Optimization Methods and Software

ISSN: (Print) (Online) Journal homepage: www.tandfonline.com/journals/goms20

Shape-changing trust-region methods using multipoint symmetric secant matrices

J. J. Brust, J. B. Erway & R. F. Marcia

To cite this article: J. J. Brust, J. B. Erway & R. F. Marcia (24 Jan 2024): Shape-changing trust-region methods using multipoint symmetric secant matrices, Optimization Methods and Software, DOI: 10.1080/10556788.2023.2296441

To link to this article: https://doi.org/10.1080/10556788.2023.2296441

OPTIMIZATION METHODS & SOFTWARE
https://doi.org/10.1080/10556788.2023.2296441

Taylor & Francis
Taylor & Francis Group

Shape-changing trust-region methods using multipoint symmetric secant matrices

J. J. Brust [a], J. B. Erway [b] and R. F. Marcia[c]

[a]Mathematics, University of California San Diego, La Jolla, CA, USA; [b]Mathematics, Wake Forest University, Winston-Salem, NC, USA; [c]Applied Mathematics, University of California Merced, Merced, CA, USA

ABSTRACT
In this work, we consider methods for large-scale and nonconvex unconstrained optimization. We propose a new trust-region method whose subproblem is defined using a so-called 'shape-changing' norm together with densely-initialized multipoint symmetric secant (MSS) matrices to approximate the Hessian. Shape-changing norms and dense initializations have been successfully used in the context of traditional quasi-Newton methods, but have yet to be explored in the case of MSS methods. Numerical results suggest that trust-region methods that use densely-initialized MSS matrices together with shape-changing norms outperform MSS with other trust-region methods.

ARTICLE HISTORY
Received 23 September 2022
Accepted 13 December 2023

KEYWORDS
Large-scale optimization; nonconvex optimization; trust-region methods; quasi-Newton methods

MATHEMATICS SUBJECT CLASSIFICATIONS
65K110; 90C30; 90C06; 49M37

1. Introduction

In this paper, we propose a new solver for general large nonconvex problems of the following form:

$$\min f(x), \tag{1}$$

where $x \in \Re^n$ and f is continuously differentiable. Generally speaking, solvers for unconstrained optimization fall into three categories: Line search (see, e.g. [21,26], trust-region methods [15,31], and cubic regularization methods [14,25]. While traditional line search methods that use local quadratic models require each of these models are convex, trust-region methods are able to approximate nonconvexity in the underlying function using nonconvex quadratic models. In this work, we focus on trust-region methods with possibly-indefinite Hessian approximations.

Trust-region methods generate a sequence of iterates $\{x_k\}$ by solving at each iteration a *trust-region* subproblem:

$$\min_{s\in\Re^n} Q(s) = g_k^T s + \frac{1}{2} s^T B_k s \quad \text{subject to } \|s\| \leq \delta, \tag{2}$$

where $g_k = \nabla f(x_k)$ and $B_k \approx \nabla^2 f(x_k)$. Given x_k, the next iterate is then computed as $x_{k+1} = x_k + s_k$, where s_k is an approximate solution to (2). The choice of norm in (2)

CONTACT J. B. Erway erwayjb@wfu.edu Department of Mathematics,Wake Forest University, PO Box 7388, Winston-Salem, NC 27109, USA

affects the difficulty to solve the trust-region subproblem. In this manuscript, we consider two *shape-changing* [7] norms to define the trust region. By using one of the norms, the trust-region subproblem decouples into two subproblems, each with closed-form solutions. Meanwhile, the other shape-changing norm allows for a similar decoupling into two subproblems, one with a closed-form solution and the other that is a low-dimensional trust-region subproblem that is easily solved.

In large-scale optimization, it can be the case that the Hessian of the objective function is either too computationally expensive to compute or store. In these cases, first-order methods such as steepest descent and quasi-Newton methods may give the fastest convergence. Multipoint symmetric secant (MSS) methods can be thought of as generalizations of quasi-Newton methods in that they attempt to enforce multiple secant conditions at one time. As with quasi-Newton methods, MSS methods generate a sequence of matrices $\{B_k\}$ to approximate the Hessian of f at x_k using a sequence of low-rank updates. Specifically, at each iteration, B_k is updated using a recursion relation where $B_{k+1} = B_k + U_k$, and U_k is a low-rank update (e.g.rank$(U_k) \leq 2$). However, these methods must be user-initiated by selecting an initial B_0. Conventionally, the initial matrix for limited-memory quasi-Newton matrices is chosen to be a scalar multiple of the identity, e.g. $B_0 = \gamma_k I$, $\gamma_k \in \Re$. This choice leads to minimal storage requirements; only γ_k must be stored in addition to the quasi-Newton pairs. More recently, dense initializations have been proposed that are also low-memory initializations in which only two constants must be stored in addition to the usual quasi-Newton pairs [3,19]. These dense initializations implicitly decompose $\Re^n$ into two orthogonal subspaces (where one subspace corresponds to the span of the eigenvectors that have already been computed and the other corresponding to its orthogonal complement), assigning a parameter to each subspace. The dense initialization can be viewed as a generalization of the traditional initialization, i.e. the scalar multiple of the identity. With a judicious choice of parameters the dense initialization has been shown to outperform the traditional initialization [3,19]. In this paper, we consider the same splitting of $\Re^n$ in a shape-changing trust-region setting.

In this manuscript, we propose a densely-initialized MSS method that relies on a new compact formulation of MSS matrices. This new compact formulation allows for a more computationally efficient solution to trust-region subproblems for solving large nonconvex problems of the form (1). The motivation for this research comes from three different recent papers on first-order methods for solving general nonconvex unconstrained optimization problems where (1) a limited-memory symmetric rank-one (L-SR1) quasi-Newton trust-region method with a shape-changing norm was found to be competitive with other standard trust-region methods [4], (2) a limited-memory Broyden-Fletcher-Goldfab-Shanno (L-BFGS) method with a dense initialization together with a shape-changing trust-region method outperformed other L-BFGS trust-region methods [3] and (3) a MSS trust-region method that uses the Euclidean norm to define the trust region together with a dense initialization outperformed both conventional initalizations and other standard quasi-Newton trust-region methods [19].

This paper is organized in five sections. Background on MSS matrices, shape-changing norms, and the dense initialization is given in Section 2. In Section 3, we present the contributions of this research; namely, we propose densely-initialized MSS trust-region methods that use the shape-changing norms. Numerical results on the CUTEst test set are presented in Section 4 comparing the proposed method to other quasi-Newton methods. Finally, concluding remarks are in Section 5.

1.1. Notation and glossary

Throughout this paper, capital letters denote matrices and lower-case letter are reserved for vectors. Moreover, vectors with an asterisk denote optimal solutions. The symbol e_i denotes the ith canonical basis vector whose dimension depends on context. Finally, 'sgn' denotes the signum function.

1.2. Dedication

We dedicate this paper to Oleg P. Burdakov. The work presented here synthesizes three of Oleg's many areas of research: shape-changing norms [4,6,7], dense initalizations of quasi-Newton methods [3], and MSS and secant methods [8–10,12]. This manuscript is written in his memory. *Nostrovia*, Oleg!

2. Background

In this section, we review MSS matrices, including a recursion relation and compact formulation, and the shape-changing norm.

2.1. MSS matrices

MSS matrices are generated similarly to traditional quasi-Newton matrices: A sequence of matrices $\{B_k\}$ is recursively formed using a sequence of low-rank updates. Specifically, if $\{x_k\}$ is a sequence of updates obtained to solve (1), then define $s_k = x_{k+1} - x_k$ and $y_k = \nabla f(x_{k+1}) - f(x_k)$. The pairs $\{(s_k, y_k)\}$ are often referred to as *quasi-Newton* pairs. In methods for large-scale optimization, *limited-memory* versions of these methods are used that store only the most recently-computed m pairs; m is typically referred to as the 'memory' of the method. In practice, $m \ll n$, e.g. $m \in [3, 7]$ (see, e.g. [13]). In this work, we assume a limited-memory framework where m is small; further, we let l denote the current number of stored pairs.

Let the matrices S_k and Y_k denote the matrices whose columns are formed by the stored quasi-Newton pairs:

$$S_k = [s_{k-1} \;\; s_{k-2} \;\; \dots \;\; s_{k-l}] \in \Re^{n\times l} \quad \text{and} \quad Y_k = [y_{k-1} \;\; y_{k-2} \;\; \dots \;\; y_{k-l}] \in \Re^{n\times l}, \tag{3}$$

where $l \leq m$. While quasi-Newton matrices must satisfy the so-called *secant condition* $B_{k+1}s_k = y_k$, MSS matrices seek to satisfy multiple secant conditions: $B_kS_k = Y_k$. Generally speaking, it is impossible to satisfy the secant conditions and also require B_k to be symmetric [30]. (This can be seen by multiplying the multiple secant equations by S_k^T on the left and noticing that while $S_k^TB_kS_k$ is symmetric whenever $B_k = B_k^T$, it is not generally true that $S_k^TY_k$ will be symmetric [30].)

In [8,11,12], Burdakov proposes relaxing the secant conditions by symmetrizing the product $S_k^TY_k$ using the following symmetrization:

$$\text{sym}(A) = \begin{cases} A_{ij}, & i \geq j \\ A_{ji}, & i < j. \end{cases}$$

With this symmetrization, $S_k^T B_k S_k = \text{sym}(S_k^T Y_k)$ and yields the following recursion relation for B_k:

$$B_{k+1} = B_k + \frac{(y_k - B_k s_k)c_k^T + c_k(y_k - B_k s_k)^T}{s_k^T c_k} - \frac{(y_k - B_k s_k)^T s_k c_k c_k^T}{(s_k^T c_k)^2}, \tag{4}$$

where $c_k \in \Re^n$ is any vector such that $c_k^T s_i = 0$ for all $0 \leq i < k$ and $c_k^T s_k \neq 0$ [2,8]. Notice that this is a rank-two update; in practice, this update can generate indefinite approximations to the Hessian. More generally, the recursion relation (4), without the orthogonality conditions on c_k, is not new in the literature: When $c_k = s_k$, the update is the Powell-symmetric-Broyden (PSB), and $c_k = y_k$ yields the Davidon-Fletcher-Powell (DFP) update.

Given any initial B_0, the general compact formulation for MSS matrices is $B_k = B_0 + \Psi_k M_k \Psi_k^T$, where

$$\Psi_k \triangleq [S_k \quad (Y_k - B_0 S_k)] \quad \text{and} \quad M_k \triangleq \begin{bmatrix} W(S_k^T B_0 S_k - (T_k + E_k + T_k^T))W & W \\ W & 0 \end{bmatrix}, \tag{5}$$

where $W = (S_k^T S_k)^{-1}$, T_k is the strict upper triangular portion of $S_k^T Y_k$, and E_k is the diagonal of $S_k^T Y_k$ [2]. The compact formulation requires that S_k has full rank. One way this can be accomplished is using a rank-revealing decomposition and then removing columns of S_k that are linearly dependent. In [19], the LDL^T decomposition of $S_k^T S_k$ is used to find linear dependence. In this case, if a column of S_k is removed then the corresponding column of Y_k must also be removed in order for M_k to be well-defined. See [19] for a full discussion.

2.2. The spectral decomposition

Consider the compact formulation of B_k, with (5):

$$B_k = B_0 + \Psi_k M_k \Psi_k^T,$$

where $\Psi_k \in \Re^{n \times 2l}$, $M_k \in \Re^{2l \times 2l}$, and l is the number of stored quasi-Newton pairs. Then, given the so-called 'thin' QR factorization of Ψ_k, namely $\Psi_k = QR$, we obtain the following expression for B_k:

$$B_k = B_0 + QRM_kR^TQ^T, \tag{6}$$

where $Q \in \Re^{n \times 2l}$, $R \in \Re^{2l \times 2l}$, and $RM_kR^T \in \Re^{2l \times 2l}$. Because of its small size, the spectral decomposition of RM_kR^T is computable. Suppose $U\hat{\Lambda}U^T$ is the spectral decomposition of RM_kR^T with $\hat{\Lambda} = \text{diag}\{\hat{\lambda}_1, \ldots, \hat{\lambda}_{2l}\}$, and $B_0 = \gamma_k I$ is the conventional one-parameter initialization. Then, B_k can be written as $B_k = P\Lambda P^T$, where

$$P = [QU \quad (QU)^\perp] \quad \text{and} \quad \Lambda = \begin{bmatrix} \hat{\Lambda} + \gamma_k I & 0 \\ 0 & \gamma_k I \end{bmatrix}.$$

While the above derivation is found in [3–6,18], in practice, the factorization may also be accomplished using the LDL^T factorization [3,4,6,19]. (For more details on this spectral decomposition of B_k see [3,4,6,19].) For simplicity, we define

$$P_\parallel = QU \quad \text{and} \quad P_\perp = (QU)^\perp, \tag{7}$$

and make use of these definitions throughout the duration of the manuscript.

In order to solve the trust-region subproblem at each iteration, it will be necessary to be able to implicitly perform matrix-vector products with $P_{\|}$. It is possible to form $P_{\|}$ without storing Q. To see this, note that

$$P_{\|} = QU = \Psi_k R^{-1} U. \tag{8}$$

In order for R to be invertible, Ψ_k must have full rank. Similar to [3,4,6,19], we propose using the LDL^T decomposition to identify columns of Ψ_k that are linearly dependent. For a full discussion on ensuring both S_k and Ψ_k have linearly independent columns, see [19].

2.3. Shape-changing norms

Trust-region subproblems have the form of (2), but any norm may be used in the constraint that defines the trust region. The most commonly-chosen norm is the Euclidean norm; other popular choices found in the literature are the one-norm and infinity-norm. One important advantage in using the Euclidean norm to define the trust region is that there are optimality conditions that characterize a global solution [20,24]:

Theorem 2.1: *The vector $s^* \in \Re^n$ is a global solution of*

$$\min_{s \in \Re^n} Q(s) = g_k^T s + \frac{1}{2} s^T B s \quad \textit{subject to } \|s\|_2 \leq \delta, \tag{9}$$

if and only if $\|s^*\|_2 \leq \delta$ and there exists a unique $\sigma^* \geq 0$ such that $B + \sigma^* I$ is positive semidefinite and

$$(B + \sigma^* I)s^* = -g \quad \textit{and} \quad \sigma^*(\delta - \|s^*\|_2) = 0. \tag{10}$$

So-called 'exact' subproblems solvers aim to explicitly find a pair (s^*, σ^*) that satisfies (10) in order to solve each subproblem to high accuracy [5,6,17,24]. While convergence proofs of the overall trust-region method only require an approximate solution of each subproblem [27–29], these 'exact' solvers bet on that solving each subproblem to high precision will lead to fewer overall iterations of the trust-region method.

The 'shape-changing' norms were first defined by Burdakov and Yuan [7]. These shape-changing norms make use of the matrix of eigenvectors $P_{\|}$ and $P_{\perp}$ (see 7), and thus, the size and shape of the trust-region change every iteration:

$$\|s\|_{P,\infty} = \max(\|P_{\|}^T s\|_{\infty}, \|P_{\perp}^T s\|_2) \tag{11}$$

$$\|s\|_{P,2} = \max(\|P_{\|}^T s\|_2, \|P_{\perp}^T s\|_2). \tag{12}$$

For simplicity, we refer to (11) as the (P, ∞) norm, and (12) as the $(P, 2)$ norm.

In [6], Burdakov et al. show that these norms are equivalent to the Euclidean norm and the equivalence factors are independent of P. Importantly, these norms allow each subproblem to be decomposed into two small subproblems, each of which are either easy to solve or have a closed-form solution. These norms have been successfully used in L-BFGS and L-SR1 trust-region settings [4,6]. For more details on these norms, see [6,7]. Implementation details for solving trust-region subproblems defined using the shape-changing norm are presented in Section 3.2.

2.4. The dense initialization

The conventional initialization for a quasi-Newton method is a constant diagonal initialization, i.e. $B_0 = \gamma_k I$, $\gamma_k \in \Re$. This initialization performs well in practice and enjoys ease of use with no additional memory requirements other than storing a scalar–for these reasons it is the most popular initialization. Other low-memory initializations include nonconstant diagonal matrices. Until the dense initalization was first proposed for L-BFGS matrices, low-memory intializations were limited to diagonal matrices.

The dense initialization exploits the partitioning of $\Re^n$ into two subspaces: (i) the eigenspace associated with the eigenvalues

$$\hat{\lambda}_1 + \gamma_k, \ldots, \hat{\lambda}_{2l} + \gamma_k,$$

and (ii) the eigenspace associated with the eigenvalue γ_k, assuming that RM_kR^T is nonsingular. Specifically, note that

$$B_0 = \gamma_k I = \gamma_k PP^T = \gamma_k P_{\|} P_{\|}^T + \gamma_k P_{\perp} P_{\perp}^T.$$

In lieu of using one parameter for both spaces, the dense initalization uses two:

$$\tilde{B}_0 = \zeta_k P_{\|} P_{\|}^T + \zeta_k^C P_{\perp} P_{\perp}^T, \tag{13}$$

where $\zeta_k, \zeta_k^C \in \Re$. For the duration of the paper, $\tilde{B}_0$ will denote a dense initial matrix. Using the dense initialization, the compact formulation becomes $B_k = P\Lambda P^T$, where

$$P = [QU \quad (QU)^{\perp}] \quad \text{and} \quad \Lambda = \begin{bmatrix} \hat{\Lambda} + \zeta_k I & 0 \\ 0 & \zeta_k^C I \end{bmatrix}. \tag{14}$$

3. Implementation

In this section, we demonstrate how the dense initialization for an MSS method can be used in a shape-changing norm. This section presents the contributions of this research.

3.1. A second compact formulation

The spectral decomposition $B_k = P\Lambda P^T$ where P and Λ are given by (14) relies only on the existence of a compact formulation for B_k. In this subsection, we derive an alternative compact formulation for B_k in the case of a dense initialization that is compatible with the derivation of the spectral decomposition in Section 2.2.

Consider the compact formulation for the dense initialization:

$$B_k = \tilde{B}_0 + \Psi_k M_k \Psi_k^T,$$

where Ψ_k and M_k are given in (5). The following lemma appears in [19] and allows us to define an alternative compact formulation in Theorem 3.2:

Lemma 3.1: *Suppose $\tilde{B}_0$ is as in (13). Then, $\tilde{B}_0 S_k = \zeta_k S_k$.*

Proof: See [19, Corollary 3.4]. ■

Theorem 3.2: *Suppose $\tilde{B}_0$ is as in (13) and S_k is full rank, then B_k can be written as*

$$B_k = \tilde{B}_0 + \tilde{\Psi}_k \tilde{M}_k \tilde{\Psi}_k^T,$$

where

$$\tilde{\Psi}_k = [S_k \quad Y_k] \quad \textit{and} \quad \tilde{M}_k = \begin{bmatrix} -\zeta_k W - W(T_k + E_k + T_k^T)W & W \\ W & 0 \end{bmatrix}, \tag{15}$$

and $W = (S_k^T S_k)^{-1}$, T_k is the strict upper triangular portion of $S_k^T Y_k$, and E_k is the diagonal of $S_k^T Y_k$.

Proof: Consider the compact formulation where Ψ_k and M_k are given by (5) together with the dense initialization:

$$B_k = \tilde{B}_0 + \Psi_k M_k \Psi_k^T,$$

where

$$\Psi_k = [S_k \quad (Y_k - \tilde{B}_0 S_k)] \quad \text{and} \quad M_k = \begin{bmatrix} W(S_k^T \tilde{B}_0 S_k - (T_k + E_k + T_k^T))W & W \\ W & 0 \end{bmatrix}.$$

By Lemma 3.1, $\Psi_k = [S_k \;\; (Y_k - \zeta_k S_k)]$ and M_k can be simplified as follows:

$$\begin{aligned} M_k &= \begin{bmatrix} W(S_k^T \tilde{B}_0 S_k - (T_k + E_k + T_k^T))W & W \\ W & 0 \end{bmatrix} \\ &= \begin{bmatrix} \zeta_k W - W(T_k + E_k + T_k^T)W & W \\ W & 0 \end{bmatrix}. \end{aligned}$$

Putting this together yields that $\Psi_k M_k \Psi_k^T$ can be written as

$$\begin{aligned} \Psi_k M_k \Psi_k^T &= [S_k \quad Y_k - \zeta_k S_k] \begin{bmatrix} \zeta_k W - W(T_k + E_k + T_k^T)W & W \\ W & 0 \end{bmatrix} \begin{bmatrix} S_k^T \\ (Y_k - \zeta_k S_k)^T \end{bmatrix} \\ &= -\zeta_k S_k W S_k^T - S_k W (T_k + E_k + T_k^T) W S_k^T + S_k W Y_k^T + Y_k W S_k^T \\ &= [S_k \quad Y_k] \begin{bmatrix} -\zeta_k W - W(T_k + E_k + T_k^T)W & W \\ W & 0 \end{bmatrix} \begin{bmatrix} S_k^T \\ Y_k^T \end{bmatrix}. \end{aligned}$$

Thus, $B_k = \tilde{B}_0 + \hat{\Psi}_k \hat{M}_k \hat{\Psi}_k^T$, where $\tilde{\Psi}_k$ and $\tilde{M}_k$ are given by (15), respectively. ■

It is the case that the compact formulation given in Theorem 3.2 is the compact formulation Burdakov in [12] derived for the case of the conventional initialization $B_0 = \gamma_k I$, $\gamma_k \in \Re$; however, his method of derivation required a single-parameter initialization.

There are several benefits of using the compact formulation given in Theorem 3.2 over the compact formulation defined by (5). Namely, Ψ_k can be formed without any computations; whereas (5) requires scalar multiplication with ζ_k and n subtractions. Moreover, more importantly, whenever a new ζ_k is computed Ψ_k in (5) must be recomputed, possibly from scratch; however, in (15), ζ_k is not needed to form $\tilde{\Psi}_k$. It is for these reasons that the proposed method uses this second alternative compact formulation.

We note further that B_k in (15) can also be represented by

$$\widehat{\Psi}_k = [S_k W \quad Y_k] \quad \text{and} \quad \widehat{M}_k = \begin{bmatrix} -\zeta_k W^{-1} - (T_k + E_k + T_k^T) & I \\ I & 0 \end{bmatrix}.$$

Since $W^{-1} = S_k^T S_k$, $\widehat{M}_k$ can be formed without inverting a small matrix. In order to keep computational cost low, it is not necessary to form the product $S_k W$. Instead, when a product with an arbitrary vector p and $\widehat{\Psi}_k$ is needed, one can compute this using only matrix-vector products as follows: $\widehat{\Psi}_k^T p = \begin{bmatrix} W(S_k^T p) \\ Y_k^T \end{bmatrix}$. We make this representation available as an additional option for an L-MSS method; however, our numerical experiments favoured the results with (15). For this reason, we assume the representation (15) for the duration of the paper.

3.2. Solving the trust-region subproblem

In this section, we demonstrate how to solve the trust-region subproblem defined by a shape-changing norm, where an MSS matrix is used to approximate the Hessian at each iterate. We generally follow the presentations in [4,6], altering the presentation to allow for a dense initialization.

3.2.1. The (P, ∞) shape-changing norm

In this section, we consider a trust-region subproblem whose constraint is defined by the (P, ∞) norm:

$$\min_{s \in \Re^n} Q(s) = g_k^T s + \frac{1}{2} s^T B_k s \quad \text{subject to } \|s\|_{P,\infty} \leq \delta, \tag{16}$$

Consider $v = P^T s$ where $P = [P_\| \;\; P_\perp]$ as in Section 2.2. Applying this change of variables by substituting in Pv for s in (16) yields the following quadratic function:

$$Q(Pv) = g_k^T (Pv) + \tfrac{1}{2} (Pv)^T B_k (Pv). \tag{17}$$

Letting

$$v_\| = P_\|^T s, \quad v_\perp = P_\perp^T s, \quad g_\| = P_\|^T g, \quad g_\perp = P_\perp^T g,$$

then (17) simplifies as follows:

$$\begin{aligned} Q(Pv) &= g_k^T (Pv) + \tfrac{1}{2} v^T \Lambda v \\ &= g_\|^T v_\| + g_\perp^T v_\perp + \tfrac{1}{2} (v_\|^T (\hat{\Lambda} + \zeta_k I_{2l}) v_\| + \zeta_k^C \|v_\perp\|_2^2). \end{aligned} \tag{18}$$

Notice that (18) is separable, and thus, (16) can be decoupled into two trust-region subproblems:

$$\min_{v_\| \in \Re^{2l}} q_\|(v_\|) = g_\|^T v_\| + \frac{1}{2} (v_\|^T (\hat{\Lambda} + \zeta_k I_{2l}) v_\|) \quad \text{subject to } \|v_\|\|_\infty \leq \delta, \tag{19}$$

$$\min_{v_\perp \in \Re^{n-2l}} q_\perp(v_\perp) = g_\perp^T v_\perp + \frac{1}{2} \zeta_k^C \|v_\perp\|_2^2 \quad \text{subject to } \|v_\perp\|_2 \leq \delta. \tag{20}$$

Both of these subproblems have closed-form solutions. In particular, the closed-form solution to (19) is given by [4,6]:

$$[v_{\|}^*]_i = \begin{cases} -\dfrac{[g_{\|}]_i}{\lambda_i} & \text{if } \left|\dfrac{[g_{\|}]_i}{\lambda_i}\right| \leq \delta_k \text{ and } \lambda_i > 0, \\ c & \text{if } [g_{\|}]_i = 0 \text{ and } \lambda_i = 0, \\ -\text{sgn}([g_{\|}]_i)\delta_k & \text{if } [g_{\|}]_i \neq 0 \text{ and } \lambda_i = 0, \\ \pm\delta_k & \text{if } [g_{\|}]_i = 0 \text{ and } \lambda_i < 0, \\ -\dfrac{\delta_k}{|[g_{\|}]_i|}[g_{\|}]_i & \text{otherwise,} \end{cases}$$

where $\lambda_i = \hat{\lambda}_i + \zeta_k$ for $i = 1, \ldots, 2l$ and $c \in [-\delta_k, \delta_k]$. The closed form solution to (20) is

$$v_{\perp}^* = \begin{cases} -\dfrac{1}{\zeta_k^C} g_{\perp} & \text{if } \zeta_k^C > 0 \text{ and } \|g_{\perp}\|_2 \leq \delta_k |\zeta_k^C| \\ \delta_k u & \text{if } \zeta_k^C \leq 0 \text{ and } \|g_{\perp}\|_2 = 0 \\ -\dfrac{\delta_k}{\|g_{\perp}\|_2} g_{\perp} & \text{otherwise,} \end{cases}$$

where $u \in \Re^{n-2l}$ is a unit vector with respect to the two-norm [4,6]. Notice that $\|v_{\perp}^*\|_2$ is at times inversely-proportional to ζ_k^C when ζ_k^C is positive; in other words, a very large and positive ζ_k^C results in a small $\beta = 1/\zeta_k$ when $\|g_{\perp}\|_2$ is not too relatively large.

Having obtained optimal $v_{\|}^*$ and $v_{\perp}^*$, s^* can be recovered using the relationship $Pv = s$ and noting that $P_{\perp} P_{\perp}^T = (I - P_{\|} P_{\|}^T)$ to give

$$\begin{aligned} s^* &= P \begin{bmatrix} v_{\|}^* \\ v_{\perp}^* \end{bmatrix} \\ &= P_{\|} v_{\|}^* + P_{\perp} v_{\perp}^*. \end{aligned} \tag{21}$$

To compute $P_{\perp} v_{\perp}^*$, we use the same strategy as in [4]; that is, picking u to be $u = \frac{P_{\perp}^T e_i}{\|P_{\perp}^T e_i\|_2}$, where i is the first index such that $\|P_{\perp}^T e_i\| \neq 0$, then

$$s^* = P_{\|}(v_{\|}^* - P_{\|}^T w^*) + w^*, \tag{22}$$

where

$$w^* = \begin{cases} -\dfrac{1}{\zeta_k^C} g & \text{if } \zeta_k^C > 0 \text{ and } \|g_{\perp}\|_2 \leq \delta_k |\zeta_k^C| \\ \dfrac{\delta_k}{\|P_{\perp}^T e_i\|_2} e_i & \text{if } \zeta_k^C \leq 0 \text{ and } \|g_{\perp}\|_2 = 0 \\ -\dfrac{\delta_k}{\|g_{\perp}\|_2} g & \text{otherwise.} \end{cases}$$

Note that the quantities $\|g_{\perp}\|_2$ and $\|P_{\perp}^T e_i\|_2$ can be computed using following relationships:

$$\|g_{\perp}\|_2^2 + \|g_{\|}\|_2^2 = \|g\|_2^2 \quad \text{and} \quad \|P_{\perp}^T e_i\|_2^2 + \|P_{\|}^T e_i\|_2^2 = 1.$$

Thus, the solution s^* for the (P, ∞) trust-region subproblem can be computed using only $P_{\|}$ via (8).

3.2.2. The $(P, 2)$ shape-changing norm

In this section, we consider a trust-region subproblem whose constraint is defined by the $(P, 2)$ norm:

$$\min_{s\in\Re^n} Q(s) = g_k^T s + \frac{1}{2} s^T B_k s \quad \text{subject to } \|s\|_{P,2} \leq \delta, \tag{23}$$

Different from the (P, ∞)-norm, the subproblem does not have a closed-form solution; however, it can be decoupled into two subproblems–one that has a closed-form solution and one that is a low-dimensional two-norm subproblem that is easily solved. To see this, consider the same approach as in the (P, ∞)-norm case. Applying the same change of variables $v = P^T s$, yields (18) as before. The problem is separable and decouples into the following trust-region subproblems:

$$\min_{v_\parallel \in \Re^{2l}} q_\parallel(v_\parallel) = g_\parallel^T v_\parallel + \frac{1}{2}(v_\parallel^T(\hat{\Lambda} + \zeta_k I_{2l})v_\parallel) \quad \text{subject to } \|v_\parallel\|_2 \leq \delta, \tag{24}$$

$$\min_{v_\perp \in \Re^{n-2l}} q_\perp(v_\perp) = g_\perp^T v_\perp + \frac{1}{2}\zeta_k^C \|v_\perp\|_2^2 \quad \text{subject to } \|v_\perp\|_2 \leq \delta. \tag{25}$$

Since (25) is identical to (20), its closed-form solution is given in Section 3.2.1. Subproblem (24) is a low-dimensional problem since l is typically chosen to be a small number (e.g. less than 10). Moreover, $\nabla^2 q_\parallel(v_\parallel)$ is a diagonal matrix. For this reason, any standard subproblem solver (including direct methods) may be used to solve this subproblem (e.g. see [15] for possible methods). However, in this work, we propose using the method found in [5].

The OBS method found in [5] is an 'exact' subproblem solver when L-SR1 matrices are used as the approximate Hessian. The method computes solutions to satisfy optimality conditions given in Theorem 2.1 by exploiting the compact formulation of L-SR1 matrices. For this work, we use a modified version of the OBS method that makes use of the compact formulation for MSS matrices. Specifically, given an MSS matrix and its compact formulation (Section 3.1), the partial spectral decomposition $B_k = P\Lambda P^T$ can be computed as in Section 14, where

$$P = [QU \quad (QU)^\perp] \quad \text{and} \quad \Lambda = \begin{bmatrix} \hat{\Lambda} + \zeta_k I & 0 \\ 0 & \zeta_k^C I \end{bmatrix}.$$

The optimality conditions given by Theorem 2.1 for the $(P, 2)$ subproblem are as follows:

$$(\hat{\Lambda} + (\sigma^* + \zeta_k)I)v_\parallel^* = -g_\parallel, \tag{26}$$

$$\sigma^*(\|v_\parallel^*\|_2 - \delta) = 0 \tag{27}$$

$$\|v_\parallel\|_2 \leq \delta, \tag{28}$$

$$\sigma^* \geq 0, \tag{29}$$

$$\hat{\lambda}_i + (\sigma^* + \zeta_k) \geq 0 \quad \text{for } 1 \leq i \leq 2l. \tag{30}$$

Let $\hat{\lambda}_{2l}$ denote the smallest entry in $\hat{\Lambda}$. A solution of (26)–(30) can be computed by considering three general cases that depend on the sign of $\hat{\lambda}_{2l} + \zeta_k$. Details for each case is given in [4,5].

Having obtained $v_\perp^*$ and $v_\parallel^*$, the solution to the $(P, 2)$-norm shape-changing subproblem is computed using (22). As with the (P, ∞)-norm case, matrix-vector products with $P_\parallel$ do not require forming $P_\parallel$ explicitly; instead, we use the factored form in (8).

4. Numerical results

In this section, we report results of various experiments using the limited-memory MSS method (L-MSSM) and other limited-memory quasi-Newton methods. For these results, we used 60 problems from the CUTEst test set [22] with $n \geq 1000$. Specifically, all problems with the classification 'OUR2' with $n \geq 1000$ were chosen,[1] which includes all problems with an objective function that is nonconstant, nonlinear, nonquadratic, and not the sum of squares. The 60 problems were: ARWHEAD, BOX, BOXPOWER, BROYDN7D, COSINE, CRAGGLVY, CURLY10, CURLY20, CURLY30, DIXMAANA, DIXMAANB, DIXMAANC, DIXMAAND, DIXMAANE, DIXMAANF, DIXMAANG, DIXMAANH, DIXMAANI, DIXMAANJ, DIXMAANK, DIXMAANL, DIXMAANM, DIXMAANN, DIXMAANO, DIXMAANP, DQRTIC, EDENSCH, EG2, ENGVAL1, FLETBV3M, FLETCBV2, FLETCBV3, FLETCHBV, FLETCHCR, FMINSRF2, FMINSURF, GENHUMPS, INDEF, INDEFM, JIMACK, NCB20, NCB20B, NONCVXU2, NONCVXUN, NONDQUAR, POWELLSG, POWER, QUARTC, SCHMVETT, SCOSINE, SCURLY10, SCURLY20, SCURLY30, SENSORS, SINQUAD, SPARSINE, SPARSQUR, SSCOSINE, TOINTGSS, and VAREIGVL.

In our comparisons we use the following five algorithms to solve the trust-region subproblems with various choices for the approximate Hessian:

Abbreviation	Description
SC-INF	the (P, ∞)-norm subproblem solver with $B_0 = \gamma_k I$
SC-INF-D	the (P, ∞)-norm subproblem solver with a dense initialization
SC-L2	the $(P, 2)$-norm subproblem solver with $B_0 = \gamma_k I$
SC-L2-D	the $(P, 2)$-norm subproblem solver with a dense initialization
trCG	truncated CG [15, Algorithm 7.5.1]

For these experiments, trCG was implemented in MATLAB by the authors. The L-MSS shape-changing subproblem solvers were implemented in an algorithm similar to [4, Algorithm 5]. A feature of this algorithm is that the L-MSS matrix is updated by every pair $\{(s_i, y_i)\}_{i=k-m+1}^{k}$ as long as the s_i are linearly independent (updates are skipped if this condition is not met). In the case of an MSS L2 method, previous numerical results found that $m = 3$ outperformed larger memory choices of $m = 5$ and $m = 7$ [5,19]. Our own experiments for this paper confirmed these results. For this reason, $m = 3$ is used for all experiments with MSS matrices.

Comparisons on the test set are made using extended performance profiles as in [23]. These profiles are an extension of the well-known profiles of Dolan and Moré [16]. We compare total computational time (and function calls) 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

$$\rho_s(\tau) = \text{card}\{p : \pi_{p,s} \leq \tau\}/n_p \quad \text{and} \quad \pi_{p,s} = t_{p,s} \Big/ \min_{1 \leq i \leq S, i \neq s} t_{p,i},$$

where $t_{p,s}$ is the 'output' (i.e. time) 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 extended performance profiles are the same as the classical ones for $\tau \geq 1$. (In the profiles we include a dashed vertical grey line, to indicate $\tau = 1$.) The solvers are compared on 60 large-scale CUTEst problems. We consider (1) to be solved when $\|\nabla f(x_k)\|_\infty < \varepsilon$ with $\varepsilon = 5.0 \times 10^{-4}$. In all the performance profiles, $\rho_s(\tau) < 1$, which indicates that no solver was able to solve all the problems in the test set; however, the value of $\rho_s(\tau)$ at $\tau = 32$ indicates the percentage of problems solved in the test set.

For the single-parameter initialization, we use the representation $B_0 = \gamma_k I$, where

$$\gamma_k = \max_{k-1 \leq i \leq k-q} \left\{ \frac{y_i^T y_i}{y_i^T s_i} \right\}, \tag{31}$$

and q is the number of stored iterates used to compute γ_k. This initialization is based on work in [4] that showed that this initialization and the value of $q = 5$ works well for single-parameter initializations using SC-INF, SC-L2, L2, and trCG in the case of limited-memory Symmetric Rank-1 (L-SR1) Hessian approximations. For the dense initialization,

$$\zeta_k = \max_{k-1 \leq i \leq k-q} \left\{ \frac{y_i^T y_i}{y_i^T s_i} \right\} \quad \text{and} \quad \zeta_k^C = \frac{y_k^T y_k}{y_k^T s_k},$$

with $q = 5$, i.e. ζ_k was chosen to be the single-parameter initialization and ζ_k^C was chosen to be the well-known initialization for quasi-Newton methods [1]. In [19], a variation of this initialization was found to outperform other choices for ζ_k and ζ_k^C in the case of MSS matrices. Note that q is the number of of stored updates to form the initialization parameters and not the 'memory' given by m, i.e. the number of stored updates to form B_k.

4.1. Initialization experiments

In this section, we compare the performance of the dense initialization with the single parameter initialization using the two shape-changing norms for the trust-region subproblems. In this set of experiments, only L-MSSM matrices were used to approximate the Hessian, and a maximum of 5000 iterations were allowed. We begin with the following two experiments:

- Experiment I.A: Comparison between the single parameter initialization (L-MSSM-SC-L2) with the dense initialization (L-MSSM-SC-L2-D) using the shape-changing $(P, 2)$ norm.
- Experiment I.B: Comparison between the single parameter initialization (L-MSSM-SC-INF) with the dense initialization (L-MSSM-SC-INF-D) using the shape-changing (P, ∞) norm.

Figure 1 shows that L-MSSM-SC-L2 outperforms L-MSSM-SC-L2-D in computational time, while Figure 2 shows that L-MSSM-SC-INF-D outperforms L-MSSM-SC-INF in both computational time and function calls. These results lead us to the following third experiment, which compares the better method from the previous two experiments:

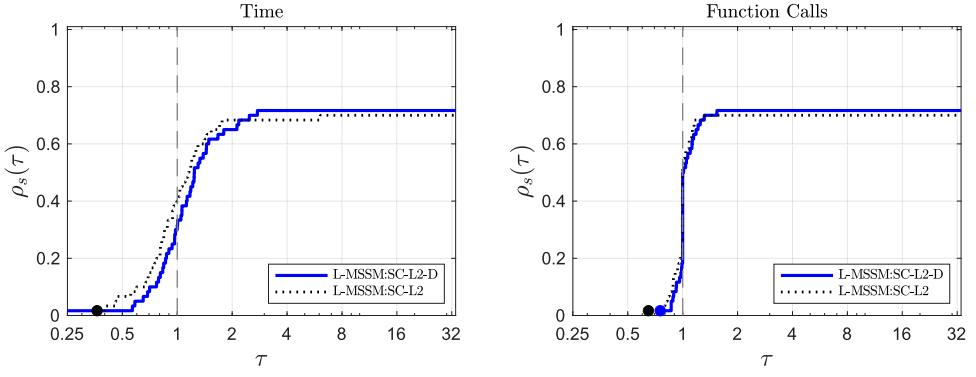

Figure 1. Experiment I.A. Comparison on time and function calls with the shape-changing $(P, 2)$ solver.

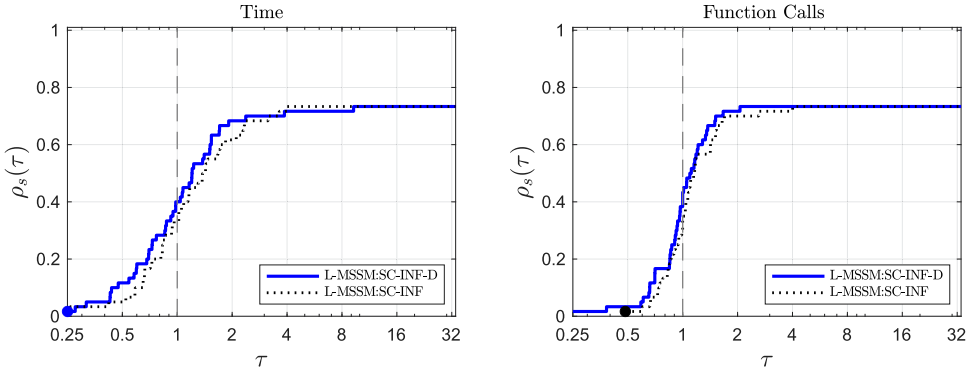

Figure 2. Experiment I.B. Comparison on time and function calls with the shape-changing (P, ∞) solver.

- Experiment I.C: Comparison between the single parameter initialization solver using the shape-changing $(P, 2)$ norm (L-MSSM-SC-L2) with the dense initialization solver using the shape-changing (P, ∞) norm (L-MSSM-SC-INF-D).

Figure 3 reports the results of Experiment I.C, where L-MSSM-SC-INF-D appears to do better on the test set in terms of time and function evaluations. Moreover, this method solves more problems on the test set than L-MSSM-SC-L2.

4.2. Subproblem solvers

In this section, we present Experiment II, which compares the best performing solver from Experiment I (L-MSSM-SC-INF-D) to truncated CG (trCG). To compare these approaches as trust-region subproblem solvers, only L-MSSM approximations of the Hessian are used to approximate the Hessian for both methods. The maximum number of allowed iterations in this experiment was 5000. The results of this experiment are presented in Figure 4. In terms of time, L-MSSM-SC-INF-D outperforms trCG. This may be due to the fact that CG is an iterative method; in contrast, the (P, ∞)-norm solver analytically computes the solution. In terms of function evaluations, at $\tau = 1$, the (P, ∞)-solver outperforms trCG–indicating that on any given problem in the subset, the (P, ∞)-solver

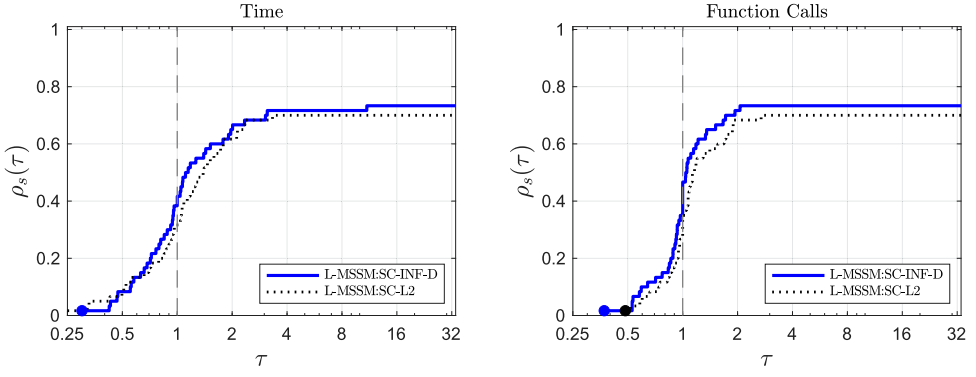

Figure 3. Experiment I.C. Comparison on time and function calls with the the shape-changing (P, ∞) solver with the dense initialization and the $(P, 2)$ solver with the single-parameter initialization.

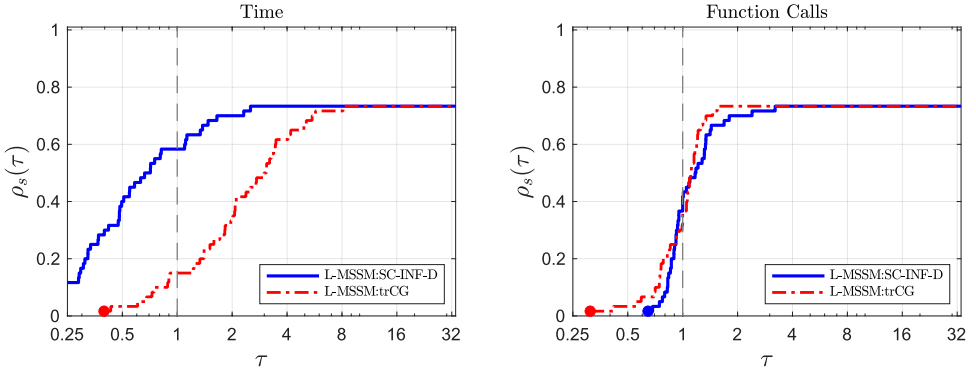

Figure 4. Experiment II. Comparison of (P, ∞) with the dense initialization to trCG subproblem solvers with L-MSSM matrices.

will require fewer function evaluations most of the time. However, over the entire test set, trCG performs slightly better in terms of function evaluations.

4.3. L-SR1 comparison

In this section, we compare the performance of the shape-changing norms using L-SR1 and L-MSSM approximations for the Hessian. For these experiments, the maximum number of iterations is 50, 000. The memory parameters for L-SR1 were chosen based on results in [4], where $m = 5$ (the quasi-Newton method memory parameter) and $q = 7$ (the number of stored iterates used to compute $B_0 = \gamma_k I$) appear to be the best combination. We present four experiments.

- Experiment III.A: Comparison between solvers using L-SR1 matrices with the single parameter initialization with L-MSSM matrices with the single parameter initialization using the shape-changing $(P, 2)$ norm.

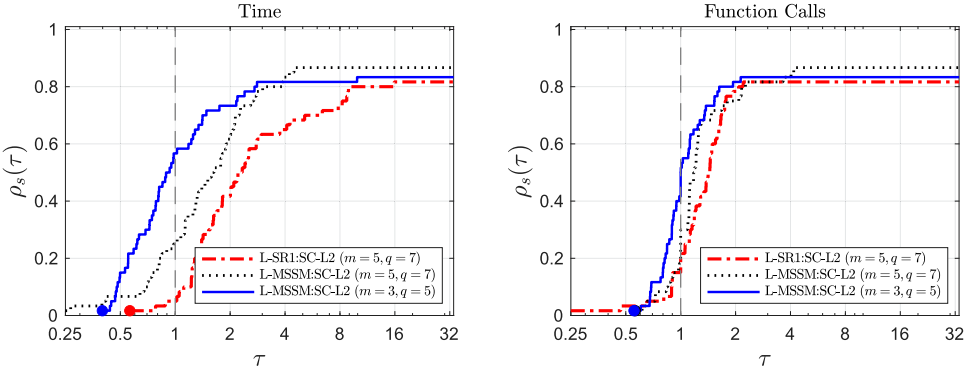

Figure 5. Experiment III.A: Comparison between L-SR1 and L-MSSM matrices using the single-parameter initialization with the shape-changing $(P, 2)$ norm.

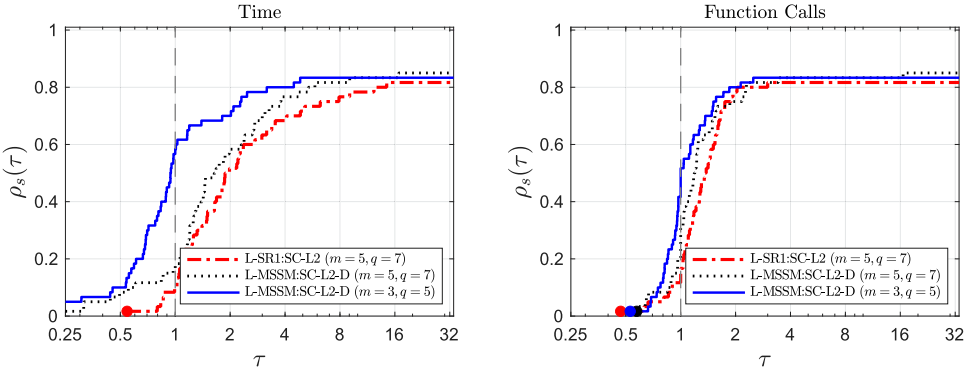

Figure 6. Experiment III.B: Comparison between L-SR1 and L-MSSM matrices using the dense initialization with the shape-changing $(P, 2)$ norm.

- Experiment III.B: Comparison between solvers using L-SR1 matrices with the single parameter initialization with L-MSSM matrices with the dense parameter initialization using the shape-changing $(P, 2)$ norm.
- Experiment III.C: Comparison between solvers using L-SR1 matrices with the single parameter initialization with L-MSSM matrices with the single parameter initialization using the shape-changing (P, ∞) norm.
- Experiment III.D: Comparison between solvers using L-SR1 matrices with the single parameter initialization with L-MSSM matrices with the dense parameter initialization using the shape-changing (P, ∞) norm.

In these experiments, we used the single parameter initialization for L-SR1. For L-MSSM, we used both the optimal memory sizes of $m = 3$ and $q = 5$ as well as the same memory size used for L-SR1 ($m = 5$ and $q = 7$). Figures 5–8 report the results for these four experiments. In all cases, the solvers that use the L-MSSM matrices outperform those that use the L-SR1 matrices, both in computational time and function evaluations. In particular, based on the results of Figure 3, it is not surprising that the performance profile comparing L-SR1 with the dense initialization L-MSSM (Figure 8) is more striking.

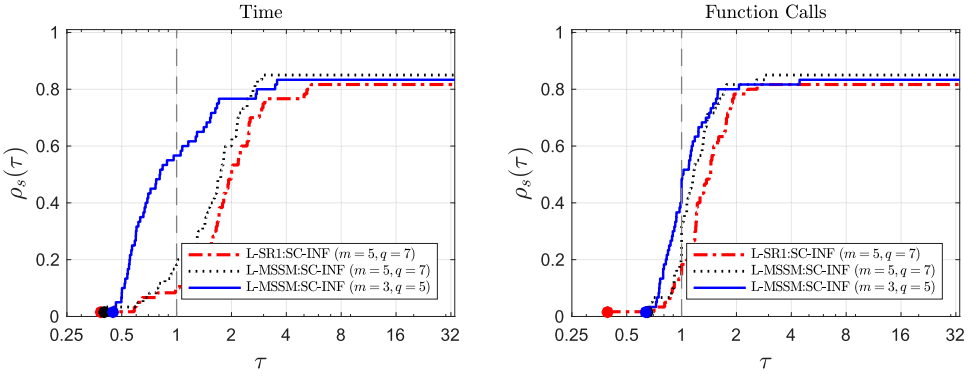

Figure 7. Experiment III.C: Comparison between L-SR1 and L-MSSM matrices using the single-parameter initialization with the shape-changing (P, ∞) norm.

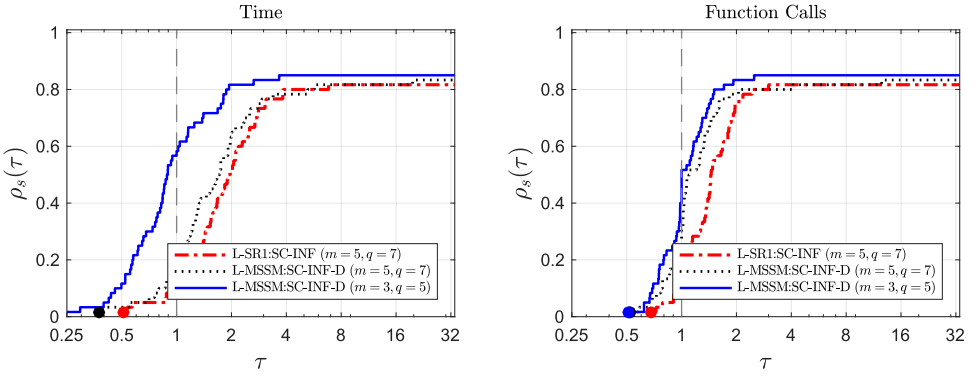

Figure 8. Experiment III.D: Comparison between L-SR1 and L-MSSM matrices using the dense initialization with the shape-changing (P, ∞) norm.

5. Concluding remarks

In this paper, we proposed L-MSS methods that make use of the dense initialization and two shape-changing norms. Numerical results suggest that methods using densely-initialized MSS matrix approximations of the Hessian together with the shape-changing norms outperform other trust-region methods. Based on the results in this paper, we suggest default settings of $m = 3$ and $q = 5$ for both SC-L2 and SC-INF when using either the dense or single-parameter initializations.

Note

1. See https://www.cuter.rl.ac.uk/Problems/mastsif.shtml for further classification information.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This research work was partially funded by the National Science Foundation (NSF) [grant number IIS-1741490].

Notes on contributors

J. Brust was in the Department of Mathematics at the University of California, San Diego. (Now in the School of Mathematical and Statistical Sciences) at Arizona State University. His research is in large-scale optimization, numerical linear algebra and data-science.

J. Erway is a Professor of Mathematics at Wake Forest University. Her interests include numerical optimization and numerical linear algebra.

R. Marcia is a Professor of Applied Mathematics at the University of California, Merced. His research interests include large-scale optimization, signal processing, data science, numerical linear algebra, and mathematical biology.

ORCID

J. J. Brust http://orcid.org/0000-0002-3748-2440
J. B. Erway http://orcid.org/0000-0001-6838-140X

References

[1] J. Barzilai and J.M. Borwein, *Two-point step size gradient methods*, IMA J. Numer. Anal. 8(1) (1988), pp. 141–148.
[2] J.J. Brust, *Large-scale quasi-Newton trust-region methods: High-accuracy solvers, dense initializations, and extensions*, Ph.D. thesis, UC Merced, 2018.
[3] J. Brust, O. Burdakov, J.B. Erway, and R.F. Marcia, *A dense initialization for limited-memory quasi-Newton methods*, Comput. Optim. Appl. 74(1) (2019), pp. 121–142.
[4] J. Brust, O. Burdakov, J. Erway, and R. Marcia, *Algorithm 1030: SC-SR1: MATLAB software for limited-memory SR1 trust-region methods*, ACM Trans. Math. Softw. 48(4) (2022), pp. 1–33.
[5] J. Brust, J.B. Erway, and R.F. Marcia, *On solving L-SR1 trust-region subproblems*, Comput. Optim. Appl. 66(2) (2017), pp. 245–266.
[6] O. Burdakov, L. Gong, S. Zikrin, and Y.-X. Yuan, *On efficiently combining limited-memory and trust-region techniques*, Math. Program. Comput. 9(1) (2017), pp. 101–134.
[7] 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*, Huangzhou, China, 2002, p. 21.
[8] O.P. Burdakov, *Methods of the secant type for systems of equations with symmetric Jacobian matrix*, Numer. Funct. Anal. Optim. 6(2) (1983), pp. 183–195.
[9] O.P. Burdakov, *Stable versions of the secants method for solving systems of equations*, USSR Comput. Math. Math. Phys. 23(5) (1983), pp. 1–10.
[10] O.P. Burdakov, *On superlinear convergence of some stable variants of the secant method*, Z. Angew. Math. Mech. 66(12) (1986), pp. 615–622.
[11] O.P. Burdakov, *Stable symmetric secant methods with restart*, Cybern. Syst. Anal. 27 (1991), pp. 390–396.
[12] O.P. Burdakov, J.M. Martínez, and E.A. Pilotta, *A limited-memory multipoint symmetric secant method for bound constrained optimization*, Ann. Oper. Res. 117(1–4) (2002), pp. 51–70.
[13] R.H. Byrd, J. Nocedal, and R.B. Schnabel, *Representations of quasi-Newton matrices and their use in limited memory methods*, Math. Program. 63(1–3) (1994), pp. 129–156.
[14] C. Cartis, N.I.M. Gould, and P.L. Toint, *Evaluation Complexity of Algorithms for Nonconvex Optimization: Theory, Computation and Perspectives*, SIAM, Philadelphia, PA, 2022.

[15] A.R. Conn, N.I.M. Gould, and P.L. Toint, *Trust Region Methods*, SIAM, Philadelphia, PA, 2000.
[16] E. Dolan and J.J. Moré, *Benchmarking optimization software with performance profiles*, Math. Program. 91 (2002), pp. 201–213.
[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) (2014), pp. 1–12.
[18] J.B. Erway and R.F. Marcia, *On efficiently computing the eigenvalues of limited-memory quasi-Newton matrices*, SIAM J. Matrix Anal. Appl. 36 (2015), pp. 1338–1359.
[19] J.B. Erway and M. Rezapour, *A new multipoint symmetric secant method with a dense initial matrix*, Optim. Methods Softw. 38(4) (2023), pp. 698–722.
[20] D.M. Gay, *Computing optimal locally constrained steps*, SIAM J. Sci. Stat. Comput. 2(2) (1981), pp. 186–197.
[21] P.E. Gill, W. Murray, and M.H. Wright, *Practical Optimization*, SIAM, Philadelphia, PA, 2019.
[22] N.I.M. Gould, D. Orban, and P.L. Toint, *Cutest: A constrained and unconstrained testing environment with safe threads for mathematical optimization*, Comput. Optim. Appl. 60(3) (2015), pp. 545–557.
[23] A. Mahajan, S. Leyffer, and C. Kirches, *Solving mixed-integer nonlinear programs by QP diving*, Tech. Rep. ANL/MCS-P2071-0312, Mathematics and Computer Science Division, Argonne National Laboratory, Lemont, IL, 2012.
[24] J.J. Moré and D.C. Sorensen, *Computing a trust region step*, SIAM J. Sci. Stat. Comput. 4(3) (1983), pp. 553–572.
[25] Y. Nesterov and B. Polyak, *Cubic regularization of Newton method and its global performance*, Math. Program. 108(1) (2006), pp. 177–205.
[26] J. Nocedal and S. Wright, *Numerical Optimization*, Springer Science & Business Media, New York, NY, 2006.
[27] M.J.D. Powell, *A hybrid method for nonlinear equations*, in *Numerical Methods for Nonlinear Algebraic Equations*, P. Rabinowitz, ed., Gordon and Breach, 1970, pp. 87–114.
[28] M.J.D. Powell, *A new algorithm for unconstrained optimization*, in *Nonlinear Programming (Proc. Sympos., Univ. of Wisconsin, Madison, Wis., 1970)*, J.B. Rosen, O.L. Mangasarian, and K. Ritter, editors, Academic Press, 1970, pp. 31–65.
[29] M.J.D. Powell, *Convergence properties of a class of minimization algorithms*, in *Nonlinear Programming 2 (Proc. Sympos. Special Interest Group on Math. Programming, Univ. of Wisconsin, Madison, Wis., 1974)*, O.L. Mangasarian, R.R. Meyer, and S.M Robinson, editors, Academic Press, 1974, pp. 1–27.
[30] R.B. Schnabel, *Quasi-Newton methods using multiple secant equations*, Technical Report, Colorado University at Boulder Department of Computer Science, 1983.
[31] Y.-X. Yuan, *A review of trust region algorithms for optimization*, in *Proceedings of the Fourth International Congress on Industrial and Applied Mathematics*, J.H. Ball and J.C.R. Hunt, eds., Oxford University Press, Oxford, 2000, pp. 271–282.