MATHEMATICAL ENGINEERING
TECHNICAL REPORTS

Solving the Trust Region Subproblem by a Generalized Eigenvalue Problem

Satoru ADACHI, Satoru IWATA,
Yuji NAKATSUKASA and Akiko TAKEDA

METR 2015–14 April 2015

DEPARTMENT OF MATHEMATICAL INFORMATICS
GRADUATE SCHOOL OF INFORMATION SCIENCE AND TECHNOLOGY
THE UNIVERSITY OF TOKYO
BUNKYO-KU, TOKYO 113-8656, JAPAN

WWW page: http://www.keisu.t.u-tokyo.ac.jp/research/techrep/index.html

Solving the trust region subproblem by a generalized eigenvalue problem

Satoru Adachi Satoru Iwata Yuji Nakatsukasa Akiko Takeda*

March 31, 2015

Abstract

The trust region subproblem is usually solved via an iterative process of solving linear systems or eigenvalue problems. An alternative approach is to use semidefinite programming, but this also involves solving linear systems iteratively. In some cases, many iterations are needed for convergence. In this work we advocate an algorithm that solves just one generalized eigenvalue problem, or more specifically just one eigenpair. In addition to being easy to implement and to analyze and predict the efficiency (due to lack of iterations), our algorithm allows for a non-standard norm without a change of variables and is suited both to the dense and the large-sparse cases. We also discuss how to deal with the so-called hard case. Experiments suggest that our algorithm is superior to existing ones both in accuracy and especially efficiency, particularly for large-sparse problems.

keywords: Trust-region subproblem, generalized eigenvalue problem, non-standard inner product, hard-case

1 Introduction

The trust region subproblem (TRS)

$$\underset{\|p\|_B \leq \gamma}{\text{minimize}} \ g^\top p + \frac{1}{2} p^\top A p, \tag{1}$$

where $g \in \mathbb{R}^n$, $A, B \in \mathbb{R}^{n \times n}$ are symmetric and $B \succ O$ is symmetric positive definite, is commonly solved as a subproblem for dealing with general nonlinear optimization problems [5], [22, Ch. 4], in which A is the Hessian and g is the gradient of the objective function at the current approximate solution. Note that we allow for the constraint $\|p\|_B \leq \gamma$ in a nonstandard norm defined by a general positive definite $B \neq I$, in which $\|p\|_B = \|B^{1/2}p\| = \sqrt{p^\top B p}$; an appropriate and nontrivial choice $B \neq I$ can be important for example when working in

*This work was supported by JSPS Scientific Research Grants No. 26540007 and No. 26870149.

a properly scaled trust region to solve the nonlinear problem efficiently [19]. For more on TRS and its solution see the book [5].

The necessary and sufficient condition for the global solution to the TRS is the following:

Theorem 1.1 *A vector p_* is an optimal solution to the TRS* (1) *if and only if there exists $\lambda_* \geq 0$ such that*

$$\begin{aligned} \|p_*\|_B &\leq \gamma, && (2)\\ (A+\lambda_* B)p_* &= -g, && (3)\\ \lambda_*(\gamma - \|p_*\|_B) &= 0, && (4)\\ A+\lambda_* B &\succeq O. && (5) \end{aligned}$$

This result is well known when $B = I$ and mentioned in [5, Thm. 7.4.1] and [14] as a necessary condition, and can be obtained by modifying the discussion in [21, 22] to general $B \succ O$ by a change of variables, or by finding the KKT (Karush-Kuhn-Tucker) conditions and invoking the result in [6]; we give more details in Section 2.

To our knowledge, most existing methods for solving TRS assume $B = I$ and involve an iterative process during which a parameter is adjusted to one that corresponds to the solution. For example,

1. Moré and Sorensen [21]: iteratively solves symmetric positive-definite linear systems via the Cholesky factorization. During the iteration a shift parameter is adjusted. Safeguard techniques are sometimes necessary to ensure convergence to the solution. This is the standard approach for dense and moderate size problems, say $n \leq 1000$.

2. Sorensen [29], refined and implemented by Rojas et. al [26, 27]: iteratively computes the smallest eigenvalue of a parameterized matrix $P(\alpha)$, where α is adjusted during the iteration to find the solution. Some safeguard technique is needed to guarantee convergence. This method is well suited when A is large and sparse.

3. Rendl and Wolkowicz [25]: solves TRS via semidefinite programming (SDP). The standard SDP solver based on an interior-point method also involves an iteration of linear systems. More generally, quadratic programming with one quadratic constraint can be solved by an SDP [3, App. B].

4. Gould et al. [12]: pursues a more practical goal of reducing the overall cost of solving the nonlinear optimization problem via TRS, in which a Krylov subspace $\mathcal{K}(A, x)$ is computed iteratively (or $\mathcal{K}(P^{-1}A, x)$ with a preconditioner P) by the Lanczos method and a projected trust-region problem of smaller size is solved. This algorithm makes little attempt to obtain high accuracy in the TRS solution.

The traditional approach to deal with $B \neq I$ is to reduce the problem to an equivalent one with $B = I$ by a change of variables: defining $\tilde{p} := B^{\frac{1}{2}}p$, one can reduce (1) to an equivalent TRS with $B = I$, i.e., the minimization of $(B^{-\frac{1}{2}}g)^\top \tilde{p} + \frac{1}{2}\tilde{p}^\top (B^{-\frac{1}{2}}AB^{-\frac{1}{2}})\tilde{p}$ subject

to $\|\tilde{p}\| \leq \gamma$. However, this reduction involves computing the matrix square root or the Cholesky factor of B and their inverse, which can be expensive when n is large and B is not easily invertible, and numerically unstable when B is ill-conditioned. Furthermore, even if the Cholesky factorization of B is easy to compute (e.g., when it is tridiagonal or banded), the matrix $B^{-\frac{1}{2}}AB^{-\frac{1}{2}}$ generally loses the problem structure: e.g., the inverse of an irreducible tridiagonal matrix is dense.

The algorithms in [21, 25] are designed for $B = I$, and so are most publically available implementations [25, 27]: an extension is described in [5, Sec. 7.5.6] by generalizing their parameterized $(n+1) \times (n+1)$ matrix $N(\alpha) := \begin{bmatrix} \alpha & g^\top \\ g & A \end{bmatrix}$ to the pencil $\begin{bmatrix} \alpha & g^\top \\ g & A \end{bmatrix} - \lambda \begin{bmatrix} 1 & \\ & B \end{bmatrix}$ where λ is the Lagrange multiplier, but this would still involve iterations with respect to α, and our focus here is to remove such iterations. We note that Gould et. al. [12] deal with general norms $\|\cdot\|_B$ for positive definite B (their algorithm requires that B is easily invertible and that B approximates A).

While [12] suggests stopping the iteration for TRS once it attains a sufficient reduction in the objective value for the original optimization problem, showing numerical evidence that it often suffices to obtain an approximate solution to the TRS, in this paper we treat TRS as a problem of independent interest and attempt to solve the TRS as accurately as possible. Just as those employed in [25, 29], our approach is based on eigenvalue problems. However, unlike most algorithms proposed in the literature we look for an algorithm that solves the TRS by a *single* generalized eigenvalue problem. Other approaches that iteratively solve the TRS include the dogleg method [22, Sec. 4.1], optimization of difference of convex functions [31], an improvement of the Moré-Sorensen algorithm using Taylor series approximation [14], and one based on the BFGS method [1]. All these algorithms require iteratively running a computational routine, and the number of iterations is often unpredictable and potentially large.

Another aspect of our method is that it can compute all the KKT points in addition to the global solution. For example, the global maximizer can also be obtained. We note that Martinez [18] shows that there can be at most two local minima for the TRS, and we can check the number of local minima (whether it is one or two).

One exception to the iteration-based methods is the one by Gander, Golub and von Matt [9], which reduces TRS to a quadratic eigenvalue problem, which they linearize to a standard eigenvalue problem of size $2n$. However, in that paper they report that their eigenvalue-based approach is slower and less accurate than the method based on the secular equation by Moré and Sorensen [21]. This is perhaps why this approach appears to have received less attention than those mentioned above.

The algorithm we advocate here, however, results in an extension of [9], which turns out to be both efficient and accurate. It seems that the slow speed and loss of accuracy reported in [9] was largely due to the relatively unrefined eigenvalue solver available those days. In addition to showing that the accuracy and speed are both greatly improved by today's much developed eigensolvers, our approach, which is based on a different derivation but results in closely related eigenvalue problems, further improves the algorithm by allowing $B \neq I$ and

preserving symmetry. Moreover, the paper [9] does not discuss how to deal with the "hard case", which we do in detail here.

This work was initially motivated by the following observation. Consider the simple case $B = I$ and let $A = VDV^\top$ be the eigenvalue decomposition with eigenvalues $d_i, i = 1, \ldots, n$. We focus on the solution with $\|p\| = \gamma$, which is generally the more difficult case than $\|p\| < \gamma$ (see Section 2.1). Then by (3) we have $p = -V(D + \lambda I)^{-1}V^\top g$, and writing out the condition $\|p\| = \gamma$ and defining $\hat{g} = V^\top g$, one sees that the solution can be obtained via a rational equation in λ of the form

$$\gamma^2 = \sum_{j=1}^{n} \frac{\hat{g}_j^2}{(d_j + \lambda)^2}. \tag{6}$$

This equation has been presented in the literature [10, 21, 22], but it is usually treated as a "difficult" nonlinear equation that needs to be solved through an iterative process. Nonetheless, (6) is nothing else than a rational rootfinding problem, which can mathematically be reduced to a polynomial rootfinding problem by multiplying out $\prod_{j=1}^{n}(d_j + \lambda)^2$, which can then be solved by a *single* eigenvalue problem via linearization such as the companion matrix, without iterations (except those used within the eigensolver). Although numerically this "polynomialization" is usually not recommended (and we will not pursue it), this observation does suggest that a method based on iterations is perhaps unnecessary.

In this work we show (or rather rediscover) that indeed we can solve the TRS in one step by a generalized eigenvalue problem. The lack of iterations makes the algorithm easy to implement, and the runtime predictable. Moreover, our algorithm flexibly adapts to the structure of the problem. For example, it can appropriately deal with the case where A, B are large sparse matrices.

This paper is organized as follows. In Section 2 we detail the optimality conditions and the KKT conditions. We then describe our algorithm in Section 3. Section 4 discusses how to deal with the hard case. In Section 6 we compare our algorithm with existing methods. Section 7 shows numerical experiments to illustrate and compare the performance.

2 Optimality conditions

By introducing in (1) the change of variables $\tilde{A} = B^{-\frac{1}{2}}AB^{-\frac{1}{2}}$, $\tilde{g} = B^{-\frac{1}{2}}g$, and $\tilde{p} = B^{\frac{1}{2}}p$, we obtain the following equivalent problem:

$$\begin{aligned} &\text{minimize} \quad \tilde{g}^\top\tilde{p} + \frac{1}{2}\tilde{p}^\top\tilde{A}\tilde{p}, \\ &\text{subject to} \quad ||\tilde{p}|| \leq \gamma. \end{aligned} \tag{7}$$

This is the standrad TRS with $B = I$, and the necessary and sufficiency condition for the global solution is known as follows [10, 21, 22].

Lemma 2.1 *A vector $\tilde{p}_*$ is an optimal solution for (7) if and only if there exists $\lambda_* \geq 0$ such that*

$$\begin{aligned} ||\tilde{p}_*|| &\leq \gamma, \\ (\tilde{A} + \lambda_* I)\tilde{p}_* &= -\tilde{g}, \\ \lambda_*(\gamma - ||\tilde{p}_*||) &= 0, \\ \tilde{A} + \lambda_* I &\succeq O. \end{aligned}$$

It is also known that λ_* is unique. From the equivalence between (7) and (1) and the fact that $||\tilde{p}|| = ||p||_B$, we obtain Theorem 1.1 by reverting the change of variables.

2.1 Complementary slackness

Equation (4) shows that the TRS solution belongs to either of the following two cases, which are not necessarily disjoint.

1. $\lambda_* = 0$,
2. $\gamma - \|p_*\|_B = 0$.

Roughly, the two cases represent the TRS solutions in the interior (first case) or on the boundary (second) of the trust region $\|p_*\|_B \leq \gamma$. Dealing with the first case is immediate as is well known [21]; in this case $Ap_* = -g$, which determines p_* uniquely if A is nonsingular, and we need to check whether $\|p_*\|_B \leq \gamma$ is satisfied; if it is, then this gives a candidate for the TRS solution that lies in the interior (boundary if $\|p_*\|_B = \gamma$ happens to hold) of the trust region. If further $A \succeq O$ (which may not be easily verifiable), then this is the TRS solution, we give more details in Section 5. If A is singular then the linear system $Ax = -g$ is generically not solvable, and the TRS solution is in the second case $\|p_*\|_B = \gamma$. If $Ax = -g$ happens to be solvable then p_* is of the form $p_* = -A^\dagger g + Nv$ where $A^\dagger$ denotes the pseudoinverse and N is a basis for the null space of A, which has to be orthogonal to g so it has no effect on the objective value. We choose v so that $\|p_*\|_B = \gamma$ (details are given in Section 4).

In practice, checking the positive semidefiniteness of A may be a costly operation (requiring $O(n^3)$ if A is dense), so instead of checking it we simply solve $Ap = -g$ for p via a direct solver or via MINRES (or the conjugate gradient (CG) algorithm if A is known to be positive definite), and if p is within the trust region we keep it as a candidate solution and compare it with the solution obtained from the second case.

In what follows we elaborate on the second case $\|p_*\|_B = \gamma$.

2.2 KKT conditions

The first three conditions (2)–(4) of the TRS optimality conditions in Theorem 1.1 represent the KKT conditions. The last (5) will turn out to show that indeed the solution corresponds to the KKT point with the largest Lagrange multiplier.

By (3), for any KKT multiplier λ, unless the matrix $A + \lambda B$ is singular we can write p as a function of λ as

$$p(\lambda) = -(A + \lambda B)^{-1} g. \tag{8}$$

Since we focus on the case $\|p\|_B = \gamma$, plugging (8) into this equation we obtain

$$g^\top (A + \lambda B)^{-1} B (A + \lambda B)^{-1} g = \gamma^2. \tag{9}$$

Now let W be the matrix that achieves the simultaneous diagonalization by congruence [11]

$$W^\top (A, B) W = (D, I) \tag{10}$$

where $D = -\mathrm{diag}(\mu_1, \ldots, \mu_n)$ (note the minus sign as here we define μ_i as the eigenvalues of the pencil $A + \lambda B$, not $A - \lambda B$) with $\mu_1 \leq \mu_2 \leq \cdots \leq \mu_n$. Then (9) can be written as

$$\gamma^2 = (W^\top g)^\top (D + \lambda I)^{-2} (W^\top g).$$

Writing $W = [w_1, \ldots, w_n]$ this is equivalent to

$$(h(\lambda) :=) \ \sum_{j=1}^{n} \frac{(w_j^\top g)^2}{(\lambda - \mu_j)^2} = \gamma^2, \tag{11}$$

which is a rational equation with respect to λ. Our aim will be to construct a generalized eigenvalue problem whose eigenvalues contain the values of λ satisfying (11).

One can verify that exactly one value of λ exists such that (9) holds with $A + \lambda B \succeq O$. To see this, note that the poles of $h(\lambda)$ are at the eigenvalues of the pencil $A + \lambda B$, and on the interval $\lambda \in (\mu_n, \infty)$ the function $h(\lambda)$ is strictly decreasing. This establishes the claim when $w_n^\top g \neq 0$; see Section 4 for a treatment of the case $w_n^\top g = 0$. In either case, this value of λ will be equal to the Lagrange multiplier λ_* in the TRS solution that our algorithm shall compute, as we show in the next section.

3 Algorithm

The methodology here parallels that of [16], in which a more specialized problem of finding the point-ellipsoid distance was considered.

For the matrix pencil $A + \lambda B$, we denote its range and null space by $\mathcal{R}(A + \lambda B)$ and $\mathcal{N}(A + \lambda B)$, respectively.

3.1 Solution via generalized eigenvalues

The starting point is to introduce two matrix pencils whose eigenvalues include the desired KKT multiplier λ_*. We define the $(2n + 1) \times (2n + 1)$ matrix pencil

$$M(\lambda) = \begin{bmatrix} \gamma^2 & 0 & g^\top \\ 0 & -B & A + \lambda B \\ g & A + \lambda B & O \end{bmatrix} \tag{12}$$

and the $2n \times 2n$ matrix pencil

$$\tilde{M}(\lambda) = \begin{bmatrix} -B & A + \lambda B \\ A + \lambda B & -\frac{gg^\top}{\gamma^2} \end{bmatrix}. \tag{13}$$

The crucial facts are that the eigenvalues of these pencils provide the values of λ satisfying the KKT conditions, and furthermore we can obtain the pair (λ_*, p_*) satisfying the optimality conditions (3)–(5) by computing the largest real eigenpair. We establish these facts in the next two results.

Lemma 3.1 *For every KKT multiplier $\lambda \neq 0$ satisfying (2)–(4), we have* $\det M(\lambda) = \det \tilde{M}(\lambda) = 0$.

Proof. Let λ be a KKT Lagrange multiplier satisfying (2)–(4). If $\det(A + \lambda B) = 0$ at λ, then it follows from $g \in \mathcal{R}(A + \lambda B)$ that $\det M(\lambda) = \det \tilde{M}(\lambda) = 0$.

We now deal with λ such that $\det(A + \lambda B) \neq 0$. Define $p(\lambda) = -(A + \lambda B)^{-1} g$ as in (8), and let $X(\lambda) = \begin{bmatrix} 1 & & \\ p(\lambda) & I & \\ & & I \end{bmatrix}$. Then $X(\lambda)$ is unimodular $\det X(\lambda) \equiv 1$, and we have

$$\begin{aligned} \det M(\lambda) &= \det X(\lambda)^T M(\lambda) X(\lambda) \qquad (14) \\ &= \det \begin{bmatrix} \gamma^2 - p(\lambda)^\top B p(\lambda) & p(\lambda)^\top B & 0 \\ B p(\lambda) & -B & A + \lambda B \\ 0 & A + \lambda B & O \end{bmatrix} \\ &= (-1)^n \det(A + \lambda B)^2 \{\gamma^2 - p(\lambda)^\top B p(\lambda)\}. \qquad (15) \end{aligned}$$

If $(\lambda, p(\lambda))$ satisfies the KKT conditions with $\lambda \neq 0$, then recalling (9) we have $\gamma^2 - p(\lambda)^\top B p(\lambda) = 0$, and so $\det M(\lambda) = 0$.

Now defining $T = \begin{bmatrix} 1 & & -\frac{1}{\gamma^2} g^\top \\ & I_n & \\ & & I_n \end{bmatrix}$, we have

$$T^\top M(\lambda) T = \begin{bmatrix} \gamma^2 & & \\ & -B & A + \lambda B \\ & A + \lambda B & -\frac{1}{\gamma^2} g g^\top \end{bmatrix} = \begin{bmatrix} \gamma^2 & \\ & \tilde{M}(\lambda) \end{bmatrix}. \tag{16}$$

It then follows that $\det M(\lambda) = \gamma^2 \det \tilde{M}(\lambda)$, and hence together with the above result $\det \tilde{M}(\lambda) = \det M(\lambda) = 0$ at any nonzero KKT multiplier λ. □

The above lemma shows that the TRS solution on the boundary satisfies $\det M(\lambda) = 0$ and $\det \tilde{M}(\lambda) = 0$, both of which can be solved via a generalized eigenvalue problem. The eigenvalues λ contain the Lagrange multipliers at the KKT points, so the multiplier for the TRS solution must be one of the $2n$ finite eigenvalues (note that $M(\lambda)$ has one eigenvalue at infinity, which is not the one of interest).

Fortunately, both $M(\lambda)$ and $\tilde{M}(\lambda)$ are *regular* matrix pencils, that is, their determinants are nonzero for some λ and the number of eigenvalues is equal to their size. To see this, observe that $\tilde{M}(\infty) := \begin{bmatrix} O & B \\ B & O \end{bmatrix}$ is nonsingular, and that $\tilde{M}(\lambda)$ is obtained from $M(\lambda)$ by taking its Schur complement. Therefore the number of eigenvalues is finite, more precisely, $2n+1$ and $2n$, respectively, and $2n$ of them match the $2n$ roots of the rational equation (6). Furthermore, (16) shows that the eigenvectors corresponding to the finite eigenvalues of M and $\tilde{M}$ are closely related:

$$\tilde{M}(\lambda) \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = 0 \quad \Rightarrow \quad M(\lambda) \begin{bmatrix} -\frac{1}{\gamma^2} g^\top y_2 \\ y_1 \\ y_2 \end{bmatrix} = 0. \tag{17}$$

Note that the converse implication $\Leftarrow$ also holds unless $y_1 = y_2 = 0$, which is an eigenvector of $M(\lambda)$ at ∞. The next results show that in fact the λ_* of the TRS solution corresponds to the largest real eigenvalue of $M(\lambda)$ and $\tilde{M}(\lambda)$, and generically the solution p_* can be obtained from the corresponding eigenvector.

Theorem 3.1 *For the TRS solution (λ_*, p_*) on the boundary $\|p_*\|_B = \gamma$ satisfying (2)–(5), the multiplier λ_* is equal to the largest real eigenvalue of $M(\lambda)$ (excluding $\lambda = \infty$) and $\tilde{M}(\lambda)$. Furthermore, if $\lambda_* > \mu_n$ (where μ_n is the largest eigenvalue of $A + \lambda B$), then denoting the eigenvectors of $M(\lambda)$ and $\tilde{M}(\lambda)$ at $\lambda = \lambda_*$ by $\begin{bmatrix} \theta \\ y_1 \\ y_2 \end{bmatrix}$ and $\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$, both θ and $g^\top y_2$ are nonzero, and p_* can be obtained by*

$$p_* = -\frac{\gamma^2}{g^\top y_2} y_1 = \frac{1}{\theta} y_1. \tag{18}$$

Proof. The fact $\lambda_* = \lambda_{\max}$ has been shown in [6, 16] for special cases: [6] for $B = I$ and [16] for $A = I, g = 0$; see also [20]. By a change of coordinates we can extend these results to the TRS (1).

Alternatively, we can directly obtain the fact as follows. First, from $A + \lambda_* B \succeq O$ in (5) we must have $\lambda_* \geq \mu_n$ where μ_n is the largest eigenvalue of $A + \lambda B$. To see this, recall from (15) that each eigenvalue of $M, \tilde{M}$ is either an eigenvalue of $A + \lambda B$, or a solution to $h(\lambda) = \gamma^2$ in (11), and $h(\lambda)$ is strictly decreasing on (μ_n, ∞) and $h(\infty) = 0$. Since if $w_n^\top g \neq 0$ then $\lim_{\lambda \to \mu_n + 0} h(\lambda) = +\infty$, so exactly one $\lambda_{\max} \in (\mu_n, \infty)$ satisfies $h(\lambda) = \gamma^2$. If $w_n^\top g = 0$ then either $h(\mu_n) \geq \gamma^2$ or $h(\mu_n) < \gamma^2$. If $h(\mu_n) \geq \gamma^2$ then $\lambda_* = \lambda_{\max} \in [\mu_n, \infty)$ by the same argument. If $h(\mu_n) < \gamma^2$ then we must have $\lambda_* = \lambda_{\max} = \mu_n$ by (5) (this is the only case where $h(\lambda_*) \neq \gamma^2$: the "hard case").

We next discuss how to obtain p_*. We first note that in view of (11), $h(\lambda)$ has a pole at μ_n unless g is orthogonal to $\mathcal{N}(A + \mu_n B)$, (the eigenspace of $A + \lambda B$ corresponding to μ_n), so the assumption $\lambda_* > \mu_n$ must hold.

From the eigenvector $\left[\begin{smallmatrix} y_1 \\ y_2 \end{smallmatrix}\right]$ such that $\tilde{M}(\lambda_*) \left[\begin{smallmatrix} y_1 \\ y_2 \end{smallmatrix}\right] = 0$, that is,

$$\begin{bmatrix} -B & A \\ A & -\frac{gg^\top}{\gamma^2} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = -\lambda_* \begin{bmatrix} 0 & B \\ B & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \tag{19}$$

with $\lambda_* > \mu_n$, the TRS solution p_* is a multiple of y_1: to verify this we shall show that $(A + \lambda_* B)y_1$ is a multiple of g, which suffices as $A + \lambda_* B$ is nonsingular since $\lambda_* > \mu_n$, and hence $p_* = -(A + \lambda_* B)^{-1} g$.

From the lower block of (19), we have $(A + \lambda_* B)y_1 = \frac{g(g^\top y_2)}{\gamma^2}$, which is a scalar multiple of g. Therefore, provided that $g^\top y_2 \neq 0$, we obtain (18) as required. The last equality $-\frac{\gamma^2}{g^\top y_2} y_1 = \frac{1}{\theta} y_1$ follows from (17).

It remains to show that $\theta, g^\top y_2$ are nonzero if $\lambda_* > \mu_n$. By (17) and the discussion following it, for $\theta \neq 0$ is equivalent to $g^\top y_2 \neq 0$, so it suffices to prove $\theta \neq 0$. First observe that the eigenvalues of $M, \tilde{M}$ that are larger than μ_n must be simple, from the determinant expansion (15), hence y_1, y_2 are unique up to scalar scaling. We can show that the first element θ of the eigenvector of $M(\lambda)$ at λ_* is 0 only if λ_* is an eigenvalue of $A + \lambda B$, contradicting the assumption $\lambda_* > \mu_n$. Indeed if $\theta = 0$ then y_2 is also 0; this is because $\begin{bmatrix} \gamma^2 & 0 & g^\top \\ 0 & -B & A + \lambda_* B \\ g & A + \lambda_* B & O \end{bmatrix} \begin{bmatrix} 0 \\ y_1 \\ y_2 \end{bmatrix} = 0$ implies $(A + \lambda_* B)y_1 = -By_1 + (A + \lambda_* B)y_2 = 0$, and hence $(A + \lambda_* B)B^{-1}(A + \lambda_* B)y_2 = 0$. Using the diagonalization $W^\top (A, B)W = (D, I)$, this yields $W^\top (\lambda_* I + D)^2 W y_2 = 0$, from which we obtain $W^\top (\lambda_* I + D)W y_2 = 0$, which is equivalent to $(A + \lambda_* B)y_2 = 0$. We cannot have $y_2 = 0$, as then we need $\left[\begin{smallmatrix} -B \\ A + \lambda_* B \end{smallmatrix}\right] y_1 = 0$, so $y_1 = 0$, meaning the whole eigenvector is zero. Hence (λ_*, y_2) is an eigenpair of $A + \lambda B$, a contradiction. Hence $\theta \neq 0$. □

How to Compute the Optimal Solution p_*

In practice, computing the solution p_* from (18) may introduce unnecessary numerical errors, and we choose simply to obtain

$$p_* = -\text{sign}(g^\top y_2)\gamma \frac{y_1}{\|y_1\|_B}, \tag{20}$$

which is on the boundary to working precision: $\|p_*\|_B = \gamma$.

We note that the $2n$ size of the matrix pencil $\tilde{M}(\lambda)$ is the smallest possible since (11) is a rational equation, which genericallly becomes a degree $2n$ polynomial when the denominator is multiplied out.

In the hard case finding the solution is not straightforward; this is described in Section 4.

Which matrix pencil to use, $M(\lambda)$ or $\tilde{M}(\lambda)$?

The generalized eigenvalue problem $M(\lambda)$ has one additional eigenvalue at ∞, but the matrices involved are explicitly sparse. On the other hand, $\tilde{M}$ is one size smaller with no eigenvalue at infinity, but contains $gg^\top$, which is rank-one but dense. Thus the choice between M and $\tilde{M}$ should be made based on the properties of the eigensolver available. Some eigensolvers–such as MATLAB's `eigs`–allow the user to provide just a routine that multiplies the matrices to a vector; in this case the rank-one structure can be exploited. For this reason we use $\tilde{M}$ in our MATLAB experiments.

Comparison with the algorithm by Gander, Golub and von Matt

The algorithm by Gander, Golub and von Matt [9] considers the case $B = I$ and finds the largest eigenvalue of the $2n \times 2n$ matrix

$$\begin{bmatrix} A & -I \\ -\frac{gg^\top}{\gamma^2} & A \end{bmatrix}. \tag{21}$$

This is a nonsymmetric matrix, with a dense bottom-left block. We can obtain (21) when $B = I$ by the equivalence transformation of right-multiplying $\left[\begin{smallmatrix} O & I \\ I & O \end{smallmatrix}\right]$ to $\tilde{M}$. Therefore, we arrived at a different derivation of (21) and generalized it to $B \neq I$. In [9] solving TRS via (21) is not recommended over the secular equation approach based on [18], observing the inaccurate results in their experiments.

As we shall see in our experiments, this seems to have been largely due to the relatively undeveloped eigenvalue solvers available at the time: with today's eigensolvers the algorithm is both fast and accurate. This paper revives such approaches based on a generalized eigenproblem and extends their applicability to general B, proposes a way to exploit symmetry and deals with the hard case.

3.2 The rightmost eigenvalue is real

We have seen that λ_* at the TRS solution is equal to the largest real eigenvalue of $M(\lambda)$ and $\tilde{M}(\lambda)$. For the eigensolver it helps to further know that λ_* is indeed the rightmost eigenvalue, that is, there is no nonreal eigenvalue that lies to the right of λ_*.

Proposition 3.1 *The rightmost eigenvalues of $\tilde{M}(\lambda)$ and $M(\lambda)$ (excluding ∞) are both real and equal to λ_*.*

Proof. It suffices to show that the rightmost eigenvalue of $\tilde{M}(\lambda)$ is real. Suppose that $\tilde{\lambda} = \alpha + \beta\mathrm{i}$ where $\alpha, \beta \in \mathbb{R}$, $\alpha > \mu_n$ and $\beta > 0$ is a nonreal eigenvalue. Then $\alpha - \beta\mathrm{i}$ must also be an eigenvalue. Now if $\alpha > \lambda_* \geq \mu_n$, then recalling (11) we have $h(\lambda) = \sum_{j=1}^n \frac{(w_j^\top g)^2}{(\lambda-\mu_j)^2}$, and since the imaginary part of $\frac{1}{(\tilde{\lambda}-\mu_j)^2}$ is strictly negative for all j, we conclude that the imaginary part of $h(\tilde{\lambda})$ must also be strictly negative. Hence $h(\tilde{\lambda})$ cannot be equal to γ^2, so $\tilde{\lambda}$ is not an eigenvalue of $\tilde{M}(\lambda)$. □

We have shown that the TRS solution can be obtained from the rightmost (which is the largest real) eigenpair of $M(\lambda)$ or $\tilde{M}(\lambda)$, so we need to compute only the largest eigenvalue, for which in many cases solvers are available that are more efficient than computing the whole eigenvalues via the standard QR or QZ algorithms; for example the Arnoldi algorithm provides an effective means for computing extremal eigenvalues of large-sparse matrices. In view of Proposition 3.1, we can compute the desired eigenpair for example by the MATLAB command `eigs(M,'lr')`.

4 Dealing with the "hard case"

The so-called "hard case" is a difficulty that is known to arise in the standard $B = I$ case, and it is of course present also when $B \succ O$ is a general positive definite matrix. Although mathematically the hard case happens only on a set of problems of measure zero, it can happen for matrices with structures and numerically there are "nearly hard cases," which can be equally challenging. Indeed a number of studies such as [21, 26] have focused on the hard case with $B = I$.

We first state its definition for general $B \succ 0$ and explain why the difficulties arise.

Definition 4.1 *When λ_* for the TRS solution is the largest eigenvalue μ_n of the pencil $A + \lambda B$ and $g \perp \mathcal{N}(A + \lambda_* B)$, the TRS is called the "hard case".*

The reason this case is difficult is the following: Recall (11). *Assuming that* at least one of the terms $w_i^\top g$ for all i such that $\mu_i = \mu_n$ is nonzero, the rational function $h(\lambda)$ can be easily seen to have a pole at μ_n and therefore the real solution to $h(\lambda) = \gamma^2$ to the right of μ_n corresponds to the TRS solution.

This argument is invalid in one situation[1]: as alluded to in Section 2.1, when $w_i^\top g = 0$ for all such i, or equivalently when g is orthogonal to all the eigenvectors corresponding to μ_n. This is precisely the condition $g \perp \mathcal{N}(A + \lambda_* B)$. In this case the rational equation cannot be guaranteed to have a solution at λ larger than μ_n. Under the condition $g \perp \mathcal{N}(A + \lambda_* B)$ there are still two cases: (i) $\lambda_* > \mu_n$, and (ii) otherwise, $\lambda_* = \mu_n$. The second case is the hard case by our definition. Note that in the literature sometimes the hard case is defined simply by $g \perp \mathcal{N}(A + \lambda_* B)$ [22]. In [7] the hard case is further separated into two cases as we did here.

The reason we choose to define the hard case as above is that our algorithm faces difficulty only in that situation. In our algorithm, in which the generalized eigenvalue problem essentially solves (9), the case (i) is no problem at all because the computed eigenvalue is strictly larger than μ_n, and the soluton p_* can be extracted from the eigenvector of M or $\tilde{M}$ just as when $w_n^\top g \neq 0$.

In the second case, however, we do face difficulties: The largest eigenvalue of $\tilde{M}$ and M is $\lambda_* = \mu_n$, but the matrix $A + \lambda_* B$ is singular and recalling (15), the vector $p(\lambda_*)$ (or $(A + \lambda_* B)^\dagger g$) does not provide the TRS solution. Indeed the linear system $(A + \lambda_* B)x = -g$ has infinitely many solutions. The challenge is therefore to find the solution p_* such that $(A + \lambda_* B)p_* = -g$ and $\|p_*\|_B = \gamma$.

The hard case has been a major challenge in the literature for $B = I$, and many approaches have been introduced; see for example [21, 26]. Here our solution is based on that in [7], which we modify to adapt to a general positive definite B.

[1] It is worth noting that the condition $w_i^\top g = 0$ means g is orthogonal to the eigenvector in the standard inner product, not B-orthogonal $w_i^\top B g = 0$ as one might expect since the TRS (1) employs the B-norm.

4.1 Solution for the hard case

Theorem 4.1 *Suppose the TRS problem* (1) *belongs to the "hard case" and* p_*, λ_* *satisfy* (3)–(5) *and* $||p_*||_B = \gamma$ *with* $\lambda_* = \mu_n$. *Let* $d = \dim(\mathcal{N}(A + \lambda_* B))$ *and* $V := [v_1, \cdots, v_d]$ *be a basis of* $\mathcal{N}(A + \lambda_* B)$ *that is* B*-orthogonal, i.e.,* $V^\top B V = I$. *Then, defining*

$$H := (A + \lambda_* B + \alpha \sum_{i=1}^{d} B v_i v_i^\top B),$$

where $\alpha > 0$ *is an arbitrary positive scalar,* H *is positive definite, and hence nonsingular. Furthermore,*

$$q := -H^{-1} g = \operatorname{argmin}\{\|p\|_B | (A + \lambda_* B)p = -g\}. \tag{22}$$

that is, q *is the minimum-norm solution to the linear system* $(A + \lambda_* B)p = -g$ *in the* B*-norm, and therefore for any nonzero* $v \in \mathcal{N}(A + \lambda_* B)$ *there exists a scalar* $\eta \in \mathbb{R}$ *such that the TRS solution is* $p_* = q + \eta v$.

Proof. First, we prove that H is nonsingular. Let W be the matrix that simultaneously diagonalize A, B as in (10). Then $W^\top (A + \lambda_* B) W = \operatorname{diag}(\lambda_* - \mu_1, \cdots, \lambda_* - \mu_{n-d}, 0, \cdots, 0)$ is a diagonal matrix and $W^\top B W = I$.

We claim that defining $G = \sum_{i=1}^{d} B v_i (B v_i)^\top = B V V^\top B$ we have

$$W^\top G W = \operatorname{diag}(0, \ldots, 0, I_d).$$

To see this, writing $W = [W_1\ W_2]$ we have $(A + \lambda_* B) W_2 = (A + \lambda_* B) V = 0$ and $W_2^\top B W_2 = V^\top B V = I_d$. Since V, W_2 are of the same size, these two equalities imply that there exists an orthogonal matrix $Q \in \mathbb{R}^{d \times d}$ such that $V = W_2 Q$, and therefore

$$\begin{aligned} W^\top (B V V^\top B) W &= [W_1\ W_2]^\top B W_2 Q (W_2 Q)^\top B [W_1\ W_2] \\ &= \left([W_1\ W_2]^\top B W_2\right) W_2^\top B [W_1\ W_2] \\ &= \begin{bmatrix} O \\ I_d \end{bmatrix} \begin{bmatrix} O & I_d \end{bmatrix} = \begin{bmatrix} O & \\ & I_d \end{bmatrix}, \end{aligned}$$

where we have used the fact $W^\top B W = [W_1\ W_2]^\top B [W_1\ W_2] = I_n$.

Therefore,

$$\begin{aligned} W^\top H W &= (W^\top A W + \lambda_* I_n + \alpha \sum_{i=1}^{d} W^\top B v_i v_i^\top B W) \\ &= \operatorname{diag}(\lambda_* - \mu_1, \cdots, \lambda_* - \mu_{n-d}, \alpha I_d) \end{aligned}$$

is positive definite, so by Sylvester's law of inertia it follows that H is also positive definite, hence nonsingular.

We next show that $q := -H^{-1}g$ is a solution to the singular linear system $(A+\lambda_* B)p = -g$, which necessarily has infinitely many solutions. To see that $(A+\lambda_* B)q = -g$, note that $-g = (A+\lambda_* B)p$ implies (below I stands for I_{n-d})

$$-g = W^{-\top}\begin{bmatrix} D+\lambda_* I & \\ & 0 \end{bmatrix} W^{-1}p,$$

and also that

$$q := -H^{-1}g = -W\begin{bmatrix} D+\lambda_* I & \\ & \alpha I \end{bmatrix}^{-1} W^{\top}g$$

from which we obtain

$$\begin{aligned}
(A+\lambda_* B)q &= -W^{-\top}\begin{bmatrix} D+\lambda_* I & \\ & 0 \end{bmatrix} W^{-1}W\begin{bmatrix} D+\lambda_* I & \\ & \alpha I \end{bmatrix}^{-1} W^{\top}g \\
&= -W^{-\top}\begin{bmatrix} I & \\ & 0 \end{bmatrix} W^{\top}g \\
&= W^{-\top}\begin{bmatrix} I & \\ & 0 \end{bmatrix} W^{\top}W^{-\top}\begin{bmatrix} D+\lambda_* I & \\ & 0 \end{bmatrix} W^{-1}p \\
&= W^{-\top}\begin{bmatrix} D+\lambda_* I & \\ & 0 \end{bmatrix} W^{-1}p = -g.
\end{aligned}$$

To prove (22), note that we can write a general solution to $(A+\lambda_* B)p = -g$ as $p = q+v$ where $v \in \mathcal{N}(A+\lambda_* B)$. We shall show that $\|q\|_B \le \|q+v\|_B$ for any such v. We have

$$\|q+v\|_B^2 = (q+v)^{\top}B(q+v) = q^{\top}Bq + 2v^{\top}Bq + v^{\top}Bv.$$

Since B is positive definite the third term is nonnegative, it suffices to show that $v^{\top}Bq = 0$ for all $v \in \mathcal{N}(A+\lambda_* B)$. To see this, using $q = -H^{-1}g = H^{-1}(A+\lambda_* B)q$ we obtain

$$Bq = BH^{-1}(A+\lambda_* B)q,$$

and note that

$$BH^{-1}(A+\lambda_* B) = (A+\lambda_* B)H^{-1}B$$

which we can also verify using the decomposition with respect to W (essentially because diagonal matrices commute):

$$\begin{aligned}
BH^{-1}(A+\lambda_* B) &= W^{-\top}W^{-1}\left(W\begin{bmatrix} D+\lambda_* I & \\ & \alpha I_d \end{bmatrix}^{-1} W^{\top}\right)\left(W^{-\top}\begin{bmatrix} D+\lambda_* I & \\ & 0 \end{bmatrix} W^{-1}\right) \\
&= W^{-\top}\begin{bmatrix} I & \\ & 0 \end{bmatrix} W^{-1} \\
&= \left(W^{-\top}\begin{bmatrix} D+\lambda_* I & \\ & 0 \end{bmatrix} W^{-1}\right)\left(W\begin{bmatrix} D+\lambda_* I & \\ & \alpha I_d \end{bmatrix}^{-1} W^{\top}\right) W^{-\top}W^{-1} \\
&= (A+\lambda_* B)H^{-1}B.
\end{aligned}$$

Therefore we obtain for all $v \in \mathcal{N}(A + \lambda_* B)$

$$v^\top Bq = v^\top (BH^{-1}(A + \lambda_* B)q) = v^\top ((A + \lambda_* B)H^{-1}Bq) = (v^\top (A + \lambda_* B))H^{-1}Bq = 0,$$

as required.

Since in the hard case the solution is on the boundary with Lagrange multiplier λ_* equal to μ_n, we are able to obtain a vector $q + \eta v$ on the boundary by choosing the scalar η appropriately to obtain a global solution to the TRS satisfying (2)–(5). Put another way, if even the minimum-norm solution had norm $\|q\|_B > \gamma$ then this implies the situation was not the hard case. □

Note that, as the proof of the theorem suggests, in the hard case the TRS solution p_* is generally not unique; the goal is to find one solution, which we denote simply by p_*.

This theorem says that p_* can be computed by finding a null vector z of $A+\lambda_* B$, forming H, and solving the nonsingular linear system $Hx + g = 0$. Finding η is then an easy task of solving a scalar quadratic equation. Note that due to the low-rank term $\alpha \sum_{i=1}^{d} Bv_i v_i^\top B$, H can be dense even when A, B are sparse. Nonetheless, the linear system $Hx + g = 0$ can be solved efficiently by the CG algorithm (employing an appropriate preconditioner if available), which (as with any Krylov-type algorithm) only requires a routine for computing matrix-vector multiplications with respect to the coefficient matrix.

4.2 Detecting the hard case in our algorithm

Let us consider what will happen in the hard case. Suppose that $\lambda_* = \mu_n$ and that $Ay_2 = \lambda_* B y_2$ with $y_2^\top g = 0$. Then we see that

$$\tilde{M}(\lambda_*) \begin{bmatrix} 0 \\ y_2 \end{bmatrix} = \begin{bmatrix} -B & A + \lambda B \\ A + \lambda B & -\frac{gg^\top}{\gamma^2} \end{bmatrix} \begin{bmatrix} 0 \\ y_2 \end{bmatrix} = 0. \tag{23}$$

This means the top part of the eigenvector for $\tilde{M}(\lambda)$, which we usually extract as the solution p_*, is zero. The two conditions in the hard case is precisely the situation in which the vector that we attempt to extract becomes zero.

This suggests that we can detect the hard case by looking at the first half elements y_1 of the computed eigenvector $y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$ which we take to have unit norm $\|y\| = 1$: the hard case is when $y_1 = 0$. In practice, due to roundoff errors the computed vector $\hat{y}_1$ (here and below, quantities wearing a hat represent computed approximations) has nonzero but smalll elements in the hard case, so we need a threshold for $\|y_1\|$ below which we regard the problem as belonging to the hard case. Note that if the top part $\hat{y}_1$ is small, denoting it by ϵ, we have

$$\tilde{M}(\hat{\lambda}_*) \begin{bmatrix} \epsilon \\ \hat{y}_2 \end{bmatrix} = \begin{bmatrix} -B & A + \hat{\lambda}_* B \\ A + \hat{\lambda}_* B & -\frac{gg^\top}{\gamma^2} \end{bmatrix} \begin{bmatrix} \epsilon \\ \hat{y}_2 \end{bmatrix}. \tag{24}$$

For this to be (numerically) zero, by the first block row we have

$$(A + \hat{\lambda}_* B)\hat{y}_2 = B\epsilon.$$

Since the right-hand side is $O(\epsilon)$, it follows that $(\hat{\lambda}_*, \hat{y}_2)$ can be regarded as a numerically stable eigenpair for (A, B).

To choose an appropriate threshold we analyze the accuracy of the computed $\hat{y}_1$ as an approximation to p_*. The computed eigenvector $\hat{y} = \left[\begin{smallmatrix} \hat{y}_1 \\ \hat{y}_2 \end{smallmatrix}\right]$ is normalized to have unit norm, and has accuracy $O(\frac{\text{residual}}{\text{gap}})$ [30, Ch. 5]. Here the residual is $\|M(\hat{\lambda}_*)\hat{y}\|$, which is generally $O(u)$ with a numerically stable algorithm[2], where u is the unit roundoff, and gap is the distance between the eigenvalues: $\min_i |\lambda_* - \mu_i|$. Moreover, the loss of accuracy in extracting a vector of norm $\|y_1\|$ as a part of a unit-norm vector is a factor $O(\frac{1}{\|y_1\|})$. Overall the accuracy is estimated to be $O(\frac{1}{\|y_1\|\text{gap}})$.

On the other hand, if we treat the problem as the hard case, we need to compute the null vectors $\mathcal{N}(A + \hat{\lambda}_* B)$. Numerically these are the vectors v for which $\|(A + \hat{\lambda}_* B)v\|$ is negligible. With the tolerance τ for detecting the hard case, recalling (24), we expect the vectors we consider will have $\|(A + \hat{\lambda}_* B)v\| = O(\tau)$, suggesting this entails an error of size $O(\tau)$.

We suggest choosing the threshold τ based on which is likely to give the more accurate solution: roughly, based on the above discussion τ satisfies

$$\frac{u}{\tau \text{gap}} = \tau. \tag{25}$$

In double precision arithmetics $u \approx 10^{-16}$, and we choose τ to be about $10^{-8}\sqrt{1/\text{gap}}$. Since the gap is unknown beforehand, by default we choose to take $\tau = 10^{-4}$. A possible further improvement would be to estimate the gap within the algorithm.

5 Pseudocode

We now describe a pseudocode of our TRS algorithm in Algorithm 5.1.

We note that there can be slight modifications in the process depending on the situation. For example, we can conclude that p_0 is the (unique) optimal TRS solution if (i) it is feasible $\|p_0\|_B \leq \gamma^2$, and (ii) $A \succ O$. Therefore, if the positive definiteness of A is known or easily verifiable, then after step 1 in Algorithm 5.1, we check if $\|p_0\|_B \leq \gamma^2$ and $A \succeq O$, and if they both hold, then we can dismiss the remaining steps in Algorithm 5.1 and take p_0 as the TRS solution.

Note that the CG algorithm is originally designed for positive definite linear systems, and if CG does not converge then this implies that A is not positive definite. However, in practice CG often converges even when A is indefinite [23], so we cannot conclude that A is positive definite just because the linear system $Ap_0 = -g$ was solved by CG.

Another situation is when A is known (or easily verifiable) to be indefinite with one or more negative eigenvalues. In this case a TRS solution must lie on the boundary, and so there is no point in executing step 1; we directly proceed to step 2.

[2]To avoid unnecessary cluttering we assume $\|A\|, \|B\|$ are $O(1)$. This causes no loss of generality as we can scale the matrices without changing the TRS solution: $A \leftarrow c_1 A$ by taking $g \leftarrow c_1 g$, and $B \leftarrow c_2 B$ by taking $\gamma \leftarrow c_2 \gamma$.

Algorithm 5.1 Solve the TRS (1).

1: (Consider the case $\lambda_* = 0$) Solve $Ap_0 = -g$ by the CG algorithm, and keep p_0 if it is feasible, i.e., $\|p_0\|_B \le \gamma$.

2: Compute the rightmost eigenvalue λ_* of $\tilde{M}(\lambda)$ and an eigenvector $\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$ such that

$$\begin{bmatrix} -B & A \\ A & -\frac{gg^\top}{\gamma^2} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = -\lambda_* \begin{bmatrix} 0 & B \\ B & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}. \tag{26}$$

3: If $\|y_1\| \le \tau$ (default: $\tau = 10^{-4}$), then treat as hard case: run Algorithm 5.2 to obtain p_1.

4: Otherwise, obtain p_1 by $p_1 := -\mathrm{sign}(g^\top y_2)\gamma \frac{y_1}{\|y_1\|_B}$.

5: The solution p_* is either p_1 or p_0 (if it exists), whichever gives the smaller objective value.

As noted previously, when the matrices are large and sparse it is advisable to use a suitable eigensolver such as Arnoldi [2]. In addition to the desired eigenvalues, such algorithms generally also provide the associated eigenvectors.

The process to deal with the hard case TRS is as follows.

Algorithm 5.2 Algorithm for hard-case TRS. By default, $\tau = 10^{-4}$.

1: Compute the eigenvectors of (A, B) corresponding to λ_* (i.e., the null vectors of $A+\lambda_* B$).

2: Solve $Hq + g = 0$ for q by the CG method.

3: Take an eigenvector $v \in \mathcal{N}(A + \lambda_* B)$ computed above, and find $\eta \in \mathbb{R}$ such that $\|q + \eta v\|_B = \gamma$.

4: Return $q + \eta v$ as a candidate for the global TRS solution.

6 Comparison with existing methods

Here we compare our algorithm with previous methods and argue that ours has attractive properties in terms of efficiency and simplicity. Moreover, numerical experiments suggest that its accuracy can be notably better; see Section 7.

6.1 Efficiency

The complexity of our algorithm is $O(n^3)$ for dense A, B, and when A, B are large and sparse it is essentially the cost of an Arnoldi-type method for computing one rightmost eigenpair, which is typically in the order of a constant times the cost of a matrix-vector product, or a shifted-and-inverted linear system $(M_0 + \sigma M_1)x = b$ (for the generalized eigenproblem $\tilde{M}(\lambda) = M_0 + \lambda M_1$; the cost also depends on the separataion of eigenvalues etc). Roughly speaking, this is in the same ballpark as the cost of the existing algorithms [12, 21, 29] when $B = I$, both in the dense and large-sparse cases.

Recall that conventional algorithms solve $n \times n$ linear systems or eigenvalue problems iteratively, whereas ours solves a double-sized $2n \times 2n$ eigenproblem once. The advantage of a noniterative approach becomes significant especially when the eigenproblem scales mildly with n. For example, in the dense case where eigensolvers (and linear systems) require $O(n^3)$ cost, our approach is roughly comparable to a conventional algorithm that iterates 8 times. If the eigensolver needs $O(n^p)$ cost with $p < 3$, then this number reduces to 2^p.

Furthermore, note that our algorithm is applicable to any positive definite B without a change of variables, which makes it significantly faster when $B \neq I$, see the experiments. In other words, another advantage of our approach is that it does not require altering the algorithm completely when the situation changes from dense to large-sparse or $B = I$ to general $B \succ O$; all we need is to choose an appropriate eigensolver (e.g., `eig` or `eigs`).

Another important aspect is that it is possibly less than twice as expensive to solve the $2n \times 2n$ eigenvalue problem $\tilde{M}$ rather than the size-$(n+1)$ eigenproblem with respect to the matrix $N(\alpha) = \left[\begin{smallmatrix} \alpha & g^\top \\ g & A \end{smallmatrix}\right]$. This is because the dominant cost in the Arnoldi iteration is in matrix-vector multiplication, and multiplying the matrix (21) to a vector $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ can be done by computing $A[x_1\ x_2]$ and a vector-vector multiplication $g^\top[x_1, x_2]$. Now, computing $A[x_1, x_2]$ is usually faster than two independent matrix-vector multiplications Ax_1, Ax_2 due to the use of a higher level blas routine [11].

See the experiments to observe the practical speed, which illustrate that it is much faster than existing approaches especially when the matrices are sparse.

6.2 Ease of implementation

As mentioned before, the main feature of our approach is that the TRS can essentially (i.e. except in the hard case) be solved in one step by a generalized eigenvalue problem.

Besides the aesthetic pleasure of directly giving a solution without iterations, another advantage of our approach is its ease of implementation. For example, in MATLAB we can solve step 2 (which is the essential part of solving the generic TRS) in just four lines (in which $\tilde{M}(\lambda) = M_0 + \lambda M_1$):

```
M0 = [-B  A;A -g*g'/gamma^2];
M1 = [zeros(n) B;B zeros(n)];
lam = max(eig(M0,-M1));
p = -(A+lam*B)\g;
```

This is strikingly simple when compared with those of the existing algorithms. When A, B are large-scale and sparse, it is advised to replace `eig` with `eigs` and $\tilde{M}$ with M. Specifically, in the large-scale case, after the second line the code should be

```
[v,lam] = eigs(@(x)M0x(x),2*n,-M1,1,'lr');
p = v(n+1:end);
p = p/sqrt(p'*B*p)*gamma;
```

The last line is a normalization to force $\|p\|_B = \gamma$, because the computed eigenvector is normalized so that $\|v\|_2 = 1$. Here `M0x(x)` is a function handle that left-multiplies M_0 to the input `x`. This saves memory over storing the matrix M_0 because this way we essentially only need to store the matrices A and B, along with the vectors g, v. This way, although $M, \tilde{M}$ are of doubled size compared with A, B or $N(\alpha) = \left[\begin{smallmatrix} \alpha & g^\top \\ g & A \end{smallmatrix}\right]$, when A, B are sparse, storing $M, \tilde{M}$ requires no more storage, essentially requiring only A, B and g.

6.3 TRS as a subproblem

Our approach appears to be the first that solves the TRS exactly and is suited to the large-scale sparse case with $B \neq I$ without requiring a change of variables or outer iterations. However, its performance for efficiently solving the overall nonlinear optimization problem using TRS as a subproblem is not easily predictable, especially in view of the observation made in [12] that it is sometimes not necessary to solve the TRS to very high accuracy. While we suspect that our approach is also amenable to situations where a low accuracy would suffice (e.g. by stopping the Arnoldi iteration early), a comprehensive speed comparison with existing methods is outside the scope of this work, and in our experiments we attempt to solve TRS as accurately as numerically possible.

7 Numerical experiments

We now turn to experiments to examine the performance of the proposed methods for the hard case. All experiments were carried out in MATLAB 2013A on a Blade server machine with Xeon CPU and 64 GB memory.

We compare Algorithm 5.1 based on a generalized eigenvalue problem, shown as GEP in the figures, with the code by Fortin-Wolkowicz [8] which is based on the algorithm by Rendl and Wolkowicz [25] (shown as FRW), along with Rojas, Santos and Sorensen [27], shown as RSS. We examine the performance in the following cases: (i) $B = I$ and A is sparse, (ii) $B = I$ and A is dense, (iii) $B \neq I$ and A, B are sparse, and (iv) the hard case with $B = I$. The reason we experiment mainly with $B = I$ is that unlike ours, the other implementations do not directly handle $B \neq I$.

In each set of experiments, we ran the codes 20 times and report the average runtime and accuracy. To measure the accuracy, we have computed the relative objective function difference as follows:

$$\frac{f(\hat{p}_*) - f(\hat{p}_{\text{best}})}{|f(\hat{p}_{\text{best}})|}. \tag{27}$$

Here $\hat{p}_*$ denotes the computed solution of each method and $\hat{p}_{\text{best}}$ is the solution with the smallest objective value among the three algorithms. The reason the accuracy measure is always positive in the plots below is that the algorithm that achieves $\hat{p}_{\text{best}}$ varies from problem to problem, and we report the average of 20 runs.

When $B = I$ and A is sparse We first set $B = I$ and let A be a random sparse matrix generated by the MATLAB command `sprandsym(n,density)` with `density=1e-4`, which means the number of nonzero elements in A is about $10^{-4}n$. This construction follows the experiments in [8]. The vector g is randomly generated by `randn(n,1)`. Throughout we take $\gamma = 1$, unless otherwise specified.

We vary the matrix size n in $[10^3, 10^5]$. With the eigensolvers available today, when A, B are sparse, we expect the algorithm to be applicable to very large problems.

Figure 1 (left) shows the runtime in seconds. We see that our algorithm is significantly faster than the rest. The result indicates the method scales somewhere between $O(n)$ and $O(n^2)$. Recalling Section 6.1, this makes our non-iterative algorithm preferable, because the iterative algorithms often need to solve more than 10 eigenproblems of half the size.

The right plot of Figure 1, which shows the difference in the objective values, illustrates that our algorithm obtained solutions within about 10^{-15} of the optimal for every problem, which are accurate enough to be regarded as exact solutions in finite precision arithmetic.

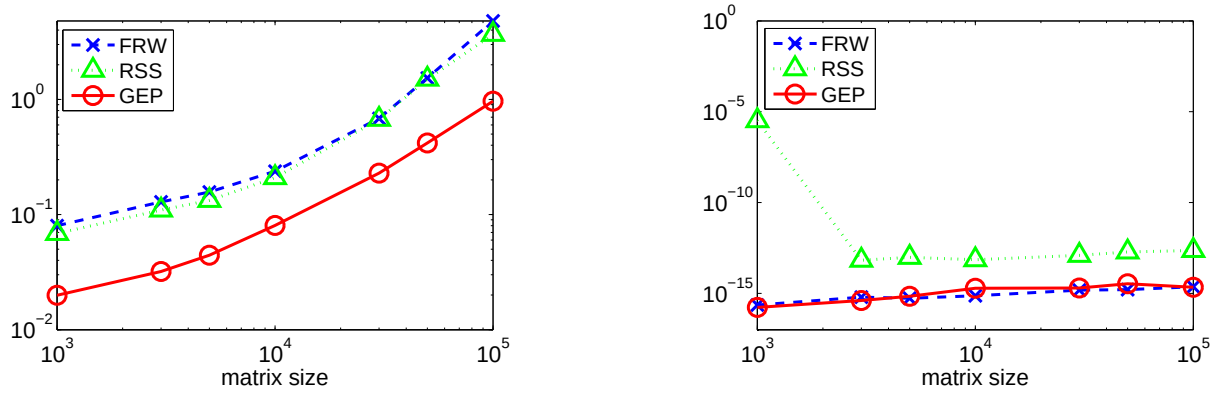

Figure 1: Runtime (left) and accuracy (right) when $B = I$ and A is sparse.

When $B = I$ and A is dense We next examine the dense case. We generate A by forming n random real numbers μ_i, then generating a random orthogonal matrix Q and setting $A = Q^\top \mathrm{diag}(\mu_i) Q$. Clearly we are limited to much smaller matrix size n than in the previous sparse case; here we take $n \leq 5000$.

The results are shown in Figure 2. The accuracy behaved much the same as in the dense case. For the speed, the difference is more benign than in the sparse case. We can explain this qualitatively as follows: in the dense case all the algorithms require $O(n^3)$ operations, and recalling the discussion in Section 6, our algorithm is expected to be fast especially when the generalized eigenproblem can be solved efficiently. In the sparse case its cost is often much less than $O(n^3)$ as we saw above, and this is why we achieve significant speedup in Figure 1. Nonetheless, our algorithm has comparable efficiency with other approaches even in the dense case.

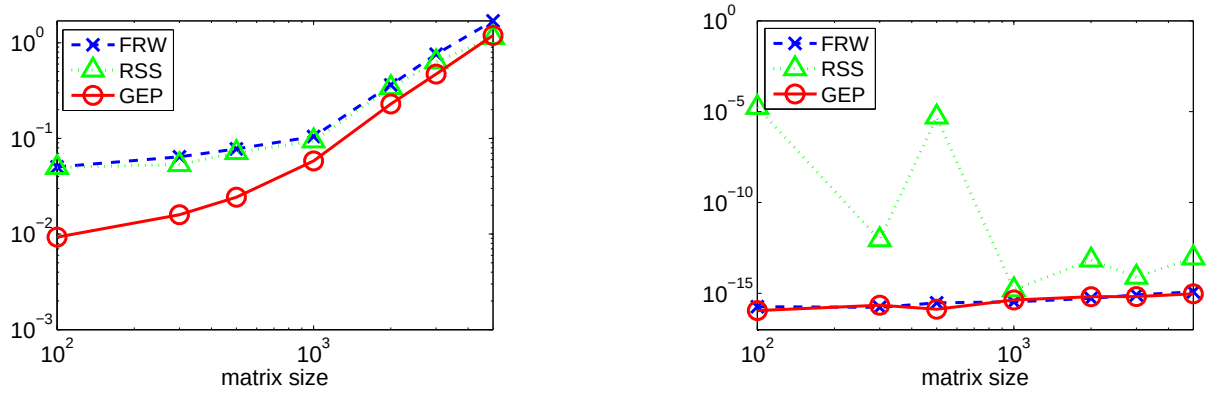

Figure 2: Runtime (left) and accuracy (right) when $B = I$ and A is dense.

When $B \succ 0$ and A is sparse We now examine problems with $B \neq I$. We let A, B be tridiagonal matrices (the Hessian A is tridiagonal if the objective function in the nonlinear optimization problem depends only on adjacent variables x_{i-1}, x_i, x_{i+1}), defined as

$$\texttt{A} = \texttt{sprandsym}(\texttt{n}, \texttt{density}), \quad \texttt{B} = \text{tridiag}(\texttt{1}, \texttt{3}, \texttt{1}), \tag{28}$$

and g is a random vector as before.

Note that the other approaches are not directly applicable when $B \neq I$; previously using a change-of-variables has been suggested. Therefore to invoke these algorithms we first compute the Cholesky factorization $B = LL^\top$, then form $L^{-1}AL^{-\top}$ and apply the codes to $A \leftarrow L^{-1}AL^{-\top}, g \leftarrow L^{-1}g$ and $B \leftarrow I$. Our algorithm does not require this; it handles the $B \neq I$ case exactly the same way as when $B = I$.

Figure 3 shows the results. In this example the Cholesky factorization and triangular substitution are taking the dominant runtime of the other algorithms, and the matrix A after the transformation is dense; these are why ours achieves a speedup of a very large factor. As before, our algorithm consistently produced accurate solutions.

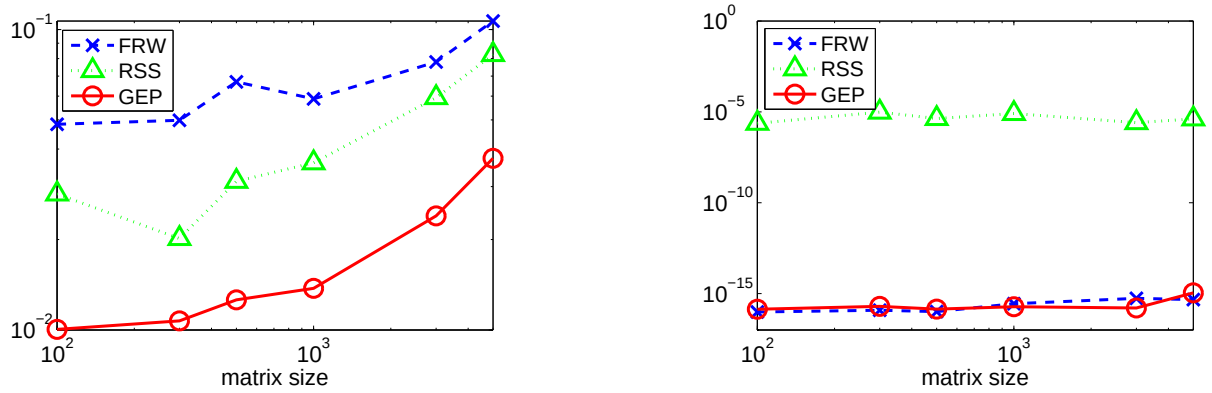

Figure 3: Runtime (left) and accuracy (right) when $B \neq I$, $B \succ 0$ and A are sparse.

Hard case We now turn to the hard case. As discussed in Section 4, the algorithm needs special treatments; just like in other algorithms. We also introduced a solution in such cases as described in Section 4.1.

We let $B = I$ and A be a random sparse matrix generated by the MATLAB command `sprandsym(n,1e-4)`. To generate a "hard case", we first set g to be a random vector and calculate the largest eigenvalue λ_* of the pencil $A + \lambda B$ with a corresponding eigenvector v. We then update g as $g \leftarrow g - (v^\top g/||v||^2)v$. To enforce the hard case we set γ to be large: $\gamma = 10^3$. Indeed we verified that all the examples thus generated belonged to the hard case.

We set the convergence criteria of the CG method as follows: the relative residual is less than 10^{-8}, or the maximum allowed number of iterations 500 is reached.

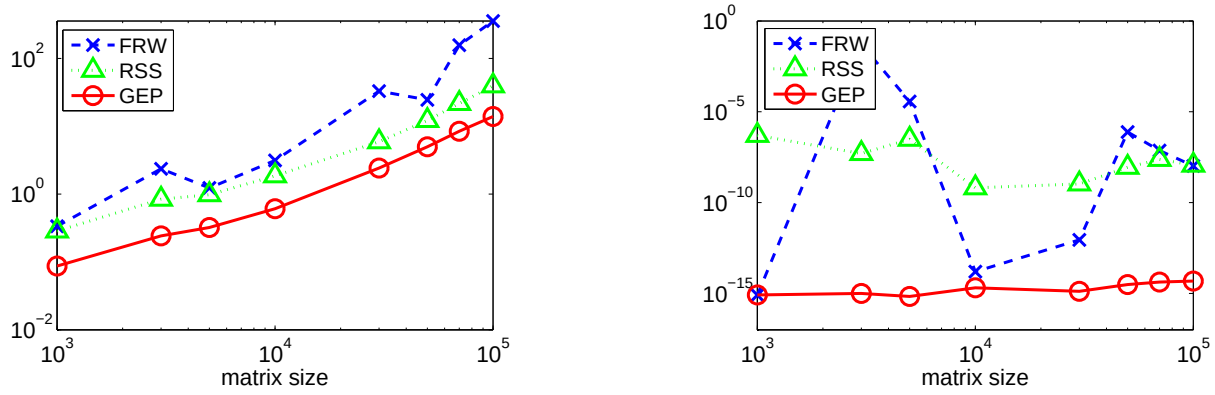

Figure 4: Runtime (left) and accuracy (right) for hard case when $B = I$ and A is sparse.

Observe that our algorithm is still the fastest, and gave the most reliable solutions.

Problems whose exact solution is known We can generate a TRS problem in the hard case with known exact solutions as follows: first, set

$$A = \mathrm{diag}(-1, 2, 3, \cdots, n), \quad g = (0, -3\alpha\gamma, 0, \cdots, 0)^\top,$$

$B = I$, $\gamma = 1$ and $\alpha = 10^{-2}$. In this case, the optimal value of TRS (1) is $m = -(1+3\alpha^2)\gamma^2/2$. We then generate an orthogonal matrix Q by the MATLAB command `qr(rand(n))` and update A, g as $A \leftarrow QAQ^\top$, $g \leftarrow Qg$, which does not change the optimal objective value.

This way we are able to examine the actual error of the algorithms, and have confirmed that our algorithm indeed computes good approximants to exact solutions, and the previous accuracy plots give reliable estimates for the errors.

Summary of experiments

The results of our experiments can be summarized as follows.

- Algorithm 5.1 based on one generalized eigenproblem is consistently reliable and its accuracy is often significantly better than other methods.

- Algorithm 5.1 is faster, especially when the matrices are sparse and/or $B \neq I$.

8 Discussion

As described in [16], the real eigenvalues of $M(\lambda)$ correspond to the KKT points for the TRS, and as discussed above, the largest eigenvalue provides a solution that minimizes the objective function $g^\top p + \frac{1}{2} p^\top A p$. In fact, more can be said: as shown in [6], the objective function value is an increasing function of λ, so we can also *maximize* the objective function by finding the *smallest* real eigenvalue of $M(\lambda)$. Further analysis of the KKT points is given in [17]. The fact that we can both maximize and minimize the objective function is perhaps unsurprising as we impose no positive definiteness assumption on A, so the objective function is nonconvex and there is no fundamental difference between minimizing and maximizing it.

Future directions include performance benchmarking on parallel systems and comparing with recent algorithms such as [13, 14, 15], and also in the context of solving nonlinear optimization problems [12]. Also worth considering are extending the eigenvalue-based approach to solve other trust-region type problems [4, 24], and dealing with a general QCQP with one constraint. We note that a related algorithm for QCQP with two constraints is developed in [28], which also mentions in its appendix an algorithm for one constraint. However, that algorithm involves the computation of all the eigenvalues, and thus has $O(n^3)$ complexity in all cases. It remains open to develop a more efficient algorithm when e.g. the matrices are sparse.

Acknowledgement

We thank Nick Gould for comments on an early draft and providing references.

References

[1] M. S. Apostolopoulou, D. G. Sotiropoulos, C. A. Botsaris, and P. Pintelas. A practical method for solving large-scale TRS. *Optimization Letters*, 5(2):207–227, 2011.

[2] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst. *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*. SIAM, Philadelphia, PA, USA, 2000.

[3] S. P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[4] S. Burer and K. M. Anstreicher. Second-order-cone constraints for extended trust-region subproblems. *SIAM J. Optim.*, 23(1):432–451, 2013.

[5] A. R. Conn, N. I. M. Gould, and P. L. Toint. *Trust Region Methods*, volume 1. SIAM, Philadelphia, PA, USA, 2000.

[6] G. E. Forsythe and G. H. Golub. On the stationary values of a second-degree polynomial on the unit sphere. *J. SIAM*, 13(4):1050–1068, 1965.

[7] C. Fortin and H. Wolkowicz. The trust region subproblem and semidefinite programming. *Optim. Method. Softw.*, 19(1):41–67, 2004.

[8] C. Fortin and H. Wolkowicz. Trust region subroutine algorithm, algorithm with links to documentation, 2010. `http://www.math.uwaterloo.ca/~hwolkowi/henry/software/trustreg.d`.

[9] W. Gander, G. H. Golub, and U. von Matt. A constrained eigenvalue problem. *Linear Algebra Appl.*, 114:815–839, 1989.

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

[11] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, 4th edition, 2012.

[12] N. I. M. Gould, S. Lucidi, M. Roma, and P. L. Toint. Solving the trust-region subproblem using the Lanczos method. *SIAM J. Optim.*, 9(2):504–525, 1999.

[13] N. I. M. Gould, D. Orban, and P. L. Toint. GALAHAD, a library of thread-safe fortran 90 packages for large-scale nonlinear optimization. *ACM Trans. Math. Soft.*, 29(4):353–372, 2003.

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

[15] E. Hazan and T. Koren. A linear-time algorithm for trust region problems. arXiv:1401.6757, 2014.

[16] S. Iwata, Y. Nakatsukasa, and A. Takeda. Global optimization methods for extended Fisher discriminant analysis. In *Proc. Seventh AISTATS, JMLR W&CP 33*, pages 411–419, 2014.

[17] S. Lucidi, L. Palagi, and M. Roma. On some properties of quadratic programs with a convex quadratic constraint. *SIAM J. Optim.*, 8(1):105–122, 1998.

[18] J. M. Martínez. Local minimizers of quadratic functions on Euclidean balls and spheres. *SIAM J. Optim.*, 4(1):159–176, 1994.

[19] J. J. Moré. *Recent developments in algorithms and software for trust region methods.* Mathematical Programming: the State of the Art, Springer, 1983.

[20] J. J. Moré. Generalizations of the trust region problem. *Optim. Method. Softw.*, 2(3-4):189–209, 1993.

[21] J. J. Moré and D. C. Sorensen. Computing a trust region step. *SIAM J. Sci. Stat. Comp.*, 4(3):553–572, 1983.

[22] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer New York, 2nd edition, 1999.

[23] C. C. Paige, B. N. Parlett, and H. A. Van der Vorst. Approximate solutions and eigenvalue bounds from Krylov subspaces. *Numer. Lin. Alg. Appl.*, 2(2):115–133, 1995.

[24] T. K. Pong and H. Wolkowicz. The generalized trust region subproblem. *Comput. Optim. Appl.*, 58(2):273–322, 2014.

[25] F. Rendl and H. Wolkowicz. A semidefinite framework for trust region subproblems with applications to large scale minimization. *Math. Prog.*, 77(1):273–299, 1997.

[26] M. Rojas, S. A. Santos, and D. C. Sorensen. A new matrix-free algorithm for the large-scale trust-region subproblem. *SIAM J. Optim.*, 11(3):611–646, 2001.

[27] 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. Soft.*, 34(2):11:1–11:28, 2008.

[28] S. Sakaue, Y. Nakatsukasa, A. Takeda, and S. Iwata. A polynomial-time algorithm for nonconvex quadratic optimization with two quadratic constraints. METR 2015-03, University of Tokyo, January 2015.

[29] D. C. Sorensen. Minimization of a large-scale quadratic function subject to a spherical constraint. *SIAM J. Optim.*, 7(1):141–161, 1997.

[30] G. W. Stewart and J.-G. Sun. *Matrix Perturbation Theory (Computer Science and Scientific Computing)*. Academic Press, 1990.

[31] P. D. Tao and L. T. H. An. A DC optimization algorithm for solving the trust-region subproblem. *SIAM J. Optim.*, 8(2):476–505, 1998.