UNIVERSITY OF CALIFORNIA, MERCED

LARGE-SCALE QUASI-NEWTON TRUST-REGION METHODS: HIGH-ACCURACY SOLVERS, DENSE INITIALIZATIONS, AND EXTENSIONS

A dissertation submitted in partial satisfaction of the requirements for the degree Doctor of Philosophy

in

Applied Mathematics

by

Johannes J. Brust

Committee in charge:

Professor Roummel F. Marcia, Chair
Professor Harish S. Bhat,
Professor Jennifer B. Erway,
Dr. Cosmin G. Petra,
Professor Noemi Petra

2018

The Dissertation of Johannes Joachim Brust is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Harish S. Bhat

Jennifer B. Erway

Cosmin G. Petra

Noemi Petra

Roummel F. Marica, Chair

University of California, Merced

2018

ACKNOWLEDGEMENTS

This dissertation describes optimization methods and matrix factorizations that are the result of multiple years of guided research, alongside the opportunity to collaborate with highly recognized mathematicians. Therefore, I gratefully acknowledge the excellent oversight of my faculty mentor Professor Roummel F. Marcia, and the invaluable inputs of Professor Jennifer B. Erway, Dr. Cosmin G. Petra, Professor Oleg P. Burdakov, and Professor Ya-Xiang Yuan.

I am aware of the influence that a set of mathematicians had on the development of my passion for mathematics. Therefore I want to thank Professor Harish S. Bhat, Dr. Taras Bileski, Professor Dries Vermeulen, and Professor Noemi Petra, too.

I proudly report that the Graduate Division at UC Merced supported the preparation of this dissertation in the form of the Graduate Dean's Dissertation Fellowship.

Johannes J. Brust, Merced, April 2018

Contents

TABLES, FIGURES

Tables

Figures

CHAPTER 1

INTRODUCTION

1.1 MOTIVATION

Optimization algorithms are mathematical methods that are important to solving problems from diverse disciplines, such as machine learning, quantum chemistry, and finance. In particular, methods for large-scale and non-convex optimization are relevant to various real-world problems that aim to minimize cost, error, or risk, or maximize output, profit or probability. Mathematically, the unconstrained minimization problem is represented as

$$\underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimize}}\ f(\mathbf{x}), \tag{1.1}$$

where $f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$ is a nonlinear and possibly non-convex objective function. This dissertation concentrates on efficient large-scale *trust-region quasi-Newton* methods because trust-region methods incorporate a mechanism, which makes them directly applicable to convex and non-convex optimization problems. Furthermore, much influential progress to effectively solve (1.1) is based on sophisticated ideas from numerical linear algebra. For this reason, we also emphasize methods from linear algebra and apply them to concepts in optimization.

1.2 MULTIVARIABLE OPTIMIZATION

Practical methods for the general problem in (1.1) estimate the solution by a sequence of iterates $\{\mathbf{x}_k\}$, which progressively decrease the objective function

$$f(\mathbf{x}_k) \geq f(\mathbf{x}_{k+1}). \tag{1.2}$$

At a *stationary* point the gradient of the objective function is zero, which is why the iterates also need to satisfy $\nabla f(\mathbf{x}_{k+1}) \to \mathbf{0}$. The sequence of iterates is typically defined

by the update formula

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{s}_k,$$

where $\mathbf{s}_k \in \mathbb{R}^n$ is the so-called *search direction* or *step*. There are two conceptually different methods for computing $\mathbf{s}_k$. The first of the two methods is the so-called *line-search* method. This method initially fixes a desirable search-direction, say $\bar{\mathbf{s}}_k$, and then varies the length of this direction by means of a scalar $\alpha > 0$, i.e. this method searches for a minimum along the line $\alpha\bar{\mathbf{s}}_k$. The line-search parameter, α, is typically determined so that the objective function decreases and the step length is not too short. The second of the two methods is the so-called *trust-region* method. This method first fixes the length of a search-direction, say $\bar{\Delta} > 0$, and then computes a desirable vector such that the objective function decreases and sufficient progress is made. The trust-region method is regarded as the computationally more costly of the two methods per iteration, but its search-directions are also regarded to be of higher quality than those of the line-search methods. Common to both methods is a quadratic approximation of the objective function, around the current iterate $\mathbf{x}_k$:

$$f(\mathbf{x}_k + \mathbf{s}) \approx f(\mathbf{x}_k) + \nabla f(\mathbf{x}_k)^T\mathbf{s} + \frac{1}{2}\mathbf{s}^T\mathbf{B}_k\mathbf{s}, \tag{1.3}$$

where $\mathbf{B}_k \in \mathbb{R}^{n\times n}$ is either the Hessian matrix of second derivatives ($\mathbf{B}_k = \nabla^2 f(\mathbf{x}_k)$) or an approximation to it ($\mathbf{B}_k \approx \nabla^2 f(\mathbf{x}_k)$). Both the line-search and the trust-region methods compute steps based on minimizing the quadratic approximation in (1.3), as a means of minimizing the nonlinear objective function $f(\mathbf{x})$.

1.3 TRUST-REGION METHOD

The origins of trust-region methods are based on two seminal papers [Lev44, Mar63] for solving nonlinear least-squares problems (cf. [CGT00]). In the early 1980's the name "trust-region method" was coined through the articles [Sor82, MS83], in which theory and a practical method for small and medium sized problems was developed. Recent advances in unconstrained trust-region methods are in the context of large-scale problems [Yua15]. The search directions in trust-region methods are computed as the solutions to the so-called *trust-region subproblems*

$$\mathbf{s}_k = \underset{\|\mathbf{s}\|\leq\Delta_k}{\arg\min}\, Q(\mathbf{s}) = \mathbf{g}_k^T\mathbf{s} + \frac{1}{2}\mathbf{s}^T\mathbf{B}_k\mathbf{s}, \tag{1.4}$$

where $\mathbf{g}_k = \nabla f(\mathbf{x}_k)$ is the gradient of the objective function, $\Delta_k > 0$ is the trust-region *radius*, and where $\|\cdot\|$ represents a given norm. An interpretation of the expression in (1.4) is that the quadratic approximation, $Q(\mathbf{s}) \approx f(\mathbf{x}_k + \mathbf{s}) - f(\mathbf{x}_k)$, is only accurate within the region specified by a given norm – this is the "trust-region". Trust-region

methods are broadly classified into two groups; if a method nearly exactly computes the solution to the trust-region subproblem, then it is a *high accuracy* method, alternatively, if it approximately solves the trust-region subproblem then it is an *approximate* method [RSS01, BGYZ16]. Approximate methods were originally intended for large optimization problems, and typically do not make additional assumptions on the properties of the Hessian matrix. A prominent approximate method is the one due to Steihaug [Ste83]. However, when additional assumptions on the properties of the Hessian are made, then recently developed methods are able to solve even large-scale trust-regions subproblems with high accuracy. In particular, when limited memory quasi-Newton matrices approximate the true Hessian matrix, then the combination of quasi-Newton matrices with trust-region methods has spanned the development of *large-scale quasi-Newton trust-region methods*. Examples of these methods are the ones by Erway and Marica [EM14], Burke et al. [BWX96], and Burdakov et al. [BGYZ16]. In this dissertation we focus on methods for large-scale problems that compute high accuracy solutions of trust-region subproblems. Unlike the methods in [EM14, BWX96] and [BGYZ16], who target convex trust-region subproblems, we analyze the solution of potentially non-convex subproblems, too. Moreover, we address the question of how to initialize limited-memory quasi-Newton matrices in a trust-region algorithm, and extend an effective unconstrained trust-region method to linear equality constrained problems.

Because of the constraint $\|\mathbf{s}\| \leq \Delta_k$ in (1.4), the trust-region subproblem always has a solution, even in the case when the quadratic approximation is not convex. For example, if the ℓ_2 norm is used in (1.4), then solving a non-convex trust-region subproblem in two dimensions amounts to minimizing the multivariable quadratic function $Q(\mathbf{s})$ within a disk:

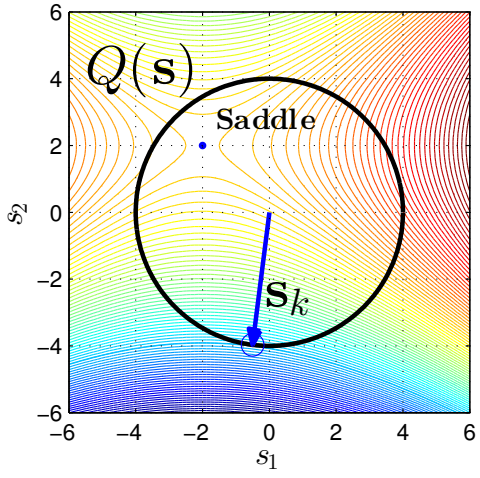


FIGURE 1.1 *Trust-region subproblem in two dimensions. The quadratic approximation $Q(\mathbf{s})$ is not convex, has a saddle point and is unbounded. The trust-region subproblem has a finite solution represented by $\mathbf{s}_k$.*

At the solution $\mathbf{s}_k$ one of two conditions may hold. Either it is within the trust-region, i.e. $\|\mathbf{s}_k\| < \Delta_k$, or the solution is at the boundary of the trust-region, i.e. $\|\mathbf{s}_k\| = \Delta_k$. Therefore a strategy to compute the solution to the trust-region subproblem is:

1. If $Q(\mathbf{s})$ is convex, then compute $\nabla Q(\mathbf{s}^*) = \mathbf{g}_k + \mathbf{B}_k\mathbf{s}^* = \mathbf{0}$. If moreover $\|\mathbf{s}^*\| \leq \Delta_k$ then set $\mathbf{s}_k = \mathbf{s}$.

2. Otherwise find the optimal pair $(\mathbf{s}^*, \sigma^*) \in (\mathbb{R}^n, \mathbb{R})$ such that $(\mathbf{B}_k + \sigma^*\mathbf{I})\mathbf{s}^* = -\mathbf{g}_k$, the boundary condition $\|\mathbf{s}^*\| = \Delta_k$ holds, and $\mathbf{B}_k + \sigma^*\mathbf{I}$ is postive definite.

In order to measure the accuracy of the quadratic approximation typical trust-region methods compute a performance ratio, which relates the actual improvements in the objective function to the predicted improvements of the approximation. The performance ratio is defined as

$$\rho_k = \frac{f(\mathbf{x}_{k+1}) - f(\mathbf{x}_k)}{Q(\mathbf{s}_k)}.$$

If $\rho_k \geq 1$ then $\mathbf{s}_k$ resulted in large desirable actual improvements. The other extreme occurs when $\rho_k \leq 0$, in which case the objective function worsened, as the denominator of ρ_k is always non-positive since $Q(\mathbf{s}_k) \leq Q(\mathbf{0}) \leq 0$. Practical techniques specify a certain threshold $0 < c \leq 1$ such that if $\rho_k > c$, then the step is still regarded as a desirable direction. Otherwise the radius Δ_k is decreased, and a new solution to the subproblem in (1.4) is computed. In a general trust-region algorithm, a lower bound (c_-) and an upper bound (c_+) on c will be provided. Moreover, the positive parameters d_- and d_+ are used to shrink ($\Delta_k \leftarrow d_-\Delta_k$) or enlarge ($\Delta_k \leftarrow d_+\Delta_k$) the trust-region radius. We summarize the trust-region approach in the form of an algorithm.

ALGORITHM 1.1
Initialize: $\mathbf{x}_0, \mathbf{g}_0, \mathbf{B}_0, \Delta_0, 0 < c_- < c < c_+, 0 < d_- < 1 < d_+, 0 < \varepsilon$
For $k = 1, 2, \ldots$

1. If $\|\mathbf{g}_k\|_2 > \varepsilon$ go to 2., otherwise terminate;

2. Compute $\mathbf{s}_k = \underset{\|\mathbf{s}\| \leq \Delta_k}{\arg\min}\, Q(\mathbf{s})$;

3. Compute $\rho_k = (f(\mathbf{x}_k + \mathbf{s}_k) - f(\mathbf{x}_k))/Q(\mathbf{s}_k)$;

4. If $\rho_k > c$ go to 6, otherwise go to 5;

5. Set $\Delta_k = d_-\Delta_k$, go to 2;

6. If $\rho_k > c_+$, set $\Delta_k = d_+\Delta_k$, otherwise set $\Delta_k = \Delta_k$;

7. Set $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{s}_k$, update $\mathbf{g}_{k+1}, \mathbf{B}_{k+1}$ go to 1.;

The computationally most intensive component of Algorithm 1.1 is the solution of the subproblem in Step 2. Chapter 2 analyzes efficient solutions of large-scale trust-region subproblems. In particular, the matrix $\mathbf{B}_k$ will represent so-called *quasi-Newton* matrices, and solving the trust-region subproblem will exploit the structure of the quasi-Newton matrices.

1.4 QUASI-NEWTON MATRICES

Because quasi-Newton matrices form an integral part of this dissertation, we review their basic concepts in this section. The original ideas on quasi-Newton matrices were developed by Davidon [Dav59, Dav90]. In particular, these methods rely on the insight from Davidon that properties of the Hessian matrix can be efficiently approximated using low-rank matrix updates. Specifically, the Hessian matrix can be viewed as a linear mapping from the space of changes in iterates $\mathbf{x}_{k+1} - \mathbf{x}_k$ to the space of changes in gradients $\mathbf{g}_{k+1} - \mathbf{g}_k$. This property may be understood as a multi-dimensional analog of the chain rule

$$\mathbf{d}\nabla f(\mathbf{x}) = \mathbf{d}\begin{bmatrix} \frac{\partial^2 f(\mathbf{x})}{\partial x_1} \\ \vdots \\ \frac{\partial f(\mathbf{x})}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial^2 f(\mathbf{x})}{\partial x_1^2} dx_1 + \cdots + \frac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_n} dx_n \\ \vdots \\ \frac{\partial^2 f(\mathbf{x})}{\partial x_n \partial x_1} dx_1 + \cdots + \frac{\partial^2 f(\mathbf{x})}{\partial x_n^2} dx_n \end{bmatrix} = \nabla^2 f(\mathbf{x})\mathbf{dx}.$$

Approximating the continuous changes by $\mathbf{d}\nabla f(\mathbf{x}) \approx \mathbf{g}_{k+1} - \mathbf{g}_k \equiv \mathbf{y}_k$ and by $\mathbf{dx} \approx \mathbf{x}_{k+1} - \mathbf{x}_k \equiv \mathbf{s}_k$, then desirable estimates of the Hessian matrix, $\mathbf{B}_{k+1} \approx \nabla^2 f(\mathbf{x}_{k+1})$, and its inverse, $\mathbf{H}_{k+1} \approx (\nabla^2 f(\mathbf{x}_{k+1}))^{-1}$, satisfy

$$\mathbf{y}_k = \mathbf{B}_{k+1}\mathbf{s}_k \quad \text{and} \quad \mathbf{H}_{k+1}\mathbf{y}_k = \mathbf{s}_k. \tag{1.5}$$

The conditions in (1.5) are the *secant-conditions*, which characterize the family of quasi-Newton matrices. Since the Hessian is symmetric, all quasi-Newton matrices must be symmetric, too. Moreover, it is desirable that quasi-Newton matrices retain past information and are easily updated. In order to address the latter requirements, quasi-Newton matrices are computed via recursive formulas that use low rank matrix updates. The most common update formulas are those of rank-1 or rank-2 matrices. Thus the most widespread quasi-Newton matrices, which approximate the inverse Hessian, are represented as

$$\mathbf{H}_{k+1} = \mathbf{H}_k + \alpha \mathbf{a}\mathbf{a}^T + \beta \mathbf{b}\mathbf{b}^T, \tag{1.6}$$

where both scalars $\alpha, \beta \in \mathbb{R}$ and both vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$ are determined such that (1.5) holds. Another advantage of representing the quasi-Newton formulas in the form of recursive low-rank updates is that the inverse to $\mathbf{H}_{k+1}$ is analytically computed using the Sherman-Morrison-Woodbury formula. A prominent quasi-Newton matrix is obtained

if the update is a rank-1 matrix. Therefore, with say $\beta = 0$, the secant-condition (1.5) implies

$$\mathbf{H}_{k+1} = \mathbf{H}_k + \frac{1}{\mathbf{y}_k^T(\mathbf{s}_k - \mathbf{H}_k\mathbf{y}_k)} (\mathbf{s}_k - \mathbf{H}_k\mathbf{y}_k) (\mathbf{s}_k - \mathbf{H}_k\mathbf{y}_k)^T . \tag{1.7}$$

This so-called symmetric rank-1 matrix (SR1) is unique, in the sense that it is the only symmetric rank-1 update that also satisfies the secant-conditions [Fle70, Pow70]. The rank-2 matrix credited as the original quasi-Newton matrix [FP63, Dav59, Dav90] is known as the Davidon-Fletcher-Powell (DFP) matrix. The DFP matrix was derived in [Dav59] by setting either $\mathbf{a}$ or $\mathbf{b}$ in (1.6) equal to $\mathbf{s}_k$, and then by determining the remaining unknowns from the secant-condition (1.5). Another well know quasi-Newton formula is the Broyden-Fletcher-Goldfarb-Shanno (BFGS) update. The SR1, DFP and BFGS matrices are all members of Broyden's family of quasi-Newton matrices [Bro67]. A quasi-Newton formula, which is less known is the multipoint symmetric secant (MSS) update [Bur83]. We will devote a chapter to a trust-region method based on the MSS matrix. The quasi-Newton matrices share symmetry, the secant conditions, and recursive low-rank updates. Quasi-Newton matrices differ in the size of the low-rank update, and in their definiteness. For instance, as long as $\mathbf{s}_i^T\mathbf{y}_i > 0$ for $i = 0, 1, \cdots, k$, then the BFGS and DFP updates generate positive definite matrices, when the initial matrix, $\mathbf{H}_0$, is positive-definite, too. For reference, we mention the definiteness and rank of the quasi-Newton matrices from this dissertation:

Name	**Rank**	**Definiteness**
BFGS	2	positive definite
SR-1	1	indefinite
DFP	2	positive definite
MSS	2	indefinite

TABLE 1.1
Summary of properties of quasi-Newton matrices.

From a theoretical point of view, indefinite quasi-Newton matrices are attractive because they can potentially approximate the true hessian matrix more accurately, when it is indefinite [BKS96, CGT91]. However for many practical implementations the BFGS matrix is the method of choice, because it can be easily maintained to stay positive definite, and for its convincing practical performances [LN89]

1.4.1 COMPACT REPRESENTATION

The *compact representation* of quasi-Newton matrices is the representation of the recursively defined update formulas from (1.6), in the form of an initial matrix plus a matrix update. In this sense it can be understood as a particular type of matrix factorization. The compact representation of the BFGS and SR1 matrices was formally developed by Byrd et al. [BNS94]. Originally Broyden [Bro67] developed the concept

of representing the recursive formulas of quasi-Newton matrices in the form of linear systems. The compact representation of the full Broyden class of quasi-Newton matrices was recently established in [DEM16]. The matrix $\mathbf{B}_k = \mathbf{H}_k^{-1}$ defines the trust-region subproblems in (1.4), which is why we will describe the compact representation in terms of $\mathbf{B}_k$ instead of $\mathbf{H}_k$. By describing the compact representation of $\mathbf{B}_k$, we incur no loss of substance because the inverses of quasi-Newton matrices are computed analytically using the Sherman-Morrison-Woodbury formula. The compact representation of $\mathbf{B}_k$ uses the following matrices which stores previously computed pairs $\{\mathbf{s}_i, \mathbf{y}_i\}$ for $0 \leq i \leq k-1$:

$$\mathbf{S}_k = \begin{bmatrix} \mathbf{s}_0 & \mathbf{s}_1 & \cdots & \mathbf{s}_{k-1} \end{bmatrix} \quad \text{and} \quad \mathbf{Y}_k = \begin{bmatrix} \mathbf{y}_0 & \mathbf{y}_1 & \cdots & \mathbf{y}_{k-1} \end{bmatrix}. \tag{1.8}$$

Then the compact representation of $\mathbf{B}_k$ has the following form:

$$\mathbf{B}_k = \mathbf{B}_0 + \mathbf{\Psi}_k \mathbf{M}_k \mathbf{\Psi}_k^T, \tag{1.9}$$

where $\mathbf{B}_0 \in \mathbb{R}^{n \times n}$ is the *initialization*, and the square symmetric matrix $\mathbf{M}_k \in \mathbb{R}^{2k \times 2k}$ is different for each update formula. For all quasi-Newton matrices in this dissertation $\mathbf{\Psi}_k = \begin{bmatrix} \mathbf{B}_0\mathbf{S}_k & \mathbf{Y}_k \end{bmatrix} \in \mathbb{R}^{n \times 2k}$, except for the MSS and SR1 matrices. For the MSS matrices $\mathbf{\Psi}_k = \begin{bmatrix} \mathbf{S}_k & (\mathbf{Y}_k - \mathbf{B}_0\mathbf{S}_k) \end{bmatrix} \in \mathbb{R}^{n \times 2k}$, while for SR1 matrices $\mathbf{\Psi}_k = (\mathbf{Y}_k - \mathbf{B}_0\mathbf{S}_k) \in \mathbb{R}^{n \times k}$. We assume that $\mathbf{\Psi}_k$ is of full column rank. If $k \ll n$, then the matrix product $\mathbf{\Psi}_k \mathbf{M}_k \mathbf{\Psi}_k^T$ resembles a vector outer product

$$\mathbf{\Psi}_k \mathbf{M}_k \mathbf{\Psi}_k^T = \begin{bmatrix} | \\ \mathbf{\Psi}_k \\ | \end{bmatrix} \big[\mathbf{M}_k\big] \big[\text{———} \;\; \mathbf{\Psi}_k^T \;\; \text{———}\big],$$

which enables efficient computations of matrix-vector applies. That is, computing $(\mathbf{\Psi}_k \mathbf{M}_k \mathbf{\Psi}_k^T)\mathbf{s}$ for a vector $\mathbf{s} \in \mathbb{R}^n$ can be done in complexity $\mathcal{O}(4kn)$, instead of $\mathcal{O}(n^2)$. Moreover, by storing only the matrices $\mathbf{\Psi}_k$ and $\mathbf{M}_k$, the compact representation, without the initial matrix, requires only $2kn + (2k)^2$ storage entries, instead of n^2 locations.

1.4.2 LIMITED-MEMORY COMPACT REPRESENTATIONS

Among the first limited-memory methods is one developed by Nocedal in 1980 [Noc80] for the recursion on of the BFGS formula. The main characteristic of limited-memory methods is that only a small subset of the pairs $(\mathbf{s}_i, \mathbf{y}_i), i = 0, 1, \cdots, k-1$ is used to update the quasi-Newton matrices. The most common application of a limited-memory strategy is to store a fixed number l with $l \ll n$ of the most recent pairs, so that the

matrices $\mathbf{S}_k$ and $\mathbf{Y}_k$ are tall and rectangular with

$$\mathbf{S}_k = \begin{bmatrix} \mathbf{s}_{k-l} & \mathbf{s}_{k-l+1} & \cdots & \mathbf{s}_{k-1} \end{bmatrix} \quad \text{and} \quad \mathbf{Y}_k = \begin{bmatrix} \mathbf{y}_{k-l} & \mathbf{y}_{k-l+1} & \cdots & \mathbf{y}_{k-1} \end{bmatrix}.$$

When the newest pair $(\mathbf{s}_k, \mathbf{y}_k)$ has been computed, then the next matrices $\mathbf{S}_{k+1}$ and $\mathbf{Y}_{k+1}$ are obtained from $\mathbf{S}_k$ and $\mathbf{Y}_k$ by shifting their columns to the left by one index and inserting $\mathbf{s}_k$ as the last column. That is, the updated matrix $\mathbf{S}_{k+1}$ becomes $\mathbf{S}_{k+1} = \begin{bmatrix} \mathbf{s}_{k-l+1} & \mathbf{s}_{k-l+2} & \cdots & \mathbf{s}_k \end{bmatrix}$, while $\mathbf{Y}_{k+1} = \begin{bmatrix} \mathbf{y}_{k-l+1} & \mathbf{y}_{k-l+2} & \cdots & \mathbf{y}_k \end{bmatrix}$. In limited-memory methods the initial matrix $\mathbf{B}_0$ is chosen to simplify the computations, too. Typically, $\mathbf{B}_0$ is taken to be a multiple of the identity matrix in the form of $\mathbf{B}_0^{(k-1)} = \mathbf{B}_0 = \gamma_{k-1}\mathbf{I}_n$, where $\gamma_{k-1} = \frac{\|\mathbf{y}_{k-1}\|^2}{\mathbf{y}_{k-1}^T \mathbf{s}_{k-1}}$. Chapter 3 of this dissertation proposes a large-scale quasi-Newton trust-region method when the initial matrix is not chosen as a multiple of the identity matrix. As is standard notation, the name of a quasi-Newton formula prepended with an captial 'L-', symbolizes the limited-memory version of that particular quasi-Newton matrix. For example, L-BFGS represents the limited memory version of the BFGS matrix. In particular, for L-BFGS, the matrix $\mathbf{\Psi}_k$ is of dimension $\mathbb{R}^{n\times 2l}$ instead of $\mathbb{R}^{n\times 2k}$. Similarly, the matrix $\mathbf{M}_k$ is of dimension $(2l \times 2l)$, instead of $(2k \times 2k)$. Since our goal is to develop methods for large-scale optimization problems all quasi-Newton matrices in this dissertation are assumed to be limited-memory matrices.

1.4.3 PARTIAL EIGENDECOMPOSITION

The structure of the compact representation from (1.9) enables an efficient factorization of the matrix into a partial spectral decomposition. Here, as in [BGYZ16, EM15], an implicit QR factorization of the rectangular matrix $\mathbf{\Psi}_k$ is used in the process. An alternative approach, which was proposed in [Lu96], is to use an implicit SVD factorization of $\mathbf{\Psi}_k$, instead. We take $\mathbf{\Psi}_k$ to be of dimensions $n \times 2l$. Now, consider the problem of computing the eigenvalues of $\mathbf{B}_k$:

$$\mathbf{B}_k = \gamma_{k-1}\mathbf{I}_n + \mathbf{\Psi}_k \mathbf{M}_k \mathbf{\Psi}_k^T,$$

where $\mathbf{B}_0 = \gamma_{k-1}\mathbf{I}_n$. The "thin" QR factorization of $\mathbf{\Psi}_k$ can be written as $\mathbf{\Psi}_k = \mathbf{QR}$ where $\mathbf{Q} \in \mathbb{R}^{n\times 2l}$ and $\mathbf{R} \in \mathbb{R}^{2l\times 2l}$ is invertible because, by assumption, $\mathbf{\Psi}_k$ has full column rank. Then,

$$\mathbf{B}_k = \gamma_{k-1}\mathbf{I}_n + \mathbf{QRM}_k\mathbf{R}^T\mathbf{Q}^T.$$

The matrix $\mathbf{RMR}^T \in \mathbb{R}^{2l\times 2l}$ is of a relatively small size, and thus, it is computationally inexpensive to form its spectral decomposition. We define the spectral decomposition of $\mathbf{RM}_k\mathbf{R}^T$ as $\mathbf{U}\hat{\mathbf{\Lambda}}\mathbf{U}^T$, where $\mathbf{U} \in \mathbb{R}^{2l\times 2l}$ is an orthogonal matrix whose columns are made up of eigenvectors of $\mathbf{RM}_k\mathbf{R}^T$ and $\hat{\mathbf{\Lambda}} = \text{diag}(\hat{\lambda}_1, \ldots, \hat{\lambda}_{2l})$ is a diagonal matrix whose entries are the associated eigenvalues.

Thus,

$$\mathbf{B}_k = \gamma_{k-1}\mathbf{I}_n + \mathbf{Q}\mathbf{U}\hat{\mathbf{\Lambda}}\mathbf{U}^T\mathbf{Q}^T.$$

Since both $\mathbf{Q}$ and $\mathbf{U}$ have orthonormal columns, $\mathbf{P}_{\parallel} \triangleq \mathbf{Q}\mathbf{U} \in \mathbb{R}^{n\times 2l}$ also has orthonormal columns. Let $\mathbf{P}_{\perp}$ denote the matrix whose columns form an orthonormal basis for $\left(\mathbf{P}_{\parallel}\right)^{\perp}$. Thus, the spectral decomposition of $\mathbf{B}_k$ is defined as $\mathbf{B}_k = \mathbf{P}\mathbf{\Lambda}\mathbf{P}^T$, where

$$\mathbf{P} \equiv \begin{bmatrix} \mathbf{P}_{\parallel} & \mathbf{P}_{\perp} \end{bmatrix} \quad \text{and} \quad \mathbf{\Lambda} \equiv \begin{bmatrix} \mathbf{\Lambda}_1 & 0 \\ 0 & \mathbf{\Lambda}_2 \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{\Lambda}} + \gamma_{k-1}\mathbf{I}_{2l} & 0 \\ 0 & \gamma_{k-1}\mathbf{I}_{n-2l} \end{bmatrix}, \tag{1.10}$$

where $\mathbf{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_n) = \text{diag}(\hat{\lambda}_1 + \gamma_{k-1}, \ldots, \hat{\lambda}_{2l} + \gamma_{k-1}, \gamma_{k-1}, \ldots, \gamma_{k-1})$, $\mathbf{\Lambda}_1 \in \mathbb{R}^{2l\times 2l}$, and $\Lambda_2 \in \mathbb{R}^{(n-2l)\times(n-2l)}$. The remaining eigenvalues, found on the diagonal of $\mathbf{\Lambda}_2$, are equal to $\lambda_{2l+1} = \gamma_{k-1}$. (For further details, see [BGYZ16, EM15].) In this dissertation, we assume the first $2l$ eigenvalues in $\mathbf{\Lambda}$ are ordered in increasing values, i.e., $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_{2l}$. Finally, throughout this dissertation we denote the leftmost eigenvalue of $\mathbf{B}_k$ by $\lambda_{\min}$, which is computed as $\lambda_{\min} = \min\{\lambda_1, \gamma_{k-1}\}$.

We emphasize three important properties of the eigendecomposition. First, all eigenvalues of $\mathbf{B}_k$ are explicitly obtained and represented by $\mathbf{\Lambda}$. Second, only the first $2l$ eigenvectors of $\mathbf{B}_k$ are explicitly computed; they are represented by $\mathbf{P}_{\parallel}$. In particular, since $\mathbf{\Psi}_k = \mathbf{Q}\mathbf{R}$, then

$$\mathbf{P}_{\parallel} = \mathbf{Q}\mathbf{U} = \mathbf{\Psi}_k\mathbf{R}^{-1}\mathbf{U}. \tag{1.11}$$

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 $\mathbf{\Psi}_k$, $\mathbf{R}$, and $\mathbf{U}$. Third, the eigenvalues given by the parameter γ_{k-1} can be interpreted as an estimate of the curvature of f in the space spanned by the columns of $\mathbf{P}_{\perp}$.

1.5 EXISTING QUASI-NEWTON TRUST-REGION METHODS

When $\mathbf{B}_k$ is a quasi-Newton matrix in the trust-region subproblem (1.4), then the corresponding trust-region method is called a *quasi-Newton trust-region* method. Methods of this type are proposed in [Ger04, BKS96, Pow70], among others. The approaches in Powell [Pow70] can be considered as virtually the first quasi-Newton trust-region methods, while the name of these methods appears to be associated with Gertz [Ger04]. In Byrd et al. [BKS96] an analysis of a SR1 trust-region method is proposed, which is based on the argument that indefinite SR1 matrices are well suited for trust-region methods. Their argument is based on the fact that the trust-region subproblem can produce desirable steps even when the quadratic approximation $Q(\mathbf{s})$ in (1.4) is not convex (cf. Section 1.3). The previous references all use the recursive formulas of quasi-

Newton matrices, instead of limited-memory compact representations. In the context of large-scale optimization, however, limited-memory methods are the *de facto* standard. Large-scale limited-memory quasi-Newton trust-region subproblem solvers were proposed in [EM14, BWX96, BGYZ16], for the case of L-BFGS matrices. In fact, two other PhD dissertations include limited-memory quasi-Newton trust-region methods. In [Lu96] an ℓ_2-norm trust-region method based on the L-SR1 matrix is developed. In [Zik14] a so-called shape-changing L-BFGS trust-region method is described. The contributions of this dissertation are laid out in the next section, however significant differences to previous research are that we combine L-SR1 matrices with norms other than the ℓ_2-norm, focus on novel initial matrices in a L-BFGS trust-region method, develop a method based on the less known limited-memory multipoint symmetric secant (L-MSS) matrix, and propose a limited-memory quasi-Newton trust-region method for equality constrained problems. A significant component of this dissertation is also devoted to the corresponding ideas from numerical linear algebra. Therefore factorizations of oblique projection matrices, compact representations of quasi-Newton matrices with linear equality constraints, or compact representations with dense initial matrices, are also developed.

1.6 CONTRIBUTIONS OF THE DISSERTATION

The contributions of this dissertation are in the context of limited-memory quasi-Newton trust-region methods. We start with a general overview of our contributions, and specify more details in a later enumeration.

In Chapter 2, we propose two methods for solving quasi-Newton trust-region subproblems using L-SR1 updates, where the Hessian approximations are potentially indefinite. This is in contrast to existing methods for L-BFGS updates, where the Hessian approximations are guaranteed to be positive definite. The first of these methods is based on the published paper, "On solving L-SR1 trust-region subproblems," J. J. Brust, J. B. Erway, and R. F. Marcia, *Computational Optimization and Applications*, 66:2, pp. 245-266, 2017. The second method is based on the submitted paper (currently under first revision), "Shape-changing L-SR1 trust-region methods," J. J. Brust, O. P. Burdakov, J. B. Erway, R. F. Marcia, and Y.-x. Yuan.

Next, in Chapter 3, we propose a limited-memory quasi-Newton trust-region method for unconstrained minimization, which uses a dense initial matrix instead of a standard multiple of the identity intitial matrix. This method is described in the manuscript, "Dense Initializations for Limited-Memory Quasi-Newton Methods" J. J. Brust, O. P. Burdakov, J. B. Erway, and R. F. Marcia, which was submitted to the *SIAM Journal on Optimization*.

In Chapter 4 we apply the compact representation of the multipoint symmetric

secant quasi-Newton matrix in an trust-region method for unconstrained minimization. In particular, we develop an novel approach for solving limited memory multipoint symmetric secant (L-MSS) trust-region subproblems.

In Chapter 5 we develop a large-scale trust-region method for linear equality constrained minimization problems. This method extends non-standard trust-region norms from unconstrained minimization to constrained problems. A manuscript of this method is currently in preparation.

Finally, in Chapter 6, we propose matrix factorizations of oblique projection matrices and a practical method to compute their singular values. Oblique projection matrices often arise in constrained optimization methods.

We now enumerate our contributions in more detail.

1. THE ORTHONORMAL BASIS SR1 (OBS) METHOD: This is a ℓ_2-norm trust-region subproblem solver, which connects two strategies to compute exact subproblem solutions. The proposed method combines an approach from [BWX96], based on the Sherman-Morrison-Woodbury formula, with an approach in [BGYZ16] that uses an orthonormal basis (OB), in order to compute L-SR1 subproblem solutions. Because L-SR1 matrices are not guaranteed to be positve definite, we analyze the subproblem solutions on a case-by-case basis, depending on the definiteness of the quasi-Newton matrix. In particular, we propose a formula to compute the trust-region subproblem solution, when an eigendecomposition of the L-SR1 matrix is used, and the so-called "hard case" [MS83] occurs. The *hard case* can occur under two conditions: (1) the quasi-Newton matrix is indefinite, as is true for L-SR1 matrices, and (2) the gradient ($\mathbf{g}_k$) is orthogonal to the eigenvectors of the quasi-Newton matrix, which correspond to the smallest eigenvalue. When the trust-region subproblem solution lies at the boundary, then trust-region methods use a one dimensional root finding procedure to specify the solution. We propose an improved initial value for Newton's one dimensional root finding method, which uses the eigenvalues of the quasi-Newton matrix and does not require safeguarding, which is common in many trust-region methods (see e.g., [MS83]).

2. THE SHAPE-CHANGING SR1 (SC-SR1) METHOD: Instead of the ℓ_2-norm, this method is developed for trust-region subproblems defined by shape-changing norms in combination with L-SR1 matrices. Because the shape-changing norms were originally developed in the context of positive definite L-BFGS matrices, we analyze how to compute subproblem solutions with these norms when indefinite L-SR1 matrices are used. In particular, we characterize the global trust-region subproblem solution with one of the shape-changing norms in the form of a general optimality condition.

3. THE DENSE $\mathbf{B}_0$ METHOD: This is a large-scale L-BFGS trust-region method for

unconstrained minimization. It uses dense initial matrices, instead of multiple of identity matrices, to initialize the L-BFGS approximations of the algorithm. The dense initial matrices compute two curvature estimates of the true Hessian matrix, in order to update the limited-memory quasi-Newton matrices. This is unlike the most common practice of using only one curvature estimate [BNS94]. In particular, we develop various alternatives for the two curvature estimates of the dense initial matrices. Moreover, we propose a general formula of the compact representation of quasi-Newton matrices, which make use of the dense initializations. In other words, we propose the compact representation of limited-memory quasi-Newton matrices that use two curvature estimates of the true Hessian matrix, instead of one.

4. THE MULTIPOINT SYMMETRIC SECANT (MSSM) METHOD: This method uses the indefinite limited-memory multipoint symmetric secant matrix in a trust-region method for unconstrained minimization. Because L-MSS matrices are not necessarily positive definite, they may better approximate indefinite Hessian matrices. Since these matrices have a compact representation, they are readily applicable for large-scale problems. We propose two approaches for a L-MSS trust-region method: One approach uses an orthonormal basis to compute a partial eigendecomposition of the L-MSS matrices, and then computes trust-region subproblem solutions based on the eigendecomposition. The second approach makes use of a set of properties of MSS matrices, and proposes a closed-form solution of ℓ_2-norm L-MSSM subproblems.

5. THE EQUALITY CONSTRAINED TRUST-REGION METHOD: This is a large-scale quasi-Newton trust-region method for linear equality constrained minimization. It combines shape-changing norms with equality constrained trust-region subproblems. In unconstrained optimization, shape-changing norms are used to obtain an analytic solution of quasi-Newton trust-region subproblems. In order to also compute analytic solutions when linear equality constraints are imposed, we develop two novel factorizations, which are based on the compact representation of quasi-Netwton matrices. First we develop the compact representation of the (1,1) block of the Karush-Kuhn-Tucker KKT optimality matrix. Secondly, we propose the partial eigendecomposition of the latter (1,1) block. Combining our matrix factorizations with the shape-changing norms, yields a method for computing analytic trust-region subproblem solutions even when linear equality constraints are present. We also develop a method when the ℓ_2-norm is applied, and compare the shape-changing norm method with the ℓ_2-norm method.

6. OBLIQUE PROJECTION MATRICES: We analyze and develop factorizations of oblique projection matrices. These matrices arise in applications such as con-

strained optimization and least-squares problems. Since the condition number of a matrix can be used as an indicator of the reliability of computations with that particular matrix, computing condition numbers can be of practical interest for various applications. Unlike orthogonal projection matrices, the singular values of oblique projection matrices are not trivially inferred. We propose the eigendecomposition of oblique projection matrices and develop methods to compute singular values and condition numbers of oblique projections.

The work of this dissertation was done in collaboration with several leading researchers in the field of trust-region methods and optimization. Prof. Jennifer Erway from Wake Forest University developed the Interior-Point Sequential Subspace Minimization (IP-SSM) trust-region method [EG10]. Prof. Oleg Burdakov from Linköping University in Sweden and Prof. Ya-Xiang Yuan from the Chinese Academy of Science developed the shape-changing norm [BGYZ16] that we use in Chapters 2-4. Dr. Cosmin Petra is a Computational Mathematician at Lawrence Livermore National Laboratory (LLNL), and the Chapters 5 and 6 are based on research done during my summer internship at LLNL under his supervision in 2017.

1.6.1 CLASSIFYING THE PROPOSED METHODS

As a point of reference, the optimization methods of this dissertation are classified according to different problem formulations. Underlying all proposed methods are the assumptions that the objective function $f(\mathbf{x})$ is at least twice continuously differentiable, but the Hessian matrix is not explicitly available. The quasi-Newton matrices used to approximate the Hessian are all represented as a compact matrix factorization, which means that the proposed methods are for large-scale problems. If the Hessian matrix is available, then methods that make use of its information may be more desirable. For notation we define the symbols

$$\texttt{L} \equiv \text{Large-scale (large } n\text{)}, \quad \texttt{NCX} \equiv \text{Non-convex}, \quad \texttt{l2} \equiv \ell_2\text{-norm},$$

and $\texttt{SC} \equiv$ Shape-changing norms (cf. Chapter 2, eq. (2.18)).

The proposed solvers for trust-region subproblems in (1.4) are summarized in Table 1.2. These solvers can be viewed as nonlinear programming methods, when the objective is a multivariable quadratic function defined by quasi-Newton matrices.

Alternatives for the proposed OBS method are the L-BFGS trust-region subproblem solvers from [BWX96, EM14]. The alternative for the proposed SC-SR1 is the shape-changing L-BFGS trust-region subproblem solver from [BGYZ16]. The proposed MSSM method is similar to the OBS and SC-SR1 methods, with the main difference of using the multipoint symmetric secant quasi-Newton matrix, instead of the L-SR1 matrix. All proposed subproblem solvers are advantageous in situations when the true Hessian is

METHOD	QUASI-NEWTON	CONSTRAINTS	STRENGTHS
OBS	L-SR1	`12`	`L,NCX`
SC-SR1	L-SR1	`SC`	`L,NCX`
MSSM	L-MSS	`12`, `SC`	`L,NCX`

TABLE 1.2

Classification of proposed trust-region subproblem solvers. Here the label `NCX` means that the method is well suited for non-convex subproblems.

indefinite, since the L-SR1 and L-MSS matrices are indefinite.

Table 1.3 summarizes the proposed minimization methods for general nonlinear and potentially non-convex objective functions.

METHOD	QUASI-NEWTON	CONSTRAINTS	STRENGTHS
Dense	Any	–	`L`
Constrained	Any	$\underset{m\times n}{\mathbf{A}}\ \mathbf{x} = \underset{m\times 1}{\mathbf{b}}$	`L`, $m \ll n$

TABLE 1.3

Classification of proposed minimization methods. Here Dense is the dense initialization method proposed in Chapter 3, and Constrained represents the method developed in Chapter 5.

The prominent alternative to the dense initialization method is the line-search L-BFGS approach [ZBN97]. An alternative approach for the constrained method is the large-scale L-BFGS trust-region algorithm for general equality constraints in [LNP98]. In numerical comparisons with a benchmark line-search approach, the *Dense* method does perform particularly well (cf. Figure 3.5). Moreover, numerical experiments indicate that *Dense* method does well on difficult problems (cf. Figure 3.9), whereas for easier problems a hybrid trust-region line-search method as in [BGYZ16] may be advantageous. The *Constrained* method is for general minimization with linear equality constraints. One of its main advantages are fast computations of iterates, because it uses an analytic formula for the solutions of trust-region subproblems. The method assumes that $m \ll n$, and full rank equality constraints.

CHAPTER 2

THE TRUST-REGION SUBPROBLEM SOLVERS

This chapter is based on two manuscripts. The first of these is the published paper, "On solving L-SR1 trust-region subproblems," J. J. Brust, J. B. Erway, and R. F. Marcia, Computational Optimization and Applications, 66:2, pp. 245-266, 2017. The second is the paper submitted to Transactions on Mathematical Software (currently under first revision), "Shape-changing L-SR1 trust-region methods," J. J. Brust, O. P. Burdakov, J. B. Erway, R. F. Marcia, and Y.-x. Yuan.

2.1 SUBPROBLEM SOLVERS

A computationally demanding component in trust-region methods is the solution of the subproblems at each iteration (Step 2 in Algorithm 1). Therefore this chapter proposes two efficient methods to accurately solve large-scale trust-region subproblems. Specifically, we focus on highly accurate solutions of

$$\underset{\|\mathbf{s}\|\leq\Delta_k}{\text{minimize}}\ Q(\mathbf{s}) = \mathbf{g}_k^T\mathbf{s} + \frac{1}{2}\mathbf{s}^T\mathbf{B}_k\mathbf{s}, \tag{2.1}$$

where $\mathbf{B}_k$ is a limited-memory compact quasi-Newton matrix. Our analysis uses the limited-memory symmetric rank-1 (L-SR1) matrix because it is a potentially indefinite quasi-Newton matrix. In other words, the subproblem's objective function, $Q(\mathbf{s})$, is not necessarily convex.

High-accuracy L-SR1 subproblem solvers are of interest in large-scale optimization for two reasons: (1) In previous works, it has been shown that more accurate subproblem solvers can require fewer overall trust-region iterations, and thus, fewer overall function and gradient evaluations [EG10, EGG09, EM14]; and (2) it has been shown that under certain conditions SR1 matrices converge to the true Hessian–a property that has not been proven for other quasi-Newton updates [CGT91]. While these convergence results

have been proven for SR1 matrices, we are not aware of similar results for L-SR1 matrices.

Solving large trust-region subproblems defined by indefinite matrices are especially challenging, with optimal solutions lying on the boundary of the trust-region. Since L-SR1 matrices are not guaranteed to be positive definite, additional care must be taken to handle indefiniteness and the so-called *hard case* (see, e.g., [CGT00, MS83]). To our knowledge, there are only three solvers designed to solve the quasi-Newton subproblems to high accuracy for large-scale optimization. Specifically, the MSS method [EM14] is an adaptation of the Moré-Sorensen method [MS83] to the limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) quasi-Newton setting. Burke et al. [BWX96] proposed a method based on the Sherman-Morrison-Woodbury formula, and more recently, in [BGYZ16], Burdakov et al. solve a trust-region subproblem where the trust region is defined using *shape-changing* norms. All of these methods are based on the positive definite L-BFGS quasi-Newton matrix. In contrast, the methods in this chapter are developed for indefinite quasi-Newton matrices by handling three additional nontrivial cases: (1) the singular case, (2) the so-called *hard case*, and (3) the general indefinite case. We know of no high-accuracy solvers designed specifically for L-SR1 trust-region subproblems for large-scale optimization of the form (2.4) that are able to handle these cases associated with SR1 matrices. It should be noted that large-scale solvers exist for the general trust-region subproblem that are not designed to exploit any specific structure of $\mathbf{B}_k$. Examples of these include the Large-Scale Trust-Region Subproblem (LSTRS) algorithm [RSS01, RSS08] and the Sequential Subspace Method SSM [Hag01, HP04].

Because the methods, which we propose in this chapter are based on an implicit eigendecomposition of the L-SR1 matrix we first describe its compact representation.

2.2 L-SR1 MATRICES

The symmetric rank-1 quasi-Newton matrix has been proposed, among others, by Fletcher [Fle70] and Powell [Pow70]. Specifically, starting from an initial matrix $\mathbf{B}_0$, the recursive SR1 formula is given by

$$\mathbf{B}_{k+1} \triangleq \mathbf{B}_k + \frac{(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)^T}{(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)^T\mathbf{s}_k}, \tag{2.2}$$

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 chapter 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 l most-recently computed pairs $\{(\mathbf{s}_i, \mathbf{y}_i)\}$, where $l \ll n$ (for example, Byrd et al. [BNS94] suggest $l \in [3, 7]$). For simplicity of notation, we assume that the current iteration number k is less than the number of allowed stored limited-memory pairs l.

The SR1 update is a member of the Broyden class of updates (see, e.g., [NW06]). Unlike widely-used updates such as the BFGS and the DFP updates, the SR1 formula 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, [CGT91] give conditions under which the SR1 update formula generates a sequence of matrices that converge to the true Hessian.

2.3 THE L-SR1 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 describe the compact formulation of SR1 matrices. To begin, recall the matrices:

$$\mathbf{S}_k = \begin{bmatrix} \mathbf{s}_{k-l} & \mathbf{s}_{k-l+1} & \cdots & \mathbf{s}_{k-1} \end{bmatrix} \quad \text{and} \quad \mathbf{Y}_k = \begin{bmatrix} \mathbf{y}_{k-l} & \mathbf{y}_{k-l+1} & \cdots & \mathbf{y}_{k-1} \end{bmatrix}.$$

With these, the matrix $\mathbf{S}_k^T\mathbf{Y}_k \in \mathbb{R}^{l\times l}$ can be written as the sum of the following three matrices:

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

where $\mathbf{L}_k$ is strictly lower triangular, $\mathbf{E}_k$ is diagonal, and $\mathbf{T}_k$ is strictly upper triangular. Then, $\mathbf{B}_k$ can be written as

$$\mathbf{B}_k \;=\; \gamma_{k-1}\mathbf{I} + \mathbf{\Psi}_k\mathbf{M}_k\mathbf{\Psi}_k^T, \tag{2.3}$$

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

$$\mathbf{\Psi}_k \;=\; \mathbf{Y}_k - \gamma_{k-1}\mathbf{S}_k \quad \text{and} \quad \mathbf{M}_k \;=\; (\mathbf{E}_k + \mathbf{L}_k + \mathbf{L}_k^T - \gamma_{k-1}\mathbf{S}_k^T\mathbf{S}_k)^{-1}.$$

This compact representation is due to Byrd et al. [BNS94, Theorem 5.1]. For the duration of this chapter, 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 [BNS94, 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, then modifications proposed in [BGYZ16] can be used instead. Notice that the computation of $\mathbf{M}_k$ is relatively inexpensive, since it is a very small square matrix. Importantly, since the SR1 matrix is indefinite the scalar $\infty < \gamma_{k-1} < \infty$ may take any sign.

2.4 THE OBS METHOD

2.4.1 MOTIVATION

We will now describe a method for minimizing the trust-region subproblem defined by a limited-memory symmetric rank-one (L-SR1) matrix subject to a two-norm constraint, i.e.,

$$\underset{\mathbf{s}\in\mathbb{R}^n}{\text{minimize}} \quad Q(\mathbf{s}) = \mathbf{g}_k^T\mathbf{s} + \frac{1}{2}\mathbf{s}^T\mathbf{B}_k\mathbf{s} \quad \text{subject to} \quad \|\mathbf{s}\|_2 \leq \Delta_k. \tag{2.4}$$

Methods that solve the trust-region subproblem to high accuracy are often based on the optimality conditions for a global solution to the trust-region subproblem (see, e.g., Gay [Gay81], Moré and Sorensen [MS83] or Conn, Gould and Toint [CGT00]), given in the following theorem:

Theorem 2.1. *Let Δ_k be a positive constant. A vector $\mathbf{s}^*$ is a global solution of the trust-region subproblem (2.4) if and only if $\|\mathbf{s}^*\|_2 \leq \Delta_k$ and there exists a unique $\sigma^* \geq 0$ such that $\mathbf{B}_k + \sigma^*\mathbf{I}_n$ is positive semidefinite and*

$$(\mathbf{B}_k + \sigma^*\mathbf{I}_n)\mathbf{s}^* = -\mathbf{g}_k \quad \textit{and} \quad \sigma^*(\Delta_k - \|\mathbf{s}^*\|_2) = 0. \tag{2.5}$$

Moreover, if $\mathbf{B}_k + \sigma^*\mathbf{I}_n$ is positive definite, then the global minimizer is unique.

The result in Theorem 2.1 is based the Lagrangian objective function defined as $\mathcal{L}(\mathbf{s},\sigma) \equiv Q(\mathbf{s}) + \frac{\sigma}{2}\|\mathbf{s}\|_2^2$ for a Lagrange multiplier $\sigma \geq 0$. Computing the stationary point of the Lagrangian $\nabla\mathcal{L}(\mathbf{s}^*,\sigma^*) = \mathbf{0}$ implies the equation $(\mathbf{B}_k+\sigma^*\mathbf{I}_n)\mathbf{s}^* = -\mathbf{g}_k$. The second equality in (2.5) is a complementarity condition, which states that at a solution either $\sigma^* = 0$, or $\mathbf{s}^*$ lies on the boundary, i.e., $\|\mathbf{s}^*\|_2 = \Delta_k$.

A well known method, which seeks a solution pair of the form $(\mathbf{s}^*,\sigma^*)$ that satisfies both equations in (2.5) is the Moré-Sorensen method [MS83]. The method alternates between updating $\mathbf{s}^*$ and σ^*; specifically, the method fixes σ^*, solving for $\mathbf{s}^*$ using the first equation and then fixes $\mathbf{s}^*$, solving for σ^* using the second equation. In order to solve for $\mathbf{s}^*$ in the first equation, the Moré-Sorensen method uses the Cholesky factorization of $\mathbf{B}_k+\sigma\mathbf{I}_n$; for this reason, this method is prohibitively expensive for general large-scale optimization when $\mathbf{B}_k$ does not have a structure that can be exploited. However, the Moré-Sorensen method is arguably the best *direct* method for solving the trust-region subproblem. While the Moré-Sorensen direct method uses a safeguarded Newton method to find σ^*, the method proposed in this section makes use of Newton method's together with a judicious initial guess so that safeguarding is not needed to obtain σ^*. Moreover, unlike the Moré-Sorensen method, the proposed method computes $\mathbf{s}^*$ by formula, and in this sense, is an *iteration-free* method.

2.4.2 PROPOSED METHOD

The method proposed in this section, called the "Orthonormal Basis L-SR1" (OBS) method, is able to solve the trust-region subproblem to high accuracy even when the L-SR1 matrix is indefinite. The method makes use of two separate techniques. One technique uses (1) a Newton method to find σ^* that is initialized so its iterates converge monotonically to σ^* without any safeguarding when global solutions lie on the boundary of the trust region, and (2) the compact formulation of SR1 matrices together with the strategy found in [BWX96] to compute $\mathbf{s}^*$ directly by formula. The other technique is newly proposed. This technique computes an optimal pair $(\mathbf{s}^*, \sigma^*)$ using an orthonormal basis for the eigenspace of $\mathbf{B}_k$. The idea of using an orthonormal basis to represent $\mathbf{s}^*$ is not new; this approach is found in [BGYZ16]. Here, we apply this approach to the cases when $\mathbf{B}_k$ is singular and indefinite.

We begin by providing an overview of the OBS method. To solve the trust-region subproblem, we first attempt to compute an unconstrained minimizer $\mathbf{s}_u$ to (2.4). If the objective function $Q(\mathbf{s})$ is strictly convex (i.e., $\mathbf{B}_k \succ 0$) and the unconstrained minimizer lies inside the trust region, the optimal solution for the trust-region subproblem is given by $\mathbf{s}^* = \mathbf{s}_u$ and $\sigma^* = 0$. This computation is simplified by first finding the eigenvalues of $\mathbf{B}_k$ (see (1.10)); the solution $\mathbf{s}_u$ to the unconstrained problem is found using a strategy proposed in [BWX96], adapted for L-SR1 matrices. If $\|\mathbf{s}_u\|_2 > \Delta_k$ or is not well-defined, a global solution of the trust-region subproblem must lie on the boundary, i.e., it is a root of the following function, also known as the *secular-equation*:

$$\phi(\sigma) = \frac{1}{\|\mathbf{s}(\sigma)\|_2} - \frac{1}{\Delta_k}. \tag{2.6}$$

When a global solution is on the boundary, we consider three cases separately: (i) $\mathbf{B}_k$ is positive definite and $\|\mathbf{s}_u\|_2 > \Delta_k$, (ii) $\mathbf{B}_k$ is positive semidefinite, and (iii) $\mathbf{B}_k$ is indefinite. We note that the so-called *hard case* can only occur in the third case. Details for each case are provided below; however, we begin by considering the unconstrained case.

Computing the unconstrained minimizer. The OBS method begins by computing the eigenvalues of $\mathbf{B}_k$ as in Section 1.4.3. If $\mathbf{B}_k$ is positive definite, the method computes $\|\mathbf{s}_u\|_2$ using properties of orthogonal matrices. If $\|\mathbf{s}_u\|_2 \leq \Delta_k$, then $(\mathbf{s}^*, \sigma^*) = (\mathbf{s}_u, 0)$. We begin by presenting the computation of $\|\mathbf{s}_u\|_2$, which is only performed when $\mathbf{B}_k$ is positive definite. We include σ in the derivation for completeness even though $\sigma = 0$ when finding the unconstrained minimizer.

The unconstrained minimizer $\mathbf{s}_u$ is the solution to the first optimality condition in (2.5); however, the unconstrained minimizer can also be found by rewriting the optimality condition using the spectral decomposition of $\mathbf{B}_k$. Specifically, suppose $\mathbf{B}_k =$

$\mathbf{P}\Lambda\mathbf{P}^T$ is the spectral decomposition of $\mathbf{B}_k$, then

$$-\mathbf{g}_k = (\mathbf{B}_k + \sigma\mathbf{I}_n)\mathbf{s} = (\mathbf{P}\Lambda\mathbf{P}^T + \sigma\mathbf{I}_n)\mathbf{s} = \mathbf{P}(\Lambda + \sigma\mathbf{I}_n)\mathbf{v},$$

where $\mathbf{v} = \mathbf{P}^T\mathbf{s}$. Since $\mathbf{P}$ is orthogonal, the first optimality condition expressed in (2.1) can be written as

$$(\Lambda + \sigma\mathbf{I}_n)\mathbf{v} \quad = \quad -\mathbf{P}^T\mathbf{g}_k. \tag{2.7}$$

Note that the spectral decomposition of $\mathbf{B}_k$ transforms the first system in (2.5) into a solve with a diagonal matrix in (2.7). If we express the right hand side as

$$\mathbf{P}^T\mathbf{g}_k = [\ \mathbf{P}_\| \quad \mathbf{P}_\perp]^T\mathbf{g}_k = \begin{bmatrix}\mathbf{P}_\|^T\mathbf{g}_k \\ \mathbf{P}_\perp^T\mathbf{g}_k\end{bmatrix} \triangleq \begin{bmatrix}\mathbf{g}_\| \\ \mathbf{g}_\perp\end{bmatrix},$$

then

$$\|\mathbf{s}(\sigma)\|_2^2 = \|\mathbf{v}(\sigma)\|_2^2 = \left\{\sum_{i=1}^{k+1}\frac{(\mathbf{g}_\|)_i^2}{(\lambda_i+\sigma)^2}\right\} + \frac{\|\mathbf{g}_\perp\|_2^2}{(\gamma_{k-1}+\sigma)^2}. \tag{2.8}$$

Thus, the length of the unconstrained minimizer $\mathbf{s}_u = \mathbf{s}(0)$ is computed as $\|\mathbf{s}_u\|_2 = \|\mathbf{v}(0)\|_2$, where $\mathbf{g}_\| = \mathbf{P}_\|^T\mathbf{g}_k = (\mathbf{Q}\mathbf{U})^T\mathbf{g}_k = (\mathbf{\Psi}_k\mathbf{R}^{-1}\mathbf{U})^T\mathbf{g}_k$ and $\|\mathbf{g}_\perp\|_2^2 = \|\mathbf{g}_k\|_2^2 - \|\mathbf{g}_\|\|_2^2$. Notice that determining $\|\mathbf{s}_u\|_2$ does not require forming $\mathbf{s}_u$ explicitly. Moreover, we are able to compute $\|\mathbf{g}_\perp\|_2$ without having to compute $\mathbf{g}_\perp = \mathbf{P}_\perp^T\mathbf{g}_k$, which requires computing $\mathbf{P}_\perp$, whose columns form a basis orthogonal to $\mathbf{P}_\|$.

If $\|\mathbf{s}_u\|_2 \leq \Delta_k$, then $\mathbf{s}^* = \mathbf{s}_u$ and $\sigma^* = 0$. To compute $\mathbf{s}_u$, we use the Sherman-Morrison-Woodbury formula for the inverse of $\mathbf{B}_k$ as in [BWX96], adapted from the BFGS setting into the SR1 setting:

$$\mathbf{s}^* = -\frac{1}{\tau^*}\left[\mathbf{I}_n - \mathbf{\Psi}_k(\tau^*\mathbf{M}_k^{-1} + \mathbf{\Psi}_k^T\mathbf{\Psi}_k)^{-1}\mathbf{\Psi}_k^T\right]\mathbf{g}_k, \tag{2.9}$$

where $\tau^* = \gamma_{k-1}$. Notice that this formula calls for the inversion of $(\tau^*\mathbf{M}_k^{-1} + \mathbf{\Psi}_k^T\mathbf{\Psi}_k)$; however, the size of this matrix is small $(l \times l)$, making the computation practical.

On the other hand, if $\|\mathbf{s}_u\|_2 > \Delta_k$, then the solution $\mathbf{s}^*$ must lie on the boundary. We now consider the three cases as mentioned above.

Case (i): $\mathbf{B}_k$ is positive definite and $\|\mathbf{s}_u\|_2 > \Delta_k$. Since the unconstrained minimizer lies outside the trust region and $\|\mathbf{s}_u\|_2 = \|\mathbf{s}(0)\|_2$, then $\phi(\sigma)$ given by (2.6) is such that $\phi(0) < 0$. In this case, the OBS method uses Newton's method to find σ^*. (Details on Newton's method are provided in Subsection 2.4.3.) Finally, setting $\tau^* = \gamma_{k-1} + \sigma^*$, the global solution of the trust-region subproblem, $\mathbf{s}^*$, is computed using (2.9).

Case (ii): $\mathbf{B}_k$ is singular and positive semidefinite. Since $\gamma_{k-1} \neq 0$ and $\mathbf{B}_k$ is positive semidefinite, the leftmost eigenvalue is $\lambda_1 = 0$. Let r be the multiplicity of

the zero eigenvalue; that is, $\lambda_1 = \lambda_2 = \ldots = \lambda_r = 0 < \lambda_{r+1}$. For $\sigma > 0$, the matrix $(\Lambda + \sigma \mathbf{I}_n)$ is invertible, and thus, $\|\mathbf{s}(\sigma)\|_2$ in (2.8) is well-defined for $\sigma \in (0, \infty)$. If $\lim_{\sigma \to 0^+} \phi(\sigma) < 0$, the OBS method uses Newton's method to find σ^*. (Details on Newton's method are provided in Subsection 2.4.3.) Setting $\tau^* = \gamma_{k-1} + \sigma^*$, $\mathbf{s}^*$ is computed using (2.9).

We now consider the remaining case: $\lim_{\sigma \to 0^+} \phi(\sigma) \geq 0$. By [CGT00, Lemma 7.3.1], $\phi(\sigma)$ is strictly increasing on the interval $(0, \infty)$. Thus, ϕ can only have a root in the interval $[0, \infty]$ at $\sigma = 0$. We now show that $(\mathbf{s}^*, \sigma^*)$ is a global solution of the trust-region subproblem with $\sigma^* = 0$ and

$$\mathbf{s}^* = -\mathbf{B}_k^{\dagger} \mathbf{g}_k = -\mathbf{P}(\Lambda + \sigma^* \mathbf{I}_n)^{\dagger} \mathbf{P}^T \mathbf{g}_k,$$

where $\dagger$ denotes the Moore-Penrose pseudoinverse. The second optimality condition holds in (2.5) since $\sigma^* = 0$. It can be shown that the first optimality condition holds by using the fact that $\mathbf{g}_k$ must be perpendicular to the eigenspace corresponding to the 0 eigenvalue of $\mathbf{B}_k$, i.e., $(\mathbf{P}_{\|}^T \mathbf{g}_k)_i = 0$ for $i = 1, \ldots, r$ (see [MS83]).

In this subcase, the trust-region subproblem solution $\mathbf{s}^*$ can be computed as follows (here `c1` represents the condition $\sigma^* \neq -\gamma_{k-1}$):

$$\begin{aligned}
\mathbf{s}^* &= -\mathbf{P}(\Lambda + \sigma^* \mathbf{I}_n)^{\dagger} \mathbf{P}^T \mathbf{g}_k \\
&= \begin{cases} -\mathbf{P}_{\|}(\Lambda_1 + \sigma^* I)^{\dagger} \mathbf{P}_{\|}^T \mathbf{g}_k - \dfrac{1}{\gamma_{k-1} + \sigma^*} \mathbf{P}_{\perp} \mathbf{P}_{\perp}^T \mathbf{g}_k & \text{if } \texttt{c1}, \\ -\mathbf{P}_{\|}(\Lambda_1 + \sigma^* \mathbf{I}_n)^{-1} \mathbf{P}_{\|}^T \mathbf{g}_k & \text{otherwise} \end{cases} \\
&= \begin{cases} -\mathbf{\Psi}_k \mathbf{R}^{-1} \mathbf{U} (\Lambda_1 + \sigma^* \mathbf{I}_n)^{\dagger} \mathbf{g}_{\|} - \dfrac{1}{\gamma_{k-1} + \sigma^*} (\mathbf{I}_n - \mathbf{\Psi}_k \mathbf{R}^{-1} \mathbf{R}^{-T} \mathbf{\Psi}_k^T) \mathbf{g}_k & \text{if } \texttt{c1}, \\ -\mathbf{\Psi}_k \mathbf{R}^{-1} \mathbf{U} (\Lambda_1 + \sigma^* \mathbf{I}_n)^{-1} \mathbf{g}_{\|} & \text{otherwise}, \end{cases}
\end{aligned} \tag{2.10}$$

which makes use of a chain of equalities:

$$\mathbf{P}_{\perp} \mathbf{P}_{\perp}^T \mathbf{g}_k = (\mathbf{I}_n - \mathbf{P}_{\|} \mathbf{P}_{\|}^T) \mathbf{g}_k = (\mathbf{I}_n - \mathbf{\Psi}_k \mathbf{R}^{-1} \mathbf{R}^{-T} \mathbf{\Psi}_k^T) \mathbf{g}_k.$$

The actual computation of $\mathbf{s}^*$ in (2.10) requires only matrix-vector products; no additional large matrices need to be formed to find a global solution of the trust-region subproblem.

Case (iii): $\mathbf{B}_k$ is indefinite. Since $\mathbf{B}_k$ is indefinite, $\lambda_{\min} = \min\{\lambda_1, \gamma_{k-1}\} < 0$. Let r be the algebraic multiplicity of the leftmost eigenvalue. For $\sigma > -\lambda_{\min}$, $(\Lambda + \sigma \mathbf{I}_n)$ is invertible, and thus, $\|\mathbf{s}(\sigma)\|_2$ in (2.8) is well defined in the interval $(-\lambda_{\min}, \infty)$.

If $\lim_{\sigma \to -\lambda_{\min}^+} \phi(\sigma) < 0$, then there exists $\sigma^* \in (-\lambda_{\min}, \infty)$ with $\phi(\sigma^*) = 0$ that can be obtained as in Case (i) using Newton's method (see Sec. 2.4.3). The solution $\mathbf{s}^*$ is computed via (2.9) with $\tau^* = \gamma_{k-1} + \sigma^*$.

If $\lim_{\sigma \to -\lambda_{\min}^+} \phi(\sigma) \geq 0$, then $\mathbf{g}_k$ must be orthogonal to the eigenspace associated with the leftmost eigenvalue of $\mathbf{B}_k$ [MS83]. In other words, if $\lambda_{\min} = \lambda_1$, then $(\mathbf{g}_\parallel)_i = 0$ for $i = 1, \ldots, r$; otherwise, if $\lambda_{\min} = \gamma_{k-1}$, then $\|\mathbf{g}_\perp\|_2 = 0$. We now consider the cases of equality and inequality separately.

If $\lim_{\sigma \to -\lambda_{\min}^+} \phi(\sigma) = 0$, then $\sigma^* = -\lambda_{\min} > 0$, and a global solution of the trust-region subproblem is given by

$$\mathbf{s}^* = -(\mathbf{B}_k + \sigma^* \mathbf{I}_n)^\dagger \mathbf{g}_k = -\mathbf{P}(\Lambda + \sigma^* \mathbf{I}_n)^\dagger \mathbf{P}^T \mathbf{g}_k.$$

As in Case (ii), $\mathbf{s}^*$ is obtained from (2.10) and can be shown to satisfy the optimality conditions (2.5).

Finally, if $\lim_{\sigma \to -\lambda_{\min}^+} \phi(\sigma) > 0$, then

$$\lim_{\sigma \to -\lambda_{\min}^+} \|\mathbf{s}(\sigma)\|_2 = \lim_{\sigma \to -\lambda_{\min}^+} \| - (\mathbf{B}_k + \sigma^* \mathbf{I}_n)^{-1} \mathbf{g}_k\|_2 < \Delta_k.$$

This corresponds to the so-called *hard case*. The optimal solution is given by

$$\mathbf{s}^* = \hat{\mathbf{s}}^* + \mathbf{z}^*, \quad \text{where} \quad \hat{\mathbf{s}}^* = -\left(\mathbf{B}_k + \sigma^* \mathbf{I}_n\right)^\dagger \mathbf{g}_k, \quad \mathbf{z}^* = \alpha \mathbf{u}_{\min}, \tag{2.11}$$

and where $\mathbf{u}_{\min}$ is an eigenvector associated with $\lambda_{\min}$ and α is computed so that $\|\mathbf{s}^*\|_2 = \Delta_k$ [MS83]. As in Case (ii), we avoid forming $\mathbf{P}_\perp$ using (2.10) to compute $\hat{\mathbf{s}}^*$. The computation of $\mathbf{u}_{\min}$ depends on whether $\lambda_{\min}$ is found in Λ_1 or Λ_2 in (1.10). If $\lambda_{\min} = \lambda_1$ then the first column of $\mathbf{P}$ is a leftmost eigenvector of $\mathbf{B}_k$, and thus, $\mathbf{u}_{\min}$ is set to the first column of $\mathbf{P}_\parallel$. On the other hand, if $\lambda_{\min} = \gamma_{k-1}$, then any vector in the column space of $\mathbf{P}_\perp$ will be an eigenvector of $\mathbf{B}_k$ corresponding to $\lambda_{\min}$. Since $\text{Range}(\mathbf{P}_\parallel)^\perp = \text{Range}(\mathbf{P}_\perp)$, the projection matrix $(\mathbf{I}_n - \mathbf{P}_\parallel \mathbf{P}_\parallel^T)$ maps onto the column space of $\mathbf{P}_\perp$. For simplicity, we map one canonical basis vector at a time (starting with $\mathbf{e}_1$) into the space spanned by the columns of $\mathbf{P}_\perp$ until we obtain a nonzero vector. Since $\dim(\mathbf{P}_\parallel) = l \ll n$, this process is practical and will result with a vector that lies in $\text{Range}(\mathbf{P}_\perp)$; that is, $\mathbf{u}_{\min} \equiv (\mathbf{I}_n - \mathbf{P}_\parallel \mathbf{P}_\parallel^T)\mathbf{e}_j$ for at least one j in $\{1 \leq j \leq l+1\}$ with $\|\mathbf{u}_{\min}\|_2 \neq 0$. (We note that both λ_1 and γ_{k-1} cannot both be $\lambda_{\min}$ since $\lambda_1 = \hat{\lambda}_1 + \gamma_{k-1}$ and $\hat{\lambda}_1 \neq 0$ (see Section 1.10)).

The following theorem provides details for computing optimal trust-region subproblem solutions characterized by Theorem 2.1 for the case when $\mathbf{B}_k$ is indefinite.

Theorem 2.2. *Consider the trust-region subproblem given by*

$$\underset{\mathbf{s} \in \mathbb{R}^n}{\textit{minimize}} \quad Q(\mathbf{s}) = \mathbf{g}_k^T \mathbf{s} + \frac{1}{2} \mathbf{s}^T \mathbf{B}_k \mathbf{s} \quad \textit{subject to} \quad \|\mathbf{s}\|_2 \leq \Delta_k,$$

where $\mathbf{B}_k$ is indefinite. Suppose $\mathbf{B}_k = \mathbf{P}\Lambda\mathbf{P}^T$ is the spectral decomposition of $\mathbf{B}_k$, and

without loss of generality, assume $\Lambda = diag(\lambda_1, \ldots, \lambda_n)$ *is such that* $\lambda_{\min} = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$. *Further, suppose* $\mathbf{g}_k$ *is orthogonal to the eigenspace associated with* $\lambda_{\min}$, *i.e.,* $\mathbf{g}_k^T \mathbf{P} \mathbf{e}_j = 0$ *for* $j = 1, \ldots, r$, *where* $r \geq 1$ *is the algebraic multiplicity of* $\lambda_{\min}$. *Then, if the optimal solution of the subproblem is with* $\sigma^* = -\lambda_{\min}$, *then the global solutions to the trust-region subproblem are given by* $\mathbf{s}^* = \hat{\mathbf{s}}^* + \mathbf{z}^*$ *where* $\hat{\mathbf{s}}^* = -\left(\mathbf{B}_k + \sigma^* \mathbf{I}_n\right)^{\dagger} \mathbf{g}_k$ *and* $\mathbf{z}^* = \pm \alpha \mathbf{u}_{\min}$, *where* $\mathbf{u}_{\min}$ *is a unit vector in the eigenspace associated with* $\lambda_{\min}$ *and* $\alpha = \sqrt{\Delta_k^2 - \|\hat{\mathbf{s}}^*\|_2^2}$. *Moreover,*

$$Q(\hat{\mathbf{s}}^* \pm \alpha \mathbf{z}^*) = \frac{1}{2} \mathbf{g}_k^T \hat{\mathbf{s}}^* - \frac{1}{2} \sigma^* \Delta_k^2. \tag{2.12}$$

Proof. By [MS83], a global solution of trust-region subproblem is given by $\mathbf{s}^* = \hat{\mathbf{s}}^* + \mathbf{z}^*$ where $\hat{\mathbf{s}}^* = -\left(\mathbf{B}_k + \sigma^* \mathbf{I}_n\right)^{\dagger} \mathbf{g}_k$, $\mathbf{z}^* = \bar{\alpha}\, \mathbf{u}_{\min}$, and $\bar{\alpha}$ is such that $\|\mathbf{s}^*\|_2 = \Delta_k$. It remains to show that both roots of the quadratic equation $\|\hat{\mathbf{s}}^* + \alpha \mathbf{u}_{\min}\|_2^2 = \Delta_k^2$ are given by $\alpha = \pm \sqrt{\Delta_k^2 - \|\hat{\mathbf{s}}^*\|_2^2}$ and that (2.12) holds.

To see this, we begin by showing that $(\hat{\mathbf{s}}^*)^T \mathbf{z}^* = 0$. Let $r \geq 1$ be the algebraic multiplicity of $\lambda_{\min}$. Then, $\hat{\mathbf{s}}^* = -(\mathbf{B}_k + \sigma^* \mathbf{I}_n)^{\dagger} \mathbf{g}_k = -\mathbf{P}(\Lambda + \sigma^* \mathbf{I}_n)^{\dagger} \mathbf{P}^T \mathbf{g}_k = -\mathbf{P}\mathbf{v}(\sigma^*)$, where $\mathbf{v}(\sigma^*) \equiv (\Lambda + \sigma^* \mathbf{I}_n)^{\dagger} \mathbf{P}^T \mathbf{g}_k$. Notice that by definition of the pseudoinverse, $\mathbf{v}(\sigma^*)_i = 0$ for $i = 0, \ldots, r$. Since $\mathbf{u}_{\min}$ is in the eigenspace associated with $\lambda_{\min}$, then it can be written as a linear combination of the first r columns of $\mathbf{P}$, i.e., $\mathbf{u}_{\min} = \sum_{i=1}^{r} \tilde{u}_i \mathbf{P} \mathbf{e}_i$ for some $\{\tilde{u}_i\} \in \Re$ where $\mathbf{e}_i$ denotes the ith canonical basis vector. Then,

$$(\hat{\mathbf{s}}^*)^T \mathbf{z} = \alpha\, (\hat{\mathbf{s}}^*)^T \mathbf{u}_{\min} = \alpha (\mathbf{P}\mathbf{v}(\sigma^*))^T \left(\sum_{i=1}^{r} \tilde{u}_i \mathbf{P} \mathbf{e}_i \right) = \alpha \mathbf{v}(\sigma^*)^T \sum_{i=1}^{r} \tilde{u}_i \mathbf{e}_i = 0,$$

since the first r entries of $\mathbf{v}(\sigma^*)$ are zero. Since $\hat{\mathbf{s}}^*$ is orthogonal to $\mathbf{z}^*$, then

$$\alpha = \pm \sqrt{\Delta_k^2 - \|\hat{\mathbf{s}}^*\|_2^2}.$$

To see (2.12), consider the following:

$$\begin{aligned} Q(\hat{\mathbf{s}}^* \pm \alpha\, \mathbf{u}_{\min}) &= (\hat{\mathbf{s}}^* \pm \alpha\, \mathbf{u}_{\min})^T \mathbf{g}_k + \frac{1}{2} (\hat{\mathbf{s}}^* \pm \alpha\, \mathbf{u}_{\min})^T \mathbf{B}_k (\hat{\mathbf{s}}^* \pm \alpha\, \mathbf{u}_{\min}) \\ &= (\hat{\mathbf{s}}^* \pm \alpha\, \mathbf{u}_{\min})^T (\mathbf{g}_k - \frac{1}{2} \mathbf{g}_k - \frac{1}{2} \sigma^* (\hat{\mathbf{s}}^* \pm \alpha\, \mathbf{u}_{\min})) \\ &= \frac{1}{2} (\hat{\mathbf{s}}^* \pm \alpha\, \mathbf{u}_{\min})^T \mathbf{g}_k - \frac{1}{2} \sigma^* \|\hat{\mathbf{s}}^* \pm \alpha\, \mathbf{u}_{\min}\|_2^2 \\ &= \frac{1}{2} \mathbf{g}_k^T \hat{\mathbf{s}}^* - \frac{1}{2} \sigma^* \Delta_k^2, \end{aligned} \tag{2.13}$$

since $\mathbf{u}_{\min}^T \mathbf{g}_k = \left(\sum_{i=1}^{r} \tilde{u}_i \mathbf{P} \mathbf{e}_i\right)^T \mathbf{g}_k = \left(\sum_{i=1}^{r} \tilde{u}_i \mathbf{e}_i^T \mathbf{P}^T\right) \mathbf{g}_k = 0$ since $\mathbf{g}_k$ is orthogonal to the eigenspace associated with $\lambda_{\min}$. □ ■

The OBS method is summarized in Algorithm 2.1.

ALGORITHM 2.1

Compute $\mathbf{\Psi}_k = \mathbf{QR}$; or $\mathbf{\Psi}_k^T\mathbf{\Psi}_k = \mathbf{R}^T\mathbf{R}$;

Compute $\mathbf{RM}_k\mathbf{R}^T = \mathbf{U}\hat{\Lambda}\mathbf{U}^T$ with $\hat{\lambda}_1 \leq \hat{\lambda}_2 \leq \cdots \leq \hat{\lambda}_{k+1}$;

Set $\Lambda_1 = \hat{\Lambda} + \gamma_{k-1}\mathbf{I}_n$;

Set $\lambda_{\min} = \min(\lambda_1, \gamma_{k-1})$;

Define $\mathbf{P}_{\|} \equiv \mathbf{\Psi}_k\mathbf{R}^{-1}\mathbf{U}$ and $\mathbf{g}_{\|} \equiv \mathbf{P}_{\|}^T\mathbf{g}_k$;

Compute $a_j = (\mathbf{g}_{\|})_j$, $j = 1, \ldots, l$; $a_{l+1} = \sqrt{\|\mathbf{g}_k\|_2^2 - \|\mathbf{g}_{\|}\|_2^2}$;

1. If $\lambda_{\min} > 0$ and $\bar{\phi}(0) \geq 0$, then set $\sigma^* = 0$; compute $\mathbf{s}^*$ by (2.9); terminate;
2. If $\lambda_{\min} \leq 0$ and $\bar{\phi}(-\lambda_{\min}) \geq 0$, then set $\sigma^* = -\lambda_{\min}$; compute $\mathbf{s}^*$ by (2.10); If $\lambda_{\min} < 0$, then compute $\mathbf{z}^*$ by (2.11), update $\mathbf{s}^* \leftarrow \mathbf{s}^* + \mathbf{z}^*$; terminate;
3. Otherwise find σ^* with $\phi(\sigma^*) = 0$, $\sigma^* \in (\max\{-\lambda_{\min}, 0\}, \infty)$ by Newton's method; compute $\mathbf{s}^*$ by (2.9) with $\tau^* = \sigma^* + \gamma_{k-1}$; terminate;

2.4.3 NEWTON'S METHOD

Newton's method is used to find a root of $\phi(\sigma)$ whenever

$$\lim_{\sigma \to -\lambda_{\min}^+} \phi(\sigma) = \lim_{\sigma \to -\lambda_{\min}^+} \frac{1}{\|\mathbf{s}(\sigma)\|_2} - \frac{1}{\Delta_k} < 0.$$

Since $\|\mathbf{s}(\sigma)\|_2$ does not exist at the eigenvalues of $\mathbf{B}_k$, we first define the continuous extension of $\phi(\sigma)$, whose domain is all of $\mathbb{R}$. Let $a_i = (\mathbf{g}_{\|})_i$ for $1 \leq i \leq l$, $a_{l+1} = \|\mathbf{g}_{\perp}\|_2$, and $\lambda_{l+1} = \gamma_{k-1}$. Combining the terms in (2.8) that correspond to the same eigenvalues and eliminating all terms with zero numerators, we have that for $\sigma \neq \lambda_i$, $\|\mathbf{s}(\sigma)\|_2^2$ can be written as

$$\|\mathbf{s}(\sigma)\|_2^2 = \sum_{i=1}^{l+1} \frac{a_i^2}{(\lambda_i + \sigma)^2} = \sum_{i=1}^{L} \frac{\bar{a}_i^2}{(\bar{\lambda}_i + \sigma)^2},$$

such that for $i = 1, \ldots, L$, $\bar{a}_i \neq 0$ and $\bar{\lambda}_i$ are *distinct* eigenvalues of $\mathbf{B}_k$ with $\bar{\lambda}_1 < \bar{\lambda}_2 < \cdots < \bar{\lambda}_L$. Note that the last sum is well-defined for $\sigma = \lambda_j \neq -\bar{\lambda}_i$, for $1 \leq i \leq L$. Then,

the continuous extension $\bar{\phi}(\sigma)$ of $\phi(\sigma)$ is given by:

$$\bar{\phi}(\sigma) = \begin{cases} -\dfrac{1}{\Delta_k} & \text{if } \sigma = -\bar{\lambda}_i, \quad 1 \le i \le L \\ \dfrac{1}{\sqrt{\displaystyle\sum_{i=1}^{L} \frac{\bar{a}_i^2}{(\bar{\lambda}_i + \sigma)^2}}} - \dfrac{1}{\Delta_k} & \text{otherwise.} \end{cases}$$

A crucial characteristic of $\bar{\phi}$ is that it takes on the value of the limit of ϕ at $\sigma = -\lambda_i$, for $1 \le i \le l+1$. In other words, for each $i \in \{1, \ldots, l+1\}$,

$$\lim_{\sigma \to -\lambda_i} \phi(\sigma) = \bar{\phi}(-\lambda_i).$$

The derivative of $\bar{\phi}(\sigma)$ is used only for Newton's method and is computed as follows:

$$\bar{\phi}'(\sigma) = \left(\sum_{i=1}^{L} \frac{\bar{a}_i^2}{(\bar{\lambda}_i + \sigma)^2} \right)^{-\frac{3}{2}} \sum_{i=1}^{L} \frac{\bar{a}_i^2}{(\bar{\lambda}_i + \sigma)^3} \qquad \text{if } \sigma \neq -\bar{\lambda}_i, \ 1 \le i \le L. \tag{2.14}$$

Note that $\bar{\phi}'(-\lambda_j)$ exists as long as $-\lambda_j \neq -\bar{\lambda}_i$, for $1 \le i \le L$. Furthermore, for $\sigma > -\bar{\lambda}_1$, $\bar{\phi}'(\sigma) > 0$, i.e., $\bar{\phi}(\sigma)$ is strictly increasing on the interval $[-\bar{\lambda}_1, \infty)$. Finally, it can be shown that $\bar{\phi}''(\sigma) < 0$ for $\sigma > -\bar{\lambda}_1$, i.e., $\bar{\phi}(\sigma)$ is concave on the interval $[-\bar{\lambda}_1, \infty)$. For illustrative purposes, we plot examples of $\bar{\phi}(\sigma)$ in Fig. 1 for the different cases we considered in this Section 2.5.2. Note that we use Newton's method to find σ^* when (a) $\lambda_{\min} \ge 0$ and $\bar{\phi}(0)$ (see Figs. 1(b) and (c)), or (b) $\lambda_{\min} < 0$ and $\bar{\phi}(-\lambda_{\min}) < 0$ (see Figs. 1(d) and (e)).

We now define an initial iterate such that Newton's method is guaranteed to converge to σ^* monotonically.

Theorem 2.3. *Suppose* $\bar{\phi}(\max\{0, -\lambda_{\min}\}) < 0$. *Let*

$$\hat{\sigma} \triangleq \max_{1 \le i \le k+2} \left\{ \frac{|a_i|}{\Delta_k} - \lambda_i \right\} = \frac{|a_j|}{\Delta_k} - \lambda_j \tag{2.15}$$

for some $1 \le j \le k+2$. *Newton's method applied to* $\bar{\phi}(\sigma)$ *with initial iterate* $\sigma^{(0)} \triangleq \max\{0, \hat{\sigma}\}$ *is guaranteed to converge to* σ^* *monotonically.*

Proof. Since $\bar{\phi}(\sigma)$ is strictly increasing and concave on $[-\lambda_{\min}, \infty)$ and $\bar{\phi}(\sigma^*) = 0$, it is sufficient to show that (i) $-\lambda_{\min} \le \sigma^{(0)} \le \sigma^*$, and (ii) $\bar{\phi}'(\sigma^{(0)})$ exists (see e.g., [KC02]).

We note that $\hat{\sigma} \ge -\lambda_{\min}$, and thus, $\sigma^{(0)} \ge \max\{0, -\lambda_{\min}\} \ge -\lambda_{\min}$. To show that $\sigma^{(0)} \le \sigma^*$, we show that $\bar{\phi}(\sigma^{(0)}) \le \bar{\phi}(\sigma^*) = 0$.

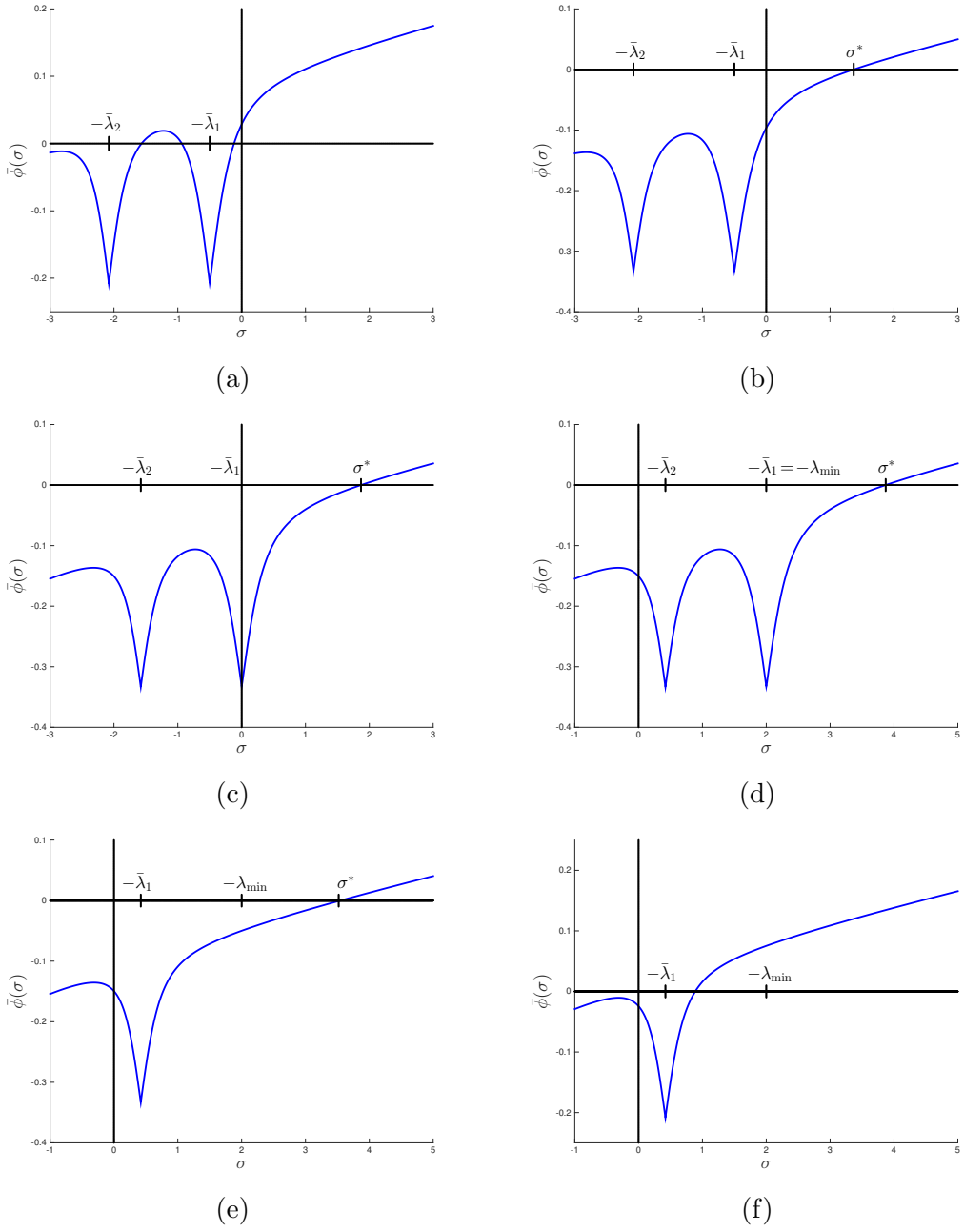


FIGURE 2.1 *Graphs of the function $\bar{\phi}(\sigma)$. (a) The positive-definite case where the unconstrained minimizer is within the trust-region radius, i.e., $\bar{\phi}(0) \geq 0$, and $\sigma^* = 0$. (b) The positive-definite case where the unconstrained minimizer is infeasible, i.e., $\bar{\phi}(0) < 0$. (c) The singular case where $\bar{\lambda}_1 = \lambda_{min} = 0$. (d) The indefinite case where $\bar{\lambda}_1 = \lambda_{min} < 0$. (e) When the coefficients a_i corresponding to λ_{min} are all 0, $\bar{\phi}(\sigma)$ does not have a singularity at λ_{min}. Note that this case is not the hard case since $\bar{\phi}(-\lambda_{\min}) < 0$. (f) The hard case where there does not exist $\sigma^* > -\lambda_{min}$ such that $\bar{\phi}(\sigma^*) = 0$.*

If $\hat{\sigma} = |a_j|/\Delta_k - \lambda_j$ with $|a_j| \neq 0$, then evaluating $\|\mathbf{s}(\sigma)\|_2$ at $\sigma = \hat{\sigma}$ yields

$$\|\mathbf{s}(\hat{\sigma})\|_2^2 \;=\; \sum_{i=1}^{k+2} \frac{a_i^2}{(\lambda_i + \hat{\sigma})^2} \;\geq\; \frac{a_j^2}{(\lambda_j + \hat{\sigma})^2} \;=\; \frac{a_j^2}{(\lambda_j + \frac{|a_j|}{\Delta_k} - \lambda_j)^2} \;=\; \Delta_k^2,$$

and thus, $\bar{\phi}(\hat{\sigma}) \leq 0$. Since $\bar{\phi}(\max\{0, -\lambda_{\min}\}) < 0$, then $\bar{\phi}(\sigma^{(0)}) \leq 0$. If $|a_j| = 0$, then $\hat{\sigma} = -\lambda_j$. Since $-\lambda_i \leq -\lambda_{\min}$ for all i, $\hat{\sigma} = -\lambda_{\min}$. Thus, $\bar{\phi}(\sigma^{(0)}) = \bar{\phi}(\max\{0, -\lambda_{\min}\}) < 0$. Consequently, $\bar{\phi}(\sigma^{(0)}) \leq 0$, and therefore, $\sigma^{(0)} \leq \sigma^*$ since $\bar{\phi}(\sigma)$ is monotonically increasing.

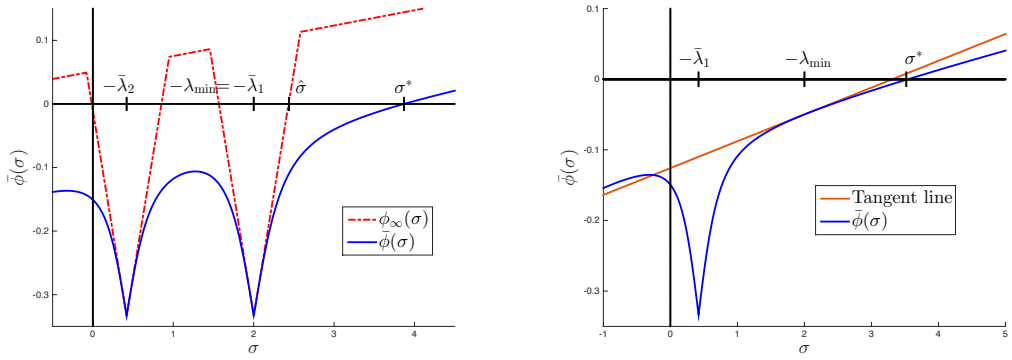
Next, we show that $\bar{\phi}'(\sigma^{(0)})$ exists. On the interval $(-\lambda_{\min}, \infty)$, $\bar{\phi}(\sigma)$ is differentiable (see (2.14)). Therefore, if $\sigma^{(0)} > -\lambda_{\min}$, then $\bar{\phi}'(\sigma^{(0)})$ exists. If $\sigma^{(0)} = -\lambda_{\min}$, then $\hat{\sigma} = -\lambda_{\min}$, which implies that $a_1 = \cdots = a_r = 0$ or $a_{k+2} = 0$ (see (2.15)). From the definition of $\bar{\phi}(\sigma)$, $\lambda_{\min} \neq \bar{\lambda}_i$ for $1 \leq i \leq L$. Thus, $\bar{\phi}(\sigma)$ is differentiable at $\sigma = -\lambda_{\min} = \sigma^{(0)}$. □

■

We note that when $a_j \neq 0$ in (2.15), $\hat{\sigma}$ is the largest σ that solves the secular equation with the infinity norm:

$$\phi_\infty(\hat{\sigma}) = \frac{1}{\|\mathbf{v}(\hat{\sigma})\|_\infty} - \frac{1}{\Delta_k} = 0.$$

We illustrate the choice of initial iterate for Newton's method in Fig. 2.



(a) (b)

FIGURE 2.2 *Choice of initial iterate for Newton's method. (a) If $a_j \neq 0$ in (2.15), then $\hat{\sigma}$ corresponds to the largest root of $\phi_\infty(\sigma)$ (in red). Here, $-\lambda_{\min} > 0$, and therefore $\sigma^{(0)} = \hat{\sigma}$. (b) If $a_j = 0$ in (2.15), then $\lambda_{\min} \neq \bar{\lambda}_1$, and therefore, $\bar{\phi}(\sigma)$ is differentiable at $-\lambda_{\min}$ since $\bar{\phi}(\sigma)$ is differentiable on $(-\bar{\lambda}_1, \infty)$. Here, $-\lambda_{\min} > 0$, and thus, $\sigma^{(0)} = \hat{\sigma} = -\lambda_{\min}$.*

Finally, we present Newton's method for computing σ^* in Algorithm 2.2.

ALGORITHM 2.2

Initialize: Tolerance $\tau > 0$

1. If $\bar{\phi}(\max\{0, -\lambda_{\min}\}) < 0$, then set $\hat{\sigma} = \max_{1 \leq j \leq k+2} \frac{|a_j|}{\Delta_k} - \lambda_j$, $\sigma = \max\{0, \hat{\sigma}\}$ go to 2.;

2. While $|\bar{\phi}(\sigma)| > \tau$, set $\sigma = \sigma - \bar{\phi}(\sigma)/\bar{\phi}'(\sigma)$ otherwise set $\sigma^* = \sigma$, terminate;

3. If $\lambda_{\min} < 0$, then set $\sigma^* = -\lambda_{\min}$, terminate;

4. Otherwise set $\sigma^* = 0$;

2.4.4 NUMERICAL EXPERIMENTS

In this section, we demonstrate the accuracy of the proposed OBS algorithm implemented in MATLAB to solve limited-memory SR1 trust-region subproblems. We generated five sets of experiments composed of problems of various sizes using random data. The Newton method to find a root of ϕ was terminated when the ith iterate satisfied $\|\phi(\sigma^{(i)})\| \leq \tau \|\phi(\sigma^{(0)})\| + \sqrt{\tau}$, where $\sigma^{(0)}$ denotes the initial iterate for Newton's method and τ corresponds to the machine precision. This is the only stopping criteria used by the OBS method since other aspects of this method compute solutions by formula. The problem sizes n range from $n = 10^3$ to $n = 10^7$. The number of limited-memory updates l was set to 5, and $\gamma_{k-1} = 0.5$ unless otherwise specified below. The pairs $\mathbf{S}_k$ and $\mathbf{Y}_k$, both $n \times l$ matrices, were generated from random data. Finally, $\mathbf{g}_k$ was generated by random data unless otherwise stated. The five sets of experiments are intended to be comprehensive: They include the unconstrained case and the three cases discussed in Subsection 2.4.2. The five experiments are as follows:

1. The matrix $\mathbf{B}_k$ is positive definite with $\|\mathbf{s}_u\|_2 \leq \Delta_k$: We ensure $\mathbf{\Psi}_k$ and $\mathbf{M}_k$ are such that $\mathbf{B}_k$ is strictly positive definite by altering the spectral decomposition of $\mathbf{R}\mathbf{M}_k\mathbf{R}^T$. We choose $\Delta_k = \mu\|\mathbf{s}_u\|_2$, where $\mu = 1.25$, to guarantee that the unconstrained minimizer is feasible. The graph of $\bar{\phi}(\sigma)$ corresponding to this case is illustrated in Fig. 1(a).

2. The matrix $\mathbf{B}_k$ is positive definite with $\|\mathbf{s}_u\|_2 > \Delta_k$: We ensure $\mathbf{\Psi}_k$ and $\mathbf{M}_k$ are such that $\mathbf{B}_k$ is strictly positive definite by altering the spectral decomposition of $\mathbf{R}\mathbf{M}_k\mathbf{R}^T$. We choose $\Delta_k = \mu\|\mathbf{s}_u\|_2$, where μ is randomly generated between 0 and 1, to guarantee that the unconstrained minimizer is infeasible. The graph of $\bar{\phi}(\sigma)$ corresponding to this case is illustrated in Fig. 1(b).

3. The matrix $\mathbf{B}_k$ is positive semidefinite and singular with $\mathbf{s} = -\mathbf{B}_k^{\dagger}\mathbf{g}_k$ infeasible: We ensure $\mathbf{\Psi}_k$ and $\mathbf{M}_k$ are such that $\mathbf{B}_k$ is positive semidefinite and singular by altering the spectral decomposition of $\mathbf{R}\mathbf{M}_k\mathbf{R}^T$. Two cases are tested: (a)

$\bar{\phi}(0) < 0$ and (b) $\bar{\phi}(0) \geq 0$. Case (a) occurs when $\Delta_k = (1+\mu)\|\mathbf{s}_u\|_2$, where μ is randomly generated between 0 and 1; case (b) occurs when $\Delta_k = \mu\|\mathbf{s}_u\|_2$, where μ is randomly generated between 0 and 1. The graph of $\bar{\phi}(\sigma)$ in case (a) corresponds to Fig. 1(c). In case (b), $a_i = 0$ for $i = 1, \ldots, r$, and thus, $\bar{\phi}(\sigma)$ does not have a singularity at $\sigma = 0$, implying the graph of $\bar{\phi}(\sigma)$ for this case corresponds to Fig 1(a).

4. The matrix $\mathbf{B}_k$ is indefinite with $\bar{\phi}(-\lambda_{\min}) < 0$: We ensure $\mathbf{\Psi}_k$ and $\mathbf{M}_k$ are such that $\mathbf{B}_k$ is indefinite by altering the spectral decomposition of $\mathbf{R}\mathbf{M}_k\mathbf{R}^T$. We test two subcases: (a) the vector $\mathbf{g}_k$ is generated randomly, and (b) a random vector $\mathbf{g}_k$ is projected onto the orthogonal complement of $\mathbf{P}_{\|_1} \in \mathbb{R}^{n\times r}$ so that $a_i = 0, i = 1, \ldots, r$, where $r = 2$. For case (b), $\Delta_k = \mu\|\mathbf{s}_u\|_2$, where μ is randomly generated between 0 and 1, so that $\bar{\phi}(-\lambda_{\min}) < 0$. The graph of $\bar{\phi}(\sigma)$ in case (a) corresponds to Fig. 1(d), and $\bar{\phi}(\sigma)$ in case (b) corresponds to Fig. 1(e).

5. The hard case ($\mathbf{B}_k$ is indefinite): We ensure $\mathbf{\Psi}_k$ and $\mathbf{M}_k$ are such that $\mathbf{B}_k$ is indefinite by altering the spectral decomposition of $\mathbf{R}\mathbf{M}_k\mathbf{R}^T$. We test two subcases: (a) $\lambda_{\min} = \lambda_1 = \hat{\lambda}_1 + \gamma < 0$, and (b) $\lambda_{\min} = \gamma < 0$. In both cases, $\Delta_k = (1+\mu)\|\mathbf{s}_u\|_2$, where μ is randomly generated between 0 and 1, so that $\bar{\phi}(-\lambda_{\min}) > 0$. The graph of $\bar{\phi}(\sigma)$ for both cases of the hard case corresponds to Fig. 1(f).

We report the following: (1) `opt 1 (abs)`$=\|(\mathbf{B}_k+\sigma^*\mathbf{I}_n)\mathbf{s}^*+\mathbf{g}_k\|_2$, which corresponds to the norm of the error in the first optimality conditions; (2) `opt 1 (rel)` $=(\|(\mathbf{B}_k + \sigma^*\mathbf{I}_n)\mathbf{s}^* + \mathbf{g}_k\|_2)/\|\mathbf{g}_k\|_2$, which corresponds to the norm of the *relative* error in the first optimality conditions; (3) `opt 2`$=\sigma^*|\mathbf{s}^* - \Delta_k|$, which corresponds to the absolute error in the second optimality conditions; (4) σ^*; and (5) Time. We ran each experiment five times and report one representative result for each experiment. We show in Fig. 2.4.4 the computational time for each of the five runs in each experiment.

For comparison, we report results for the OBS method as well as the Large-Scale Trust-Region Subproblem (LSTRS) method [RSS01, RSS08]. The LSTRS method solves large trust-region subproblems by converting the subproblems into parametrized eigenvalue problems. This method uses only matrix-vector products. For these tests, we suppressed all run-time output of the LSTRS method and supplied a routine to compute matrix-vector products using the factors in the compact formulation, i.e., given a vector $\mathbf{v}$, the product with $\mathbf{B}_k$ is computed as $\mathbf{B}_k\mathbf{v} \leftarrow \gamma_{k-1}\mathbf{v} + \mathbf{\Psi}_k(\mathbf{M}_k(\mathbf{\Psi}_k^T\mathbf{v}))$. Note that the computations of $\mathbf{M}_k$ and $\mathbf{\Psi}_k$ are not included in the time counts for LSTRS.

Tables 2.1 and 2.2 shows the results of Experiment 1. In all cases, the OBS method and the LSTRS method found global solutions of the trust-region subproblems. The relative error in the OBS method is smaller than the relative error in the LSTRS method. Moreover, the OBS method solved each subproblem in less time than the LSTRS method.

TABLE 2.1
Experiment 1: OBS method with $\mathbf{B}_k$ is positive definite and $\|\mathbf{s}_u\|_2 \leq \Delta_k$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	3.24e-15	1.03e-16	0.00e+00	0.00e+00	2.12e-02
1.0e+04	1.21e-14	1.21e-16	0.00e+00	0.00e+00	2.76e-02
1.0e+05	4.61e-14	1.46e-16	0.00e+00	0.00e+00	5.46e-02
1.0e+06	1.08e-13	1.08e-16	0.00e+00	0.00e+00	5.34e-01
1.0e+07	5.31e-13	1.68e-16	0.00e+00	0.00e+00	5.34e+00

TABLE 2.2
Experiment 1: LSTRS method with $\mathbf{B}_k$ is positive definite and $\|\mathbf{s}_u\|_2 \leq \Delta_k$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	2.11e-05	6.70e-07	0.00e+00	0.00e+00	4.72e-01
1.0e+04	8.27e-07	8.28e-09	0.00e+00	0.00e+00	4.98e-01
1.0e+05	2.64e-07	8.37e-10	0.00e+00	0.00e+00	9.15e-01
1.0e+06	3.54e-09	3.53e-12	0.00e+00	0.00e+00	7.08e+00
1.0e+07	2.79e-09	8.81e-13	0.00e+00	0.00e+00	6.66e+01

TABLE 2.3
Experiment 2: OBS method with $\mathbf{B}_k$ is positive definite and $\|\mathbf{s}_u\|_2 > \Delta_k$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	3.44e-15	1.06e-16	1.75e-09	4.82e+01	2.83e-02
1.0e+04	1.35e-14	1.35e-16	5.83e-13	1.99e+01	2.70e-02
1.0e+05	3.34e-14	1.06e-16	6.15e-13	1.57e+01	6.39e-02
1.0e+06	9.58e-14	9.58e-17	1.30e-11	7.06e+01	5.38e-01
1.0e+07	4.49e-13	1.42e-16	5.39e-06	1.08e+00	5.37e+00

TABLE 2.4
Experiment 2: LSTRS method with $\mathbf{B}_k$ is positive definite and $\|\mathbf{s}_u\|_2 > \Delta_k$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	1.32e-14	4.05e-16	6.25e-04	4.82e+01	4.44e-01
1.0e+04	1.20e-13	1.20e-15	1.20e-03	1.99e+01	4.80e-01
1.0e+05	5.45e-11	1.73e-13	4.90e-04	1.57e+01	7.30e-01
1.0e+06	4.68e-10	4.68e-13	1.35e-06	7.06e+01	4.56e+00
1.0e+07	4.15e-05	1.31e-08	4.47e-05	1.08e+00	4.21e+01

Tables 2.3 and 2.4 show the results of Experiment 2. In this case, the unconstrained minimizer is not inside the trust region, making the value of σ^* strictly positive. As in the first experiment, the OBS method appears to obtain solutions to higher accuracy (columns 1, 2, and 3) and in less time (column 4) than the LSTRS method. Finally, it is worth noting that as n increases, the accuracy of the solutions obtained by the LSTRS method appears to degrade.

Tables 2.5 and 2.6 display the results of Experiment 3(a). This experiment is the first of two in which $\mathbf{B}_k$ is highly ill-conditioned. In this experiment, the LSTRS method

TABLE 2.5
Experiment 3(a): OBS method with $\mathbf{B}_k$ is positive semidefinite and singular with $\|B^{\dagger}\mathbf{g}_k\|_2 > \Delta_k$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	2.80e-14	8.89e-16	6.25e-10	3.38e-01	2.70e-02
1.0e+04	1.17e-13	1.16e-15	1.18e-08	1.03e-01	3.36e-02
1.0e+05	3.48e-12	1.10e-14	2.16e-07	8.75e-03	6.43e-02
1.0e+06	1.44e-11	1.44e-14	1.48e-09	3.62e-03	5.44e-01
1.0e+07	5.52e-10	1.74e-13	8.96e-09	2.88e-03	5.39e+00

TABLE 2.6
Experiment 3(a): LSTRS method with $\mathbf{B}_k$ is positive semidefinite and singular with $\|\mathbf{B}_k^{\dagger}\mathbf{g}_k\|_2 > \Delta_k$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	9.75e-03	3.10e-04	1.51e-16	3.41e-01	4.78e-01
1.0e+04	7.93e-02	7.91e-04	2.65e-15	1.07e-01	5.69e-01
1.0e+05	1.85e-01	5.84e-04	8.16e-16	9.57e-03	1.56e+00
1.0e+06	1.29e-01	1.29e-04	6.04e-16	1.70e-03	1.28e+01
1.0e+07	2.24e+03	7.09e-01	1.05e-10	1.30e-06	6.39e+01

appears unable to obtain solutions to high absolute accuracy (see column 2 in Table 6). Moreover, the time required by the LSTRS to obtain solutions is, in some cases, significantly more than the time required by the OBS method. In contrast, the OBS method is able to obtain high accuracy solutions. Notice that the optimal values σ^* found by both methods appear to differ. Global solutions to the subproblems solved in Experiment 3(a) lie on the boundary of the trust region. Because LSTRS was able to satisfy the second optimality condition to high accuracy but not the first, this suggests LSTRS's solution $\mathbf{s}^*$ lies on the boundary but there is some error in this solution. As n increases, the solution quality of the LSTRS method appears to decline with significant error in the case of $n = 10^7$. In this experiment, the OBS method appears to find solutions to high accuracy in comparable time to other experiments; in contrast, the LSTRS method appears to have difficulty finding global solutions.

TABLE 2.7
Experiment 3(b): OBS method with $\mathbf{B}_k$ is positive semidefinite and singular with $\|\mathbf{B}_k^{\dagger}\mathbf{g}_k\|_2 \leq \Delta_k$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	4.10e-15	1.34e-16	9.05e-10	4.85e+01	3.01e-02
1.0e+04	1.01e-14	1.02e-16	1.34e-11	6.98e+00	4.36e-02
1.0e+05	3.03e-14	9.55e-17	7.99e-14	2.25e+01	6.70e-02
1.0e+06	1.39e-13	1.39e-16	4.18e-12	3.42e+00	5.41e-01
1.0e+07	3.46e-13	1.09e-16	1.28e-11	1.08e+00	5.37e+00

The results for Experiment 3(b) are shown in Tables 2.7 and 2.8. This is the second experiment involving ill-conditioned matrices. As with Experiment 3(a), the OBS method is able to obtain high-accuracy solutions in generally less time than the LSTRS

TABLE 2.8
Experiment 3(b): LSTRS method with $\mathbf{B}_k$ is positive semidefinite and singular with $\|\mathbf{B}_k^{\dagger}\mathbf{g}_k\|_2 \leq \Delta_k$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	9.40e-15	2.97e-16	8.19e-04	4.85e+01	4.42e-01
1.0e+04	2.06e-12	2.07e-14	6.59e-04	6.98e+00	4.79e-01
1.0e+05	1.69e-11	5.34e-14	4.27e-05	2.25e+01	7.43e-01
1.0e+06	6.27e-08	6.28e-11	6.19e-05	3.42e+00	4.60e+00
1.0e+07	4.28e-05	1.35e-08	2.59e-05	1.08e+00	6.29e+01

method. The accuracy obtained by the LSTRS method appears to degrade as the size of the problem increases. In this experiment, the global solution always lies on the boundary, but the larger residuals associated the second optimality condition in Table 8 indicate that the computed solutions by LSTRS do not lie on the boundary.

TABLE 2.9
Experiment 4(a): OBS method with $\mathbf{B}_k$ is indefinite with $\bar{\phi}(-\lambda_{\min}) < 0$. The vector $\mathbf{g}_k$ is randomly generated.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	2.83e-15	9.04e-17	3.57e-12	1.89e+02	3.05e-02
1.0e+04	1.27e-14	1.27e-16	1.53e-09	1.18e+02	3.99e-02
1.0e+05	3.42e-14	1.08e-16	9.15e-13	3.92e+02	6.40e-02
1.0e+06	1.19e-13	1.20e-16	4.79e-12	5.39e+03	5.43e-01
1.0e+07	3.46e-13	1.09e-16	8.18e-11	1.94e+04	5.35e+00

TABLE 2.10
Experiment 4(a): LSTRS method with $\mathbf{B}_k$ is indefinite with $\bar{\phi}(-\lambda_{\min}) < 0$. The vector $\mathbf{g}_k$ is randomly generated.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	4.92e-14	1.57e-15	5.40e-04	1.89e+02	4.40e-01
1.0e+04	2.82e-14	2.79e-16	1.03e-03	1.18e+02	4.80e-01
1.0e+05	2.11e-13	6.69e-16	2.68e-06	3.92e+02	7.24e-01
1.0e+06	2.93e-11	2.94e-14	1.38e-07	5.39e+03	4.49e+00
1.0e+07	1.81e-10	5.74e-14	3.19e-10	1.94e+04	4.12e+01

The results for Experiment 4(a) are displayed in Tables 2.9 and 2.10. Both methods found solutions that satisfied the first optimality conditions to high accuracy. The overall solution quality from the OBS method appears better in the sense that the residuals for both optimality conditions in Table 2.9 are smaller than the residuals for both optimality conditions in Table 2.10. Finally, the OBS method took less time to solve the subproblem than the LSTRS method.

The results of Experiment 4(b) are in Tables 2.11 and 2.12. Both methods solved the subproblem to high accuracy as measured by the first optimality condition; however, the OBS method solved the subproblem to significantly better accuracy as measured by

TABLE 2.11
Experiment 4(b): OBS method with $\mathbf{B}_k$ is indefinite with $\bar{\phi}(-\lambda_{\min}) < 0$. The vector $\mathbf{g}_k$ lies in the orthogonal complement of $\mathbf{P}_{\|_1}$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	3.42e-15	1.07e-16	1.17e-09	1.31e+01	2.91e-02
1.0e+04	1.38e-14	1.38e-16	1.50e-14	2.81e+00	3.16e-02
1.0e+05	3.17e-14	1.00e-16	3.55e-13	1.82e+01	6.66e-02
1.0e+06	1.30e-13	1.30e-16	1.76e-12	4.76e+00	5.46e-01
1.0e+07	3.14e-13	9.94e-17	4.36e-11	7.58e+01	5.36e+00

TABLE 2.12
Experiment 4(b): LSTRS method with $\mathbf{B}_k$ is indefinite with $\bar{\phi}(-\lambda_{\min}) < 0$. The vector $\mathbf{g}_k$ lies in the orthogonal complement of $\mathbf{P}_{\|_1}$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	1.16e-14	3.64e-16	1.24e-03	1.31e+01	4.42e-01
1.0e+04	2.48e-12	2.49e-14	1.02e-04	2.81e+00	4.70e-01
1.0e+05	1.50e-10	4.75e-13	2.82e-04	1.82e+01	7.30e-01
1.0e+06	1.65e-08	1.65e-11	9.70e-05	4.76e+00	4.65e+00
1.0e+07	2.08e-07	6.58e-11	1.06e-05	7.58e+01	4.21e+01

the second optimality condition than the LSTRS method. All residuals associated with the first and second optimality condition are less for the solution obtained by the OBS method. Moreover, the time required to find solutions was less for the OBS method.

TABLE 2.13
Experiment 5(a): The OBS method in the hard case ($\mathbf{B}_k$ is indefinite) and $\lambda_{\min} = \lambda_1 = \hat{\lambda}_1 + \gamma_{k-1} < 0$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	1.29e-14	4.34e-16	1.93e-16	4.35e-01	3.38e-02
1.0e+04	5.87e-14	5.86e-16	2.59e-14	6.08e-01	2.73e-02
1.0e+05	2.34e-12	7.43e-15	5.79e-14	8.15e+00	8.08e-02
1.0e+06	1.33e-11	1.33e-14	1.19e-12	3.97e+00	6.72e-01
1.0e+07	1.67e-10	5.28e-14	4.43e-12	5.27e-01	6.71e+00

TABLE 2.14
Experiment 5(a): The LSTRS method in the hard case ($\mathbf{B}_k$ is indefinite) and $\lambda_{\min} = \lambda_1 = \hat{\lambda}_1 + \gamma_{k-1} < 0$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	2.10e-05	7.07e-07	1.16e-15	4.35e-01	4.70e-01
1.0e+04	3.88e+00	3.87e-02	1.50e-03	6.08e-01	4.71e-01
1.0e+05	1.27e+02	4.01e-01	5.72e-04	8.15e+00	7.65e-01
1.0e+06	2.04e+02	2.04e-01	1.45e-04	3.97e+00	4.59e+00
1.0e+07	1.64e+03	5.17e-01	2.30e-05	5.27e-01	4.23e+01

In the hard case with $\lambda_{\min}$ being a nontrivial eigenvalue, the OBS method obtains global solutions to the subproblems; however, the LSTRS method had difficulty finding high-accuracy solutions for all problem sizes. In particular, as n increases, the solution quality of the LSTRS method appears to decline with significant error in the case of $n = 10^7$. In all cases, the time required by the OBS method to find a solution was less than that of the time required by the LSTRS method.

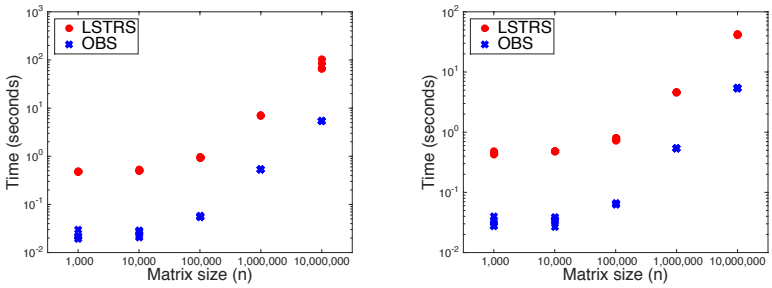
TABLE 2.15
Experiment 5(b): The OBS method in the hard case ($\mathbf{B}_k$ is indefinite) and $\lambda_{\min} = \gamma_{k-1} < 0$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	3.52e-15	1.11e-16	3.53e-09	6.35e+01	2.93e-02
1.0e+04	9.50e-15	9.48e-17	1.16e-14	2.10e+02	3.82e-02
1.0e+05	3.01e-14	9.50e-17	4.49e-13	4.49e+02	6.71e-02
1.0e+06	9.48e-14	9.47e-17	6.86e-12	1.34e+04	5.32e-01
1.0e+07	3.40e-13	1.07e-16	2.97e-12	8.91e+03	5.36e+00

TABLE 2.16
Experiment 5(b): The LSTRS method in the hard case ($\mathbf{B}_k$ is indefinite) and $\lambda_{\min} = \gamma_{k-1} < 0$.

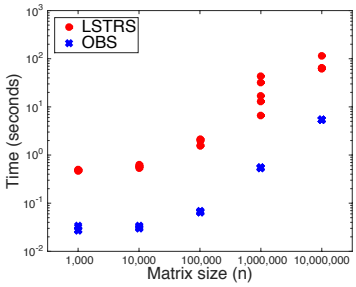
n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	2.24e-14	7.12e-16	7.36e-04	6.35e+01	4.41e-01
1.0e+04	6.35e-14	6.33e-16	1.92e-06	2.10e+02	5.02e-01
1.0e+05	2.26e-13	7.14e-16	5.09e-08	4.49e+02	7.49e-01
1.0e+06	6.61e-12	6.61e-15	4.76e-08	1.34e+04	4.32e+00
1.0e+07	8.77e-11	2.77e-14	1.05e-08	8.91e+03	4.09e+01

The results of Experiment 5(b) are in Tables 2.15 and 2.16. Unlike in Experiment 5(a), the LSTRS method was able to find solutions to high accuracy. In all cases, the OBS method was able to find solutions with higher accuracy than the LSTRS method and in less time.

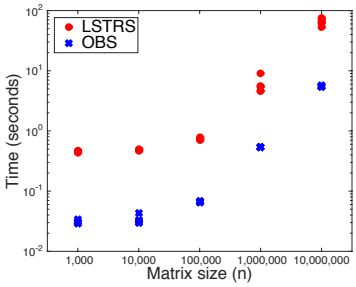


Experiment 1

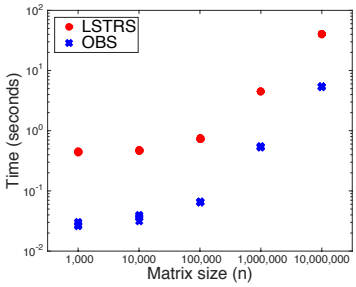
Experiment 2



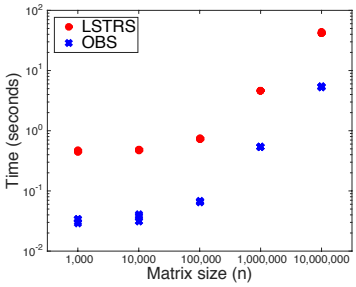
Experiment 3(a)



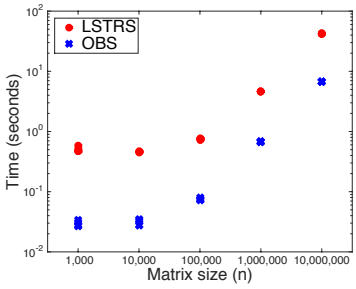
Experiment 3(b)



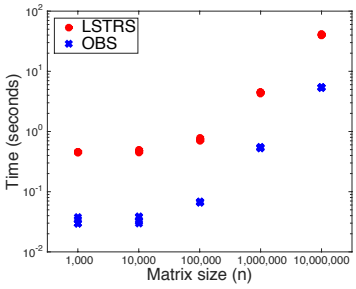
Experiment 4(a)



Experiment 4(b)



Experiment 5(a)



Experiment 5(b)

FIGURE 2.3 *Semi-log plots of the computational times (in seconds). Each experiment was run five times; computational time for the LSTRS and OBS method are shown for each run. In all cases, the OBS method outperforms LSTRS in terms of computational time.*

2.4.5 SUMMARY

In this section we presented the OBS method, which solves trust-region subproblems of the form (2.4) where $\mathbf{B}_k$ is a large L-SR1 matrix. The OBS method uses two main

strategies. In one strategy, σ^* is computed from Newton's method and initialized at a point where Newton's method is guaranteed to converge monotonically to σ^*. With σ^* in hand, $\mathbf{s}^*$ is computed directly by formula. For the other strategy, we propose a method that relies on an orthonormal basis to directly compute $\mathbf{s}^*$. (In this case, σ^* can be determined from the spectral decomposition of $\mathbf{B}_k$.) Numerical experiments suggest that the OBS method is able to solve large L-SR1 trust-region subproblems to high accuracy. Moreover, the method appears to be more robust than the LSTRS method, which does not exploit the specific structure of $\mathbf{B}_k$. In particular, the proposed OBS method achieves high accuracy in less time in all of the experiments and in all measures of optimality when compared to the LSTRS method. Future research will consider the best implementation of the OBS method in a trust-region method (see, for example, [BKS96]), including initialization of γ_{k-1} and rules for updating the matrices $\mathbf{S}_k$ and $\mathbf{Y}_k$ containing the quasi-Newton pairs.

2.5 THE SC-SR1 METHOD

2.5.1 MOTIVATION

In this section we propose a method that is very similar to the method from the previous section, except for one major difference. Instead of ℓ_2-norm trust-region subproblems, we analyze subproblems defined by *shape-changing* norms, which were originally described in [BY02]. Shape-changing norms are norms that depend on $\mathbf{B}_k$; thus, in the quasi-Newton setting where the quasi-Newton matrix $\mathbf{B}_k$ 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 [Gol80] 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 $\mathbf{B}_k$. Perhaps the most widely-used shape-changing norm is the so-called "elliptic norm" given by $\|\mathbf{x}\|_A \triangleq \mathbf{x}^T\mathbf{A}\mathbf{x}$, where $\mathbf{A}$ is a positive-definite matrix (see, e.g., [CGT00]). A well-known use of this norm is found in the Steihaug method [Ste83], and, more generally, truncated preconditioned conjugate-gradients (CG) [CGT00]; 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.

The shape-changing norms proposed in [BY02] 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 expression. 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 [BGYZ16] solves trust-region subproblems defined using these shape-changing norms when $\mathbf{B}_k$ in (2.1) is produced using L-BFGS updates. In this section, we propose solving the shape-changing

trust-region subproblem where $\mathbf{B}_k$ is obtained from L-SR1 updates. As in the previous section, we compute the subproblem solution on a case-by-case basis. In particular, we analyze the situations when $\mathbf{B}_k$ is postive definite, when $\mathbf{B}_k$ is singular, or when $\mathbf{B}_k$ is indefinite. What is different from the previous section, is that we apply the shape-changing norms instead of the ℓ_2-norm, when the trust-region subproblem solution lies at the boundary.

2.5.2 PROPOSED METHOD

The proposed method is able to solve the L-SR1 trust-region subproblem to high accuracy, even when $\mathbf{B}_k$ is indefinite. The method makes use of the eigenvalues of $\mathbf{B}_k$ and the factors of $\mathbf{P}_{\|}$. To describe the method, we first transform the trust-region subproblem (2.1) so that the quadratic objective function becomes separable. Then, we describe the shape-changing norms proposed in [BY02, BGYZ16] that decouple 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.

2.5.3 TRANSFORMING THE TRUST-REGION SUBPROBLEM

Let $\mathbf{B}_k = \mathbf{P}\Lambda\mathbf{P}^T$ be the eigendecomposition of $\mathbf{B}_k$ described in Section 1.4.3. Letting $\mathbf{v} = \mathbf{P}^T\mathbf{s}$ and $\mathbf{g_P} = \mathbf{P}^T\mathbf{g}_k$, the objective function $Q(\mathbf{s})$ in (2.1) can be written as a function of $\mathbf{v}$:

$$Q(\mathbf{s}) = \mathbf{g}_k^T\mathbf{s} + \frac{1}{2}\mathbf{s}^T\mathbf{B}_k\mathbf{s} = \mathbf{g}_\mathbf{P}^T\mathbf{v} + \frac{1}{2}\mathbf{v}^T\Lambda\mathbf{v} \equiv q(\mathbf{v}).$$

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

$$\mathbf{v} = \mathbf{P}^T\mathbf{s} = \begin{bmatrix} \mathbf{P}_{\|}^T\mathbf{s} \\ \mathbf{P}_{\perp}^T\mathbf{s} \end{bmatrix} = \begin{bmatrix} \mathbf{v}_{\|} \\ \mathbf{v}_{\perp} \end{bmatrix} \quad \text{and} \quad \mathbf{g_P} = \begin{bmatrix} \mathbf{P}_{\|}^T\mathbf{g}_k \\ \mathbf{P}_{\perp}^T\mathbf{g}_k \end{bmatrix} = \begin{bmatrix} \mathbf{g}_{\|} \\ \mathbf{g}_{\perp} \end{bmatrix},$$

where $\mathbf{v}_{\|}, \mathbf{g}_{\|} \in \mathbb{R}^l$ and $\mathbf{v}_{\perp}, \mathbf{g}_{\perp} \in \mathbb{R}^{n-l}$. Then,

$$\begin{aligned} q(\mathbf{v}) &= \begin{bmatrix} \mathbf{g}_{\|}^T & \mathbf{g}_{\perp}^T \end{bmatrix} \begin{bmatrix} \mathbf{v}_{\|} \\ \mathbf{v}_{\perp} \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \mathbf{v}_{\|}^T & \mathbf{v}_{\perp}^T \end{bmatrix} \begin{bmatrix} \Lambda_1 & \\ & \gamma_{k-1}\mathbf{I}_{n-l} \end{bmatrix} \begin{bmatrix} \mathbf{v}_{\|} \\ \mathbf{v}_{\perp} \end{bmatrix} \\ &= \mathbf{g}_{\|}^T\mathbf{v}_{\|} + \mathbf{g}_{\perp}^T\mathbf{v}_{\perp} + \frac{1}{2}\left(\mathbf{v}_{\|}^T\Lambda_1\mathbf{v}_{\|} + \gamma_{k-1}\|\mathbf{v}_{\perp}\|_2^2\right) \\ &= q_{\|}(\mathbf{v}_{\|}) + q_{\perp}(\mathbf{v}_{\perp}), \end{aligned} \tag{2.16}$$

where

$$q_{\|}(\mathbf{v}_{\|}) \equiv \mathbf{g}_{\|}^T\mathbf{v}_{\|} + \frac{1}{2}\mathbf{v}_{\|}^T\Lambda_1\mathbf{v}_{\|} \quad \text{and} \quad q_{\perp}(\mathbf{v}_{\perp}) \equiv \mathbf{g}_{\perp}^T\mathbf{v}_{\perp} + \frac{\gamma_{k-1}}{2}\|\mathbf{v}_{\perp}\|_2^2.$$

Thus, the trust-region subproblem (2.1) can be expressed as

$$\underset{\|\mathbf{P}\mathbf{v}\| \leq \Delta_k}{\text{minimize}} \quad q(\mathbf{v}) = q_{\|}\left(\mathbf{v}_{\|}\right) + q_{\perp}\left(\mathbf{v}_{\perp}\right). \tag{2.17}$$

Note that the function $q(\mathbf{v})$ is now separable in $\mathbf{v}_{\|}$ and $\mathbf{v}_{\perp}$. To completely decouple (2.17) into two minimization problems, we use a shape-changing norm so that the norm constraint $\|\mathbf{P}\mathbf{v}\| \leq \Delta_k$ decouples into separate constraints, one involving $\mathbf{v}_{\|}$ and the other involving $\mathbf{v}_{\perp}$.

2.5.4 SHAPE-CHANGING NORMS

Consider the following shape-changing norms proposed in [BY02, BGYZ16]:

$$\|\mathbf{s}\|_{\mathbf{P},2} \triangleq \max\left(\|\mathbf{P}_{\|}^T\mathbf{s}\|_2, \|\mathbf{P}_{\perp}^T\mathbf{s}\|_2\right) = \max\left(\|\mathbf{v}_{\|}\|_2, \|\mathbf{v}_{\perp}\|_2\right), \tag{2.18}$$

$$\|\mathbf{s}\|_{\mathbf{P},\infty} \triangleq \max\left(\|\mathbf{P}_{\|}^T\mathbf{s}\|_{\infty}, \|\mathbf{P}_{\perp}^T\mathbf{s}\|_2\right) = \max\left(\|\mathbf{v}_{\|}\|_{\infty}, \|\mathbf{v}_{\perp}\|_2\right). \tag{2.19}$$

We refer to them as the $(\mathbf{P}, 2)$ and the $(\mathbf{P}, \infty)$ norms, respectively. Since $\mathbf{s} = \mathbf{P}\mathbf{v}$, the trust-region constraint in (2.17) can be expressed in these norms as

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

Thus, from (2.17), the trust-region subproblem is given for the $(\mathbf{P}, 2)$ norm by

$$\underset{\|\mathbf{P}\mathbf{v}\|_{\mathbf{P},2} \leq \Delta_k}{\text{minimize}} \; q(\mathbf{v}) = \underset{\|\mathbf{v}_{\|}\|_2 \leq \Delta_k}{\text{minimize}} \; q_{\|}\left(\mathbf{v}_{\|}\right) + \underset{\|\mathbf{v}_{\perp}\|_2 \leq \Delta_k}{\text{minimize}} \; q_{\perp}\left(\mathbf{v}_{\perp}\right), \tag{2.20}$$

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

$$\underset{\|\mathbf{P}\mathbf{v}\|_{\mathbf{P},\infty} \leq \Delta_k}{\text{minimize}} \; q(\mathbf{v}) = \underset{\|\mathbf{v}_{\|}\|_{\infty} \leq \Delta_k}{\text{minimize}} \; q_{\|}\left(\mathbf{v}_{\|}\right) + \underset{\|\mathbf{v}_{\perp}\|_2 \leq \Delta_k}{\text{minimize}} \; q_{\perp}\left(\mathbf{v}_{\perp}\right). \tag{2.21}$$

As shown in [BGYZ16], these norms are equivalent to the two-norm, i.e.,

$$\begin{aligned}
\frac{1}{\sqrt{2}}\|\mathbf{s}\|_2 \;&\leq\; \|\mathbf{s}\|_{\mathbf{P},2} \;\leq\; \|\mathbf{s}\|_2 \\
\frac{1}{\sqrt{l}}\|\mathbf{s}\|_2 \;&\leq\; \|\mathbf{s}\|_{\mathbf{P},\infty} \;\leq\; \|\mathbf{s}\|_2.
\end{aligned}$$

Note that the latter equivalence factor depends on the number of stored quasi-Newton pairs l and not on the number of variables (n).

Notice that the shape-changing norms do not place equal value on the two subspaces 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.

2.5.5 SOLVING FOR THE OPTIMAL $\mathbf{v}_\perp^*$

The subproblem

$$\underset{\|\mathbf{v}_\perp\|_2 \leq \Delta_k}{\text{minimize}} \quad q_\perp(\mathbf{v}_\perp) \equiv \mathbf{g}_\perp^T \mathbf{v}_\perp + \frac{\gamma_{k-1}}{2}\|\mathbf{v}_\perp\|_2^2 \tag{2.22}$$

appears in both (2.20) and (2.21); its optimal solution can be computed by formula. For the quadratic subproblem (2.22) the solution $\mathbf{v}_\perp^*$ must satisfy the following optimality conditions adapted from [Gay81, MS83, Sor82] associated with (2.22): For some $\sigma_\perp^* \in \mathbb{R}^+$,

$$(\gamma_{k-1} + \sigma_\perp^*)\,\mathbf{v}_\perp^* = -\mathbf{g}_\perp, \tag{2.23a}$$

$$\sigma_\perp^*\,(\|\mathbf{v}_\perp^*\|_2 - \Delta_k) = 0, \tag{2.23b}$$

$$\|\mathbf{v}_\perp^*\|_2 \leq \Delta_k, \tag{2.23c}$$

$$\gamma_{k-1} + \sigma_\perp^* \geq 0. \tag{2.23d}$$

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

$$\mathbf{v}_\perp^* = \begin{cases} -\frac{1}{\gamma_{k-1}}\mathbf{g}_\perp & \text{if } \gamma_{k-1} > 0 \text{ and } \|\mathbf{g}_\perp\|_2 \leq \Delta_k|\gamma_{k-1}|, \\ \Delta_k \mathbf{u} & \text{if } \gamma_{k-1} \leq 0 \text{ and } \|\mathbf{g}_\perp\|_2 = 0, \\ -\frac{\Delta_k}{\|\mathbf{g}_\perp\|_2}\mathbf{g}_\perp & \text{otherwise,} \end{cases} \tag{2.24}$$

and

$$\sigma_\perp^* = \begin{cases} 0 & \text{if } \gamma_{k-1} > 0 \text{ and } \|\mathbf{g}_\perp\|_2 \leq \Delta_k|\gamma_{k-1}|, \\ \frac{\|\mathbf{g}_\perp\|_2}{\Delta_k} - \gamma_{k-1} & \text{otherwise,} \end{cases} \tag{2.25}$$

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

2.5.6 SOLVING FOR THE OPTIMAL $\mathbf{v}_\|^*$

In this subsection, we detail how to solve for the optimal $\mathbf{v}_\|^*$ 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 (2.17), then the subproblem in $\mathbf{v}_\|$ is

$$\underset{\|\mathbf{v}_\|\|_\infty \leq \Delta_k}{\text{minimize}} \quad q_\|(\mathbf{v}_\|) = \mathbf{g}_\|^T \mathbf{v}_\| + \frac{1}{2}\mathbf{v}_\|^T \Lambda_1 \mathbf{v}_\|. \tag{2.26}$$

The solution to this problem is computed by separately minimizing l scalar quadratic

problems of the form

$$\underset{|[\mathbf{v}_{\|}]_i| \leq \Delta_k}{\text{minimize}} \quad q_{\|,i}([\mathbf{v}_{\|}]_i) = \left[\mathbf{g}_{\|}\right]_i \left[\mathbf{v}_{\|}\right]_i + \frac{\lambda_i}{2}\left(\left[\mathbf{v}_{\|}\right]_i\right)^2, \qquad 1 \leq i \leq l. \tag{2.27}$$

The minimizer depends on the convexity of $q_{\|,i}$, i.e., the sign of λ_i. The solution to (2.27) is given as follows:

$$[\mathbf{v}_{||}^*]_i = \begin{cases} -\frac{\left[\mathbf{g}_{||}\right]_i}{\lambda_i} & \text{if } \left|\frac{\left[\mathbf{g}_{||}\right]_i}{\lambda_i}\right| \leq \Delta_k \text{ and } \lambda_i > 0, \\ c & \text{if } \left[\mathbf{g}_{\|}\right]_i = 0,\ \lambda_i = 0, \\ -\text{sgn}(\left[\mathbf{g}_{\|}\right]_i)\Delta_k & \text{if } \left[\mathbf{g}_{\|}\right]_i \neq 0,\ \lambda_i = 0, \\ \pm\Delta_k & \text{if } \left[\mathbf{g}_{\|}\right]_i = 0,\ \lambda_i < 0, \\ -\frac{\Delta_k}{\left|\left[\mathbf{g}_{||}\right]_i\right|}\left[\mathbf{g}_{||}\right]_i & \text{otherwise,} \end{cases} \tag{2.28}$$

where c is any real number in $[-\Delta_k, \Delta_k]$ and "sgn" denotes the signum function (see [BGYZ16] for details).

$(\mathbf{P}, 2)$-norm solution: If the shape-changing $(\mathbf{P}, 2)$-norm is used in (2.17), then the subproblem in $\mathbf{v}_{\|}$ is

$$\underset{\|\mathbf{v}_{\|}\|_2 \leq \Delta_k}{\text{minimize}} \quad q_{\|}\left(\mathbf{v}_{\|}\right) = \mathbf{g}_{\|}^T \mathbf{v}_{\|} + \frac{1}{2}\mathbf{v}_{\|}^T \Lambda_1 \mathbf{v}_{\|}. \tag{2.29}$$

The solution $\mathbf{v}_{\|}^*$ must satisfy the following optimality conditions [Gay81, MS83, Sor82] associated with (2.29): For some $\sigma_{\|}^* \in \mathbb{R}^+$,

$$(\Lambda_1 + \sigma_{\|}^* \mathbf{I})\mathbf{v}_{||}^* = -\mathbf{g}_{||}, \tag{2.30a}$$

$$\sigma_{\|}^*\left(\|\mathbf{v}_{||}^*\|_2 - \Delta_k\right) = 0, \tag{2.30b}$$

$$\|\mathbf{v}_{||}^*\|_2 \leq \Delta_k, \tag{2.30c}$$

$$\lambda_i + \sigma_{\|}^* \geq 0 \quad \text{for } 1 \leq i \leq l. \tag{2.30d}$$

A solution to the optimality conditions (2.30a)-(2.30d) can be computed using the method found in [BEM17]. 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 (2.30a), we write

$$\mathbf{v}_{\|}\left(\sigma_{\|}\right) = -\left(\Lambda_1 + \sigma_{\|}\mathbf{I}\right)^{-1}\mathbf{g}_{\|}, \tag{2.31}$$

with $\sigma_{\|} \neq -\lambda_i$ for $1 \leq i \leq l$.

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

$$\mathbf{v}_\parallel(0) = -\Lambda_1^{-1}\mathbf{g}_\parallel. \tag{2.32}$$

If $\mathbf{v}_\parallel(0)$ is feasible, i.e., $\|\mathbf{v}_\parallel(0)\|_2 \leq \Delta_k$ then $\mathbf{v}_\parallel^* = \mathbf{v}_\parallel(0)$ is the global minimizer; otherwise, $\sigma_\parallel^*$ is the solution to the secular equation (2.36) (discussed below). The minimizer to the problem (2.29) is then given by

$$\mathbf{v}_\parallel^* = -\left(\Lambda_1 + \sigma_\parallel^*\mathbf{I}\right)^{-1}\mathbf{g}_\parallel. \tag{2.33}$$

Case 2 ($\lambda_1 = 0$). If $\mathbf{g}_\parallel$ is in the range of Λ_1, i.e., $[g_\parallel]_i = 0$ for $1 \leq i \leq r$, then set $\sigma_\parallel = 0$ and let

$$\mathbf{v}_\parallel(0) = -\Lambda_1^\dagger \mathbf{g}_\parallel,$$

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

$$\mathbf{v}_\parallel^* = \mathbf{v}_\parallel(0) = -\Lambda_1^\dagger \mathbf{g}_\parallel$$

satisfies all optimality conditions (with $\sigma_\parallel^* = 0$). Otherwise, i.e., if either $[\mathbf{g}_\parallel]_i \neq 0$ for some $1 \leq i \leq r$ or $\|\Lambda_1^\dagger \mathbf{g}_\parallel\|_2 > \Delta_k$, then $\mathbf{v}_\parallel^*$ is computed using (2.33), where $\sigma_\parallel^*$ solves the secular equation in (2.36) (discussed below).

Case 3 ($\lambda_1 < 0$): If $\mathbf{g}_\parallel$ is in the range of $\Lambda_1 - \lambda_1\mathbf{I}$, i.e., $[g_\parallel]_i = 0$ for $1 \leq i \leq r$, then we set $\sigma_\parallel = -\lambda_1$ and

$$\mathbf{v}_\parallel(-\lambda_1) = -\left(\Lambda_1 - \lambda_1\mathbf{I}\right)^\dagger \mathbf{g}_\parallel.$$

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

$$\mathbf{v}_\parallel^* = \mathbf{v}_\parallel(-\lambda_1) + \alpha\mathbf{e}_1, \tag{2.34}$$

where $\alpha = \sqrt{\Delta_k^2 - \left\|\mathbf{v}_\parallel(-\lambda_1)\right\|_2^2}$. (This case is referred to as the "hard case" [CGT00, MS83].) Note that $v_\parallel^*$ satisfies the first optimality condition (2.30a):

$$\left(\Lambda_1 - \lambda_1\mathbf{I}\right)\mathbf{v}_\parallel^* = \left(\Lambda_1 - \lambda_1\mathbf{I}\right)\left(\mathbf{v}_\parallel(-\lambda_1) + \alpha\mathbf{e}_1\right) = -\mathbf{g}_\parallel.$$

The second optimality condition (2.30b) is satisfied by observing that

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

Finally, since $\sigma_\parallel^* = -\lambda_1 > 0$ 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 $\|(\Lambda_1 - \lambda_1\mathbf{I})^\dagger \mathbf{g}_\parallel\|_2 > \Delta_k$, then $\mathbf{v}_\parallel^*$ is computed using (2.33), where $\sigma_\parallel^*$ solves the secular equation (2.36).

The secular equation. We now summarize how to find a solution of the so-called *secular equation*. Note that from (2.31),

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

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

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

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

$$\phi_{\|}\left(\sigma_{\|}\right) = \begin{cases} \dfrac{1}{\sqrt{\displaystyle\sum_{i=1}^{L} \frac{\bar{a}_i^2}{(\bar{\lambda}_i + \sigma_{\|})^2}}} - \dfrac{1}{\Delta_k} & \text{if } \sigma_{\|} \neq -\bar{\lambda}_i \text{ where } 1 \leq i \leq L \\ -\dfrac{1}{\Delta_k} & \text{otherwise.} \end{cases} \tag{2.35}$$

From the optimality conditions (2.30b) and (2.30d), if $\sigma_{\|}^* \neq 0$, then $\sigma_{\|}^*$ solves the *secular equation*

$$\phi_{\|}\left(\sigma_{\|}^*\right) = 0, \tag{2.36}$$

with $\sigma_{\|}^* \geq \max\{0, -\lambda_1\}$. Note that $\phi_{\|}$ is monotonically increasing and concave on the interval $[-\lambda_1, \infty)$; thus, with a judicious choice of initial $\sigma_{\|}^0$, Newton's method can be used to efficiently compute $\sigma_{\|}^*$ in (2.36) (see [BEM17]).

The details on the solution method for subproblem (2.29) are as described in Subsection 2.4.2 from the previous Section 2.4.

2.5.7 COMPUTING $\mathbf{s}^*$

Given $\mathbf{v}^* = [\ \mathbf{v}_{\|}^*\ \ \mathbf{v}_{\perp}^*\ ]^T$, the solution to the trust-region subproblem (2.1) using either the $(\mathbf{P}, 2)$ or the $(\mathbf{P}, \infty)$ norms is

$$\mathbf{s}^* = \mathbf{P}\mathbf{v}^* = \mathbf{P}_{\|}\mathbf{v}_{\|}^* + \mathbf{P}_{\perp}\mathbf{v}_{\perp}^*. \tag{2.37}$$

(Recall that using either of the two norms generates the same $\mathbf{v}_{\perp}^*$ but different $\mathbf{v}_{\|}^*$.) It remains to show how to form $\mathbf{s}^*$ in (2.37). Matrix-vector products involving $\mathbf{P}_{\|}$ are possible using (1.11), and thus, $\mathbf{P}_{\|}\mathbf{v}_{\|}^*$ can be computed; however, an explicit formula to compute products with $\mathbf{P}_{\perp}$ is not available. To compute the second term, $\mathbf{P}_{\perp}\mathbf{v}_{\perp}^*$, we

observe that $\mathbf{v}_\perp^*$, as given in (2.24), is a multiple of either $\mathbf{g}_\perp = \mathbf{P}_\perp^T \mathbf{g}_k$ or a vector $\mathbf{u}$ with unit length, depending on the sign of γ_{k-1} 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 $i \in \{1, 2, \ldots, l+1\}$ 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 $\operatorname{rank}(\mathbf{P}_\perp) = n - l$.) Thus, we obtain

$$\mathbf{s}^* = \mathbf{P}_\parallel \mathbf{v}_\parallel^* + \left(\mathbf{I} - \mathbf{P}_\parallel \mathbf{P}_\parallel^T\right) \mathbf{w}^*, \tag{2.38}$$

where

$$\mathbf{w}^* = \begin{cases} -\frac{1}{\gamma_{k-1}} \mathbf{g}_k & \text{if } \gamma_{k-1} > 0 \text{ and } \|\mathbf{g}_\perp\|_2 \leq \Delta_k |\gamma_{k-1}|, \\ \frac{\Delta_k}{\left\|\mathbf{P}_\perp^T \mathbf{e}_i\right\|_2} \mathbf{e}_i & \text{if } \gamma_{k-1} \leq 0 \text{ and } \|\mathbf{g}_\perp\|_2 = 0, \\ -\frac{\Delta_k}{\|\mathbf{g}_\perp\|_2} \mathbf{g}_k & \text{otherwise.} \end{cases} \tag{2.39}$$

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

$$\left\|\mathbf{g}_\parallel\right\|_2^2 + \|\mathbf{g}_\perp\|_2^2 = \|\mathbf{g}_k\|_2^2, \text{ and } \|\mathbf{P}_\parallel^T \mathbf{e}_i\|_2^2 + \|\mathbf{P}_\perp^T \mathbf{e}_i\|_2^2 = 1. \tag{2.40}$$

Then $\|\mathbf{g}_\perp\|_2 = \sqrt{\|\mathbf{g}_k\|_2^2 - \|\mathbf{g}_\parallel\|_2^2}$ 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.

2.5.8 COMPUTATIONAL COMPLEXITY

We estimate the cost of one iteration using the proposed method to solve the trust-region subproblem defined by shape-changing norms (2.18) and (2.19). We make the practical assumption that $\gamma_{k-1} > 0$.

Theorem 2.4. *The dominant computational cost of solving one trust-region subproblem for the proposed method is $4ln$ 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 $\mathbf{\Psi}_k = \mathbf{Y}_k - \gamma_{k-1} \mathbf{S}_k$ of dimensions $\mathbf{\Psi}_k \in \mathbb{R}^{n \times l}$ explicitly. Rather, we compute matrix-vector products with $\mathbf{\Psi}_k$ by computing matrix-vector products with $\mathbf{Y}_k$ and $\mathbf{S}_k$. Second, to form $\mathbf{\Psi}_k^T \mathbf{\Psi}_k$, we only store and update the small $l \times l$ matrices $\mathbf{Y}_k^T \mathbf{Y}_k$, $\mathbf{S}_k^T \mathbf{Y}_k$, and $\mathbf{S}_k^T \mathbf{S}_k$. This update involves $3ln$ vector inner products. Third, assuming we have already obtained the Cholesky factorization of $\mathbf{\Psi}_k^T \mathbf{\Psi}_k$ associated with the previously-stored limited-memory pairs, it is possible to update the Cholesky factorization of the new $\mathbf{\Psi}_k^T \mathbf{\Psi}_k$ at a cost of $\mathcal{O}(l^2)$ [Ben65, GGMS74].

We now consider the dominant cost for a single subproblem solve. The eigendecomposition $\mathbf{R}\mathbf{M}_k\mathbf{R}^T = \mathbf{U}\hat{\mathbf{\Lambda}}\mathbf{U}^T$ costs $\mathcal{O}(l^3) = \left(\frac{l^2}{n}\right)(ln)$, where $l \ll n$. To compute $\mathbf{s}^*$ in (2.38), one needs to compute $\mathbf{v}^*$ from Subsection 2.5.6 and $\mathbf{w}^*$ from (2.39). The dominant cost for computing $\mathbf{v}^*$ and $\mathbf{w}^*$ is forming $\mathbf{\Psi}^T \mathbf{g}_k$, which requires $2ln$ operations. (In practice, this quantity is computed while solving the previous trust-region subprob-

lem and can be stored to avoid recomputing when solving the current subproblem–see [BGYZ16] for details.) Note that given $\mathbf{P}_{\|}^T \mathbf{g}_k$, the computation of $\mathbf{s}^*$ in (2.38) costs $2ln + 2ln = 4ln$. Thus, the dominant term in the total number of floating point operations is $4ln$.

We note that the floating point operation count of $O(4ln)$ is the same cost as for L-BFGS [Noc80].

2.5.9 CHARACTERIZATION OF GLOBAL SOLUTIONS

We provide a result on how to characterize global solutions to the trust-region subproblem defined by shape-changing norm $(\mathbf{P}, 2)$-norm. The following theorem is adapted from well-known optimality conditions for the two-norm trust-region subproblem [Gay81, MS83].

Theorem 2.5. *A vector* $\mathbf{s}^* \in \mathbb{R}^n$ *such that* $\left\|\mathbf{P}_{\|}^T \mathbf{s}^*\right\|_2 \leq \Delta_k$ *and* $\left\|\mathbf{P}_{\perp}^T \mathbf{s}^*\right\|_2 \leq \Delta_k$, *is a global solution of (2.1) defined by the* $(\mathbf{P}, 2)$*-norm if and only if there exists unique* $\sigma_{\|}^* \geq 0$ *and* $\sigma_{\perp}^* \geq 0$ *such that*

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

where $\mathbf{C}_{\|} \equiv \sigma_{\perp}^* \mathbf{I} + \left(\sigma_{\|}^* - \sigma_{\perp}^*\right) \mathbf{P}_{\|} \mathbf{P}_{\|}^T$, *the matrix* $\mathbf{B}_k + \mathbf{C}_{\|}$ *is positive semi-definite, and* $\mathbf{P} = \begin{bmatrix} \mathbf{P}_{\|} & \mathbf{P}_{\perp} \end{bmatrix}$ *and* $\Lambda_1 = diag(\lambda_1, \ldots, \lambda_{k+1}) = \hat{\Lambda} + \gamma_{k-1}\mathbf{I}$ *are as in (1.10).*

2.5.10 NUMERICAL EXPERIMENTS

In this subsection, we report 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 $n = 10^6$. We report 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 (2.28).) The six experiments are outlined as follows:

(E1) $\mathbf{B}_k$ is positive definite with $\|\mathbf{v}_{\|}(0)\|_2 \geq \Delta_k$.

(E2) $\mathbf{B}_k$ is positive semidefinite and singular with $[\mathbf{g}_{\|}]_i \neq 0$ for some $1 \leq i \leq r$.

(E3) $\mathbf{B}_k$ is positive semidefinite and singular with $[\mathbf{g}_{\|}]_i = 0$ for $1 \leq i \leq r$ and $\|\Lambda^{\dagger} \mathbf{g}_{\|}\|_2 > \Delta_k$.

(E4) $\mathbf{B}_k$ is indefinite and $[\mathbf{g}_{\|}]_i = 0$ for $1 \leq i \leq r$ with $\|(\Lambda - \lambda_1 \mathbf{I})^{\dagger} \mathbf{g}_{\|}\|_2 > \Delta_k$.

(E5) $\mathbf{B}_k$ is indefinite and $[\mathbf{g}_\parallel]_i \neq 0$ for some $1 \leq i \leq r$.

(E6) $\mathbf{B}_k$ is indefinite and $[\mathbf{g}_\parallel]_i = 0$ for $1 \leq i \leq r$ with $\|\mathbf{v}_\parallel(-\lambda_1)\|_2 \leq \Delta_k$ (the "hard case").

For these experiments, $\mathbf{S}_k$, $\mathbf{Y}_k$, and $\mathbf{g}_k$ 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 $\mathbf{B}_0 = \gamma_{k-1}\mathbf{I}$, where $\gamma_{k-1} = |10 * \texttt{randn(1)}|$. Finally, the number of limited-memory updates l 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 $\phi_\parallel$. We use the same procedure as in [BEM17, 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 $|\phi_\parallel(\sigma^i)| \leq \texttt{eps} \cdot |\phi_\parallel(\sigma^0)| + \sqrt{\texttt{eps}}$, 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 2.5. For each experiment, we report the following: (i) the norm of the residual of the first optimality condition, `opt 1` $\triangleq \|(\mathbf{B}_k + \mathbf{C}_\parallel)\mathbf{s}^* + \mathbf{g}_k\|_2$, (ii) the combined complementarity condition, `opt 2` $\triangleq |\sigma_\parallel^*(\|\mathbf{P}_\parallel^T\mathbf{s}^*\|_2 - \Delta_k)| + |\sigma_\perp^*(\|\mathbf{P}_\perp^T\mathbf{s}^*\|_2 - \Delta_k)|$, (iii) $\sigma_\parallel^* + \lambda_1$, (iv) $\sigma_\perp^* + \gamma_{k-1}$, (v) $\sigma_\parallel^*$, (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 $\mathbf{B}_k + \mathbf{C}_\parallel$ is a positive semidefinite matrix is equivalent to $\gamma_{k-1} + \sigma_\perp^* \geq 0$ and $\lambda_i + \sigma_\parallel^* \geq 0$, for $1 \leq i \leq l$. Finally, we ran each experiment five times and report one representative result for each experiment.

n	`opt 1`	`opt 2`	$\sigma_\parallel^* + \lambda_1$	$\sigma_\perp^* + \gamma_{k-1}$	$\sigma_\parallel^*$	$\sigma_\perp^*$	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 2.17
Experiment 1: $\mathbf{B}$ *is positive definite with* $\|\mathbf{v}_\parallel(0)\|_2 \geq \Delta_k$.

n	`opt 1`	`opt 2`	$\sigma_\parallel^* + \lambda_1$	$\sigma_\perp^* + \gamma_{k-1}$	$\sigma_\parallel^*$	$\sigma_\perp^*$	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 2.18
Experiment 2: $\mathbf{B}$ *is positive semidefinite and singular and* $[\mathbf{g}_\parallel]_i \neq 0$ *for some* $1 \leq i \leq r$.

n	opt 1	opt 2	$\sigma_{\|}^* + \lambda_1$	$\sigma_{\perp}^* + \gamma_{k-1}$	$\sigma_{\|}^*$	$\sigma_{\perp}^*$	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 2.19

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

n	opt 1	opt 2	$\sigma_{\|}^* + \lambda_1$	$\sigma_{\perp}^* + \gamma_{k-1}$	$\sigma_{\|}^*$	$\sigma_{\perp}^*$	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

TABLE 2.20

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

Tables 2.17-2.22 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 1×10^{-10} or smaller. Columns 4 and 5 in all the tables show that $\sigma_{\|}^* + \lambda_1$ and $\sigma_{\perp}^* + \gamma_{k-1}$ 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 $\sigma_{\|}^* = -\lambda_1$. This is reflected in Table 2.22, where Column 4 shows that $\sigma_{\|}^* = -\lambda_1$ 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 2.5.8.

Additional experiments were run with $\mathbf{g}_{\|} \to 0$. 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 2.17-2.22: the optimality conditions were satisfied to high accuracy, no more than three Newton iterations were required in any experiment to find $\sigma_{\|}^*$, and the CPU times are similar to those found in the tables.

n	opt 1	opt 2	$\sigma_{\|}^* + \lambda_1$	$\sigma_{\perp}^* + \gamma_{k-1}$	$\sigma_{\|}^*$	$\sigma_{\perp}^*$	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 2.21

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

n	opt 1	opt 2	$\sigma_{\|}^* + \lambda_1$	$\sigma_{\perp}^* + \gamma_{k-1}$	$\sigma_{\|}^*$	$\sigma_{\perp}^*$	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

TABLE 2.22

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

2.6 SUMMARY

In this chapter we proposed two methods to solve large-scale trust-region subproblems (2.1), when the quadratic objective function, $Q(\mathbf{s})$, is not convex. In particular, we use the compact representation of the indefinite SR1 quasi-Newton matrix to define the trust-region subproblems. SR1 matrices have desirable convergence properties, as they tend to approximate the true hessian matrix comparatively well. However since the SR1 matrix is not guaranteed to be positive definite, the solution of the trust-region subproblems requires a case-by-case analysis.

The first method we proposed is the Orthonormal Basis SR1 method (OBS), which solves the trust-region subproblem when the constraint norm is the ℓ_2 norm. This approach first computes the unconstrained minimizer of the L-SR1 matrix. If the L-SR1 matrix is positive definite and the unconstrained minimizer is feasible, then the OBS method terminates. Otherwise, it calculates a suitable Lagrange multiplier σ^* using the partial eigendecomposition and a one-dimensional Newton's method (Algorithm 2.2) and then calculates the global solution $\mathbf{s}^*$ accordingly. We also address the so-called hard case, where we define the solution by formula.

Secondly, we proposed the SC-SR1 method. In this method the trust-region subproblem is defined by so-called shape-changing norms. Based on the partial eigendecomposition of L-SR1 matrices, the shape-changing norms decouple the trust-region subproblem into two simpler problems. With one of the shape-changing norms analytic subproblem solutions are computed, while with the other norm only a small ℓ_2 trust-region subproblem needs to be solved. We also propose formulas for the solution of the hard-case, and derive the subproblem optimality conditions of the shape-changing norm that does not compute analytic solutions.

CHAPTER 3

THE DENSE INITIAL MATRIX TRUST-REGION METHOD

This chapter is based on the manuscript "*Dense Initializations for Limited-Memory Quasi-Newton Methods*", J. J. Brust, O. P. Burdakov, J. B. Erway, and R. F. Marcia, which is submitted to the *SIAM Journal on Optimization*.

3.1 MOTIVATION

In this chapter we propose a new dense initialization for quasi-Newton methods to solve problems of the form

$$\underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimize}}\ f(\mathbf{x}),$$

where $f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$ is a general nonconvex function that is at least continuously differentiable. The dense initialization matrix is designed to be updated each time a new quasi-Newton pair is computed (i.e., as often as once an iteration); however, in order to retain the efficiency of limited-memory quasi-Newton methods, the dense initialization matrix and the generated sequence of quasi-Newton matrices are not explicitly formed. This proposed initialization makes use of an eigendecomposition-based separation of $\mathbb{R}^n$ into two orthogonal subspaces – one for which there is approximate curvature information and the other for which there is no reliable curvature information. A different scaling parameter may then be used for each subspace. This initialization has broad applications for general quasi-Newton trust-region and line search methods. In fact, this work can be applied to any quasi-Newton method that uses an update with a compact representation, which includes any member of the Broyden class of updates. In this chapter, we explore its use in one specific application; in particular we consider a limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) trust-region method where each subproblem is defined using a shape-changing norm [BGYZ16]. The reason for this choice is that the dense initialization is naturally well-suited for solving L-BFGS trust-

region subproblems defined by this norm. Numerical results on the CUTEst test set suggest that the dense initialization outperforms other L-BFGS methods.

The L-BFGS update is the most widely-used quasi-Newton update for large-scale optimization; it is defined by the recursion formula

$$\mathbf{B}_{k+1} = \mathbf{B}_k - \frac{1}{\mathbf{s}_k^T \mathbf{B}_k \mathbf{s}_k} \mathbf{B}_k \mathbf{s}_k \mathbf{s}_k^T \mathbf{B}_k + \frac{1}{\mathbf{s}_k^T \mathbf{y}_k} \mathbf{y}_k \mathbf{y}_k^T, \tag{3.1}$$

where

$$\mathbf{s}_k \equiv \mathbf{x}_{k+1} - \mathbf{x}_k \quad \text{and} \quad \mathbf{y}_k \equiv \nabla f(\mathbf{x}_{k+1}) - \nabla f(\mathbf{x}_k), \tag{3.2}$$

and $\mathbf{B}_0 \in \mathbb{R}^{n \times n}$ is a suitably-chosen initial matrix. This rank-two update to $\mathbf{B}_k$ preserves positive definiteness when $\mathbf{s}_k^T \mathbf{y}_k > 0$. For large-scale problems only $l \ll n$ of the most recent updates $\{\mathbf{s}_i, \mathbf{y}_i\}$ with $k-l \leq i \leq k-1$ are stored in L-BFGS methods. (Typically, $l \in [3, 7]$ (see, e.g., [BNS94]).)

There are several desirable properties for picking the initial matrix $\mathbf{B}_0$. First, in order for the sequence $\{\mathbf{B}_k\}$ generated by (3.1) to be symmetric and positive definite, it is necessary that $\mathbf{B}_0$ is symmetric and positive definite. Second, it is desirable for $\mathbf{B}_0$ to be easily invertible so that solving linear systems with any matrix in the sequence is computable using the so-called "two-loop recursion" [BNS94] or other recursive formulas for $\mathbf{B}_k^{-1}$ (for an overview of other available methods see [EM17]). For these reasons, $\mathbf{B}_0$ is often chosen to be a scalar multiple of the identity matrix, i.e.,

$$\mathbf{B}_0 = \gamma_k \mathbf{I}_n, \quad \text{with} \quad \gamma_k > 0. \tag{3.3}$$

For BFGS matrices, the conventional choice for γ_k is

$$\gamma_k = \frac{\mathbf{y}_k^T \mathbf{y}_k}{\mathbf{s}_k^T \mathbf{y}_k}, \tag{3.4}$$

which can be viewed as a spectral estimate for of $\nabla^2 f(\mathbf{x}_k)$ [NW06]. (This choice was originally proposed in [SP78] using a derivation based on optimal conditioning.) It is worth noting that this choice of γ_k can also be derived as the minimizer of the scalar minimization problem

$$\gamma_k = \operatorname*{argmin}_{\gamma} \left\| \mathbf{B}_0^{-1} \mathbf{y}_k - \mathbf{s}_k \right\|_2^2, \tag{3.5}$$

where $\mathbf{B}_0^{-1} = \gamma^{-1} \mathbf{I}_n$. For numerical studies on this choice of initialization, see, e.g., the references listed within [BWX96].

In the next sections, we consider a specific dense initialization in lieu of the usual diagonal initialization. The aforementioned separation of $\mathbb{R}^n$ into two complementary subspaces allows us to use different parameters in order to estimate the curvature of the underlying Hessian in these subspaces. An alternative view of this initialization is that it

makes use of *two* spectral estimates of $\nabla^2 f(\mathbf{x}_k)$. Finally, the proposed initialization also allows for efficiently solving and computing products with the resulting quasi-Newton matrices.

3.2 BACKGROUND

The method described in this chapter solves a sequence of L-BFGS trust-region subproblems defined by a shape-changing norm. Therefore we will first describe the compact representation of L-BFGS matrices, and then review their partial eigendecomposition. As in Section 2.5 from Chapter 2, the partial eigendecomposition enables the definition of the shape-changing norm.

3.2.1 THE L-BFGS COMPACT REPRESENTATION

The special structure of the recursion formula for L-BFGS matrices admits the limited-memory compact representation from Subsections 1.4.1 and 1.4.2.

Using the l most recently computed pairs $\{\mathbf{s}_j\}$ and $\{\mathbf{y}_j\}$ given in (3.2), we recall the following matrices

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

With $\mathbf{L}_k$ taken to be the strictly lower triangular part of the matrix of $\mathbf{S}_k^T\mathbf{Y}_k$, and $\mathbf{D}_k$ defined as the diagonal of $\mathbf{S}_k^T\mathbf{Y}_k$, the compact representation of an L-BFGS matrix is

$$\mathbf{B}_k = \mathbf{B}_0 + \mathbf{\Psi}_k\mathbf{M}_k\mathbf{\Psi}_k^T, \tag{3.6}$$

where

$$\mathbf{\Psi}_k \triangleq [\ \mathbf{B}_0\mathbf{S}_k\ \ \mathbf{Y}_k\ ] \quad \text{and} \quad \mathbf{M}_k \triangleq -\begin{bmatrix} \mathbf{S}_k^T\mathbf{B}_0\mathbf{S}_k & \mathbf{L}_k \\ \mathbf{L}_k^T & -\mathbf{D}_k \end{bmatrix}^{-1} \tag{3.7}$$

(see [BNS94] for details). Note that $\mathbf{\Psi}_k \in \mathbb{R}^{n\times 2l}$, and $\mathbf{M}_k \in \mathbb{R}^{2l\times 2l}$ is invertible provided $\mathbf{s}_i^T\mathbf{y}_i > 0$ for all i [BNS94, Theorem 2.3]. An advantage of the compact representation is that if $\mathbf{B}_0$ is appropriately chosen, then computing products with $\mathbf{B}_k$ or solving linear systems with $\mathbf{B}_k$ can be done efficiently [EM17].

3.2.2 PARTIAL EIGENDECOMPOSITION OF $\mathbf{B}_k$

The partial eigendecomposition of the compact representation of any quasi-Newton matrix that is defined by the initial matrix $\mathbf{B}_0 = \gamma_{k-1}\mathbf{I}_n$ is described in Chapter 1, Subsection 1.4.3. Therefore, the L-BFGS compact representation from (3.6) can be efficiently eigendecomposed, using either a partial QR decomposition [BGYZ16]or a partial singular value decomposition (SVD) [Lu96] (cf. Section 1.4.3). The approach

that uses the QR decomposition, based on the assumption that $\mathbf{\Psi}_k$ has rank $r = 2l$ (For the rank-deficient case, see the techniques found in [BGYZ16].), lets

$$\mathbf{\Psi}_k = \mathbf{QR},$$

be the so-called "thin" QR factorization of $\mathbf{\Psi}_k$, where $\mathbf{Q} \in \mathbb{R}^{n\times r}$ and $\mathbf{R} \in \mathbb{R}^{r\times r}$. Since the matrix $\mathbf{RM}_k\mathbf{R}^T$ is a small $(r \times r)$ matrix with $r \ll n$ (recall that $r = 2l$, where l is typically between 3 and 7), it is computationally feasible to calculate its eigendecomposition; thus, suppose $\mathbf{U}\hat{\mathbf{\Lambda}}\mathbf{U}^T$ is the eigendecomposition of $\mathbf{RM}_k\mathbf{R}^T$. Then,

$$\mathbf{\Psi}_k\mathbf{M}_k\mathbf{\Psi}_k^T = \mathbf{QRM}_k\mathbf{R}^T\mathbf{Q}^T = \mathbf{QU}\hat{\mathbf{\Lambda}}\mathbf{U}^T\mathbf{Q}^T = \mathbf{\Psi}_k\mathbf{R}^{-1}\mathbf{U}\hat{\mathbf{\Lambda}}\mathbf{U}^T\mathbf{R}^{-T}\mathbf{\Psi}_k^T.$$

Defining

$$\mathbf{P}_\| = \mathbf{\Psi}_k\mathbf{R}^{-1}\mathbf{U}, \tag{3.8}$$

gives that

$$\mathbf{\Psi}_k\mathbf{M}_k\mathbf{\Psi}_k^T = \mathbf{P}_\|\hat{\mathbf{\Lambda}}\mathbf{P}_\|^T. \tag{3.9}$$

Thus, for $\mathbf{B}_0 = \gamma_{k-1}\mathbf{I}_n$, the eigendecomposition of $\mathbf{B}_k$ can be written as

$$\mathbf{B}_k = \gamma_{k-1}\mathbf{I} + \mathbf{\Psi}_k\mathbf{M}_k\mathbf{\Psi}_k^T = \mathbf{P}\Lambda\mathbf{P}^T, \tag{3.10}$$

where

$$\mathbf{P} \equiv \begin{bmatrix} \mathbf{P}_\| & \mathbf{P}_\perp \end{bmatrix}, \quad \Lambda \triangleq \begin{bmatrix} \hat{\mathbf{\Lambda}} + \gamma_{k-1}\mathbf{I}_r & \\ & \gamma_{k-1}\mathbf{I}_{n-r} \end{bmatrix}, \tag{3.11}$$

and $\mathbf{P}_\perp \in \mathbb{R}^{n\times(n-r)}$ is defined as the orthogonal complement of $\mathbf{P}_\|$, i.e., $\mathbf{P}_\perp^T\mathbf{P}_\perp = \mathbf{I}_{n-r}$ and $\mathbf{P}_\|^T\mathbf{P}_\perp = \mathbf{0}_{r\times(n-r)}$. Hence, $\mathbf{B}_k$ has r eigenvalues given by the diagonal elements of $\hat{\mathbf{\Lambda}} + \gamma_{k-1}\mathbf{I}_r$ and the remaining eigenvalues are γ_{k-1} with multiplicity $n - r$.

PRACTICAL COMPUTATIONS

Using the above method yields the eigenvalues of $\mathbf{B}_k$ as well as the ability to compute products with $\mathbf{P}_\|$. Formula (3.8) indicates that $\mathbf{Q}$ is not required to be explicitly formed in order to compute products with $\mathbf{P}_\|$. For this reason, it is desirable to avoid forming $\mathbf{Q}$ by computing only $\mathbf{R}$ via the Cholesky factorization of $\mathbf{\Psi}_k^T\mathbf{\Psi}_k$, i.e., $\mathbf{\Psi}_k^T\mathbf{\Psi}_k = \mathbf{R}^T\mathbf{R}$ (see [BGYZ16]).

At an additional expense, the eigenvectors stored in the columns of $\mathbf{P}_\|$ may be formed and stored. For the shape-changing trust-region method used in this chapter, it is not required to store $\mathbf{P}_\|$. In contrast, the matrix $\mathbf{P}_\perp$ is prohibitively expensive to form. It turns out that for this work it is only necessary to be able to compute

projections into the subspace $\mathbf{P}_\perp \mathbf{P}_\perp^T$, which can be done using the identity

$$\mathbf{P}_\perp \mathbf{P}_\perp^T = \mathbf{I} - \mathbf{P}_\| \mathbf{P}_\|^T. \tag{3.12}$$

3.2.3 A SHAPE-CHANGING L-BFGS TRUST-REGION METHOD

We now consider the trust-region subproblem defined by the shape-changing infinity norm (2.19) from Section 2.5.4:

$$\underset{\|\mathbf{s}\|_{\mathbf{P},\infty} \leq \Delta_k}{\text{minimize}} \; Q(\mathbf{s}) = \mathbf{g}_k^T \mathbf{s} + \frac{1}{2} \mathbf{s}^T \mathbf{B}_k \mathbf{s}, \tag{3.13}$$

where

$$\|\mathbf{s}\|_{\mathbf{P},\infty} \triangleq \max \left(\|\mathbf{P}_\|^T \mathbf{s}\|_\infty, \|\mathbf{P}_\perp^T \mathbf{s}\|_2 \right) \tag{3.14}$$

and $\mathbf{P}_\|$ and $\mathbf{P}_\perp$ are given in (3.11). Note that the ratio $\|\mathbf{s}\|_2 / \|\mathbf{s}\|_{\mathbf{P},\infty}$ does not depend on n, but only moderately depends on r. (In particular, $1 \leq \|\mathbf{s}\|_2 / \|\mathbf{s}\|_{\mathbf{P},\infty} \leq \sqrt{r+1}$.) Because this norm depends on the eigenvectors of $\mathbf{B}_k$, the shape of the trust region changes each time the quasi-Newton matrix is updated, which is possibly every iteration of a trust-region method. (See [BGYZ16] for more details and other properties of this norm.) The motivation for this choice of norm is that the the trust-region subproblem (3.13) decouples into two separate problems for which closed-form solutions exist.

We now review the closed-form solution to (3.13), as detailed in [BGYZ16]. Let

$$\mathbf{v} = \mathbf{P}^T \mathbf{s} = \begin{bmatrix} \mathbf{P}_\|^T \mathbf{s} \\ \mathbf{P}_\perp^T \mathbf{s} \end{bmatrix} \triangleq \begin{bmatrix} \mathbf{v}_\| \\ \mathbf{v}_\perp \end{bmatrix} \quad \text{and} \quad \mathbf{P}^T \mathbf{g}_k = \begin{bmatrix} \mathbf{P}_\|^T \mathbf{g}_k \\ \mathbf{P}_\perp^T \mathbf{g}_k \end{bmatrix} \triangleq \begin{bmatrix} \mathbf{g}_\| \\ \mathbf{g}_\perp \end{bmatrix}. \tag{3.15}$$

With this change of variables, the objective function of (3.13) becomes

$$\begin{aligned} Q(\mathbf{P}\mathbf{v}) &= \mathbf{g}_k^T \mathbf{P}\mathbf{v} + \frac{1}{2} \mathbf{v}^T \left(\hat{\mathbf{\Lambda}} + \gamma_{k-1} \mathbf{I}_n \right) \mathbf{v} \\ &= \mathbf{g}_\|^T \mathbf{v}_\| + \mathbf{g}_\perp^T \mathbf{v}_\perp + \frac{1}{2} \left(\mathbf{v}_\|^T \left(\hat{\mathbf{\Lambda}} + \gamma_{k-1} \mathbf{I}_r \right) \mathbf{v}_\| + \gamma_{k-1} \|\mathbf{v}_\perp\|_2^2 \right) \\ &= \mathbf{g}_\|^T \mathbf{v}_\| + \frac{1}{2} \mathbf{v}_\|^T \left(\hat{\mathbf{\Lambda}} + \gamma_{k-1} \mathbf{I}_r \right) \mathbf{v}_\| + \mathbf{g}_\perp^T \mathbf{v}_\perp + \frac{1}{2} \gamma_{k-1} \|\mathbf{v}_\perp\|_2^2 . \end{aligned}$$

The trust-region constraint $\|\mathbf{s}\|_{\mathbf{P},\infty} \leq \Delta_k$ implies $\left\|\mathbf{v}_\|\right\|_\infty \leq \Delta_k$ and $\|\mathbf{v}_\perp\|_2 \leq \Delta_k$, which decouples (3.13) into the following two trust-region subproblems:

$$\underset{\left\|\mathbf{v}_\|\right\|_\infty \leq \Delta_k}{\text{minimize}} \; q_\| \left(\mathbf{v}_\| \right) \quad \triangleq \quad \mathbf{g}_\|^T \mathbf{v}_\| + \frac{1}{2} \mathbf{v}_\|^T \left(\hat{\mathbf{\Lambda}} + \gamma_{k-1} \mathbf{I}_r \right) \mathbf{v}_\| \tag{3.16}$$

$$\underset{\|\mathbf{v}_\perp\|_2 \leq \Delta_k}{\text{minimize}} \; q_\perp \left(\mathbf{v}_\perp \right) \quad \triangleq \quad \mathbf{g}_\perp^T \mathbf{v}_\perp + \frac{1}{2} \gamma_{k-1} \|\mathbf{v}_\perp\|_2^2 . \tag{3.17}$$

Observe that the resulting minimization problems are considerably simpler than the

original problem since in both cases the Hessian of the new objective functions are diagonal matrices. The solutions to these decoupled problems have closed-form analytical solutions [BGYZ16, BBE+16]. Specifically, letting $\lambda_i \triangleq \hat{\lambda}_i + \gamma_{k-1}$, the solution to (3.16) is given coordinate-wise by

$$[\mathbf{v}_{||}^*]_i = \begin{cases} -\frac{[\mathbf{g}_{||}]_i}{\lambda_i} & \text{if } \left|\frac{[\mathbf{g}_{||}]_i}{\lambda_i}\right| \leq \Delta_k \text{ and } \lambda_i > 0, \\ c & \text{if } \left[\mathbf{g}_{\|}\right]_i = 0 \text{ and } \lambda_i = 0, \\ -\text{sgn}(\left[\mathbf{g}_{\|}\right]_i)\Delta_k & \text{if } \left[\mathbf{g}_{\|}\right]_i \neq 0 \text{ and } \lambda_i = 0, \\ \pm\Delta_k & \text{if } \left[\mathbf{g}_{\|}\right]_i = 0 \text{ and } \lambda_i < 0, \\ -\frac{\Delta_k}{|[\mathbf{g}_{||}]_i|}\left[\mathbf{g}_{||}\right]_i & \text{otherwise,} \end{cases} \quad , \tag{3.18}$$

where c is any real number in $[-\Delta_k, \Delta_k]$ and 'sgn' denotes the signum function. Meanwhile, the minimizer of (3.17) is given by

$$\mathbf{v}_\perp^* = \beta \mathbf{g}_\perp, \tag{3.19}$$

where

$$\beta = \begin{cases} -\frac{1}{\gamma_{k-1}} & \text{if } \gamma_{k-1} > 0 \text{ and } \|\mathbf{g}_\perp\|_2 \leq \Delta_k |\gamma_{k-1}|, \\ -\frac{\Delta_k}{\|\mathbf{g}_\perp\|_2} & \text{otherwise.} \end{cases} \tag{3.20}$$

Note that the solution to (3.13) is then

$$\mathbf{s}^* = \mathbf{P}\mathbf{v}^* = \mathbf{P}_\| \mathbf{v}_\|^* + \mathbf{P}_\perp \mathbf{v}_\perp^* = \mathbf{P}_\| \mathbf{v}_\|^* + \beta \mathbf{P}_\perp \mathbf{g}_\perp = \mathbf{P}_\| \mathbf{v}_\|^* + \beta \mathbf{P}_\perp \mathbf{P}_\perp^T \mathbf{g}_k, \tag{3.21}$$

where the latter term is computed using (3.12). Additional simplifications yield the following expression for $\mathbf{s}^*$:

$$\mathbf{s}^* = \beta \mathbf{g}_k + \mathbf{P}_\| (\mathbf{v}_\|^* - \beta \mathbf{g}_\|). \tag{3.22}$$

The overall cost of computing the solution to (3.13) is comparable to that of using the Euclidean norm (see [BGYZ16]). The main advantage of using the shape-changing norm (3.14) is that the solution $\mathbf{s}^*$ in (3.22) has a closed-form expression.

3.3 THE PROPOSED METHOD

In this section, we present a new dense initialization and demonstrate how it is naturally well-suited for trust-region methods defined by the shape-changing infinity norm. Finally, we present a full trust-region algorithm that uses the dense initialization, consider its computational cost, and prove global convergence.

3.3.1 DENSE INITIAL MATRIX $\widehat{\mathbf{B}}_0$

In this section, we propose a new dense initialization for quasi-Newton methods. Importantly, in order to retain the efficiency of quasi-Newton methods the dense initialization matrix and subsequently updated quasi-Newton matrices are never explicitly formed. This initialization can be used with any quasi-Newton update for which there is a compact representation; however, for simplicity, we continue to refer to the BFGS update throughout this section. For notational purposes, we use the initial matrix $\mathbf{B}_0$ to represent the multiple of identity matrix, and $\widehat{\mathbf{B}}_0$ to denote the proposed dense initialization. Similarly, $\{\mathbf{B}_k\}$ and $\{\widehat{\mathbf{B}}_k\}$ will be used to denote the sequences of matrices obtained using the initializations $\mathbf{B}_0$ and $\widehat{\mathbf{B}}_0$, respectively.

Our goal in choosing an alternative initialization is four-fold: (i) to be able to treat subspaces differently depending on whether curvature information is available or not, (ii) to preserve properties of symmetry and positive-definiteness, (iii) to be able to efficiently compute products with the resulting quasi-Newton matrices, and (iv) to be able to efficiently solve linear systems involving the resulting quasi-Newton matrices. The initialization proposed in this paper leans upon two different parameter choices that can be viewed as an estimate of the curvature of $\nabla^2 f(\mathbf{x}_k)$ in two subspaces: one spanned by the columns of $\mathbf{P}_\parallel$ and another spanned by the columns of $\mathbf{P}_\perp$.

The usual initialization for a BFGS matrix $\mathbf{B}_k$ is $\mathbf{B}_0 = \gamma_{k-1}\mathbf{I}_n$, where $\gamma_{k-1} > 0$. Note that this initialization is equivalent to

$$\mathbf{B}_0 = \gamma_{k-1}\mathbf{P}\mathbf{P}^T = \gamma_{k-1}\mathbf{P}_\parallel\mathbf{P}_\parallel^T + \gamma_{k-1}\mathbf{P}_\perp\mathbf{P}_\perp^T.$$

In contrast, for fixed $\gamma_{k-1}, \gamma_{k-1}^\perp \in \mathbb{R}$, consider the following symmetric, and in general, dense initialization matrix:

$$\widehat{\mathbf{B}}_0 = \gamma_{k-1}\mathbf{P}_\parallel\mathbf{P}_\parallel^T + \gamma_{k-1}^\perp\mathbf{P}_\perp\mathbf{P}_\perp^T, \tag{3.23}$$

where $\mathbf{P}_\parallel$ and $\mathbf{P}_\perp$ are the matrices of eigenvectors defined in Section 3.2.2. We now derive the eigendecomposition of $\widehat{\mathbf{B}}_k$.

Theorem 3.1. *Let $\widehat{\mathbf{B}}_0$ be defined as in* (3.23)*. Then $\widehat{\mathbf{B}}_k$ generated using (3.1) has the eigendecomposition*

$$\widehat{\mathbf{B}}_k = \begin{bmatrix} \mathbf{P}_\parallel & , \mathbf{P}_\perp \end{bmatrix} \begin{bmatrix} \hat{\mathbf{\Lambda}} + \gamma_{k-1}\mathbf{I}_r & \\ & \gamma_{k-1}^\perp\mathbf{I}_{n-r} \end{bmatrix} \begin{bmatrix} \mathbf{P}_\parallel & \mathbf{P}_\perp \end{bmatrix}^T, \tag{3.24}$$

where $\mathbf{P}_\parallel, \mathbf{P}_\perp$, and $\hat{\Lambda}$ are given in (3.8), (3.11), and (3.9), respectively.

Proof. First note that the columns of $\mathbf{S}_k$ are in Range$(\mathbf{\Psi}_k)$, where $\mathbf{\Psi}_k$ is defined in (3.7). From (3.8), Range$(\mathbf{\Psi}_k)$ = Range$(\mathbf{P}_\parallel)$; thus, $\mathbf{P}_\parallel\mathbf{P}_\parallel^T\mathbf{S}_k = \mathbf{S}_k$ and $\mathbf{P}_\perp^T\mathbf{S}_k = \mathbf{0}_{(n-r)\times r}$. This

gives that

$$\widehat{\mathbf{B}}_0\mathbf{S}_k = \gamma_{k-1}\mathbf{P}_{\|}\mathbf{P}_{\|}^T\mathbf{S}_k + \gamma_{k-1}^{\perp}\mathbf{P}_{\perp}\mathbf{P}_{\perp}^T\mathbf{S}_k = \gamma_{k-1}\mathbf{S}_k = \mathbf{B}_0\mathbf{S}_k. \tag{3.25}$$

Combining the compact representation of $\widehat{\mathbf{B}}_k$ ((3.6) and (3.7)) together with (3.25) yields

$$\begin{aligned}
\widehat{\mathbf{B}}_k &= \widehat{\mathbf{B}}_0 - \begin{bmatrix}\widehat{\mathbf{B}}_0\mathbf{S}_k & \mathbf{Y}_k\end{bmatrix}\begin{bmatrix}\mathbf{S}_k^T\widehat{\mathbf{B}}_0\mathbf{S}_k & \mathbf{L}_k \\ \mathbf{L}_k^T & -\mathbf{D}_k\end{bmatrix}^{-1}\begin{bmatrix}\mathbf{S}_k^T\widehat{\mathbf{B}}_0 \\ \mathbf{Y}_k^T\end{bmatrix} \\
&= \widehat{\mathbf{B}}_0 - [\mathbf{B}_0\mathbf{S}_k \ \mathbf{Y}_k]\begin{bmatrix}\mathbf{S}_k^T\mathbf{B}_0\mathbf{S}_k & \mathbf{L}_k \\ \mathbf{L}_k^T & -\mathbf{D}_k\end{bmatrix}^{-1}\begin{bmatrix}\mathbf{S}_k^T\mathbf{B}_0 \\ \mathbf{Y}_k^T\end{bmatrix} \\
&= \gamma_{k-1}\mathbf{P}_{\|}\mathbf{P}_{\|}^T + \gamma_{k-1}^{\perp}\mathbf{P}_{\perp}\mathbf{P}_{\perp}^T + \mathbf{P}_{\|}\hat{\mathbf{\Lambda}}\mathbf{P}_{\|}^T \\
&= \mathbf{P}_{\|}\left(\hat{\Lambda} + \gamma_{k-1}\mathbf{I}_r\right)\mathbf{P}_{\|}^T + \gamma_{k-1}^{\perp}\mathbf{P}_{\perp}\mathbf{P}_{\perp}^T,
\end{aligned}$$

which is equivalent to (3.24). □

It can be easily verified that (3.24) holds also for $\mathbf{P}_{\|}$ defined in [BGYZ16] for possibly rank-deficient $\mathbf{\Psi}_k$. applies only to the special case when $\mathbf{\Psi}_k$ is full-rank.)

Theorem 3.1 shows that the matrix $\widehat{\mathbf{B}}_k$ that results from using the initialization (3.23) shares the same eigenvectors as $\mathbf{B}_k$, generated using $\mathbf{B}_0 = \gamma_{k-1}\mathbf{I}_n$. Moreover, the eigenvalues corresponding to the eigenvectors stored in the columns of $\mathbf{P}_{\|}$ are the same for $\widehat{\mathbf{B}}_k$ and $\mathbf{B}_k$. The only difference in the eigendecompositions of $\widehat{\mathbf{B}}_k$ and $\mathbf{B}_k$ is in the eigenvalues corresponding to the eigenvectors stored in the columns of $\mathbf{P}_{\perp}$. This is summarized in the following corollary.

Corollary 3.2. *Suppose $\mathbf{B}_k$ is a BFGS matrix initialized with $\mathbf{B}_0 = \gamma_{k-1}\mathbf{I}_n$ and $\widehat{\mathbf{B}}_k$ is a BFGS matrix initialized with (3.23). Then $\mathbf{B}_k$ and $\widehat{\mathbf{B}}_k$ have the same eigenvectors; moreover, these matrices have r eigenvalues in common given by $\lambda_i \triangleq \hat{\lambda}_i + \gamma_{k-1}$ where $\hat{\Lambda} = diag(\hat{\lambda}_1, \ldots, \hat{\lambda}_r)$.*

Proof. The corollary follows immediately by comparing (3.10) with (3.24).

The results of Theorem 3.1 and Corollary 3.2 may seem surprising at first since every term in the compact representation ((3.6) and (3.7)) depends on the initialization; moreover, $\widehat{\mathbf{B}}_0$ is, generally speaking, a dense matrix while $\mathbf{B}_0$ is a diagonal matrix. However, viewed from the perspective of (3.23), the parameter $\gamma_{k-1}^{\perp}$ only plays a role in scaling the subspace spanned by the columns of $\mathbf{P}_{\perp}$.

The initialization $\widehat{\mathbf{B}}_0$ allows for two separate curvature approximations for the BFGS matrix: one in the space spanned by columns of $\mathbf{P}_{\|}$ and another in the space spanned by the columns of $\mathbf{P}_{\perp}$. In the next subsection, we show that this initialization is naturally well-suited for solving trust-region subproblems defined by the shape-changing infinity norm.

3.3.2 THE TRUST-REGION SUBPROBLEM

Here we will show that the use of $\widehat{\mathbf{B}}_0$ provides the same subproblem separability as $\mathbf{B}_0$ does in the case of the shape-changing infinity norm.

For $\widehat{\mathbf{B}}_0$ given by (3.23), consider the objective function of the trust-region subproblem (3.13) resulting from the change of variables (3.15):

$$\begin{aligned} Q(\mathbf{Pv}) &= \mathbf{g}_k^T\mathbf{Pv} + \frac{1}{2}\mathbf{v}^T\mathbf{P}^T\widehat{\mathbf{B}}_k\mathbf{Pv} \\ &= \mathbf{g}_{\|}^T\mathbf{v}_{\|} + \frac{1}{2}\mathbf{v}_{\|}^T\left(\hat{\mathbf{\Lambda}} + \gamma_{k-1}\mathbf{I}_r\right)\mathbf{v}_{\|} + \mathbf{g}_{\perp}^T\mathbf{v}_{\perp} + \frac{1}{2}\gamma_{k-1}^{\perp}\left\|\mathbf{v}_{\perp}\right\|_2^2. \end{aligned}$$

Thus, (3.13) decouples into two subproblems: The corresponding subproblem for $q_{\|}(\mathbf{v}_{\|})$ remains (3.16) and the subproblem for $q_{\perp}(\mathbf{v}_{\perp})$ becomes

$$\underset{\|\mathbf{v}_{\perp}\|_2 \leq \Delta_k}{\text{minimize}}\; q_{\perp}\left(\mathbf{v}_{\perp}\right) \triangleq \mathbf{g}_{\perp}^T\mathbf{v}_{\perp} + \frac{1}{2}\gamma_{k-1}^{\perp}\left\|\mathbf{v}_{\perp}\right\|_2^2. \tag{3.26}$$

The solution to (3.26) is now given by

$$\mathbf{v}_{\perp}^* = \widehat{\beta}\mathbf{g}_{\perp}, \tag{3.27}$$

where

$$\widehat{\beta} = \begin{cases} -\frac{1}{\gamma_{k-1}^{\perp}} & \text{if } \gamma_{k-1}^{\perp} > 0 \text{ and } \|\mathbf{g}_{\perp}\|_2 \leq \Delta_k|\gamma_{k-1}^{\perp}|, \\ -\frac{\Delta_k}{\|\mathbf{g}_{\perp}\|_2} & \text{otherwise.} \end{cases} \tag{3.28}$$

Thus, the solution $\mathbf{s}^*$ can be expressed as

$$\mathbf{s}^* = \widehat{\beta}\mathbf{g}_k + \mathbf{P}_{\|}(\mathbf{v}_{\|}^* - \widehat{\beta}\mathbf{g}_{\|}), \tag{3.29}$$

which can computed as efficiently as the solution in (3.22) for conventional initial matrices since they differ only by the scalar ($\widehat{\beta}$ in (3.29) versus β in (3.22)).

3.3.3 DETERMINING THE PARAMETER $\gamma_{k-1}^{\perp}$

The values γ_{k-1} and $\gamma_{k-1}^{\perp}$ can be updated at each iteration. Since we have little information about the underlying function f in the subspace spanned by the columns of $\mathbf{P}_{\perp}$, it is reasonable to make conservative choices for $\gamma_{k-1}^{\perp}$. Note that in the case that $\gamma_{k-1}^{\perp} > 0$ and $\|\mathbf{g}_{\perp}\|_2 \leq \Delta_k|\gamma_{k-1}^{\perp}|$, the parameter $\gamma_{k-1}^{\perp}$ scales the solution $\mathbf{v}_{\perp}^*$ (see 3.28); large values of $\gamma_{k-1}^{\perp}$ will reduce these step lengths in the space spanned by $\mathbf{P}_{\perp}$. Since the space $\mathbf{P}_{\perp}$ does not explicitly use information produced by past iterations, it seems desirable to choose $\gamma_{k-1}^{\perp}$ to be large. However, the larger that $\gamma_{k-1}^{\perp}$ is chosen, the closer $\mathbf{v}_{\perp}^*$ will be to the zero vector. Also note that if $\gamma_{k-1}^{\perp} < 0$ then the solution to the subproblem (3.26) will always lie on the boundary, and thus, the actual value of $\gamma_{k-1}^{\perp}$

becomes irrelevant. Moreover, for values $\gamma_{k-1}^{\perp} < 0$, $\widehat{\mathbf{B}}_k$ is not guaranteed to be positive definite. For these reasons, we suggest sufficiently large and positive values for $\gamma_{k-1}^{\perp}$ related to the curvature information gathered in $\gamma_1, \ldots, \gamma_{k-1}$. Specific choices for $\gamma_{k-1}^{\perp}$ are presented in the numerical results section.

3.3.4 THE ALGORITHM AND ITS PROPERTIES

In Algorithm 3., we present a basic trust-region method that uses the proposed dense initialization. In this setting, we consider the computational cost of the proposed method, and we prove global convergence of the overall trust-region method. Since $\mathbf{P}$ may change every iteration, the corresponding norm $\|\cdot\|_{\mathbf{P},\infty}$ may change each iteration. Note that initially there are no stored quasi-Newton pairs $\{\mathbf{s}_j, \mathbf{y}_j\}$. In this case, we assume $\mathbf{P}_{\perp} = \mathbf{I}_n$ and $\mathbf{P}_{\|}$ does not exist, i.e., $\mathbf{B}_0 = \gamma_0^{\perp}\mathbf{I}_n$.

ALGORITHM 3.1

Set $\mathbf{x}_0 \in \mathbb{R}^n$, $\Delta_0 > 0$, $\epsilon > 0$, $\gamma_0^{\perp} > 0$, $0 \leq \tau_1 < \tau_2 < 0.5 < \tau_3 < 1$, $0 < \eta_1 < \eta_2 \leq 0.5 < \eta_3 < 1 < \eta_4$; Compute $\mathbf{g}_0$;

For $k = 1, 2, \ldots$

1. If $\|\mathbf{g}_k\| \leq \epsilon$ terminate;
2. Compute $\mathbf{\Psi}_k, \mathbf{R}^{-1}, \mathbf{M}_k, \mathbf{U}, \hat{\Lambda}$ and Λ from Section 3.2; Compute $\mathbf{s}^*$ from (3.21), compute $\rho_k = \frac{f(\mathbf{x}_k+\mathbf{s}^*)-f(\mathbf{x}_k)}{Q(\mathbf{s}^*)}$;
3. If $\rho_k \geq \tau_1$, then set $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{s}^*$, update $\mathbf{g}_{k+1}$, $\mathbf{s}_k$, $\mathbf{y}_k$, γ_k and $\gamma_k^{\perp}$; If $\rho_k < \tau_1$ set $\mathbf{x}_{k+1} = \mathbf{x}_k$;
4. If $\rho_k \leq \tau_2$, then set $\Delta_{k+1} = \min(\eta_1\Delta_k, \eta_2\|\mathbf{s}_k\|_{\mathbf{P},\infty})$ go to 1.;
5. If $\rho_k \geq \tau_3$ and $\|\mathbf{s}_k\|_{\mathbf{P},\infty} \geq \eta_3\Delta_k$, then set $\Delta_{k+1} = \eta_4\Delta_k$; Otherwise set $\Delta_{k+1} = \Delta_k$;

The only difference between Algorithm 3.1 and the proposed LMTR algorithm in [BGYZ16] is the initialization matrix. Computationally speaking, the use of a dense initialization in lieu of a diagonal initialization plays out only in the computation of $\mathbf{s}^*$ by (3.21). However, there is no computational cost difference: The cost of computing the value for β using (3.28) in Algorithm 3.1 instead of (3.20) in the LMTR algorithm is the same. Thus, the dominant cost per iteration for both Algorithm 3.1 and the LMTR algorithm is $4ln$ operations (see [BGYZ16] for details). Note that this is the same cost-per-iteration as the line search L-BFGS algorithm [BNS94].

In the next result, we provide the global convergence theory for Algorithm 3.1. This result is based on the convergence analysis presented in [BGYZ16].

Theorem 3.3. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be twice-continuously differentiable and bounded below on $\mathbb{R}^n$. Suppose that there exists a scalar $c_1 > 0$ such that*

$$\|\nabla^2 f(\mathbf{x})\| \leq c_1, \ \forall \mathbf{x} \in \mathbb{R}^n. \tag{3.30}$$

Furthermore, suppose for $\widehat{\mathbf{B}}_0$ defined by (3.23), that there exists a positive scalar c_2 such that

$$\gamma_{k-1}, \gamma_{k-1}^{\perp} \in (0, c_2], \ \forall k \geq 0, \tag{3.31}$$

and there exists a scalar $c_3 \in (0, 1)$ such that the inequality

$$\mathbf{s}_j^T \mathbf{y}_j \geq c_3 \|\mathbf{s}_j\| \|\mathbf{y}_j\| \tag{3.32}$$

holds for each quasi-Newton pair $\{\mathbf{s}_j, \mathbf{y}_j\}$. Then, if the stopping criteria is suppressed, the infinite sequence $\{\mathbf{x}_k\}$ generated by Algorithm 3.1 satisfies

$$\lim_{k \to \infty} \|\nabla f(\mathbf{x}_k)\| = \mathbf{0}. \tag{3.33}$$

Proof. From (3.31), we have $\|\hat{B}_0\| \leq c_2$, which holds for each $k \geq 0$. Then, by [[BGYZ16], Lemma 3], there exists $c_4 > 0$ such that

$$\|\widehat{\mathbf{B}}_0\| \leq c_4.$$

Then, (3.33) follows from [[BGYZ16], Theorem 1]. ■

In the following section, we consider $\gamma_{k-1}^{\perp}$ parameterized by two scalars, c and λ:

$$\gamma_{k-1}^{\perp}(c, \lambda) = \lambda c \gamma_{k-1}^{\max} + (1 - \lambda)\gamma_{k-1}, \tag{3.34}$$

where $c \geq 1, \lambda \in [0, 1]$, and

$$\gamma_{k-1}^{\max} \triangleq \max_{1 \leq i \leq k-1} \gamma_i,$$

where γ_{k-1} is taken to be the conventional initialization given by (3.4). (This choice for $\gamma_{k-1}^{\perp}$ will be further discussed in Section 3.4.) We now show that Algorithm 3.1 converges for these choices of $\gamma_{k-1}^{\perp}$. Assuming that (3.30) and (3.32) hold, it remains to show that (3.31) holds for these choices of $\gamma_{k-1}^{\perp}$. To see that (3.31) holds, notice that in this case,

$$\gamma_{k-1} = \frac{\mathbf{y}_{k-1}^T \mathbf{y}_{k-1}}{\mathbf{s}_{k-1}^T \mathbf{y}_{k-1}} \leq \frac{\mathbf{y}_{k-1}^T \mathbf{y}_{k-1}}{c_3 \|\mathbf{s}_{k-1}\| \|\mathbf{y}_{k-1}\|} \leq \frac{\|\mathbf{y}_{k-1}\|}{c_3 \|\mathbf{s}_{k-1}\|}.$$

Substituting in for the definitions of $\mathbf{y}_{k-1}$ and $\mathbf{s}_{k-1}$ yields that

$$\gamma_{k-1} \leq \frac{\|\nabla f(\mathbf{x}_k) - \nabla f(\mathbf{x}_{k-1})\|}{c_3 \|\mathbf{x}_k - \mathbf{x}_{k-1}\|},$$

implying that (3.31) holds. Thus, Algorithm 3.1 converges for these choices for $\gamma_{k-1}^{\perp}$.

3.3.5 IMPLEMENTATION DETAILS

In this section, we describe how we incorporate the dense initialization within the existing LMTR algorithm [BGYZ16]. At the beginning of each iteration, the LMTR algorithm with dense initialization checks if the unconstrained minimizer (also known as the *full quasi-Newton trial step*),

$$\mathbf{s}_u^* = -\widehat{\mathbf{B}}_k^{-1}\mathbf{g}_k \tag{3.35}$$

lies inside the trust region defined by the two-norm. Because our proposed method uses a dense initialization, the so-called "two-loop recursion" [6] is not applicable for computing the unconstrained minimizer $\mathbf{s}_u^*$ in (3.35). However, products with $\widehat{\mathbf{B}}_k^{-1}$ can be performed using the compact representation without involving a partial eigendecomposition, i.e.,

$$\widehat{\mathbf{B}}_k^{-1} = \frac{1}{\gamma_{k-1}^{\perp}}\mathbf{I}_n + \mathbf{\Psi}_k \widehat{\mathbf{M}}_k \mathbf{\Psi}_k^T, \tag{3.36}$$

where $\mathbf{\Psi}_k = [\ \mathbf{S}_k \ \ \mathbf{Y}_k\ ]$,

$$\widehat{\mathbf{M}}_k = \begin{bmatrix} \mathbf{T}_k^{-T}(\mathbf{E}_k + \gamma_{k-1}^{-1}\mathbf{Y}_k^T\mathbf{Y}_k)\mathbf{T}_k^{-1} & -\gamma_{k-1}^{-1}\mathbf{T}_k^{-T} \\ -\gamma_{k-1}^{-1}\mathbf{T}_k^{-1} & \mathbf{0}_m \end{bmatrix} + \alpha_{k-1}(\mathbf{\Psi}_k^T\mathbf{\Psi}_k)^{-1},$$

$\alpha_{k-1} = \left(\dfrac{1}{\gamma_{k-1}} - \dfrac{1}{\gamma_{k-1}^{\perp}}\right)$, $\mathbf{T}_k$ is the upper triangular part of the matrix $\mathbf{S}_k^T\mathbf{Y}_k$, and $\mathbf{E}_k$ is its diagonal. Thus, the inequality

$$\|\mathbf{s}_u^*\|_2 \leq \Delta_k \tag{3.37}$$

is easily verified without explicitly forming $\mathbf{s}_u^*$ using the identity

$$\|\mathbf{s}_u^*\|_2^2 = \mathbf{g}_k^T\widehat{\mathbf{B}}_k^{-2}\mathbf{g}_k = \gamma_{k-1}^{-2}\|\mathbf{g}_k\|^2 + 2\gamma_{k-1}^{-1}\mathbf{u}_k^T\hat{\mathbf{M}}_k\mathbf{u}_k + \mathbf{u}_k^T\widehat{\mathbf{M}}_k(\mathbf{\Psi}_k^T\mathbf{\Psi}_k)\widehat{\mathbf{M}}_k\mathbf{u}_k. \tag{3.38}$$

Here, as in the LMTR algorithm, the vector $\mathbf{u}_k = \mathbf{\Psi}_k^T\mathbf{g}_k$ is computed at each iteration when updating the matrix $\mathbf{\Psi}_k^T\mathbf{\Psi}_k$. Thus, the computational cost of $\|\mathbf{s}_u^*\|_2$ is low because the matrices $\mathbf{\Psi}_k^T\mathbf{\Psi}_k$ and $\widehat{\mathbf{M}}_k$ are small in size. The norm equivalence for the shape-changing infinity norm studied in [BGYZ16] guarantees that (3.37) implies that the inequality $\|\mathbf{s}_u^*\|_{\mathbf{P},\infty} \leq \Delta_k$ is satisfied; in this case, $\mathbf{p}_u^*$ is the exact solution of the trust-region subproblem defined by the shape-changing infinity norm.

If (3.37) holds, the algorithm computes $\mathbf{s}_u^*$ for generating the trial point $\mathbf{x}_k + \mathbf{s}_u^*$. It can be easily seen that the cost of computing $\mathbf{s}_u^*$ is $4ln$ operations, i.e. it is the same as for computing search direction in the line search L-BFGS algorithm [6].

On the other hand, if (3.37) does not hold, then for producing a trial point, the partial eigendecomposition is computed, and the trust-region subproblem is decoupled and solved exactly as described in Section 3.3.2.

3.4 NUMERICAL EXPERIMENTS

We perform numerical experiments on 65 large-scale ($1000 \leq n \leq 10000$) CUTEst [GOT03] test problems, made up of all the test problems in [BGYZ16] plus an additional three (FMINSURF, PENALTY2, and TESTQUAD [GOT03]) since at least one of the methods in the experiments detailed below converged on one of these three problems. The same trust-region method and default parameters as in [BGYZ16, Algorithm 1] were used for the outer iteration. At most five quasi-Newton pairs $\{\mathbf{s}_k, \mathbf{y}_k\}$ were stored, i.e., $l = 5$. The relative stopping criterion was

$$\|\mathbf{g}_k\|_2 \leq \epsilon \max\left(1, \|\mathbf{x}_k\|_2\right),$$

with $\epsilon = 10^{-10}$. The initial step, $\mathbf{s}_0$, was determined by a backtracking line-search along the normalized steepest descent direction. The rank of $\boldsymbol{\Psi}_k$ was estimated by the number of positive diagonal elements in the diagonal matrix of the $\mathrm{LDL}^{\mathrm{T}}$ decomposition (or eigendecomposition of $\boldsymbol{\Psi}_k^T \boldsymbol{\Psi}_k$) that are larger than the threshold $\epsilon_r = (10^{-7})^2$. (Note that the columns of $\boldsymbol{\Psi}_k$ are normalized.)

We provide performance profiles (see [DM02]) for the number of iterations (`iter`) where the trust-region step is accepted and the average time (`time`) for each solver on the test set of problems. The performance metric, ρ, for the 65 problems is defined by

$$\rho_s(\tau) = \frac{\text{card}\left\{p : \pi_{p,s} \leq \tau\right\}}{65} \quad \text{and} \quad \pi_{p,s} = \frac{t_{p,s}}{\min\limits_{1 \leq i \leq S} t_{p,i}},$$

where $t_{p,s}$ is the "output" (i.e., time or iterations) 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. We observe as in [BGYZ16] that the first runs significantly differ in time from the remaining runs, and thus, we ran each algorithm ten times on each problem, reporting the average of the final eight runs.

In this section, we present the following six types of experiments involving LMTR:

1. A comparison of results for different values of $\gamma_{k-1}^{\perp}(c, \lambda)$.

2. Two versions of computing full quasi-Newton trial step (see Section 3.5) are compared. One version uses the dense initialization to compute $\mathbf{s}_u^*$ as described in Section 3.5; the other uses the conventional initialization, i.e., $\mathbf{s}_u^*$ is computed as $\mathbf{s}_u^* = \mathbf{B}_k^{-1}\mathbf{g}_k$. In the both cases, the dense initialization is used for computing trial steps obtained from explicitly solving the trust-region subproblem (Section 3.2) when the full quasi-Newton trial step is not accepted.

3. A comparison of alternative ways of computing the partial eigendecomposition (Section 2.2), namely, those based on QR and SVD factorizations.

4. A comparison of LMTR together with a dense initialization and the line search L-BFGS method with the conventional initialization.

5. A comparison of LMTR with a dense initialization and L-BFGS-TR [BGYZ16], which computes a scaled quasi-Newton direction that lies inside a trust region. This method can be viewed as a hybrid line search and trust-region algorithm.

6. A comparison of the dense and conventional initializations.

In the experiments below, the dense initial matrix $\widehat{\mathbf{B}}_0$ corresponding to $\gamma_{k-1}^{\perp}(c,\lambda)$ given in (3.34) will be denoted by

$$\widehat{\mathbf{B}}_0(c,\lambda) \equiv \gamma_{k-1}\mathbf{P}_{\|}\mathbf{P}_{\|}^T + \gamma_{k-1}^{\perp}(c,\lambda)\mathbf{P}_{\perp}\mathbf{P}_{\perp}^T.$$

Using this notation, the conventional initialization $\mathbf{B}_0(\gamma_{k-1})$ can be written as $\widehat{\mathbf{B}}_0(1,0)$.

Experiment 1. In this experiment, we consider various scalings of a proposed $\gamma_{k-1}^{\perp}$ using LMTR. As argued in Section 3.3.3, it is reasonable to choose $\gamma_{k-1}^{\perp}$ to be large and positive; in particular, $\gamma_{k-1}^{\perp} \geq \gamma_{k-1}$. Thus, we consider the parametrized family of choices $\gamma_{k-1}^{\perp} \triangleq \gamma_{k-1}^{\perp}(c,\lambda)$ given in (3.34). These choices correspond to conservative strategies for computing steps in the space spanned by $\mathbf{P}_{\perp}$ (see the discussion in Section 3.3.3). Moreover, these can also be viewed as conservative strategies since the trial step computed using $\mathbf{B}_0$ will always be larger in Euclidean norm than the trial step computed using $\widehat{\mathbf{B}}_0$ using (3.34). To see this, note that in the parallel subspace the solutions will be identical using both initializations since the solution $\mathbf{v}_{\|}^*$ does not depend on $\gamma_{k-1}^{\perp}$ (see (3.18)); in contrast, in the orthogonal subspace, $\|\mathbf{v}_{\perp}^*\|$ inversely depends on $\gamma_{k-1}^{\perp}$ (see (3.27) and (3.28)).

We report results using different values of c and λ for $\gamma_k^{\perp}(c,\lambda)$ on two sets of tests. On the first set of tests, the dense initialization was used for the entire LMTR algoirthm. However, for the second set of tests, the dense initialization was not used for the computation of the unconstrained minimizer $\mathbf{s}_u^*$; that is, LMTR was run using $\mathbf{B}_k$ (initialized with $\mathbf{B}_0 = \gamma_{k-1}I$ where γ_{k-1} is given in (3.4)) for the computation of the unconstrained

Parameters		
c	λ	$\gamma_{k-1}^{\perp}$
1	1	$\gamma_{k-1}^{\max}$
2	1	$2\gamma_{k-1}^{\max}$
1	$\frac{1}{2}$	$\frac{1}{2}\gamma_{k-1}^{\max} + \frac{1}{2}\gamma_{k-1}$
1	$\frac{1}{4}$	$\frac{1}{4}\gamma_{k-1}^{\max} + \frac{3}{4}\gamma_{k-1}$

TABLE 3.1
Values for $\gamma_{k-1}^{\perp}$ used in Experiment 1.

minimizer $\mathbf{s}_u^* = -\mathbf{B}_k^{-1}\mathbf{g}_k$. However, if the unconstrained minimizer was not taken to be the approximate solution of the subproblem, $\widehat{\mathbf{B}}_k$ with the dense initialization was used for the shape-changing component of the algorithm with $\gamma_{k-1}^{\perp}$ defined as in (3.34). The values of c and λ chosen for Experiment 1 are found in Table 3.4. (See Section 3.3.5 for details on the LMTR algorithm.)

Figure 3.1 displays the performance profiles using the chosen values of c and λ to define $\gamma_{k-1}^{\perp}$ in the case when the dense initialization was used for both the computation of the unconstrained minimizer p_u^* as well as for the shape-changing component of the algorithm, which is denoted in the legend of plots in Figure 3.1 by the use of an asterisk $(*)$. The results of Figure 3.1 suggest the choice of $c = 1$ and $\lambda = \frac{1}{2}$ outperform the other chosen combinations for c and λ. In experiments not reported here, larger values of c did not appear to improve performance; for $c < 1$, performance deteriorated. Moreover, other choices for λ, such as $\lambda = \frac{3}{4}$, did not improve results beyond the choice of $\lambda = \frac{1}{2}$.

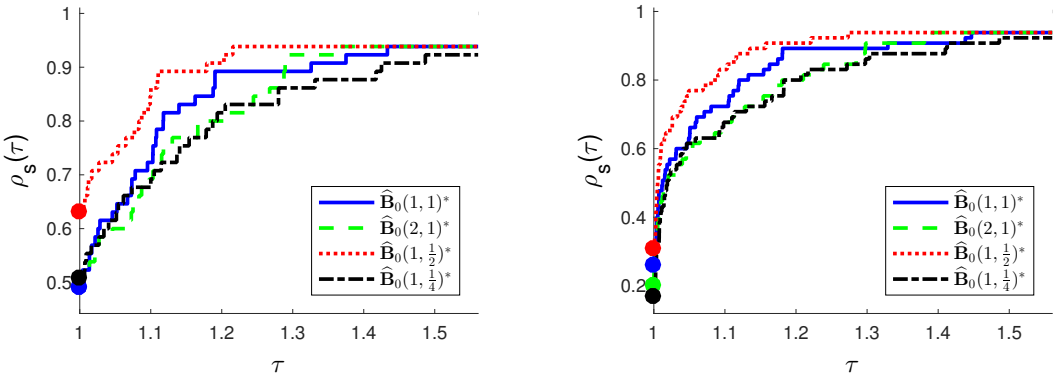


FIGURE 3.1 *Performance profiles comparing* `iter` *(left) and* `time` *(right) for the different values of $\gamma_{k-1}^{\perp}$ given in Table 3.4. In the legend, $\widehat{\mathbf{B}}_0(c,\lambda)$ denotes the results from using the dense initialization with the given values for c and λ to define $\gamma_{k-1}^{\perp}$. In this experiment, the dense initialization was used for all aspects of the algorithm.*

Figure 3.2 reports the performance profiles for using the chosen values of c and λ to define $\gamma_{k-1}^{\perp}$ in the case when the dense initialization was only used for the shape-changing component of the LMTR algorithm–denoted in the legend of plots in Figure 3.2

by the absence of an asterisk (∗). In this test, the combination of $c = 1$ and $\lambda = 1$ as well as $c = 1$ and $\lambda = \frac{1}{2}$ appear to slightly outperform the other two choices for $\gamma_k^{\perp}$ in terms of both then number of iterations and the total computational time. Based on the results in Figure 3.2, we do not see a reason to prefer either combination $c = 1$ and $\lambda = 1$ or $c = 1$ and $\lambda = \frac{1}{2}$ over the other.

Note that for the CUTEst problems, the full quasi-Newton trial step is accepted as the solution to the subproblem on the overwhelming majority of problems. Thus, if the scaling $\gamma_{k-1}^{\perp}$ is used only when the full trial step is rejected, it has less of an affect on the overall performance of the algorithm; i.e., the algorithm is less sensitive to the choice of $\gamma_{k-1}^{\perp}$. For this reason, it is not surprising that the performance profiles in Figure 3.2 for the different values of $\gamma_{k-1}^{\perp}$ are more indistinguishable than those in Figure 3.1.

Finally, similar to the results in the case when the dense initialization was used for the entire algorithm (Figure 3.1), other values of c and λ did not significantly improve the performance provided by $c = 1$ and $\lambda = \frac{1}{2}$.

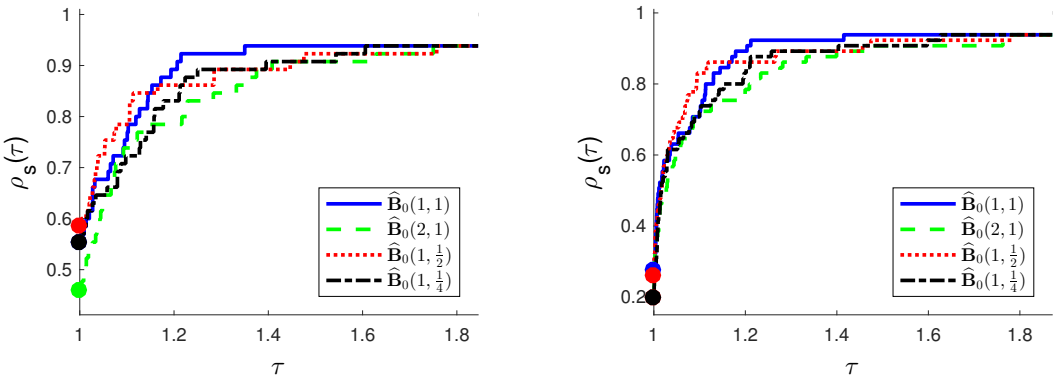


FIGURE 3.2 *Performance profiles comparing* `iter` *(left) and* `time` *(right) for the different values of* $\gamma_{k-1}^{\perp}$ *given in Table 3.4. In the legend,* $\widehat{\mathbf{B}}_0(c, \lambda)$ *denotes the results from using the dense initialization with the given values for* c *and* λ *to define* $\gamma_{k-1}^{\perp}$*. In this experiment, the dense initialization was only used for the shape-changing component of the algorithm.*

Experiment 2. This experiment was designed to test whether it is advantageous to use the dense initialization for all aspects of the LMTR algorithm or just for the shape-changing component of algorithm. For any given trust-region subproblem, using the dense initialization for computing the unconstrained minimizer is computationally more expensive than using a diagonal initialization; however, it is possible that extra computational time associated with using the dense initialization for all aspects of the LMTR algorithm may yield a more overall efficient solver. For these tests, we compare the top performer in the case when the dense initialization is used for all aspects of LMTR, i.e., $(\gamma_{k-1}^{\perp}(1, \frac{1}{2}))$, to one of the top performers in the case when the dense initialization is used only for the shape-changing component of the algorithm, i.e., $(\gamma_{k-1}^{\perp}(1, 1))$.

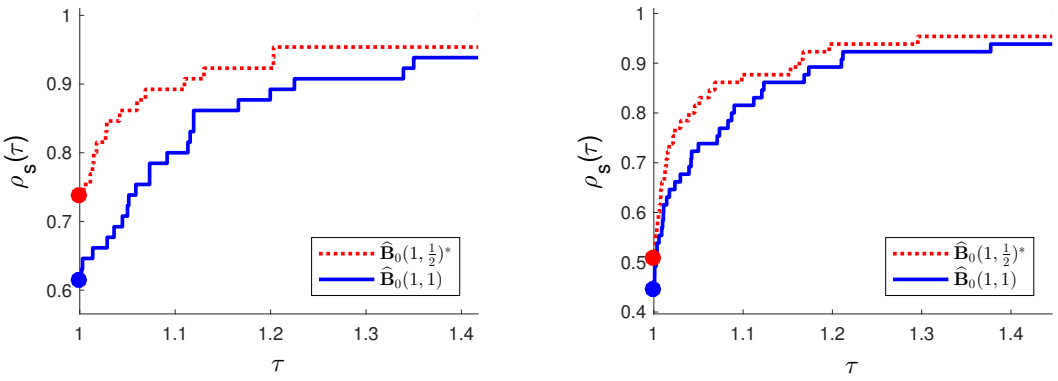


FIGURE 3.3 *Performance profiles of* `iter` *(left) and* `time` *(right) for Experiment 2. In the legend, the asterisk after* $\widehat{\mathbf{B}}_0(1,\frac{1}{2})^*$ *signifies that the dense initialization was used for all aspects of the LMTR algorithm; without the asterisk,* $\widehat{\mathbf{B}}_0(1,1)$ *signifies the test where the dense initialization is used only for the shape-changing component of the algorithm.*

The performance profiles comparing the results of this experiment are presented in Figure 3.3. These results suggest that using the dense initialization with $\gamma_{k-1}^{\perp}(1,\frac{1}{2})$ for all aspects of the LMTR algorithm is more efficient than using dense initializations only for the shape-changing component of the algorithm. In other words, even though using dense initial matrices for the computation of the unconstrained minimizer imposes an additional computational burden, it generates steps that expedite the convergence of the overall trust-region method.

Experiment 3. As noted in Section 3.2.2, a partial SVD may be used in place of a partial QR decomposition to derive alternative formulas for computing products with $\mathbf{P}_{\|}$. Specifically, if the SVD of $\boldsymbol{\Psi}_k^T\boldsymbol{\Psi}_k$ is given by $\mathbf{W}\boldsymbol{\Sigma}^2\mathbf{W}^T$ and the SVD of $\boldsymbol{\Sigma}\mathbf{W}^T\mathbf{M}_k^{-1}\mathbf{W}\boldsymbol{\Sigma}$ is given by $\mathbf{G}\hat{\boldsymbol{\Lambda}}\mathbf{G}^T$, then $\mathbf{P}_{\|}$ can be written as follows:

$$\mathbf{P}_{\|} = \boldsymbol{\Psi}_k\mathbf{W}\boldsymbol{\Sigma}^{-1}\mathbf{G}. \tag{3.39}$$

Alternatively, in [Lu96], $\mathbf{P}_{\|}$ is written as

$$\mathbf{P}_{\|} = \boldsymbol{\Psi}_k\mathbf{M}_k^{-1}\mathbf{W}\boldsymbol{\Sigma}\mathbf{G}\hat{\boldsymbol{\Lambda}}^{-1}. \tag{3.40}$$

Note that both of the required SVD computations for this approach involve $r\times r$ matrices, where $r \leq 2l \ll n$.

For this experiment, we consider LMTR with the dense initialization $\widehat{\mathbf{B}}_0(1,\frac{1}{2})^*$ used for all aspects of the algorithm (i.e., the top performer in Experiment 2). We compare an implementation of this method using the QR decomposition to compute products with $\mathbf{P}_{\|}$ to the two SVD-based methods. The results of this experiment given in Figure 3.4 suggest that the QR decomposition outperforms the two SVD decompositions in

terms of both the number of iterations and time. (Note that the QR factorization was used for both Experiments 1 and 2.)

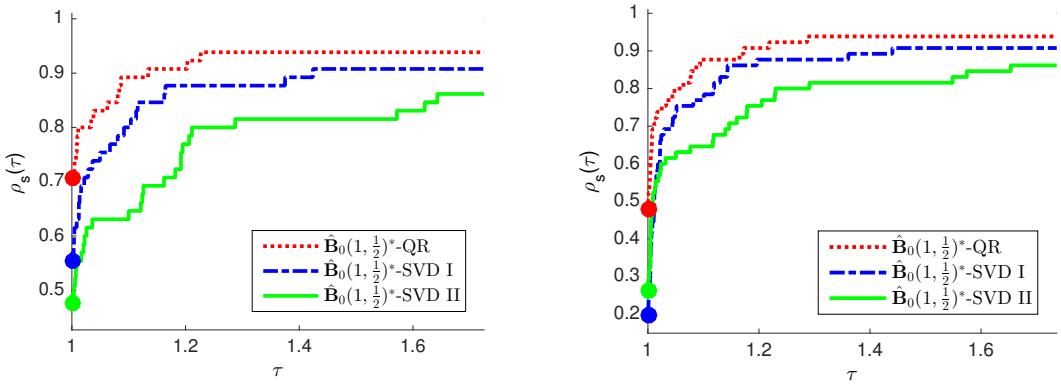


FIGURE 3.4 *Performance profiles of* `iter` *(left) and* `time` *(right) for Experiment 3 comparing three formulas for computing products with* $\mathbf{P}_{\|}$. *In the legend,* "QR" *denotes results using (3.8),* "SVD I" *denotes results using (3.39), and* "SVD II" *denotes results using (3.40). These results used the dense initialization with* $\gamma_{k-1}^{\perp}(1,\frac{1}{2})$.

Experiment 4. In this experiment, we compare the performance of the dense initialization $\gamma_{k-1}^{\perp}(1, 0.5)$ to that of the line-search L-BFGS algorithm. For this comparison, we used the publicly-available MATLAB wrapper [Bec15] for the FORTRAN L-BFGS-B code developed by Nocedal et al. [ZBN97]. The initialization for L-BFGS-B is $B_0 = \gamma_{k-1}I$ where γ_{k-1} is given by (3.4). To make the stopping criterion equivalent to that of L-BFGS-B, we modified the stopping criterion of our solver to [ZBN97]:

$$\|\mathbf{g}_k\|_\infty \leq \epsilon.$$

The dense initialization was used for all aspects of LMTR.

The performance profiles for this experiment is given in Figure 3.5. On this test set, the dense initialization outperforms L-BFGS-B in terms of both the number of iterations and the total computational time.

Experiment 5. In this experiment, we compare LMTR with a dense initialization to L-BFGS-TR [BGYZ16], which computes an L-BFGS trial step whose length is bounded by a trust-region radius. This method can be viewed as a hybrid L-BFGS line search and trust-region algorithm because it uses a standard trust-region framework (as LMTR) but computes a trial point by minimizing the quadratic model in the trust region along the L-BFGS direction. In [BGYZ16], it was determined that this algorithm outperforms two other versions of L-BFGS that use a Wolfe line search. (For further details, see [BGYZ16].)

Figure 3.6 displays the performance profiles associated with this experiment on the

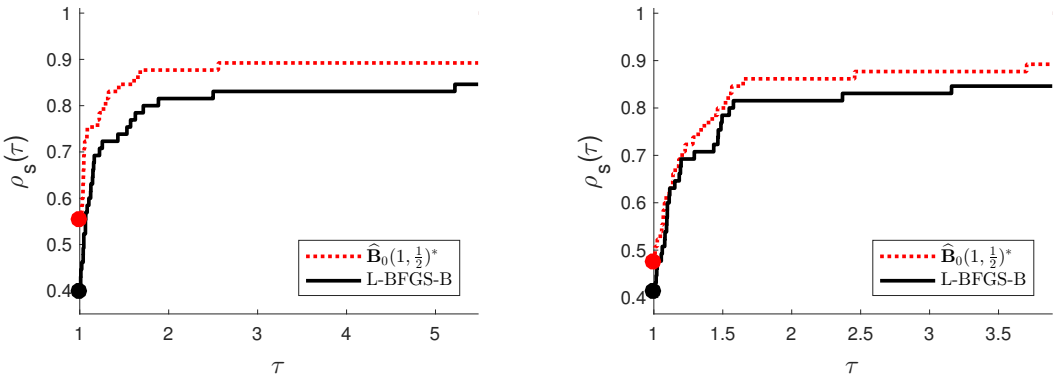


FIGURE 3.5 *Performance profiles of* `iter` *(left) and* `time` *(right) for Experiment 4 comparing LMTR with the dense initialization with* $\gamma_{k-1}^{\perp}(1,\frac{1}{2})$ *to L-BFGS-B.*

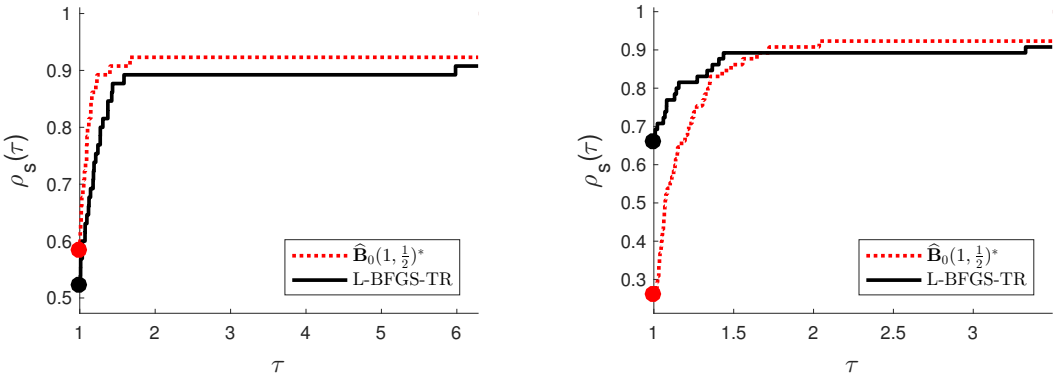


FIGURE 3.6 *Performance profiles of* `iter` *(left) and* `time` *(right) for Experiment 5 comparing LMTR with the dense initialization with* $\gamma_{k-1}^{\perp}(1,\frac{1}{2})$ *to L-BFGS-TR.*

entire set of test problems. For this experiment, the dense initialization with $\gamma_{k-1}^{\perp}(1,\frac{1}{2})$ was used in all aspects of the LMTR algorithm. In terms of total number of iterations, LMTR with the dense initialization outperformed L-BFGS-TR; however, L-BFGS-TR appears to have outperformed LMTR with the dense initialization in computational time.

Figure 3.6 (left) indicates that the quality of the trial points produced by solving the trust-region subproblem exactly using LMTR with the dense initialization is generally better than in the case of the line search applied to the L-BFGS direction. However, Figure 3.6 (right) shows that LMTR with the dense initialization requires more computational effort than L-BFGS-TR. For the CUTEst set of test problems, L-BFGS-TR does not need to perform a line search for the majority of iterations; that is, the full quasi-Newton trial step is accepted in a majority of the iterations. Therefore, we also compared the two algorithms on a subset of the most difficult test problems–namely, those for which an *active* line search is needed to be performed by L-BFGS-TR. To this end, we select, as in [BGYZ16], those of the CUTEst problems in which the full L-BFGS

(i.e., the step size of one) was rejected in at least 30% of the iterations. The number of problems in this subset is 14. The performance profiles associated with this reduced test set are in Figure 3.7. On this smaller test set, LMTR outperforms L-BFGS-TR both in terms of total number of iterations and computational time.

Finally, Figures 3.6 and 3.7 suggest that when function and gradient evaluations are expensive (e.g., simulation-based applications), LMTR together with the dense initialization is expected to be more efficient than L-BFGS-TR since both on both test sets LMTR with the dense initialization requires fewer overall iterations. Moreover, Figure 3.7 suggests that on problems where the L-BFGS search direction often does not provide sufficient decrease of the objective function, LMTR with the dense initialization is expected to perform better.

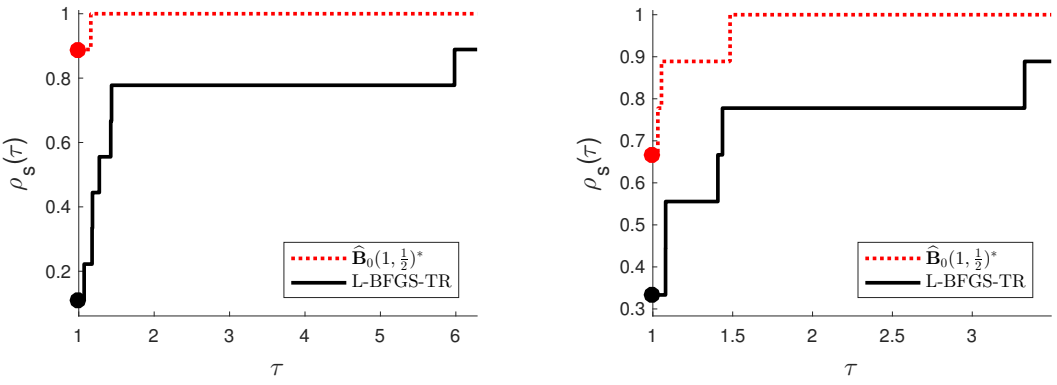


FIGURE 3.7 *Performance profiles of* `iter` *(left) and* `time` *(right) for Experiment 5 comparing LMTR with the dense initialization with* $\gamma_{k-1}^{\perp}(1,\frac{1}{2})$ *to L-BFGS-TR on the subset of 14 problems for which L-BFGS-TR implements a line search more than* 30% *of the iterations.*

Experiment 6. In this experiment, we compare the results of LMTR using the dense initialization to that of LMTR using the conventional diagonal initialization $\mathbf{B}_0 = \gamma_{k-1}\mathbf{I}_n$ where γ_{k-1} is given by (3.3). The dense initialization selected was chosen to be the top performer from Experiment 2 (i.e., $\gamma_{k-1}^{\perp}(1,\frac{1}{2})$), and the QR factorization was used to compute products with $\mathbf{P}_{\|}$.

From Figure 3.8, the dense initialization with $\gamma_{k-1}^{\perp}(1,\frac{1}{2})$ outperforms the conventional initialization for LMTR in terms of iteration count; however, it is unclear whether the algorithm benefits from the dense initialization in terms of computational time. The reason for this is that the dense initialization is being used for all aspects of the LMTR algorithm; in particular, it is being used to compute the full quasi-Newton step p_u^* (see the discussion in Experiment 1), which is typically accepted most iterations on the CUTEst test set. Therefore, as in Experiment 5, we compared LMTR with the dense initialization and the conventional initialization on the subset of 14 problems in which the unconstrained minimizer is rejected at least 30% of the iterations. The performance

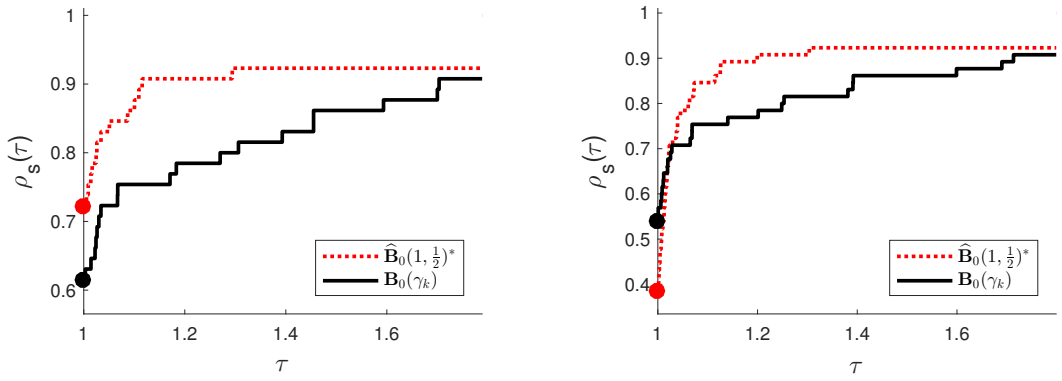


FIGURE 3.8 *Performance profiles of* `iter` *(left) and* `time` *(right) for Experiment 6 comparing LMTR with the dense initialization with* $\gamma_{k-1}^{\perp}(1,\frac{1}{2})$ *to LMTR with the conventional initialization.*

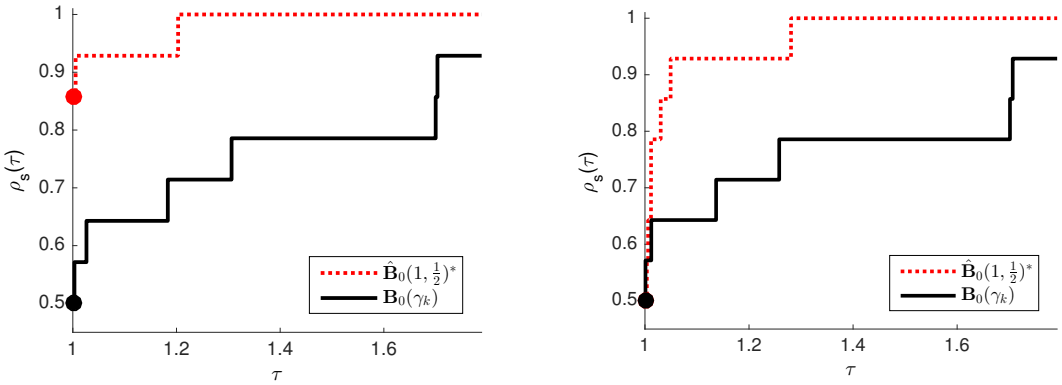


FIGURE 3.9 *Performance profiles of* `iter` *(left) and* `time` *(right) for Experiment 6 comparing LMTR with the dense initialization with* $\gamma_{k-1}^{\perp}(1,\frac{1}{2})$ *to LMTR with the conventional initialization on the subset of 14 problems in which the unconstrained minimizer is rejected at* 30% *of the iterations.*

profiles associated with this reduced set of problems are found in Figure 3.9. The results from this experiment clearly indicate that on these more difficult problems the dense initialization outperforms the conventional initialization in both iteration count and computational time.

3.5 SUMMARY

In this chapter we propose a large-scale quasi-Newton trust-region method, that uses novel initial matrices to update the quasi-Newton matrix. When the trust-region subproblems are defined by shape-changing norm, the computational costs per iteration with the proposed initial matrices are the same as with conventional multiple-of-identity initial matrices. However, unlike multiple-of-identity initial matrices, the proposed dense initial matrices distinguish two subspaces. One of these subspaces corresponds to

information generated by the quasi-Newton approximation, while in the other subspace not much is known about the objective function. By deemphasizing the significance of search directions that lie in the subspace with little information, the proposed initial matrices improve the performance of a benchmark trust-region quasi-Newton algorithm. In particular, we propose various alternatives for choices of two curvature estimates, which correspond to the two subspaces of the dense initial matrices. Finally, we develop the compact representation of quasi-Newton matrices with the proposed initial matrices. This means that the novel initializations are generally applicable to any optimization method that uses compact quasi-Newton matrices.

CHAPTER 4

THE MULTIPOINT SYMMETRIC SECANT METHOD

4.1 MOTIVATION

In this chapter we develop a large-scale quasi-Newton trust-region method, in which the quasi-Newton matrix is defined by an indefinite rank-2 update. The formula that we will analyze was independently developed in [DM77] and in [Bur83]. We will use the interpretation from [Bur83], in which a collection of secant equations –the so-called *multipoint* or *multiple* secant conditions– motivate the development of a different quasi-Newton formula. The multiple secant conditions are the generalization of the secant-condition (1.5) from Chapter 1. Instead of enforcing one equation of the form $\mathbf{B}_{k+1}\mathbf{s}_k = \mathbf{y}_k$, the multiple secant equations specify the system of conditions

$$\mathbf{B}_{k+1}\,[\ \mathbf{s}_k\ \ \mathbf{s}_{k-1}\ \ \cdots\ \ \mathbf{s}_0\ ] = [\ \mathbf{y}_k\ \ \mathbf{y}_{k-1}\ \ \cdots\ \ \mathbf{y}_0\ ]\,. \tag{4.1}$$

Therefore the quasi-Newton matrix that is based on the equations (4.1) will be denoted by *multipoint symmetric secant matrix* (MSSM).

4.2 THE MSSM QUASI-NEWTON MATRIX

In this section we apply the convention from [Bur83], in which the matrices $\mathbf{S}_k$ and $\mathbf{Y}_k$ are defined as

$$\mathbf{S}_k = [\ \mathbf{s}_{k-1}\ \ \mathbf{s}_{k-2}\ \ \cdots\ \ \mathbf{s}_0\ ] \quad \text{and} \quad \mathbf{Y}_k = [\ \mathbf{y}_{k-1}\ \ \mathbf{y}_{k-2}\ \ \cdots\ \ \mathbf{y}_0\ ]\,.$$

Moreover, we assume that $\mathbf{S}_k \in \mathbb{R}^{n\times k}$ and $\mathbf{Y}_k \in \mathbb{R}^{n\times k}$ are both of full column rank. Until otherwise noted, suppose that $k = n$, which means that $\mathbf{S}_k$ and $\mathbf{Y}_k$ are both invertible. The idea that a quasi-Newton matrix should satisfy multiple-secant-conditions of the form $\mathbf{B}_k\mathbf{S}_k = \mathbf{Y}_k$ is desirable, but typically not possible. A practical difficulty is that the multiple-secant-conditions and the requirement of symmetry of the quasi-Newton matrix are under normal circumstances inconsistent. In fact, the necessary condition for the existence of a symmetric matrix $\mathbf{B}_k$, which satisfies the multiple-secant-conditions is that the product $\mathbf{S}_k^T\mathbf{Y}_k$ is symmetric [Sch83], i.e.,

$$\mathbf{S}_k^T\left(\mathbf{B}_k\mathbf{S}_k\right) = \mathbf{S}_k^T\left(\mathbf{Y}_k\right). \tag{4.2}$$

However, typically $\mathbf{S}_k^T\mathbf{Y}_k$ is not symmetric. Therefore an approach was developed in [Bur83], which proposes a compromise between these opposing conditions. In particular, a symmetrization transformation is applied to $\mathbf{S}_k^T\mathbf{Y}_k$, in order to guarantee symmetry. The symmetrization transformation, however, comes at the cost of only approximately satisfying the multiple-secant-conditions. We now review the approach of the MSSM matrix.

Let $\mathbf{S}_k^T\mathbf{Y}_k$ be decomposed as

$$\mathbf{S}_k^T\mathbf{Y}_k = \mathbf{L}_k + \mathbf{E}_k + \mathbf{T}_k, \tag{4.3}$$

where $\mathbf{L}_k$ is the strictly lower triangular matrix, $\mathbf{E}_k$ is the diagonal matrix, and $\mathbf{T}_k$ is the strictly upper triangular matrix of $\mathbf{S}_k^T\mathbf{Y}_k$. The three matrices $\mathbf{L}_k, \mathbf{E}_k$ and $\mathbf{T}_k$ are each of dimension $\mathbb{R}^{k\times k}$. With this, the symmetrization transformation applied to $\mathbf{S}_k^T\mathbf{Y}_k$ is defined as

$$\operatorname{sym}\left(\mathbf{S}_k^T\mathbf{Y}_k\right) = \operatorname{sym}\left(\mathbf{L}_k + \mathbf{E}_k + \mathbf{T}_k\right) \equiv \mathbf{L}_k + \mathbf{E}_k + \mathbf{L}_k^T = \mathbf{\Gamma}_k, \tag{4.4}$$

where $\mathbf{L}_k + \mathbf{E}_k + \mathbf{L}_k^T = \mathbf{\Gamma}_k$, is used for notation only. The effect of the symmetrization transformation is that the elements below the main diagonal of $\mathbf{S}_k^T\mathbf{Y}_k$ are copied into the elements above the main diagonal. Based on the symmetrization, the MSS matrix is determined as the quasi-Newton matrix, $\mathbf{B}_k$, which satisfies the system

$$\mathbf{S}_k^T\left(\mathbf{B}_k\mathbf{S}_k\right) = \operatorname{sym}\left(\mathbf{S}_k^T\mathbf{Y}_k\right) = \mathbf{\Gamma}_k. \tag{4.5}$$

Subsequently, the unique matrix $\mathbf{B}_k$, is represented from (4.5) as

$$\mathbf{B}_k = \mathbf{S}_k^{-T}\mathbf{\Gamma}_k\mathbf{S}_k^{-1}. \tag{4.6}$$

4.2.1 THE UPDATE FORMULA

In the previous section it was assumed that $k = n$, which resulted in the formula (4.6) to compute $\mathbf{B}_k = \mathbf{B}_n$. The recursive update formula of the MSSM is obtained by analyzing the transition from $k = n$ to the next iterate $k+1 = n+1$. In particular, let $\mathbf{S}_{k+1} \in \mathbb{R}^{n\times n}$ and $\mathbf{Y}_{k+1} \in \mathbb{R}^{n\times n}$ be represented as

$$\mathbf{S}_{k+1} = \begin{bmatrix} \mathbf{s}_k & \mathbf{s}_{k-1} & \cdots & \mathbf{s}_1 \end{bmatrix} \quad \text{and} \quad \mathbf{Y}_{k+1} = \begin{bmatrix} \mathbf{y}_k & \mathbf{y}_{k-1} & \cdots & \mathbf{y}_1 \end{bmatrix}.$$

A matrix factorization is used to express $\mathbf{S}_{k+1}$ and $\mathbf{Y}_{k+1}$ in terms of $\mathbf{S}_k, \mathbf{Y}_k, \mathbf{s}_k$ and $\mathbf{y}_k$. In particular, with the permutation matrix $\bar{\mathbf{P}} = \begin{bmatrix} \mathbf{e}_n & \mathbf{e}_1 & \mathbf{e}_2 & \cdots & \mathbf{e}_{n-1} \end{bmatrix}$ (here $\mathbf{e}_i$ represents the ith column of $\mathbf{I}_n$), which inserts the last column into the first, and shifts all other columns one index to the right, then $\mathbf{S}_{k+1}$ is represented as

$$\mathbf{S}_{k+1} = \begin{bmatrix} \mathbf{s}_{k-1} & \mathbf{s}_{k-2} & \cdots & \mathbf{s}_1 & \mathbf{s}_k \end{bmatrix} \bar{\mathbf{P}} = \left(\mathbf{S}_k - \mathbf{s}_0\mathbf{e}_n^T + \mathbf{s}_k\mathbf{e}_n^T\right)\bar{\mathbf{P}} = \left(\mathbf{S}_k + (\mathbf{s}_k - \mathbf{S}_k\mathbf{e}_n)\mathbf{e}_n^T\right)\bar{\mathbf{P}}.$$

This expression is a consequence of the observation that the rank-1 matrix $(\mathbf{s}_k - \mathbf{S}_k\mathbf{e}_n)\mathbf{e}_n^T$ only has one non-zero column, i.e.,

$$(\mathbf{s}_k - \mathbf{S}_k\mathbf{e}_n)\mathbf{e}_n^T = \begin{bmatrix} & & \\ \mathbf{0}_{n\times(n-1)} & & (\mathbf{s}_k - \mathbf{s}_0) \\ & & \end{bmatrix}.$$

In an analogous way, the matrix $\mathbf{Y}_{k+1}$ is obtained as an update of $\mathbf{Y}_k$

$$\mathbf{Y}_{k+1} = \left(\mathbf{Y}_k + (\mathbf{y}_k - \mathbf{Y}_k\mathbf{e}_n)\mathbf{e}_n^T\right)\bar{\mathbf{P}}.$$

Since $\mathbf{S}_{k+1}$ and $\mathbf{Y}_{k+1}$ are represented in terms of rank-1 modifications, their inverses are analytically computed by the Sherman-Morrison-Woodbury formula. The recursive update formula of the MSS matrix is subsequently based on substituting $\mathbf{S}_{k+1}$ and $\mathbf{Y}_{k+1}$ into the right hand side of (4.6), that is

$$\mathbf{B}_{k+1} = \left(\left(\mathbf{S}_k + (\mathbf{s}_k - \mathbf{S}_k\mathbf{e}_n)\mathbf{e}_n^T\right)\bar{\mathbf{P}}\right)^{-T} \mathbf{\Gamma}_{k+1} \left(\left(\mathbf{S}_k + (\mathbf{s}_k - \mathbf{S}_k\mathbf{e}_n)\mathbf{e}_n^T\right)\bar{\mathbf{P}}\right)^{-1},$$

where $\mathbf{\Gamma}_{k+1} = \operatorname{sym}\left(\mathbf{S}_{k+1}^T\mathbf{Y}_{k+1}\right)$. We describe the derivation of the recursive MSSM update formula in the Appendix (see Section A). Here we state that for a vector $\mathbf{c}_k$, which satisfies $\mathbf{c}_k^T\mathbf{S}_k = \mathbf{0}$ and $\mathbf{c}_k^T\mathbf{s}_k \neq 0$, the recursive formula of the MSS matrix is defined by the rank-2 update

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \frac{1}{\mathbf{s}_k^T\mathbf{c}_k}\left(\mathbf{c}_k(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)^T + (\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)\mathbf{c}_k^T\right) - \frac{1}{(\mathbf{s}_k^T\mathbf{c}_k)^2}((\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)^T\mathbf{s}_k)\mathbf{c}_k\mathbf{c}_k^T. \tag{4.7}$$

The MSSM is closely related to the symmetric rank-1 matrix (SR1). In fact, it can be interpreted as a generalization of the SR1. Namely, if the vector $\tilde{\mathbf{c}}_k = \mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k$, satisfies $\tilde{\mathbf{c}}_k^T\mathbf{S}_k = \mathbf{0}$ and $\tilde{\mathbf{c}}_k^T\mathbf{s}_k \neq 0$ then the formula in (4.7) with $\mathbf{c}_k = \tilde{\mathbf{c}}_k$ reduces to the formula of the SR1 update $\mathbf{B}_{k+1} = \mathbf{B}_k + \frac{1}{\mathbf{s}_k^T(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)}(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)^T$. Similar to the SR1 update, the MSS formula also results in indefinite quasi-Newton matrices.

4.2.2 THE MSSM COMPACT REPRESENTATION

The compact representation of the MSS matrix was originally developed in [BMP02], based on an approach that is different from the ones found in the literature [BNS94, DEM16]. First, we suppose that $k < n$, and that the subsequent MSS matrices are generated by the formula (4.7). Next let the orthogonal complement of $\mathbf{S}_k$ take the representation

$$\mathbf{S}_k^{\perp} \equiv \mathbf{C}_k = \begin{bmatrix} \mathbf{c}_k{}^{(1)} & \cdots & \mathbf{c}_k{}^{(n-k)} \end{bmatrix} \in \mathbb{R}^{n\times(n-k)},$$

where $\mathbf{c}_k{}^{(1)} = \mathbf{c}_k/\|\mathbf{c}_k\|_2$. Since $\mathbf{C}_k$ is the orthogonal complement of $\mathbf{S}_k$ it satisfies the equations

$$\mathbf{S}_k^T\mathbf{C}_k = \mathbf{0}_{k\times(n-k)} \quad \text{and} \quad \mathbf{C}_k^T\mathbf{C}_k = \mathbf{I}_{n-k}.$$

Moreover, since $\mathbf{s}_k^T\mathbf{c}_k \neq 0$ by the definition of (4.7), then

$$\mathbf{S}_{k+1}^T\mathbf{c}_k = \begin{bmatrix} \mathbf{s}_k & \mathbf{S}_k \end{bmatrix}^T \mathbf{c}_k = \begin{bmatrix} \mathbf{s}_k^T\mathbf{c}_k \\ 0 \\ \vdots \\ 0 \end{bmatrix} = (\mathbf{s}_k^T\mathbf{c}_k)\mathbf{e}_1.$$

Thus, at the previous iteration $(k-1)$, the equality $\mathbf{S}_k^T\mathbf{c}_{k-1} = (\mathbf{s}_{k-1}^T\mathbf{c}_{k-1})\mathbf{e}_1$ held. With these definitions, the MSS matrices from (4.7) satisfy a set of matrix identities. We will briefly review the properties of MSS matrices because they are the basis to the compact representation. Additionally, we will use the properties of the MSS matrices in order to develop a novel method of solving multipoint symmetric secant trust-region subproblems. The first matrix identities satisfied by $\mathbf{B}_k$ are

$$\mathbf{S}_k^T\mathbf{B}_k\mathbf{S}_k = \mathbf{\Gamma}_k, \qquad 0 < k < n.$$

This identity is the result of applying $\mathbf{S}_k^T$ and $\mathbf{S}_k$ to the recursive formula in (4.7), i.e.,

$$\begin{aligned}\mathbf{S}_k^T\mathbf{B}_k\mathbf{S}_k &= \mathbf{S}_k^T\mathbf{B}_{k-1}\mathbf{S}_k + \mathbf{e}_1(\mathbf{y}_{k-1} - \mathbf{B}_{k-1}\mathbf{s}_{k-1})^T\mathbf{S}_k + \mathbf{S}_k^T(\mathbf{y}_{k-1} - \mathbf{B}_{k-1}\mathbf{s}_{k-1})\mathbf{e}_1^T \\ &\quad - (\mathbf{y}_{k-1} - \mathbf{B}_{k-1}\mathbf{s}_{k-1})^T\mathbf{s}_{k-1}\mathbf{e}_1\mathbf{e}_1^T \\ &= \begin{bmatrix} \mathbf{s}_{k-1}^T\mathbf{y}_{k-1} & \multicolumn{2}{c}{\longrightarrow\ \mathbf{y}_{k-1}^T\mathbf{S}_{k-1}\ \longrightarrow} \\ \downarrow & & \\ \mathbf{S}_{k-1}^T\mathbf{y}_{k-1} & \multicolumn{2}{c}{\begin{bmatrix} \mathbf{S}_{k-1}^T\mathbf{B}_{k-1}\mathbf{S}_{k-1} \end{bmatrix}} \\ \downarrow & & \end{bmatrix} \\ &= \mathbf{\Gamma}_k.\end{aligned}$$

The third equality is obtained by observing that $\mathbf{S}_{k-1}^T\mathbf{y}_{k-1} = \mathbf{L}_k\mathbf{e}_1$ and that $\mathbf{s}_{k-1}^T\mathbf{y}_{k-1} = \mathbf{e}_1^T\mathbf{E}_k\mathbf{e}_1$. Then by recursively unwinding the block $[\mathbf{S}_{k-1}^T\mathbf{B}_{k-1}\mathbf{S}_{k-1}]$, the process is repeated so that $\mathbf{S}_k^T\mathbf{B}_k\mathbf{S}_k = \mathbf{L}_k + \mathbf{E}_k + \mathbf{L}_k^T = \mathbf{\Gamma}_k$. A second set of equations that define the MSS matrix are

$$\mathbf{S}_k^T\mathbf{B}_k\mathbf{C}_k = \mathbf{Y}_k^T\mathbf{C}_k, \qquad 0 < k < n.$$

These equations are obtained by similar recursive arguments as before, i.e. $\mathbf{S}_k^T$ and $\mathbf{C}_k$ are applied to the recursive formula in (4.7), so that

$$\begin{aligned}\mathbf{S}_k^T\mathbf{B}_k\mathbf{C}_k &= \mathbf{S}_k^T\left(\mathbf{B}_{k-1}\mathbf{C}_k + \frac{1}{\mathbf{s}_{k-1}^T\mathbf{c}_{k-1}}\mathbf{c}_{k-1}(\mathbf{y}_{k-1} - \mathbf{B}_{k-1}\mathbf{s}_{k-1})^T\mathbf{C}_k\right) \\ &= \mathbf{S}_k^T\mathbf{B}_{k-1}\mathbf{C}_k + \mathbf{e}_1(\mathbf{y}_{k-1} - \mathbf{B}_{k-1}\mathbf{s}_{k-1})^T\mathbf{C}_k \\ &= \begin{bmatrix} \mathbf{y}_{k-1}^T\mathbf{C}_k \\ \mathbf{S}_{k-1}^T\mathbf{B}_{k-1}\mathbf{C}_k \end{bmatrix} \\ &= \mathbf{Y}_k^T\mathbf{C}_k.\end{aligned}$$

The last equality is the result of an induction on $\mathbf{S}_{k-1}^T\mathbf{B}_{k-1}\mathbf{C}_k$. Finally, applying $\mathbf{C}_k^T$ and $\mathbf{C}_k$ to the recursive formula in (4.7) and using an induction, then the identity $\mathbf{C}_k^T\mathbf{B}_k\mathbf{C}_k = \mathbf{C}_k^T\mathbf{B}_0\mathbf{C}_k = \gamma_k\mathbf{I}_{n-k}$ is established. Collecting the properties of the MSSM , then the multipoint symmetric secant matrix satisfies the system of matrix equations

$$\begin{bmatrix} \mathbf{S}_k & \mathbf{C}_k \end{bmatrix}^T\mathbf{B}_k\begin{bmatrix} \mathbf{S}_k & \mathbf{C}_k \end{bmatrix} = \begin{bmatrix} \mathbf{\Gamma}_k & \mathbf{Y}_k^T\mathbf{C}_k \\ \mathbf{C}_k^T\mathbf{Y}_k & \mathbf{C}_k^T\mathbf{B}_0\mathbf{C}_k \end{bmatrix}. \tag{4.8}$$

The application of the inverse $\begin{bmatrix} \mathbf{S}_k & \mathbf{C}_k \end{bmatrix}^{-T} = \begin{bmatrix} \mathbf{S}_k(\mathbf{S}_k^T\mathbf{S}_k)^{-1} & \mathbf{C}_k \end{bmatrix}$ to the system in (4.8) is a key step in deriving the MSS compact representation. We describe the details of the derivation in the Appendix (see Section B). Here we report that the compact

representation of the MSSM is given by

$$\mathbf{B}_k = \mathbf{B}_0 + \mathbf{\Psi}_k \mathbf{M}_k \mathbf{\Psi}_k^T, \tag{4.9}$$

where we define $(\mathbf{S}_k^T \mathbf{S}_k)^{-1} \equiv \mathbf{\Xi}_k$ and

$$\mathbf{\Psi}_k \equiv [\ \mathbf{S}_k\ \ (\mathbf{Y}_k - \mathbf{B}_0 \mathbf{S}_k)\ ], \quad \mathbf{M}_k \equiv \begin{bmatrix} \mathbf{\Xi}_k(\mathbf{S}_k^T \mathbf{B}_0 \mathbf{S}_k - (\mathbf{T}_k + \mathbf{E}_k + \mathbf{T}_k^T))\mathbf{\Xi}_k & \mathbf{\Xi}_k \\ \mathbf{\Xi}_k & \mathbf{0}_{k\times k} \end{bmatrix}. \tag{4.10}$$

We will use the limited-memory version of the MSS matrix, so that $\mathbf{\Psi}_k$ is of dimension $\mathbb{R}^{n\times 2l}$ and $\mathbf{M}_k$ is of dimension $\mathbb{R}^{2l\times 2l}$.

4.3 SOLVING THE MSSM TRUST-REGION SUBPROBLEM

The MSS matrix, such as the SR1 matrix, is indefinite. Therefore a reasonable approach is to extend the two methods from Chapter 2, i.e., the OBS method, and the SC-SR1 method, to the L-MSS matrix. In particular the MSSM method uses a rank-2 quasi-Newton update formula, instead of the rank-1 formula of the SR1. We will develop two approaches to solve trust-region subproblems of the form (1.4), where $\mathbf{B}_k$ is the multipoint-symmetric-secant matrix. In the first approach, we use a partial eigendecomposition of the compact representation of the MSSM. This approach has the advantage that subproblems defined by the shape-changing norms are solved as in Chapter 2. Alternatively, if the trust-region subproblems are defined by the ℓ_2-norm, then the properties of the MSS matrix, cf. (4.8), can be used to solve the trust-region subproblem by only computing one eigenvalue. This second approach may be more attractive, when computing the partial eigendecomposition is difficult to do. This may be the case when the number of stored limited memory pairs $\{\mathbf{s}_{k-i}, \mathbf{y}_{k-i}\}_{i=1}^{l}$ grows large.

Approach I. In this approach we factor the compact representation of the L-MSSM, using an implicit spectral decomposition (cf. Subsection 1.4.3), so that

$$\mathbf{B}_k = \mathbf{B}_0 + \mathbf{\Psi}_k \mathbf{M}_k \mathbf{\Psi}_k^T = \mathbf{P}\mathbf{\Lambda}\mathbf{P}^T, \tag{4.11}$$

where the initial matrix, $\mathbf{B}_0$, is either a multiple of the identity matrix, or a dense initialization as proposed in Chapter 3. We will use an initial matrix of the form $\mathbf{B}_0 = \gamma_k \mathbf{I}$, in this section. By the change of variables $\mathbf{v} = \mathbf{P}^T\mathbf{s}$, the trust-region subproblem is transformed into a subproblem that is easier to solve, i.e.,

$$\underset{\|\mathbf{s}\|\le\Delta_k}{\text{minimize}}\ \mathbf{g}_k^T\mathbf{s} + \frac{1}{2}\mathbf{s}^T\mathbf{B}_k\mathbf{s} \quad = \quad \underset{\|\mathbf{P}\mathbf{v}\|\le\Delta_k}{\text{minimize}}\ \mathbf{g}_k^T(\mathbf{P}\mathbf{v}) + \frac{1}{2}\mathbf{v}^T\mathbf{\Lambda}\mathbf{v}.$$

Depending on the choice of norm, then the solutions $\mathbf{s}^*$ to the trust-region subproblems are either as described in Algorithm 2.2 (cf. Section 2.4.2), when the ℓ_2-norm is used, or as described in Section 2.5.7, when the shape-changing norms are used. This approach to solve the subproblem is very effective, as long as the implicit spectral decomposition of $\mathbf{B}_k$ can be computed efficiently.

Approach II. An alternative solution method to the ℓ_2 L-MSSM trust-region subproblem is based on the change of variables

$$\mathbf{s} = \begin{bmatrix} \mathbf{S}_k & \mathbf{C}_k \end{bmatrix} \mathbf{w}. \tag{4.12}$$

Here $\mathbf{S}_k \in \mathbb{R}^{n\times l}$ and $\mathbf{C}_k \in \mathbb{R}^{n\times(n-l)}$ represents the orthogonal complement of $\mathbf{S}_k$, i.e. $\mathbf{C}_k = \mathbf{S}_k^{\perp}$. The vector $\mathbf{w} \in \mathbb{R}^n$ is a vector of new variables. We will use the large orthonormal matrix $\mathbf{C}_k$ in the derivation of the solution, however it is not necessary to explicitly compute this matrix. Moreover, this approach does not compute the partial eigendecomposition of the L-MSSM. Instead, observe from the properties of the MSS matrix (4.8), that the trust-region objective function, has the equivalent representation

$$Q(\mathbf{s}) = \mathbf{g}_k^T\mathbf{s} + \frac{1}{2}\mathbf{s}^T\mathbf{B}_k\mathbf{s} = \mathbf{g}_k^T \begin{bmatrix} \mathbf{S}_k & \mathbf{C}_k \end{bmatrix} \mathbf{w} + \frac{1}{2}\mathbf{w}^T \begin{bmatrix} \boldsymbol{\Gamma}_k & \mathbf{Y}_k^T\mathbf{C}_k \\ \mathbf{C}_k^T\mathbf{Y}_k & \gamma_k\mathbf{I}_{n-l} \end{bmatrix} \mathbf{w} \equiv q(\mathbf{w}), \tag{4.13}$$

where the initial matrix is taken to be a multiple of the identity matrix. Next we separate $\mathbf{w}$ into two components, and define $\begin{bmatrix} \mathbf{S}_k & \mathbf{C}_k \end{bmatrix}^T \mathbf{g}_k$ also in terms of two blocks, namely we use

$$\mathbf{w} \equiv \begin{bmatrix} \mathbf{w}_{\|} \\ \mathbf{w}_{\perp} \end{bmatrix}, \quad \text{and} \quad \begin{bmatrix} \mathbf{S}_k & \mathbf{C}_k \end{bmatrix}^T \mathbf{g}_k = \begin{bmatrix} \mathbf{S}_k^T\mathbf{g}_k \\ \mathbf{C}_k^T\mathbf{g}_k \end{bmatrix} \equiv \begin{bmatrix} \mathbf{g}_{\|} \\ \mathbf{g}_{\perp} \end{bmatrix}.$$

Here the vectors $\mathbf{w}_{\|}$ and $\mathbf{g}_{\|}$ are of dimension $\mathbb{R}^l$, respectively, whereas the vectors $\mathbf{w}_{\perp}$ and $\mathbf{g}_{\perp}$ are of dimension $\mathbb{R}^{n-l}$, respectively. With these separations, the subproblem objective function becomes

$$q(\mathbf{w}) = \mathbf{g}_{\|}^T\mathbf{w}_{\|} + \mathbf{g}_{\perp}^T\mathbf{w}_{\perp} + \frac{1}{2}\left(\mathbf{w}_{\|}^T\boldsymbol{\Gamma}_k\mathbf{w}_{\|} + 2\mathbf{w}_{\|}^T\mathbf{Y}_k^T\mathbf{C}_k\mathbf{w}_{\perp} + \gamma_k\left\|\mathbf{w}_{\perp}\right\|_2^2\right) \equiv q(\mathbf{w}_{\|}, \mathbf{w}_{\perp}). \tag{4.14}$$

Moreover the square of the constraint is also expressed in terms of $\mathbf{w}_{\|}$ and $\mathbf{w}_{\perp}$, i.e.,

$$\|\mathbf{s}\|_2^2 = \|\begin{bmatrix} \mathbf{S}_k & \mathbf{C}_k \end{bmatrix}\mathbf{w}\|_2^2 = \left\|\mathbf{S}_k\mathbf{w}_{\|}\right\|_2^2 + \|\mathbf{w}_{\perp}\|_2^2,$$

where we use the orthogonality conditions $\mathbf{S}_k^T\mathbf{C}_k = \mathbf{0}_{l\times(n-l)}$. Similar as in Chapter 2, the global solution of the trust-region subproblem is characterized as the stationary

point of the Lagrangian function

$$\mathcal{L}(\mathbf{w}_{\|}, \mathbf{w}_{\perp}, \sigma) \equiv q(\mathbf{w}_{\|}, \mathbf{w}_{\perp}) + \frac{\sigma}{2}\left(\left\|\mathbf{S}_k\mathbf{w}_{\|}\right\|_2^2 + \|\mathbf{w}_{\perp}\|_2^2\right),$$

where $\sigma \geq 0$ is a lagrange multiplier. Setting the gradient of the Lagrangian at the solution equal to zero, that is, $\nabla\mathcal{L}(\mathbf{w}_{\|}^*, \mathbf{w}_{\perp}^*, \sigma^*) = \mathbf{0}$, yields the nonlinear system of equations

$$\begin{bmatrix} \left(\boldsymbol{\Gamma}_k + \sigma^*\mathbf{S}_k^T\mathbf{S}_k\right) & \mathbf{Y}_k^T\mathbf{C}_k \\ \mathbf{C}_k^T\mathbf{Y}_k & (\gamma_k + \sigma^*)\mathbf{I}_{n-l} \end{bmatrix} \begin{bmatrix} \mathbf{w}_{\|}^* \\ \mathbf{w}_{\perp}^* \end{bmatrix} = - \begin{bmatrix} \mathbf{g}_{\|} \\ \mathbf{g}_{\perp} \end{bmatrix}. \tag{4.15}$$

The solutions to the system in (4.15) are

$$\mathbf{w}_{\perp}^* = \frac{-1}{\gamma_k + \sigma^*}\mathbf{C}_k^T\left(\mathbf{g}_k + \mathbf{Y}_k\mathbf{w}_{\|}^*\right), \tag{4.16}$$

and

$$\mathbf{w}_{\|}^* = \left(\boldsymbol{\Gamma}_k + \sigma^*\mathbf{S}_k^T\mathbf{S}_k + \frac{1}{\gamma_k + \sigma^*}\mathbf{Y}_k^T\mathbf{C}_k\mathbf{C}_k^T\mathbf{Y}_k\right)^{-1}\left(-\mathbf{g}_{\|} + \frac{1}{\gamma_k + \sigma^*}\mathbf{Y}_k^T\mathbf{C}_k\mathbf{C}_k^T\mathbf{g}_k\right). \tag{4.17}$$

The matrix $\mathbf{C}_k\mathbf{C}_k^T$ is the orthogonal projection onto the nullspace of $\mathbf{S}_k$, and has the expression $\mathbf{C}_k\mathbf{C}_k^T = \mathbf{I}_n - \mathbf{S}_k(\mathbf{S}_k^T\mathbf{S}_k)^{-1}\mathbf{S}_k^T$. Therefore, once σ^* is known then $\mathbf{w}_{\|}^*$ is computed explicitly. Moreover, $\mathbf{s}^*$ is also obtained by an analytic formula

$$\begin{aligned} \mathbf{s}^* &= \begin{bmatrix} \mathbf{S}_k & \mathbf{C}_k \end{bmatrix}\mathbf{w}^* \\ &= \mathbf{S}_k\mathbf{w}_{\|}^* + \mathbf{C}_k\mathbf{w}_{\perp}^* \\ &= \mathbf{S}_k\mathbf{w}_{\|}^* - \frac{1}{\gamma_k + \sigma^*}\left(\mathbf{I}_n - \mathbf{S}_k\left(\mathbf{S}_k^T\mathbf{S}_k\right)^{-1}\mathbf{S}_k^T\right)\left(\mathbf{g}_k + \mathbf{Y}_k\mathbf{w}_{\|}^*\right). \end{aligned} \tag{4.18}$$

Computing σ^*.

In order to describe our approach to compute σ^* note from equations (4.17), and (4.18) that the optimal vectors $\mathbf{w}_{\perp}^*$ and $\mathbf{w}_{\|}^*$ are functions of the optimal scalar $\sigma^* \geq 0$. At any other nonnegative value $\sigma \geq 0$ we use the representations

$$\mathbf{w}_{\perp} = \mathbf{w}_{\perp}(\sigma) = \frac{-1}{\gamma_k + \sigma}\mathbf{C}_k^T\left(\mathbf{g}_k + \mathbf{Y}_k\mathbf{w}_{\|}\right),$$

and

$$\mathbf{w}_{\|} = \mathbf{w}_{\|}(\sigma) = \left(\boldsymbol{\Gamma}_k + \sigma\mathbf{S}_k^T\mathbf{S}_k + \frac{1}{\gamma_k + \sigma}\mathbf{Y}_k^T\mathbf{C}_k\mathbf{C}_k^T\mathbf{Y}_k\right)^{-1}\left(-\mathbf{g}_{\|} + \frac{1}{\gamma_k + \sigma}\mathbf{Y}_k^T\mathbf{C}_k\mathbf{C}_k^T\mathbf{g}_k\right).$$

Moreover we define

$$\mathbf{G}_{\|} = \mathbf{G}_{\|}(\sigma) \equiv (\gamma_k + \sigma)\boldsymbol{\Gamma}_k + \sigma(\gamma_k + \sigma)\mathbf{S}_k^T\mathbf{S}_k + \mathbf{Y}_k^T\mathbf{C}_k\mathbf{C}_k^T\mathbf{Y}_k,$$

and

$$\bar{\mathbf{g}}_{\|} = \bar{\mathbf{g}}_{\|}(\sigma) \equiv \left(-(\gamma_k + \sigma)\mathbf{g}_{\|} + \mathbf{Y}_k^T \mathbf{C}_k \mathbf{C}_k^T \mathbf{g}_k\right),$$

so that $\mathbf{G}_{\|}(\sigma)\mathbf{w}_{\|}(\sigma) = \bar{\mathbf{g}}_{\|}(\sigma)$. Now let the secular equation be expressed as

$$\phi(\sigma) = \frac{1}{\sqrt{\psi(\sigma)}} - \frac{1}{\Delta_k},$$

where

$$\psi = \psi(\sigma) \equiv \|\mathbf{s}\|_2^2 = \left\|\mathbf{S}_k \mathbf{w}_{\|}\right\|_2^2 + \|\mathbf{w}_{\perp}\|_2^2.$$

Then the optimal σ^* will be a root of $\psi(\sigma)$, i.e., $\psi(\sigma^*) = 0$. Observe that

$$\frac{d\psi(\sigma)}{d\sigma} = 2\mathbf{w}_{\|}^T \mathbf{S}_k^T \mathbf{S}_k \mathbf{w}_{\|}' + 2\mathbf{w}_{\perp}^T \mathbf{w}_{\perp}',$$

where $\mathbf{w}_{\|}' = \frac{d\mathbf{w}_{\|}(\sigma)}{d\sigma}$ and $\mathbf{w}_{\perp}' = \frac{d\mathbf{w}_{\perp}(\sigma)}{d\sigma}$. Taking derivatives of $\mathbf{G}_{\|}\mathbf{w}_{\|} = \bar{\mathbf{g}}_{\|}$, then

$$\mathbf{G}_{\|}'\mathbf{w}_{\|} + \mathbf{G}_{\|}\mathbf{w}_{\|}' = -\bar{\mathbf{g}}_{\|}' \quad \text{implies} \quad \mathbf{w}_{\|}' = -\mathbf{G}_{\|}^{-1}\left(\bar{\mathbf{g}}_{\|}' + \mathbf{G}_{\|}'\mathbf{w}_{\|}\right),$$

where $\mathbf{G}_{\|}'$ and $\mathbf{g}_{\|}'$ represent the derivatives of $\mathbf{G}_{\|}$ and $\mathbf{g}_{\|}$ with respect to σ,

$$\mathbf{G}_{\|}' = \gamma_k \mathbf{\Gamma}_k + (\gamma_k + 2\sigma)\mathbf{S}_k^T \mathbf{S}_k, \quad \text{and} \quad \mathbf{g}_{\|}' = -\gamma_k \mathbf{g}_{\|}.$$

In other words, after computing $\mathbf{w}_{\|}$, then $\mathbf{w}_{\|}'$ can be found. Similarly we compute

$$\mathbf{w}_{\perp}' = \frac{1}{(\gamma_k + \sigma)^2}\mathbf{C}_k^T\left((1 - (\gamma_k + \sigma))\mathbf{g}_k + \mathbf{Y}_k\left(\mathbf{w}_{\|} - (\gamma_k + \sigma)\mathbf{w}_{\|}'\right)\right).$$

Note that $\mathbf{w}_{\perp}^T\mathbf{w}_{\perp}'$ is explicitly computable, since $\mathbf{C}_k\mathbf{C}_k^T$ is available. With this we find that

$$\frac{d\phi(\sigma)}{d\sigma} = \frac{-1}{2\left(\sqrt{\psi}\right)^3}\frac{d\psi(\sigma)}{d\sigma} = -\frac{\mathbf{w}_{\|}^T\mathbf{S}_k^T\mathbf{S}_k\mathbf{w}_{\|}' + \mathbf{w}_{\perp}^T\mathbf{w}_{\perp}'}{\left(\sqrt{\mathbf{w}_{\|}^T\mathbf{S}_k^T\mathbf{S}_k\mathbf{w}_{\|} + \mathbf{w}_{\perp}^T\mathbf{w}_{\perp}}\right)^3},$$

is given explicitly. Using the expressions for $\phi(\sigma)$ and $\frac{d\phi(\sigma)}{d\sigma}$, we can apply Newton's method to solve for an optimal σ^* such that $\phi(\sigma^*) = 0$. Therefore our alternative trust-region solution strategy for the trust-region subproblem (1.4) with the ℓ_2-norm constraint is summarized in Algorithm 4.1

ALGORITHM 4.1

1. Compute $\lambda_{\min}$, the smallest eigenvalue;
2. If $0 \leq \lambda_{\min}$ set $\sigma^* = 0$, compute $\mathbf{s}^*$ from (4.18). If $\|\mathbf{s}^*\|_2 \leq \Delta_k$ terminate; otherwise go to 3;

3. Solve for $\sigma^* + \lambda_{\min} \geq 0$, $\phi(\sigma^*) = 0$ by Newton's method; compute $\mathbf{s}^*$ from (4.18), terminate;

In step 1 of Algorithm 4.1 the smallest eigenvalue $\lambda_{\min}$ is computed. This eigenvalue can be obtained by a technique similar to that described in Section 1.4.3. Based on the implicit QR factorization of $\mathbf{\Psi}_k$, we use the identity $\mathbf{\Psi}_k\mathbf{M}_k\mathbf{\Psi}_k^T = \mathbf{Q}\mathbf{R}\mathbf{M}_k\mathbf{R}^T\mathbf{Q}^T$. The matrix $\mathbf{R}\mathbf{M}_k\mathbf{R}^T$ is a small $2l \times 2l$ matrix, and its smallest eigenvalue $\hat{\lambda}_{\min}$ can be computed at complexity $\mathcal{O}(l^2)$ (computing l eigenvalues is of complexity $\mathcal{O}(l^3)$). Subsequently, the smallest eigenvalue of $\mathbf{B}_k$ is given by $\lambda_{\min} = \gamma_{k-1} + \hat{\lambda}_{\min}$.

4.4 NUMERICAL EXPERIMENTS

This section reports the results of implementations of Approach I and Approach II (cf. Section 4.3). Because Approach I is applicable with both, the ℓ_2-norm, and the shape-changing norms, we carried out experiments based on both of these norms. For Approach II we applied the ℓ_2-norm.

4.4.1 APPROACH I: MSSM SUBPROBLEMS WITH THE ℓ_2-NORM

Similar as in Section 2.4.4 we generated five sets of experiments composed of problems ranging from $n = 10^3$ to $n = 10^7$. The number of limited-memory updates l was set to 5, and thus $2l = 10$. The initial matrix is a multiple of identity matrix $\mathbf{B}_0 = \gamma_{k-1}\mathbf{I}_n$ with $\gamma_{k-1} = 0.5$. The pairs $\mathbf{S}_k$ and $\mathbf{Y}_k$, both $n \times l$ matrices, were generated from random data. Finally, $\mathbf{g}_k$ was generated by random data and the 'unconstrained' minimizer is computed from the compact representation in (4.10) by the Sherman-Morrison-Woodbury formula as $\mathbf{s}_u = -\left(\gamma_{k-1}\mathbf{I}_n + \mathbf{\Psi}_k\mathbf{M}_k\mathbf{\Psi}_k^T\right)^{-1}\mathbf{g}_k$.

1. The matrix $\mathbf{B}_k$ is positive definite with $\|\mathbf{s}_u\|_2 \leq \Delta_k$.

2. The matrix $\mathbf{B}_k$ is positive definite with $\|\mathbf{s}_u\|_2 > \Delta_k$.

3. The matrix $\mathbf{B}_k$ is positive semidefinite and singular with $\mathbf{s} = -\mathbf{B}_k^{\dagger}\mathbf{g}_k$ infeasible.

4. The matrix $\mathbf{B}_k$ is indefinite with $\|(\mathbf{B}_k - \lambda_1\mathbf{I}_n)\mathbf{g}_k)\|_2 > \Delta_k$ (a) the vector $\mathbf{g}_k$ is generated randomly, and (b) a random vector $\mathbf{g}_k$ is projected onto the orthogonal complement of the space spanned by the smallest eigenvalue of $\mathbf{B}_k$.

5. The hard case ($\mathbf{B}_k$ is indefinite): We ensure $\mathbf{\Psi}_k$ and $\mathbf{M}_k$ are such that $\mathbf{B}_k$ is indefinite. We test two subcases: (a) $\lambda_{\min} = \lambda_1 = \hat{\lambda}_1 + \gamma_{k-1} < 0$, and (b) $\lambda_{\min} = \gamma_{k-1} < 0$. In both cases, $\Delta_k = (1 + \mu)\|\mathbf{s}_u\|_2$, where μ is randomly generated between 0 and 1, so that $\|(\mathbf{B}_k - \lambda_1\mathbf{I}_n)\mathbf{g}_k)\|_2 > \Delta_k$.

We report the following: (1) `opt 1 (abs)`$=\|(\mathbf{B}_k+\sigma^*\mathbf{I}_n)\mathbf{s}^*+\mathbf{g}_k\|_2$, which corresponds to the norm of the error in the first optimality conditions; (2) `opt 1 (rel)` $=(\|(\mathbf{B}_k +$

$\sigma^*\mathbf{I}_n)\mathbf{s}^* + \mathbf{g}_k\|_2)/\|\mathbf{g}_k\|_2$, which corresponds to the norm of the *relative* error in the first optimality conditions; (3) `opt 2`$=\sigma^*|\mathbf{s}^* - \Delta_k|$, which corresponds to the absolute error in the second optimality conditions; (4) σ^*, and (5) Time. We ran each experiment five times and report one representative result for each experiment.

TABLE 4.1
Experiment 1: $\mathbf{B}_k$ *is positive definite and* $\|\mathbf{s}_u\|_2 \leq \Delta_k$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	3.65e-15	1.21e-16	0.00e+00	0.00e+00	7.44e-04
1.0e+04	1.03e-14	1.04e-16	0.00e+00	0.00e+00	2.36e-03
1.0e+05	3.64e-14	1.14e-16	0.00e+00	0.00e+00	2.70e-02
1.0e+06	1.00e-13	1.00e-16	0.00e+00	0.00e+00	3.94e-01
1.0e+07	4.13e-13	1.31e-16	0.00e+00	0.00e+00	3.85e+00

Table 4.1 shows the results of Experiment 1. In all cases, the method found global solutions of the trust-region subproblems.

TABLE 4.2
Experiment 2: $\mathbf{B}_k$ *is positive definite and* $\|\mathbf{s}_u\|_2 > \Delta_k$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	4.58e-15	1.43e-16	4.14e-15	4.66e+00	1.07e-03
1.0e+04	1.06e-14	1.05e-16	1.72e-14	1.93e+00	2.90e-03
1.0e+05	3.83e-14	1.21e-16	9.75e-13	8.16e-01	2.93e-02
1.0e+06	1.13e-13	1.13e-16	2.19e-13	3.08e+00	3.78e-01
1.0e+07	3.12e-13	9.87e-17	1.16e-11	6.94e+00	1.35e+01

Table 4.2 shows the results of Experiment 2. In this case, the unconstrained minimizer is not inside the trust region, making the value of σ^* strictly positive.

TABLE 4.3
Experiment 3(a): $\mathbf{B}_k$ *is positive semidefinite and singular with* $\|\mathbf{B}_k^\dagger \mathbf{g}_k\|_2 > \Delta_k$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	4.02e-15	1.34e-16	3.59e-15	2.70e+00	1.31e-03
1.0e+04	1.04e-14	1.05e-16	1.82e-13	5.85e+01	2.38e-03
1.0e+05	4.51e-14	1.42e-16	1.75e-13	1.76e+00	2.70e-02
1.0e+06	9.40e-14	9.39e-17	5.39e-12	5.67e+00	3.66e-01
1.0e+07	3.23e-13	1.02e-16	2.59e-10	8.70e+00	3.61e+00

Table 4.3 displays the results of Experiment 3(a). This experiment is the first of two in which $\mathbf{B}_k$ is highly ill-conditioned. The proposed method is able to obtain high accuracy solutions. Global solutions to the subproblems solved in Experiment 3(a) lie on the boundary of the trust region.

The results for Experiment 3(b) are shown in Table 4.4. This is the second experiment involving ill-conditioned matrices. As with Experiment 3(a), the method is able

TABLE 4.4
Experiment 3(b): $\mathbf{B}_k$ is positive semidefinite and singular with $\|\mathbf{B}_k^{\dagger}\mathbf{g}_k\|_2 \leq \Delta_k$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	4.41e-15	1.38e-16	0.00e+00	0.00e+00	9.72e-04
1.0e+04	1.51e-14	1.50e-16	0.00e+00	0.00e+00	3.15e-03
1.0e+05	3.55e-14	1.12e-16	0.00e+00	0.00e+00	3.13e-02
1.0e+06	1.68e-13	1.68e-16	0.00e+00	0.00e+00	4.10e-01
1.0e+07	3.98e-13	1.26e-16	0.00e+00	0.00e+00	4.29e+00

to obtain high-accuracy solutions. In this experiment, the global solution always lies on the boundary.

TABLE 4.5
Experiment 4(a): $\mathbf{B}_k$ is indefinite with $\|(\mathbf{B}_k - \lambda_1\mathbf{I}_n)\mathbf{g}_k)\|_2 > \Delta_k$. The vector $\mathbf{g}_k$ is randomly generated.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	3.14e-15	1.01e-16	0.00e+00	2.44e+01	8.82e-04
1.0e+04	9.08e-15	9.17e-17	9.89e-15	8.91e+01	3.03e-03
1.0e+05	4.02e-14	1.27e-16	6.83e-14	3.07e+02	3.02e-02
1.0e+06	1.03e-13	1.03e-16	2.73e-11	9.95e+02	3.78e-01
1.0e+07	3.00e-13	9.48e-17	1.10e-10	3.16e+03	3.61e+00

The results for Experiment 4(a) are displayed in Table 4.5. The method found solutions that satisfies the first optimality condition high accuracy.

TABLE 4.6
Experiment 4(b): $\mathbf{B}_k$ is indefinite with $\|(\mathbf{B}_k - \lambda_1\mathbf{I}_n)\mathbf{g}_k)\|_2 > \Delta_k$. The vector $\mathbf{g}_k$ lies in the orthogonal complement of the smallest eigenvector.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	2.97e-15	9.35e-17	1.01e-14	6.07e+01	8.05e-04
1.0e+04	1.19e-14	1.20e-16	2.66e-15	4.98e-01	2.30e-03
1.0e+05	3.85e-14	1.22e-16	1.46e-12	1.84e+00	2.94e-02
1.0e+06	9.63e-14	9.62e-17	2.81e-11	3.91e+01	4.01e-01
1.0e+07	2.94e-13	9.28e-17	3.64e-11	2.35e+01	3.62e+00

The results of Experiment 4(b) are in Table 4.6 . The method solved the subproblem to high accuracy as measured by the optimality conditions.

In the hard case with $\lambda_{\min}$ being a nontrivial eigenvalue, the method obtains a global solution as measured by the optimality conditions.

The results of Experiment 5(b) are in Table 4.8. The method was able to find solutions in the two instances of the hard-case.

TABLE 4.7
Experiment 5(a): The hard case ($\mathbf{B}_k$ is indefinite) and $\lambda_{\min} = \lambda_1 = \hat{\lambda}_1 + \gamma_{k-1} < 0$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	4.88e-15	1.54e-16	1.58e-15	8.88e-01	7.67e-04
1.0e+04	2.48e-13	2.51e-15	2.20e-15	6.19e-01	2.61e-03
1.0e+05	1.01e-12	3.18e-15	1.40e-13	5.17e-01	3.42e-02
1.0e+06	1.58e-11	1.58e-14	2.28e-12	4.72e-01	4.19e-01
1.0e+07	1.09e-10	3.45e-14	1.57e-10	5.89e-01	4.38e+00

TABLE 4.8
Experiment 5(b): The hard case ($\mathbf{B}_k$ is indefinite) and $\lambda_{\min} = \gamma_{k-1} < 0$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	5.68e-14	2.93e-14	1.24e-13	8.71e+00	8.84e-04
1.0e+04	4.40e-14	1.61e-14	4.04e-14	5.69e+00	2.79e-03
1.0e+05	1.04e-13	3.36e-14	1.37e-12	1.21e+01	3.35e-02
1.0e+06	1.32e-13	5.34e-14	1.00e-13	3.13e+00	4.40e-01
1.0e+07	1.70e-13	7.89e-14	9.61e-12	1.89e+00	4.41e+00

4.4.2 APPROACH I: MSSM SUBPROBLEMS WITH THE $(\mathbf{P}, 2)$-NORM

In this section, we report numerical experiments with shape-changing $(\mathbf{P}, 2)$-norm (cf. eq. (2.18)) implemented in MATLAB. The algorithm was used on randomly-generated problems of size $n = 10^3$ to $n = 10^7$. We report five experiments when there is no closed-form solution to the shape-changing trust-region subproblem and one experiment designed to test the method in the so-called "hard case". These six cases only occur using the shape-changing $(\mathbf{P}, 2)$-norm, since the alternative $(\mathbf{P}, \infty)$-norm yields analytic solutions. To compute subproblem solutions with MSS matrices we apply the procedures as proposed in 2.5.2. The difference is that instead of the L-SR1 matrix $\mathbf{B}_k$ from (2.3) we apply the compact representation of the L-MSSM matrix $\mathbf{B}_k$ as defined by (4.10). The six experiments are outlined as follows:

(E1) $\mathbf{B}_k$ is positive definite with $\|\mathbf{v}_{\|}(0)\|_2 \geq \Delta_k$.

(E2) $\mathbf{B}_k$ is positive semidefinite and singular with $[\mathbf{g}_{\|}]_i \neq 0$ for some $1 \leq i \leq r$.

(E3) $\mathbf{B}_k$ is positive semidefinite and singular with $[\mathbf{g}_{\|}]_i = 0$ for $1 \leq i \leq r$ and $\|\Lambda^{\dagger}\mathbf{g}_{\|}\|_2 > \Delta_k$.

(E4) $\mathbf{B}_k$ is indefinite and $[\mathbf{g}_{\|}]_i = 0$ for $1 \leq i \leq r$ with $\|(\Lambda - \lambda_1\mathbf{I})^{\dagger}\mathbf{g}_{\|}\|_2 > \Delta_k$.

(E5) $\mathbf{B}_k$ is indefinite and $[\mathbf{g}_{\|}]_i \neq 0$ for some $1 \leq i \leq r$.

(E6) $\mathbf{B}_k$ is indefinite and $[\mathbf{g}_{\|}]_i = 0$ for $1 \leq i \leq r$ with $\|\mathbf{v}_{\|}(-\lambda_1)\|_2 \leq \Delta_k$ (the "hard case").

For these experiments, $\mathbf{S}_k$, $\mathbf{Y}_k$, and $\mathbf{g}_k$ 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 MSS matrix was set to $\mathbf{B}_0 = \gamma_{k-1}\mathbf{I}$, where $\gamma_{k-1} = |10 * \texttt{randn(1)}|$. Finally, the number of limited-memory updates l was set to 5 ($2l = 10$), and r was set to 2. In the five cases when there is no closed-form solution, the method uses Newton's 1D zero solver to find a root of $\phi_{\|}$. We use the same procedure as in [BEM17, 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.

In order to report on the accuracy of the subproblem solves, we make use of the optimality conditions found in Theorem 2.5. For each experiment, we report the following: (i) the norm of the residual of the first optimality condition, $\texttt{opt 1} \triangleq \|(\mathbf{B}_k + \mathbf{C}_{\|})\mathbf{s}^* + \mathbf{g}_k\|_2$, (ii) the combined complementarity condition, $\texttt{opt 2} \triangleq |\sigma_{\|}^*(\|\mathbf{P}_{\|}^T\mathbf{s}^*\|_2 - \Delta_k)| + |\sigma_{\perp}^*(\|\mathbf{P}_{\perp}^T\mathbf{s}^*\|_2 - \Delta_k)|$, (iii) $\sigma_{\|}^* + \lambda_1$, (iv) $\sigma_{\perp}^* + \gamma_{k-1}$, (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 $\mathbf{B}_k + \mathbf{C}_{\|}$ is a positive semidefinite matrix is equivalent to $\gamma_{k-1} + \sigma_{\perp}^* \geq 0$ and $\lambda_i + \sigma_{\|}^* \geq 0$, for $1 \leq i \leq 2l$. Finally, we ran each experiment five times and report one representative result for each experiment.

n	opt 1	opt 2	$\sigma_{\|}^* + \lambda_1$	$\sigma_{\perp}^* + \gamma_{k-1}$	$\sigma_{\|}^*$	$\sigma_{\perp}^*$	itns	time
1.0e+03	2.49e-14	1.66e-14	2.98e+01	4.10e+02	1.10e+01	3.91e+02	3	8.71e-04
1.0e+04	8.80e-14	1.44e-13	1.84e+01	7.52e+02	2.88e+00	7.37e+02	3	3.88e-03
1.0e+05	8.98e-13	4.25e-13	1.19e+01	9.47e+02	8.29e+00	9.44e+02	3	2.72e-02
1.0e+06	1.01e-11	0.00e+00	2.63e+01	1.21e+04	1.32e+01	1.21e+04	4	2.87e-01
1.0e+07	1.14e-10	6.40e-11	4.00e+00	5.29e+03	3.34e-01	5.29e+03	3	2.83e+00

TABLE 4.9
Experiment 1: $\mathbf{B}_k$ *is positive definite with* $\|\mathbf{v}_{\|}(0)\|_2 \geq \Delta_k$.

n	opt 1	opt 2	$\sigma_{\|}^* + \lambda_1$	$\sigma_{\perp}^* + \gamma_{k-1}$	$\sigma_{\|}^*$	$\sigma_{\perp}^*$	itns	time
1.0e+03	1.41e-13	1.67e-13	5.52e-01	8.94e+00	5.52e-01	0.00e+00	2	6.06e-04
1.0e+04	2.38e-13	2.72e-15	1.18e-01	8.60e+00	1.18e-01	0.00e+00	2	1.95e-03
1.0e+05	3.32e-12	1.40e-15	2.82e-02	9.05e+00	2.82e-02	0.00e+00	2	2.77e-02
1.0e+06	2.30e-12	2.64e-15	1.55e-02	9.20e+00	1.55e-02	0.00e+00	2	2.87e-01
1.0e+07	1.59e-10	1.48e-10	2.64e-04	5.46e-01	2.64e-04	0.00e+00	1	2.93e+00

TABLE 4.10
Experiment 2: $\mathbf{B}_k$ *is positive semidefinite and singular and* $[\mathbf{g}_{\|}]_i \neq 0$ *for some* $1 \leq i \leq r$.

Tables 4.9–4.14 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 1×10^{-10} or smaller. Columns 4 and 5 in all the tables show that $\sigma_{\|}^* + \lambda_1$ and $\sigma_{\perp}^* + \gamma_{k-1}$ are nonnegative

n	opt 1	opt 2	$\sigma_{\|}^* + \lambda_1$	$\sigma_{\perp}^* + \gamma_{k-1}$	$\sigma_{\|}^*$	$\sigma_{\perp}^*$	itns	time
1.0e+03	3.09e-14	7.46e-13	1.87e+01	7.44e+02	1.87e+01	7.29e+02	3	2.35e-03
1.0e+04	1.43e-13	4.38e-14	7.71e+01	2.01e+03	7.71e+01	2.00e+03	3	2.01e-03
1.0e+05	2.41e-12	1.93e-13	2.48e-01	5.81e+02	2.48e-01	5.78e+02	3	2.75e-02
1.0e+06	1.05e-11	1.99e-12	1.84e+01	7.15e+03	1.84e+01	7.15e+03	3	2.89e-01
1.0e+07	8.87e-11	4.97e-11	5.23e+00	2.61e+04	5.23e+00	2.61e+04	3	2.92e+00

TABLE 4.11

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

n	opt 1	opt 2	$\sigma_{\|}^* + \lambda_1$	$\sigma_{\perp}^* + \gamma_{k-1}$	$\sigma_{\|}^*$	$\sigma_{\perp}^*$	itns	time
1.0e+03	2.45e-14	5.02e-15	2.17e+00	8.33e+01	2.43e+00	8.31e+01	3	6.71e-04
1.0e+04	1.37e-13	6.03e-14	6.34e+00	7.26e+02	6.37e+00	7.14e+02	3	2.25e-03
1.0e+05	7.85e-13	3.47e-12	1.19e+00	1.02e+03	1.26e+00	1.01e+03	2	2.85e-02
1.0e+06	1.45e-11	9.39e-12	7.87e+00	5.13e+03	8.35e+00	5.12e+03	3	2.85e-01
1.0e+07	9.33e-11	4.05e-11	9.66e+00	1.59e+04	1.04e+01	1.59e+04	3	2.90e+00

TABLE 4.12

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

n	opt 1	opt 2	$\sigma_{\|}^* + \lambda_1$	$\sigma_{\perp}^* + \gamma_{k-1}$	$\sigma_{\|}^*$	$\sigma_{\perp}^*$	itns	time
1.0e+03	1.18e-14	2.41e-12	3.08e-01	3.12e+01	9.61e-01	2.38e+01	4	7.67e-04
1.0e+04	1.42e-13	3.40e-14	1.28e+00	9.92e+01	1.60e+00	9.79e+01	4	2.00e-03
1.0e+05	3.80e-13	2.18e-13	2.37e+00	3.17e+02	3.13e+00	2.96e+02	2	2.50e-02
1.0e+06	1.01e-11	1.30e-12	1.92e+00	1.00e+03	2.69e+00	9.99e+02	4	3.00e-01
1.0e+07	7.39e-11	2.45e-11	4.91e-01	3.16e+03	1.12e+00	3.14e+03	2	2.84e+00

TABLE 4.13

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

n	opt 1	opt 2	$\sigma_{\|}^* + \lambda_1$	$\sigma_{\perp}^* + \gamma_{k-1}$	$\sigma_{\|}^*$	$\sigma_{\perp}^*$	itns	time
1.0e+03	2.53e-14	2.58e-17	0.00e+00	1.01e+02	4.66e-01	8.91e+01	0	6.09e-04
1.0e+04	1.06e-13	1.03e-13	0.00e+00	1.60e+02	9.46e-03	1.54e+02	0	2.20e-03
1.0e+05	1.57e-12	9.26e-13	0.00e+00	3.83e+02	4.09e-01	3.78e+02	0	2.94e-02
1.0e+06	1.24e-11	9.47e-12	0.00e+00	6.22e+03	8.39e-01	6.20e+03	0	2.94e-01
1.0e+07	8.39e-11	1.62e-11	0.00e+00	8.10e+03	6.63e-01	8.09e+03	0	3.10e+00

TABLE 4.14

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

with $\sigma_{\|} \geq 0$ and $\sigma_{\perp} \geq 0$ (Columns 6 and 7, respectively). Thus, the solutions obtained with the L-MSS matrices 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 $\sigma_{\|}^* = -\lambda_1$. This is reflected in Table 4.14, where Column 4 shows that $\sigma_{\|}^* = -\lambda_1$ and Column 8 reports no Newton iterations.

The final column reports the time required by the method to solve each subproblem. Consistent with the best limited-memory methods, the time required to solve each subproblem appears to grow linearly with n.

4.4.3 APPROACH II: MSSM SUBPROBLEMS WITH THE ℓ_2-NORM

Similar as in Section 4.4.3 we generated five sets of experiments composed of problems ranging from $n = 10^3$ to $n = 10^7$. The number of limited-memory updates l was set to 5, and thus $2l = 10$. The initial matrix is a multiple of identity matrix $\mathbf{B}_0 = \gamma_{k-1}\mathbf{I}_n$ with $\gamma_{k-1} = 0.5$. The pairs $\mathbf{S}_k$ and $\mathbf{Y}_k$, both $n \times l$ matrices, were generated from random data. Finally, $\mathbf{g}_k$ was generated by random data and the 'unconstrained' minimizer is computed from the compact representation in (4.10) by the Sherman-Morrison-Woodbury formula as $\mathbf{s}_u = -\left(\gamma_{k-1}\mathbf{I}_n + \mathbf{\Psi}_k\mathbf{M}_k\mathbf{\Psi}_k^T\right)^{-1}\mathbf{g}_k$.

1. The matrix $\mathbf{B}_k$ is positive definite with $\|\mathbf{s}_u\|_2 \leq \Delta_k$.
2. The matrix $\mathbf{B}_k$ is positive definite with $\|\mathbf{s}_u\|_2 > \Delta_k$.
3. The matrix $\mathbf{B}_k$ is positive semidefinite and singular with $\mathbf{s} = -\mathbf{B}_k^{\dagger}\mathbf{g}_k$ infeasible.
4. The matrix $\mathbf{B}_k$ is indefinite.
5. The hard case ($\mathbf{B}_k$ is indefinite): $\lambda_{\min} = \lambda_1 = \hat{\lambda}_1 + \gamma_{k-1} < 0$ and $\|(\mathbf{B}_k - \lambda_1\mathbf{I}_n)\mathbf{g}_k)\|_2 > \Delta_k$.

We report the following: (1) `opt 1 (abs)`$=\|(\mathbf{B}_k+\sigma^*\mathbf{I}_n)\mathbf{s}^*+\mathbf{g}_k\|_2$, which corresponds to the norm of the error in the first optimality conditions; (2) `opt 1 (rel)` $=(\|(\mathbf{B}_k + \sigma^*\mathbf{I}_n)\mathbf{s}^* + \mathbf{g}_k\|_2)/\|\mathbf{g}_k\|_2$, which corresponds to the norm of the *relative* error in the first optimality conditions; (3) `opt 2`$=\sigma^*|\mathbf{s}^* - \Delta_k|$, which corresponds to the absolute error in the second optimality conditions; (4) σ^*, and (5) Time. We ran each experiment five times and report one representative result for each experiment.

TABLE 4.15
Experiment 1: $\mathbf{B}_k$ *is positive definite and* $\|\mathbf{s}_u\|_2 \leq \Delta_k$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	4.24e-15	1.34e-16	0.00e+00	0.00e+00	1.97e-03
1.0e+04	1.35e-14	1.34e-16	0.00e+00	0.00e+00	3.30e-03
1.0e+05	3.63e-14	1.15e-16	0.00e+00	0.00e+00	2.11e-02
1.0e+06	1.28e-13	1.28e-16	0.00e+00	0.00e+00	2.19e-01
1.0e+07	3.57e-13	1.13e-16	0.00e+00	0.00e+00	2.21e+00

Table 4.15 shows the results of Experiment 1. In all cases, the method found global solutions of the trust-region subproblems.

Table 4.16 shows the results of Experiment 2. In this case, the unconstrained minimizer is not inside the trust region, making the value of σ^* strictly positive.

TABLE 4.16
Experiment 2: $\mathbf{B}_k$ *is positive definite and* $\|\mathbf{s}_u\|_2 > \Delta_k$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	4.02e-15	1.30e-16	4.98e-15	2.99e+01	2.31e-03
1.0e+04	1.37e-14	1.35e-16	1.20e-10	1.60e+01	3.61e-03
1.0e+05	3.52e-14	1.11e-16	1.35e-12	1.65e+02	2.16e-02
1.0e+06	1.44e-13	1.44e-16	2.71e-12	1.73e+01	2.23e-01
1.0e+07	4.15e-13	1.31e-16	2.67e-11	3.69e+01	2.21e+00

TABLE 4.17
Experiment 3: $\mathbf{B}_k$ *is positive semidefinite and singular with* $\|\mathbf{B}_k^{\dagger}\mathbf{g}_k\|_2 > \Delta_k$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	9.88e-14	3.13e-15	6.28e-16	2.21e-02	2.26e-03
1.0e+04	3.44e-13	3.39e-15	2.45e-15	3.83e-02	3.89e-03
1.0e+05	4.10e-12	1.30e-14	5.65e-13	7.21e-03	2.13e-02
1.0e+06	1.63e-13	1.63e-16	1.91e-11	3.70e-01	2.20e-01
1.0e+07	1.14e-10	3.60e-14	6.42e-14	5.24e-04	2.20e+00

Table 4.17 displays the results of Experiment 3. In this experiment $\mathbf{B}_k$ is singular, and therefore not well conditioned. The proposed method is able to obtain high accuracy solutions. Global solutions to the subproblems solved in Experiment 3 lie on the boundary of the trust region.

TABLE 4.18
Experiment 4: $\mathbf{B}_k$ *is indefinite.*

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	1.66e-14	5.29e-16	3.72e-15	4.19e-01	2.32e-03
1.0e+04	5.33e-13	5.35e-15	8.36e-12	9.49e-01	3.76e-03
1.0e+05	4.26e-14	1.35e-16	1.77e-09	4.59e+00	2.19e-02
1.0e+06	6.64e-13	6.64e-16	1.43e-12	9.26e-01	2.25e-01
1.0e+07	4.09e-13	1.29e-16	4.52e-10	2.70e+00	2.21e+00

The results for Experiment 4 are displayed in Table 4.18. The method found solutions that satisfies the first optimality condition high accuracy.

TABLE 4.19
Experiment 5: The hard case ($\mathbf{B}_k$ *is indefinite) and* $\lambda_{\min} = \lambda_1 = \hat{\lambda}_1 + \gamma_{k-1} < 0$.

n	opt 1 (abs)	opt 1 (rel)	opt 2	σ^*	Time
1.0e+03	9.80e-14	3.13e-15	1.16e-14	8.18e-01	2.53e-03
1.0e+04	1.66e-12	1.66e-14	1.31e-14	1.54e-01	5.25e-03
1.0e+05	1.74e-11	5.50e-14	1.66e-13	1.01e-01	4.05e-02
1.0e+06	2.76e-10	2.76e-13	2.09e-12	1.39e-01	4.64e-01
1.0e+07	1.82e-10	5.76e-14	2.34e-10	5.00e-01	4.59e+00

In the hard case with $\lambda_{\min}$ being a nontrivial eigenvalue, the method obtains a global solution as measured by the optimality conditions.

CHAPTER 5

LINEAR EQUALITY CONSTRAINED TRUST-REGION METHODS

This chapter is based on the manuscript "*Large-Scale Quasi-Newton Shape-Changing Trust-Region Method with Low Dimensional Linear Equality Constraints*" J. B. Brust, R. F. Marcia, and C. G. Petra, which is currently in preparation.

5.1 BACKGROUND

5.1.1 PROBLEM FORMULATION

In this chapter we focus on the linear equality constraint minimization problem

$$\underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimize}}\ f(\mathbf{x}) \quad \text{subject to} \quad \mathbf{A}\mathbf{x} = \mathbf{b}, \tag{5.1}$$

where the objective function $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and the constraints are defined by $\mathbf{A} \in \mathbb{R}^{m\times n}$ and $\mathbf{b} \in \mathbb{R}^m$. We assume that $m \ll n$ and that the rectangular matrix $\mathbf{A}$ is of full row rank. Because it has been shown that trust-region methods are effective for solving constrained optimization problems [CDJT84, Var85, LNP98], we propose solving the problem in (5.1) using trust-region methods. In particular, we concentrate on trust-region methods that use limited-memory quasi-Newton matrices, and exploit the structure of these matrices to compute the trust-region subproblem solutions.

5.1.2 CONSTRAINED TRUST-REGION METHOD

The constrained trust-region method is obtained as an extension of the unconstrained method described in Chapter 1, Section 1.3. Recall the quadratic subproblems of the form

$$\mathbf{s}_k = \underset{\|\mathbf{s}\| \leq \Delta_k}{\arg\min}\, Q(\mathbf{s}) = \mathbf{s}^T\mathbf{g}_k + \frac{1}{2}\mathbf{s}^T\mathbf{B}_k\mathbf{s} \tag{5.2}$$

where the solution $\mathbf{s}_k$ is used to compute the next iterate $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{s}_k$.

In order to use trust-region methods to solve (5.1), we note that if the current iterate $\mathbf{x}_k$ is feasible, i.e., $\mathbf{A}\mathbf{x}_k = \mathbf{b}$, then the next iterate $\mathbf{x}_{k+1}$ is also feasible, i.e., $\mathbf{A}\mathbf{x}_{k+1} = \mathbf{b}$, only if $\mathbf{A}\mathbf{s}_k = \mathbf{0}$. Thus, the trust-region subproblem corresponding to (5.1) is given by

$$\underset{\|\mathbf{s}\| \leq \Delta_k}{\text{minimize}}\, Q(\mathbf{s}) = \mathbf{s}^T\mathbf{g}_k + \frac{1}{2}\mathbf{s}^T\mathbf{B}_k\mathbf{s} \quad \text{subject to} \quad \mathbf{A}\mathbf{s} = \mathbf{0}. \tag{5.3}$$

Without the equality constraint in (5.3), the trust-region subproblem can be solved to high accuracy with the methods from Chapter 2 or using e.g. [BWX96, EM14, BGYZ16]. However, these methods cannot be used directly to solve (5.3) with the equality constraint.

For this chapter, the matrix $\mathbf{B}_k$ represents the limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) matrix. Furthermore, we propose solving (5.3) with two norms. Specifically, we use the shape-changing $(\mathbf{P}, \infty)$-norm in (2.19) from Subsection 2.5.4, as well as the ℓ_2-norm. Unlike in the previous chapters, we compute the compact representation of part of the Hessian of the Lagrangian function (without the norm constraint) rather than the Hessian of the objective function $Q(\mathbf{s})$. This compact representation allows $Q(\mathbf{s})$ to be decomposed into two subproblems. And by using a shape-changing norm, closed-form solutions can then be computed. We use the compact representation of the inverse L-BFGS matrix $\mathbf{H}_k = \mathbf{B}_k^{-1}$ in this chapter, because it simplifies the development of our methods. In particular, let $\delta_k = \frac{\mathbf{y}_{k-1}^T\mathbf{s}_{k-1}}{\mathbf{y}_{k-1}^T\mathbf{y}_{k-1}}$, and let the matrices $\mathbf{L}_k, \mathbf{D}_k$ and $\mathbf{T}_k$ be as defined in Section 3.2, then the compact representation of $\mathbf{H}_k$ is

$$\mathbf{H}_k = \delta_k\mathbf{I} + \hat{\mathbf{\Psi}}_k\hat{\mathbf{M}}_k\hat{\mathbf{\Psi}}_k^T, \tag{5.4}$$

where

$$\hat{\mathbf{M}}_k \equiv \begin{bmatrix} \mathbf{T}_k^{-T}(\mathbf{D}_k + \delta_k\mathbf{Y}_k^T\mathbf{Y}_k)\mathbf{T}_k^{-1} & -\delta_k\mathbf{T}_k^{-T} \\ -\delta_k\mathbf{T}_k^{-1} & \mathbf{0} \end{bmatrix},$$

$\hat{\mathbf{\Psi}}_k = [\ \mathbf{S}_k\ \ \mathbf{Y}_k\ ] \in \mathbb{R}^{n\times 2l}$ and with $\hat{\mathbf{M}}_k \in \mathbb{R}^{2l\times 2l}$ (see [BNS94] for details). Note that the methods that will be described in this chapter are applicable to any quasi-Newton matrix with a compact representation, not only the L-BFGS matrix.

5.2 LARGE-SCALE QUASI-NEWTON METHODS

5.2.1 THE KARUSH-KUHN-TUCKER (KKT) MATRIX

When the norm constraint $\|\mathbf{s}\| \leq \Delta_k$ in (5.3) is not present, then the solution of this 'unconstrained' problem can be characterized as a stationary point of the Lagrangian objective function

$$\mathcal{L}(\mathbf{s}, \rho) = Q(\mathbf{s}) + \rho^T \mathbf{A}\mathbf{s},$$

where $\rho \in \mathbb{R}^m$ is a vector of Lagrange multipliers. The stationary point $(\mathbf{s}_u, \rho_u)$ is obtained by setting the gradient of the Lagrangian equal to zero, i.e., $\nabla\mathcal{L}(\mathbf{s}_u, \rho_u) = \mathbf{0}$. This gives rise to the Karush-Kuhn-Tucker (KKT) system

$$\begin{bmatrix} \mathbf{B}_k & \mathbf{A}^T \\ \mathbf{A} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{s}_u \\ \rho_u \end{bmatrix} = \begin{bmatrix} -\mathbf{g}_k \\ \mathbf{0} \end{bmatrix}. \tag{5.5}$$

Let $\mathbf{K}$ denote the KKT matrix in (5.5). Note that $\mathbf{K}$ can be explicitly inverted as

$$\mathbf{K}^{-1} = \begin{bmatrix} \mathbf{H}_k - \mathbf{H}_k\mathbf{A}^T(\mathbf{A}\mathbf{H}_k\mathbf{A})^{-1}\mathbf{A}\mathbf{H}_k & \mathbf{H}_k\mathbf{A}^T(\mathbf{A}\mathbf{H}_k\mathbf{A})^{-1} \\ (\mathbf{H}_k\mathbf{A}^T(\mathbf{A}\mathbf{H}_k\mathbf{A})^{-1})^T & (\mathbf{A}\mathbf{H}_k\mathbf{A})^{-1} \end{bmatrix} \equiv \begin{bmatrix} \mathbf{K}_{11}^{-1} & \mathbf{K}_{12}^{-1} \\ \mathbf{K}_{21}^{-1} & \mathbf{K}_{22}^{-1} \end{bmatrix}.$$

The block-wise notation reveals that the stationary vector $\mathbf{s}_u = -\mathbf{K}_{11}^{-1}\mathbf{g}_k$ depends only on the upper left hand block, i.e. block $\mathbf{K}_{11}^{-1}$. In the next section we will compute the compact representation of

$$\mathbf{K}_{11}^{-1} = \mathbf{H}_k - \mathbf{H}_k\mathbf{A}^T\left(\mathbf{A}\mathbf{H}_k\mathbf{A}\right)^{-1}\mathbf{A}\mathbf{H}_k \equiv \mathbf{H}_k\mathbf{W}_k, \tag{5.6}$$

where $\mathbf{W}_k = \mathbf{I} - \mathbf{A}^T\left(\mathbf{A}\mathbf{H}_k\mathbf{A}\right)^{-1}\mathbf{A}\mathbf{H}_k$. The compact representation in (5.6) facilitates computing the solutions to the trust-region subproblems in (5.3). Furthermore, the compact representation enables us to develop the partial eigendecomposition of (5.6), which we will combine with a shape-changing norm in order to compute an analytic solution to (5.3).

5.2.2 COMPACT REPRESENTATION OF $\mathbf{K}_{11}^{-1}$

We now describe the compact representation of (5.6) with a lemma:

Lemma 5.1. *The (1,1) block of the inverse KKT matrix in* (5.6) *has the compact representation*

$$\mathbf{K}_{11}^{-1} = \delta_k\mathbf{I} + \mathbf{\Psi}_k\mathbf{M}_k\mathbf{\Psi}_k^T, \tag{5.7}$$

where

$$\mathbf{\Psi}_k \equiv [\mathbf{A}^T \ \hat{\mathbf{\Psi}}_k], \quad \mathbf{M}_k \equiv -\begin{bmatrix} \mathbf{M}_{11} & \mathbf{M}_{12} \\ \mathbf{M}_{12}^T & \mathbf{M}_{22} \end{bmatrix}, \quad \mathbf{K}_k = \mathbf{A}\hat{\mathbf{\Psi}}_k\hat{\mathbf{M}}_k,$$

and where

$$\mathbf{M}_{11} = \delta_k^2(\mathbf{A}\mathbf{H}_k\mathbf{A}^T)^{-1}, \quad \mathbf{M}_{12} = \delta_k^{-1}\mathbf{M}_{11}\mathbf{K}_k, \quad \mathbf{M}_{22} = \delta_k^{-2}\mathbf{K}_k^T\mathbf{M}_{11}\mathbf{K}_k - \hat{\mathbf{M}}_k.$$

Proof. Define $\mathbf{K}_k \equiv \mathbf{A}\hat{\mathbf{\Psi}}_k\hat{\mathbf{M}}_k$, and observe that

$$\mathbf{A}\mathbf{H}_k = \delta_k\mathbf{A} + \mathbf{A}\hat{\mathbf{\Psi}}_k\hat{\mathbf{M}}_k\hat{\mathbf{\Psi}}_k^T \equiv \delta_k\mathbf{A} + \mathbf{K}_k\hat{\mathbf{\Psi}}_k^T = [\delta_k\mathbf{I}_m \quad \mathbf{K}_k] \begin{bmatrix} \mathbf{A} \\ \hat{\mathbf{\Psi}}_k^T \end{bmatrix}.$$

Now let $\mathbf{M}_{11} \equiv \delta_k^2(\mathbf{A}\mathbf{H}_k\mathbf{A}^T)^{-1}$, so that

$$\mathbf{H}_k\mathbf{A}^T(\mathbf{A}\mathbf{H}_k\mathbf{A}^T)^{-1}\mathbf{A}\mathbf{H}_k = [\mathbf{A}^T \quad \hat{\mathbf{\Psi}}_k] \begin{bmatrix} \mathbf{M}_{11} & \delta_k^{-1}\mathbf{M}_{11}\mathbf{K}_k \\ \delta_k^{-1}\mathbf{K}_k^T\mathbf{M}_{11} & \delta_k^{-2}\mathbf{K}_k^T\mathbf{M}_{11}\mathbf{K}_k \end{bmatrix} \begin{bmatrix} \mathbf{A} \\ \hat{\mathbf{\Psi}}_k^T \end{bmatrix}.$$

Next, expressing $\mathbf{H}_k$, from (5.4), as

$$\mathbf{H}_k = \delta_k\mathbf{I} + \hat{\mathbf{\Psi}}_k\hat{\mathbf{M}}_k\hat{\mathbf{\Psi}}_k^T = \delta_k\mathbf{I} + [\mathbf{A}^T \quad \hat{\mathbf{\Psi}}_k] \begin{bmatrix} \mathbf{0}_m & \\ & \hat{\mathbf{M}}_k \end{bmatrix} \begin{bmatrix} \mathbf{A}_n \\ \hat{\mathbf{\Psi}}_k^T \end{bmatrix},$$

and combining the previous identities, yields

$$\begin{aligned} \mathbf{K}_{11}^{-1} &= \mathbf{H}_k - \mathbf{H}_k\mathbf{A}^T\left(\mathbf{A}\mathbf{H}_k\mathbf{A}\right)^{-1}\mathbf{A}\mathbf{H}_k \\ &= \delta_k\mathbf{I} - [\mathbf{A}^T \quad \hat{\mathbf{\Psi}}_k] \begin{bmatrix} \mathbf{M}_{11} & \mathbf{M}_{12} \\ \mathbf{M}_{12}^T & \mathbf{M}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{A}_n \\ \hat{\mathbf{\Psi}}_k^T \end{bmatrix} \\ &\equiv \delta_k\mathbf{I} + \mathbf{\Psi}_k\mathbf{M}_k\mathbf{\Psi}_k^T, \end{aligned}$$

where $\mathbf{M}_{12} \equiv \delta_k^{-1}\mathbf{M}_{11}\mathbf{K}_k$, $\mathbf{M}_{22} \equiv \delta_k^{-2}\mathbf{K}_k^T\mathbf{M}_{11}\mathbf{K}_k - \hat{\mathbf{M}}_k$, and where $\mathbf{\Psi}_k$ and $\mathbf{M}_k$ are defined so that the last equivalence holds. ■

5.3 TRUST-REGION APPROACH WITH AN ℓ_2 CONSTRAINT

If the ℓ_2-norm is used in (5.3), that is, $\|\mathbf{s}\|_2 \leq \Delta_k$ is used, then the solution, defined by the optimal Lagrange multiplier $\sigma^* \geq 0$ and a vector of Lagrange multipliers $\hat{\rho}^*$, is the vector $\hat{\mathbf{s}}^*$ that satisfies the system

$$\begin{bmatrix} (\mathbf{B}_k + \sigma^*\mathbf{I}) & \mathbf{A}^T \\ \mathbf{A} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{s}}^* \\ \hat{\rho}^* \end{bmatrix} = \begin{bmatrix} -\mathbf{g}_k \\ \mathbf{0} \end{bmatrix}. \tag{5.8}$$

In solving this latter system, we express the optimal vectors $\hat{\mathbf{s}}^* \equiv \hat{\mathbf{s}}^*(\sigma^*)$ as:

$$\begin{aligned}\hat{\mathbf{s}}^*(\sigma^*) &= -(\mathbf{B}_k + \sigma^*\mathbf{I})^{-1}\left(\mathbf{I} - \mathbf{A}^T\left(\mathbf{A}(\mathbf{B}_k + \sigma^*\mathbf{I})^{-1}\mathbf{A}^T\right)^{-1}\mathbf{A}(\mathbf{B}_k + \sigma^*\mathbf{I})^{-1}\right)\mathbf{g}_k \\ &= -\hat{\mathbf{H}}_k^*\hat{\mathbf{W}}_k^*\mathbf{g}_k, \end{aligned} \tag{5.9}$$

where $\hat{\mathbf{H}}_k^* = (\mathbf{B}_k + \sigma^*\mathbf{I})^{-1}$, and $\hat{\mathbf{W}}_k^* = \mathbf{I} - \mathbf{A}^T\left(\mathbf{A}(\mathbf{B}_k + \sigma^*\mathbf{I})^{-1}\mathbf{A}^T\right)^{-1}\mathbf{A}(\mathbf{B}_k + \sigma^*\mathbf{I})^{-1}$. When an arbitrary σ is used, instead of the optimal σ^*, we use the expression $\hat{\mathbf{s}}(\sigma) = \hat{\mathbf{H}}_k\hat{\mathbf{W}}_k\mathbf{g}_k$ to represent (5.9). Now, if $\|\hat{\mathbf{s}}(0)\|_2 \leq \Delta_k$, then the expression in (5.9) with $\sigma^* = 0$, yields the subproblem solution. Otherwise we use Newton's method to solve for $\sigma^* > 0$ in the so-called *secular-equation* [CGT00]

$$\phi(\sigma^*) = \frac{1}{\|\hat{\mathbf{s}}^*(\sigma^*)\|_2} - \frac{1}{\Delta_k} = 0. \tag{5.10}$$

The solution of the latter scalar equation corresponds to $\|\hat{\mathbf{s}}^*(\sigma^*)\|_2 = \Delta_k$. First note that

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

where $\hat{\mathbf{s}}'(\sigma)$ represents the derivative of $\hat{\mathbf{s}}(\sigma)$. The vector $\hat{\mathbf{s}}'(\sigma)$ is computed by differentiating both sides of the system in (5.8), i.e.,

$$\frac{d}{d\sigma}\left(\begin{bmatrix} (\mathbf{B}_k + \sigma\mathbf{I}) & \mathbf{A}^T \\ \mathbf{A} & \mathbf{0} \end{bmatrix}\begin{bmatrix} \hat{\mathbf{s}} \\ \hat{\rho} \end{bmatrix}\right) = \frac{d}{d\sigma}\left(\begin{bmatrix} -\mathbf{g}_k \\ \mathbf{0} \end{bmatrix}\right),$$

and subsequently solving the resulting equations for $\hat{\mathbf{s}}'$:

$$\begin{bmatrix} (\mathbf{B}_k + \sigma\mathbf{I}) & \mathbf{A}^T \\ \mathbf{A} & \mathbf{0} \end{bmatrix}\begin{bmatrix} \hat{\mathbf{s}}' \\ \hat{\rho}' \end{bmatrix} = \begin{bmatrix} -\hat{\mathbf{s}}(\sigma) \\ \mathbf{0} \end{bmatrix}.$$

The expression for $\hat{\mathbf{s}}' \equiv \hat{\mathbf{s}}'(\sigma)$ is

$$\begin{aligned}\hat{\mathbf{s}}'(\sigma) &= -(\mathbf{B}_k + \sigma\mathbf{I})^{-1}\left(\mathbf{I} - \mathbf{A}^T\left(\mathbf{A}(\mathbf{B}_k + \sigma\mathbf{I})^{-1}\mathbf{A}^T\right)^{-1}\mathbf{A}(\mathbf{B}_k + \sigma\mathbf{I})^{-1}\right)\hat{\mathbf{s}}(\sigma) \\ &= -\hat{\mathbf{H}}_k\hat{\mathbf{W}}_k\hat{\mathbf{s}}(\sigma).\end{aligned}$$

Now observe that the matrix $\hat{\mathbf{H}}_k\hat{\mathbf{W}}_k$ is symmetric positive semidefinite for positive values of σ, so that

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

for $\sigma \geq 0$. This implies that $\phi'(\sigma) \geq 0$ for $\sigma \geq 0$. In other words, $\phi(\sigma)$ is monotonically increasing for positive σ. Because Newton's method is applied in the case that $\|\hat{\mathbf{s}}(0)\|_2 = \|\hat{\mathbf{s}}(0)\|_2 \geq \Delta_k$, corresponding to $\phi(0) < 0$, a unique $\sigma^* > 0$ can be computed that solves the secular-equation in (5.10).

The algorithms developed in this article assume that a feasible initial point $\mathbf{x}_0 \in \mathbb{R}^n$ is given. The approach with a ℓ_2 norm is summarized in Algorithm 5.1

ALGORITHM 5.1:

1. Initialization: $\Delta_0 > 0$,$0 \leq d_- \leq 1 \leq d_+$, $\delta_0 = 1/\|\mathbf{W}\mathbf{g}_0\|_2$, $\mathbf{\Psi}_0 = \mathbf{A}^T$, $\mathbf{M}_0 = \delta_0(\mathbf{A}\mathbf{A}^T)^{-1}$, $\mathbf{B}_0 = \mathbf{I}_n$, $k = 0$;
2. Set: $\mathbf{s}_k = -\left(\delta_k\mathbf{I} - \mathbf{\Psi}_k\mathbf{M}_k\mathbf{\Psi}_k^T\right)\mathbf{g}_k$;
3. If $\|\mathbf{s}_k\|_2 \leq \Delta_k$ and $f(\mathbf{x}_k + \mathbf{s}_k) < f(\mathbf{x}_k)$ then go to 6. otherwise go to 4.;
4. Set $\mathbf{s}_k = \mathbf{s}^*$(eq.(5.9));
5. If $f(\mathbf{x}_k + \mathbf{s}_k) < f(\mathbf{x}_k)$ then go to 6., otherwise $\Delta_k = d_-\Delta_k$ go to 4.;
6. Set $\Delta_k = d_+\Delta_k$, $\mathbf{x}_k = \mathbf{x}_k + \mathbf{s}_k$, update $\mathbf{\Psi}_k, \mathbf{M}_k, \mathbf{B}_k, \mathbf{g}_k, \delta_k, k = k + 1$ go to 2.;

5.4 TRUST-REGION APPROACH WITH A SHAPE-CHANGING CONSTRAINT

Instead of the ℓ_2-norm, we will next describe how to apply a shape-changing norm in the context of a constrained trust-region method. First, we use a transformation based on the eigenvectors of the (1,1) block of the inverse KKT matrix from (5.6), in order to simplify the trust-region subproblem. Then we propose an effective method to obtain the eigendecomposition of the relevant KKT block. Finally, we decouple the transformed trust-region subproblem by applying a shape-changing norm, and compute analytic solutions.

5.4.1 TRANSFORMATION OF THE TRUST-REGION SUBPROBLEM

In this subsection we transform the quadratic objective function from (5.3), by a change of variables. The transformed objective can then be combined with a shape-changing-norm, to effectively solve the subproblem. Observe that the nullspace of $\mathbf{K}_{11}^{-1}$ is spanned by the columns of $\mathbf{A}^T$, i.e. $\mathbf{K}_{11}^{-1}\mathbf{A}^T = \mathbf{0}$. Therefore, the eigendecomposition of $\mathbf{K}_{11}^{-1}$ can be represented as

$$\mathbf{K}_{11}^{-1} = \mathbf{P}\Lambda\mathbf{P}^T = [\mathbf{P}_{\|1} \quad \mathbf{P}_2] \begin{bmatrix} \mathbf{0}_m & \\ & \mathbf{\Lambda}_2^{-1} \end{bmatrix} [\mathbf{P}_{\|1} \quad \mathbf{P}_2]^T, \tag{5.11}$$

where the orthonormal eigenvectors are contained in the matrix $\mathbf{P} = [\mathbf{P}_{\|1} \quad \mathbf{P}_2] \in \mathbb{R}^{n\times n}$ ($\mathbf{I} = \mathbf{P}\mathbf{P}^T = \mathbf{P}^T\mathbf{P}$), and where $\mathbf{\Lambda}_2^{-1} \in \mathbb{R}^{(n-m)\times(n-m)}$ symbolizes a diagonal matrix

of eigenvalues. In this notation the rectangular matrix $\mathbf{P}_{\|1} \in \mathbb{R}^{n\times m}$ represents the eigenvectors corresponding to the zero eigenvalues of $\mathbf{K}_{11}^{-1}$. With this, we deduce, since $\mathbf{A}^T \in \text{Range}\left(\mathbf{P}_{\|1}\right)$, the following identity: $\mathbf{P}_2^T\mathbf{A}^T = \mathbf{0}_{(n-m)\times m}$. In order to diagonalize and to decouple the subproblem in (5.3), we first propose Lemma 1 that describes the relation between the eigenvectors in the rectangular matrix $\mathbf{P}_2 \in \mathbb{R}^{n\times(n-m)}$ and the quasi-Newton matrix $\mathbf{B}_k$.

Lemma 5.2. *The identity* $\mathbf{P}_2^T\mathbf{B}_k\mathbf{P}_2 = \mathbf{\Lambda}_2$ *holds for the eigenvectors and eigenvalues from* (5.11).

Proof. Multiply the (1,1) block of $\mathbf{K}\mathbf{K}^{-1}$ by $\mathbf{P}_2^T$ from the left and by $\mathbf{P}_2$ from the right, so that:

$$\begin{aligned}\mathbf{P}_2^T[\mathbf{I}_n \quad \mathbf{0}_{n\times m}]\mathbf{K}\mathbf{K}^{-1}\begin{bmatrix}\mathbf{I}_n\\ \mathbf{0}_{m\times n}\end{bmatrix}\mathbf{P}_2 &= \mathbf{P}_2^T\left(\mathbf{B}_k\mathbf{K}_{11}^{-1} + \mathbf{A}^T\mathbf{K}_{21}^{-1}\right)\mathbf{P}_2\\ &= \mathbf{P}_2^T\left(\mathbf{B}_k\mathbf{P}\Lambda\mathbf{P}^T + \mathbf{A}^T\mathbf{K}_{21}^{-1}\right)\mathbf{P}_2\\ &= \mathbf{P}_2^T\mathbf{B}_k\mathbf{P}_2\mathbf{\Lambda}_2^{-1}.\end{aligned}$$

Now, since $\mathbf{K}\mathbf{K}^{-1} = \mathbf{I}_{n+m}$, and since the matrix $\mathbf{P}_2$ has orthonormal columns, it holds that

$$\mathbf{P}_2^T\mathbf{B}_k\mathbf{P}_2\mathbf{\Lambda}_2^{-1} = \mathbf{I}_{n-m}.$$

Therefore we conclude that $\mathbf{P}_2^T\mathbf{B}_k\mathbf{P}_2 = \mathbf{\Lambda}_2$. ■

To diagonalize the constrained subproblem we apply the change of variables $\mathbf{v} = \begin{bmatrix}\mathbf{v}_{\|1}\\ \mathbf{v}_2\end{bmatrix} = \begin{bmatrix}\mathbf{P}_{\|1}^T\mathbf{s}\\ \mathbf{P}_2^T\mathbf{s}\end{bmatrix} = \mathbf{P}^T\mathbf{s}$. First, we analyze the effect of this change on the linear constraints:

$$\mathbf{A}\mathbf{s} = \mathbf{A}\mathbf{P}\mathbf{v} = [\mathbf{A}\mathbf{P}_{\|1} \quad \mathbf{0}_{m\times(n-m)}]\mathbf{v} = \mathbf{A}\mathbf{P}_{\|1}\mathbf{v}_{\|1} = \mathbf{0}.$$

Since $\mathbf{A}^T$ is in the range of $\mathbf{P}_{\|1}$, we may represent $\mathbf{A}^T$ as $\mathbf{A}^T = \mathbf{P}_{\|1}\mathbf{R}_1$ with an $m \times m$ invertible matrix $\mathbf{R}_1$. Thus the constraint

$$\mathbf{A}\mathbf{P}_{\|1}\mathbf{v}_{\|1} = \left(\mathbf{R}_1^T\mathbf{P}_{\|1}^T\right)\mathbf{P}_{\|1}\mathbf{v}_{\|1} = \mathbf{R}_1^T\mathbf{v}_{\|1} = \mathbf{0},$$

implies that $\mathbf{v}_{\|1} = \mathbf{0}$. Second, we examine the effect of the change of variables on the quadratic approximation $Q(\mathbf{s})$:

$$\begin{aligned}Q(\mathbf{s}) = Q(\mathbf{P}\mathbf{v}) &= \mathbf{g}_k^T(\mathbf{P}\mathbf{v}) + \frac{1}{2}\mathbf{v}^T\mathbf{P}^T\mathbf{B}_k\mathbf{P}\mathbf{v}\\ &= \mathbf{g}_k^T(\mathbf{P}_2\mathbf{v}_2) + \frac{1}{2}\mathbf{v}_2^T\mathbf{P}_2^T\mathbf{B}_k\mathbf{P}_2\mathbf{v}_2\\ &= \mathbf{g}_k^T(\mathbf{P}_2\mathbf{v}_2) + \frac{1}{2}\mathbf{v}_2^T\mathbf{\Lambda}_2\mathbf{v}_2\\ &\equiv q(\mathbf{v}_2),\end{aligned}$$

where we apply Lemma 5.2 to obtain the last equality. Observe that $Q(\mathbf{P}\mathbf{v}) \equiv q(\mathbf{v}_2)$ is an quadratic objective function defined by the diagonal matrix of eigenvalues $\mathbf{\Lambda}_2$ and by the $n-m$ unknowns denoted as $\mathbf{v}_2$. Because the matrix in the transformed quadratic objective function is diagonal, it will be straightforward to later separate it into different subspaces. Since the change of variables relies on the eigendecomposition of the matrix $\mathbf{K}_{11}^{-1}$, we first develop an effective partial eigenfactorization of $\mathbf{K}_{11}^{-1}$.

5.4.2 PARTIAL EIGENDECOMPOSITION OF $\mathbf{K}_{11}^{-1}$

Our approach to computing the eidendecomposition of (5.7) is based on factoring the low rank matrix $\mathbf{U}_k \equiv \mathbf{\Psi}_k \mathbf{M}_k \mathbf{\Psi}_k^T$. Note that since $\text{Rank}(\mathbf{U}_k) = 2l + m$, the matrix $\mathbf{U}_k$ has only $2l+m$ non-zero eigenvalues. We denote the orthonormal basis that corresponds to the nullspace of $\mathbf{U}_k$ as $\mathbf{P}_\perp \in \mathbb{R}^{n\times(n-(2l+m))}$. In other words, the matrix $\mathbf{P}_\perp$ satisfies $\mathbf{U}_k\mathbf{P}_\perp = \mathbf{0}$. However, since the matrix $\mathbf{P}_\perp$ would be prohibitively expensive to compute when n becomes large, we do not explicitly compute it in our approach. Next observe, from e.g. (5.6), that $\mathbf{K}_{11}^{-1}\mathbf{A}^T = (\delta_k\mathbf{I} - \mathbf{U}_k)\mathbf{A}^T = \mathbf{0}$ so that $\mathbf{U}_k\mathbf{A}^T = \delta_k\mathbf{A}^T$. Therefore the set of eigenvectors of $\mathbf{U}_k$, corresponding to the repeated eigenvalue δ_k, can be expressed as $\mathbf{P}_{\|1} = \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1/2}$. We thus represent the factorization of $\mathbf{U}_k$ as

$$\mathbf{U}_k = [\mathbf{A}^T \ \hat{\mathbf{\Psi}}_k]\mathbf{M}_k \begin{bmatrix} \mathbf{A} \\ \hat{\mathbf{\Psi}}_k^T \end{bmatrix} = [\mathbf{P}_{\|1} \ \mathbf{P}_{\|2} \ \mathbf{P}_\perp] \begin{bmatrix} \delta_k\mathbf{I}_m & & \\ & \mathbf{D}_2^{-1} & \\ & & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{P}_{\|1}^T \\ \mathbf{P}_{\|2}^T \\ \mathbf{P}_\perp^T \end{bmatrix}, \tag{5.12}$$

where $\mathbf{D}_2^{-1} \in \mathbb{R}^{2l\times 2l}$ is a diagonal matrix of eigenvalues, and the columns of the rectangular matrices $\mathbf{P}_{\|1} \in \mathbb{R}^{n\times m}$, $\mathbf{P}_{\|2} \in \mathbb{R}^{n\times 2l}$, $\mathbf{P}_\perp \in \mathbb{R}^{n\times(n-(2l+m))}$ represent mutually orthogonal eigenvectors. Next we describe how to compute the non-zero eigenvalues in the matrix $\mathbf{D}_2^{-1}$ and its corresponding eigenvectors in $\mathbf{P}_{\|2}$. Subsequently, we combine the results to arrive at the implicit eigendecomposition of (5.7). First, deflate the right hand side in (5.12), by substituting $\mathbf{P}_{\|1} = \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1/2}$, and by suppressing the zero eigenvalues. In this way we obtain an identity for the eigenvectors $\mathbf{P}_{\|2}$ and eigenvalues $\mathbf{D}_2^{-1}$:

$$[\mathbf{A}^T \ \hat{\mathbf{\Psi}}_k] \begin{bmatrix} \mathbf{M}_{11} - \left(\mathbf{A}\mathbf{A}^T\right)^{-1} & \mathbf{M}_{12} \\ \mathbf{M}_{12}^T & \mathbf{M}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{A} \\ \hat{\mathbf{\Psi}}_k^T \end{bmatrix} = \mathbf{P}_{\|2}\mathbf{D}_2^{-1}\mathbf{P}_{\|2}^T. \tag{5.13}$$

Since $\mathbf{0} = \mathbf{P}_{\|1}^T\mathbf{P}_{\|2} = (\mathbf{A}\mathbf{A}^T)^{-1/2}\mathbf{A}\mathbf{P}_{\|2}$, which means that $\mathbf{A}\mathbf{P}_{\|2} = \mathbf{0}$, then we apply the orthogonal projection $\mathbf{W} = \mathbf{I} - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A}$ from the left and from the right to the expression in (5.13), so that

$$\mathbf{W}\hat{\mathbf{\Psi}}_k\mathbf{M}_{22}\hat{\mathbf{\Psi}}_k^T\mathbf{W}^T = \mathbf{P}_{\|2}\mathbf{D}_2^{-1}\mathbf{P}_{\|2}^T.$$

Next compute the non-zero eigenvalues of $\mathbf{W}\hat{\mathbf{\Psi}}_k\mathbf{M}_{22}\hat{\mathbf{\Psi}}_k^T\mathbf{W}^T$ using an implicit QR factorization of the matrix $\mathbf{W}\hat{\mathbf{\Psi}}_k \in \mathbb{R}^{n\times 2l}$. In particular, let $\mathbf{W}\hat{\mathbf{\Psi}}_k = \mathbf{QR}$ where we do not explicitly compute the orthogonal matrix $\mathbf{Q} \in \mathbb{R}^{n\times 2l}$, and where $\mathbf{R} \in \mathbb{R}^{2l\times 2l}$ is an upper triangular matrix. Then we explicitly compute the eigendecomposition of the small matrix $\mathbf{R}\mathbf{M}_{22}\mathbf{R}^T = \mathbf{V}\tilde{\mathbf{D}}_2^{-1}\mathbf{V}^T$, where $\mathbf{V} \in \mathbb{R}^{2l\times 2l}$ is orthogonal and $\tilde{\mathbf{D}}_2^{-1}$ is a diagonal matrix of eigenvalues. With this

$$\mathbf{W}\hat{\mathbf{\Psi}}_k\mathbf{M}_{22}\hat{\mathbf{\Psi}}_k^T\mathbf{W}^T = \mathbf{QR}\mathbf{M}_{22}\mathbf{R}^T\mathbf{Q}^T = \mathbf{QV}\tilde{\mathbf{D}}_2^{-1}\mathbf{R}^T\mathbf{V}^T.$$

Now, since $\mathbf{QV}$ is a matrix of orthonormal columns, and $\tilde{\mathbf{D}}_2^{-1}$ is diagonal we find that $\mathbf{QV} = \mathbf{P}_{\|2}$ and $\tilde{\mathbf{D}}_2^{-1} = \mathbf{D}_2^{-1}$. Note that $\mathbf{Q}$ is not explicitly computed, instead we represent it with the identity $\mathbf{QR} = \mathbf{W}\hat{\mathbf{\Psi}}_k$ as $\mathbf{Q} = \mathbf{W}\hat{\mathbf{\Psi}}_k\mathbf{R}^{-1}$. Thus $\mathbf{P}_{\|2}$ has the explicit form

$$\mathbf{P}_{\|2} = \mathbf{QV} = \mathbf{W}\hat{\mathbf{\Psi}}_k\mathbf{R}^{-1}\mathbf{V} = \left(\hat{\mathbf{\Psi}}_k - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A}\hat{\mathbf{\Psi}}_k\right)\mathbf{R}^{-1}\mathbf{V}.$$

To compute the upper triangular matrix $\mathbf{R}$ we set $\mathbf{R} = \mathbf{L}\mathbf{D}^{1/2}$, where we compute the LDL^T factorization $(\mathbf{W}\hat{\mathbf{\Psi}}_k)^T\mathbf{W}\hat{\mathbf{\Psi}}_k = (\mathbf{QR})^T\mathbf{QR} = \mathbf{LDL}^T$. Based on the previous analysis we now obtain the implicit eigendecomposition of $\mathbf{K}_{11}^{-1}$ from (5.7). Since the columns of $\mathbf{P} = [\mathbf{P}_{\|1}\ \mathbf{P}_{\|2}\ \mathbf{P}_\perp] \in \mathbb{R}^{n\times n}$ are mutually orthogonal, we obtain the factorization

$$\begin{aligned}
\mathbf{K}_{11}^{-1} &= \delta_k\mathbf{I} - [\mathbf{A}^T\ \hat{\mathbf{\Psi}}_k]\begin{bmatrix}\mathbf{M}_{11} & \mathbf{M}_{12}\\ \mathbf{M}_{12}^T & \mathbf{M}_{22}\end{bmatrix}\begin{bmatrix}\mathbf{A}\\ \hat{\mathbf{\Psi}}_k^T\end{bmatrix}\\
&= \delta_k\mathbf{I} - [\mathbf{P}_{\|1}\ \mathbf{P}_{\|2}\ \mathbf{P}_\perp]\begin{bmatrix}\delta_k\mathbf{I}_m & & \\ & \mathbf{D}_2^{-1} & \\ & & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{P}_{\|1}^T\\ \mathbf{P}_{\|2}^T\\ \mathbf{P}_\perp^T\end{bmatrix}\\
&= [\mathbf{P}_{\|1}\ \mathbf{P}_{\|2}\ \mathbf{P}_\perp]\begin{bmatrix}\mathbf{0}_m & & \\ & \delta_k\mathbf{I}_{2l} - \mathbf{D}_2^{-1} & \\ & & \delta_k\mathbf{I}_{n-(2l+m)}\end{bmatrix}\begin{bmatrix}\mathbf{P}_{\|1}^T\\ \mathbf{P}_{\|2}^T\\ \mathbf{P}_\perp^T\end{bmatrix},
\end{aligned} \tag{5.14}$$

where

$$\mathbf{P}_{\|1} = \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1/2}, \quad \mathbf{P}_{\|2} = \left(\hat{\mathbf{\Psi}}_k - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A}\hat{\mathbf{\Psi}}_k\right)\mathbf{R}^{-1}\mathbf{V}, \quad \mathbf{D}_2^{-1} = \tilde{\mathbf{D}}_2^{-1}.$$

Observe that the eigendecomposition only requires the computation of a few eigenvalues $(2l)$, corresponding to $(\delta_k\mathbf{I}_{2l} - \tilde{\mathbf{D}}_2^{-1})$ explicitly. Moreover, we do not compute the columns of the matrix $\mathbf{P}_\perp$ explicitly.

5.4.3 SOLVING THE SHAPE-CHANGING SUBPROBLEM

This subsection describes the solution to the subproblem in (5.3), with the shape-changing $(\mathbf{P}, \infty)$-norm:

$$\|\mathbf{s}\|_{\mathbf{P},\infty} \equiv \max\left(\left\|\mathbf{v}_{\|}\right\|_{\infty}, \|\mathbf{v}_{\perp}\|_2\right). \tag{5.15}$$

Here we define $\mathbf{v}_{\|} = \mathbf{P}_{\|}^T\mathbf{s} = \begin{bmatrix}\mathbf{P}_{\|1}^T\mathbf{s} \\ \mathbf{P}_{\|2}^T\mathbf{s}\end{bmatrix} = \begin{bmatrix}\mathbf{v}_{\|1} \\ \mathbf{v}_{\|2}\end{bmatrix}$, and $\mathbf{v}_{\perp} = \mathbf{P}_{\perp}^T\mathbf{s}$. Then, using the change of variable $\mathbf{s} = \mathbf{P}\mathbf{v}$, the original minimization has an equivalent expression

$$\underset{\substack{\|\mathbf{s}\|_{\mathbf{P},\infty} \le \Delta_k \\ \mathbf{A}\mathbf{s} = \mathbf{0}}}{\text{minimize}} \; Q(\mathbf{s}) \quad = \quad \underset{\substack{\|\mathbf{P}\mathbf{v}\|_{\mathbf{P},\infty} \le \Delta_k \\ \mathbf{v}_{\|1} = \mathbf{0}}}{\text{minimize}} \; Q(\mathbf{P}\mathbf{v}).$$

The objective function $Q(\mathbf{P}\mathbf{v})$ and the shape-changing norm now further decouple. First recall, from Subsection 5.4.1, that $Q(\mathbf{P}\mathbf{v}) = q(\mathbf{v}_2)$. Then, denoting $\mathbf{P}_2 = [\mathbf{P}_{\|1}\ \mathbf{P}_{\|2}]$, we define the vectors $\mathbf{P}_2^T\mathbf{g}_k \equiv \begin{bmatrix}\mathbf{P}_{\|2}^T\mathbf{g}_k \\ \mathbf{P}_{\perp}^T\mathbf{g}_k\end{bmatrix} = \begin{bmatrix}\mathbf{g}_{\|2} \\ \mathbf{g}_{\perp}\end{bmatrix}$, and $\mathbf{v}_2 \equiv \begin{bmatrix}\mathbf{v}_{\|2} \\ \mathbf{v}_{\perp}\end{bmatrix}$. With the notation $\mathbf{\Lambda}_2 = \begin{bmatrix}(\delta_k\mathbf{I} - \tilde{\mathbf{D}}_2^{-1})^{-1} & \\ & \gamma_k\mathbf{I}\end{bmatrix} \equiv \begin{bmatrix}\bar{\Lambda}_2 & \\ & \gamma_k\mathbf{I}\end{bmatrix}$, and the result of Lemma 5.2, then $Q(\mathbf{P}\mathbf{v})$ separates once more:

$$\begin{aligned} Q(\mathbf{P}\mathbf{v}) = q(\mathbf{v}_2) &= \mathbf{g}_k^T(\mathbf{P}_2\mathbf{v}_2) + \frac{1}{2}\mathbf{v}_2^T\mathbf{P}_2^T\mathbf{B}_k\mathbf{P}_2\mathbf{v}_2 \\ &= \mathbf{g}_{\|2}^T\mathbf{v}_{\|2} + \mathbf{g}_{\perp}^T\mathbf{v}_{\perp} + \frac{1}{2}\left(\mathbf{v}_{\|2}^T\bar{\Lambda}_2\mathbf{v}_{\|2} + \gamma_k\|\mathbf{v}_{\perp}\|_2^2\right). \end{aligned}$$

In other words, the objective function can be further decoupled into the variables $\mathbf{v}_{\|2}$ and $\mathbf{v}_{\perp}$. Simultaneously, the shape-changing constraint also decouples into $\mathbf{v}_{\|2}$ and $\mathbf{v}_{\perp}$. Since $\mathbf{v}_{\|1} = \mathbf{0}$, then the norm constraint satisfies the equivalence relation:

$$\max\left(\left\|\mathbf{v}_{\|}\right\|_{\infty}, \|\mathbf{v}_{\perp}\|_2\right) \le \Delta_k \quad \textit{is equivalent to} \quad \max\left(\left\|\mathbf{v}_{\|2}\right\|_{\infty}, \|\mathbf{v}_{\perp}\|_2\right) \le \Delta_k.$$

Using our analysis, the minimization of the subproblem (5.3) with the shape-changing norm as defined in (5.15), is equivalent to the minimization of two decoupled diagonalized problems

$$\underset{\|\mathbf{v}_{\|2}\|_{\infty} \le \Delta_k}{\text{minimize}} \; \mathbf{g}_{\|2}^T\mathbf{v}_{\|2} + \frac{1}{2}\mathbf{v}_{\|2}^T\bar{\mathbf{\Lambda}}_2\mathbf{v}_{\|2} + \underset{\|\mathbf{v}_{\perp}\|_2 \le \Delta_k}{\text{minimize}} \; \mathbf{g}_{\perp}^T\mathbf{v}_{\perp} + \frac{\gamma_k}{2}\|\mathbf{v}_{\perp}\|_2^2.$$

The two separated minimization problems can be solved analytically. Specifically, letting $\lambda_i \equiv [\bar{\mathbf{\Lambda}}_2]_{ii} = [(\delta_k \mathbf{I} - \tilde{\mathbf{D}}_2^{-1})^{-1}]_{ii}$, then the solution of $\mathbf{v}_{\|2}^*$ is given, coordinate-wise, by

$$[\mathbf{v}_{\|2}^*]_i = \begin{cases} -\frac{\left[\mathbf{g}_{\|2}\right]_i}{\lambda_i} & \text{if } \left|\frac{\left[\mathbf{g}_{\|2}\right]_i}{\lambda_i}\right| \leq \Delta_k \text{ and } \lambda_i > 0, \\ c & \text{if } \left[\mathbf{g}_{\|2}\right]_i = 0 \text{ and } \lambda_i = 0, \\ -\text{sgn}(\left[\mathbf{g}_{\|2}\right]_i)\Delta_k & \text{if } \left[\mathbf{g}_{\|2}\right]_i \neq 0 \text{ and } \lambda_i = 0, \\ \pm\Delta_k & \text{if } \left[\mathbf{g}_{\|2}\right]_i = 0 \text{ and } \lambda_i < 0, \\ -\frac{\Delta_k}{\left|\left[\mathbf{g}_{\|2}\right]_i\right|}\left[\mathbf{g}_{\|2}\right]_i & \text{otherwise,} \end{cases} \quad , \tag{5.16}$$

where c is any real number in $[-\Delta_k, \Delta_k]$ and 'sgn' denotes the signum function. The minimizer $\mathbf{v}_\perp^*$ is given by

$$\mathbf{v}_\perp^* = \beta \mathbf{g}_\perp, \tag{5.17}$$

where

$$\beta = \begin{cases} -\delta_k & \text{if } \delta_k > 0 \text{ and } \|\delta_k \mathbf{g}_\perp\|_2 \leq \Delta_k, \\ -\frac{\Delta_k}{\|\mathbf{g}_\perp\|_2} & \text{otherwise.} \end{cases} \tag{5.18}$$

Thus, $\mathbf{v}^* = \begin{bmatrix} \mathbf{0} \\ \mathbf{v}_{\|2}^* \\ \mathbf{v}_\perp^* \end{bmatrix}$, yields the solution $\mathbf{s}^*$, by using the transformation $\mathbf{s}^* = \mathbf{P}\mathbf{v}^*$. Since our method does not compute the large orthonormal matrix $\mathbf{P}_\perp$, the next subsection describes a direct formula to compute $\mathbf{s}^*$.

5.4.4 COMPUTING THE SOLUTION $\mathbf{s}^*$

Observe that since $\mathbf{P} = [\mathbf{P}_{\|1}\ \mathbf{P}_{\|2}\ \mathbf{P}_\perp]$ is orthogonal, then $\mathbf{P}\mathbf{P}^T = \mathbf{P}_{\|1}\mathbf{P}_{\|1}^T + \mathbf{P}_{\|2}\mathbf{P}_{\|2}^T + \mathbf{P}_\perp\mathbf{P}_\perp^T = \mathbf{I}_n$. Therefore, $\mathbf{P}_\perp\mathbf{P}_\perp^T = \mathbf{I}_n - \mathbf{P}_{\|1}\mathbf{P}_{\|1}^T - \mathbf{P}_{\|2}\mathbf{P}_{\|2}^T$ and the previous computations, yield

$$\begin{aligned} \mathbf{s}^* &= \mathbf{P}\mathbf{v}^* \\ &= [\mathbf{P}_{\|1}\ \mathbf{P}_{\|2}\ \mathbf{P}_\perp] \begin{bmatrix} \mathbf{0} \\ \mathbf{v}_{\|2}^* \\ \mathbf{v}_\perp^* \end{bmatrix} \\ &= \mathbf{P}_{\|2}\mathbf{v}_{\|2}^* + \mathbf{P}_\perp\mathbf{v}_\perp^* \\ &= \mathbf{P}_{\|2}\mathbf{v}_{\|2}^* + \beta\mathbf{P}_\perp\mathbf{P}_\perp^T\mathbf{g}_k \\ &= \mathbf{P}_{\|2}\left(\mathbf{v}_{\|2}^* - \beta\mathbf{g}_{\|2}\right) + \beta\left(\mathbf{I}_n - \mathbf{A}^T\left(\mathbf{A}\mathbf{A}^T\right)^{-1}\mathbf{A}\right)\mathbf{g}_k. \end{aligned} \tag{5.19}$$

Ultimately, using the explicit form of $\mathbf{P}_{\|2}$ in (5.14), then $\mathbf{s}^*$ is expressed by the formula

$$\begin{aligned}\mathbf{s}^* = &\left(\hat{\mathbf{\Psi}}_k - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A}\hat{\mathbf{\Psi}}_k\right)\mathbf{R}^{-1}\mathbf{V}\left(\mathbf{v}^*_{\|2} - \beta\mathbf{g}_{\|2}\right)\\ &+\beta\left(\mathbf{g}_k - \mathbf{A}^T\left(\mathbf{A}\mathbf{A}^T\right)^{-1}\mathbf{A}\mathbf{g}_k\right).\end{aligned} \tag{5.20}$$

The minimization using the shape-changing infinity from this section is summarized in Algorithm 5.2:

ALGORITHM 5.2:

1. Initialization: $\Delta_0 > 0$,$0 \leq d_- \leq 1 \leq d_+$, $\delta_0 = 1/\|\mathbf{W}\mathbf{g}_0\|_2$, $\mathbf{\Psi}_0 = \mathbf{A}^T$, $\mathbf{M}_0 = \delta_0(\mathbf{A}\mathbf{A}^T)^{-1}$, $\mathbf{B}_0 = \mathbf{I}_n$, $k = 0$;
2. Set: $\mathbf{s}_k = -\left(\delta_k\mathbf{I} - \mathbf{\Psi}_k\mathbf{M}_k\mathbf{\Psi}_k^T\right)\mathbf{g}_k$;
3. If $\|\mathbf{s}_k\|_2 \leq \Delta_k$ and $f(\mathbf{x}_k + \mathbf{s}_k) < f(\mathbf{x}_k)$ then go to 6. otherwise go to 4.;
4. Set $\mathbf{s}_k = \mathbf{s}^*$(eq.(5.20));
5. If $f(\mathbf{x}_k + \mathbf{s}_k) < f(\mathbf{x}_k)$ then go to 6., otherwise $\Delta_k = d_-\Delta_k$ go to 4.;
6. Set $\Delta_k = d_+\Delta_k$, $\mathbf{x}_k = \mathbf{x}_k + \mathbf{s}_k$, update $\mathbf{\Psi}_k, \mathbf{M}_k, \mathbf{B}_k, \mathbf{g}_k, \delta_k, k = k + 1$ go to 2.;

5.5 ANALYSIS

This section analyses the convergence properties of the two methods proposed in this article. The analysis is based on the *sufficient decrease* principle, for trust-region methods [CGT00]. The sufficient decrease principle requires that a computed minimizer to the trust-region subproblem reduces the quadratic approximation $Q(\mathbf{s})$, by an satisfactory amount. Specifically, in [YuaNA], for constrained trust-region subproblems, the sufficient decrease condition is formulated as:

$$Q(\mathbf{0}) - Q(\mathbf{s}) \geq c\varepsilon_k \cdot \min(\varepsilon_k/\|\mathbf{B}_k\|_2, \Delta_k), \tag{5.21}$$

where $c > 0$, $\varepsilon_k = \|\mathbf{g}_k - \mathbf{A}^T\rho\|_2$ and $\rho \in \mathbb{R}^m$ is a vector of Lagrange multipliers. If the steps in a trust-region algorithm satisfy the condition (5.21), then global convergence of the algorithm is deduced, as the method is guaranteed a sufficient improvement at each accepted trial step [CGT00, YuaNA].

Observe that both proposed algorithms (statements 2.), first test the 'unconstrained' minimizer $\mathbf{s}_k = \mathbf{s}_u = -\mathbf{K}_{11}^{-1}\mathbf{g}_k$ from (5.5). Only if the length of this direction exceeds the trust-region constraint, $\|\mathbf{s}_u\|_2 \geq \Delta_k$, then the steps computed by the two methods will be different. The next subsection demonstrates that $\mathbf{s}_u$ satisfies the sufficient decrease condition (5.21).

5.5.1 SUFFICIENT DECREASE WITH THE 'UNCONSTRAINED' MINIMIZER $\mathbf{s}_u$

First observe, from the definition of $\mathbf{W}_k$ in (5.6), that $\mathbf{W}_k^T\mathbf{H}_k\mathbf{W}_k = \mathbf{H}_k\mathbf{W}_k$. Then substitute the unconstrained minimizer, $\mathbf{s}_k = \mathbf{s}_u = -\mathbf{K}_{11}^{-1}\mathbf{g}_k = -\mathbf{H}_k\mathbf{W}_k\mathbf{g}_k$ into the left hand side of (5.21), to obtain

$$\begin{aligned} Q(\mathbf{0}) - Q(\mathbf{s}_k) &= \mathbf{g}_k^T\mathbf{H}_k\mathbf{W}_k\mathbf{g}_k - 1/2\mathbf{g}_k^T\mathbf{W}_k^T\mathbf{H}_k\mathbf{W}_k\mathbf{g}_k \\ &= 1/2\mathbf{g}_k^T\mathbf{W}_k^T\mathbf{H}_k\mathbf{W}_k\mathbf{g}_k \\ &\geq \frac{1}{2\hat{\lambda}_{\max}}\mathbf{g}_k^T\mathbf{W}_k^T\mathbf{W}_k\mathbf{g}_k \\ &\equiv 1/2\frac{(\varepsilon_k^u)^2}{\|\mathbf{B}_k\|_2} \\ &= 1/2\varepsilon_k^u \cdot \min(\varepsilon_k^u/\|\mathbf{B}_k\|_2, \Delta_k), \end{aligned} \tag{5.22}$$

where $\varepsilon_k^u = \|\mathbf{W}_k\mathbf{g}_k\|_2 = \|\mathbf{g}_k - \mathbf{A}^T\rho_u\|_2$ and where ρ_u is specified by the system in (5.5). Comparing the last equality from (5.22) with (5.21), we conclude that the unconstrained minimizer satisfies the sufficient decrease condition.

5.5.2 SUFFICIENT DECREASE WITH THE ℓ_2 CONSTRAINT

In Algorithm 5.1, with the ℓ_2 constraint the search direction is computed as

$$\hat{\mathbf{s}}_k^* = -\hat{\mathbf{H}}_k^*\hat{\mathbf{W}}_k^*\mathbf{g}_k = -(\mathbf{B}_k + \sigma^*\mathbf{I})^{-1}\hat{\mathbf{W}}_k^*\mathbf{g}_k,$$

where $\hat{\mathbf{W}}_k^*$ is as defined below (5.9). Next it is demonstrated that $\hat{\mathbf{s}}_k^*$ satisfies the sufficient decrease condition. In particular, we propose

Lemma 5.3. *The solution*

$$\hat{\mathbf{s}}_k^* = -(\mathbf{B}_k + \sigma^*\mathbf{I}_n)^{-1}\left(\mathbf{I} - \mathbf{A}^T\left(\mathbf{A}(\mathbf{B}_k + \sigma^*\mathbf{I}_n)^{-1}\mathbf{A}^T\right)^{-1}\mathbf{A}(\mathbf{B}_k + \sigma^*\mathbf{I}_n)^{-1}\right)\mathbf{g}_k$$

where $\sigma^* \geq 0$, of the trust-region subproblem in (5.2) *defined by the ℓ_2-norm satisfies the sufficient decrease condition from* (5.21).

Proof. First, when $\sigma^* = 0$ then $\hat{\mathbf{s}}_k^* = \mathbf{s}_u$, so that the sufficient decrease condition is as in (5.22). If $\sigma^* > 0$, then $\hat{\mathbf{s}}_k^*$ lies on the boundary, i.e. $\|\hat{\mathbf{s}}_k^*\|_2 = \Delta_k$. In this case, observe that

$$(\hat{\mathbf{s}}_k^*)^T(\mathbf{B}_k + \sigma^*\mathbf{I})\hat{\mathbf{s}}_k^* = (\hat{\mathbf{s}}_k^*)^T\mathbf{B}_k\mathbf{s}_k + \sigma^*\Delta_k^2 = -(\hat{\mathbf{s}}_k^*)^T\hat{\mathbf{W}}_k^*\mathbf{g}_k = \mathbf{g}_k^T(\hat{\mathbf{W}}_k^*)^T\hat{\mathbf{H}}_k^*\hat{\mathbf{W}}_k^*\mathbf{g}_k.$$

Moreover, from the definition of $\hat{\mathbf{W}}_k^*$, it holds that $(\hat{\mathbf{W}}_k^*)^T\hat{\mathbf{H}}_k^*\hat{\mathbf{W}}_k^* = \hat{\mathbf{H}}_k^*\hat{\mathbf{W}}_k^*$. Therefore

$$\begin{aligned}
Q(\mathbf{0}) - Q(\hat{\mathbf{s}}_k^*) &= -(\mathbf{g}_k^T\hat{\mathbf{s}}_k^* + 1/2(\hat{\mathbf{s}}_k^*)^T\mathbf{B}_k\hat{\mathbf{s}}_k^*) \\
&= -\left(\mathbf{g}_k^T\hat{\mathbf{s}}_k^* + 1/2(\mathbf{g}_k^T(\hat{\mathbf{W}}_k^*)^T\hat{\mathbf{H}}_k^*\hat{\mathbf{W}}_k^*\mathbf{g}_k - 1/2\sigma^*\Delta_k^2)\right) \\
&= -\left(-\mathbf{g}_k^T\hat{\mathbf{H}}_k^*\hat{\mathbf{W}}_k^*\mathbf{g}_k + 1/2(\mathbf{g}_k^T(\hat{\mathbf{W}}_k^*)^T\hat{\mathbf{H}}_k^*\hat{\mathbf{W}}_k^*\mathbf{g}_k - 1/2\sigma^*\Delta_k^2)\right) \\
&= 1/2\mathbf{g}_k^T(\hat{\mathbf{W}}_k^*)^T\hat{\mathbf{H}}_k^*\hat{\mathbf{W}}_k^*\mathbf{g}_k + 1/2\sigma^*\Delta_k^2.
\end{aligned}$$

Since $\hat{\mathbf{H}}_k^* = (\mathbf{B}_k + \sigma^*\mathbf{I})^{-1}$ we deduce that $\frac{\|\hat{\mathbf{W}}_k^*\mathbf{g}_k\|_2^2}{\hat{\lambda}_{\max}+\sigma^*} \leq \mathbf{g}_k^T(\hat{\mathbf{W}}_k^*)^T\hat{\mathbf{H}}_k^*\hat{\mathbf{W}}_k^*\mathbf{g}_k$. Moreover since

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

an upper bound on σ^* is: $\sigma^* \leq \frac{\|\hat{\mathbf{W}}_k^*\mathbf{g}_k\|_2}{\Delta_k}$. Letting $\varepsilon_k^l = \|\hat{\mathbf{W}}_k^*\mathbf{g}_k\|_2 = \|\mathbf{g}_k - \mathbf{A}^T\hat{\rho}^*\|_2$, where $\hat{\rho}^*$ is specified by the system in (5.8), then the inequalities hold

$$\begin{aligned}
Q(\mathbf{0}) - Q(\mathbf{s}_k) &= 1/2\mathbf{g}_k^T(\hat{\mathbf{W}}_k^*)^T\hat{\mathbf{H}}_k^*\hat{\mathbf{W}}_k^*\mathbf{g}_k + 1/2\sigma^*\Delta_k^2 \\
&\geq 1/2\mathbf{g}_k^T(\hat{\mathbf{W}}_k^*)^T\hat{\mathbf{H}}_k^*\hat{\mathbf{W}}_k^*\mathbf{g}_k \\
&\geq 1/2\left(\frac{(\varepsilon_k^l)^2}{\hat{\lambda}_{\max}+\sigma^*}\right) \\
&\geq 1/2\left(\frac{(\varepsilon_k^l)^2}{\|\mathbf{B}_k\|_2 + \varepsilon_k^l\Delta_k^{-1}}\right).
\end{aligned}$$

Now consider two cases. Case 1 is defined by the condition $\|\mathbf{B}_k\|_2 > \varepsilon_k^l\Delta_k^{-1}$, which means that $Q(\mathbf{0}) - Q(\mathbf{s}_k) \geq 1/4 \cdot (\varepsilon_k^l)^2/\|\mathbf{B}_k\|_2$. Case 2 is defined when $\|\mathbf{B}_k\|_2 < \varepsilon_k^l\Delta_k^{-1}$, which means that $Q(\mathbf{0}) - Q(\mathbf{s}_k) \geq 1/4 \cdot \varepsilon_k^l\Delta_k$. Therefore we conclude that the sufficient decrease condition of (5.21), with the ℓ_2 constraint, holds as:

$$Q(\mathbf{0}) - Q(\mathbf{s}_k) \geq 1/4\varepsilon_k^l \cdot \min(\varepsilon_k^l/\|\mathbf{B}_k\|_2, \Delta_k),$$

where $\varepsilon_k^l = \|\mathbf{g}_k - \mathbf{A}^T\hat{\rho}^*\|_2$. ■

5.5.3 SUFFICIENT DECREASE WITH THE SHAPE-CHANGING CONSTRAINT

Note from Section 5.4.1 that the quadratic objective function has an equivalent representation in a transformed space, i.e. $Q(\mathbf{s}) = q(\mathbf{v}_2)$. This relation connects the sufficient decrease condition in the transformed space to the same condition in the original space.

Lemma 5.4. *The solution*

$$\mathbf{s}^* = \mathbf{P}\mathbf{v}^* = \mathbf{P}\begin{bmatrix} \mathbf{0} \\ \mathbf{v}_2^* \end{bmatrix},$$

in (5.19) *to the trust-region subproblem in* (5.3) *defined by the shape-changing norm* (5.15)*, satisfies the sufficient decrease condition* (5.21)*.*

Proof. Represent

$$\mathbf{v}_2^* = -\begin{bmatrix} \theta_{1,1} & & & \\ & \ddots & & \\ & & \theta_{2l,2l} & \\ & & & \beta\mathbf{I}_{n-(m+2l)} \end{bmatrix} \mathbf{g}_2 \equiv -\mathbf{\Theta}_2\mathbf{g}_2,$$

where

$$\theta_{i,i} = \begin{cases} \frac{1}{\lambda_i} & \text{if } \left|\frac{[\mathbf{g}_{\|2}]_i}{\lambda_i}\right| \leq \Delta_k \\ -\frac{\Delta_k[\mathbf{g}_{\|2}]_i}{|[\mathbf{g}_{\|2}]_i|} & \text{otherwise,} \end{cases}$$

and where β is as defined in (5.18). Observe that the component-wise inequality $\mathbf{\Theta}_2\mathbf{\Lambda}_2 \leq \mathbf{I}$ holds. Thus

$$\begin{aligned} Q(\mathbf{0}) - Q(\mathbf{s}^*) = -q(\mathbf{v}_2^*) &= -\left(\mathbf{g}_2^T\mathbf{v}_2^* + \frac{1}{2}(\mathbf{v}_2^*)^T\mathbf{\Lambda}_2\mathbf{v}_2^*\right) \\ &= -\left(-\mathbf{g}_2^T\mathbf{\Theta}_2\mathbf{g}_2 + \frac{1}{2}\mathbf{g}_2^T\mathbf{\Theta}_2\mathbf{\Lambda}_2\mathbf{\Theta}_2\mathbf{g}_2\right) \\ &\geq \frac{1}{2}\mathbf{g}_2^T\mathbf{\Theta}_2\mathbf{g}_2 \\ &\geq \frac{1}{2}\mathbf{g}_2^T\mathbf{g}_2\min\left(1/\lambda_{\max}, \Delta_k/\|\mathbf{g}_2\|_2\right) \\ &= \frac{1}{2}\|\mathbf{g}_2\|_2\min\left(\|\mathbf{g}_2\|_2/\|\mathbf{\Lambda}_2\|_2, \Delta_k\right). \end{aligned}$$

Moreover,

$$\|\mathbf{g}_2\|_2^2 = \mathbf{g}_k^T\mathbf{P}_2\mathbf{P}_2^T\mathbf{g}_k = \|(\mathbf{I} - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A})\mathbf{g}_k\|_2^2 \equiv \|\mathbf{g}_k - \mathbf{A}^T\rho_k^P\|_2^2,$$

where $\rho_k^P \equiv (\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A}\mathbf{g}_k$. Since $\mathbf{P}_2^T\mathbf{B}_k\mathbf{P}_2 = \mathbf{\Lambda}_2$ and therefore $\|\mathbf{B}_k\|_2 \geq \|\mathbf{\Lambda}_2\|_2$, the sufficient decrease condition from (5.21) holds as

$$\begin{aligned} Q(\mathbf{0}) - Q(\mathbf{s}^*) = -q(\mathbf{v}_2^*) &\geq 1/2\|\mathbf{g}_2\|_2\min\left(\|\mathbf{g}_2\|_2/\|\mathbf{\Lambda}_2\|_2, \Delta_k\right) \\ &\geq 1/2\varepsilon_k^P \cdot \min\left(\varepsilon_k^P/\|\mathbf{B}_k\|_2, \Delta_k\right), \end{aligned}$$

where $\varepsilon_k^P = \|\mathbf{g}_k - \mathbf{A}^T\rho_k^P\|_2$. ■

5.5.4 CONVERGENCE

The convergence of algorithms 5.1 and 5.2 will be established in a theorem, which invokes the theory developed by Conn et al. [CGT00]. To be consistent with [CGT00], our

result is based on the assumptions: A. The objective function $f(\mathbf{x})$ is twice continuously differentiable, it is bounded from below ($f(\mathbf{x}) \geq k_-$), and the Hessian is bounded from above ($\nabla^2 f(\mathbf{x}) \leq k_+$). B. The constraints are twice continuously differentiable, and they are consistent. C. A first order constraint qualification holds at a stationary point $\mathbf{x}^*$. D. The quadratic approximation $Q(\mathbf{s})$ is twice continuously differentiable, and E. The quasi-Newton matrix $\mathbf{B}_k$ is invertible for all k, i.e., the lowest eigenvalue $\lambda_{\min}$ is bounded from 0, and the largest eigenvalue $\lambda_{\max}$ is bounded from infinity. These properties are shown for the L-BFGS matrix in [BGYZ16]. Finally we note that

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

which relates the shape-changing norm to the ℓ_2-norm (cf. Section 2.5.4), and ensures a measure of 'closeness' to the ℓ_2-norm . We thus propose

Theorem 5.5. *Suppose that the eigenvalues of $\mathbf{B}_k$ are bounded, i.e., $0 < c_l \leq \hat{\lambda}_{min} \leq \hat{\lambda}_{max} < c_u$ for some constants c_l and c_u. Then every limit point of the sequence of iterates $\mathbf{x}_k, \mathbf{x}_{k+1}, \ldots$ generated by algorithm 5.1, or by algorithm 5.2, is first order critical.*

Proof. The algorithms proposed in this section have the same form as Algorithm 12.2.1 p. 452 [CGT00], which is included here for completeness. Note that the algorithm is reproduced almost literally, except for slight adaptations in order to be consistent with the problem formulation of this chapter.

ALGORITHM 5.3 (Algorithm 12.2.1 in [CGT00])

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

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

Step 2: Step calculation. Compute a step $\mathbf{s}_k$ that sufficiently reduces the model $Q(\mathbf{s})$ in the sense of (5.23) and (5.24), while $\mathbf{s}_k$ satisfies the constraints from Step 1;

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

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

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

Step 4: Trust-region radius update. Set

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

Increment k by 1 and go to Step 1.

Algorithm 12.2.1 converges to a first order critical point, as long as the steps $\mathbf{s}_k$ satisfy the sufficient-decrease condition

$$Q(\mathbf{0}) - Q(\mathbf{s}_k) \geq c\pi_k \min\left(\pi_k / \|\mathbf{B}_k\|_2, \Delta_k\right), \tag{5.23}$$

where $0 < c < 1$, and

$$\pi_k = \left| \begin{array}{c} \underset{\|\mathbf{s}\|_2 \leq 1}{\text{minimize}} \ \mathbf{g}_k^T \mathbf{s} \end{array} \right| \quad \text{subject to} \quad \mathbf{A}\mathbf{s} = \mathbf{0}. \tag{5.24}$$

Observe that by solving the minimization in (5.24) that π_k is expressed as

$$\pi_k = \left\| (\mathbf{I}_n - \mathbf{A}^T (\mathbf{A}\mathbf{A}^T)^{-1} \mathbf{A}) \mathbf{g}_k \right\|_2 = \varepsilon_k^P,$$

where ε_k^P is as specified in the proof of lemma 5.4. Therefore we conclude that algorithm 5.2 satisfies the sufficient decrease condition (5.23), and consequently converges to a critical point. In the ℓ_2-norm algorithm we have

$$\begin{aligned} \varepsilon_k^l &= \left\| \mathbf{g}_k - \mathbf{A}^T \hat{\rho}^* \right\|_2 \\ &= \left\| (\mathbf{I}_n - \mathbf{A}^T (\mathbf{A}(\mathbf{B}_k + \sigma^* \mathbf{I}_n)^{-1} \mathbf{A}^T)^{-1} (\mathbf{A}(\mathbf{B}_k + \sigma^* \mathbf{I}_n)^{-1}) \mathbf{g}_k \right\|_2 \\ &\geq \frac{\hat{\lambda}_{\min} + \sigma^*}{\hat{\lambda}_{\max} + \sigma^*} \left\| (\mathbf{I}_n - \mathbf{A}^T (\mathbf{A}\mathbf{A}^T)^{-1} \mathbf{A}) \mathbf{g}_k \right\|_2 \\ &= \frac{\hat{\lambda}_{\min} + \sigma^*}{\hat{\lambda}_{\max} + \sigma^*} \pi_k. \end{aligned}$$

By assumption there exist two positive constants c_l and c_u, so that $0 < c_l \leq \hat{\lambda}_{\min} \leq \hat{\lambda}_{\max} < c_u$ and thus

$$\varepsilon_k^l \geq c_l / c_u \pi_k,$$

where $c_l / c_u \geq 1$. Therefore we conclude that algorithm 5.1 also satisfies (5.23), and thus converges to a critical point. ■

5.6 NUMERICAL EXPERIMENTS

This section describes numerical experiments of comparing the two methods that we developed in this article, namely Algorithms 1 and 2, which we label as TR–ℓ_2 and TR–$(\mathbf{P}, \infty)$, respectively. We perform four sets of experiments. In Experiment I, we generated synthetic convex quadratic problems with linear equality constraints as test problems. In Experiment II, we considered problems from CUTEst with linear constraints. Among the selected linear problems we filter for the ones that have fewer constraints than unknowns. Even though many of the problems selected in this way include inequality and bound constraints, the test is are carried out as if all constraints were equality constraints. In Experiment III, we use 62 large-scale unconstrained CUTEst problems, and impose synthetically generated linear equality constraints on the unconstrained problems. The fourth part applies extensions of our methods in order to solve a nonlinearly constraint problem. Performance profiles (see [DM02]) are provided, when they yield additional insights. In particular, we compare the number of iterations (`iter`) (when the trust-region step is accepted) and the average time (`time`) for each solver on the test set of problems. The performance metric, ρ, with a given number of test problems, n_p, is

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

where $t_{p,s}$ is the "output" (i.e., time or iterations) 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. Throughout this section, the two proposed algorithms are regarded to have converged when two conditions are simultaneously satisfied:

$$\left\|\mathbf{g}_k - \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1}\mathbf{A}\mathbf{g}_k\right\|_2 \leq \epsilon_1 \max\left(1, \|\mathbf{x}_k\|_2\right) \quad \text{and} \quad \|\mathbf{A}\mathbf{x}_k - \mathbf{b}\|_2 \leq \epsilon_2.$$

Typically we set $\epsilon_1 = 1 \times 10^{-3}$ and $\epsilon_2 = 1 \times 10^{-5}$. Other parameters in the algorithm are $d_- = 1/4$, $d_+ = 2$ and $l = 5$. The implementations and tests are carried out in MATLAB.

5.6.1 EXPERIMENT I

The purpose of this experiment is to test the convergence properties of the algorithms, and to compare their performances as the problem dimension n varies. In particular, we randomly generate the problem data

$$\mathbf{Q} \in \mathbb{R}^{n \times n}, \quad \mathbf{g} \in \mathbb{R}^n, \quad \mathbf{A} \in \mathbb{R}^{m \times n}, \quad \mathbf{b} \in \mathbb{R}^m,$$

where the matrix $\mathbf{Q}$ is positive semidefinite, and where we define the objective functions as $f(\mathbf{x}) = \mathbf{g}^T\mathbf{x}+1/2\mathbf{x}^T\mathbf{Q}\mathbf{x}$. We set $m = 10$ and vary $n \in \{20, 50, 100, 1000, 5000, 7000, 10000\}$. The results of running the two methods are summarized in Fig. 5.1:

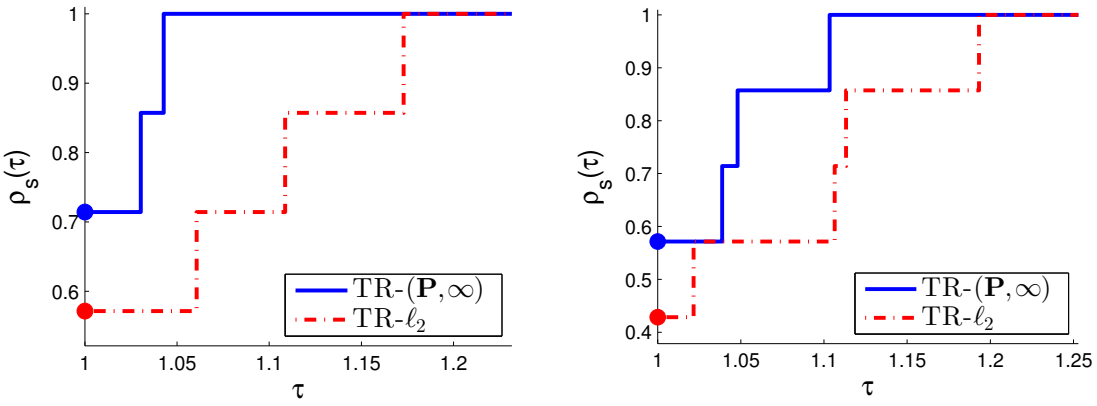


FIGURE 5.1 *Performance profiles comparing* `iter` *(left) and* `time` *(right) of applying TR–ℓ_2 and TR–$(\mathbf{P}, \infty)$ on convex quadratic problems with varying dimension sizes.*

We observe that both solvers converge on all test problems, and that TR–$(\mathbf{P}, \infty)$ performs well in terms of time and iterations.

5.6.2 EXPERIMENT II

The purpose of this experiment is to apply our algorithms to a set of standard test problems that are of the form (5.1). In this context, we filter the CUTEst library for problems with linear constraints, and dimensions of the form $1 \leq m \leq 200$ and $201 \leq n < \infty$. Among the problems that are selected, using the above search criteria, many include linear inequality constraints, or bounds on the variables. In our tests we treat all inequalities as equality constraints, and do not attempt to satisfy the bounds. The selected problems are

We report whether a algorithm converged on a particular problem (`conv`), the number of function evaluation it required (`fval`) and times (`time`).

We observe that for this set of problems, the number of function evaluations for both solvers exactly coincide. A reason for this is that the algorithms, on the problems without an error, stopped quickly within only two iterations. We reiterate that even though Table 5.2 indicates that an algorithm converged, this only means that the stopping criteria were met, when all constraints are treated as linear equality constraints. The computed solutions may be very different from the solutions to the original problems.

Problem	Equalities	Inequalities	Bounds
PRIMAL1	0	1	1
PRIMAL2	0	1	1
PRIMAL3	0	1	1
PRIMAL4	0	1	1
PRIMALC1	0	1	1
PRIMALC2	0	1	1
PRIMALC5	0	1	1
PRIMALC8	0	1	1
STATIC3	1	0	0
TABLE7	1	0	1
TABLE8	1	0	1

TABLE 5.1

CUTEst problems with linear constraints that satisfy $1 \leq m \leq 200$ *and* $201 \leq n < \infty$. *Here a '1' indicates that the particular constraint type is present in the problem. For example PRIMAL1 has no equality constraints, but it has inequality and bound constraints. Here* m *is the sum of the number of constraints from each type, i.e.,* $m = m_{Eq.} + m_{In.} + m_{Bo.}$

Problem	`conv` ℓ-2	`fval` ℓ-2	`time` ℓ-2	`conv` $(\mathbf{P}, \infty)$	`fval` $(\mathbf{P}, \infty)$	`time` $(\mathbf{P}, \infty)$
PRIMAL1	1	8	2.7×10^{-3}	1	8	3.0×10^{-3}
PRIMAL2	1	7	4.6×10^{-3}	1	7	4.8×10^{-3}
PRIMAL3	1	6	6.1×10^{-3}	1	6	6.3×10^{-3}
PRIMAL4	1	5	6.4×10^{-3}	1	5	7.9×10^{-3}
PRIMALC1	1	33	1.6×10^{-3}	1	33	1.6×10^{-3}
PRIMALC2	1	33	1.5×10^{-3}	1	33	1.5×10^{-3}
PRIMALC5	1	30	1.7×10^{-3}	1	30	1.8×10^{-3}
PRIMALC8	1	34	1.9×10^{-3}	1	34	2.0×10^{-3}
STATIC3	0	77	5.6×10^{-3}	0	77	5.9×10^{-3}
TABLE7	1	1	6.0×10^{-3}	1	1	8.3×10^{-3}
TABLE8	1	1	3.1×10^{-3}	1	1	2.4×10^{-3}

TABLE 5.2

CUTEst problems with linear constraints that satisfy $1 \leq m \leq 200$ *and* $201 \leq n < \infty$.

5.6.3 EXPERIMENT III

This experiment is to benchmark our algorithms on a set of large-scale problems. In this test we use 62 large-scale unconstrained CUTEst problems, and add randomly generated linear equality constraints to the objective functions. The number of equality constraints is fixed to $m = 10$. To set up the test problems, first we fix a seed to the random number generator using the command `rng(090317);`. Then we invoke the unconstrained CUTEst objective function. The linear constraints are generated using the command `A = randn(m,n)/norm(x0);` where `x0` is the initial vector formed by the initialization of the CUTEst problem. We provide a list of the CUTEst objective functions from this experiment in the Appendix.The results of running the two methods are summarized in Fig. 5.2:

We observe that TR–$(\mathbf{P}, \infty)$ performs well in terms of time, which may be contributed to the fact that this method computes a trust-region step using a formula.

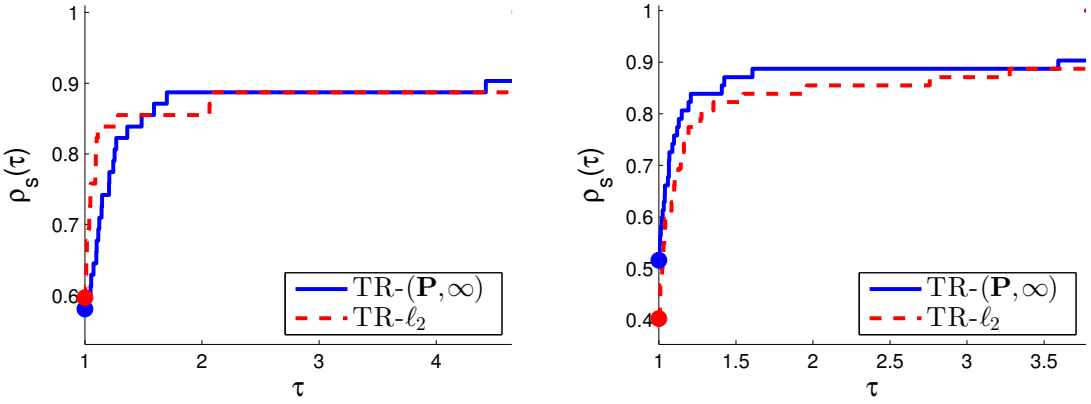


FIGURE 5.2 *Performance profiles comparing* `iter` *(left) and* `time` *(right) of applying TR–ℓ_2 and TR–$(\mathbf{P},\infty)$ on large-scale CUTEST problems with randomly added linear equality constraints.*

5.7 SUMMARY

In this chapter we develop two limited memory quasi-Newton trust-region methods, when linear equality constraints are present. The methods differ by the use of the norm that defines the trust-region subproblem. The advantage of the novel method based on the so-called shape-changing norm is that a search direction step can be computed using an analytic formula. Numerical experiments indeed indicate that the proposed method yields savings in computational times, when compared with the ℓ_2-norm trust-region solver implementation.

CHAPTER 6

OBLIQUE PROJECTION MATRICES

This chapter is based on the manuscript "*On the Eigendecomposition and Singular Values of Oblique Projection Matrices*", J. J. Brust, R. F. Marcia and C. G. Petra which is currently in preparation.

6.1 MOTIVATION

We present the eigendecomposition of oblique (non-symmetric) reduced-rank projection matrices, and develop an efficient algorithm to compute their singular values. The eigendecomposition can be used in computing pseudo-inverses in applications of oblique projection matrices, while the singular values define the spectral norm of these matrices. Oblique projection matrices arise in contexts such as systems of linear inequalities, constrained optimization and signal processing [CE02, BS94]. In previous research [Ste89, O'L90, FS01], bounds on the spectral norm of oblique projections are proposed. However, instead of computing bounds on the spectral norm of oblique projections, we compute the spectral norm directly based on an analysis of the form of the singular values.

6.2 REPRESENTATION OF OBLIQUE PROJECTIONS

The oblique projection matrix $\mathbf{W} \in \mathbb{R}^{n\times n}$ is defined by the properties

$$\mathbf{WW} = \mathbf{W} \quad \text{and} \quad \mathbf{W} \neq \mathbf{W}^T, \tag{6.1}$$

where $\text{Rank}(\mathbf{W}) = n - m$. Based on the first property, $\mathbf{W}$ itself spans the space of eigenvectors that have associated eigenvalues of repeated ones. Since the matrix is also of low-rank, the remaining eigenvalues are zeros. Thus any diagonalizable oblique

projection matrix can be represented as:

$$\mathbf{W} = \mathbf{I}_n - \mathbf{X}\mathbf{M}\mathbf{Y}^T, \tag{6.2}$$

where the matrices $\mathbf{X} \in \mathbb{R}^{n\times m}$, and $\mathbf{Y} \in \mathbb{R}^{n\times m}$ are full column rank matrices, and where $\mathbf{M} = (\mathbf{Y}^T\mathbf{X})^{-1}$. Because of the first property in (6.1), the matrix $\mathbf{X}\mathbf{M}\mathbf{Y}^T = \mathbf{I}_n - \mathbf{W}$ is an oblique projection matrix itself. We assume that $m \ll n$.

6.3 RELATED WORK

In Steward [Ste89] and O'Leary [O'L90] an analysis of the spectral norm of oblique projections is proposed. In particular, the matrices are defined by $\mathbf{I}_n - \mathbf{W}$ where $\mathbf{Y} = \mathbf{X}\mathbf{D}$, and where $\mathbf{D} \in \mathbb{R}^{n\times n}$ is a positive definite diagonal matrix. The main result of the two articles is that

$$\left\|\mathbf{X}(\mathbf{X}^T\mathbf{D}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{D}\right\|_2 \leq \left(\min_I \inf_+ (\mathbf{U}_I)\right)^{-1},$$

where $\inf_+ (\mathbf{U}_I)$ symbolizes the smallest non-zero singular value of any submatrix $\mathbf{U}_I$ of an orthonormal basis $\mathbf{U} \in \mathbb{R}^{n\times m}$ to $\mathbf{X}$. In this chapter, we analyze the eigendecomposition of oblique projections by expressing them in the form $\mathbf{W} = \mathbf{I}_n - \mathbf{X}(\mathbf{X}^T\mathbf{Y})^{-1}\mathbf{Y}^T$, and explicitly compute their singular values. As a corollary, the spectral norm is then inferred as the largest singular value.

6.4 EIGENDECOMPOSITION

Begin with the observation $(\mathbf{X}\mathbf{M}\mathbf{Y}^T)\mathbf{X} = \mathbf{X}$, so that

$$\mathbf{W}\mathbf{X} = (\mathbf{I}_n - \mathbf{X}\mathbf{M}\mathbf{Y}^T)\mathbf{X} = \mathbf{0}.$$

Morover, since $\mathbf{X}$ is of full column rank this also means that the columns of $\mathbf{X}$ span the nullspace of $\mathbf{W}$. Denote an orthonormal basis of Range($\mathbf{X}$) by $\mathbf{Q} \in \mathbb{R}^{n\times m}$. Then define the orthogonal complement of $\mathbf{Q}$ by $\mathbf{Q}_\perp \in \mathbb{R}^{n\times(n-m)}$, so that

$$\mathbf{I}_n = [\ \mathbf{Q}\ \ \mathbf{Q}_\perp\ ][\ \mathbf{Q}\ \ \mathbf{Q}_\perp\ ]^T = [\ \mathbf{Q}\ \ \mathbf{Q}_\perp\ ]^T[\ \mathbf{Q}\ \ \mathbf{Q}_\perp\ ]. \tag{6.3}$$

Then observe that $\mathbf{X}\mathbf{Q}_\perp = \mathbf{0}_{n\times(n-m)}$, and therefore

$$\mathbf{W}(\mathbf{W}\mathbf{Q}_\perp) = \mathbf{W}\mathbf{Q}_\perp.$$

This implies that, since $\mathbf{Q}_\perp$ is the orthogonal complement of the nullspace of $\mathbf{W}$, the matrix

$$\mathbf{W}\mathbf{Q}_\perp = \mathbf{Q}_\perp - \mathbf{X}\mathbf{M}\mathbf{Y}^T\mathbf{Q}_\perp \equiv \mathbf{Z} \tag{6.4}$$

corresponds to the eigenvectors with eigenvalue 1. Using the columns of $\mathbf{Z}$ we form the matrix $\mathbf{S} = [\ \mathbf{XM}\ \ \mathbf{Z}\ ] \in \mathbb{R}^{n\times n}$, which allows us to eigendecompose $\mathbf{W}$. In particular, $\mathbf{S}$ and its inverse $\mathbf{S}^{-1}$ are key in the diagonalization of $\mathbf{W}$. The inverse of $\mathbf{S}$ is described next.

Lemma 6.1. *The square matrix*

$$\mathbf{S} = [\ \mathbf{XM}\ \ \mathbf{Z}\ ], \tag{6.5}$$

has the explicit inverse

$$\mathbf{S}^{-1} = \begin{bmatrix} \mathbf{Y}^T \\ \mathbf{Q}_\perp^T \end{bmatrix}. \tag{6.6}$$

Proof. The proof is based on showing that $\mathbf{S}^{-1}$ is the right, and the left inverse of $\mathbf{S}$, respectively. First observe from (6.3) that

$$\mathbf{Q}_\perp\mathbf{Q}_\perp^T = \mathbf{I}_n - \mathbf{Q}\mathbf{Q}^T,$$

and thus $\mathbf{W}(\mathbf{Q}_\perp\mathbf{Q}_\perp^T) = \mathbf{W}(\mathbf{I}_n - \mathbf{Q}\mathbf{Q}^T) = \mathbf{W}$. Thus, for the right inverse

$$\begin{aligned}
\mathbf{S}\mathbf{S}^{-1} &= [\ \mathbf{XM}\ \ \mathbf{Z}\ ] \begin{bmatrix} \mathbf{Y}^T \\ \mathbf{Q}_\perp^T \end{bmatrix} \\
&= \mathbf{X}\mathbf{M}\mathbf{Y}^T + \mathbf{Z}\mathbf{Q}_\perp^T \\
&= \mathbf{X}\mathbf{M}\mathbf{Y}^T + (\mathbf{Q}_\perp - \mathbf{X}\mathbf{M}\mathbf{Y}^T\mathbf{Q}_\perp)\mathbf{Q}_\perp^T \\
&= \mathbf{X}\mathbf{M}\mathbf{Y}^T + (\mathbf{I}_n - \mathbf{X}\mathbf{M}\mathbf{Y}^T)(\mathbf{I}_n - \mathbf{Q}\mathbf{Q}^T) \\
&= \mathbf{X}\mathbf{M}\mathbf{Y}^T + (\mathbf{I}_n - \mathbf{X}\mathbf{M}\mathbf{Y}^T) \\
&= \mathbf{I}_n.
\end{aligned}$$

For the left inverse, since $\mathbf{Q}_\perp^T\mathbf{X} = \mathbf{0}$, then $\mathbf{Q}_\perp^T\mathbf{W} = \mathbf{Q}_\perp^T$. Moreover, from (6.1) $\mathbf{Y}^T\mathbf{W} =$

$\mathbf{0}$, so that

$$\begin{aligned}\mathbf{S}^{-1}\mathbf{S} &= \begin{bmatrix} \mathbf{Y}^T \\ \mathbf{Q}_\perp^T \end{bmatrix} \begin{bmatrix} \mathbf{XM} & \mathbf{Z} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{Y}^T\mathbf{XM} & \mathbf{Y}^T\mathbf{Z} \\ \mathbf{Q}_\perp^T\mathbf{XM} & \mathbf{Q}_\perp^T\mathbf{Z} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{I}_m & \mathbf{Y}^T\mathbf{W}\mathbf{Q}_\perp \\ \mathbf{Q}_\perp^T\mathbf{XM} & \mathbf{Q}_\perp^T\mathbf{Q}_\perp \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{I}_m & \\ & \mathbf{I}_{n-m} \end{bmatrix}.\end{aligned}$$

■

Based on Lemma 6.1, we propose the eigendecomposition of $\mathbf{W}$ in the next theorem.

Theorem 6.2. *The oblique projection matrix* $\mathbf{W}$ *from* (6.2)*, is eigendecomposed as*

$$\mathbf{W} = \mathbf{S}\mathbf{\Lambda}\mathbf{S}^{-1}, \tag{6.7}$$

where the left eigenvectors are the columns of $\mathbf{S}^{-1}$ *from* (6.6) *and the right eigenvectors are the columns of* $\mathbf{S}$ *from* (6.5)*, respectively. The diagonal matrix* $\mathbf{\Lambda} \in \mathbb{R}^{n\times n}$ *of eigenvalues is given by*

$$\mathbf{\Lambda} = \begin{bmatrix} \mathbf{0}_m & \\ & \mathbf{I}_{n-m} \end{bmatrix}.$$

Proof. Observe the fundamental identity

$$\begin{aligned}\mathbf{S}\mathbf{\Lambda}\mathbf{S}^{-1} &= \begin{bmatrix} \mathbf{XM} & \mathbf{Z} \end{bmatrix} \begin{bmatrix} \mathbf{0}_m & \\ & \mathbf{I}_{n-m} \end{bmatrix} \begin{bmatrix} \mathbf{Y}^T \\ \mathbf{Q}_\perp^T \end{bmatrix} \\ &= \mathbf{Z}\mathbf{Q}_\perp^T \\ &= (\mathbf{I}_n - \mathbf{XMY}^T)\mathbf{Q}_\perp\mathbf{Q}_\perp^T \\ &= (\mathbf{I}_n - \mathbf{XMY}^T)(\mathbf{I}_n - \mathbf{QQ}^T) \\ &= (\mathbf{I}_n - \mathbf{XMY}^T) \\ &= \mathbf{W}.\end{aligned}$$

■

6.5 SINGULAR VALUES

To compute the singular values of $\mathbf{W}$ we compute the eigenvalues of $\mathbf{WW}^T$. Specifically, we proceed by applying three similarity transformations, which yield

Theorem 6.3. *The singular values of the oblique projection matrix* $\mathbf{W}$ *from* (6.2) *are the diagonal elements of the matrix*

$$\mathbf{\Theta} = \begin{bmatrix} \mathbf{I}_m + \mathbf{\Sigma}^T\mathbf{\Sigma} & & \\ & \mathbf{I}_{n-2m} & \\ & & \mathbf{0}_{m\times m} \end{bmatrix}. \tag{6.8}$$

Proof. The singular values of $\mathbf{W}$ are equivalent to the eigenvalues of $\mathbf{W}\mathbf{W}^T$. Therefore we apply similarity transformations to $\mathbf{W}\mathbf{W}^T$, which reduce the matrix to a triangular form. First let $\mathbf{P} = [\ \mathbf{Q}_\perp\ \ \mathbf{Q}\ ]$ so that

$$\mathbf{P}^T\mathbf{W}\mathbf{W}^T\mathbf{P} = \begin{bmatrix} \mathbf{I}_{n-m} & \mathbf{Q}_\perp^T\mathbf{W}^T\mathbf{Q} \\ \mathbf{Q}^T\mathbf{W}\mathbf{Q}_\perp & \mathbf{Q}^T\mathbf{W}\mathbf{W}^T\mathbf{Q} \end{bmatrix}.$$

Then let upper triangular matrices be given by

$$\mathbf{R} = \begin{bmatrix} \mathbf{I}_{n-m} & -\mathbf{Q}_\perp^T\mathbf{W}^T\mathbf{Q} \\ & \mathbf{I}_m \end{bmatrix},$$

and

$$\mathbf{R}^{-1} = \begin{bmatrix} \mathbf{I}_{n-m} & \mathbf{Q}_\perp^T\mathbf{W}^T\mathbf{Q} \\ & \mathbf{I}_m \end{bmatrix}.$$

The second similarity transformation is computed as

$$\begin{aligned} \mathbf{R}^{-1}\mathbf{P}^T\mathbf{W}\mathbf{W}^T\mathbf{P}\mathbf{R} &= \begin{bmatrix} \mathbf{I} & \mathbf{Q}_\perp^T\mathbf{W}^T\mathbf{Q} \\ & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{Q}_\perp^T\mathbf{W}^T\mathbf{Q} \\ \mathbf{Q}^T\mathbf{W}\mathbf{Q}_\perp & \mathbf{Q}^T\mathbf{W}\mathbf{W}^T\mathbf{Q} \end{bmatrix} \begin{bmatrix} \mathbf{I} & -\mathbf{Q}_\perp^T\mathbf{W}^T\mathbf{Q} \\ & \mathbf{I} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{I}_{n-m} & \mathbf{Q}_\perp^T\mathbf{W}^T\mathbf{Q} \\ & \mathbf{I}_m \end{bmatrix} \begin{bmatrix} \mathbf{I}_{n-m} & \mathbf{0}_{(n-m)\times m} \\ \mathbf{Q}^T\mathbf{W}\mathbf{Q}_\perp & \mathbf{0}_{m\times m} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{I}_{n-m} + (\mathbf{Q}_\perp^T\mathbf{W}^T\mathbf{Q})(\mathbf{Q}_\perp^T\mathbf{W}^T\mathbf{Q})^T & \mathbf{0}_{(n-m)\times m} \\ \mathbf{Q}^T\mathbf{W}\mathbf{Q}_\perp & \mathbf{0}_{m\times m} \end{bmatrix}. \end{aligned}$$

Therefore m eigenvalues of $\mathbf{W}\mathbf{W}^T$ are 0, while the remaining eigenvalues are the eigenvalues of the matrix $\mathbf{I}_{n-m} + (\mathbf{Q}_\perp^T\mathbf{W}^T\mathbf{Q})(\mathbf{Q}_\perp^T\mathbf{W}^T\mathbf{Q})^T \equiv \mathbf{I} + \mathbf{\Psi}\mathbf{\Psi}^T$, where $\mathbf{\Psi} \equiv \mathbf{Q}_\perp^T\mathbf{W}^T\mathbf{Q}$. A key observation is that the dimensions of $\mathbf{\Psi}$ are $(n-m)\times m$. Since $m \ll n$, then $m \ll (n-m)$. Denote the SVD of $\mathbf{\Psi}$ by

$$\mathbf{\Psi} = \mathbf{U}\Sigma\mathbf{V}^T,$$

where $\mathbf{U} \in \mathbb{R}^{(n-m)\times(n-m)}$, $\mathbf{\Sigma} \in \mathbb{R}^{(n-m)\times m}$ with $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_m \geq 0$, and $\mathbf{V} \in \mathbb{R}^{m\times m}$. Then $\mathbf{I} + \mathbf{\Psi}\mathbf{\Psi}^T = \mathbf{U}(\mathbf{I} + \Sigma\Sigma^T)\mathbf{U}^T$. Applying the similarity transformation

$\mathbf{K} = \begin{bmatrix} \mathbf{U} \\ \mathbf{I}_m \end{bmatrix}$, yields the lower triangular matrix

$$\begin{aligned}
\mathbf{K}^T\mathbf{R}^{-1}\mathbf{P}^T\mathbf{W}\mathbf{W}^T\mathbf{P}\mathbf{R}\mathbf{K} &= \begin{bmatrix} \mathbf{U} & \\ & \mathbf{I} \end{bmatrix}^T \begin{bmatrix} \mathbf{I} + (\mathbf{Q}_\perp^T\mathbf{W}^T\mathbf{Q})(\mathbf{Q}_\perp^T\mathbf{W}^T\mathbf{Q})^T & \mathbf{0} \\ \mathbf{Q}^T\mathbf{W}\mathbf{Q}_\perp & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{U} & \\ & \mathbf{I} \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{U}^T(\mathbf{I}_{n-m} + \mathbf{\Psi}\mathbf{\Psi}^T)\mathbf{U} & \mathbf{0} \\ \mathbf{Q}\mathbf{W}^T\mathbf{Q}_\perp^T & \mathbf{0} \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{I}_{n-m} + \mathbf{\Sigma}\mathbf{\Sigma}^T & \mathbf{0}_{(n-m)\times m} \\ \mathbf{Q}\mathbf{W}^T\mathbf{Q}_\perp^T & \mathbf{0}_{m\times m} \end{bmatrix}. \qquad (6.9)
\end{aligned}$$

Therefore the non-zero eigenvalues of $\mathbf{W}\mathbf{W}^T$ are the non-zeros elements of the diagonal matrix $\mathbf{I}_{n-m}+\mathbf{\Sigma}\mathbf{\Sigma}^T$, and the matrix $\mathbf{\Theta}$ from (6.8) combines all eigenvalues of $\mathbf{W}\mathbf{W}^T$. ■

6.6 ALGORITHM

The non-zero singular values σ_i can be efficiently computed as the eigenvalues of the small $(m \times m)$ matrix $\mathbf{\Psi}^T\mathbf{\Psi}$, i.e.,

$$\mathbf{\Psi}^T\mathbf{\Psi} = (\mathbf{Q}_\perp\mathbf{W}^T\mathbf{Q})^T\mathbf{Q}_\perp\mathbf{W}^T\mathbf{Q} = \mathbf{Q}^T\mathbf{W}\mathbf{W}^T\mathbf{Q} = \mathbf{V}\Sigma^T\Sigma\mathbf{V}^T.$$

The algorithm to compute the eigenvalues of $\mathbf{W}\mathbf{W}^T$ is:

ALGORITHM 6.1:

1. Compute $\mathbf{Q}$, the orthonormal basis of $\mathbf{X}$;
2. Compute the eigenvalues of $\left(\mathbf{Q}^T\mathbf{W}\mathbf{W}^T\mathbf{Q}\right) = \mathbf{\Sigma}^2$ with $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_m$;
3. Form
$$\text{eig}\left(\mathbf{W}\mathbf{W}^T\right) = \begin{bmatrix} \mathbf{I}_m + \mathbf{\Sigma}^2 & & \\ & \mathbf{I}_{n-2m} & \\ & & \mathbf{0}_{m\times m} \end{bmatrix};$$

Corollary 6.4. *The spectral norm is computed by*

$$\|\mathbf{W}\|_2 = \sqrt{1+\sigma_1^2}.$$

6.7 NUMERICAL EXPERIMENTS

We demonstrate the effectiveness of Algorithm 6.1 when the oblique projections in (6.2) become large. In particular, we randomly generate the matrices $\mathbf{X} \in \mathbb{R}^{n\times m}$ and $\mathbf{Y} \in \mathbb{R}^{n\times m}$, with $\mathbf{M} = (\mathbf{Y}^T\mathbf{X})^{-1}$. Throughout the experiments m is fixed to the

value of $m = 4$ while n varies between $10^1 \leq n \leq 10^6$. For comparison we compute the singular values of (6.2) also using a standard build-in MATLAB function. For dimensions of $n > 5,000$, MATLAB exceeded our computational time budget of 30 seconds, which is why we omit to report its results beyond $n = 5,000$. To compare the accuracy of Algorithm 6.1 with the build-in function, we define the quantity $\texttt{error} \equiv \sqrt{\sum_{i=1}^{n}((\sigma_i^2)^{\text{ALG.6.1}} - (\sigma_i^2)^{\text{MTLB.}})^2}/n$. Here $(\sigma_i^2)^{\text{ALG.6.1}}$ and $(\sigma_i^2)^{\text{MTLB.}}$ represent singular values, computed by the two respective methods. We also display the scaled spectral norm of $\|\mathbf{W}\|_2/n$ and report the time (in seconds) to compute all singular values with the two methods.

n	$\|\mathbf{W}\|_2/n$ (ALG.6.1)	$\|\mathbf{W}\|_2/n$ (MTLB.)	`time` (ALG.6.1)	`time` (MTLB.)	`error`
10	0.5866	0.5866	0.0003	0.0001	1.0×10^{-15}
100	0.9168	0.9168	0.0001	0.0026	9.0×10^{-14}
1000	0.4712	0.4712	0.0001	0.3036	5.3×10^{-9}
3000	0.6829	0.6829	0.0002	6.6539	1.3×10^{-11}
5000	0.6845	0.6845	0.0002	28.6290	2.6×10^{-11}
10000	0.1938	N/A	0.0003	N/A	N/A
100000	0.5508	N/A	0.0044	N/A	N/A
1000000	0.6409	N/A	0.0382	N/A	N/A

TABLE 6.1

Comparison of Algorithm 6.1 with the build-in MATLAB function `eig` *to compute the singular values of oblique projection matrices* (6.2)*. The build-in function is only used to compute singular up to* $n = 5,000$*, because beyond this value it becomes exceedingly slow.*

6.8 SUMMARY

In this chapter we describe the eigendecomposition of oblique projection matrices, which may be used in the computation of pseudo-inverses. We derive expressions for the singular values of oblique projection matrices, which enabled us to develop an effective algorithm to compute the singular values of large-scale oblique projections. The proposed method may be potentially used to assess the reliability of computations with large-scale oblique projection matrices, because it can be used to compute the spectral norm efficiently.

CHAPTER 7

SUMMARY

In this dissertation, I have focused on the development of novel methods for large-scale quasi-Newton trust-region methods. A significant component of the dissertation is in the realm of applying and inventing approaches from numerical linear algebra. For large-scale quasi-Newton trust-region subproblems, two high-accuracy solvers are proposed; the OBS method, and the SC-SR1 method, which use partial eigendecompositions of L-SR1 quasi-Newton compact factorizations. In Chapter 3 we develop a trust-region method for large-scale unconstrained minimization. The novelty introduced by this method is that instead of a standard multiple of identity initial quasi-Newton matrix a more sophisticated dense initial matrix is used. Chapter 4 proposes a trust-region method when the less known indefinite limited-memory multipoint symmetric secant (L-MSS) matrix approximates the Hessian. Based on L-MSS matrices two approaches to solve trust-region subproblems are developed. One approach is based on an partial eigendecomposition of the quasi-Newton matrix, while the other approach exploits properties of MSS matrices to derive a formula for the ℓ_2-norm trust-region subproblem solution. The final two chapters propose methods in the context of large-scale equality constrained optimization. Specifically, we develop a matrix factorization of the (1,1) block of the inverse Karush-Kuhn-Tucker matrix, which in combination with non standard norms, yields analytic solutions of linear equality constrained trust-region subproblems. In addition, we find the eigendecomposition of oblique projection matrices and develop an algorithm to efficiently compute the singular values of these matrices.

Overall, I envision my future efforts to focus on the development of novel mathematical methods that are available in the form of software tools. I am highly motivated to continue any established collaborations, and will actively seek new opportunities in order to pursue my goals.

Appendix A

THE RECURSIVE MSSM UPDATE FORMULA

This appendix spells out the details of deriving the recursive update formula in (4.7). Here we define $\bar{\mathbf{s}} = \mathbf{s}_k - \mathbf{s}_0$ and $\bar{\mathbf{y}} = \mathbf{y}_k - \mathbf{y}_0$, so that $\mathbf{S}_{k+1}$ and $\mathbf{Y}_{k+1}$ are written as

$$\mathbf{S}_{k+1} = \left(\mathbf{S}_k + \bar{\mathbf{s}}\mathbf{e}_n^T\right)\bar{\mathbf{P}} \quad \text{and} \quad \mathbf{Y}_{k+1} = \left(\mathbf{Y}_k + \bar{\mathbf{y}}\mathbf{e}_n^T\right)\bar{\mathbf{P}}.$$

The product $\mathbf{S}_{k+1}^T\mathbf{Y}_{k+1}$ is computed as

$$\mathbf{S}_{k+1}^T\mathbf{Y}_{k+1} = \bar{\mathbf{P}}^T\left(\mathbf{S}_k^T\mathbf{Y}_k + \mathbf{S}_k\bar{\mathbf{y}}\mathbf{e}_n^T + \mathbf{e}_n\bar{\mathbf{s}}^T\mathbf{Y}_k + \mathbf{e}_n\bar{\mathbf{s}}^T\bar{\mathbf{y}}\mathbf{e}_n^T\right)\bar{\mathbf{P}} \equiv \bar{\mathbf{P}}^T\boldsymbol{\Theta}\bar{\mathbf{P}},$$

where we define $\boldsymbol{\Theta} \equiv \mathbf{S}_k^T\mathbf{Y}_k + \mathbf{S}_k\bar{\mathbf{y}}\mathbf{e}_n^T + \mathbf{e}_n\bar{\mathbf{s}}^T\mathbf{Y}_k + \mathbf{e}_n\bar{\mathbf{s}}^T\bar{\mathbf{y}}\mathbf{e}_n^T$ in order to simplify the notation. The symmetrization transformation has a special property when it is applied to a matrix that is permuted by $\bar{\mathbf{P}}^T$ and $\bar{\mathbf{P}}$. In [Bur83] eq. (2.4) it is established that for any square matrix $\mathbf{B} \in \mathbb{R}^{n\times n}$

$$\text{sym}\left(\bar{\mathbf{P}}^T\mathbf{B}\bar{\mathbf{P}}\right) = \bar{\mathbf{P}}^T\left(\text{sym}(\mathbf{B}) + \left(\mathbf{B} - \mathbf{B}^T\right)\mathbf{e}_n\mathbf{e}_n^T + \mathbf{e}_n\mathbf{e}_n^T\left(\mathbf{B}^T - \mathbf{B}\right)\right)\bar{\mathbf{P}}.$$

In the same reference, it is also noted that the symmetrization transformation is a linear operation in terms of its arguments, i.e., $\text{sym}\left(\mathbf{B} + \mathbf{C}\right) = \text{sym}\left(\mathbf{B}\right) + \text{sym}\left(\mathbf{C}\right)$ for

any square matrices $\mathbf{B}, \mathbf{C} \in \mathbb{R}^{n \times n}$. Therefore

$$\begin{aligned}
\boldsymbol{\Gamma}_{k+1} &= \operatorname{sym}\left(\mathbf{S}_{k+1}^T \mathbf{Y}_{k+1}\right) \\
&= \operatorname{sym}\left(\bar{\mathbf{P}}^T \boldsymbol{\Theta} \bar{\mathbf{P}}\right) \\
&= \bar{\mathbf{P}}^T \left(\operatorname{sym}\left(\boldsymbol{\Theta}\right) + \left(\boldsymbol{\Theta} - \boldsymbol{\Theta}^T\right) \mathbf{e}_n \mathbf{e}_n^T + \mathbf{e}_n \mathbf{e}_n^T \left(\boldsymbol{\Theta}^T - \boldsymbol{\Theta}\right)\right) \bar{\mathbf{P}} \\
&= \bar{\mathbf{P}}^T \Bigg(\operatorname{sym}\left(\mathbf{S}_k^T \mathbf{Y}_k\right) + \left(\mathbf{S}_k^T \mathbf{y}_k - \mathbf{Y}_k^T \mathbf{s}_0\right) \mathbf{e}_n^T \\
&\qquad + \mathbf{e}_n \left(\mathbf{y}_k^T \mathbf{S}_k - \mathbf{s}_0^T \mathbf{Y}_k\right) + \bar{\mathbf{s}}^T \bar{\mathbf{y}} \mathbf{e}_n \mathbf{e}_n^T \Bigg) \bar{\mathbf{P}}.
\end{aligned}$$

Since $\mathbf{S}_{k+1}$ is assumed to be a square invertible matrix, its inverse can be computed by the Sherman-Morrison-Woodbury formula

$$\begin{aligned}
\mathbf{S}_{k+1}^{-1} &= \bar{\mathbf{P}}^T \left(\mathbf{S}_k^{-1} + \frac{1}{\mathbf{s}_k^T \mathbf{S}_k^{-T} \mathbf{e}_n} \left(\mathbf{e}_n - \mathbf{S}_k^{-1} \mathbf{s}_k\right) \left(\mathbf{S}_k^{-T} \mathbf{e}_n\right)^T \right) \\
&\equiv \bar{\mathbf{P}}^T \left(\mathbf{S}_k^{-1} + \alpha \left(\mathbf{e}_n - \mathbf{S}_k^{-1} \mathbf{s}_k\right) \mathbf{c}_k^T\right),
\end{aligned}$$

where $\mathbf{c}_k \equiv \mathbf{S}_k^{-T} \mathbf{e}_n$ and $\alpha \equiv 1 / \mathbf{s}_k^T \mathbf{c}_k$. Our goal now is to separate the expression

$$\begin{aligned}
\mathbf{B}_{k+1} &= \mathbf{S}_{k+1}^{-T} \boldsymbol{\Gamma}_{k+1} \mathbf{S}_{k+1}^{-1} \\
&= \mathbf{S}_{k+1}^{-T} \Bigg(\bar{\mathbf{P}}^T \left(\operatorname{sym}\left(\mathbf{S}_k^T \mathbf{Y}_k\right) + \left(\mathbf{S}_k^T \mathbf{y}_k - \mathbf{Y}_k^T \mathbf{s}_0\right) \mathbf{e}_n^T \right. \\
&\qquad \left. + \mathbf{e}_n \left(\mathbf{y}_k^T \mathbf{S}_k - \mathbf{s}_0^T \mathbf{Y}_k\right) + \bar{\mathbf{s}}^T \bar{\mathbf{y}} \mathbf{e}_n \mathbf{e}_n^T \right) \bar{\mathbf{P}} \Bigg) \mathbf{S}_{k+1}^{-1},
\end{aligned}$$

into simpler components in order to reveal the recursive relation. Therefore we start with the term $\mathbf{S}_{k+1}^{-T} \bar{\mathbf{P}}^T \operatorname{sym}\left(\mathbf{S}_k^T \mathbf{Y}_k\right) \bar{\mathbf{P}} \mathbf{S}_{k+1}^{-1}$. Since $\operatorname{sym}\left(\mathbf{S}_k^T \mathbf{Y}_k\right) = \boldsymbol{\Gamma}_k$ and $\boldsymbol{\Gamma}_k \mathbf{e}_n = \left(\mathbf{L}_k + \mathbf{E}_k + \mathbf{L}_k^T\right) \mathbf{e}_n = \mathbf{Y}_k^T \mathbf{s}_0$, then

$$\begin{aligned}
\mathbf{S}_{k+1}^{-T} \bar{\mathbf{P}}^T \operatorname{sym}\left(\mathbf{S}_k^T \mathbf{Y}_k\right) \bar{\mathbf{P}} \mathbf{S}_{k+1}^{-1} = \mathbf{B}_k + \alpha \Bigg(& \left(\mathbf{S}_k^{-T} \mathbf{Y}_k^T \mathbf{s}_0 - \mathbf{B}_k \mathbf{s}_k\right) \mathbf{c}_k^T \\
& + \mathbf{c}_k \left(\mathbf{S}_k^{-T} \mathbf{Y}_k^T \mathbf{s}_0 - \mathbf{B}_k \mathbf{s}_k\right)^T \Bigg) \\
& + \alpha^2 \mathbf{c}_k \left(\mathbf{e}_n - \mathbf{S}_k^{-1} \mathbf{s}_k\right)^T \boldsymbol{\Gamma}_k \left(\mathbf{e}_n - \mathbf{S}_k^{-1} \mathbf{s}_k\right) \mathbf{c}_k^T.
\end{aligned}$$

Next we note that

$$\begin{aligned}
\mathbf{S}_{k+1}^{-T} \bar{\mathbf{P}}^T \left(\left(\mathbf{S}_k^T \mathbf{y}_k - \mathbf{Y}_k^T \mathbf{s}_0\right) \mathbf{e}_n^T\right) \bar{\mathbf{P}} \mathbf{S}_{k+1}^{-1} = \alpha \Bigg(& \mathbf{y}_k + \mathbf{S}_k^{-T} \mathbf{Y}_k^T \mathbf{s}_0 \\
& + \alpha \mathbf{c}_k \left(\mathbf{e}_n - \mathbf{S}_k^{-1} \mathbf{s}_k\right)^T \left(\mathbf{S}_k^T \mathbf{y}_k - \mathbf{Y}_k^T \mathbf{s}_0\right) \Bigg) \mathbf{c}_k^T,
\end{aligned}$$

and observe that

$$\bar{\mathbf{s}}^T\bar{\mathbf{y}}\mathbf{S}_{k+1}^{-T}\bar{\mathbf{P}}^T\mathbf{e}_n\mathbf{e}_n^T\bar{\mathbf{P}}\mathbf{S}_{k+1}^{-1} = \alpha^2\bar{\mathbf{s}}^T\bar{\mathbf{y}}\mathbf{c}_k\mathbf{c}_k^T.$$

Now, combining the previous expressions results in

$$\begin{aligned}
\mathbf{B}_{k+1} &= \mathbf{S}_{k+1}^{-T}\mathbf{\Gamma}_{k+1}\mathbf{S}_{k+1}^{-1} \\
&= \mathbf{B}_k + \alpha\left(\mathbf{S}_k^{-T}\mathbf{Y}_k^T\mathbf{s}_0 - \mathbf{B}_k\mathbf{s}_k\right)\mathbf{c}_k^T + \alpha\mathbf{c}_k\left(\mathbf{S}_k^{-T}\mathbf{Y}_k^T\mathbf{s}_0 - \mathbf{B}_k\mathbf{s}_k\right)^T \\
&\quad + \alpha\left(\mathbf{y}_k + \mathbf{S}_k^{-T}\mathbf{Y}_k^T\mathbf{s}_0 + \alpha\mathbf{c}_k\left(\mathbf{e}_n - \mathbf{S}_k^{-1}\mathbf{s}_k\right)^T\left(\mathbf{S}_k^T\mathbf{y}_k - \mathbf{Y}_k\mathbf{s}_0\right)\right)\mathbf{c}_k^T \\
&\quad + \alpha^2\mathbf{c}_k\Big(\bar{\mathbf{s}}^T\bar{\mathbf{y}} + 2\left(\mathbf{e}_n - \mathbf{S}_k^{-1}\mathbf{s}_k\right)^T\left(\mathbf{S}_k^T\mathbf{y}_k - \mathbf{Y}_k\mathbf{s}_0\right) \\
&\qquad + \left(\mathbf{e}_n - \mathbf{S}_k^{-1}\mathbf{s}_k\right)^T\mathbf{\Gamma}_k\left(\mathbf{e}_n - \mathbf{S}_k^{-1}\mathbf{s}_k\right)\Big)\mathbf{c}_k^T \\
&= \mathbf{B}_k + \alpha\left(\left(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k\right)\mathbf{c}_k^T + \mathbf{c}_k\left(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k\right)^T\right) \\
&\quad + \alpha^2\mathbf{c}_k\Big(\bar{\mathbf{s}}^T\bar{\mathbf{y}} + 2\left(\mathbf{e}_n - \mathbf{S}_k^{-1}\mathbf{s}_k\right)^T\left(\mathbf{S}_k^T\mathbf{y}_k - \mathbf{Y}_k\mathbf{s}_0\right) \\
&\qquad + \left(\mathbf{e}_n - \mathbf{S}_k^{-1}\mathbf{s}_k\right)^T\mathbf{\Gamma}_k\left(\mathbf{e}_n - \mathbf{S}_k^{-1}\mathbf{s}_k\right)\Big)\mathbf{c}_k^T \\
&= \mathbf{B}_k + \alpha\left(\left(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k\right)\mathbf{c}_k^T + \mathbf{c}_k\left(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k\right)^T\right) - \alpha^2\mathbf{s}_k^T(\mathbf{y}_k - \mathbf{B}_k\mathbf{s}_k)\mathbf{c}_k\mathbf{c}_k^T. \quad \text{(A.1)}
\end{aligned}$$

By substituting $\alpha = 1/\mathbf{s}_k^T\mathbf{c}_k$ into (A.1), we verify that this equation is the same as the one from (4.7). In [Bur83] it is observed that the recursive MSSM formula remains unchanged if instead of $\mathbf{c}_k = \mathbf{S}_k^{-T}\mathbf{e}_n$ any multiple of this vector is chosen, e.g., $\mathbf{d}_k = \beta\mathbf{c}_k = \beta\mathbf{S}_k^{-T}\mathbf{e}_n$ for $\beta \in \mathbb{R}$. Based on this observation, it is deduced that if $k < n$, (the matrix $\mathbf{S}_k \in \mathbb{R}^{n\times k}$ does not have a square inverse), that any vector $\mathbf{c}_k$ can be used to define (A.1) as long as $\mathbf{c}_k^T\left[\begin{array}{ccc}\mathbf{s}_{k-1} & \cdots & \mathbf{s}_0\end{array}\right] = \mathbf{0}_{1\times(k-1)}$. In other words, $\mathbf{c}_k^T$ shares the properties of a column in the inverse matrix, if this matrix were to exist.

Appendix B

THE MSSM COMPACT REPRESENTATION

In this appendix the details of deriving the compact representation from (4.9) are described. In particular, we start from the expression

$$\begin{aligned}\mathbf{B}_k &= ([\ \mathbf{S}_k\ \ \mathbf{C}_k\ ]^T)^{-1} \begin{bmatrix} \mathbf{\Gamma}_k & \mathbf{Y}_k^T\mathbf{C}_k \\ \mathbf{C}_k^T\mathbf{Y}_k & \mathbf{C}_k^T\mathbf{B}_0\mathbf{C}_k \end{bmatrix} ([\ \mathbf{S}_k\ \ \mathbf{C}_k\ ])^{-1} \\ &= \begin{bmatrix}\ \mathbf{S}_k(\mathbf{S}_k^T\mathbf{S}_k)^{-1}\ \ \mathbf{C}_k\ \end{bmatrix} \begin{bmatrix} \mathbf{\Gamma}_k & \mathbf{Y}_k^T\mathbf{C}_k \\ \mathbf{C}_k^T\mathbf{Y}_k & \mathbf{C}_k^T\mathbf{B}_0\mathbf{C}_k \end{bmatrix} \begin{bmatrix}\mathbf{S}_k(\mathbf{S}_k^T\mathbf{S}_k)^{-1}\ \ \mathbf{C}_k\end{bmatrix}^T. \end{aligned} \tag{B.1}$$

Expanding the representation in (B.1) we obtain

$$\begin{aligned}\mathbf{B}_k = {} & \mathbf{S}_k(\mathbf{S}_k^T\mathbf{S}_k)^{-1}\mathbf{\Gamma}_k(\mathbf{S}_k^T\mathbf{S}_k)^{-1}\mathbf{S}_k^T + \\ & \mathbf{C}_k\mathbf{C}_k^T\mathbf{Y}_k(\mathbf{S}_k^T\mathbf{S}_k)^{-1}\mathbf{S}_k^T + \mathbf{S}_k(\mathbf{S}_k^T\mathbf{S}_k)^{-1}\mathbf{Y}_k^T\mathbf{C}_k\mathbf{C}_k^T + \mathbf{C}_k\mathbf{C}_k^T\mathbf{B}_0\mathbf{C}_k\mathbf{C}_k^T. \end{aligned} \tag{B.2}$$

From the definition of $\mathbf{C}_k$, the matrix $\mathbf{C}_k\mathbf{C}_k^T$ is the orthogonal projection onto the nullspace of $\mathbf{S}_k$, and it has the expression

$$\mathbf{C}_k\mathbf{C}_k^T = \mathbf{I}_n - \mathbf{S}_k(\mathbf{S}_k^T\mathbf{S}_k)^{-1}\mathbf{S}_k^T.$$

First compute

$$\begin{aligned}\mathbf{C}_k\mathbf{C}_k^T\mathbf{Y}_k(\mathbf{S}_k^T\mathbf{S}_k)^{-1}\mathbf{S}_k^T &= \mathbf{Y}_k(\mathbf{S}_k^T\mathbf{S}_k)^{-1}\mathbf{S}_k^T - \mathbf{S}_k(\mathbf{S}_k^T\mathbf{S}_k)^{-1}\mathbf{S}_k^T\mathbf{Y}_k(\mathbf{S}_k^T\mathbf{S}_k)^{-1}\mathbf{S}_k^T \\ &\equiv \mathbf{Y}_k\mathbf{\Xi}_k\mathbf{S}_k^T - \mathbf{S}_k\mathbf{\Xi}_k\mathbf{S}_k^T\mathbf{Y}_k\mathbf{\Xi}_k\mathbf{S}_k^T, \end{aligned}$$

where $(\mathbf{S}_k^T\mathbf{S}_k)^{-1} \equiv \mathbf{\Xi}_k$. Secondly compute

$$\begin{aligned}\mathbf{C}_k\mathbf{C}_k^T\mathbf{B}_0\mathbf{C}_k\mathbf{C}_k^T &= \mathbf{B}_0 - \mathbf{B}_0\mathbf{S}_k(\mathbf{S}_k^T\mathbf{S}_k)^{-1}\mathbf{S}_k^T - \mathbf{S}_k(\mathbf{S}_k^T\mathbf{S}_k)^{-1}\mathbf{S}_k^T\mathbf{B}_0 \\ &\quad + \mathbf{S}_k(\mathbf{S}_k^T\mathbf{S}_k)^{-1}\mathbf{S}_k^T\mathbf{B}_0\mathbf{S}_k(\mathbf{S}_k^T\mathbf{S}_k)^{-1}\mathbf{S}_k^T \\ &\equiv \mathbf{B}_0 - \mathbf{B}_0\mathbf{S}_k\mathbf{\Xi}_k\mathbf{S}_k^T - \mathbf{S}_k\mathbf{\Xi}_k\mathbf{S}_k^T\mathbf{B}_0 + \mathbf{S}_k\mathbf{\Xi}_k\mathbf{S}_k^T\mathbf{B}_0\mathbf{S}_k\mathbf{\Xi}_k\mathbf{S}_k^T.\end{aligned}$$

With the latter two terms the expression in (B.2) becomes

$$\begin{aligned}\mathbf{B}_k = \mathbf{B}_0 + \mathbf{S}_k\mathbf{\Xi}_k(\mathbf{\Gamma}_k - \mathbf{S}_k^T\mathbf{Y}_k - \mathbf{Y}_k^T\mathbf{S}_k + \mathbf{S}_k^T\mathbf{B}_0\mathbf{S}_k)\mathbf{\Xi}_k\mathbf{S}_k^T + \\ (\mathbf{Y}_k - \mathbf{B}_0\mathbf{S}_k)\mathbf{\Xi}_k\mathbf{S}_k^T + \mathbf{S}_k\mathbf{\Xi}_k(\mathbf{Y}_k^T - \mathbf{S}_k^T\mathbf{B}_0).\end{aligned}$$

With equations (4.3) and (4.4) from Section 4.2, then

$$\begin{aligned}\mathbf{\Gamma}_k - \mathbf{S}_k^T\mathbf{Y}_k - \mathbf{Y}_k^T\mathbf{S}_k &= \mathbf{L}_k + \mathbf{E}_k + \mathbf{L}_k^T - (\mathbf{L}_k + \mathbf{E}_k + \mathbf{T}_k) - (\mathbf{L}_k^T + \mathbf{E}_k + \mathbf{T}_k^T) \\ &= -(\mathbf{T}_k + \mathbf{E}_k + \mathbf{T}_k^T).\end{aligned}$$

By combining the previous two terms, we obtain

$$\begin{aligned}\mathbf{B}_k &= \mathbf{B}_0 + \mathbf{S}_k\mathbf{\Xi}_k\left(\mathbf{S}_k^T\mathbf{B}_0\mathbf{S}_k - (\mathbf{L}_k^T + \mathbf{E}_k + \mathbf{T}_k^T)\right)\mathbf{\Xi}_k\mathbf{S}_k^T \\ &\quad + (\mathbf{Y}_k - \mathbf{B}_0\mathbf{S}_k)\mathbf{\Xi}_k\mathbf{S}_k^T + \mathbf{S}_k\mathbf{\Xi}_k(\mathbf{Y}_k^T - \mathbf{S}_k^T\mathbf{B}_0) \\ &= \mathbf{B}_0 + \mathbf{\Psi}_k\mathbf{M}_k\mathbf{\Psi}_k^T,\end{aligned}$$

where

$$\mathbf{\Psi}_k \equiv \left[\mathbf{S}_k \;\; (\mathbf{Y}_k - \mathbf{B}_0\mathbf{S}_k)\right], \quad \mathbf{M}_k \equiv \begin{bmatrix} \mathbf{\Xi}_k\left(\mathbf{S}_k^T\mathbf{B}_0\mathbf{S}_k - (\mathbf{L}_k^T + \mathbf{E}_k + \mathbf{T}_k^T)\right)\mathbf{\Xi}_k & \mathbf{\Xi}_k \\ \mathbf{\Xi}_k & \mathbf{0} \end{bmatrix}.$$

This is the compact representation of the MSSM matrix as in (4.9).

Appendix C

TABLE OF CUTEST PROBLEMS

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

TABLE C.1

Unconstrained CUTEst problems used in EXPERIMENT III.

Bibliography

[BBE+16] J. Brust, O. Burdakov, J. B. Erway, R. F. Marcia, and Yuan Ya-xiang. Shape-changing L-SR1 trust-region methods. Technical Report 2016-2, Wake Forest University, 2016.

[Bec15] S. Becker. LBFGSB (L-BFGS-B) mex wrapper. https://www.mathworks.com/matlabcentral/fileexchange/35104-lbfgsb–l-bfgs-b–mex-wrapper, 2012–2015.

[BEM17] 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.

[Ben65] J. M. Bennett. Triangular factors of modified matrices. *Numerische Mathematik*, 7(3):217–221, 1965.

[BGYZ16] O. Burdakov, L. Gong, Y.-X. Yuan, and S. Zikrin. On efficiently combining limited memory and trust-region techniques. *Mathematical Programming Computation*, 9:101–134, 2016.

[BKS96] R. H. Byrd, H. F. Khalfan, and R. B. Schnabel. Analysis of a symmetric rank-one trust region method. *SIAM Journal on Optimization*, 6(4):1025–1039, 1996.

[BMP02] O. Burdakov, J. Martinez, and E. Pilotta. A limited-memory multipoint symmetric secant method for bound constrained optimization. *Annals Of Operations Research*, 117:51–70, 2002.

[BNS94] 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.

[Bro67] C. G Broyden. Quasi-newton methods and their application to function minimisation. *Mathematics of Computation*, 21:368–381, 1967.

[BS94] R. T. Behrens and L. L. Scharf. Signal processing applications of oblique projection operators. *IEEE Transactions on Signal Processing*, 42, 1994.

[Bur83] O. Burdakov. Methods of the secant type for systems of equations with symmetric Jacobian matrices. *Numerical Functional Analysis and Optimization*, 6(117):183–195, 1983.

[BWX96] J. V. Burke, A. Wiegmann, and L. Xu. Limited memory BFGS updating in a trust-region framework. Technical Report, University of Washington, 1996.

[BY02] 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.

[CDJT84] M. R. Celis, J. E. Dennis Jr., and R. A. Tapia. A trust region strategy for equality constrained optimization. Technical Report 84-1, Rice University, 1984.

[CE02] Y. Censor and T. Elfving. Block-iterative algorithms with diagonally scaled oblique projections for the linear feasibility problem. *SIAM J. Matrix Anal. Appl.*, 24(1):40–58, 2002.

[CGT91] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. Convergence of quasi-Newton matrices generated by the symmetric rank one update. *Math. Program.*, 50(2):177–195, March 1991.

[CGT00] A. R. Conn, N. I. M. Gould, and P. L. Toint. *Trust-Region Methods*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2000.

[Dav59] W. C. Davidon. Variable metric method for minimization. Technical Report ANL-5990, Argonne National Laboratory, 1959.

[Dav90] W. C. Davidon. Variable metric method for minimization. *SIAM J. Optim.*, 1(1), 1990.

[DEM16] O. DeGuchy, J. B. Erway, and R. F. Marcia. Compact representation of the full Broyden class of quasi-Newton updates. Technical Report 2016-4, Wake Forest University, 2016.

[DM77] J. E. Dennis and J. J. Moré. Quasi-newton methods, motivation and theory. *SIAM Rev.*, 19(1):46–89, 1977.

[DM02] E. Dolan and J.J Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91:201–213, 2002.

[EG10] 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.

[EGG09] 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.

[EM14] 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.

[EM15] 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.

[EM17] J. B. Erway and R. F. Marcia. On solving large-scale limited-memory quasi-Newton equations. *Linear Algebra and its Applications*, 515:196–225, 2017.

[Fle70] R. Fletcher. A new approach to variable metric algorithms. *The Computer Journal*, 13(3), 1970.

[FP63] R. Fletcher and M. J. D. Powell. A rapidly convergent descent method for minimization. *The Computer Journal*, 6:163–168, 1963.

[FS01] A. Forsgren and G. Sporre. On weighted linear least-squares problems related to interior methods for convex quadratic programming. *SIAM J. Matrix Anal. Appl.*, 23:42–56, 2001.

[Gay81] D. M. Gay. Computing optimal locally constrained steps. *SIAM J. Sci. Statist. Comput.*, 2(2):186–197, 1981.

[Ger04] E. M. Gertz. A quasi-newton trust-region method. *Math. Program.*, Ser. A 100:447–470, 2004.

[GGMS74] 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.

[GL01] P. E. Gill and M. W. Leonard. Reduced-hessian quasi-newton methods for unconstrained optimization. *SIAM J. Optim.*, 12:209–237, 2001.

[GL03] P. E. Gill and M. W. Leonard. Limited-memory reduced-hessian methods for large-scale unconstrained optimization. *SIAM J. Optim.*, 14:380–401, 2003.

[Gol80] D. Goldfarb. The use of negative curvature in minimization algorithms. Technical Report 80-412, Cornell University, 1980.

[GOT03] N. I. M. Gould, D. Orban, and P. L. Toint. CUTEr and SifDec: A constrained and unconstrained testing environment, revisited. *ACM Trans. Math. Software*, 29(4):373–394, 2003.

[Hag01] W. W. Hager. Minimizing a quadratic over a sphere. *SIAM J. Optim.*, 12(1):188–208, 2001.

[HP04] W. W. Hager and S. Park. Global convergence of SSM for minimizing a quadratic over a sphere. *Math. Comp.*, 74(74):1413–1423, 2004.

[KC02] D. R. Kincaid and E. W. Cheney. *Numerical analysis: mathematics of scientific computing*, volume 2. American Mathematical Soc., 2002.

[Lev44] K. Levenberg. A method for the solution of certain non-linear propblems in least squares. *Quarterly of Applied Mathematics*, 2(2):164–168, 1944.

[LN89] D. C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Math. Program.*, 45:503–528, 1989.

[LNP98] M. Lalee, J. Nocedal, and T. Plantenga. On the implementation of an algorithm for large-scale equality constrained optimization. *SIAM J. Optim.*, 8(3):682–706, 1998.

[Lu96] X. Lu. *A study of the limited memory SR1 method in practice*. PhD thesis, University of Colorado, 1996.

[Mar63] D.W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *J. Soc. Indust. Appl. Math.*, 11(2):431–441, 1963.

[MS83] J. J. Moré and D. C. Sorensen. Computing a trust region step. *SIAM J. Sci. and Statist. Comput.*, 4:553–572, 1983.

[Noc80] J. Nocedal. Updating quasi-Newton matrices with limited storage. *Math. Comput.*, 35:773–782, 1980.

[NW06] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, 2nd edition, 2006.

[O'L90] P. O'Leary. On bounds for scaled projections and pseudoinverses. *Linear Algebra and its Applications*, 132:115–117, 1990.

[Pow70] M.J.D. Powell. A new algorithm for unconstrained optimization. In *Nonlinear Programming*, pages 31–65, May 1970.

[RSS01] Marielba Rojas, Sandra A Santos, and Danny C Sorensen. A new matrix-free algorithm for the large-scale trust-region subproblem. *SIAM Journal on Optimization*, 11(3):611–646, 2001.

[RSS08] M. Rojas, S. A. Santos, and D. C. Sorensen. Algorithm 873: LSTRS: MATLAB software for large-scale trust-region subproblems and regularization. *ACM Trans. Math. Softw.*, 34(2):11:1–11:28, March 2008.

[Sch83] R. B. Schnabel. Quasi-newton methods using multiple secant equations; cu-cs-247-83. Technical Report 244, Computer Science Technical Reports, 1983.

[Sor82] D. C. Sorensen. Newton's method with a model trust region modification. *SIAM J. Numer. Anal.*, 19(2):409–426, 1982.

[SP78] D. F. Shanno and K. H. Phua. Matrix conditioning and nonlinear optimization. *Mathematical Programming*, 14(1):149–160, 1978.

[Ste83] T. Steihaug. The conjugate gradient method and trust regions in large scale optimization. *SIAM J. Numer. Anal.*, 20:626–637, 1983.

[Ste89] G.W. Stewart. On scaled projections and pseudoinverses. *Linear Algebra and its Applications*, 112:189–193, 1989.

[Var85] A. Vardi. A trust region algorithm for equality constrained minimization: Convergence properties and implementation. *SIAM J. Numer. Anal.*, 22(3), 1985.

[Yua15] Y.X. Yuan. Recent advances in trust region algorithm. *Math. Program., Ser. B*, pages 249–281, 2015.

[YuaNA] Y.-X. Yuan. Trust region algorithms for constrained optimization. Technical report, State Key Laboratory of Scientific and Engineering Computing, NA.

[ZBN97] C. Zhu, R.H Byrd, and J. Nocedal. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software*, 23:550–560, 1997.

[Zik14] S. Zikrin. *Large-Scale Optimization Methods with Application to Design of Filter Networks.* PhD thesis, Linköping University, 2014.