt{2}}\|\mathbf{p}\|_2 \ \leq \ &\|\mathbf{p}\|_{\mathbf{P},2} \ \leq \ \|\mathbf{p}\|_2 \\ \frac{1}{\sqrt{k+1}}\|\mathbf{p}\|_2 \ \leq \ &\|\mathbf{p}\|_{\mathbf{P},\infty} \ \leq \ \|\mathbf{p}\|_2. \end{aligned}$$

Note that the latter equivalence factor depend on the number of stored quasi-Newton pairs $(k+1)$ and not on the number of variables (n).

Notice that the shape-changing norms do not place equal value on the two subspace since the region defined by the subspaces is of different size and shape in each of them. However, because of norm equivalence, the shape-changing region insignificantly differs from the region defined by the two-norm, the most commonly-used choice of norm.

We now show how to solve the decoupled subproblems.

3.3. **Solving for the optimal $\mathbf{v}_\perp^*$.** The subproblem

$$\underset{\|\mathbf{v}_\perp\|_2\leq\delta}{\text{minimize}} \quad q_\perp\left(\mathbf{v}_\perp\right) \equiv \mathbf{g}_\perp^T\mathbf{v}_\perp + \frac{\gamma}{2}\|\mathbf{v}_\perp\|_2^2 \tag{14}$$

appears in both (12) and (13); its optimal solution can be computed by formula. For the quadratic subproblem (14) the solution $\mathbf{v}_\perp^*$ must satisfy the following optimality conditions found in [12, 20, 24] associated with (14): For some $\sigma_\perp^* \in \mathbb{R}^+$,

$$(\gamma + \sigma_\perp^*)\,\mathbf{v}_\perp^* = -\mathbf{g}_\perp, \tag{15a}$$

$$\sigma_\perp^*\left(\|\mathbf{v}_\perp^*\|_2 - \delta\right) = 0, \tag{15b}$$

$$\|\mathbf{v}_\perp^*\|_2 \leq \delta, \tag{15c}$$

$$\gamma + \sigma_\perp^* \geq 0. \tag{15d}$$

Note that the optimality conditions are satisfied by $(\mathbf{v}_\perp^*, \sigma_\perp^*)$ given by

$$\mathbf{v}_\perp^* = \begin{cases} -\frac{1}{\gamma}\mathbf{g}_\perp & \text{if } \gamma > 0 \text{ and } \|\mathbf{g}_\perp\|_2 \leq \delta|\gamma|, \\ \delta\mathbf{u} & \text{if } \gamma \leq 0 \text{ and } \|\mathbf{g}_\perp\|_2 = 0, \\ -\frac{\delta}{\|\mathbf{g}_\perp\|_2}\mathbf{g}_\perp & \text{otherwise}, \end{cases} \tag{16}$$

and

$$\sigma_\perp^* = \begin{cases} 0 & \text{if } \gamma > 0 \text{ and } \|\mathbf{g}_\perp\|_2 \leq \delta|\gamma|, \\ \frac{\|\mathbf{g}_\perp\|_2}{\delta} - \gamma & \text{otherwise}, \end{cases} \tag{17}$$

where $\mathbf{u} \in \mathbb{R}^{n-(k+1)}$ is any unit vector with respect to the two-norm.

3.4. **Solving for the optimal $\mathbf{v}_\parallel^*$.** In this section, we detail how to solve for the optimal $\mathbf{v}_\parallel^*$ when either the $(\mathbf{P}, \infty)$-norm or the $(\mathbf{P}, 2)$-norm is used to define the trust-region subproblem.

$(\mathbf{P}, \infty)$-norm solution. If the shape-changing $(\mathbf{P}, \infty)$-norm is used in (9), then the subproblem in $\mathbf{v}_\parallel$ is

$$\underset{\|\mathbf{v}_\parallel\|_\infty \leq \delta}{\text{minimize}} \quad q_\parallel\left(\mathbf{v}_\parallel\right) = \mathbf{g}_\parallel^T\mathbf{v}_\parallel + \frac{1}{2}\mathbf{v}_\parallel^T\Lambda\mathbf{v}_\parallel. \tag{18}$$

The solution to this problem is computed by separately minimizing $(k+1)$ scalar quadratic problems of the form

$$\underset{|[\mathbf{v}_\parallel]_i| \leq \delta}{\text{minimize}} \quad q_{\parallel,i}([\mathbf{v}_\parallel]_i) = \left[\mathbf{g}_\parallel\right]_i\left[\mathbf{v}_\parallel\right]_i + \frac{\lambda_i}{2}\left(\left[\mathbf{v}_\parallel\right]_i\right)^2, \qquad 1 \leq i \leq (k+1). \tag{19}$$

The minimizer depends on the convexity of $q_{\parallel,i}$, i.e., the sign of λ_i. The solution to (19) is given as follows:

$$[\mathbf{v}_\parallel^*]_i = \begin{cases} -\frac{\left[\mathbf{g}_{||}\right]_i}{\lambda_i} & \text{if } \left|\frac{\left[\mathbf{g}_{||}\right]_i}{\lambda_i}\right| \leq \delta \text{ and } \lambda_i > 0, \\ c & \text{if } \left[\mathbf{g}_\parallel\right]_i = 0, \ \lambda_i = 0, \\ -\text{sgn}(\left[\mathbf{g}_\parallel\right]_i)\delta & \text{if } \left[\mathbf{g}_\parallel\right]_i \neq 0, \ \lambda_i = 0, \\ \pm\delta & \text{if } \left[\mathbf{g}_\parallel\right]_i = 0, \ \lambda_i < 0, \\ -\frac{\delta}{\left|\left[\mathbf{g}_{||}\right]_i\right|}\left[\mathbf{g}_\parallel\right]_i & \text{otherwise}, \end{cases} \tag{20}$$

where c is any real number in $[-\delta, \delta]$ and "sgn" denotes the signum function (see [3] for details).

$(\mathbf{P}, 2)$-norm solution: If the shape-changing $(\mathbf{P}, 2)$-norm is used in (9), then the subproblem in $\mathbf{v}_\parallel$ is

$$\underset{\|\mathbf{v}_\parallel\|_2 \leq \delta}{\text{minimize}} \quad q_\parallel\left(\mathbf{v}_\parallel\right) = \mathbf{g}_\parallel^T\mathbf{v}_\parallel + \frac{1}{2}\mathbf{v}_\parallel^T\Lambda\mathbf{v}_\parallel. \tag{21}$$

The solution $\mathbf{v}_\parallel^*$ must satisfy the following optimality conditions [12, 20, 24] associated with (21): For some $\sigma_\parallel^* \in \mathbb{R}^+$,

$$(\Lambda + \sigma_{\|}^* \mathbf{I})\mathbf{v}_{\|}^* = -\mathbf{g}_{\|}, \tag{22a}$$

$$\sigma_{\|}^* \left(\|\mathbf{v}_{\|}^*\|_2 - \delta\right) = 0, \tag{22b}$$

$$\|\mathbf{v}_{\|}^*\|_2 \leq \delta, \tag{22c}$$

$$\lambda_i + \sigma_{\|}^* \geq 0 \quad \text{for } 1 \leq i \leq (k+1). \tag{22d}$$

A solution to the optimality conditions (22a)-(22d) can be computed using the method found in [2]. For completeness, we outline the method here; this method depends on the sign of λ_1. Throughout these cases, we make use of the expression of $\mathbf{v}_{\|}$ as a function of $\sigma_{\|}$. That is, from the first optimality condition (22a), we write

$$\mathbf{v}_{\|}\left(\sigma_{\|}\right) = -\left(\Lambda + \sigma_{\|}\mathbf{I}\right)^{-1} \mathbf{g}_{\|}, \tag{23}$$

with $\sigma_{\|} \neq -\lambda_i$ for $1 \leq i \leq (k+1)$.

Case 1 ($\lambda_1 > 0$). When $\lambda_1 > 0$, the unconstrained minimizer is computed (setting $\sigma_{\|}^* = 0$):

$$\mathbf{v}_{\|}(0) = -\Lambda^{-1}\mathbf{g}_{\|}. \tag{24}$$

If $\mathbf{v}_{\|}(0)$ is feasible, i.e., $\|\mathbf{v}_{\|}(0)\|_2 \leq \delta$ then $\mathbf{v}_{\|}^* = \mathbf{v}_{\|}(0)$ is the global minimizer; otherwise, $\sigma_{\|}^*$ is the solution to the secular equation (28) (discussed below). The minimizer to the problem (21) is then given by

$$\mathbf{v}_{\|}^* = -\left(\Lambda + \sigma_{\|}^*\mathbf{I}\right)^{-1} \mathbf{g}_{\|}. \tag{25}$$

Case 2 ($\lambda_1 = 0$). If $\mathbf{g}_{\|}$ is in the range of Λ, i.e., $[g_{\|}]_i = 0$ for $1 \leq i \leq r$, then set $\sigma_{\|} = 0$ and let

$$\mathbf{v}_{\|}(0) = -\Lambda^{\dagger}\mathbf{g}_{\|},$$

where $\dagger$ denotes the pseudo-inverse. If $\|\mathbf{v}_{\|}(0)\|_2 \leq \delta$, then

$$\mathbf{v}_{\|}^* = \mathbf{v}_{\|}(0) = -\Lambda^{\dagger}\mathbf{g}_{\|}$$

satisfies all optimality conditions (with $\sigma_{\|}^* = 0$). Otherwise, i.e., if either $[\mathbf{g}_{\|}]_i \neq 0$ for some $1 \leq i \leq r$ or $\|\Lambda^{\dagger}\mathbf{g}_{\|}\|_2 > \delta$, then $\mathbf{v}_{\|}^*$ is computed using (25), where $\sigma_{\|}^*$ solves the secular equation in (28) (discussed below).

Case 3 ($\lambda_1 < 0$): If $\mathbf{g}_{\|}$ is in the range of $\Lambda - \lambda_1\mathbf{I}$, i.e., $[g_{\|}]_i = 0$ for $1 \leq i \leq r$, then we set $\sigma_{\|} = -\lambda_1$ and

$$\mathbf{v}_{\|}(-\lambda_1) = -\left(\Lambda - \lambda_1\mathbf{I}\right)^{\dagger} \mathbf{g}_{\|}.$$

If $\|\mathbf{v}_{\|}(-\lambda_1)\|_2 \leq \delta$, then the solution is given by

$$\mathbf{v}_{\|}^* = \mathbf{v}_{\|}(-\lambda_1) + \alpha\mathbf{e}_1, \tag{26}$$

where $\alpha = \sqrt{\delta^2 - \left\|\mathbf{v}_{\|}(-\lambda_1)\right\|_2^2}$. (This case is referred to as the "hard case" [7, 20].) Note that $v_{\|}^*$ satisfies the first optimality condition (22a):

$$\left(\Lambda - \lambda_1\mathbf{I}\right)\mathbf{v}_{\|}^* = \left(\Lambda - \lambda_1\mathbf{I}\right)\left(\mathbf{v}_{\|}(-\lambda_1) + \alpha\mathbf{e}_1\right) = -\mathbf{g}_{\|}.$$

The second optimality condition (22b) is satisfied by observing that

$$\|\mathbf{v}_{\|}^*\|_2^2 = \|\mathbf{v}_{\|}(-\lambda_1)\|_2^2 + \alpha^2 = \delta^2.$$

Finally, since $\sigma_{\|}^* = -\lambda_1 > 0$ the other optimality conditions are also satisfied.

On the other hand, if $[\mathbf{g}_\parallel]_i \neq 0$ for some $1 \leq i \leq r$ or $\|(\Lambda - \lambda_1 \mathbf{I})^\dagger \mathbf{g}_\parallel\|_2 > \delta$, then $\mathbf{v}_\parallel^*$ is computed using (25), where $\sigma_\parallel^*$ solves the secular equation (28).

The secular equation. We now summarize how to find a solution of the so-called *secular equation.* Note that from (23),

$$\|\mathbf{v}_\parallel(\sigma_\parallel)\|_2^2 = \sum_{i=1}^{k+1} \frac{(\mathbf{g}_\parallel)_i^2}{(\lambda_i + \sigma_\parallel)^2}.$$

If we combine the terms above that correspond to the same eigenvalues and remove the terms with zero numerators, then for $\sigma_\parallel \neq -\lambda_i$, we have

$$\|\mathbf{v}_\parallel(\sigma_\parallel)\|_2^2 = \sum_{i=1}^{\ell} \frac{\bar{a}_i^2}{(\bar{\lambda}_i + \sigma_\parallel)^2},$$

where $\bar{a}_i \neq 0$ for $i = 1, \ldots, \ell$ and $\bar{\lambda}_i$ are *distinct* eigenvalues of $\mathbf{B}$ with $\bar{\lambda}_1 < \bar{\lambda}_2 < \cdots < \bar{\lambda}_\ell$. Next, we define the function

$$\phi_\parallel(\sigma_\parallel) = \begin{cases} \dfrac{1}{\sqrt{\displaystyle\sum_{i=1}^{\ell} \frac{\bar{a}_i^2}{(\bar{\lambda}_i + \sigma_\parallel)^2}}} - \dfrac{1}{\delta} & \text{if } \sigma_\parallel \neq -\bar{\lambda}_i \text{ where } 1 \leq i \leq \ell \\ -\dfrac{1}{\delta} & \text{otherwise.} \end{cases} \tag{27}$$

From the optimality conditions (22b) and (22d), if $\sigma_\parallel^* \neq 0$, then $\sigma_\parallel^*$ solves the *secular equation*

$$\phi_\parallel(\sigma_\parallel) = 0, \tag{28}$$

with $\sigma_\parallel \geq \max\{0, -\lambda_1\}$. Note that $\phi_\parallel$ is monotonically increasing and concave on the interval $[-\lambda_1, \infty)$; thus, with a judicious choice of initial $\sigma_\parallel^0$, Newton's method can be used to efficiently compute $\sigma_\parallel^*$ in (28) (see [2]).

More details on the solution method for subproblem (21) are given in [2].

3.5. **Computing $\mathbf{p}^*$.** Given $\mathbf{v}^* = [\mathbf{v}_\parallel^* \ \mathbf{v}_\perp^*]^T$, the solution to the trust-region subproblem (1) using either the $(\mathbf{P}, 2)$ or the $(\mathbf{P}, \infty)$ norms is

$$\mathbf{p}^* = \mathbf{P}\mathbf{v}^* = \mathbf{P}_\parallel \mathbf{v}_\parallel^* + \mathbf{P}_\perp \mathbf{v}_\perp^*. \tag{29}$$

(Recall that using either of the two norms generates the same $v_\perp^*$ but different $v_\parallel^*$.) It remains to show how to form $\mathbf{p}^*$ in (29). Matrix-vector products involving $\mathbf{P}_\parallel$ are possible using (7), and thus, $\mathbf{P}_\parallel \mathbf{v}_\parallel^*$ can be computed; however, an implicit formula to compute products $\mathbf{P}_\perp$ is not available. To compute the second term, $\mathbf{P}_\perp \mathbf{v}_\perp^*$, we observe that $\mathbf{v}_\perp^*$, as given in (16), is a multiple of either $\mathbf{g}_\perp = \mathbf{P}_\perp^T \mathbf{g}$ or a vector $\mathbf{u}$ with unit length, depending on the sign of γ and the magnitude of $\mathbf{g}_\perp$. In the latter case, define $\mathbf{u} = \frac{\mathbf{P}_\perp^T \mathbf{e}_i}{\|\mathbf{P}_\perp^T \mathbf{e}_i\|_2}$, where $i \in \{1, 2, \ldots, k+2\}$ is the first index such that $\left\|\mathbf{P}_\perp^T \mathbf{e}_i\right\|_2 \neq 0$. (Such an $\mathbf{e}_i$ exists since $\text{rank}(\mathbf{P}_\perp) = n - (k+1)$.) Thus, we obtain

$$\mathbf{p}^* = \mathbf{P}_\parallel \mathbf{v}_\parallel^* + \left(\mathbf{I} - \mathbf{P}_\parallel \mathbf{P}_\parallel^T\right) \mathbf{w}^*, \tag{30}$$

where

$$\mathbf{w}^* = \begin{cases} -\frac{1}{\gamma}\mathbf{g} & \text{if } \gamma > 0 \text{ and } \|\mathbf{g}_\perp\|_2 \leq \delta|\gamma|, \\ \frac{\delta}{\left\|\mathbf{P}_\perp^T \mathbf{e}_i\right\|_2}\mathbf{e}_i & \text{if } \gamma \leq 0 \text{ and } \|\mathbf{g}_\perp\|_2 = 0, \\ -\frac{\delta}{\|\mathbf{g}_\perp\|_2}\mathbf{g} & \text{otherwise.} \end{cases} \tag{31}$$

The quantities $\|\mathbf{g}_\perp\|_2$ and $\left\|\mathbf{P}_\perp^T \mathbf{e}_i\right\|_2$ are computed using the orthogonality of P, which implies

$$\left\|\mathbf{g}_\parallel\right\|_2^2 + \|\mathbf{g}_\perp\|_2^2 = \|\mathbf{g}\|_2^2, \quad \text{and} \quad \|\mathbf{P}_\parallel^T \mathbf{e}_i\|_2^2 + \|\mathbf{P}_\perp^T \mathbf{e}_i\|_2^2 = 1. \tag{32}$$

Then $\|\mathbf{g}_\perp\|_2 = \sqrt{\|\mathbf{g}\|_2^2 - \|\mathbf{g}_\parallel\|_2^2}$ and $\|\mathbf{P}_\perp^T \mathbf{e}_i\|_2 = \sqrt{1 - \|\mathbf{P}_\parallel^T \mathbf{e}_i\|_2^2}$. Note that $\mathbf{v}_\perp^*$ is never explicitly computed.

3.6. **Computational Complexity.** We estimate the cost of one iteration using the proposed method to solve the trust-region subproblem defined by shape-changing norms (10) and (11). We make the practical assumption that $\gamma > 0$.

Theorem 1. *The dominant computational cost of solving one trust-region subproblem for the proposed method is* $4mn$ *floating point operations.*

Proof. Computational savings can be achieved by reusing previously computed matrices and not forming certain matrices explicitly. We begin by highlighting these cases. First, we do not form $\mathbf{\Psi} = \mathbf{Y} - \gamma\mathbf{S}$ explicitly. Rather, we compute matrix-vector products with $\mathbf{\Psi}$ by computing matrix-vector products with $\mathbf{Y}$ and $\mathbf{S}$. Second, to form $\mathbf{\Psi}^T\mathbf{\Psi}$, we only store and update the small $m \times m$ matrices $\mathbf{Y^T Y}$, $\mathbf{S^T Y}$, and $\mathbf{S^T S}$. This update involves only $3m$ vector inner products. Third, assuming we have already obtained the Cholesky factorization of $\mathbf{\Psi}^T\mathbf{\Psi}$ associated with the previously-stored limited-memory pairs, it is possible to update the Cholesky factorization of the new $\mathbf{\Psi}^T\mathbf{\Psi}$ at a cost of $O(m^2)$ [1, 13].

We now consider the dominant cost for a single subproblem solve. The eigendecomposition $\mathbf{RMR}^T = \mathbf{U}\hat{\mathbf{\Lambda}}\mathbf{U}^T$ costs $O(m^3) = \left(\frac{m^2}{n}\right) O(mn)$, where $m \ll n$. To compute $\mathbf{p}^*$ in (30), one needs to compute $\mathbf{v}^*$ from Section 3.4 and $\mathbf{w}^*$ from (31). The dominant cost for computing $\mathbf{v}^*$ and $\mathbf{w}^*$ is forming $\mathbf{\Psi}^T\mathbf{g}$, which requires $4mn$ operations. (In practice, this quantity is computed while solving the previous trust-region subproblem and can be stored to avoid recomputing when solving the current subproblem–see [3] for details.) Note that given $\mathbf{P}_\parallel^T\mathbf{g}$, the computation of $\mathbf{p}^*$ in (30) costs $4mn$. Finally, the dominant cost to update $\mathbf{\Psi}^T\mathbf{\Psi}$ is $4mn$. Thus, the dominant term in the total number of floating point operations is $4mn$. □

We note that the floating point operation count of $O(4kn)$ is the same cost as for L-BFGS [22].

3.7. **Characterization of global solutions.** It is possible to characterize global solutions to the trust-region subproblem defined by shape-changing norm $(\mathbf{P}, 2)$-norm. The following theorem is based on well-known optimality conditions for the two-norm trust-region subproblem [12, 20].

Theorem 2. *A vector* $\mathbf{p}^* \in \mathbb{R}^n$ *such that* $\left\|\mathbf{P}_\parallel^T \mathbf{p}^*\right\|_2 \leq \delta$ *and* $\left\|\mathbf{P}_\perp^T \mathbf{p}^*\right\|_2 \leq \delta$*, is a global solution of (1) defined by the* $(\mathbf{P}, 2)$*-norm if and only if there exists unique*

$\sigma_\|^* \geq 0$ *and* $\sigma_\perp^* \geq 0$ *such that*

$$\left(\mathbf{B}+\mathbf{C}_\|\right)\mathbf{p}^*+\mathbf{g}=0, \quad \sigma_\|^*\left(\left\|\mathbf{P}_\|^T\mathbf{p}^*\right\|_2-\delta\right)=0, \quad \sigma_\perp^*\left(\left\|\mathbf{P}_\perp^T\mathbf{p}^*\right\|_2-\delta\right)=0,$$

where $\mathbf{C}_\| \triangleq \sigma_\perp^*\mathbf{I}+\left(\sigma_\|^*-\sigma_\perp^*\right)\mathbf{P}_\|\mathbf{P}_\|^T$*, the matrix* $\mathbf{B}+\mathbf{C}_\|$ *is positive semi-definite, and* $\mathbf{P}=[\mathbf{P}_\| \ \mathbf{P}_\perp]$ *and* $\Lambda = diag(\lambda_1,\ldots,\lambda_{k+1}) = \hat{\Lambda}+\gamma\mathbf{I}$ *are as in (6).*

4. Numerical experiments

In this section, we report on numerical experiments with the proposed shape-changing SR1 (SC-SR1) algorithm implemented in MATLAB to solve limited-memory SR1 trust-region subproblems. The SC-SR1 algorithm was tested on randomly-generated problems of size $n = 10^3$ to $n = 10^6$, organized as five experiments when there is no closed-form solution to the shape-changing trust-region subproblem and one experiment designed to test the SC-SR1 method in the so-called "hard case". These six cases only occur using the $(\mathbf{P}, 2)$-norm trust region. (In the case of the $(\mathbf{P}, \infty)$ norm, $\mathbf{v}_\|^*$ has the closed-form solution given by (20).) The six experiments are outlined as follows:

- (E1) $\mathbf{B}$ is positive definite with $\|\mathbf{v}_\|(0)\|_2 \geq \delta$.
- (E2) $\mathbf{B}$ is positive semidefinite and singular with $[\mathbf{g}_\|]_i \neq 0$ for some $1 \leq i \leq r$.
- (E3) $\mathbf{B}$ is positive semidefinite and singular with $[\mathbf{g}_\|]_i = 0$ for $1 \leq i \leq r$ and $\|\Lambda^\dagger\mathbf{g}_\|\|_2 > \delta$.
- (E4) $\mathbf{B}$ is indefinite and $[\mathbf{g}_\|]_i = 0$ for $1 \leq i \leq r$ with $\|(\Lambda-\lambda_1\mathbf{I})^\dagger\mathbf{g}_\|\|_2 > \delta$.
- (E5) $\mathbf{B}$ is indefinite and $[\mathbf{g}_\|]_i \neq 0$ for some $1 \leq i \leq r$.
- (E6) $\mathbf{B}$ is indefinite and $[\mathbf{g}_\|]_i = 0$ for $1 \leq i \leq r$ with $\|\mathbf{v}_\|(-\lambda_1)\|_2 \leq \delta$ (the "hard case").

For these experiments, $\mathbf{S}$, $\mathbf{Y}$, and $\mathbf{g}$ were randomly generated and then altered to satisfy the requirements described above by each experiment. All randomly-generated vectors and matrices were formed using the MATLAB `randn` command, which draws from the standard normal distribution. The initial SR1 matrix was set to $\mathbf{B}_0 = \gamma\mathbf{I}$, where $\gamma = |10 * \texttt{randn(1)}|$. Finally, the number of limited-memory updates $(k+1)$ was set to 5, and r was set to 2. In the five cases when there is no closed-form solution, SC-SR1 uses Newton's method to find a root of $\phi_\|$. We use the same procedure as in [2, Algorithm 2] to initialize Newton's method since it guarantees monotonic and quadratic convergence to σ^*. The Newton iteration was terminated when the ith iterate satisfied $\|\phi_\|(\sigma^i)\| \leq \texttt{eps}\cdot\|\phi_\|(\sigma^0)\| + \sqrt{\texttt{eps}}$, where σ^0 denotes the initial iterate for Newton's method and `eps` is machine precision. This stopping criteria is both a relative and absolute criteria, and it is the only stopping criteria used by SC-SR1.

In order to report on the accuracy of the subproblem solves, we make use of the optimality conditions found in Theorem 2. For each experiment, we report the following: (i) the norm of the residual of the first optimality condition, `opt 1` $\triangleq \|(\mathbf{B}+\mathbf{C}_\|)\mathbf{p}^*+\mathbf{g}\|_2$, (ii) the combined complementarity condition, `opt 2` $\triangleq |\sigma_\|^*(\|\mathbf{P}_\|^T\mathbf{p}^*\|_2-\delta)| + |\sigma_\perp^*(\|\mathbf{P}_\perp^T\mathbf{p}^*\|_2-\delta)|$, (iii) $\sigma_\|^*+\lambda_1$, (iv) $\sigma_\perp^*+\gamma$, (v) $\sigma_\|^*$, (vi) $\sigma_\perp^*$, (vii) the number of Newton iterations ("itns"), and (viii) time. The quantities (iii) and (iv) are reported since the optimality condition that $\mathbf{B}+\mathbf{C}_\|$ is a positive semidefinite matrix is equivalent to $\gamma+\sigma_\perp^* \geq 0$ and $\lambda_i+\sigma_\|^* \geq 0$, for $1 \leq i \leq (k+1)$.

Finally, we ran each experiment five times and report one representative result for each experiment.

TABLE 1. Experiment 1: $\mathbf{B}$ is positive definite with $\|\mathbf{v}_{\|}(0)\|_2 \geq \delta$.

n	opt 1	opt 2	$\sigma_{\|}^* + \lambda_1$	$\sigma_{\perp}^* + \gamma$	$\sigma_{\|}^*$	$\sigma_{\perp}^*$	itns	time
1.0e+03	3.09e-14	2.74e-12	4.50e+00	4.86e+01	1.14e+00	4.72e+01	2	6.99e-04
1.0e+04	4.64e-14	3.59e-11	5.78e+00	3.07e+02	1.58e+00	3.07e+02	2	1.51e-03
1.0e+05	4.05e-13	9.99e-14	4.09e+00	9.01e+02	1.03e+00	9.00e+02	2	1.39e-02
1.0e+06	8.69e-12	1.35e-09	9.73e+00	4.08e+03	2.43e+00	4.08e+03	1	1.87e-01

TABLE 2. Experiment 2: $\mathbf{B}$ is positive semidefinite and singular and $[\mathbf{g}_{\|}]_i \neq 0$ for some $1 \leq i \leq r$.

n	opt 1	opt 2	$\sigma_{\|}^* + \lambda_1$	$\sigma_{\perp}^* + \gamma$	$\sigma_{\|}^*$	$\sigma_{\perp}^*$	itns	time
1.0e+03	1.60e-14	8.21e-15	1.01e+02	1.14e+03	1.01e+02	1.13e+03	3	8.62e-04
1.0e+04	1.92e-13	5.10e-09	2.22e+00	1.87e+02	2.22e+00	1.85e+02	3	1.45e-03
1.0e+05	1.83e-12	1.55e-11	5.41e+00	1.04e+03	5.41e+00	1.00e+03	2	1.44e-02
1.0e+06	5.06e-12	3.74e-12	4.75e+02	2.26e+05	4.75e+02	2.26e+05	2	1.88e-01

TABLE 3. Experiment 3: $\mathbf{B}$ is positive semidefinite and singular with $[\mathbf{g}_{\|}]_i = 0$ for $1 \leq i \leq r$ and $\|\Lambda^{\dagger}\mathbf{g}_{\|}\|_2 > \delta$.

n	opt 1	opt 2	$\sigma_{\|}^* + \lambda_1$	$\sigma_{\perp}^* + \gamma$	$\sigma_{\|}^*$	$\sigma_{\perp}^*$	itns	time
1.0e+03	4.88e-14	1.46e-14	3.87e+00	3.51e+02	3.87e+00	3.45e+02	2	5.82e-04
1.0e+04	1.91e-13	5.05e-11	3.71e+00	6.29e+02	3.71e+00	6.24e+02	1	1.33e-03
1.0e+05	1.94e-12	7.18e-13	5.06e+00	3.24e+03	5.06e+00	3.23e+03	2	1.38e-02
1.0e+06	1.99e-11	1.88e-11	4.88e+00	1.40e+04	4.88e+00	1.40e+04	1	1.90e-01

TABLE 4. Experiment 4: $\mathbf{B}$ is indefinite and $[\mathbf{g}_{\|}]_i = 0$ for $1 \leq i \leq r$ with $\|(\Lambda - \lambda_1\mathbf{I})^{\dagger}\mathbf{g}_{\|}\|_2 > \delta$.

n	opt 1	opt 2	$\sigma_{\|}^* + \lambda_1$	$\sigma_{\perp}^* + \gamma$	$\sigma_{\|}^*$	$\sigma_{\perp}^*$	itns	time
1.0e+03	2.81e-14	8.28e-15	6.29e+00	1.19e+03	7.13e+00	1.18e+03	2	6.53e-04
1.0e+04	1.25e-13	6.94e-14	3.70e+00	1.67e+03	4.52e+00	1.66e+03	2	1.89e-03
1.0e+05	2.99e-12	2.67e-12	7.77e+00	6.66e+03	8.41e+00	6.65e+03	1	1.40e-02
1.0e+06	1.92e-11	1.73e-11	7.06e+00	1.75e+04	7.19e+00	1.75e+04	1	1.88e-01

Tables I-VI show the results of the experiments. In all tables, the residual of the two optimality conditions `opt 1` and `opt 2` are on the order of $1e^{-10}$ or smaller. Columns 4 and 5 in all the tables show that $\sigma_{\|}^* + \lambda_1$ and $\sigma_{\perp}^* + \gamma$ are nonnegative with $\sigma_{\|} \geq 0$ and $\sigma_{\perp} \geq 0$ (Columns 6 and 7, respectively). Thus, the solutions obtained by SC-SR1 for these experiments satisfy the optimality conditions to high accuracy.

Also reported in each table are the number of Newton iterations. In the first five experiments no more than four Newton iterations were required to obtain $\sigma_{\|}$ to high accuracy (Column 8). In the hard case, no Newton iterations are required since $\sigma_\parallel^* = -\lambda_1$. This is reflected in Table VI, where Column 4 shows that $\sigma_\parallel^* = -\lambda_1$ and Column 8 reports no Newton iterations.)

The final column reports the time required by SC-SR1 to solve each subproblem. Consistent with the best limited-memory methods, the time required to solve each subproblem appears to grow linearly with n, as predicted in Section 3.6.

Additional experiments were run with $\mathbf{g}_\parallel \to 0$. In particular, the experiments were rerun with g scaled by factors of $10^{-2}, 10^{-4}, 10^{-6}, 10^{-8}$, and 10^{-10}. All experiments resulted in tables similar to those in Tables I-VI: the optimality conditions were satisfied to high accuracy, no more than three Newton iterations were required in any experiment to find $\sigma_\parallel^*$, and the CPU times are similar to those found in the tables.

TABLE 5. Experiment 5: $\mathbf{B}$ is indefinite and $[\mathbf{g}_\parallel]_i \neq 0$ for some $1 \leq i \leq r$.

n	opt 1	opt 2	$\sigma_\parallel^* + \lambda_1$	$\sigma_\perp^* + \gamma$	$\sigma_\parallel^*$	$\sigma_\perp^*$	itns	time
1.0e+03	9.49e-15	3.10e-12	4.94e-01	7.49e+01	1.38e+00	6.71e+01	2	6.39e-04
1.0e+04	1.07e-13	1.19e-14	1.22e+01	8.54e+02	1.23e+01	8.43e+02	4	1.63e-03
1.0e+05	7.77e-13	2.27e-13	3.98e-01	3.45e+02	9.45e-01	3.40e+02	3	1.38e-02
1.0e+06	9.75e-12	1.50e-10	3.27e-01	1.23e+03	1.26e+00	1.23e+03	2	1.91e-01

TABLE 6. Experiment 6: $\mathbf{B}$ is indefinite and $[\mathbf{g}_\parallel]_i = 0$ for $1 \leq i \leq r$ with $\|\mathbf{v}_\parallel(-\lambda_1)\|_2 \leq \delta$ (the "hard case").

n	opt 1	opt 2	$\sigma_\parallel^* + \lambda_1$	$\sigma_\perp^* + \gamma$	$\sigma_\parallel^*$	$\sigma_\perp^*$	itns	time
1.0e+03	1.62e-14	6.37e-15	0.00e+00	9.37e+02	6.08e-01	9.17e+02	0	1.33e-03
1.0e+04	1.00e-13	8.04e-14	0.00e+00	3.66e+02	9.19e-01	3.62e+02	0	1.39e-03
1.0e+05	1.52e-12	4.25e-13	0.00e+00	1.28e+03	5.59e-01	1.28e+03	0	1.54e-02
1.0e+06	1.38e-11	1.07e-11	0.00e+00	9.50e+03	9.93e-01	9.49e+03	0	2.05e-01

5. CONCLUDING REMARKS

In this paper, we presented a high-accuracy trust-region subproblem solver for when the Hessian is approximated by L-SR1 matrices. The method makes use of special shape-changing norms that decouple the original subproblem into two separate subproblems, one of which has a closed-form solution. Numerical experiments verify that solutions are computed to high accuracy in cases when there are no closed-form solutions and also in the so-called "hard case".

REFERENCES

[1] J. M. Bennett. Triangular factors of modified matrices. *Numerische Mathematik*, 7(3):217–221, 1965.
[2] 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.
[3] O. Burdakov, L. Gong, S. Zikrin, and Y.-X. Yuan. On efficiently combining limited-memory and trust-region techniques. *Mathematical Programming Computation*, pages 1–34, 2016.
[4] O. Burdakov and Y.-X. Yuan. On limited-memory methods with shape changing trust region. In *Proceedings of the First International Conference on Optimization Methods and Software*, page p. 21, 2002.

[5] 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.

[6] A. R. Conn, N. I. M. Gould, and P. L. Toint. Convergence of quasi-Newton matrices generated by the symmetric rank one update. *Math. Programming*, 50(2, (Ser. A)):177–195, 1991.

[7] A. R. Conn, N. I. M. Gould, and P. L. Toint. *Trust-Region Methods*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2000.

[8] J. B. Erway and P. E. Gill. A subspace minimization method for the trust-region step. *SIAM Journal on Optimization*, 20(3):1439–1461, 2010.

[9] J. B. Erway, P. E. Gill, and J. D. Griffin. Iterative methods for finding a trust-region step. *SIAM Journal on Optimization*, 20(2):1110–1131, 2009.

[10] J. B. Erway and R. F. Marcia. Algorithm 943: MSS: MATLAB software for L-BFGS trust-region subproblems for large-scale optimization. *ACM Transactions on Mathematical Software*, 40(4):28:1–28:12, June 2014.

[11] J. B. Erway and R. F. Marcia. On efficiently computing the eigenvalues of limited-memory quasi-newton matrices. *SIAM Journal on Matrix Analysis and Applications*, 36(3):1338–1359, 2015.

[12] D. M. Gay. Computing optimal locally constrained steps. *SIAM J. Sci. Statist. Comput.*, 2(2):186–197, 1981.

[13] P. E. Gill, G. H. Golub, W. Murray, and M. A. Saunders. Methods for modifying matrix factorizations. *Mathematics of Computation*, 28(126):505–535, 1974.

[14] D. Goldfarb. The use of negative curvature in minimization algorithms. Technical Report 80-412, Cornell University, 1980.

[15] N. I. M. Gould, D. P. Robinson, and H. S. Thorne. On solving trust-region and other regularised subproblems in optimization. *Mathematical Programming Computation*, 2(1):21–57, 2010.

[16] I. Griva, S. G. Nash, and A. Sofer. *Linear and nonlinear programming*. Society for Industrial and Applied Mathematics, Philadelphia, 2009.

[17] C. T. Kelley and E. W. Sachs. Local convergence of the symmetric rank-one iteration. *Computational Optimization and Applications*, 9(1):43–63, 1998.

[18] H. F. Khalfan, R. H. Byrd, and R. B. Schnabel. A theoretical and experimental study of the symmetric rank-one update. *SIAM Journal on Optimization*, 3(1):1–24, 1993.

[19] X. Lu. *A study of the limited memory SR1 method in practice*. PhD thesis, Department of Computer Science, University of Colorado at Boulder, 1996.

[20] J. J. Moré and D. C. Sorensen. Computing a trust region step. *SIAM J. Sci. and Statist. Comput.*, 4:553–572, 1983.

[21] K. G. Murty and S. N. Kabadi. Some np-complete problems in quadratic and nonlinear programming. *Mathematical Programming*, 39(2):117–129, 1987.

[22] J. Nocedal. Updating quasi-Newton matrices with limited storage. *Math. Comput.*, 35:773–782, 1980.

[23] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, 2nd edition, 2006.

[24] D. C. Sorensen. Newton's method with a model trust region modification. *SIAM J. Numer. Anal.*, 19(2):409–426, 1982.

[25] T. Steihaug. The conjugate gradient method and trust regions in large scale optimization. *SIAM J. Numer. Anal.*, 20:626–637, 1983.

[26] W. Sun and Y.-x. Yuan. *Optimization theory and methods*, volume 1 of *Springer Optimization and Its Applications*. Springer, New York, 2006. Nonlinear programming.

[27] B. Vavasis. *Nonlinear Optimization: Complexity Issues*. International Series of Monographs on Computer Science. Oxford University Press, Oxford, England, 1992.

[28] H. Wolkowicz. Measures for symmetric rank-one updates. *Mathematics of Operations Research*, 19(4):815–830, 1994.

E-mail address: jbrust@ucmerced.edu

Applied Mathematics, University of California, Merced, Merced, CA 95343

E-mail address: oleg.burdakov@liu.se

Department of Mathematics, Linköping University, SE-581 83 Linköping, Sweden

E-mail address: erwayjb@wfu.edu

Department of Mathematics, Wake Forest University, Winston-Salem, NC 27109

E-mail address: rmarcia@ucmerced.edu

Applied Mathematics, University of California, Merced, Merced, CA 95343

E-mail address: yyx@lsec.cc.ac.cn

State Key Laboratory of Scientific and Engineering, Institute of Computational Mathematics and Scientific/Engineering Computing, AMSS, Chinese Academy of Sciences, Beijing, China.