SIAM J. SCI. COMPUT.
Vol. 40, No. 4, pp. A2005–A2032

A NESTED LANCZOS METHOD FOR THE TRUST-REGION SUBPROBLEM*

LEI-HONG ZHANG† AND CHUNGEN SHEN‡

Abstract. The trust-region subproblem (TRS) minimizes a quadratic $f(\boldsymbol{s}) = \boldsymbol{s}^{\mathrm{T}}H\boldsymbol{s}/2 + \boldsymbol{s}^{\mathrm{T}}\boldsymbol{g}$ over the ellipsoidal constraint $\|\boldsymbol{s}\|_M \leq \Delta$ for a symmetric and positive definite matrix M. For a large scale TRS, a Lanczos-type approach, namely, the generalized Lanczos trust-region (GLTR) method was introduced by Gould, Lucidi, Roma, and Toint [*SIAM J. Optim.*, 9 (1999), pp. 504–525], and extends nicely the classical Lanczos method for the eigenvalue problem to TRS. Basically, GLTR attempts to obtain a feasible approximation in the Krylov subspace $\mathcal{K}_k(M^{-1}H, M^{-1}\boldsymbol{g})$ in an efficient way. For an accurate approximation, the dimension k of $\mathcal{K}_k(M^{-1}H, M^{-1}\boldsymbol{g})$ is usually modest for a well-conditioned TRS, but can be large for ill-conditioned problems. This causes numerical difficulties in the computational costs, memory requirements, and numerical stability. This paper introduces an efficient nested restarting strategy for GLTR and resolves these numerical troubles. Convergence analysis and numerical testings are carried out to support our improvements upon GLTR.

Key words. Lanczos method, Krylov subspace, restarting, nested GMRES, trust-region subproblem, ill-conditioned

AMS subject classifications. 90C20, 90C06, 65F10, 65F15, 65K05

DOI. 10.1137/17M1145914

1. Introduction.

The main task of this paper is to develop an efficient restarting Lanczos-type method for solving the large scale trust-region subproblem (TRS) [21, 22]

$$\min_{\|\boldsymbol{s}\|_M \leq \Delta} \frac{1}{2}\boldsymbol{s}^{\mathrm{T}}H\boldsymbol{s} + \boldsymbol{s}^{\mathrm{T}}\boldsymbol{g}, \tag{1.1}$$

where $H \in \mathbb{R}^{n\times n}$ is symmetric, $\boldsymbol{g} \in \mathbb{R}^n$, $\Delta > 0$, and $\|\boldsymbol{s}\|_M = \sqrt{\boldsymbol{s}^{\mathrm{T}}M\boldsymbol{s}}$ with a symmetric and positive definite weight matrix M. TRS aims at minimizing a quadratic function $f(\boldsymbol{s}) := \frac{1}{2}\boldsymbol{s}^{\mathrm{T}}H\boldsymbol{s} + \boldsymbol{s}^{\mathrm{T}}\boldsymbol{g}$ over an ellipsoid and lies among fundamental problems in numerical optimization as well as in other practical applications. It originally was fully studied as a subproblem in the prominent trust-region method (refer to [16, 22] for a general discussion and the references therein for in-depth extensions), and also finds numerous other important applications. A full list of these applications is ever increasing; here we only mention a few noticeable applications, arising from the Lorentz eigenvalue problem [37], the Tikhonov regularization [25, 26, 27, 31, 32], the constrained eigenvalue problem [11, 13, 34], graph partitioning [19], and the Levenberg–Marquardt approach [22].

Because the model (1.1) plays a vital role in the various applications, there are a lot of efficient algorithms in the literature. In general, according to [1], algorithms

*Submitted to the journal's Methods and Algorithms for Scientific Computing section September 1, 2017; accepted for publication (in revised form) March 26, 2018; published electronically July 3, 2018.

http://www.siam.org/journals/sisc/40-4/M114591.html

Funding: The work of the first author was supported in part by the National Natural Science Foundation of China via grants NSFC-11671246, NSFC-91730303, and NSFC-11371102.

†School of Mathematics and Shanghai Key Laboratory of Financial Information Technology, Shanghai University of Finance and Economics, 777 Guoding Road, Shanghai 200433, China (zhang.leihong@mail.shufe.edu.cn).

‡College of Science, University of Shanghai for Science and Technology, Shanghai 200093, China (shenchungen@gmail.com).

for (1.1) can be grouped into three broad categories: *accurate methods* for the dense TRS [1, 21, 22], *accurate methods* for the large-sparse TRS [1, 10, 15, 17, 18, 24, 25, 26, 27, 29, 31], and *approximate methods* [16, 17, 30, 33] which aim at reducing the overall cost of solving the nonlinear minimization problem via TRS, instead of finding the exact minimizer of TRS.

As a highly efficient accurate method for small-size dense TRS (1.1), the Moré–Sorensen method [21] is one of the most widely used matrix factorization methods, and algorithms for large-scale sparse problems frequently use it or its appropriate modification as a subroutine to solve the relevant subproblem resulting from the original problem (1.1). For example, in [16], a generalized Lanczos trust-region (GLTR) method ([16, Algorithm 5.1]; see also [2, Chapter 5]) is proposed as an improved Steihaug [30]–Toint [33] truncated conjugate-gradient (tCG) iteration for (1.1).

The idea behind GLTR is the Lanczos process plus the Rayleigh–Ritz procedure. Similar to the classical Lanczos method for the eigenvalue problem (see, e.g., [23]), to handle the large-scale TRS, the original TRS (1.1) is first projected onto the Krylov subspace to form a small size TRS, which is then solved by a certain efficient accurate method. The details of this procedure are formulated in [16] (see also section 2 and [36]) and we call it the Lanczos method for TRS (LTRS). GLTR can be viewed as an efficient implementation of LTRS by relying on tCG as well as the intimate relationship between the three-term preconditioned Lanczos process and the preconditioned CG (PCG) iteration (see [16]). More recently, [36] offers a theoretical analysis on the convergence behavior of LTRS/GLTR.

Apart from a favorable numerical performance on some problems and the theoretical convergence analysis of LTRS/GLTR, there are, however, drawbacks for solving a large and ill-conditioned TRS; namely, the initial vector for generating the Krylov subspace is fixed to $M^{-1}\boldsymbol{g}$ (we shall give an explanation for this in section 2), and the Krylov subspace needs to be expanded continuously to obtain an accurate approximation. The fixed initial vector $M^{-1}\boldsymbol{g}$ induces numerical trouble when the dimension k of the Krylov subspace gets large. In particular, as k increases, not only do the overhead computational complexity and memory requirements increase, but the numerical stability (e.g., the orthogonality between Lanczos vectors) deteriorates gradually. The increase in computational costs and memory demands come from the expansion of the Lanczos vectors on the one hand, and also from the Rayleigh–Ritz procedure in solving a sequence of projected TRS's whose dimensions increase continuously, on the other hand. Breakdown in the Lanczos process can happen in theory, but the roundoff error always prevents its appearance in practice, leading to an infinitely increasing k.

In this paper we attempt to develop an efficient restarting method to alleviate the above mentioned numerical difficulties. The key for our restarting strategy is from the convergence analysis of LTRS/GLTR in [36]. Basically, we find that the convergence behavior of LTRS/GLTR is essentially the same as applying *implicitly* a Krylov subspace method to a "linear" system $(H + \lambda_{\rm opt} M)\boldsymbol{s}_{\rm opt} = -\boldsymbol{g}$, or its preconditioned "linear" system $M^{-1}(H + \lambda_{\rm opt} M)\boldsymbol{s}_{\rm opt} = -M^{-1}\boldsymbol{g}$, where $\boldsymbol{s}_{\rm opt}, \lambda_{\rm opt}$ are the global solution of TRS (1.1) and the corresponding Lagrange multiplier, respectively. Here we remark that $\lambda_{\rm opt}$ is unknown, and hence $M^{-1}(H + \lambda_{\rm opt} M)\boldsymbol{s}_{\rm opt} = -M^{-1}\boldsymbol{g}$ is not indeed a linear system, and thus traditional Krylov subspace methods cannot be applied directly. The Lanczos method LTRS/GLTR, from this point of view, can be regarded as a Krylov subspace approach that simultaneously produces the approximations for $\boldsymbol{s}_{\rm opt}$ and $\lambda_{\rm opt}$ (more detailed discussion will be presented in sections 2 and 3). The connection between LTRS/GLTR with the implicit realization for the

preconditioned system $M^{-1}(H+\lambda_{\rm opt}M)\boldsymbol{s}_{\rm opt} = -M^{-1}\boldsymbol{g}$ leads us to a nested restarting Lanczos method for TRS in this paper.

After a brief review of the TRS and LTRS, we present a convergence result (Theorem 2.2) for LTRS in section 2. The intuition of our restarting strategy follows from this convergence result and is also discussed in section 2. In section 3, we introduce our nested restarting Lanczos method for TRS and also provide some related properties. The convergence behaviors are investigated in section 4. We show in section 5 that the performance of the nested restarting Lanczos method can be improved by an initialization step using tCG. The effectiveness and the efficiency of our nested restarting Lanczos method is evaluated in section 6 in various situations, and final conclusions are drawn in section 7.

Notation. Throughout this paper, all vectors are column vectors and are typeset in bold lower case letters. The $n\times n$ identity matrix is I_n and $\boldsymbol{e}_j$ is the jth column of an identity matrix whose size is determined by the context. For any symmetric matrix C, $\lambda_i(C)$ stands for the ith smallest eigenvalue. When C is positive semidefinite (positive definite), we denote it by $C \succcurlyeq 0$ ($\succ 0$). For a given nonzero vector $\boldsymbol{x}$, the kth Krylov subspace generated by a square matrix C is $\mathcal{K}_k(C,\boldsymbol{x}) := \operatorname{span}(\boldsymbol{x}, C\boldsymbol{x}, \ldots, C^{k-1}\boldsymbol{x})$. The dimension of $\mathcal{K}_k(C,\boldsymbol{x})$ in general strictly increases by 1 after each step k, and thus often we have $\dim \mathcal{K}_k(C,\boldsymbol{x}) = k$ until $k = \chi(\boldsymbol{x})$, the *grade* of $\boldsymbol{x}$ with respect to C. $\chi(\boldsymbol{x})$ is the smallest nonnegative integer k such that $\dim \mathcal{K}_k(C,\boldsymbol{x}) = \dim \mathcal{K}_{k+1}(C,\boldsymbol{x})$; in this case, $\mathcal{K}_{\chi(\boldsymbol{x})}(C,\boldsymbol{x})$ is an invariant subspace of C.

Let $\mathbb{P}_k$ denote all polynomials with degree at most k. For the convergence of Krylov subspace methods, the Chebyshev polynomials frequently play a central role. The kth Chebyshev polynomial of the first kind $\mathscr{T}_k(t) \in \mathbb{P}_k$ is

$$\mathscr{T}_k(t) = \begin{cases} \cos(k \arccos t) & \text{for } |t| \le 1, \\ \frac{1}{2}\left(\left(t+\sqrt{t^2-1}\right)^k + \left(t+\sqrt{t^2-1}\right)^{-k}\right) & \text{for } |t| \ge 1. \end{cases}$$

2. The Lanczos method for TRS. We first briefly review the Lanczos method for solving TRS (LTRS) introduced in [16, section 5] (see also [2, 36]). LTRS basically mimics the classical Lanczos method for the eigenvalue problem, which consists of the preconditioned Lanczos process and the Rayleigh–Ritz procedure (see [23, section 11.3] and [6, Definition 7.1]). The kth step can be structured into the following three procedures:

(A) Produce the kth Krylov subspace via the preconditioned Lanczos process [23, Algorithm 4.2].
(B) Project TRS (1.1) onto the kth Krylov subspace to yield a smaller size TRS.
(C) Solve the resulting smaller size TRS to achieve an approximate of TRS (1.1).

More precisely, LTRS starts with $\boldsymbol{q}_1 = M^{-1}\boldsymbol{g}/\gamma_1$, with $\gamma_1 = \|\boldsymbol{g}\|_{M^{-1}}$, and the preconditioned Lanczos process produces an M-orthonormal basis $Q_k = [\boldsymbol{q}_1, \ldots, \boldsymbol{q}_k] \in \mathbb{R}^{n\times k}$ satisfying $Q_k^{\rm T} M Q_k = I_k$ of the kth Krylov subspace $\mathcal{K}_k(M^{-1}H, M^{-1}\boldsymbol{g})$. Note

$$\mathcal{K}_k(M^{-1}H, M^{-1}\boldsymbol{g}) = M^{-\frac{1}{2}}\mathcal{K}_k(M^{-\frac{1}{2}}HM^{-\frac{1}{2}}, M^{-\frac{1}{2}}\boldsymbol{g}). \tag{2.1}$$

Assuming $\dim \mathcal{K}_k(M^{-1}H, M^{-1}\boldsymbol{g}) = k$, the preconditioned Lanczos process partially reduces H to the tridiagonal form

$$T_k = Q_k^{\rm T} H Q_k = \begin{bmatrix} \delta_1 & \gamma_2 & & & \\ \gamma_2 & \delta_2 & \gamma_3 & & \\ & \cdot & \cdot & \cdot & \\ & & \gamma_{k-1} & \delta_{k-1} & \gamma_k \\ & & & \gamma_k & \delta_k \end{bmatrix},$$

and it bears the compact relation

$$HQ_k - MQ_kT_k = \gamma_{k+1}M\boldsymbol{q}_{k+1}\boldsymbol{e}_k^{\mathrm{T}} \quad \text{and} \quad Q_k^{\mathrm{T}}MQ_k = I_k$$

with $Q_k\boldsymbol{e}_1 = \boldsymbol{q}_1$. Projecting the original TRS (1.1) onto $\mathcal{K}_k(M^{-1}H, M^{-1}\boldsymbol{g})$ and noting $\|Q_k\boldsymbol{h}\|_M = \|\boldsymbol{h}\|_2$ for any $\boldsymbol{h} \in \mathbb{R}^k$, we have the following reduced trust-region subproblem:

$$\min_{\|\boldsymbol{h}\|_2 \le \Delta} \widehat{f}(\boldsymbol{h}) \quad \text{with} \quad \widehat{f}(\boldsymbol{h}) := \frac{1}{2}\boldsymbol{h}^{\mathrm{T}}T_k\boldsymbol{h} + \gamma_1\boldsymbol{h}^{\mathrm{T}}\boldsymbol{e}_1. \tag{2.2}$$

Let $\boldsymbol{h}_k$ be the minimizer of (2.2); it can be readily verified that

$$\boldsymbol{s}(k) := Q_k\boldsymbol{h}_k \in \mathcal{K}_k(M^{-1}H, M^{-1}\boldsymbol{g}) \quad \text{solves} \min_{\boldsymbol{s}\in\mathcal{K}_k(M^{-1}H, M^{-1}\boldsymbol{g}),\ \|\boldsymbol{s}\|_M \le \Delta} f(\boldsymbol{s}), \tag{2.3}$$

and thus naturally serves as an approximation to the global optimal solution $\boldsymbol{s}_{\mathrm{opt}}$ of (1.1).

The breakdown in the preconditioned Lanczos process happens when $k{=}\chi(M^{-1}\boldsymbol{g})$, the grade[1] of $M^{-1}\boldsymbol{g}$ with respect to $M^{-1}H$. Let $\iota := \chi(M^{-1}\boldsymbol{g})$ and the breakdown can be reflected (at least in exact arithmetic) by $\gamma_{\iota+1} = 0$ while $\gamma_k \neq 0$ for all $1 \le k \le \iota$. In such a case, it holds that $HQ_\iota = MQ_\iota T_\iota$. The breakdown for the *nondegenerate case* to be defined below implies $\boldsymbol{s}(k) = \boldsymbol{s}_{\mathrm{opt}}$, and we refer to [36, 38] for more detailed discussions on the accuracy of the approximate solution $\boldsymbol{s}(k)$ in general.

Next we give an explanation on why the initial vector $M^{-1}\boldsymbol{g}$ is fixed to generate the Krylov subspace $\mathcal{K}_k(M^{-1}H, M^{-1}\boldsymbol{g})$, and also state the underlying principle to restart the Lanczos method for TRS. To this end, we begin with an important result on TRS due to Gay [12] and Moré and Sorensen [21]:

THEOREM 2.1 (see Gay [12] and Moré and Sorensen [21]). *The vector $\boldsymbol{s}_{\mathrm{opt}}$ is a global optimal solution of the trust-region problem* (1.1) *if and only if $\boldsymbol{s}_{\mathrm{opt}}$ is feasible, i.e., $\|\boldsymbol{s}_{\mathrm{opt}}\|_M \le \Delta$, and there is a scalar $\lambda_{\mathrm{opt}} \ge 0$ such that the following conditions are satisfied:*

$$(H + \lambda_{\mathrm{opt}}M)\boldsymbol{s}_{\mathrm{opt}} = -\boldsymbol{g}, \quad \lambda_{\mathrm{opt}}(\Delta - \|\boldsymbol{s}_{\mathrm{opt}}\|_M) = 0, \quad \textit{and} \quad H + \lambda_{\mathrm{opt}}M \succcurlyeq 0.$$

Let the eigen-decomposition of $M^{-\frac{1}{2}}HM^{-\frac{1}{2}}$ be $P\Theta P^{\mathrm{T}}$ with $P^{\mathrm{T}}P = I_n$ and the ordered eigenvalues $\theta_1 = \cdots = \theta_p < \theta_{p+1} \le \cdots \le \theta_n$. Denote

$$W = [\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_n] = M^{-\frac{1}{2}}P = [M^{-\frac{1}{2}}\boldsymbol{p}_1, M^{-\frac{1}{2}}\boldsymbol{p}_2, \ldots, M^{-\frac{1}{2}}\boldsymbol{p}_n]. \tag{2.4}$$

It is true that W is M-orthogonal and $W^THW = \Theta$. Let $\mathcal{E}_1 = \mathrm{span}(W_1)$ be the invariant subspace associated with the smallest eigenvalue $\theta_1 = \cdots = \theta_p$, where $W_1 = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_p] \in \mathbb{R}^{n\times p}$ and $W = [W_1, W_2]$. There are two scenarios (e.g., [1, 18, 21, 22]) when the solution is on the boundary $\|\boldsymbol{s}_{\mathrm{opt}}\|_M = \Delta$: the *degenerate case* (or the *hard case* [1, Definition 4.2]) and the *nondegenerate case* (or the *easy case*). The former[2] happens if and only if

$$\boldsymbol{g} \perp \mathcal{E}_1 \quad \text{and} \quad \|(H - \theta_1 M)^{\dagger}\boldsymbol{g}\|_M \le \Delta. \tag{2.5}$$

[1] By (2.1), the grade $\chi(M^{-1}\boldsymbol{g})$ with respect to $M^{-1}H$ is the grade of $M^{-\frac{1}{2}}\boldsymbol{g}$ with respect to $M^{-\frac{1}{2}}HM^{-\frac{1}{2}}$.

[2] We adopt the definitions of degenerate and nondegenerate cases of [1, Definition 4.2] in this paper.

The corresponding Lagrange multiplier for the degenerate case is $\lambda_{\rm opt} = -\theta_1$ and there are multiple global solutions, all of which can be expressed as

$$\boldsymbol{s}_{\rm opt} = -(H-\theta_1 M)^{\dagger}\boldsymbol{g} + \tau \mathbf{u} \quad \forall \boldsymbol{u} \in \mathcal{E}_1, \quad \|\boldsymbol{u}\|_M = 1,$$

and $\tau = \pm\sqrt{\Delta^2 - \|(H-\theta_1 M)^{\dagger}\boldsymbol{g}\|_M^2}$. The nondegenerate case is characterized by the opposite of (2.5) with $\lambda_{\rm opt} > -\theta_1$ and $\boldsymbol{s}_{\rm opt} = -(H+\lambda_{\rm opt}M)^{-1}\boldsymbol{g}$.

Denote

$$H_{\rm opt} = H + \lambda_{\rm opt} M. \tag{2.6}$$

The optimality condition indicates $H_{\rm opt}$ is positive semidefinite and the pair $(\lambda_{\rm opt}, \boldsymbol{s}_{\rm opt})$ satisfies that system $H_{\rm opt}\boldsymbol{s}_{\rm opt} = -\boldsymbol{g}$, or equivalently, the preconditioned system

$$(M^{-1}H + \lambda_{\rm opt} I_n)\boldsymbol{s}_{\rm opt} = -M^{-1}\boldsymbol{g}. \tag{2.7}$$

This is one of the keys for the convergence bounds in Theorem 2.2 ([36, Theorem 4.11]) for $f(\boldsymbol{s}(k)) - f(\boldsymbol{s}_{\rm opt})$ and $\|\boldsymbol{s}_{\rm opt} - \boldsymbol{s}(k)\|_M$. For the special case $\lambda_{\rm opt} = 0$, the Lanczos method for TRS turns out to be the full orthogonalization method (FOM) [28, Algorithm 6.4] and the approximated solution $\boldsymbol{s}(k)$ is the same as the one obtained from PCG [28, section 6.7] on the linear system $H\boldsymbol{s}_{\rm opt} = -\boldsymbol{g}$. Consequently, the standard convergence theory [28, section 6.11.3] for CG applies for this situation. For the generic and more difficult case $\lambda_{\rm opt} \neq 0$, i.e., $\|\boldsymbol{s}_{\rm opt}\|_M = \Delta$, we can implicitly apply PCG for $H_{\rm opt}\boldsymbol{s}_{\rm opt} = -\boldsymbol{g}$ to yield the approximation $\boldsymbol{s}(k)$ of (2.3), and the convergence has been established in [36]. Theorem 2.2 summaries *a priori* error bounds for LTRS.

THEOREM 2.2 (see [36, Theorem 4.11]). *Let $W \in \mathbb{R}^{n\times n}$ be M-orthogonal such that $W^{\rm T}HW = {\rm diag}(\theta_1, \ldots, \theta_n)$ with the ith smallest eigenvalue $\theta_i = \lambda_i(M^{-\frac{1}{2}}HM^{-\frac{1}{2}})$. Suppose the preconditioned Lanczos process does not break down until $\iota = \chi(M^{-1}\boldsymbol{g})$ and let the sequence $\{\boldsymbol{s}(k)\}_{k=1}^{\iota}$ be generated by* LTRS *for* TRS (1.1).

(i) *The sequence $\{f(\boldsymbol{s}(k))\}_{k=1}^{\iota}$ is nonincreasing, and $f(\boldsymbol{s}(\iota)) = f(\boldsymbol{s}_{\rm opt})$ for the* nondegenerate *case, and*

$$f(\boldsymbol{s}(\iota)) + \frac{\tau^2\theta_1}{2} \le f(\boldsymbol{s}_{\rm opt}) \le f(\boldsymbol{s}(\iota))$$

for the degenerate *case*[3]*, where $\tau^2 = \Delta^2 - \|(H-\theta_1 M)^{\dagger}\boldsymbol{g}\|_M^2 \ge 0$ and $\theta_1 \le 0$.*

(ii) *For the* nondegenerate *case, if $\|\boldsymbol{s}_{\rm opt}\|_M = \|\boldsymbol{s}(k)\|_M = \Delta$ for some $1 \le k \le \iota$, then*

$$0 \le f(\boldsymbol{s}(k)) - f(\boldsymbol{s}_{\rm opt}) \le 2(\theta_n + \lambda_{\rm opt})\zeta_k^2,$$
$$\|\boldsymbol{s}_{\rm opt} - \boldsymbol{s}(k)\|_M \le 2\sqrt{\varkappa}\,\zeta_k, \quad \textit{where}$$

$$\varkappa = \frac{\theta_n + \lambda_{\rm opt}}{\theta_1 + \lambda_{\rm opt}}, \quad \zeta_k = \min\left\{ \frac{\Delta}{\mathscr{T}_k(\eta)}, \frac{2\|M^{-\frac{1}{2}}\boldsymbol{g}\|_2 \epsilon_k^{\rm ra}(\eta)}{\theta_n - \theta_1} \right\}, \tag{2.8}$$

$\mathscr{T}_k(x)$ is the kth degree Chebyshev polynomial of the first kind,

$$\epsilon_k^{\rm ra}(\eta) = \frac{\left(\eta + \sqrt{\eta^2-1}\right)^{1-k}}{\eta^2 - 1}, \text{ and } \eta = \frac{\varkappa+1}{\varkappa-1} > 1.$$

[3]For the degenerate case, when $\mathcal{K}_\iota(M^{-1}H, M^{-1}\boldsymbol{g})$ is an invariant subspace for $M^{-1}H$, we can choose a new vector $\boldsymbol{q}$ which is M-orthonormal to $\mathcal{K}_\iota(M^{-1}H, M^{-1}\boldsymbol{g})$, and as long as $\boldsymbol{q}$ contains components in $\mathcal{E}_1$, $\boldsymbol{s}_{\rm opt}$ can be obtained [16, Theorem 5.8] in theory.

The convergence analysis [36] for Theorem 2.2 essentially reveals that LTRS can be viewed as a certain kind of Lanczos-type method for the system $H_{\rm opt}\boldsymbol{s}_{\rm opt} = -\boldsymbol{g}$, or its preconditioned system (2.7), with an unknown coefficient matrix $H_{\rm opt}$. Once $\lambda_{\rm opt}$ is known, $\boldsymbol{s}_{\rm opt}$ can be approximated by PCG starting from any initial guess $\boldsymbol{s}_0$. Note that the underlying Krylov subspace for (2.7) is $\mathcal{K}_k\left((M^{-1}H + \lambda_{\rm opt}I_n), M^{-1}\boldsymbol{r}_0\right) = \mathcal{K}_k(M^{-1}H, M^{-1}\boldsymbol{r}_0)$, which is generated from $M^{-1}\boldsymbol{r}_0$ with the residual $\boldsymbol{r}_0 = H_{\rm opt}\boldsymbol{s}_0+\boldsymbol{g}$. In practice, however, we do not have the information of $\lambda_{\rm opt}$ at the first stage and the residual $\boldsymbol{r}_0$ indeed is unknown unless $\boldsymbol{s}_0 = \boldsymbol{0}$. Thus, the only reasonable choice of $\boldsymbol{r}_0$ to initiate the Lanczos process is $\boldsymbol{r}_0 = \boldsymbol{g}$, and this is why the starting vector $M^{-1}\boldsymbol{g}$ is used in LTRS.

Motivated by the above observation, on the other hand, we are able to design a restarting procedure for LTRS because after obtaining $\boldsymbol{s}(k)$ of (2.3) we naturally have, as a byproduct, an approximation, say $\widehat{\lambda}$, of the Lagrange multiplier $\lambda_{\rm opt}$; thus the unknown coefficient matrix $H_{\rm opt}$ can consequently be estimated by $H + \widehat{\lambda}M$. Since $\boldsymbol{s}(k)$ is generally a good approximation for $\boldsymbol{s}_{\rm opt}$, the residual $(H + \widehat{\lambda}M)\boldsymbol{s}(k) + \boldsymbol{g}$ can serve as the initial vector to generate the next Krylov subspace on which the basis steps (B) and (C) in LTRS follow similarly. We will present a restarting type procedure in the next sections.

3. The nested Lanczos method for TRS (LTRSR).

3.1. The nested GMRES method (GMRESR). Our restarting procedure for the TRS is inspired by the family of recursive GMRES methods from [7] (see also [5]), which are improvements of the generalized conjugate residual (GCR) method [8] for nonsymmetric linear systems. These methods possess a nested, "inner-loop/outer-loop" structure. In principle, any linear solver may be used in the inner loop, but when GMRES is used, the resulting algorithm is known as GMRESR.

To briefly state the framework of GMRESR, at the ith iteration, let $\boldsymbol{x}_i$ and $\boldsymbol{v}_i = \boldsymbol{b} - A\boldsymbol{x}_i$ be the current approximation and the residual, respectively; the inner and outer loops of GMRESR target at finding a good correction $\boldsymbol{u}_i$ to the associated residual system $A\boldsymbol{u} = \boldsymbol{v}_i$. Suppose $\boldsymbol{u}_i^{(0)} \approx A^{-1}\boldsymbol{v}_i$ is an approximation produced by the inner loop, and if the k-step GMRES is used, $\boldsymbol{u}_i^{(0)} = \mathcal{P}_{k,i}(A)\boldsymbol{v}_i$ for some polynomial $\mathcal{P}_{k,i}(t)$ of degree $k-1$. The restarted GMRES is simply to update $\boldsymbol{x}_i$ by $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \boldsymbol{u}_i^{(0)}$ as $\boldsymbol{u}_i^{(0)}$ is the "best" (in the sense of minimizing $\|A\boldsymbol{u}-\boldsymbol{v}_i\|_2$ if GMRES is used) over $\mathcal{K}_k(A, \boldsymbol{v}_i)$. However, this updating ignores the information obtained from the previous Krylov subspaces. This is where the outer loop comes out and the GMRESR then improves $\boldsymbol{u}_i^{(0)}$ by finding the minimizer of

$$\boldsymbol{u}_i = \arg \min_{\boldsymbol{u}\in{\rm span}(U_i)} \|\boldsymbol{v}_i - A\boldsymbol{u}\|_2$$

and updating $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \boldsymbol{u}_i$, where

$${\rm span}(U_i) = {\rm span}(\mathcal{P}_{k,0}(A)\boldsymbol{v}_0, \ldots, \mathcal{P}_{k,i}(A)\boldsymbol{v}_i) = {\rm span}(\boldsymbol{u}_0^{(0)}, \ldots, \boldsymbol{u}_i^{(0)})$$

is the subspace formed by the all previous "best" corrections $\mathcal{P}_{k,j}(A)\boldsymbol{v}_j$ associated with $A\boldsymbol{u} = \boldsymbol{v}_j$ over the corresponding Krylov subspaces $\mathcal{K}_k(A, \boldsymbol{v}_j)$, $j = 0, 1, \ldots, i$. This procedure of GMRESR [7] can be simply stated as Algorithm 3.1.

Remark 3.1. There are four remarks for Algorithm 3.1:

(1) If $\mathcal{P}_{k,i}(t) = 1$ for all i, then $\boldsymbol{x}_i$ is the same as that resulting from GCR and GMRES.

Algorithm 3.1. The framework of GMRESR in [7].

1: Choose $\boldsymbol{x}_0, \epsilon > 0$ and k and let $\boldsymbol{v}_0 = \boldsymbol{b} - A\boldsymbol{x}_0, i = 0, U_{-1} = [\ ]$;
2: **while** $\|\boldsymbol{v}_i\|_2 > \epsilon$ **do**
3: Apply k steps GMRES for $A\boldsymbol{u} = \boldsymbol{v}_i$ to obtain an approximation $\boldsymbol{u}_i^{(0)}$ of $A^{-1}\boldsymbol{v}_i$ (or other suitable approximations for $A^{-1}\boldsymbol{v}_i$); i.e.,

$$\boldsymbol{u}_i^{(0)} = \mathcal{P}_{k,i}(A)\boldsymbol{v}_i = \arg \min_{\boldsymbol{u} \in \mathcal{K}_k(A,\boldsymbol{v}_i)} \|\boldsymbol{v}_i - A\boldsymbol{u}\|_2;$$

4: Let $U_i = [U_{i-1}, \boldsymbol{u}_i^{(0)}]$;
5: Solve

$$\boldsymbol{u}_i = \arg \min_{\boldsymbol{u} \in \mathrm{span}(U_i)} \|\boldsymbol{v}_i - A\boldsymbol{u}\|_2;$$

6: Update $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \boldsymbol{u}_i,\ \boldsymbol{v}_{i+1} = \boldsymbol{v}_i - A\boldsymbol{u}_i$ and $i = i + 1$;
7: **end while**

(2) If U_i in step 4 is replaced by $U_i = \boldsymbol{u}_i^{(0)}$, then it reduces to the restarted GMRES (i.e., GMRES(k): restart for every k steps).
(3) The minimization problems in steps 3 and 5 can be equivalently stated as

$$\boldsymbol{x}_i + \boldsymbol{u}_i^{(0)} = \boldsymbol{x}_i + \mathcal{P}_{k,i}(A)\boldsymbol{v}_i = \arg \min_{\boldsymbol{x} \in \boldsymbol{x}_i + \mathcal{K}_k(A,\boldsymbol{v}_i)} \|\boldsymbol{b} - A\boldsymbol{x}\|_2 \quad \text{and}$$

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \boldsymbol{u}_i = \arg \min_{\boldsymbol{x} \in \boldsymbol{x}_i + \mathrm{span}(U_i)} \|\boldsymbol{b} - A\boldsymbol{x}\|_2, \quad \text{respectively.}$$

(4) For the efficient implementation of GMRESR [7], an $A^{\mathrm{T}}A$-orthogonal basis for $\mathrm{span}(U_i)$ is formed via the modified Gram–Schmidt process, and the updating for $\boldsymbol{x}_i$ in outer-loop iteration at steps 5 and 6 can be executed efficiently.

3.2. A nested LTRS (LTRSR). Following GMRESR in Algorithm 3.1, we will derive a nested Lanczos method for TRS. In order to put our final algorithm into a recursive inner-and-outer-loop fashion, we change the degree k (and all other quantities involved in the first Krylov subspace (2.1) appearing in section 2) of the Krylov subspace in (2.3) to k_1. Also, we use $\boldsymbol{s}_1^{(0)}$ and $\lambda_1^{(0)}$ to present the solution $\boldsymbol{s}(k)$ and the Lagrange multiplier to (2.3), respectively, where the subscript 1 denotes the first outer-loop iteration, and (0) means that the solution has not been refined in the nested structure to be defined below.

With $\boldsymbol{s}_0 = \boldsymbol{0}$ and $\lambda_0 = 0$, we suppose $(\boldsymbol{s}_i, \lambda_i)$ is an approximate pair of $(\boldsymbol{s}_{\mathrm{opt}}, \lambda_{\mathrm{opt}})$ after the ith outer-loop iteration, and the corresponding residual is $\boldsymbol{r}_i = H\boldsymbol{s}_i + \lambda_i M\boldsymbol{s}_i + \boldsymbol{g}$. We will first take a look at the correction system. By comparing the optimality $H\boldsymbol{s}_{\mathrm{opt}} + \lambda_{\mathrm{opt}} M\boldsymbol{s}_{\mathrm{opt}} + \boldsymbol{g} = \boldsymbol{0}$, we have

$$H_{\mathrm{opt}}\boldsymbol{u} + \boldsymbol{r}_i + \delta\lambda \cdot M\boldsymbol{s}_i = \boldsymbol{0} \quad \text{or} \quad M^{-1}H_{\mathrm{opt}}\boldsymbol{u} + M^{-1}\boldsymbol{r}_i + \delta\lambda \cdot \boldsymbol{s}_i = \boldsymbol{0}, \tag{3.1}$$

where $\boldsymbol{u} = \boldsymbol{s}_{\mathrm{opt}} - \boldsymbol{s}_i$ and $\delta\lambda = \lambda_{\mathrm{opt}} - \lambda_i$. It should be noticed that, different from GMRESR, the correction system (3.1) is nonlinear, from which we have

$$\boldsymbol{u} = -(M^{-1}H_{\mathrm{opt}})^{-1}M^{-1}\boldsymbol{r}_i - \delta\lambda \cdot (M^{-1}H_{\mathrm{opt}})^{-1}\boldsymbol{s}_i;$$

if we approximate the terms $(M^{-1}H_{\mathrm{opt}})^{-1}M^{-1}\boldsymbol{r}_i$ and $\delta\lambda \cdot (M^{-1}H_{\mathrm{opt}})^{-1}\boldsymbol{s}_i$ using the Krylov subspace based methods and noting $\mathcal{K}_k(M^{-1}H_{\mathrm{opt}}, \boldsymbol{x}) = \mathcal{K}_k(M^{-1}H, \boldsymbol{x})$ for any $\boldsymbol{x}$, we have

$$\begin{aligned}\boldsymbol{u} &= -(M^{-1}H_{\text{opt}})^{-1}M^{-1}\boldsymbol{r}_i - \delta\lambda\cdot(M^{-1}H_{\text{opt}})^{-1}\boldsymbol{s}_i\\ &\approx \mathcal{P}(M^{-1}H_{\text{opt}})M^{-1}\boldsymbol{r}_i + \widehat{\mathcal{P}}(M^{-1}H_{\text{opt}})\boldsymbol{s}_i\\ &\in \mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i) + \mathcal{K}_{m_{i+1}}(M^{-1}H, \boldsymbol{s}_i)\end{aligned}$$

for some polynomials $\mathcal{P}$ and $\widehat{\mathcal{P}}$ of degrees $k_{i+1}-1$ and $m_{i+1}-1$, respectively. This suggests that a good correction $\boldsymbol{u}$ for the current $\boldsymbol{s}_i$ should be in

$$\text{span}(U_i^{(0)}) := \mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i) + \mathcal{K}_{m_{i+1}}(M^{-1}H, \boldsymbol{s}_i), \tag{3.2}$$

where $U_i^{(0)}$ is a basis of $\mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i) + \mathcal{K}_{m_{i+1}}(M^{-1}H, \boldsymbol{s}_i)$. The inner-loop iteration terminates with $\boldsymbol{s}_{i+1}^{(0)} = \boldsymbol{s}_i + \boldsymbol{u}_i^{(0)}$, where $\boldsymbol{u}_i^{(0)}$ is obtained from

$$\boldsymbol{s}_{i+1}^{(0)} = \boldsymbol{s}_i + \boldsymbol{u}_i^{(0)} = \arg \min_{\substack{\boldsymbol{s}\in\boldsymbol{s}_i+\mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i)+\mathcal{K}_{m_{i+1}}(M^{-1}H,\boldsymbol{s}_i)\\ \|\boldsymbol{s}\|_M\le\Delta}} f(\boldsymbol{s}).$$

Analogous to GMRESR, we cast the ith outer-loop iteration as minimizing $f(\boldsymbol{s})$ over the subspace spanned by the previous $\boldsymbol{u}_j^{(0)}$ for $j = 0, 1, \ldots, i$. This nested procedure is summarized in Algorithm 3.2.

Algorithm 3.2. The framework of the nested Lanczos method for TRS (LTRSR).

1: Choose $\epsilon > 0$ and let $\boldsymbol{s}_0 = \boldsymbol{0}$, $\boldsymbol{r}_0 = -\boldsymbol{g}$, $i = 0, U_{-1} = [\,], k_0 = 0$;
2: **while** $\|\boldsymbol{r}_i\|_{M^{-1}} > \epsilon$ and $i < i_{\max}$ **do**
3: Obtain the solution $\boldsymbol{s}_{i+1}^{(0)}$ and the associated Lagrange multiplier $\lambda_{i+1}^{(0)}$ from

$$\boldsymbol{s}_{i+1}^{(0)} = \boldsymbol{s}_i + \boldsymbol{u}_i^{(0)} := \arg \min_{\substack{\boldsymbol{s}\in\boldsymbol{s}_i+\mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i)+\mathcal{K}_{m_{i+1}}(M^{-1}H,\boldsymbol{s}_i)\\ \|\boldsymbol{s}\|_M\le\Delta}} f(\boldsymbol{s}); \tag{3.3}$$

4: Let $U_i = [U_{i-1}, \boldsymbol{u}_i^{(0)}]$;
5: Obtain the solution $\boldsymbol{s}_{i+1}$ and the associated Lagrange multiplier λ_{i+1} of

$$\boldsymbol{s}_{i+1} = \boldsymbol{s}_i + \boldsymbol{u}_i := \arg \min_{\boldsymbol{s}\in\boldsymbol{s}_i+\text{span}(U_i),\ \|\boldsymbol{s}\|_M\le\Delta} f(\boldsymbol{s}); \tag{3.4}$$

6: Update $\boldsymbol{r}_{i+1} = H\boldsymbol{s}_{i+1} + \lambda_{i+1}M\boldsymbol{s}_{i+1} + \boldsymbol{g}$ and $i = i+1$;
7: **end while**

3.3. Implementation details. In this subsection, we shall give the implementation details involved in Algorithm 3.2.

First, note that both (3.3) and (3.4) can be restated as

$$\arg \min_{\boldsymbol{s}\in\boldsymbol{s}_i+\mathcal{U},\ \|\boldsymbol{s}\|_M\le\Delta} f(\boldsymbol{s}) \tag{3.5}$$

for some subspace $\mathcal{U}$. Suppose $\dim\mathcal{U} = \zeta$ and $U \in \mathbb{R}^{n\times\zeta}$ is an M-orthonormal basis for $\mathcal{U}$; thus the next approximation say $\widetilde{\boldsymbol{s}}$ can be expressed as $\widetilde{\boldsymbol{s}} = \boldsymbol{s}_i + U\boldsymbol{h}$, $\boldsymbol{h} \in \mathbb{R}^\zeta$. Note that

$$f(\widetilde{\boldsymbol{s}}) = \frac{1}{2}\boldsymbol{h}^{\mathrm{T}}U^{\mathrm{T}}HU\boldsymbol{h} + \boldsymbol{h}^{\mathrm{T}}U^{\mathrm{T}}(H\boldsymbol{s}_i + \boldsymbol{g}) + f(\boldsymbol{s}_i) = \frac{1}{2}\boldsymbol{h}^{\mathrm{T}}T\boldsymbol{h} + \boldsymbol{h}^{\mathrm{T}}\boldsymbol{c}_i + f(\boldsymbol{s}_i),$$

where $T = U^{\mathrm{T}}HU$ and $\boldsymbol{c}_i = U^{\mathrm{T}}(H\boldsymbol{s}_i + \boldsymbol{g})$; furthermore,

$$\|\boldsymbol{s}_i + U\boldsymbol{h}\|_M^2 = \|\boldsymbol{h}\|_2^2 + 2\boldsymbol{h}^{\mathrm{T}}U^{\mathrm{T}}M\boldsymbol{s}_i + \|\boldsymbol{s}_i\|_M^2 = \|\boldsymbol{h}\|_2^2 + 2\boldsymbol{h}^{\mathrm{T}}\boldsymbol{y}_i + \|\boldsymbol{s}_i\|_M^2,$$

where $\boldsymbol{y}_i = U^{\mathrm{T}}M\boldsymbol{s}_i$. Thus the optimal correction $U\boldsymbol{h} \in \mathcal{U}$ in (3.5) can be achieved by

$$\min_{\|\boldsymbol{h}\|_2^2 + 2\boldsymbol{h}^{\mathrm{T}}\boldsymbol{y}_i \le \Delta^2 - \|\boldsymbol{s}_i\|_M^2} \frac{1}{2}\boldsymbol{h}^{\mathrm{T}}T\boldsymbol{h} + \boldsymbol{h}^{\mathrm{T}}\boldsymbol{c}_i$$

which, by denoting $\boldsymbol{z} = \boldsymbol{h} + \boldsymbol{y}_i$ and $\boldsymbol{m}_i = \boldsymbol{c}_i - T\boldsymbol{y}_i$, is equivalent to

$$\min_{\|\boldsymbol{z}\|_2 \le \Delta_{i+1}} \frac{1}{2}\boldsymbol{z}^{\mathrm{T}}T\boldsymbol{z} + \boldsymbol{z}^{\mathrm{T}}\boldsymbol{m}_i, \quad \text{where} \quad \Delta_{i+1} = \sqrt{\Delta^2 - \|\boldsymbol{s}_i\|_M^2 + \|\boldsymbol{y}_i\|_2^2}. \tag{3.6}$$

Let $\boldsymbol{z}_i$ and $\widetilde{\lambda}$ be solution and the Lagrange multiplier of (3.6), respectively; then $\widetilde{\boldsymbol{s}} = \boldsymbol{s}_i + U(\boldsymbol{z}_i - \boldsymbol{y}_i)$ solves (3.5). Note that (3.6) is also a smaller size TRS and the efficiency of the final restarting Lanczos method can benefit from the full horsepower of any sophisticated solver (for example, the Moré–Sorensen method [21] or the eigenvalue-based solver [1]) suitable for small- to medium-size TRSs.

We next discuss the procedure of finding the M-orthogonal basis $U_i^{(0)}$ in (3.2) and $\mathrm{span}(U_i)$, respectively. In general, one can call the block Lanczos process (see, e.g., [3, 14]) to generate the orthogonal basis for $\mathcal{K}_{k_{i+1}}(M^{-1}H, [M^{-1}\boldsymbol{r}_i, \boldsymbol{s}_i])$ when $k_{i+1} = m_{i+1}$. Alternatively, we can obtain an orthogonal basis by first carrying out the standard preconditioned Lanczos process [23, Algorithm 4.2] to obtain an M-orthogonal basis for $\mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i)$, and then applying the Gram–Schmidt method process to expand it to an M-orthogonal basis $U_i^{(0)}$ for $\mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i) + \mathcal{K}_{m_{i+1}}(M^{-1}H, \boldsymbol{s}_i)$. This is stated in Algorithm 3.3. Note that when the breakdown in the Lanczos process in step 1 occurs before k_{i+1}, then $\ell = \chi(M^{-1}\boldsymbol{r}_i)$, i.e., the grade of $M^{-1}\boldsymbol{r}_i$ with respect to $M^{-1}H$; otherwise $\ell = k_{i+1}$. Also, when taking the roundoff errors into consideration, the modified Gram–Schmidt and/or the full re-orthogonalization [6, Algorithm 7.2] should be incorporated in steps 4 and 6 to keep the M-orthogonalization numerically. The modified Gram–Schmidt process can be used to have an M-orthogonal basis U_i for $\mathrm{span}(U_i)$ numerically.

We last mention the primary computational and memory requirements of LTRSR. For (3.3) in the inner loop, it first uses about $k_{i+1} + m_{i+1}$ times of H-vector products

Algorithm 3.3. A process to form an M-orthogonal basis $U_i^{(0)}$ for $\mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i) + \mathcal{K}_{m_{i+1}}(M^{-1}H, \boldsymbol{s}_i)$.

1: Obtain an M-orthogonal basis $U \in \mathbb{R}^{n\times\ell}$ for $\mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i)$ by the Lanczos process [23, Algorithm 4.2];
2: **for** $j = 1, 2, \ldots, m_{i+1}$ **do**
3: **if** $j = 1$ **then**
4: $\boldsymbol{p} = \boldsymbol{s}_i - UU^{\mathrm{T}}M\boldsymbol{s}_i$
5: **else**
6: $\boldsymbol{p} = M^{-1}HU(:, j+\ell-1) - U(U^{\mathrm{T}}HU(:, j+\ell-1))$
7: **end if**
8: $\gamma = \|\boldsymbol{p}\|_M$; If $\gamma = 0$ then **break**
9: $U = [U, \boldsymbol{p}/\gamma]$
10: **end for**
11: $U_i^{(0)} = U$

and $k_{i+1}+m_{i+1}$ times of M^{-1}-vector products (i.e., solving linear systems with the coefficient matrix M) to form the basis $U_i^{(0)}$, and then costs $O((k_{i+1}+m_{i+1})^3)$ flops to solve (3.3). For (3.4), it calls a procedure to M-orthogonalize $\boldsymbol{u}_i^{(0)}$ against U_{i-1} and then costs $O(i^3)$ flops to obtain $\boldsymbol{s}_{i+1}$. Moreover, we can restrict the number of columns of U_i to ς by discarding the first columns to control the size for (3.4). The main memory requirements are storages of $U_i^{(0)}$ and U_i, which can be kept modest by controlling $k_{i+1}+m_{i+1}$ and ς.

4. Convergence of the LTRSR. We now explore some convergence behavior of the LTRSR in Algorithm 3.2. We begin with the investigation of the breakdown behavior of LTRSR.

4.1. Breakdown in LTRSR.

THEOREM 4.1. *For $i = 0, 1, \ldots$ until convergence in Algorithm* 3.2 *and* $\min(k_i, m_i) \geq 1$,

(i) $f(\boldsymbol{s}_i) \geq f(\boldsymbol{s}_{i+1}^{(0)}) \geq f(\boldsymbol{s}_{i+1})$;

(ii) *as the optimality conditions for* (3.3) *and* (3.4)*, respectively, it holds that*

$$\boldsymbol{r}_{i+1}^{(0)} \perp \mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i) + \mathcal{K}_{m_{i+1}}(M^{-1}H, \boldsymbol{s}_i) \quad \text{and} \quad \boldsymbol{r}_{i+1} \perp \operatorname{span}(U_i),$$

where $\boldsymbol{r}_{i+1}^{(0)} = H\boldsymbol{s}_{i+1}^{(0)} + \lambda_{i+1}^{(0)} M\boldsymbol{s}_{i+1}^{(0)} + \boldsymbol{g}$ and $\boldsymbol{r}_{i+1}$ is given by step 6;

(iii) *if $\boldsymbol{r}_i \neq \boldsymbol{0}$, then $f(\boldsymbol{s}_i) > f(\boldsymbol{s}_{i+1}^{(0)})$; in other words, the ith inner-loop iteration will always improve the objective function whenever $\boldsymbol{r}_i \neq \boldsymbol{0}$.*

Proof. The assertion (i) is trivial. The conclusion (ii) is indeed the first-order necessary condition of (3.3) and (3.4), respectively, and can be verified by, e.g., [35, Theorem 4.1]. For (iii), we show that $f(\boldsymbol{s}_i) = f(\boldsymbol{s}_{i+1}^{(0)})$ implies $\boldsymbol{r}_i = \boldsymbol{0}$. To this end, we note that $f(\boldsymbol{s}_i) = f(\boldsymbol{s}_{i+1}^{(0)})$ indicates that $\boldsymbol{s}_i$ and λ_i are the solution and the corresponding Lagrange multiplier of (3.3), respectively. Thus, by (ii), we have

$$\boldsymbol{r}_i = (H + \lambda_i M)\boldsymbol{s}_i + \boldsymbol{g} \perp \mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i) + \mathcal{K}_{m_{i+1}}(M^{-1}H, \boldsymbol{s}_i),$$

which by $M^{-1}\boldsymbol{r}_i \in \mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i)$ implies $\boldsymbol{r}_i^{\mathrm{T}} M^{-1}\boldsymbol{r}_i = 0$ leading to $\boldsymbol{r}_i = \boldsymbol{0}$. ∎

We next consider the breakdown in the inner-loop iteration.

LEMMA 4.2. *Suppose $\boldsymbol{g} \not\perp \mathcal{E}_1$ (implying that the original TRS* (1.1) *is nondegenerate) and at the ith iteration of Algorithm* 3.2, $\min(k_{i+1}, m_{i+1}) \geq 1$. *Then $\mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i) + \mathcal{K}_{m_{i+1}}(M^{-1}H, \boldsymbol{s}_i)$ is not orthogonal to $\mathcal{E}_1$.*

Proof. In (2.4), $\mathcal{E}_1 = \operatorname{span}(W_1)$ with $W_1 = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_p]$ and $W = [W_1, W_2]$. The assertion is true if $\boldsymbol{s}_i$ has a component in $\mathcal{E}_1$. When $\boldsymbol{s}_i \in \operatorname{span}(W_2)$, we have $M^{-1}\boldsymbol{g} = WW^{\mathrm{T}}\boldsymbol{g} = W_1(W_1^{\mathrm{T}}\boldsymbol{g}) + W_2(W_2^{\mathrm{T}}\boldsymbol{g})$ with $W_1^{\mathrm{T}}\boldsymbol{g} \neq 0$ by assumption. Since both $\mathcal{E}_1$ and $\operatorname{span}(W_2)$ are invariant subspaces of $M^{-1}H + \lambda_i I_n$, we know that $M^{-1}\boldsymbol{r}_i = W_1(W_1^{\mathrm{T}}\boldsymbol{g}) + (M^{-1}H + \lambda_i I_n)\boldsymbol{s}_i + W_2(W_2^{\mathrm{T}}\boldsymbol{g})$ contains a nonzero component $W_1(W_1^{\mathrm{T}}\boldsymbol{g})$ in $\mathcal{E}_1$. ∎

THEOREM 4.3. *Under the assumptions of Lemma* 4.2 *and if $\mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i) + \mathcal{K}_{m_{i+1}}(M^{-1}H, \boldsymbol{s}_i)$ is an invariant subspace of $M^{-1}H$ additionally, then $\boldsymbol{s}_{i+1}^{(0)} = \boldsymbol{s}_{\mathrm{opt}}$.*

Proof. Let $U_i^{(0)}$ be an M-orthogonal basis of $\mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i) + \mathcal{K}_{m_{i+1}}(M^{-1}H, \boldsymbol{s}_i)$. According to Lemma 4.2, if $\operatorname{span}(U_i^{(0)})$ is an invariant subspace of

$M^{-1}H$, then all the eigenvalues of $T = (U_i^{(0)})^{\mathrm{T}} H U_i^{(0)}$ are the eigenvalues of $M^{-1}H$, and the smallest one is θ_1.

Consider the ith inner-loop subproblem (3.3) and the associated problem (3.6). With $\boldsymbol{s}_{i+1}^{(0)} = \boldsymbol{s}_i + U_i^{(0)}\boldsymbol{h}_i$ for some $\boldsymbol{h}_i$, it holds that

$$\begin{aligned} M^{-1}\boldsymbol{r}_{i+1}^{(0)} &= (M^{-1}H + \lambda_{i+1}^{(0)} I_n)\boldsymbol{s}_{i+1}^{(0)} + M^{-1}\boldsymbol{g} \\ &= (M^{-1}H + \lambda_{i+1}^{(0)} I_n)(\boldsymbol{s}_i + U_i^{(0)}\boldsymbol{h}_i) + M^{-1}\boldsymbol{g} \\ &= M^{-1}\boldsymbol{r}_i + (\lambda_{i+1}^{(0)} - \lambda_i)\boldsymbol{s}_i + (M^{-1}H + \lambda_{i+1}^{(0)} I_n) U_i^{(0)}\boldsymbol{h}_i \in \mathrm{span}(U_i^{(0)}), \end{aligned}$$

where the last relation follows because $\mathrm{span}(U_i^{(0)})$ is an invariant subspace of $M^{-1}H$. Furthermore, by (iii) of Theorem 4.1, we know that $\boldsymbol{r}_{i+1}^{(0)} \perp U_i^{(0)}$, which together with the above relation $M^{-1}\boldsymbol{r}_{i+1}^{(0)} \in \mathrm{span}(U_i^{(0)})$ leads to $(\boldsymbol{r}_{i+1}^{(0)})^{\mathrm{T}} M^{-1}\boldsymbol{r}_{i+1}^{(0)} = 0$ and thus $\boldsymbol{r}_{i+1}^{(0)} = \boldsymbol{0}$. Moreover, according to Theorem 2.1 and the associated problem (3.6), the global optimality conditions ensure that $\lambda_{i+1}^{(0)} \geq 0$, $\lambda_{i+1}^{(0)}(\|\boldsymbol{s}_{i+1}^{(0)}\|_M - \Delta) = 0$ and $T + \lambda_{i+1}^{(0)} I \succcurlyeq 0$ implying $\lambda_1(T) + \lambda_{i+1}^{(0)} = \theta_1 + \lambda_{i+1}^{(0)} \geq 0$, and therefore $H + \lambda_{i+1}^{(0)} M \succcurlyeq 0$. Consequently, we know that all the conditions in Theorem 2.1 hold, and $\boldsymbol{s}_{i+1}^{(0)}$ and $\lambda_{i+1}^{(0)}$ are the solution and the corresponding Lagrange multiplier of the original problem (1.1), respectively. ∎

Remark 4.1.

(1) Note that for any i, $\boldsymbol{s}_i = \mathcal{P}(M^{-1}H)M^{-1}\boldsymbol{g}$ and $\boldsymbol{r}_i = \widetilde{\mathcal{P}}(M^{-1}H)M^{-1}\boldsymbol{g}$ for some polynomials $\mathcal{P}$ and $\widetilde{\mathcal{P}}$, respectively. Thus $\max(\chi(M^{-1}\boldsymbol{r}_i), \chi(\boldsymbol{s}_i)) \leq \chi(M^{-1}\boldsymbol{g}) = \iota$ and

$$\mathrm{span}(U_i^{(0)}) = \mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i) + \mathcal{K}_{m_{i+1}}(M^{-1}H, \boldsymbol{s}_i) \subseteq \mathcal{K}_{\iota}(M^{-1}H, M^{-1}\boldsymbol{g}).$$

Therefore for the case $\boldsymbol{g} \not\perp \mathcal{E}_1$, if $\dim(\mathrm{span}(U_i^{(0)})) = \chi(M^{-1}\boldsymbol{g})$, $\boldsymbol{s}_{\mathrm{opt}}$ is attained at the ith iteration. In general, it is hoped that $\boldsymbol{s}_{\mathrm{opt}}$ (or a highly accurate approximation) is obtained with $\dim(\mathrm{span}(U_i^{(0)})) \ll \chi(M^{-1}\boldsymbol{g})$. By (iii) of Theorem 4.1, we know that the outer-loop iteration serves as the refinement procedure so that the next inner-loop iteration returns $\boldsymbol{s}_{\mathrm{opt}}$ (or a satisfactory approximation) with $\max(m_{i+1}, k_{i+1}) \ll \chi(M^{-1}\boldsymbol{g})$.

(2) Theorem 4.3 also partially shows the advantage in using the affine space

$$\boldsymbol{s}_i + \mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i) + \mathcal{K}_{m_{i+1}}(M^{-1}H, \boldsymbol{s}_i) \tag{4.1}$$

over another choice $\boldsymbol{s}_i + \mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i)$ as the constraint for the inner-loop iteration in step 3 of Algorithm 3.2: When $\boldsymbol{s}_i$ is not $\boldsymbol{s}_{\mathrm{opt}}$, it is possible that $M^{-1}\boldsymbol{r}_i \in \mathrm{span}(W_2)$; thus, even if $\mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i) = \mathrm{span}(Q_{k_{i+1}})$ is an invariant subspace of $M^{-1}H$, the smallest eigenvalue of $T = Q_{k_{i+1}}^{\mathrm{T}} H Q_{k_{i+1}}$ is strictly larger than θ_1 and therefore $H + \lambda_{i+1}^{(0)} M$ is indefinite; in other words, for $\boldsymbol{g} \not\perp \mathcal{E}_1$, the breakdown in $\mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i)$ does not necessarily imply that $\boldsymbol{s}_{\mathrm{opt}}$ is attained. The other advantage of using (4.1) as the constraint for the inner-loop subproblem (3.3) will be revealed in Theorems 4.6 and 4.7. Numerical testing will also be carried out in section 6.2 to illustrate the importance of this double Krylov subspace $\mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i) + \mathcal{K}_{m_{i+1}}(M^{-1}H, \boldsymbol{s}_i)$ as well as the nested structure in steps 4–5 in Algorithm 3.2.

4.2. A priori error bounds. We next consider *a priori* bounds for $f(\boldsymbol{s}_{i+1}^{(0)}) - f(\boldsymbol{s}_{\rm opt})$ and $\|\boldsymbol{s}_{i+1}^{(0)} - \boldsymbol{s}_{\rm opt}\|_M$ before the occurrence of the breakdown for (3.3). Similar to the convergence analysis in [36], we have

LEMMA 4.4. *Suppose* $\|\boldsymbol{s}_{\rm opt}\|_M = \Delta$ *and* $\mathcal{W}$ *is an arbitrary subspace of* $\mathbb{R}^n$. *Let*

$$\widehat{\boldsymbol{s}} \in \arg \min_{\boldsymbol{s}\in\boldsymbol{s}_i+\mathcal{W},\ \|\boldsymbol{s}\|_M=\Delta} f(\boldsymbol{s}).$$

Then for any nonzero $\widetilde{\boldsymbol{s}} \in \boldsymbol{s}_i + \mathcal{W}$, *it holds that*

$$0 \le f(\widehat{\boldsymbol{s}}) - f(\boldsymbol{s}_{\rm opt}) \le 2(\theta_n + \lambda_{\rm opt})\|\widetilde{\boldsymbol{s}} - \boldsymbol{s}_{\rm opt}\|_M^2. \tag{4.2}$$

Furthermore, if (1.1) *is nondegenerate, then*

$$\|\widehat{\boldsymbol{s}} - \boldsymbol{s}_{\rm opt}\|_M \le 2\sqrt{\varkappa}\,\|\widetilde{\boldsymbol{s}} - \boldsymbol{s}_{\rm opt}\|_M, \tag{4.3}$$

where $\varkappa := \frac{\theta_n+\lambda_{\rm opt}}{\theta_1+\lambda_{\rm opt}}$ *is the condition number of* $M^{-\frac{1}{2}}H_{\rm opt}M^{-\frac{1}{2}}$ *given in* (2.6).

Proof. The proof is similar to that of [36, Theorem 4.3] and the details are omitted. ∎

Moreover, we also need the following classical result (see, e.g., [28, Theorem 6.25]) in our convergence analysis.

LEMMA 4.5. *On the interval* $[a, b]$ *and for a real value* $\gamma \notin [a, b]$, *we have*

$$\min_{\mathcal{L}\in\mathbb{P}_k,\mathcal{L}(\gamma)=1} \ \max_{t\in[a,b]} |\mathcal{L}(t)| = \frac{1}{\left|\mathscr{T}_k(1+2\frac{\gamma-b}{b-a})\right|}, \tag{4.4}$$

and the polynomial $\mathcal{L}(t) = \mathscr{T}_k(1+2\frac{t-b}{b-a})/\mathscr{T}_k(1+2\frac{\gamma-b}{b-a})$ *solves the above problem* (4.4).

THEOREM 4.6. *Suppose* (1.1) *is nondegenerate with* $\|\boldsymbol{s}_{\rm opt}\|_M = \Delta$, *and* $(\boldsymbol{s}_i, \lambda_i)$ *is an approximate pair to* (1.1) *with* λ_i *the associated Lagrange multiplier and* $\boldsymbol{r}_i = H\boldsymbol{s}_i + \lambda_i M\boldsymbol{s}_i + \boldsymbol{g}$. *Let* $\boldsymbol{s}_{i+1}^{(0)}$ *be the solution to* (3.3) *satisfying* $\|\boldsymbol{s}_{i+1}^{(0)}\|_M = \Delta$; *then we have*

$$0 \le f(\boldsymbol{s}_{i+1}^{(0)}) - f(\boldsymbol{s}_{\rm opt}) \le 2(\theta_n + \lambda_{\rm opt}) \left(\frac{\|\boldsymbol{s}_{\rm opt} - \boldsymbol{s}_i\|_M}{\mathscr{T}_{\nu_i}(\eta)}\right)^2, \tag{4.5a}$$

$$\|\boldsymbol{s}_{\rm opt} - \boldsymbol{s}_{i+1}^{(0)}\|_M \le 2\sqrt{\varkappa}\,\frac{\|\boldsymbol{s}_{\rm opt} - \boldsymbol{s}_i\|_M}{\mathscr{T}_{\nu_i}(\eta)}, \quad \text{where} \tag{4.5b}$$

$$\varkappa = \frac{\theta_n + \lambda_{\rm opt}}{\theta_1 + \lambda_{\rm opt}}, \quad \eta = \frac{\varkappa+1}{\varkappa-1} > 1, \quad \nu_i = \min\left(\chi(M^{-1}\boldsymbol{r}_i), k_{i+1}, \chi(\boldsymbol{s}_i), m_{i+1}\right), \tag{4.6}$$

and $\mathscr{T}_{\nu_i}(t)$ *is the* ν_i*th Chebyshev polynomial of the first kind.*

Proof. Denote

$$\phi_i = \min(k_{i+1}, \chi(M^{-1}\boldsymbol{r}_i)) \quad \text{and} \quad \vartheta_i = \min(m_{i+1}, \chi(\boldsymbol{s}_i)). \tag{4.7}$$

Note that any

$$\begin{aligned}\boldsymbol{s} &\in \boldsymbol{s}_i + \mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i) + \mathcal{K}_{m_{i+1}}(M^{-1}H, \boldsymbol{s}_i)\\ &= \boldsymbol{s}_i + \mathcal{K}_{k_{i+1}}(M^{-1}H_{\rm opt}, M^{-1}\boldsymbol{r}_i) + \mathcal{K}_{k_{i+1}}(M^{-1}H_{\rm opt}, \boldsymbol{s}_i)\end{aligned}$$

can be represented as

$$\boldsymbol{s} = \boldsymbol{s}_i + \mathcal{P}(M^{-1}H_{\rm opt})M^{-1}\boldsymbol{r}_i + \widehat{\mathcal{P}}(M^{-1}H_{\rm opt})\boldsymbol{s}_i \tag{4.8}$$

for some polynomials $\mathcal{P} \in \mathbb{P}_{\phi_i-1}$ and $\widehat{\mathcal{P}} \in \mathbb{P}_{\vartheta_i-1}$. Also, with $\delta\lambda_i = \lambda_{\rm opt} - \lambda_i$, we have

$$\boldsymbol{r}_i = (H + \lambda_i M)\boldsymbol{s}_i + \boldsymbol{g} = H_{\rm opt}(\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt}) - \delta\lambda_i \cdot M\boldsymbol{s}_i. \tag{4.9}$$

We prove our assertion for the two cases: $\phi_i \le \vartheta_i$ and $\phi_i \ge \vartheta_i$.

Case $\phi_i \le \vartheta_i$: By (4.8) and (4.9), we have

$$\begin{aligned}
\boldsymbol{s} - \boldsymbol{s}_{\rm opt} &= \boldsymbol{s}_i - \boldsymbol{s}_{\rm opt} + \mathcal{P}(M^{-1}H_{\rm opt})M^{-1}\boldsymbol{r}_i + \widehat{\mathcal{P}}(M^{-1}H_{\rm opt})\boldsymbol{s}_i \\
&= (\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt}) + \mathcal{P}(M^{-1}H_{\rm opt})M^{-1}H_{\rm opt}(\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt}) \\
&\quad - \delta\lambda_i \cdot \mathcal{P}(M^{-1}H_{\rm opt})\boldsymbol{s}_i + \widehat{\mathcal{P}}(M^{-1}H_{\rm opt})\boldsymbol{s}_i \\
&= [I_n + \mathcal{P}(M^{-1}H_{\rm opt})M^{-1}H_{\rm opt}](\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt}) \\
&\quad - \delta\lambda_i \cdot \mathcal{P}(M^{-1}H_{\rm opt})\boldsymbol{s}_i + \widehat{\mathcal{P}}(M^{-1}H_{\rm opt})\boldsymbol{s}_i \\
&= \mathcal{L}(M^{-1}H_{\rm opt})(\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt}) - \left(\delta\lambda_i \cdot \mathcal{P}(M^{-1}H_{\rm opt}) - \widehat{\mathcal{P}}(M^{-1}H_{\rm opt})\right)\boldsymbol{s}_i, \qquad (4.10)
\end{aligned}$$

where $\mathcal{L}(t) = 1 + t\mathcal{P}(t) \in \mathbb{P}_{\phi_i}$ satisfying $\mathcal{L}(0) = 1$.

Note that the polynomials $\mathcal{P}(t)$ and $\widehat{\mathcal{P}}(t)$ in (4.10) can be chosen independently to have an $\boldsymbol{s}$. Since $\phi_i \le \vartheta_i$, we can choose a polynomial $\widehat{\mathcal{P}}(t) = \delta\lambda_i \cdot \mathcal{P}(t) \in \mathbb{P}_{\vartheta_i-1}$ so that the last term in (4.10) vanishes, i.e., $(\delta\lambda_i \cdot \mathcal{P}(M^{-1}H_{\rm opt}) - \widehat{\mathcal{P}}(M^{-1}H_{\rm opt}))\boldsymbol{s}_i = \boldsymbol{0}$. With this choice, the corresponding $\boldsymbol{s} - \boldsymbol{s}_{\rm opt}$ given in (4.10) is of the form $\boldsymbol{s} - \boldsymbol{s}_{\rm opt} = \mathcal{L}(M^{-1}H_{\rm opt})(\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt})$. Note that the degree of $\mathcal{L}(t)$ is ϕ_i. Now, we minimize the above $\|\boldsymbol{s}-\boldsymbol{s}_{\rm opt}\|_M$ over all polynomials $\mathcal{L}(t) = 1+t\mathcal{P}(t)$ of degree ϕ_i satisfying $\mathcal{L}(0) = 1$ to have a specific $\widetilde{\boldsymbol{s}}$. Precisely,

$$\begin{aligned}
\|\widetilde{\boldsymbol{s}} - \boldsymbol{s}_{\rm opt}\|_M &= \min_{\mathcal{L}\in\mathbb{P}_{\phi_i},\ \mathcal{L}(0)=1} \|\mathcal{L}(M^{-1}H_{\rm opt})(\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt})\|_M \\
&= \min_{\mathcal{L}\in\mathbb{P}_{\phi_i},\ \mathcal{L}(0)=1} \|M^{\frac{1}{2}}\mathcal{L}(M^{-1}H_{\rm opt})M^{-\frac{1}{2}}M^{\frac{1}{2}}(\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt})\|_2 \\
&\le \|M^{\frac{1}{2}}(\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt})\|_2 \cdot \min_{\mathcal{L}\in\mathbb{P}_{\phi_i},\ \mathcal{L}(0)=1} \|M^{\frac{1}{2}}\mathcal{L}(M^{-1}H_{\rm opt})M^{-\frac{1}{2}}\|_2 \\
&= \|(\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt})\|_M \cdot \min_{\mathcal{L}\in\mathbb{P}_{\phi_i},\ \mathcal{L}(0)=1} \|\mathcal{L}(M^{-\frac{1}{2}}HM^{-\frac{1}{2}} + \lambda_{\rm opt}I_n)\|_2 \\
&= \|\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt}\|_M \cdot \min_{\mathcal{L}\in\mathbb{P}_{\phi_i},\ \mathcal{L}(0)=1} \max_{i=1,2,\ldots,n} |\mathcal{L}(\theta_i + \lambda_{\rm opt})| \\
&\le \|\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt}\|_M \cdot \min_{\mathcal{L}\in\mathbb{P}_{\phi_i},\ \mathcal{L}(0)=1} \max_{t\in[\theta_1+\lambda_{\rm opt},\theta_n+\lambda_{\rm opt}]} |\mathcal{L}(t)| \\
&= \frac{\|\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt}\|_M}{\mathscr{T}_{\phi_i}(\eta)}, \qquad (4.11)
\end{aligned}$$

where the last equality follows from (4.4) and η is given in (4.6). Plug (4.11) into (4.2) and (4.3) with $\mathcal{W} = \mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i) + \mathcal{K}_{m_{i+1}}(M^{-1}H, \boldsymbol{s}_i)$ and $\widehat{\boldsymbol{s}} = \boldsymbol{s}_{i+1}^{(0)}$ to have (4.5).

Case $\phi_i \ge \vartheta_i$: For this case, we first claim that if $\delta\lambda_i = 0$, then (4.10) reduces to

$$\boldsymbol{s} - \boldsymbol{s}_{\rm opt} = \mathcal{L}(M^{-1}H_{\rm opt})(\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt}) + \widehat{\mathcal{P}}(M^{-1}H_{\rm opt})\boldsymbol{s}_i,$$

in which by choosing $\widehat{\mathcal{P}}(t) = 0$ we have $\boldsymbol{s} - \boldsymbol{s}_{\rm opt} = \mathcal{L}(M^{-1}H_{\rm opt})(\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt})$. Along the same arguments leading to (4.11), we know that there is a specific $\widetilde{\boldsymbol{s}} \in \boldsymbol{s}_i + \mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i) + \mathcal{K}_{m_{i+1}}(M^{-1}H, \boldsymbol{s}_i)$ so that (4.11) and hence (4.5) follow.

We next deal with $\delta\lambda_i \neq 0$. In this situation, from (4.9), we have

$$\boldsymbol{s}_i = \frac{M^{-1}H_{\rm opt}(\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt}) - M^{-1}\boldsymbol{r}_i}{\delta\lambda_i}. \tag{4.12}$$

Substituting (4.12) into (4.8) and replacing $\check{\mathcal{P}}(t) = \widehat{\mathcal{P}}(t)/\delta\lambda_i \in \mathbb{P}_{\vartheta_i - 1}$, we have

$$\begin{aligned}
\boldsymbol{s} - \boldsymbol{s}_{\rm opt} &= \boldsymbol{s}_i - \boldsymbol{s}_{\rm opt} + \mathcal{P}(M^{-1}H_{\rm opt})M^{-1}\boldsymbol{r}_i + \widehat{\mathcal{P}}(M^{-1}H_{\rm opt})\boldsymbol{s}_i \\
&= (\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt}) + \mathcal{P}(M^{-1}H_{\rm opt})M^{-1}\boldsymbol{r}_i + \check{\mathcal{P}}(M^{-1}H_{\rm opt})M^{-1}H_{\rm opt}(\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt}) \\
&\quad - \check{\mathcal{P}}(M^{-1}H_{\rm opt})M^{-1}\boldsymbol{r}_i \\
&= [I_n + \check{\mathcal{P}}(M^{-1}H_{\rm opt})M^{-1}H_{\rm opt}](\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt}) \\
&\quad + [\mathcal{P}(M^{-1}H_{\rm opt}) - \check{\mathcal{P}}(M^{-1}H_{\rm opt})]M^{-1}\boldsymbol{r}_i \\
&= \check{\mathcal{L}}(M^{-1}H_{\rm opt})(\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt}) + [\mathcal{P}(M^{-1}H_{\rm opt}) - \check{\mathcal{P}}(M^{-1}H_{\rm opt})]M^{-1}\boldsymbol{r}_i. \qquad (4.13)
\end{aligned}$$

Since $\phi_i \geq \vartheta_i$, for any given $\check{\mathcal{P}} \in \mathbb{P}_{\vartheta_i - 1}$, we can choose $\mathcal{P} \in \mathbb{P}_{\phi_i - 1}$ so that (4.13) reduces to $\boldsymbol{s} - \boldsymbol{s}_{\rm opt} = \check{\mathcal{L}}(M^{-1}H_{\rm opt})(\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt})$. By Lemma 4.5 again, we know that there is $\widetilde{\boldsymbol{s}} \in \boldsymbol{s}_i + \mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i) + \mathcal{K}_{m_{i+1}}(M^{-1}H, \boldsymbol{s}_i)$ so that $\|\widetilde{\boldsymbol{s}} - \boldsymbol{s}_{\rm opt}\|_M \leq \|\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt}\|_M / \mathscr{T}_{\vartheta_i}(\eta)$ and our assertion follows by Lemma 4.4. ∎

Remark 4.2.

(1) Accordingly, by (4.9), one has $\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt} = (M^{-1}H_{\rm opt})^{-1}(M^{-1}\boldsymbol{r}_i + \delta\lambda_i \cdot \boldsymbol{s}_i)$, and $f(\boldsymbol{s}_{i+1}^{(0)}) - f(\boldsymbol{s}_{\rm opt})$ and $\|\boldsymbol{s}_{i+1}^{(0)} - \boldsymbol{s}_{\rm opt}\|_M$ in (4.5) can further be bounded by $\boldsymbol{r}_i$ and $|\lambda_i - \lambda_{\rm opt}|$ as follows:

$$0 \leq f(\boldsymbol{s}_{i+1}^{(0)}) - f(\boldsymbol{s}_{\rm opt}) \leq \frac{2\varkappa}{\theta_1 + \lambda_{\rm opt}} \left(\frac{\|\boldsymbol{r}_i\|_{M^{-1}} + |\lambda_i - \lambda_{\rm opt}|\Delta}{\mathscr{T}_{\nu_i}(\eta)} \right)^2, \tag{4.14a}$$

$$\|\boldsymbol{s}_{\rm opt} - \boldsymbol{s}_{i+1}^{(0)}\|_M \leq \frac{2\sqrt{\varkappa}}{\theta_1 + \lambda_{\rm opt}} \frac{\|\boldsymbol{r}_i\|_{M^{-1}} + |\lambda_i - \lambda_{\rm opt}|\Delta}{\mathscr{T}_{\nu_i}(\eta)}. \tag{4.14b}$$

These bounds reflect the convergence of LTRSR: $\boldsymbol{s}_{i+1}^{(0)}$ resulting from the inner-loop iteration is sufficiently accurate if either $\mathscr{T}_{\nu_i}(\eta)$ is large or the residual $\|\boldsymbol{r}_i\|_{M^{-1}}$ and $|\lambda_i - \lambda_{\rm opt}|$ are small. The former occurs whenever η is sufficiently larger if than 1 (i.e., the TRS (1.1) is far from degenerate) or the degree ν_i of the Krylov subspace is sufficiently large, whereas $\|\boldsymbol{r}_i\|_{M^{-1}}$ and $|\lambda_i - \lambda_{\rm opt}|$ can be both reduced in the outer-loop iteration.

(2) By the relation $\frac{1}{\mathscr{T}_{\nu_i}(\eta)} < 2\left(\frac{\sqrt{\varkappa}-1}{\sqrt{\varkappa}+1}\right)^{\nu_i}$, for (i) of Theorem 4.6, the bounds in (4.5) or (4.14) imply the linear convergence of the sequences of $\{f(\boldsymbol{s}_i^{(0)})\}$ and $\{\boldsymbol{s}_i^{(0)}\}$.

(3) For the degenerate case, by an argument similar to that for (4.5a), we can prove the following counterpart of (4.5a):

$$0 \leq f(\boldsymbol{s}_{i+1}^{(0)}) - f(\boldsymbol{s}_{\rm opt}) \leq 2(\theta_n + \lambda_{\rm opt}) \left(\|\boldsymbol{t}_i(1:p)\|_2^2 + \frac{\|\boldsymbol{t}_i(p+1:n)\|_2^2}{\mathscr{T}_{\nu_i}^2(\eta_{\rm d})} \right),$$

where $\boldsymbol{t}_i = P^{\rm T}M^{\frac{1}{2}}(\boldsymbol{s}_{\rm opt} - \boldsymbol{s}_i)$, $\eta_{\rm d} = \frac{\varkappa_{\rm d}+1}{\varkappa_{\rm d}-1} > 1$, and $\varkappa_{\rm d} = \frac{\theta_n - \theta_1}{\theta_{p+1} - \theta_1}$, with P stated in (2.4) and $\theta_1 = \cdots = \theta_p < \theta_{p+1}$.

As a comparison, we state the bounds for the objective function and the solution when only the subspace $\mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i)$ or $\mathcal{K}_{m_{i+1}}(M^{-1}H, \boldsymbol{s}_i)$ is used in the inner loop.

THEOREM 4.7. *Suppose* (1.1) *is nondegenerate with* $\|\boldsymbol{s}_{\rm opt}\|_M = \Delta$, *and* $(\boldsymbol{s}_i, \lambda_i)$ *is an approximate pair to* (1.1) *with* λ_i *the associated Lagrange multiplier and* $\boldsymbol{r}_i = H\boldsymbol{s}_i + \lambda_i M\boldsymbol{s}_i + \boldsymbol{g}$.

(i) *If* $\widehat{\boldsymbol{s}}_{i+1}^{(0)}$ *is a solution to*

$$\min_{\boldsymbol{s}\in\boldsymbol{s}_i+\mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i),\ \|\boldsymbol{s}\|_M=\Delta} f(\boldsymbol{s}), \tag{4.15}$$

then we have

$$0 \le f(\widehat{\boldsymbol{s}}_{i+1}^{(0)}) - f(\boldsymbol{s}_{\rm opt}) \le 2(\theta_n + \lambda_{\rm opt})\left(\frac{\|\boldsymbol{s}_{\rm opt} - \boldsymbol{s}_i\|_M}{\mathscr{T}_{\phi_i}(\eta)} + \frac{2|\lambda_i - \lambda_{\rm opt}|\Delta}{\theta_1 + \lambda_{\rm opt}}\right)^2, \tag{4.16a}$$

$$\|\boldsymbol{s}_{\rm opt} - \widehat{\boldsymbol{s}}_{i+1}^{(0)}\|_M \le 2\sqrt{\varkappa}\left(\frac{\|\boldsymbol{s}_{\rm opt} - \boldsymbol{s}_i\|_M}{\mathscr{T}_{\phi_i}(\eta)} + \frac{2|\lambda_i - \lambda_{\rm opt}|\Delta}{\theta_1 + \lambda_{\rm opt}}\right), \tag{4.16b}$$

where $\phi_i = \min(k_{i+1}, \chi(M^{-1}\boldsymbol{r}_i))$ *and* $\varkappa, \eta$ *are defined by* (4.6).

(ii) *If* $\breve{\boldsymbol{s}}_{i+1}^{(0)}$ *is a solution to*

$$\min_{\boldsymbol{s}\in\boldsymbol{s}_i+\mathcal{K}_{m_{i+1}}(M^{-1}H, \boldsymbol{s}_i),\ \|\boldsymbol{s}\|_M=\Delta} f(\boldsymbol{s})$$

and $\lambda_i \ne \lambda_{\rm opt}$, *then we have*

$$0 \le f(\breve{\boldsymbol{s}}_{i+1}^{(0)}) - f(\boldsymbol{s}_{\rm opt}) \le 2(\theta_n + \lambda_{\rm opt})\left(\frac{\|\boldsymbol{s}_{\rm opt} - \boldsymbol{s}_i\|_M}{\mathscr{T}_{\vartheta_i}(\eta)} + \frac{2\|\boldsymbol{r}_i\|_{M^{-1}}}{\theta_1 + \lambda_{\rm opt}}\right)^2, \tag{4.17a}$$

$$\|\boldsymbol{s}_{\rm opt} - \breve{\boldsymbol{s}}_{i+1}^{(0)}\|_M \le 2\sqrt{\varkappa}\left(\frac{\|\boldsymbol{s}_{\rm opt} - \boldsymbol{s}_i\|_M}{\mathscr{T}_{\vartheta_i}(\eta)} + \frac{2\|\boldsymbol{r}_i\|_{M^{-1}}}{\theta_1 + \lambda_{\rm opt}}\right), \tag{4.17b}$$

where $\vartheta_i = \min(m_{i+1}, \chi(\boldsymbol{s}_i))$ *and* $\varkappa, \eta$ *are defined by* (4.6).

Proof.

(i) Let $\boldsymbol{s} \in \boldsymbol{s}_i + \mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i)$ be arbitrary, and ϕ_i and ϑ_i be defined by (4.7). By (4.10) with $\widehat{\mathcal{P}} = 0$ for the case $\phi_i \le \vartheta_i$, we have

$$\boldsymbol{s} - \boldsymbol{s}_{\rm opt} = \mathcal{L}(M^{-1}H_{\rm opt})(\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt}) - \delta\lambda_i \cdot \mathcal{P}(M^{-1}H_{\rm opt})\boldsymbol{s}_i \quad \forall \mathcal{P} \in \mathbb{P}_{\phi_i - 1},$$

where $\mathcal{L}(t) = 1 + t\mathcal{P}(t) \in \mathbb{P}_{\phi_i}$ satisfies $\mathcal{L}(0) = 1$. Taking inspiration from Lemma 4.5, we let $\mathcal{L}(t) = \mathscr{T}_{\phi_i}(1 + 2\frac{t-\theta_n-\lambda_{\rm opt}}{\theta_n-\theta_1})/\mathscr{T}_{\phi_i}(\eta)$ and, by an argument similar to that for (4.11), it follows that

$$\begin{aligned}
&\min_{\boldsymbol{s}\in\boldsymbol{s}_i+\mathcal{K}_{k_{i+1}}(M^{-1}H, M^{-1}\boldsymbol{r}_i)} \|\boldsymbol{s} - \boldsymbol{s}_{\rm opt}\|_M \\
&\quad\le \frac{\|\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt}\|_M}{\mathscr{T}_{\phi_i}(\eta)} + |\delta\lambda_i| \cdot \|\mathcal{P}(M^{-1}H_{\rm opt})\boldsymbol{s}_i\|_M \\
&\quad\le \frac{\|\boldsymbol{s}_i - \boldsymbol{s}_{\rm opt}\|_M}{\mathscr{T}_{\phi_i}(\eta)} + |\delta\lambda_i| \cdot \|M^{\frac{1}{2}}\mathcal{P}(M^{-1}H_{\rm opt})M^{-\frac{1}{2}}\|_2 \cdot \Delta
\end{aligned} \tag{4.18}$$

and

$$\begin{aligned}
&\|M^{\frac{1}{2}}\mathcal{P}(M^{-1}H_{\rm opt})M^{-\frac{1}{2}}\|_2\\
&\quad=\|(\mathcal{L}(M^{-\frac{1}{2}}HM^{-\frac{1}{2}}+\lambda_{\rm opt}I_n)-I_n)(M^{-\frac{1}{2}}HM^{-\frac{1}{2}}+\lambda_{\rm opt}I_n)^{-1}\|_2\\
&\quad\le\frac{1}{\theta_1+\lambda_{\rm opt}}\cdot\max_{j=1,\ldots,n}\left|\frac{\mathscr{T}_{\phi_i}(1+2\frac{\theta_j-\theta_n}{\theta_n-\theta_1})}{\mathscr{T}_{\phi_i}(\eta)}-1\right|\le\frac{2}{\theta_1+\lambda_{\rm opt}},
\end{aligned}\tag{4.19}$$

which together with (4.18) and Lemma 4.4 shows (4.16).

(ii) For any $\boldsymbol{s}\in\boldsymbol{s}_i+\mathcal{K}_{m_{i+1}}(M^{-1}H,\boldsymbol{s}_i)$, $\boldsymbol{s}=\boldsymbol{s}_i+\widehat{\mathcal{P}}(M^{-1}H_{\rm opt})\boldsymbol{s}_i$ for some polynomial $\widehat{\mathcal{P}}\in\mathbb{P}_{\vartheta_i-1}$; setting $\check{\mathcal{P}}(t)=\widehat{\mathcal{P}}(t)/\delta\lambda_i\in\mathbb{P}_{\vartheta_i-1}$ and by (4.13) with $\mathcal{P}=0$, it follows that

$$\boldsymbol{s}-\boldsymbol{s}_{\rm opt}=\check{\mathcal{L}}(M^{-1}H_{\rm opt})(\boldsymbol{s}_i-\boldsymbol{s}_{\rm opt})-\check{\mathcal{P}}(M^{-1}H_{\rm opt})M^{-1}\boldsymbol{r}_i,$$

and thus, analogous to (4.18) and (4.19),

$$\begin{aligned}
&\min_{\boldsymbol{s}\in\boldsymbol{s}_i+\mathcal{K}_{m_{i+1}}(M^{-1}H,\boldsymbol{s}_i)}\|\boldsymbol{s}-\boldsymbol{s}_{\rm opt}\|_M\\
&\quad\le\frac{\|\boldsymbol{s}_i-\boldsymbol{s}_{\rm opt}\|_M}{\mathscr{T}_{\vartheta_i}(\eta)}+\|M^{\frac{1}{2}}\check{\mathcal{P}}(M^{-1}H_{\rm opt})M^{-\frac{1}{2}}\|_2\cdot\|\boldsymbol{r}_i\|_{M^{-1}}\\
&\quad\le\frac{\|\boldsymbol{s}_i-\boldsymbol{s}_{\rm opt}\|_M}{\mathscr{T}_{\vartheta_i}(\eta)}+\frac{2\|\boldsymbol{r}_i\|_{M^{-1}}}{\theta_1+\lambda_{\rm opt}},
\end{aligned}$$

which together with Lemma 4.4 shows (4.17). ∎

Revealed by Theorems 4.6 and 4.7, we know the advantage of choosing (4.1) as the constraint for the inner-loop iteration: *The additional subspace $\mathcal{K}_{m_{i+1}}(M^{-1}H,\boldsymbol{s}_i)$ tries to eliminate the inaccuracy induced in the Lagrange multiplier λ_i so that the convergence of the inner-loop iteration behaves similarly to the Krylov subspace methods like CG for the preconditioned "linear" system $M^{-1}H_{\rm opt}\boldsymbol{s}_{\rm opt}=-M^{-1}\boldsymbol{g}$.*

Theorems 4.6 and 4.7 also help us determine k_{i+1} and m_{i+1}. First, we observed by our numerical experiments that $|\lambda_i-\lambda_{\rm opt}|$ is roughly of the same order as $f(\boldsymbol{s}_i)-f(\boldsymbol{s}_{\rm opt})$, whereas $\|\boldsymbol{r}_i\|_{M^{-1}}$ has the same order as $\|\boldsymbol{s}_i-\boldsymbol{s}_{\rm opt}\|_M$. Revealed by Lemma 4.4, we know then that $|\lambda_i-\lambda_{\rm opt}|$ converges generally faster than $\|\boldsymbol{r}_i\|_{M^{-1}}$ does; as a result, it is more efficient to choose $k_{i+1}>m_{i+1}$, where $\mathcal{K}_{m_{i+1}}(M^{-1}H,\boldsymbol{s}_i)$ can be understood as the refinement subspace to improve the accuracy of the approximation $\widehat{\boldsymbol{s}}_{i+1}^{(0)}$ to (4.15). In this procedure, the additional $\mathcal{K}_{m_{i+1}}(M^{-1}H,\boldsymbol{s}_i)$ tries to eliminate the term $\delta\lambda_i\cdot\mathcal{P}(M^{-1}H_{\rm opt})\boldsymbol{s}_i-\widehat{\mathcal{P}}(M^{-1}H_{\rm opt})\boldsymbol{s}_i$ appearing in (4.10), and by (4.19) it holds that $\|\delta\lambda_i\mathcal{P}(M^{-1}H_{\rm opt})\boldsymbol{s}_i\|_M\le 2\Delta|\lambda_i-\lambda_{\rm opt}|/(\theta_1+\lambda_{\rm opt})$.

5. LTRSR with tCG initialization (LTRSR_tCG). It was pointed out in section 2 that if the constraint $\|\boldsymbol{s}\|_M\le\Delta$ is inactive, i.e., $\lambda_{\rm opt}=0$, then LTRS is mathematically equivalent to PCG for solving $H\boldsymbol{s}_{\rm opt}=-\boldsymbol{g}$. In this case, PCG is an efficient numerical implementation. Moreover, by the intimate relation between PCG and the three-term preconditioned Lanczos process (e.g., [16, 28]), it is true that the M-orthogonal basis Q_{k_1} for $\mathcal{K}_{k_1}(M^{-1}H,M^{-1}\boldsymbol{r}_0)$ and the corresponding tridiagonal matrix T_{k_1} can be recovered from PCG .

However, when H is indefinite, then $\boldsymbol{s}_{\rm opt}$ is on the boundary and PCG can never deliver a sufficiently accurate approximation generally. In this situation, the method tCG generates the piecewise linear path connecting the PCG iterates and stops at the

approximation where this path leaves the boundary $\|\boldsymbol{s}\|_M \le \Delta$. When $\lambda_{\rm opt} = 0$, then tCG reduces to PCG and delivers an approximation for $\boldsymbol{s}_{\rm opt} = H^{-1}\boldsymbol{g}$, whereas when $\lambda_{\rm opt} > 0$, a direction with negative curvature[4] is found, implying H is indefinite and thereby $\|\boldsymbol{s}_{\rm opt}\|_M = \Delta$.

As we pointed out in section 1, GLTR in [16] is indeed an efficient implementation of LTRS. It consists of three passes. In the first pass, tCG is applied to (1.1) and automatically detects whether the solution $\boldsymbol{s}_{\rm opt}$ is within the constraint or on its boundary[5] $\|\boldsymbol{s}\|_M = \Delta$. For the case $\lambda_{\rm opt} > 0$, as soon as $\|\boldsymbol{s}_{\rm opt}\|_M = \Delta$ is identified, in the second and third passes, GLTR subsequently solves smaller size trust-region subproblems, which, by the relationship between PCG and the preconditioned Lanczos process, result from projecting the original TRS (1.1) onto the Krylov subspaces by the Lanczos process, or equivalently by PCG (see [16] in detail). It is verified [16] that such a strategy is capable of achieving a boundary approximation efficiently on the one hand, and also maintains the efficiencies of PCG so long as the iterates lie in the interior, on the other hand.

In a fashion similar to GLTR, we can incorporate tCG into the framework of LTRSR. In particular, as the first stage for LTRSR, instead simply setting $\boldsymbol{s}_0 = \boldsymbol{0}$, we apply tCG as an initialization (Algorithm 5.1) to yield an approximate solution $\boldsymbol{s}_0$ and the associated Lagrange multiplier λ_0. We remark that Algorithm 5.1 is essentially in the spirit of GLTR. Note that the return of Algorithm 5.1 is $(\boldsymbol{s}_0, \boldsymbol{r}_0, \lambda_0, k_0)$, where $\boldsymbol{r}_0$ is the corresponding residual for restarting the next Lanczos process and k_0 is the dimension of the Krylov subspace where $\boldsymbol{s}_0$ lies. The largest possible dimension k_0 is $\max(k_0^{\max}, k_{tcg})$.

With LTRSR initialized by tCG, we arrive at a version of LTRSR, namely LTRSR_tCG, and summarize it in Algorithm 5.2.

Remark 5.1. Three remarks regarding Algorithm 5.1 and Algorithm 5.2 follow.

(1) By the relation between PCG and the preconditioned Lanczos process (e.g., [16, 28]), the computation of the tridiagonal matrix T_0 at step 4 of Algorithm 5.1 can be computed alternatively as $T_0 = Q_0^{\rm T} H Q_0$. In other words, one can skip the computation at step 4 and form $T_0 = Q_0^{\rm T} H Q_0$ in (5.1) instead.

(2) In general, we prefer to setting a relatively large k_{tcg} because we want to use the high horsepower of PCG so that LTRSR_tCG reduces to PCG in the case of $\lambda_{\rm opt} = 0$, while it finds a boundary approximation otherwise.

(3) Let $(\boldsymbol{s}_0, \boldsymbol{r}_0, \lambda_0, k_0)$ be the return of Algorithm 5.1. According to our motivation for LTRSR, the subsequent nested Lanczos procedure in LTRSR can be understood as a type of restarting for solving $(H + \lambda_{\rm opt} M)\boldsymbol{s}_{\rm opt} = -\boldsymbol{g}$ with the preconditioner M. Note that the difference between $\boldsymbol{r}_0$ and the true residual of this system at $\boldsymbol{s}_0$ is

$$\boldsymbol{r}_0 - ((H + \lambda_{\rm opt} M)\boldsymbol{s}_0 + \boldsymbol{g}) = (\lambda_0 - \lambda_{\rm opt}) M \boldsymbol{s}_0.$$

Thus the more accurate the λ_0 is, the better the performance is of the restarting. Therefore, we choose a relatively large $k_0^{\max}$ and continue to expand the subspace to $\mathcal{K}_{k_0^{\max}}(M^{-1}H, M^{-1}\boldsymbol{g})$ when PCG terminates early with detection of a negative curvature or the boundary case.

[4] A vector $\boldsymbol{p}$ is a direction of negative curvature if $\boldsymbol{p}^{\rm T} H \boldsymbol{p} < 0$.

[5] In practice, the boundary case can be detected by checking the Cholesky decomposition of the triangular matrix T, which can be carried out economically and recursively as T has an appended row and column each time.

Algorithm 5.1. tCG initialization.

Given a tolerance $\epsilon_{tcg} > 0$, k_{tcg}, and the maximum $k_0^{\max}$ for k_0, it initializes LTRSR.

$\boldsymbol{s}_0 = \boldsymbol{0},\ \boldsymbol{g}_1 = \boldsymbol{g},\ \boldsymbol{v}_1 = M^{-1}\boldsymbol{g},\ \boldsymbol{p}_1 = -\boldsymbol{v}_1,\ Q_0 = T_0 = [\,]$
for $j = 1, 2, \ldots, k_{tcg}$ **do**
  $\alpha_j = \boldsymbol{g}_j^{\mathrm{T}}\boldsymbol{v}_j / \boldsymbol{p}_j^{\mathrm{T}} H \boldsymbol{p}_j$
  Augment T_0 by one row and one column according to [16, equation (4.20)]
  $Q_0 = [Q_0, \sigma_j \boldsymbol{v}_j / \sqrt{\boldsymbol{g}_j^{\mathrm{T}}\boldsymbol{v}_j}]$, where $\sigma_j = -\operatorname{sign}(\alpha_{j-1})\sigma_{j-1}$ and $\sigma_1 = 1$
  $\boldsymbol{s}_0 = \boldsymbol{s}_0 + \alpha_j \boldsymbol{p}_j$
  if $\alpha_j \le 0$ or $\|\boldsymbol{s}_0\|_M \ge \Delta$ **then**
    if $j < k_0^{\max}$ **then**
      Expand $Q_0 \in \mathbb{R}^{n\times j}$ to an M-orthogonal basis $Q_0 \in \mathbb{R}^{n\times k_0^{\max}}$ of $\mathcal{K}_{k_0^{\max}}(M^{-1}H, M^{-1}\boldsymbol{g})$ by the Lanczos process or PCG, and $k_0 = k_0^{\max}$
    else
      $Q_0 = Q_0(:, 1 : k_0^{\max})$, $T_0 = T_0(1 : k_0^{\max}, 1 : k_0^{\max})$, and $k_0 = k_0^{\max}$
      Obtain the solution $\boldsymbol{z}_0$ and the associated Lagrange multiplier λ_0 of

$$\min_{\|\boldsymbol{z}\|_2 \le \Delta} \frac{1}{2}\boldsymbol{z}^{\mathrm{T}} T_0 \boldsymbol{z} + \|\boldsymbol{g}\|_{M^{-1}} \cdot \boldsymbol{z}^{\mathrm{T}} \boldsymbol{e}_1 \tag{5.1}$$

      Set $\boldsymbol{s}_0 = Q_0\boldsymbol{z}_0,\ \boldsymbol{r}_0 = H\boldsymbol{s}_0 + \lambda_0 M\boldsymbol{s}_0 + \boldsymbol{g}$, and **return** $(\boldsymbol{s}_0, \boldsymbol{r}_0, \lambda_0, k_0)$
    end if
  end if
  $\boldsymbol{g}_{j+1} = \boldsymbol{g}_j + \alpha_j H\boldsymbol{p}_j$ and $\boldsymbol{v}_{j+1} = M^{-1}\boldsymbol{g}_{j+1}$
  if $\|\boldsymbol{g}_{j+1}\|_{M^{-1}} \le \epsilon_{tcg}$ **then**
    Set $k_0 = j,\ \lambda_0 = 0,\ \boldsymbol{s}_0 = \boldsymbol{s}_0,\ \boldsymbol{r}_0 = H\boldsymbol{s}_0 + \boldsymbol{g}$, and **return** $(\boldsymbol{s}_0, \boldsymbol{r}_0, \lambda_0, k_0)$
  end if
  $\beta_j = \boldsymbol{g}_{j+1}^{\mathrm{T}}\boldsymbol{v}_{j+1} / \boldsymbol{g}_j^{\mathrm{T}}\boldsymbol{v}_j$
  $\boldsymbol{p}_{j+1} = -\boldsymbol{v}_{j+1} + \beta_j \boldsymbol{p}_j$
end for

Algorithm 5.2. LTRSR with tCG initialization (LTRSR_tCG).

Given $H, M \in \mathbb{R}^{n\times n}$, and $\boldsymbol{g} \in \mathbb{R}^n$, RLTRS_tCG (approximately) solves (1.1).

1: Setting: Choose $\epsilon_{tcg},\ \epsilon_0,\ k_{tcg},\ k_0^{\max},\ i_{\max}$, and a vector of number of iterations $[k_1, m_1, k_2, m_2, \ldots, k_{i_{\max}}, m_{i_{\max}}]$;
2: tCG initialization: Apply tCG (Algorithm 5.1) and it returns $(\boldsymbol{s}_0, \boldsymbol{r}_0, \lambda_0, k_0)$;
3: LTRSR: Run step 2 (starting with $i = 0$) to step 7 of LTRSR (Algorithm 3.2); it returns an approximate solution $\boldsymbol{s}_\ell$ and the associated Lagrange multiplier λ_ℓ.

6. Numerical experiments. In this section, we present some numerical results of LTRSR_tCG, and compare its performance with that of GLTR [16], an eigenvalue-based algorithm TRSgep proposed recently in [1], a dual simplex type method (denoted by FW) developed in the semidefinite framework in [9, 10], and the LSTRS method proposed in [26] based on a formulation of TRS as a parameterized eigenvalue problem. The MATLAB codes of the three algorithms TRSgep,[6] FW,[7] and LSTRS[8] are

[6] TRSgep is available at http://www.opt.mist.i.u-tokyo.ac.jp/∼nakatsukasa/codes/TRSgep.m
[7] FW is available at [10] http://www.math.uwaterloo.ca/∼hwolkowi/henry/software/trustreg.d/
[8] LSTRS is available at http://ta.twi.tudelft.nl/wagm/users/rojas/lstrs.html

available on the internet, while we implement LTRSR_tCG and GLTR in MATLAB 8.5.0(R2015a). In our implementation of LTRSR_tCG and GLTR, we use the MATLAB built-in function `trust`[9] to solve the sequential projected TRSs whenever the dimension of the projected TRS is less than 1000; otherwise, we call the method [22, Algorithm 4.3] to solve the projected TRS. Since [22, Algorithm 4.3] does not deal with the degenerate or nearly degenerate case, in our case, we modify it to cover the degenerate or nearly degenerate case. For this modification, the MATLAB built-in function `eigs` will be called to compute the left-most eigenvalue and eigenvector; but when the tridiagonal matrix T is severely ill-conditioned and leads to the failure of `eigs`, we then resort `trust` to solve the resulting projected TRS.

All tests are conducted on a PC under Windows 7 (64bit) system with an Intel Core i5-2320M CPU (3.0 GHz) and 4 GB memory. We say that an approximation $(\lambda, \boldsymbol{s})$ is accurate within the given tolerance whenever the "relative" residual

$$Rres := \frac{\|\boldsymbol{r}\|_{M^{-1}}}{\|\boldsymbol{g}\|_{M^{-1}}} = \frac{\|(H+\lambda M)\boldsymbol{s}+\boldsymbol{g}\|_{M^{-1}}}{\|\boldsymbol{g}\|_{M^{-1}}} \leq 10^{-10}.$$

The parameters of our LTRSR_tCG are set as follows:

$k_{tcg} = n$, $k_0^{\max} = \min(n, 500)$ and $k_i = 50$, $m_i = 2$ for all $1 \leq i \leq i_{\max} = 200$.

The choice of a relatively large $k_0^{\max}$ has been discussed in Remark 5.1. Also, we remark that in the first $k_0^{\max}$ iterations, LTRSR_tCG generates the same iterates as GLTR, but for the ill-conditioned problems LTRSR_tCG can improve GLTR significantly after $k_0^{\max}$ iterations. In the following, we illustrate the performance of our algorithm from three different aspects.

6.1. Effect of orthogonality of Lanczos vectors. In this subsection, we will investigate the performance of GLTR with and without re-orthogonalization of Lanczos vectors [6, Algorithm 7.2], and compare it with LTRSR_tCG. The GLTR method with full re-orthogonalization is denoted by GLTR(orth). Apart from the full re-orthogonalization procedure, GLTR(orth) is the same as GLTR. Two types of examples are chosen for testing:

(I) $H = G + G^{\mathrm{T}}$, where $G =$`randn`(n), $\boldsymbol{g} =$`randn`$(n, 1)$,

(II) $H = GG^{\mathrm{T}} - I_n$, where $G =$`randn`(n), $\boldsymbol{g} =$`randn`$(n, 1)$.

The trust region radius varies[10] from $\Delta = 10$ to $\Delta = 100$, and $M = I_n$.

To make a clear comparison, we plot Figures 6.1 and 6.2 to show how the relative residual $Rres$ in the approximate solution of each algorithm behaves against the number of (n-by-n) matrix-vector products (i.e., the number of H-vector products) with various condition number $\varkappa$ of H_{opt} given in (4.6).

Figure 6.1 (for the type (I) $H = G + G^{\mathrm{T}}$) shows that when the condition number $\varkappa$ is small, LTRSR_tCG, GLTR, and GLTR(orth) roughly have the same performance. The reason is that all three algorithms stop within 500 iterations and LTRSR_tCG is actually GLTR in this situation.

However, for test examples with $H = GG^{\mathrm{T}} - I_n$, the performances of LTRSR_tCG, GLTR, and GLTR(orth) vary greatly due to a large condition number $\varkappa$. From Figure 6.2, we see that GLTR fails in most cases within n iterations, and both GLTR(orth) and LTRSR_tCG succeed to produce an approximation within the given tolerance. An interesting phenomenon observed from the right of Figure 6.2 is that $Rres$ from

[9] The built-in MATLAB routine `trust` is available in MATLAB 7.0 (R14). The solver `trust` is only suitable for small-to-medium size TRSs as it solves the secular equation $1/\|\boldsymbol{s}(\lambda)\| - 1/\Delta = 0$ by using the full eigen-decomposition of the coefficient matrix.

[10] It is shown in [36, Remark 4.8] that the larger Δ is, the more severely ill-conditioned the TRS is.

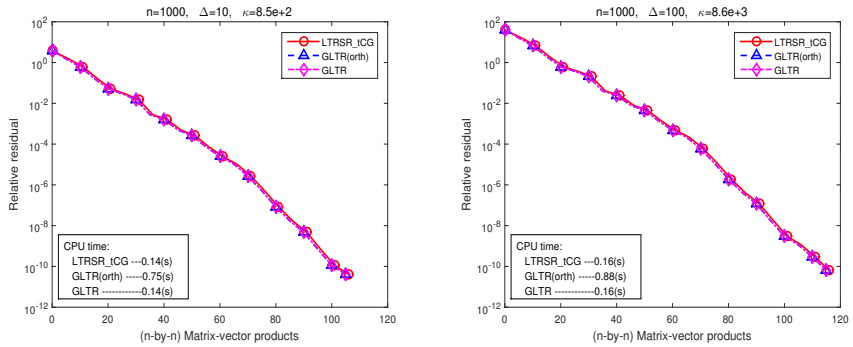

FIG. 6.1. *Relative residual vs. the number of matrix-vector products for* $H = G + G^{\mathrm{T}}$.

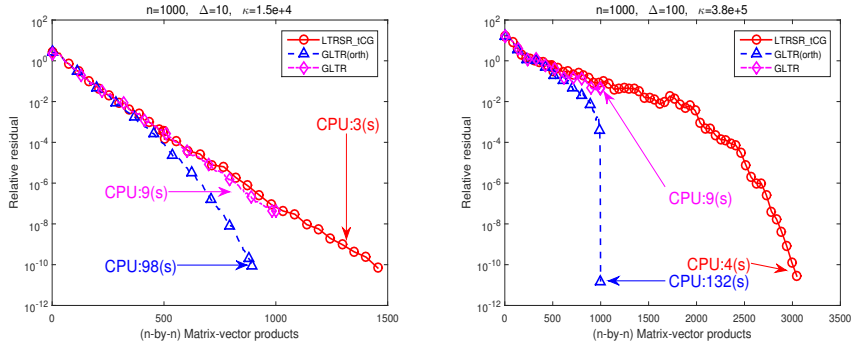

FIG. 6.2. *Relative residual vs. the number of matrix-vector products for* $H = GG^{\mathrm{T}} - I_n$.

GLTR(orth) drop suddenly in the final iteration, while *Rres* in LTRSR_tCG decrease gradually. More matrix-vector products are needed in LTRSR_tCG than GLTR(orth), but the CPU time (in seconds) used by LTRSR_tCG is the lowest. The increase of CPU times in GLTR(orth) and GLTR (which is an indication of the increase of the computational amounts) is mainly attributed to the need for solving much larger projected TRSs than those in LTRSR_tCG, and also the costs of full re-orthogonalization of Lanczos vectors in GLTR(orth). These computational savings in LTRSR_tCG justify the improvement upon GLTR and GLTR(orth).

6.2. The contribution of double Krylov subspace and nested structure. In this subsection, we evaluate the behavior of LTRSR_tCG with different choices of (k_i, m_i), and verify the contribution of the introduced Krylov subspace $\mathcal{K}_{m_i}(M^{-1}H, \boldsymbol{s}_i)$. Also, we will show the importance of the nested structure of steps 4–5 in Algorithm 3.4. For the former, according to Theorems 4.3–4.7, $k_i = m_i$ could be a good choice. Nevertheless, when $k_i = m_i$ gets bigger, more computational efforts for orthogonalization/re-orthogonalization are needed, and, most seriously, much larger projected TRSs are required to be solved. Our goal is to find a suitable pair (k_i, m_i) in practice.

Based on the numerical experiments presented in section 6.1, we test the type (II) $H = GG^{\mathrm{T}} - I_n$ with various Δ and $n = 2000, M = I_n$. Detailed results are reported in Table 6.1. The labels It_{cg}, It_{outer}, Prod, and cpu stand for the number of CG iterations, the number of outer iterations, the number of (n-by-n) matrix-vector products, and the computational CPU time, respectively.

TABLE 6.1
Numerical results of `LTRSR_tCG` *with different* k_i *and* m_i.

k_i	m_i	$n = 2000$, $\Delta = 10$, $\varkappa = 1.9\text{e}+4$					$n = 2000$, $\Delta = 100$, $\varkappa = 3.4\text{e}+5$				
		It_{cg}	It_{outer}	Prod	*Rres*	cpu	It_{cg}	It_{outer}	Prod	*Rres*	cpu
30	2	50	47	2020	8.7e-11	19.6	67	143	5188	3.0e-11	42.6
30	10	50	44	2265	8.2e-11	21.5	67	139	6160	5.4e-11	49.8
30	20	50	43	2644	8.1e-11	24.2	67	138	7489	9.2e-11	59.3
30	30	50	42	3003	7.4e-11	26.5	67	138	8859	6.8e-11	68.7
50	2	50	29	1986	6.5e-11	19.0	67	88	5113	8.0e-11	40.1
50	10	50	28	2149	8.4e-11	20.4	67	86	5687	6.1e-11	44.7
50	30	50	27	2608	6.4e-11	23.8	67	86	7387	2.5e-11	57.2
50	50	50	26	3027	4.2e-11	26.7	67	85	8986	3.4e-11	68.9
100	2	50	15	1944	6.7e-11	19.2	67	48	5343	2.9e-11	41.5
100	10	50	15	2056	5.5e-11	19.7	67	47	5608	4.5e-11	44.1
100	50	50	14	2465	7.6e-11	23.0	67	42	6693	8.7e-11	53.0
100	100	50	13	2914	8.6e-11	26.8	67	44	9145	5.5e-11	73.2
200	2	50	9	2126	2.0e-11	20.4	67	26	5577	1.9e-11	44.7
200	50	50	8	2259	1.9e-11	21.8	67	25	6526	8.2e-11	53.8
200	100	50	8	2609	8.1e-12	25.4	67	24	7425	7.8e-11	63.3
200	200	50	6	2507	5.5e-11	26.1	67	24	9725	8.8e-11	87.8

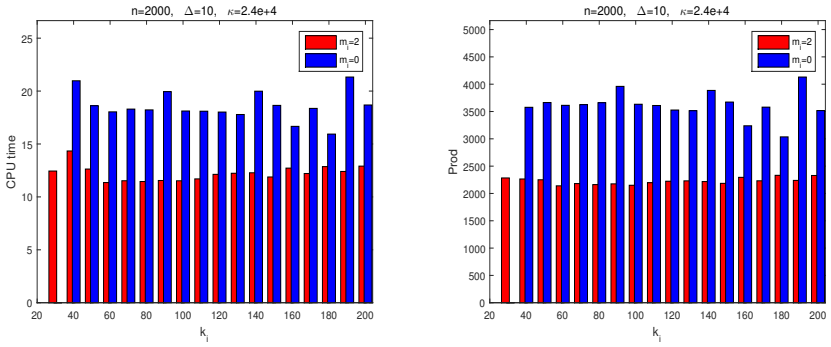

FIG. 6.3. *CPU time vs.* k_i *for* $H = GG^{\mathrm{T}} - I_n$.

From Table 6.1, we observed that as k_i and m_i increase, the number of outer iterations decrease monotonically, but neither the number of matrix-vector products nor the computational time decreases monotonically. Our experience suggests that, practically, $k_i = 50$ and $m_i = 2$ usually result in good performance. On the other hand, the Krylov subspace $\mathcal{K}_{m_i}(M^{-1}H, \boldsymbol{s}_i)$ is also very important. Figure 6.3 provides a clear picture to show that $m_i = 2$ is critical and should not be reduced to 0. In particular, in the case of $k_i = 30, 40$ and $m_i = 0$, LTRSR_tCG fails, and therefore the corresponding bars do not appear in Figure 6.3.

To demonstrate the contribution of the nested structure of steps 4–5 in Algorithm 3.2 as the refinement of $\boldsymbol{s}_{i+1}^{(0)}$ in (3.3), we report in Figure 6.4 the CPU time and the number of H-vector products with and without this nested structure for $n = 2000$ and $\Delta = 10$. Also, for $n = 2000$ and $\Delta = 100$, we observed that the one without the nested structure fails for all tested k_i. These results indicate that the nested structure is another key for LTRSR_tCG.

6.3. Comparison with other solvers. In this final subsection, we compare the performance of LTRSR_tCG with that of TRSgep, FW, and LSTRS. The default version of TRSgep is used for testing. For FW and LSTRS, since the default stopping

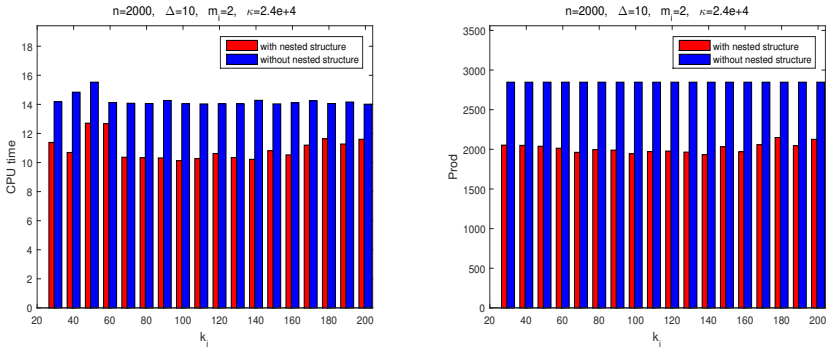

FIG. 6.4. *CPU time vs.* k_i *for* $H = GG^{\mathrm{T}} - I_n$.

conditions are different from ours, some functions or parameters of these two algorithms are modified. In particular, for FW, we modified the source codes such that it can accept a matrix-vector multiplication routine instead of the Hessian matrix, and we set `dgaptol`=1e-6. In LSTRS, we set `eigs` as its eigensolver and other parameters are their default values.

The test problems here are of the form $H = GG^{\mathrm{T}} - I_n$ where G include dense random matrices of type (II) and large and sparse ones. The set of our test sparse matrices G (from the University of Florida sparse matrix collection [4]) is a collection used in [20] and [37]. There are 10 examples in [20] for testing the locally optimal and heavy ball GMRES, and 20 matrices used in [37] for testing a Lanczos method for the extreme Lorentz eigenvalue problem. This forms a set of 30 testing examples G. The detailed characteristics of these matrices are listed in Table 6.2, where n is the dimension, nnz is the number of nonzero entries, and sparsity is nnz/n^2. The value $\Delta = 100$ and $\boldsymbol{g} = \mathtt{randn}(n, 1)$.

For random dense problems of type (II), we use two different weighted matrices: $M = I_n$ and a tridiagonal $M = tridiag(1, 3, 1)$. We generate 10 problems for each given dimension n. For each solver, we summarize the cases that the solver returns computed solutions within its own tolerance. The numerical results are presented in Tables 6.3 and 6.4. The label "% of cnvg" gives the percentage of successful cases in returning a computed solution that meets its own stopping rule, and Prod, $Rres$, and cpu are the average values over the cases counted in "% of cnvg." Since each solver uses a different stopping criterion, a computed return that meets the solver's own stopping rule does not mean that the relative residual is less than 10^{-10}; therefore, we also introduce the symbol $\circledast$ to indicate that the relative residual of the returned solution is not within 10^{-10}. When a solver fails at all 10 problems, it is indicated by the symbol $*$. We observed that the failure is mainly due to the failure during in calling eigs.

For the sparse problems in Table 6.2, we report the numerical results of the total 28 problems (the problems OPF_3754 and cvxqp3 are excluded because all the four methods failed) in Table 6.5. Estimates of the condition numbers $\varkappa$ for these problems are unattainable in MATLAB and not presented. The indicator $*$ here stands for the failure of a solver in returning an approximation. Moreover, numerical results from FW are not reported because it fails for most problems. From Tables 6.3–6.5, we can observe that TRSgep and LSTRS generally need more computational effort and are less robust than LTRSR_tCG on these test problems. Overall, for these

TABLE 6.2
Test matrices G.

matrix	n	nnz	sparsity	application
Dubcova1	16129	253009	0.10%	2D/3D problem
EX6	6545	295680	0.69%	combinatorial problem
OPF_3754	15435	141478	0.06%	power network problem
PGPgiantcompo	10680	48632	0.04%	undirected multigraph
Pres_Poisson	14822	715804	0.33%	computational fluid dynamics problem
Si5H12	19896	738598	0.19%	theoretical/quantum chemistry problem
ca-AstroPh	18772	396160	0.11%	undirected graph
ca-HepPh	12008	237010	0.16%	undirected graph
cavity05	1182	32632	2.34%	computational fluid dynamics problem sequence
cavity10	2597	76171	1.13%	computational fluid dynamics problem sequence
chipcool0	20082	281150	0.07%	model reduction problem
comsol	1500	97645	4.34%	structural problem
cvxqp3	17500	114962	0.04%	optimization problem
flowmeter5	9669	67391	0.07%	model reduction problem
fpga_trans_02	1220	7382	0.50%	subsequent circuit simulation problem
fv1	9604	85264	0.09%	2D/3D problem
fxm3_6	5026	94026	0.37%	optimization problem
man_5976	5976	225046	0.63%	structural problem
memplus	17758	99147	0.03%	circuit simulation problem
nd3k	9000	3279690	4.05%	2D/3D problem
nemeth01	9506	725054	0.80%	theoretical/quantum chemistry problem sequence
net25	9520	401200	0.44%	optimization problem
ns3Da	20414	1679599	0.40%	computational fluid dynamics problem
raefsky1	3242	293409	2.79%	computational fluid dynamics problem sequence
raefsky2	3242	293551	2.79%	subsequent computational fluid dynamics problem
rajat06	10922	46983	0.04%	circuit simulation problem
ramage02	16830	2866352	1.01%	computational fluid dynamics problem
stokes64s	12546	140034	0.09%	computational fluid dynamics problem
t2dah	11445	176117	0.13%	model reduction problem
wang3	26064	177168	0.03%	semiconductor device problem

problems, LTRSR_tCG proved to be the most stable and efficient of the four methods we considered. We note in particular that it was able to handle some of the more ill-conditioned problems for which the other methods simply failed.

As the final remark of this section, we point out that LTRSR_tCG now can be regarded as an *accurate method* [1] for both the dense and large-sparse TRS, and can be applied to solve TRS-related applications mentioned in section 1. This also implies that the full-load power of LTRSR_tCG may not necessarily improve the overall performance of the trust-region method (for example, a Fortran implementation TRU in the package GLTR in the library GALAHAD[11]) for the general unconstrained minimization consistently. Indeed, as claimed in [16], "*a more accurate approximation does not appear to significantly reduce the number of function evaluations within a standard trust-region method*," and a proper low accuracy TRS solution from an *approximate method* [1] can usually yield good performance for the overall trust-region method (note that LTRSR_tCG needs extra computational efforts to refine the low accuracy TRS solution). Efforts have been made in [36] for proper stopping criteria used inside GLTR, and our contribution of LTRSR_tCG in this paper was developed along the other line from GLTR.

[11] GALAHAD is a thread-safe library of Fortran 2003 packages for solving nonlinear optimization problems, and its version 2.6 is available at http://www.galahad.rl.ac.uk/.

TABLE 6.3
Numerical results on random dense $H = GG^{\mathrm{T}} - I_n$ and $M = I_n$.

Matrix	$\varkappa$	LTRSR_tCG				TRSgep				FW				LSTRS			
		Prod	*Rres*	cpu	% of cnvg	Prod	*Rres*	cpu	% of cnvg	Prod	*Rres*	cpu	% of cnvg	Prod	*Rres*	cpu	% of cnvg
10	3.9e+4	15	1.3e-11	0.0	100.0	48	4.7e-10	0.0	100.0 ⊛	44	1.7e-3	0.0	90.0 ⊛	212	7.1e-3	0.0	90.0 ⊛
15	1.1e+4	21	4.1e-12	0.0	100.0	200	5.2e-11	0.0	100.0 ⊛	73	5.4e-4	0.0	100.0	265	8.6e-3	0.1	90.0 ⊛
20	1.4e+4	28	1.2e-11	0.0	100.0	352	7.2e-11	0.1	100.0 ⊛	158	1.6e-3	0.0	90.0 ⊛	748	1.0e-3	0.1	100.0 ⊛
25	1.0e+5	37	7.4e-12	0.0	100.0	457	2.0e-10	0.1	90.0 ⊛	361	1.7e-3	0.1	90.0 ⊛	802	7.1e-3	0.1	90.0 ⊛
30	4.4e+4	44	1.9e-11	0.0	100.0	558	2.2e-10	0.1	100.0 ⊛	418	2.1e-3	0.1	90.0 ⊛	1197	1.5e-4	0.2	60.0 ⊛
35	2.2e+4	51	1.3e-11	0.0	100.0	611	8.8e-11	0.1	100.0 ⊛	501	2.5e-3	0.1	90.0 ⊛	1026	7.5e-6	0.2	60.0 ⊛
40	7.3e+4	59	2.3e-11	0.0	100.0	777	4.7e-10	0.1	100.0 ⊛	633	3.4e-3	0.1	80.0	2145	4.8e-11	0.4	70.0
45	5.5e+4	69	1.4e-11	0.0	100.0	989	2.0e-10	0.1	100.0 ⊛	762	1.6e-3	0.1	90.0 ⊛	1884	5.1e-5	0.3	60.0 ⊛
50	9.4e+4	77	2.2e-11	0.0	100.0	1098	2.4e-9	0.2	100.0 ⊛	875	6.1e-3	0.1	100.0	2210	2.1e-11	0.4	80.0
100	1.5e+5	164	2.1e-11	0.1	100.0	2459	7.8e-10	0.4	100.0 ⊛	3004	3.1e-3	0.6	100.0 ⊛	2318	6.1e-12	0.5	60.0
200	1.6e+5	345	2.4e-11	0.3	100.0	6294	3.7e-10	1.1	40.0 ⊛	11722	5.1e-3	2.4	90.0 ⊛	2340	1.0e-12	0.5	10.0
300	1.5e+5	1445	4.9e-11	1.1	100.0	*	*	*	*	*	*	*	*	*	*	*	*
500	2.4e+5	1917	4.4e-11	1.4	100.0	*	*	*	*	*	*	*	*	*	*	*	*
1000	4.4e+5	3020	5.6e-11	4.2	100.0	*	*	*	*	*	*	*	*	*	*	*	*
2000	7.0e+5	5102	5.8e-11	25.0	100.0	*	*	*	*	*	*	*	*	*	*	*	*
3000	6.2e+5	7053	7.2e-11	73.3	100.0	*	*	*	*	*	*	*	*	*	*	*	*
4000	8.2e+5	9056	7.5e-11	161.5	100.0	*	*	*	*	*	*	*	*	*	*	*	*
5000	1.1e+6	10705	7.7e-11	294.0	100.0	*	*	*	*	*	*	*	*	*	*	*	*

TABLE 6.4
Numerical results on random dense $H = GG^{\mathrm{T}} - I_n$ and $M = tridiag(1, 3, 1)$.

Matrix	$\varkappa$	LTRSR_tCG				TRSgep				FW				LSTRS			
		Prod	*Rres*	cpu	% of cnvg	Prod	*Rres*	cpu	% of cnvg	Prod	*Rres*	cpu	% of cnvg	Prod	*Rres*	cpu	% of cnvg
10	2.0e+04	15	3.3e-12	0.0	100.0	48	6.2e-10	0.0	100.0 ⊛	51	4.6e-04	0.0	100.0 ⊛	192	8.7e-03	0.0	100.0 ⊛
15	1.5e+04	21	7.8e-12	0.0	100.0	229	1.5e-10	0.0	100.0 ⊛	68	9.4e-04	0.0	100.0 ⊛	352	1.2e-03	0.1	100.0 ⊛
20	2.3e+05	30	1.2e-11	0.0	100.0	383	2.9e-10	0.1	90.0 ⊛	172	3.2e-03	0.0	100.0 ⊛	599	8.0e-03	0.1	60.0 ⊛
25	3.6e+04	38	8.2e-12	0.0	100.0	510	2.1e-10	0.1	100.0 ⊛	343	1.3e-03	0.1	100.0 ⊛	486	1.7e-03	0.1	80.0 ⊛
30	9.9e+04	47	1.5e-11	0.0	100.0	628	7.4e-10	0.1	90.0 ⊛	496	1.7e-03	0.1	90.0 ⊛	709	2.4e-04	0.1	60.0 ⊛
35	9.9e+04	55	1.5e-11	0.0	100.0	775	1.1e-09	0.1	100.0 ⊛	727	1.5e-03	0.1	100.0 ⊛	1465	2.7e-05	0.3	70.0 ⊛
40	3.3e+04	64	9.6e-12	0.0	100.0	867	2.0e-10	0.1	100.0 ⊛	720	2.7e-03	0.1	90.0	1909	5.6e-05	0.3	90.0 ⊛
45	2.7e+04	74	1.3e-11	0.0	100.0	1030	1.6e-10	0.1	100.0 ⊛	936	2.1e-03	0.2	90.0	2036	3.4e-11	0.4	100.0
50	1.8e+04	82	1.6e-11	0.0	100.0	1108	7.6e-11	0.2	100.0 ⊛	1002	2.1e-03	0.2	100.0	2148	3.1e-11	0.4	70.0
100	1.1e+05	187	2.8e-11	0.1	100.0	2668	6.9e-09	0.4	100.0 ⊛	3243	1.2e-03	0.6	70.0	2242	1.2e-11	0.4	100.0
200	1.6e+05	416	5.7e-11	0.5	100.0	5072	5.6e-10	0.9	20.0 ⊛	11126	2.0e-03	2.1	100.0 ⊛	2264	3.2e-12	0.5	10.0
300	1.6e+05	1461	5.4e-11	1.1	100.0	*	*	*	*	*	*	*	*	*	*	*	*
500	2.7e+05	2092	3.9e-11	1.4	100.0	*	*	*	*	*	*	*	*	*	*	*	*
1000	6.9e+05	3385	4.7e-11	3.9	100.0	*	*	*	*	*	*	*	*	*	*	*	*
2000	7.5e+05	5691	7.1e-11	16.9	100.0	*	*	*	*	*	*	*	*	*	*	*	*
3000	2.2e+06	8203	7.3e-11	50.2	100.0	*	*	*	*	*	*	*	*	*	*	*	*
4000	1.1e+06	9655	8.8e-11	99.1	100.0	*	*	*	*	*	*	*	*	*	*	*	*
5000	1.3e+06	10938	8.9e-11	168.6	100.0	*	*	*	*	*	*	*	*	*	*	*	*

TABLE 6.5
Numerical results on sparse $H = GG^{\mathrm{T}} - I_n$ with G in Table 6.2 and $M = I_n$.

Matrix	LTRSR_tCG			TRSgep			LSTRS		
	Prod	*Rres*	cpu	Prod	*Rres*	cpu	Prod	*Rres*	cpu
Dubcova1	77	2.0e-11	1.0	263	8.9e-15	1.4	240	1.9e-09	0.7⊛
EX6	210	9.5e-11	0.5	1891	7.3e-12	2.7	1983	7.3e-12	2.6
PGPgiantcompo	555	3.8e-11	2.6	4439	3.2e-12	8.2	2108	7.4e-13	2.4
Pres_Poisson	286	4.5e-11	1.5	2164	2.8e-13	7.6	464	5.7e-06	1.3⊛
Si5H12	979	8.9e-11	6.7	10176	3.3e-12	43.0	2038	1.9e-12	6.6
ca-AstroPh	1668	7.2e-11	8.6	17503	1.9e-11	63.9	2129	6.0e-13	7.4
ca-HepPh	1668	4.9e-11	5.9	35483	7.0e-11	77.6	*	*	*
cavity05	363	5.4e-11	0.5	2246	1.1e-12	0.7	516	1.3e-05	0.2⊛
cavity10	313	8.2e-11	0.5	1646	4.0e-13	0.9	385	1.1e-05	0.2⊛
chipcool0	11	5.2e-16	0.0	44	5.5e-16	0.2	18	8.7e-13	0.1
comsol	21	1.2e-12	0.0	80	5.3e-15	0.0	61	9.2e-04	0.1⊛
flowmeter0	4636	9.4e-11	9.5	*	*	*	*	*	*
flowmeter5	5590	9.1e-11	11.0	*	*	*	*	*	*
fpga_trans_02	46	3.4e-11	0.0	236	2.4e-14	0.1	225	5.0e-10	0.1⊛
fv1	82	2.0e-11	0.1	285	2.4e-14	0.4	241	4.6e-09	0.2⊛
fxm3_6	2781	5.7e-11	3.9	7098	2.1e-11	5.8	2132	7.1e-12	1.4
man_5976	1191	4.5e-11	2.9	16274	5.4e-12	19.8	2118	1.4e-12	2.0
memplus	11	7.2e-11	0.1	62	6.7e-16	0.2	26	1.2e-13	0.0
nd3k	1191	8.5e-11	13.4	19434	3.9e-11	168.1	2125	7.6e-12	17.8
nemeth01	1774	6.2e-11	7.7	36901	1.2e-11	109.7	2267	1.9e-12	5.7
net25	106	4.6e-11	0.3	6541	2.4e-11	14.3	2252	1.1e-12	4.0
ns3Da	11	5.2e-11	0.1	44	1.2e-15	0.3	65	2.2e-16	0.4
raefsky1	96	7.8e-11	0.1	424	3.8e-14	0.5	363	1.4e-08	0.4⊛
raefsky2	96	5.0e-11	0.1	376	3.2e-14	0.4	303	3.8e-08	0.3⊛
rajat06	174	8.9e-11	0.3	953	6.6e-14	1.4	515	5.9e-07	0.4⊛
ramage02	4053	7.5e-11	39.8	*	*	*	*	*	*
stokes64s	1138	8.5e-11	3.9	3471	7.7e-13	6.3	1914	1.2e-11	2.1
t2dah	1880	6.1e-11	5.2	*	*	*	*	*	*
wang3	11	7.0e-16	0.1	44	7.1e-16	0.2	18	3.4e-13	0.0

7. Concluding remarks. In this paper, we have developed a restarting approach for the Lanczos method to solve a large scale TRS. This restarting procedure can be implemented upon GLTR, and handles the numerical difficulties in GLTR, particularly for an ill-conditioned TRS.

Different from the classical restarting procedures of Krylov subspace methods for linear systems, restarting for TRS should respect the constraint. We have formulated a restarting framework in section 3.3. To make it more efficient, we integrated the nested structure of GMRESR, and also proposed a specific Krylov subspace to accelerate the convergence of this nested restarting scheme. Convergence analysis was performed and numerical testings were carried out to illustrate the performance of LTRSR_tCG in various situations.

We remark finally that our LTRSR_tCG consists of two basic technical ingredients: One is the introduction of the Krylov subspace $\mathcal{K}_{m_{i+1}}(M^{-1}H, \boldsymbol{s}_i)$ which aims at alleviating the inaccuracy of the Lagrange multiplier, and the other is the nested structure which is able to improve the efficiency of the restarting procedure. When dealing with the minimization problem subject to multiple constraints, one can introduce multiple particular Krylov subspaces to handle the inaccuracy of the corresponding Lagrange multipliers, and a proper implementation together with a nested restarting strategy might lead to efficient Krylov subspace type methods for solving a specific constrained minimization problem.

Acknowledgments. The authors are grateful to the associate editor and anonymous referees for their very careful reading, useful comments, and suggestions which significantly improved the presentation of the paper. They also thank Prof. Ren-Cang Li for suggesting the nested GMRES method and useful comments.

REFERENCES

[1] S. Adachi, S. Iwata, Y. Nakatsukasa, and A. Takeda, *Solving the trust-region subproblem by a generalized eigenvalue problem*, SIAM J. Optim., 27 (2017), pp. 269–291.
[2] A. R. Conn, N. I. M. Gould, and P. L. Toint, *Trust-Region Methods*, MOS-SIAM Ser. Optim. 1, SIAM, Philadelphia, 2000.
[3] J. K. Cullum and W. E. Donath, *A block Lanczos algorithm for computing the q algebraically largest eigenvalues and a corresponding eigenspace of large, sparse, real symmetric matrices*, in Decision and Control Including the 13th Symposium on Adaptive Processes, IEEE, Washington, DC, 1974, pp. 505–509.
[4] T. Davis and Y. Hu, *The University of Florida sparse matrix collection*, ACM Trans. Math. Software, 38 (2011), pp. 1–25.
[5] E. de Sturler, *Nested Krylov methods based on GCR*, J. Comput. Appl. Math., 67 (1996), pp. 15–41.
[6] J. W. Demmel, *Applied Numerical Linear Algebra*, SIAM, Philadelphia, 1997.
[7] H. A. Van der Vorst and C. Vuik, *GMRESR: A family of nested GMRES methods*, J. Numer. Linear Algebra. Appl., 1 (1994), pp. 369–386.
[8] S. C. Eisenstat, H. Elman, and M. H. Schultz, *Variational iterative methods for nonsymmetric systems of linear equations*, SIAM J. Numer. Anal., 20 (1983), pp. 345–357.
[9] C. Fortin and H. Wolkowicz, *The trust region subproblem and semidefinite programming*, Optim. Methods Softw., 19 (2004), pp. 41–67.
[10] C. Fortin and H. Wolkowicz, *Trust Region Subroutine Algorithm*, Algorithm with Links to Documentation, http://www.math.uwaterloo.ca/~hwolkowi/henry/software/trustreg.d (2010).
[11] W. Gander, G. H. Golub, M. Heath, and Urs von Matt, *A constrained eigenvalue problem*, Linear Algebra Appl., 114/115 (1989), pp. 815–839.
[12] D. M. Gay, *Computing optimal locally constrained steps*, SIAM J. Sci. Statist. Comput., 2 (1981), pp. 186–197.
[13] G. H. Golub and M. Heath, *Generalized cross-validation as a method for choosing a good ridge parameter*, Technometrics, 21 (1979), pp. 215–223.
[14] G. H. Golub and R. R. Underwood, *The block Lanczos method for computing eigenvalues*, in Mathematical Software III, J. R. Rick, ed., Academic Press, New York, 1977, pp. 361–377.
[15] G. H. Golub and U. von Matt, *Quadratically constrained least squares and quadratic problems*, Numer. Math., 59 (1991), pp. 561–580.
[16] 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 (1999), pp. 504–525.
[17] N. I. M. Gould, D. P. Robinson, and H. S. Thorne, *On solving trust-region and other regularised subproblems in optimization*, Math. Program. Comput., 2 (2010), pp. 21–57.
[18] W. W. Hager, *Minimizing a quadratic over a sphere*, SIAM J. Optim., 12 (2001), pp. 188–208.
[19] W. W. Hager and Y. Krylyuk, *Graph partitioning and continuous quadratic programming*, SIAM J. Discrete Math., 12 (1999), pp. 500–523.
[20] A. Imakura, R.-C. Li, and S.-L. Zhang, *Locally optimal and heavy ball GMRES methods*, Japan J. Indust. Appl. Math., 33 (2016), pp. 471–499.
[21] J. J. Moré and D. C. Sorensen, *Computing a trust region step*, SIAM J. Sci. Statist. Comput., 4 (1983), pp. 553–572.
[22] J. Nocedal and S. Wright, *Numerical Optimization*, 2nd ed., Springer, New York, 2006.
[23] B. N. Parlett, *The Symmetric Eigenvalue Problem*, Classics Appl. Math. 20, SIAM, Philadelphia, 1998.
[24] R. Rendl and H. Wolkowicz, *A semidefinite framework for trust region subproblems with applications to large scale minimization*, Math. Program., 77 (1997), pp. 273–299.
[25] 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 (2000), pp. 611–646.
[26] 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. Software, 34 (2008), pp. 1–28.

[27] M. Rojas and D. C. Sorensen, *A trust-region approach to the regularization of large-scale discrete forms of ill-posed problems*, SIAM J. Sci. Comput., 23 (2002), pp. 1842–1860.
[28] Y. Saad, *Iterative Methods for Linear Systems*, PWS, Boston, 1996.
[29] D. C. Sorensen, *Minimization of a large-scale quadratic function subject to a spherical constraint*, SIAM J. Optim., 7 (1997), pp. 141–161.
[30] . Steihaug, *The conjugate gradient method and trust regions in large scale optimization*, SIAM J. Numer. Anal., 20 (1983), pp,. 626–637.
[31] A. Tarantola, *Inverse Problem Theory*, Elsevier, Amsterdam, 1987.
[32] A. N. Tikhonov, *Regularization of incorrectly posed problems*, Soviet Math. Dokl., 4 (1963), pp. 1624–1627.
[33] P. L. Toint, *Towards an efficient sparsity exploiting Newton method for minimization*, in Sparse Matrices and Their Uses, I. S. Duff, ed., Academic Press, London, 1981, pp. 57–88.
[34] L. Xu, W. Li, and D. Schuurmans, *Fast normalized cut with linear constraints*, in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2009, IEEE, Piscataway, NJ, 2009, pp. 2866–2873.
[35] W. H. Yang, L.-H. Zhang, and R. Y. Song, *Optimality conditions of the nonlinear programming on Riemannian manifolds*, Pacific J. Optim., 10 (2014), pp. 415–434.
[36] L.-H. Zhang, C. Shen, and R.-C. Li, *On the generalized Lanczos trust-region method*, SIAM J. Optim., 27 (2017), pp. 2110–2142.
[37] L.-H. Zhang, C. Shen, W. H. Yang, and J. J. Júdice, *A Lanczos method for large-scale extreme Lorentz eigenvalue problems*, SIAM J. Matrix Anal. Appl., 39 (2018), pp. 611–631.
[38] L.-H. Zhang, W. H. Yang, C. Shen, and J. Feng, *Error bounds of the Lanczos approach for the trust-region subproblem*, Front. Math. China, 13 (2018), pp. 459–481.