Electronic Research Archive

https://www.aimspress.com/journal/era

ERA, 34(10): 7565–7582.
DOI: 10.3934/era.2026326
Received: 24 May 2026
Revised: 09 August 2026
Accepted: 24 August 2026
Published: 02 September 2026

Research article

A minimal residual method for large-scale trust-region subproblem

Yusen Zhang and Bo Feng*

School of Mathematics, Yangzhou University, Yangzhou 225000, China

* **Correspondence:** Email: bofeng@yzu.edu.cn.

Abstract: This paper was devoted to solving the large-scale trust-region subproblem (TRS), a core subproblem in nonlinear optimization with extensive applications in scientific computing fields such as the Lorentz eigenvalue problem, Tikhonov regularization, nonlinear least squares, and so on. The generalized Lanczos trust-region (GLTR) method is one of the most popular iterative algorithms for large-scale TRS. In this method, the approximate solution converged more slowly than the approximate Lagrange multiplier. Inspired by this characteristic and the generalized minimal residual method (GMRES) and minimal residual method (MINRES), we proposed a minimal residual method based on the GLTR framework (MRTR). The MRTR method fixed the approximate Lagrange multiplier from GLTR and sought the approximate solution that minimized the residual norm within the Krylov subspace. We established some theoretical results to characterize the relationship between the approximate solutions of MRTR and GLTR. The experimental results showed that the MRTR method outperformed the GLTR method in terms of iteration numbers and CPU running time, and its residual norm convergence curve was smoother with better numerical stability, while achieving approximate optimal values with the same high precision as GLTR.

Keywords: trust-region subproblem; generalized Lanczos trust-region (GLTR) method; Krylov subspace; minimal residual method

1. Introduction

In this paper, we focus on the following large-scale *trust-region subproblem* (TRS) [1, 2]:

$$\min_{\|\boldsymbol{x}\|_2 \le \Delta} \left\{ f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T A\boldsymbol{x} + \boldsymbol{x}^T \boldsymbol{g} \right\}, \tag{1.1}$$

where $A \in \mathbb{R}^{n\times n}$ is a symmetric matrix, $\boldsymbol{g} \in \mathbb{R}^n$ is a nonzero vector, $\|\cdot\|_2$ denotes the 2-norm of a vector (or matrix), and $\Delta > 0$ is the trust-region radius. Trust-region methods [1, 2] are a widely used technique for solving nonlinear optimization problems that aim to minimize objective function. Each

iteration of such methods entails finding a solution to the TRS (1.1). Within the trust-region framework, the objective function of TRS (1.1) serves as a quadratic approximation of objective function around the current iterate $\tilde{\boldsymbol{x}}$, where A denotes the Hessian matrix and $\boldsymbol{g}$ stands for the gradient of objective function evaluated at $\tilde{\boldsymbol{x}}$. In addition, the TRS (1.1) finds wide applications in various fields, including the Lorentz eigenvalue problem [3], graph partitioning problems [4], the Levenberg–Marquardt algorithm for nonlinear least squares [2], and the Tikhonov regularization [5, 6].

Numerous algorithms have been developed to address large-scale TRS (1.1). Gould et al. proposed a modified version of the Moré-Sorensen method using Taylor series expansion [7], while Steihaug and Toint [8, 9] introduced a truncated conjugate gradient (tCG) approach. In [10], Gander et al. addressed the TRS (1.1) from the perspective of an eigenproblem. Building on this approach, Adachi, Iwata, Nakatsukasa, and Takeda extended the strategy to weighted-norm TRS in [11], where the problem can be transformed into a generalized eigenvalue problem of size $2n$. The generalized Lanczos trust-region (GLTR) method [12] leverages projection techniques to reduce the problem size. Other notable methods include a nested Lanczos method [13] (regarded as an accelerated variant of GLTR), a matrix-free algorithm [5, 6], and a method based on semi-definite programming (SDP) [14]. For more details, the reader is referred to [15–20].

The GLTR method [12] is among the most widely used techniques for solving large-scale TRS (1.1) in the *easy case* (for the definition of the *easy case*, please see Section 2). The principle of GLTR consists of the Lanczos process combined with the Rayleigh-Ritz procedure. Analogous to the classical Lanczos method for eigenvalue problems (see, e.g., [21]), for solving large-scale TRS, the original problem (1.1) is first projected onto the Krylov subspace to yield a reduced-size TRS, which is subsequently solved by an efficient and accurate method. The convergence of GLTR has been investigated in [22–27]. From these convergence results, the convergence rate of the approximate Lagrange multiplier is significantly higher than that of the approximate solution in the GLTR method. Inspired by this fact and the generalized minimal residual (GMRES)/minimal residual (MINRES) methods [28], we propose a minimal residual method for solving large-scale TRS problems based on the GLTR method. This method fixes the approximate Lagrange multiplier obtained from the GLTR method and seeks the approximate solution that minimizes the residual norm within the Krylov subspace, replacing the approximate solution yielded by the GLTR to approximate the global solution. We show through several numerical examples that the proposed method yields a certain improvement over the GLTR method. Furthermore, it can be seen from the convergence curves of the residual norm that the convergence curve of the new method is smoother than that of the GLTR method (see Section 4).

The paper is organized as follows. In Section 2, we first review the theoretical characterization of the global solution to the TRS (1.1) and classify the *easy case* and *hard case* of the problem, then give a detailed introduction to the GLTR method, including its basic principle, two-phase iteration process, and existing convergence results. In Section 3, inspired by the GMRES/MINRES methods and the faster convergence rate of the Lagrange multiplier in the GLTR method, we propose a minimal residual method for large-scale TRS (MRTR), derive the explicit form of the approximate solution of the new method, and establish key theoretical results to analyze the relationship between the approximate solutions of MRTR and GLTR. Section 4 presents some numerical experiments, where we test the performance of the MRTR method on both constructed diagonal matrices and real sparse symmetric matrices from the University of Florida Sparse Matrix Collection, and compare the number of iterations, CPU running time, residual norm convergence curves, and approximate optimal values with those of

the GLTR method to verify the numerical efficiency and stability of the proposed method. Finally, concluding remarks are drawn in Section 5.

Throughout this paper, the transpose of a matrix or vector is denoted by $(\cdot)^T$; $\mathcal{N}(A)$ stands for the null space of matrix A; $\|\cdot\|$ represents the Euclidean norm of a matrix or vector. We use $\mathbf{0}$ and I to denote the zero vector (matrix) and identity matrix, respectively, with their dimensions being self-evident from the context. Let $\boldsymbol{e}_i$ denote the i-th column of I. Additionally, $A \succcurlyeq \mathbf{0}$ (resp., $A \succ \mathbf{0}$) indicates that A is a symmetric positive semi-definite (resp., positive definite) matrix.

2. Preliminaries

In this section, we briefly introduce the solutions to the TRS (1.1), and the GLTR method [12].

2.1. Solutions of the TRS (1.1)

A global solution to the TRS (1.1) is characterized by the following theorem.

Theorem 2.1. *[1, 17, 18] The vector $\boldsymbol{x}_{\text{opt}}$ is a global optimal solution of the trust-region problem (1.1), if, and only if, $\|\boldsymbol{x}_{\text{opt}}\| \le \Delta$ and there exists Lagrange multiplier $\lambda_{\text{opt}} \ge 0$ such that*

$$(A + \lambda_{\text{opt}} I)\boldsymbol{x}_{\text{opt}} = -\boldsymbol{g},\ \lambda_{\text{opt}}(\Delta - \|\boldsymbol{x}_{\text{opt}}\|) = 0 \ \text{ and } \ A + \lambda_{\text{opt}} I \succcurlyeq \mathbf{0}. \tag{2.1}$$

Let $s_{\min}$ be the smallest eigenvalue of A. In particular, the TRS (1.1) admits two scenarios [2, 17, 29]:

Easy case: $\lambda_{\text{opt}} \ge \max\{0, -s_{\min}\}$. In this case, $A + \lambda_{\text{opt}} I \succ \mathbf{0}$, and the TRS (1.1) has a *unique* solution given by:

$$\boldsymbol{x}_{\text{opt}} = -(A + \lambda_{\text{opt}} I)^{-1}\boldsymbol{g}.$$

Hard case: $\lambda_{\text{opt}} = -s_{\min}$. In this case, there exist multiple solutions that can be expressed as follows:

$$\boldsymbol{x}_{\text{opt}} = -(A + \lambda_{\text{opt}} I)^{\dagger}\boldsymbol{g} + \sqrt{\Delta^2 - \|(A + \lambda_{\text{opt}} I)^{\dagger}\boldsymbol{g}\|^2} \cdot \boldsymbol{u},$$

where the superscript † denotes the Moore-Penrose pseudoinverse of a matrix and $\boldsymbol{u} \in \mathcal{N}(A - s_{\min} I)$ with $\|\boldsymbol{u}\|_2 = 1$.

It is evident that when $A \succ \mathbf{0}$, the global solution $\boldsymbol{x}_{\text{opt}}$ must take one of two forms: either $\boldsymbol{x}_{\text{opt}} = -A^{-1}\boldsymbol{g}$ (corresponding to $\lambda_{\text{opt}} = 0$) or $\boldsymbol{x}_{\text{opt}} = -(A + \lambda_{\text{opt}} I)^{-1}\boldsymbol{g}$ (corresponding to $\lambda_{\text{opt}} > 0$) lying on the boundary. Consequently, the *hard case* arises if, and only if, $s_{\min} \le 0$. Throughout this paper, we assume that the TRS is the *easy case*.

2.2. The GLTR method

The GLTR method proposed by Gould et al. [12] is one of the most commonly used approaches for solving large-scale TRS (1.1) in the *easy case*. Specifically, the GLTR method yields an approximate solution to the original TRS by solving the following problem:

$$\min_{\|\boldsymbol{x}\| \le \Delta,\ \boldsymbol{x} \in \mathcal{K}_k} \left\{ f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T A \boldsymbol{x} + \boldsymbol{x}^T \boldsymbol{g} \right\}, \tag{2.2}$$

where $\mathcal{K}_k$ denotes the Krylov subspace

$$\mathcal{K}_k := \mathcal{K}_k(A, \boldsymbol{g}) = \text{span}\{\boldsymbol{g}, A\boldsymbol{g}, \ldots, A^{k-1}\boldsymbol{g}\} = \left\{ \sum_{i=0}^{k-1} \alpha_i A^i \boldsymbol{g} \,\middle|\, \alpha_0, \alpha_1, \ldots, \alpha_{k-1} \in \mathbb{R} \right\}.$$

We can obtain an orthonormal basis $Q_k = (\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_k)$ for $\mathcal{K}_k$ by the Lanczos process, which can be written in matrix form [28]

$$AQ_k = Q_k T_k + \beta_k \boldsymbol{q}_{k+1} \boldsymbol{e}_k^T \;\text{ with }\; \boldsymbol{g} = \beta_0 \boldsymbol{q}_1, \text{ and } \beta_0 = \|\boldsymbol{g}\|, \tag{2.3}$$

where $\boldsymbol{e}_i$ is the i-th column of the identity matrix, and

$$T_k = Q_k^T A Q_k = \begin{pmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & \beta_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \beta_{k-2} & \alpha_{k-1} & \beta_{k-1} \\ & & & \beta_{k-1} & \alpha_k \end{pmatrix} \in \mathbb{R}^{k\times k}.$$

Let

$$\boldsymbol{y}_k = \arg\min_{\|\boldsymbol{y}\|\le\Delta} \left\{ \frac{1}{2} \boldsymbol{y}^T T_k \boldsymbol{y} + \beta_0 \boldsymbol{y}^T \boldsymbol{e}_1 \right\}. \tag{2.4}$$

It can be easily verified that

$$\boldsymbol{x}_k := Q_k \boldsymbol{y}_k = \arg\min_{\|\boldsymbol{x}\|\le\Delta,\ \boldsymbol{x}\in\mathcal{K}_k} f(\boldsymbol{x}).$$

Therefore, we obtain the solution $\boldsymbol{x}_k$ to Problem (2.2), which can be used as an approximation to $\boldsymbol{x}_{\text{opt}}$.

Since T_k is a tridiagonal and irreducible matrix, the TRS (2.4) is always in the *easy case* [12, Theorem 5.3], i.e., the solution $\boldsymbol{y}_k$ is unique. By Theorem 2.1, there exists a Lagrange multiplier $\lambda_k \ge 0$, such that

$$\boldsymbol{y}_k = -\beta_0 (T_k + \lambda_k I)^{-1} \boldsymbol{e}_1 \;\text{ and }\; \boldsymbol{x}_k = -\beta_0 Q_k (T_k + \lambda_k I)^{-1} \boldsymbol{e}_1. \tag{2.5}$$

So we have from (2.3) that

$$(A + \lambda_k I)\boldsymbol{x}_k + \boldsymbol{g} = (\beta_k \boldsymbol{e}_k^T \boldsymbol{y}_k) \boldsymbol{q}_{k+1}. \tag{2.6}$$

The convergence of $\boldsymbol{x}_k$, λ_k, and $\|(A + \lambda_k I)\boldsymbol{x}_k + \boldsymbol{g}\|$ has been thoroughly investigated [24, 27]:

Theorem 2.2. *Suppose that TRS* (1.1) *is in the easy case and* $\|\boldsymbol{x}_{\text{opt}}\| = \|\boldsymbol{x}_k\| = \Delta$. *Then*

$$\frac{\|\boldsymbol{x}_k - \boldsymbol{x}_{\text{opt}}\|}{\Delta} \le 4\sqrt{\kappa} \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k,$$

$$\|(A + \lambda_k I)\boldsymbol{x}_k + \boldsymbol{g}\| \le 2\|A_{\text{opt}}\| \sqrt{\Delta^2 + \|(I - Q_k Q_k^T)\boldsymbol{x}_{\text{opt}}\|^2} \cdot \Big(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \Big)^k,$$

$$0 \le \lambda_{\text{opt}} - \lambda_k \le 2 s_k \cdot \left(1 + \frac{2k\sqrt{\kappa}}{\|A_{\text{opt}}\|} \left(\frac{\|\boldsymbol{x}_{\text{opt}}\|_{A_{\text{opt}}}}{\Delta} \right)^2 \right) \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^{2k},$$

where κ *denotes the spectral condition number of* $A + \lambda_{\text{opt}} I$ *and*

$$s_k = \frac{\Delta^2}{\|\boldsymbol{g}\|^2 \cdot \boldsymbol{e}_1^T (T_k + \lambda_{\text{opt}} I)^{-3} \boldsymbol{e}_1}.$$

The GLTR can be divided into two phases. In this first phases, $\boldsymbol{x}_k$ can recurs from the conjugate-gradient (CG) iteration [12]:

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \rho_k \boldsymbol{p}_k,\ k \geq 1$$

where

$$\boldsymbol{g}_1 = -\boldsymbol{p}_1 = \boldsymbol{g},\ \boldsymbol{x}_1 = \boldsymbol{0},\ \rho_k = \frac{\|\boldsymbol{g}_k\|^2}{\boldsymbol{p}_k^T A \boldsymbol{p}_k},\ \boldsymbol{g}_{k+1} = \boldsymbol{g}_k + \rho_k A \boldsymbol{p}_k,$$
$$\tau_k = \frac{\|\boldsymbol{g}_{k+1}\|^2}{\|\boldsymbol{g}_k\|^2},\ \boldsymbol{p}_{k+1} = -\boldsymbol{g}_{k+1} + \tau_k \boldsymbol{p}_k.$$

Moreover, $\boldsymbol{q}_k = \frac{\sigma_k \boldsymbol{g}_k}{\|\boldsymbol{g}_k\|}$ where $\sigma_k = -\operatorname{sign}(\rho_{k-1})\sigma_{k-1}$, $\sigma_1 = 1$, and

$$T_k = \begin{pmatrix} \frac{1}{\rho_1} & \frac{\sqrt{\tau_1}}{|\rho_1|} & & & \\ \frac{\sqrt{\tau_1}}{|\rho_1|} & \frac{1}{\rho_2} + \frac{\tau_1}{\rho_1} & \frac{\sqrt{\tau_2}}{|\rho_2|} & & \\ & \ddots & \ddots & \ddots & \\ & & \frac{\sqrt{\tau_{k-2}}}{|\rho_{k-2}|} & \frac{1}{\rho_{k-1}} + \frac{\tau_{k-2}}{\rho_{k-2}} & \frac{\sqrt{\tau_{k-1}}}{|\rho_{k-1}|} \\ & & & \frac{\sqrt{\tau_{k-1}}}{|\rho_{k-1}|} & \frac{1}{\rho_k} + \frac{\tau_{k-1}}{\rho_{k-1}} \end{pmatrix}. \tag{2.7}$$

In the CG iterations, $\|\boldsymbol{x}_k\|$ increases monotonically. As a result, the CG iteration terminates if $\|\boldsymbol{x}_{\text{opt}}\| < \Delta$, or $\|\boldsymbol{x}_k\| > \Delta$, or $\boldsymbol{p}_k^T A \boldsymbol{p}_k < 0$. If $\boldsymbol{x}_{\text{opt}}$ is not obtained after the CG iterations terminate, GLTR proceeds to the second phase, where it needs to solve a smaller-scale TRS (2.4) to obtain $\boldsymbol{y}_k$. In short, GLTR is divided into the following two phases:

(i) $k = 1, 2, \ldots, k_{\text{CG}}$: $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \rho_k \boldsymbol{p}_k$ (CG iteration);

(ii) $k = k_{\text{CG}} + 1, k_{\text{CG}} + 2, \ldots$ until convergence: $\boldsymbol{x}_k = Q_k \boldsymbol{y}_k$.

When $A \succ \boldsymbol{0}$ and $\|A^{-1}\boldsymbol{g}\| \leq \Delta$ ($\lambda_{\text{opt}} = 0$), the GLTR method reduces to the standard CG method and $\boldsymbol{x}_{\text{opt}} = -A^{-1}\boldsymbol{g}$. In the following, we assume that the CG iterations do not solve TRS (1.1), meaning that the second phase is activated, i.e., $\lambda_{\text{opt}} > 0$ and $\|\boldsymbol{x}_{\text{opt}}\| = \Delta$. The GLTR algorithm is listed in Algorithm 1 (see [12] in detail).

3. A minimal residual method for the TRS

Inspired by the GMRES/MINRES methods [21, 28], we seek $(\widehat{\lambda}, \widehat{\boldsymbol{x}})$ to minimize the residual norm $\|(A + \widehat{\lambda} I)\widehat{\boldsymbol{x}} + \boldsymbol{g}\|$ in the *second phase*, where $\widehat{\boldsymbol{x}} \in \mathcal{K}_k$. Note that in the GLTR method, the convergence rate of λ_k is faster than that of $\boldsymbol{x}_k$ (see Theorem 2.2). Therefore, in our method, we fix λ_k and seek $\widetilde{\boldsymbol{x}}_k \in \mathcal{K}_k$ such that

$$\widetilde{\boldsymbol{x}}_k = \arg\min_{\boldsymbol{x} \in \mathcal{K}_k} \|(A + \lambda_k I)\boldsymbol{x} + \boldsymbol{g}\|.$$

Algorithm 1 The GLTR algorithm.

Require: A, $\boldsymbol{g}$, Δ, tol, and $It_{\max}$.
Ensure: $\boldsymbol{x}_k$ and $f(\boldsymbol{x}_k)$

$\beta_0 = \|\boldsymbol{g}\|$, $\boldsymbol{g}_1 = -\boldsymbol{p}_1 = \boldsymbol{g}$, $\boldsymbol{x}_1 = \boldsymbol{0}$, $k = 1$
while $\frac{\|\boldsymbol{g}_k\|}{\|\boldsymbol{g}\|} \geq tol$ **do**
  $\rho_k = \frac{\|\boldsymbol{g}_k\|^2}{\boldsymbol{p}_k^T A \boldsymbol{p}_k}$
  Obtain T_k from T_{k-1} using (2.7)
  if $\rho_k \leq 0$ or $\|\boldsymbol{x}_k + \rho_k \boldsymbol{p}_k\| > \Delta$ **then**
    $k_{\mathrm{CG}} = k$; Go to Step 15
    break
  end if
  $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \rho_k \boldsymbol{p}_k$
  $\boldsymbol{g}_{k+1} = \boldsymbol{g}_k + \rho_k A \boldsymbol{p}_k$
  $\tau_k = \frac{\|\boldsymbol{g}_{k+1}\|^2}{\|\boldsymbol{g}_k\|^2}$
  $\boldsymbol{p}_{k+1} = -\boldsymbol{g}_{k+1} + \tau_k \boldsymbol{p}_k$
  $k = k + 1$
end while
for $k = k_{\mathrm{CG}} + 1, \ldots, It_{\max}$ **do**
  $\rho_k = \frac{\|\boldsymbol{g}_k\|^2}{\boldsymbol{p}_k^T A \boldsymbol{p}_k}$
  Solve TRS (2.4) to obtain $\boldsymbol{y}_k$
  if $\frac{\beta_k |\boldsymbol{e}_k^T \boldsymbol{y}_k|}{\|\boldsymbol{g}\|} \leq tol$ **then**
    Output $\boldsymbol{x}_k = Q_k \boldsymbol{y}_k$; **return**
  end if
  $\boldsymbol{g}_{k+1} = \boldsymbol{g}_k + \rho_k A \boldsymbol{p}_k$
  $\tau_k = \frac{\|\boldsymbol{g}_{k+1}\|^2}{\|\boldsymbol{g}_k\|^2}$
  $\boldsymbol{p}_{k+1} = -\boldsymbol{g}_{k+1} + \tau_k \boldsymbol{p}_k$
end for

We use $\widetilde{\boldsymbol{x}}_k$ instead of $\boldsymbol{x}_k$ as an approximation for $\boldsymbol{x}_{\text{opt}}$. Let

$$\widetilde{T}_k = \begin{pmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & \beta_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \beta_{k-2} & \alpha_{k-1} & \beta_{k-1} \\ & & & \beta_{k-1} & \alpha_k \\ & & & & \beta_k \end{pmatrix} = \begin{pmatrix} T_k \\ \beta_k \boldsymbol{e}_k^T \end{pmatrix}, \quad \widetilde{I}_k = \begin{pmatrix} I_k \\ \boldsymbol{0} \end{pmatrix} \in \mathbb{R}^{(k+1)\times k}.$$

Hence,

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathcal{K}_k} \|(A+\lambda_k I)\boldsymbol{x}+\boldsymbol{g}\| &= \min_{\boldsymbol{y}} \|(A+\lambda_k I)Q_k\boldsymbol{y}+\beta_0 Q_{k+1}\boldsymbol{e}_1\| \\ &\overset{(2.3)}{=} \min_{\boldsymbol{y}} \|Q_{k+1}[(\widetilde{T}_k+\lambda_k\widetilde{I}_k)\boldsymbol{y}+\beta_0\boldsymbol{e}_1]\| \\ &= \min_{\boldsymbol{y}} \|(\widetilde{T}_k+\lambda_k\widetilde{I}_k)\boldsymbol{y}+\beta_0\boldsymbol{e}_1\|. \end{aligned} \tag{3.1}$$

Let

$$\widetilde{\boldsymbol{y}}_k = -\beta_0(\widetilde{T}_k+\lambda_k\widetilde{I}_k)^{\dagger}\boldsymbol{e}_1 = \arg\min_{\boldsymbol{y}} \|(\widetilde{T}_k+\lambda_k\widetilde{I}_k)\boldsymbol{y}+\beta_0\boldsymbol{e}_1\|.$$

Thus,

$$\widetilde{\boldsymbol{x}}_k = Q_k\widetilde{\boldsymbol{y}}_k = \arg\min_{\boldsymbol{x}\in\mathcal{K}_k} \|(A+\lambda_k I)\boldsymbol{x}+\boldsymbol{g}\|,$$

is taken as an approximation to $\boldsymbol{x}_{\text{opt}}$. The proposed algorithm (MRTR) is given in Algorithm 2. The difference between the MRTR method and the GLTR method lies in Lines 17–20.

3.1. Main results

In this section, we provide some theoretical analysis for the MRTR algorithm. To this end, we need the following lemma.

Lemma 3.1. *For $k > 1$, we have*

$$|\boldsymbol{e}_1^T\,(T_k+\lambda_k I_k)^{-3}\,\boldsymbol{e}_k| \le Cq^{k-1},$$

where C is a constant and $0 < q < 1$ depends only on function $f(x) = x^{-3}$.

Proof. Note that $T_k + \lambda_k I_k$ is a symmetric positive definite matrix with bandwidth 1; thus, the conclusion follows directly from [30, p. 317, Theorem 14.1]. □

The following result establishes the relationship between $\boldsymbol{x}_k$ and $\widetilde{\boldsymbol{x}}_k$.

Theorem 3.1. *For $k \ge 1$, we have*

$$\|\boldsymbol{x}_k - \widetilde{\boldsymbol{x}}_k\| \le 2\xi_k\Delta\left(\frac{\sqrt{\kappa_k}-1}{\sqrt{\kappa_k}+1}\right)^{k-1}, \tag{3.2a}$$

$$\left|\|\boldsymbol{x}_k\| - \|\widetilde{\boldsymbol{x}}_k\|\right| \le \frac{4(\xi_k^2\Delta^2+\zeta_k)}{\|\boldsymbol{x}_k\|+\|\widetilde{\boldsymbol{x}}_k\|}\left(\frac{\sqrt{\kappa_k}-1}{\sqrt{\kappa_k}+1}\right)^{k-1}\cdot q^{k-1}, \tag{3.2b}$$

Algorithm 2 A minimal residual method for the TRS (1.1) (MRTR).

Input: A, $\boldsymbol{g}$, Δ, tol, and $It_{\max}$.
Output: $\widetilde{\boldsymbol{x}}_k$ and $f(\widetilde{\boldsymbol{x}}_k)$
$\beta_0 = \|\boldsymbol{g}\|$, $\boldsymbol{g}_1 = -\boldsymbol{p}_1 = \boldsymbol{g}$, $\boldsymbol{x}_1 = \boldsymbol{0}$, $k = 1$
while $\frac{\|\boldsymbol{g}_k\|}{\|\boldsymbol{g}\|} \geq tol$ **do**
  $\rho_k = \frac{\|\boldsymbol{g}_k\|^2}{\boldsymbol{p}_k^T A \boldsymbol{p}_k}$
  Obtain T_k from T_{k-1} using (2.7)
  if $\rho_k \leq 0$ or $\|\boldsymbol{x}_k + \rho_k \boldsymbol{p}_k\| > \Delta$ **then**
    $k_{\mathrm{CG}} = k$; Go to Step 15
    break
  end if
  $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \rho_k \boldsymbol{p}_k$
  $\boldsymbol{g}_{k+1} = \boldsymbol{g}_k + \rho_k A \boldsymbol{p}_k$
  $\tau_k = \frac{\|\boldsymbol{g}_{k+1}\|^2}{\|\boldsymbol{g}_k\|^2}$
  $\boldsymbol{p}_{k+1} = -\boldsymbol{g}_{k+1} + \tau_k \boldsymbol{p}_k$
  $k = k + 1$
end while
for $k = k_{\mathrm{CG}} + 1, \ldots, It_{\max}$ **do**
  $\rho_k = \frac{\|\boldsymbol{g}_k\|^2}{\boldsymbol{p}_k^T A \boldsymbol{p}_k}$
  Solve TRS (2.4) to obtain λ_k and compute $\widetilde{\boldsymbol{y}}_k = -\beta_0 (\widetilde{T}_k + \lambda_k \widetilde{I}_k)^{\dagger} \boldsymbol{e}_1$
  if $\frac{\|(\widetilde{T}_k + \lambda_k \widetilde{I}_k)\widetilde{\boldsymbol{y}}_k + \beta_0 \boldsymbol{e}_1\|}{\|\boldsymbol{g}\|} \leq tol$ **then**
    Output $\widetilde{\boldsymbol{x}}_k = Q_k \widetilde{\boldsymbol{y}}_k$; **return**
  end if
  $\boldsymbol{g}_{k+1} = \boldsymbol{g}_k + \rho_k A \boldsymbol{p}_k$
  $\tau_k = \frac{\|\boldsymbol{g}_{k+1}\|^2}{\|\boldsymbol{g}_k\|^2}$
  $\boldsymbol{p}_{k+1} = -\boldsymbol{g}_{k+1} + \tau_k \boldsymbol{p}_k$
end for

where

$$\xi_k = \frac{\beta_k^2 \left\| (T_k + \lambda_k I_k)^{-2} \boldsymbol{e}_k \right\|}{1 + \beta_k^2 \boldsymbol{e}_k^T (T_k + \lambda_k I_k)^{-2} \boldsymbol{e}_k} \text{ and } \zeta_k = \frac{C\beta_k^2 \beta_0 \Delta}{1 + \beta_k^2 \boldsymbol{e}_k^T (T_k + \lambda_k I_k)^{-2} \boldsymbol{e}_k}.$$

Proof. We first prove (3.2a). Recall that TRS (2.4) is in the *easy case*. Consequently, $T_k + \lambda_k I_k \succ \mathbf{0}$. This immediately implies that

$$(\widetilde{T}_k + \lambda_k \widetilde{I}_k)^T (\widetilde{T}_k + \lambda_k \widetilde{I}_k) = (T_k + \lambda_k I_k)^2 + \beta_k^2 \boldsymbol{e}_k \boldsymbol{e}_k^T \succ \mathbf{0}. \tag{3.3}$$

We further have the identity

$$\begin{aligned}
&(\widetilde{T}_k + \lambda_k \widetilde{I}_k)^\dagger \boldsymbol{e}_1 = \left[(\widetilde{T}_k + \lambda_k \widetilde{I}_k)^T (\widetilde{T}_k + \lambda_k \widetilde{I}_k) \right]^{-1} (\widetilde{T}_k + \lambda_k \widetilde{I}_k)^T \boldsymbol{e}_1 \\
&= \left[(T_k + \lambda_k I_k)^2 + \beta_k^2 \boldsymbol{e}_k \boldsymbol{e}_k^T \right]^{-1} (T_k + \lambda_k I_k) \boldsymbol{e}_1 \\
&= (T_k + \lambda_k I_k)^{-1} \boldsymbol{e}_1 - \frac{\beta_k^2 \left(\boldsymbol{e}_k^T (T_k + \lambda_k I_k)^{-1} \boldsymbol{e}_1 \right)}{1 + \beta_k^2 \boldsymbol{e}_k^T (T_k + \lambda_k I_k)^{-2} \boldsymbol{e}_k} (T_k + \lambda_k I_k)^{-2} \boldsymbol{e}_k,
\end{aligned} \tag{3.4}$$

where the last equality is derived from the Sherman-Morrison-Woodbury formula [21, Section 2.1.4]. It thus follows that

$$\widetilde{\boldsymbol{x}}_k - \boldsymbol{x}_k = Q_k(\widetilde{\boldsymbol{y}}_k - \boldsymbol{y}_k) = \frac{\beta_0 \beta_k^2 \left(\boldsymbol{e}_k^T (T_k + \lambda_k I_k)^{-1} \boldsymbol{e}_1 \right)}{1 + \beta_k^2 \boldsymbol{e}_k^T (T_k + \lambda_k I_k)^{-2} \boldsymbol{e}_k} \cdot Q_k (T_k + \lambda_k I_k)^{-2} \boldsymbol{e}_k. \tag{3.5}$$

By (2.5) and [24, Eqs (3.9) and (3.10)],

$$\left| \boldsymbol{e}_k^T (T_k + \lambda_k I_k)^{-1} \boldsymbol{e}_1 \right| = \frac{| \boldsymbol{e}_k^T \boldsymbol{y}_k |}{\beta_0} \le \frac{2\Delta}{\beta_0} \left(\frac{\sqrt{\kappa_k} - 1}{\sqrt{\kappa_k} + 1} \right)^{k-1}. \tag{3.6}$$

The desired bound (3.2a) then follows directly from (3.5) and (3.6).

Next we prove (3.2b). Recall that $\|\boldsymbol{x}_k\| = \|\boldsymbol{y}_k\| = \beta_0 \left\| (T_k + \lambda_k I_k)^{-1} \boldsymbol{e}_1 \right\|$ and $\|\widetilde{\boldsymbol{x}}_k\| = \|\widetilde{\boldsymbol{y}}_k\| = \beta_0 \left\| (\widetilde{T}_k + \lambda_k \widetilde{I}_k)^\dagger \boldsymbol{e}_1 \right\|$. It follows from (3.4) that

$$\begin{aligned}
\frac{\|\widetilde{\boldsymbol{x}}_k\|^2 - \|\boldsymbol{x}_k\|^2}{\beta_0^2} &= \frac{\beta_k^4 \left(\boldsymbol{e}_k^T (T_k + \lambda_k I_k)^{-1} \boldsymbol{e}_1 \right)^2}{\left(1 + \beta_k^2 \boldsymbol{e}_k^T (T_k + \lambda_k I_k)^{-2} \boldsymbol{e}_k \right)^2} \left\| (T_k + \lambda_k I_k)^{-2} \boldsymbol{e}_k \right\|^2 \\
&\quad - \frac{2\beta_k^2 \left(\boldsymbol{e}_k^T (T_k + \lambda_k I_k)^{-1} \boldsymbol{e}_1 \right)}{1 + \beta_k^2 \boldsymbol{e}_k^T (T_k + \lambda_k I_k)^{-2} \boldsymbol{e}_k} \cdot \boldsymbol{e}_1^T (T_k + \lambda_k I_k)^{-3} \boldsymbol{e}_k.
\end{aligned} \tag{3.7}$$

The desired bound (3.2b) then follows directly from (3.6) and Lemma 3.1. □

Remark 1. *Since $\lambda_k \to \lambda_{\text{opt}}$, we have $\kappa_k \to \kappa$ and*

$$\left(\frac{\sqrt{\kappa_k} - 1}{\sqrt{\kappa_k} + 1} \right)^{k-1} \to 0, \quad \textit{as } k \textit{ increases.}$$

Thus, the bound in (3.2a) *implies that $\boldsymbol{x}_k$ and $\widetilde{\boldsymbol{x}}_k$ approach each other as k increase. Furthermore, since $\boldsymbol{x}_k \to \boldsymbol{x}_{\text{opt}}$, it follows that $\widetilde{\boldsymbol{x}}_k \to \boldsymbol{x}_{\text{opt}}$ as k increase.*

If there exist some ℓ such that $\lambda_\ell > 0$, then $\lambda_k > 0$ [31, Theorem 2.6] and $\|\boldsymbol{x}_k\| = \Delta$ for all $k \ge \ell$. It then follows from (3.2b) *that $\|\widetilde{\boldsymbol{x}}_k\| \to \Delta$ and that the rate at which $\|\widetilde{\boldsymbol{x}}_k\|$ approaches Δ is faster than the rate at which $\widetilde{\boldsymbol{x}}_k$ approaches $\boldsymbol{x}_{\text{opt}}$.*

Combining (3.2a) and Theorem 2.2, we immediately obtain the following corollary.

Corollary 3.1. *For $k \geq 1$, we have*

$$\|\widetilde{\boldsymbol{x}}_k - \boldsymbol{x}_{\text{opt}}\| \leq 2\xi_k\Delta\left(\frac{\sqrt{\kappa_k}-1}{\sqrt{\kappa_k}+1}\right)^{k-1} + 4\Delta\sqrt{\kappa}\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^{k}.$$

The following theorem shows that, provided the Lanczos process doesn't break down, $\|(A+\lambda_k I)\widetilde{\boldsymbol{x}}_k+\boldsymbol{g}\|$ is always *strictly* less than $\|(A+\lambda_k I)\boldsymbol{x}_k+\boldsymbol{g}\|$.

Theorem 3.2. *If $\beta_k \neq 0$, then*

$$\|(A+\lambda_k I)\widetilde{\boldsymbol{x}}_k+\boldsymbol{g}\| < \|(A+\lambda_k I)\boldsymbol{x}_k+\boldsymbol{g}\|.$$

Proof. By (3.3), $\widetilde{T}_k + \lambda_k\widetilde{I}_k$ is of full column rank. Therefore, $\widetilde{\boldsymbol{y}}_k$ is the unique solution to the least squares problem (3.1). Since $\beta_k \neq 0$, it follows from (3.4) that $\widetilde{\boldsymbol{y}}_k \neq \boldsymbol{y}_k$. Hence,

$$\begin{aligned}
\|(A+\lambda_k I)\boldsymbol{x}_k+\boldsymbol{g}\| &= \|(A+\lambda_k I)Q_k\boldsymbol{y}_k+\boldsymbol{g}\| \\
&= \|Q_{k+1}\left[(\widetilde{T}_k+\lambda_k\widetilde{I}_k)\boldsymbol{y}_k+\beta_0\boldsymbol{e}_1\right]\| \\
&= \|(\widetilde{T}_k+\lambda_k\widetilde{I}_k)\boldsymbol{y}_k+\beta_0\boldsymbol{e}_1\| \\
&> \|(\widetilde{T}_k+\lambda_k\widetilde{I}_k)\widetilde{\boldsymbol{y}}_k+\beta_0\boldsymbol{e}_1\| \\
&= \|(A+\lambda_k I)\widetilde{\boldsymbol{x}}_k+\boldsymbol{g}\|.
\end{aligned}$$

This completes the proof. □

If the Lanczos process terminates at step k, i.e., $\beta_k = 0$, then $\boldsymbol{x}_k \overset{(3.4)}{=} \widetilde{\boldsymbol{x}}_k$ and $\boldsymbol{x}_k = \boldsymbol{x}_{\text{opt}}$ [27, Lemma 3.3]. Therefore, both the GLTR and MRTR methods can theoretically obtain the exact solution when the Lanczos process terminates.

The standard GLTR method follows the spirit of the Rayleigh-Ritz procedure [32]: it projects the original TRS onto the Krylov subspace via the Lanczos process, and obtains both the approximate Lagrange multiplier and the approximate solution simultaneously under a Galerkin-type projection condition.

The proposed MRTR method shares the same design logic as the refined Rayleigh-Ritz method [32,33]: it takes the approximate Lagrange multiplier yielded by GLTR as a fixed quantity, and then seeks a new approximate solution within the same Krylov subspace by minimizing the residual norm. This refinement step improves the accuracy of the approximate solution without additional matrix-vector multiplications.

Despite the similar refinement idea, the two classes of methods are essentially different in problem structure. The refined Rayleigh-Ritz method reduces a large-scale eigenvalue problem to a small-scale *singular value problem* to compute refined eigenpairs approximations. In contrast, the MRTR method transforms the large-scale TRS into a small-scale *least-squares problem* after fixing the approximate Lagrange multiplier. Due to this essential difference in the nature of the underlying problems, the mature theoretical analysis framework of the refined Rayleigh-Ritz method seems difficult to be directly borrowed or applied to the MRTR method.

For the current work, we focus on algorithm construction, implementation details, and numerical performance verification. We will carry out in-depth theoretical research along this direction in our future work.

4. Numerical experiments

In this section, we evaluate the numerical performance of the proposed method through some numerical examples. All the numerical experiments were run on an AMD R7 5800H CPU 3.20 GHz with 16 GB RAM under the Windows 11 operating system. The experimental results are obtained from using MATLAB R2022a implementation with machine precision $u_{\text{machine}} \approx 2.22 \times 10^{-16}$.

We compare the MRTR method (Algorithm 2) with the GLTR method. In all the experiments, we let $tol = 10^{-8}$, $It_{\max} = 1000$, and three distinct vectors $\boldsymbol{g}$:

$$\begin{aligned}
\boldsymbol{g}_1 &= \left(\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}, \ldots, \frac{1}{\sqrt{n}}\right)^T \in \mathbb{R}^n, \\
\boldsymbol{g}_2 &= \frac{\boldsymbol{c}_1}{\|\boldsymbol{c}_1\|} \text{ with } \boldsymbol{c}_1 = (\cos 1, \cos 2, \ldots, \cos n)^T \in \mathbb{R}^n, \text{ and} \\
\boldsymbol{g}_3 &= \frac{\boldsymbol{c}_2}{\|\boldsymbol{c}_2\|} \text{ with } \boldsymbol{c}_2 = \texttt{randn}(n, 1) \in \mathbb{R}^n
\end{aligned}$$

To ensure the orthogonality of the basis Q_k, we employ the full reorthogonalization process such that $\|Q_k^T Q_k - I_k\| = O(u_{\text{machine}})$. It is proven in [11] that the TRS can be equivalently transformed into an eigenvalue problem of a matrix. More precisely, taking TRS (2.4) as an example, λ_k is the rightmost eigenvalue of

$$\begin{pmatrix} -T_k & \frac{\beta_0^2 \boldsymbol{e}_1 \boldsymbol{e}_1^T}{\Delta^2} \\ I & -T_k \end{pmatrix} \in \mathbb{R}^{2k \times 2k}.$$

If $\lambda_k > 0$, then $\boldsymbol{y}_k = \frac{\Delta \cdot z_k(1:k)}{\|z_k(1:k)\|}$, where $z_k \in \mathbb{R}^{2k}$ is the eigenvector corresponding to λ_k. Therefore, we use the built-in MATLAB function `eig` to compute $(\lambda_k, \boldsymbol{y}_k)$ in the second phase. For the least squares problem (3.1), we use MATLAB's division command "\" to compute $\widetilde{\boldsymbol{y}}_k$.

Recall that we assume the CG iterations don't solve TRS (1.1), meaning that the second phase is activated, i.e., $\lambda_{\text{opt}} > 0$ and $\|\boldsymbol{x}_{\text{opt}}\| = \Delta$. Therefore, to ensure that the norm of output approximation $\widetilde{\boldsymbol{x}}_k$ is strictly equal to Δ numerically, we normalize $\widetilde{\boldsymbol{y}}_k$ when the residual norm in the MRTR algorithm approaches the termination criterion. More precisely, we replace lines 18–20 in Algorithm 2 with the following operations:

if $\frac{\|(\widetilde{T}_k + \lambda_k \widetilde{I}_k)\widetilde{\boldsymbol{y}}_k + \beta_0 \boldsymbol{e}_1\|}{\|\boldsymbol{g}\|} \le tol$
$\widetilde{\boldsymbol{y}}_k = \dfrac{\Delta \cdot \widetilde{\boldsymbol{y}}_k}{|\widetilde{\boldsymbol{y}}_k\|}$, $\widetilde{\boldsymbol{x}}_k = Q_k \widetilde{\boldsymbol{y}}_k$
  if $\frac{\|A\widetilde{\boldsymbol{x}}_k + \lambda_k \widetilde{\boldsymbol{x}}_k + \boldsymbol{g}\|}{\|\boldsymbol{g}\|} \le tol$
  Output $\widetilde{\boldsymbol{x}}_k$; **return**
  end if
end if

Define

$$f_{\text{rel}_{\text{err}}} := \left| \frac{f(\boldsymbol{x}_{\text{GLTR}}) - f(\boldsymbol{x}_{\text{MRTR}})}{\min\{f(\boldsymbol{x}_{\text{GLTR}}), f(\boldsymbol{x}_{\text{MRTR}})\}} \right|$$

where $f(\boldsymbol{x}_{\mathrm{GLTR}})$ and $f(\boldsymbol{x}_{\mathrm{MRTR}})$ denote the approximate objective function values obtained by the GLTR and MRTR algorithms, respectively. This metric measures the relative discrepancy between the optimal objective values output by the two iterative methods.

In this section, 'CPU(s)' denotes the running time of the algorithms, 'It' denotes the number of iterations, and $f(\widetilde{\boldsymbol{x}})$ denotes the approximate optimal values obtained by the two algorithms.

Example 1. *In this example, we consider two diagonal matrices.*

- Example 1(a)*:* $A = diag(t_{jn}^{[a,b]})$ *[26, 27], with*

$$t_{jn}^{[a,b]} = \Big(\frac{b-a}{2}\Big)\Big(t_{jn} + \Big(\frac{a+b}{b-a}\Big)\Big) \text{ with } t_{jn} = \cos\frac{(2j-1)\pi}{2n},\ j = 1, 2, \ldots, n, \tag{4.1}$$

  where $n = 10,000$, $-a = b = 100$, *and* $t_{jn}^{[a,b]}$ *is the n-th translated Chebyshev zero nodes on* $[a, b]$.
- Example 1(b)*:* $A = diag(\{i - n - 1\}_{i=1}^{n})$ *[25], with* $n = 10,000$.

In this example, we set $\Delta = 1$. *The numerical results of Example 1 are presented in Figure 1 and Table 1. Figure 1 shows the convergence curves of the residual norms* $\frac{\|(A+\lambda_k I)\widetilde{\boldsymbol{x}}+\boldsymbol{g}\|}{\|\boldsymbol{g}\|}$ *for the two algorithms.*

Table 1. Numerical experiments on Example 1.

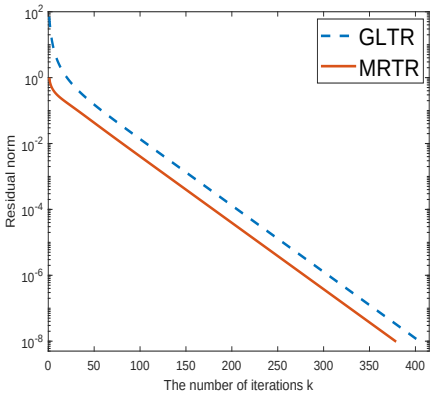
	Matrix	Method	CPU (s)	It	$f(\widetilde{\boldsymbol{x}})$	$f_{\mathrm{rel_{err}}}$
$\boldsymbol{g} = \boldsymbol{g}_1$	$A = diag(t_{jn}^{[a,b]})$	GLTR	69.0	405	-50.161553602236729	1.5×10^{-15}
		MRTR	56.8	379	-50.161553602236651	
	$A = diag(\{i - n - 1\}_{i=1}^{n})$	GLTR	282.2	645	-5.000010488558799e+03	9.1×10^{-16}
		MRTR	258.5	629	-5.000010488558803e+03	
$\boldsymbol{g} = \boldsymbol{g}_2$	$A = diag(t_{jn}^{[a,b]})$	GLTR	72.3	404	-50.161827411264966	7.1×10^{-16}
		MRTR	51.7	363	-50.161827411264966	
	$A = diag(\{i - n - 1\}_{i=1}^{n})$	GLTR	286.7	651	-5.000008114793698e+03	2.5×10^{-15}
		MRTR	280.1	650	-5.000008114793711e+03	
$\boldsymbol{g} = \boldsymbol{g}_3$	$A = diag(t_{jn}^{[a,b]})$	GLTR	62.7	383	-50.165962388210183	1.5×10^{-15}
		MRTR	50.2	358	-50.165962388210261	
	$A = diag(\{i - n - 1\}_{i=1}^{n})$	GLTR	385.5	666	-5.000005957484766e+03	1.3×10^{-15}
		MRTR	286.0	655	-5.000005957484773e+03	

Example 2. *In this example, we consider three symmetric matrices* `nd3k`, `nemeth01`, *and* `stokes64s` *from the University of Florida Sparse Matrix Collection*[*]. *The numerical results are presented in Figure 2 and Table 2. Figure 2 shows the convergence curves of the residual norms* $\frac{\|(A+\lambda_k I)\widetilde{\boldsymbol{x}}+\boldsymbol{g}\|}{\|\boldsymbol{g}\|}$ *for the two algorithms. In this example, we set* $\Delta = 10$.

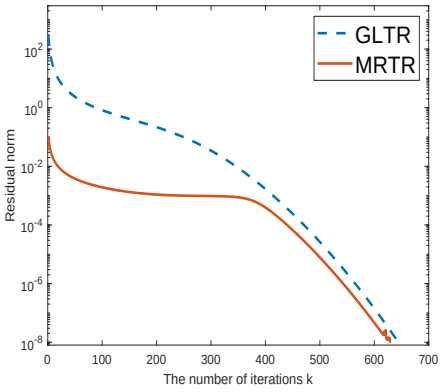
As illustrated in Figures 1 and 2, MRTR achieves a lower residual norm at every iteration step compared to GLTR. In particular, it can be seen from Figure 2 that among the three numerical examples in Example 2, the residual convergence curve of the GLTR method exhibits obvious oscillations, whereas that of the MRTR method is smooth, which reflects that the MRTR method has better stability.

It can be seen from Tables 1 and 2 that the MRTR outperforms the GLTR in terms of both running time and the number of iterations. In addition, the relative errors $f_{\mathrm{rel_{err}}}$ of the approximate optimal

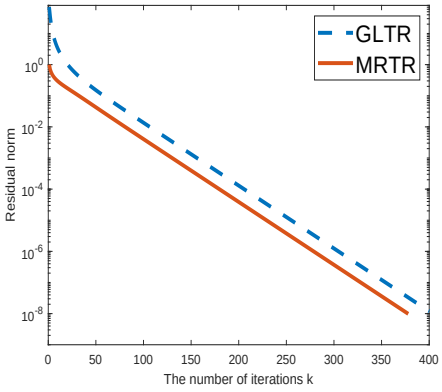
[*]https://sparse.tamu.edu.



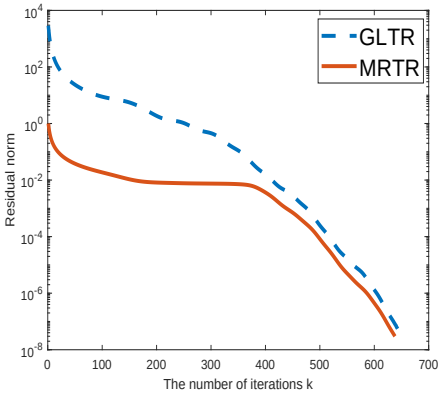
(a) Example 1(a): $A = diag(t_{jn}^{[a,b]})$, $\boldsymbol{g} = \boldsymbol{g}_1$



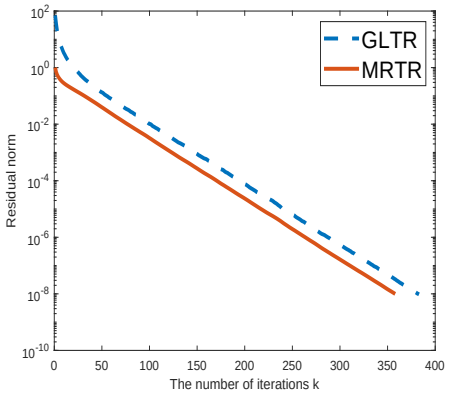
(b) Example 1(b): $A = diag(\{i - n - 1\}_{i=1}^{n})$, $\boldsymbol{g} = \boldsymbol{g}_1$



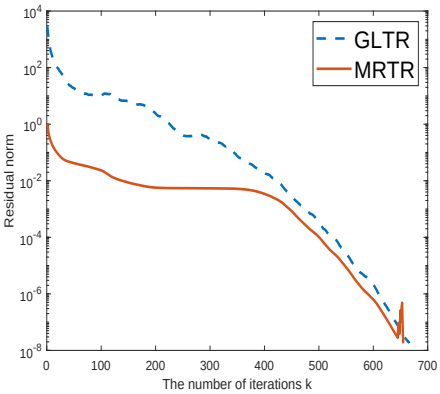
(c) Example 1(a): $A = diag(t_{jn}^{[a,b]})$, $\boldsymbol{g} = \boldsymbol{g}_2$



(d) Example 1(b): $A = diag(\{i - n - 1\}_{i=1}^{n})$, $\boldsymbol{g} = \boldsymbol{g}_2$

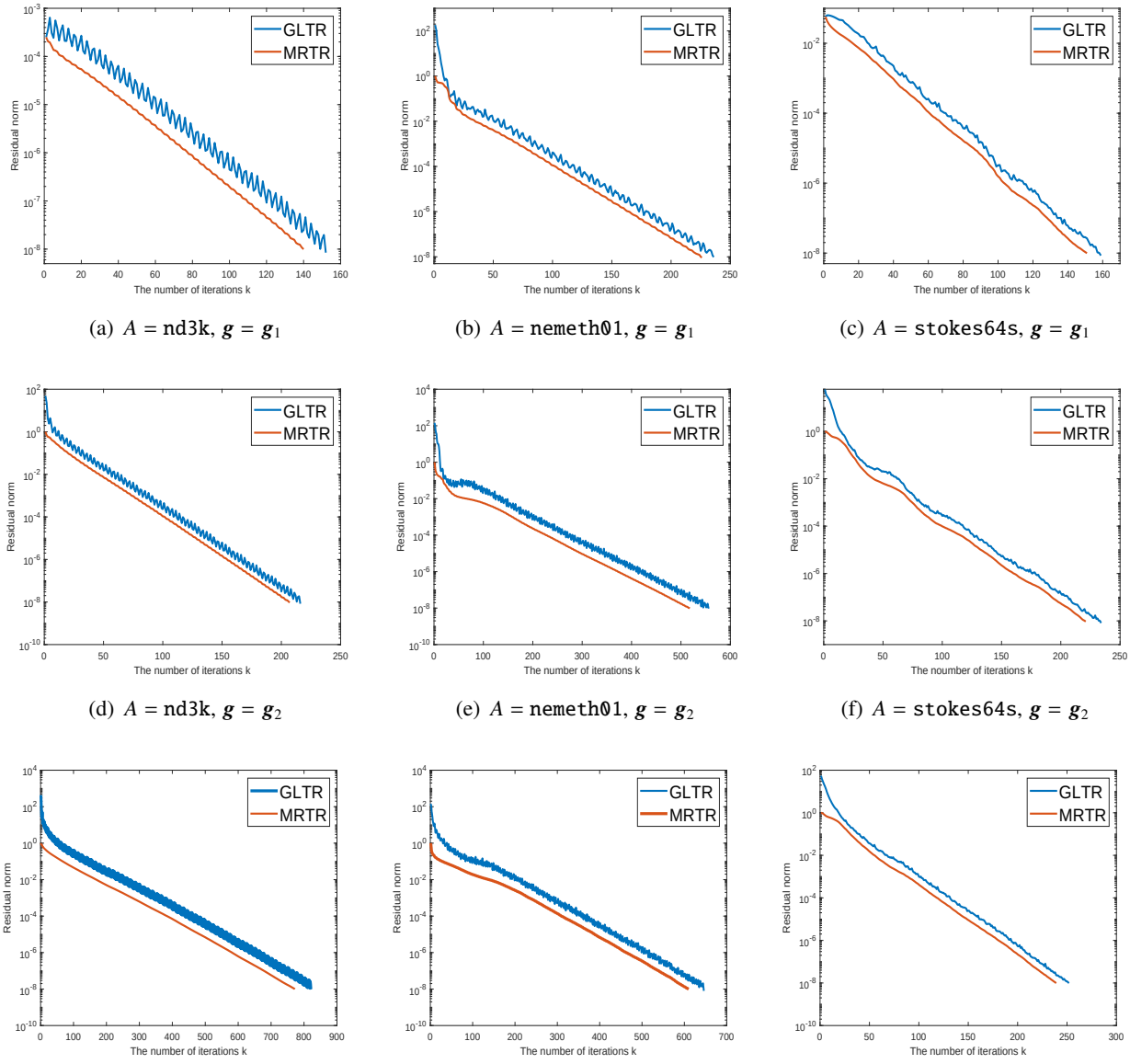


(e) Example 1(a): $A = diag(t_{jn}^{[a,b]})$, $\boldsymbol{g} = \boldsymbol{g}_3$



(f) Example 1(b): $A = diag(\{i - n - 1\}_{i=1}^{n})$, $\boldsymbol{g} = \boldsymbol{g}_3$

Figure 1. Numerical experiments on Example 1.



(a) A = nd3k, $\boldsymbol{g} = \boldsymbol{g}_1$ (b) A = nemeth01, $\boldsymbol{g} = \boldsymbol{g}_1$ (c) A = stokes64s, $\boldsymbol{g} = \boldsymbol{g}_1$

(d) A = nd3k, $\boldsymbol{g} = \boldsymbol{g}_2$ (e) A = nemeth01, $\boldsymbol{g} = \boldsymbol{g}_2$ (f) A = stokes64s, $\boldsymbol{g} = \boldsymbol{g}_2$

(g) A = nd3k, $\boldsymbol{g} = \boldsymbol{g}_3$ (h) A = nemeth01, $\boldsymbol{g} = \boldsymbol{g}_3$ (i) A = stokes64s, $\boldsymbol{g} = \boldsymbol{g}_3$

Figure 2. Numerical experiments on Example 2.

Table 2. Numerical experiments on Example 2.

	Matrix	Method	CPU (s)	It	$f(\widetilde{\boldsymbol{x}})$	$f_{\text{rel}_{\text{err}}}$
$\boldsymbol{g} = \boldsymbol{g}_1$	nd3k	GLTR	3.5	152	-9.999539149018121	1.8×10^{-16}
		MRTR	2.4	140	-9.999539149018119	
	nemeth01	GLTR	12.6	236	$-2.158396347063678 \times 10^3$	4.2×10^{-16}
		MRTR	10.2	226	$-2.158396347063679 \times 10^3$	
	stokes64s	GLTR	3.8	159	-10.001772476459125	5.3×10^{-16}
		MRTR	2.5	151	-10.001772476459120	
$\boldsymbol{g} = \boldsymbol{g}_2$	nd3k	GLTR	10.6	216	-0.279360075402613	0
		MRTR	8.4	207	-0.279360075402613	
	nemeth01	GLTR	184.9	556	$-2.154654144026191 \times 10^3$	4.2×10^{-16}
		MRTR	148.7	518	$-2.154654144026190 \times 10^3$	
	stokes64s	GLTR	12.1	234	-5.858337377260747	0
		MRTR	9.0	221	-5.858337377260747	
$\boldsymbol{g} = \boldsymbol{g}_3$	nd3k	GLTR	640.6	821	-1.793664237863516	5.5×10^{-15}
		MRTR	528.6	772	-1.793664237863506	
	nemeth01	GLTR	299.0	646	$-2.153939233613416 \times 10^3$	2.5×10^{-15}
		MRTR	245.3	610	$-2.153939233613411 \times 10^3$	
	stokes64s	GLTR	16.0	252	-6.000328380904500	1.3×10^{-15}
		MRTR	12.3	239	-6.000328380904508	

values obtained by the two methods reach the order of $O(u_{\text{machine}})$. This implies that the quality of the approximate solutions obtained by the MRTR is comparable to that of the GLTR.

Numerical experiments show that the MRTR method outperforms the GLTR method. This makes MRTR a robust and efficient choice for solving the large-scale TRS problems commonly encountered in scientific computing.

5. Conclusions

In this paper, we propose a minimal residual method for solving the large-scale TRS (MRTR) in the easy case, which is inspired by the GMRES/MINRES and the GLTR method. We first establish the theoretical relationship between $\boldsymbol{x}_k$ and $\widetilde{\boldsymbol{x}}_k$, proving that the two solutions converge to each other as the iteration proceeds and that the MRTR method yields a strictly smaller residual norm than GLTR when the Lanczos process does not break down, which verifies the theoretical superiority of MRTR in residual reduction.

Numerical experiments on both constructed diagonal matrices and real sparse symmetric matrices from the University of Florida Sparse Matrix Collection demonstrate that the MRTR method outperforms the GLTR method in terms of iteration numbers and CPU time, while achieving approximate optimal values with the same high precision as GLTR. Future research can focus on extending the MRTR method to TRS in the *hard case* and exploring its applications in more complex optimization models such as cubic regularization and constrained nonlinear optimization.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgements

This work was supported by the Basic Research Program of Jiangsu Province (Youth Program of Natural Science Foundation of Jiangsu Province, Grant No. BK20250892).

Conflict of interest

The authors declare there is no conflicts of interest.

References

1. A. R. Conn, N. I. M. Gould, P. L. Toint, *Trust-Region Methods*, SIAM, Philadelphia, PA, 2000. https://doi.org/10.1137/1.9780898719857
2. J. Nocedal, S. Wright, *Numerical Optimization*, 2^{nd} edition, Springer, New York, 2006.
3. L. Zhang, C. Shen, W. Yang, J. Júdice, A Lanczos method for large-scale extreme Lorentz eigenvalue problems, *SIAM J. Matrix Anal. Appl.*, **39** (2018), 611–631. https://doi.org/10.1137/17M1111401
4. W. W. Hager, Y. Krylyuk, Graph partitioning and continuous quadratic programming, *SIAM J. Discrete Math.*, **12** (1999), 500–523. https://doi.org/10.1137/S0895480199335829
5. M. Rojas, S. A. Santos, D. C. Sorensen, A new matrix-free algorithm for the large-scale trust-region subproblem, *SIAM J. Optim.*, **11** (2001), 611–646. https://doi.org/10.1137/S105262349928887X
6. M. Rojas, S. Santos, D. Sorensen, Algorithm 873: LSTRS: MATLAB software for large-scale trust-region subproblems and regularization, *ACM Trans. Math. Software*, **34** (2008), 1–28. https://doi.org/10.1145/1326548.1326553
7. N. M. Gould, D. P. Robinson, H. S. Thorne, On solving trust-region and other regularised subproblems in optimization, *Math. Program. Comput.*, **2** (2010), 21–57. https://doi.org/10.1007/s12532-010-0011-7
8. T. Steihaug, The conjugate gradient method and trust regions in large scale optimization, *SIAM J. Numer. Anal.*, **20** (1983), 626–637, https://doi.org/10.1137/0720042
9. P. Toint, Towards an efficient sparsity exploiting Newton method for minimization, in Sparse Matrices and Their Uses, Academic Press, London, (1981), 57–88.
10. W. Gander, G.H. Golub, U. von Matt, A constrained eigenvalue problem, *Linear Algebra Appl.*, **114–115** (1989), 815–839. https://doi.org/10.1016/0024-3795(89)90494-1
11. S. Adachi, S. Iwata, Y. Nakatsukasa, A. Takeda, Solving the trust-region subproblem by a generalized eigenvalue problem, *SIAM J. Optim.*, **27** (2017), 269–291. https://doi.org/10.1137/16M1058200
12. N. Gould, S. Lucidi, M. Roma, P. Toint, Solving the trust-region subproblem using the Lanczos method, *SIAM J. Optim.*, **9** (1999), 504–525. https://doi.org/10.1137/S1052623497322735

13. L. Zhang, C. Shen, A nested Lanczos method for the trust-region subproblem, *SIAM J. Sci. Comput.*, **40** (2018), A2005–A2032. https://doi.org/10.1137/17M1145914

14. F. Rendl, H. Wolkowicz, A semidefinite framework for trust region subproblems with applications to large scale minimization, *Math. Program.*, **77** (1997), 273–299. https://doi.org/10.1007/BF02614438

15. J. B. Erway, P. E. Gill, J. D. Griffin, Iterative methods for finding a trust-region step, *SIAM J. Optim.*, **20** (2009), 1110–1131. https://doi.org/10.1137/070708494

16. G. H. Golub, U. von Matt, Quadratically constrained least squares and quadratic problems, *Numer. Math.*, **59** (1991), 561–580. https://doi.org/10.1007/BF01385796

17. J. J. Moré, D. C. Sorensen, Computing a trust region step, *SIAM J. Sci. Stat. Comput.*, **4** (1983), 553–572. https://doi.org/10.1137/0904038

18. D. C. Sorensen, Newton's method with a model trust region modification, *SIAM J. Numer. Anal.*, **19** (1982), 409–426. https://doi.org/10.1137/0719026

19. D. Sorensen, Minimization of a large-scale quadratic function subject to a spherical constraint, *SIAM J. Optim.*, **7** (1997), 141–161. https://doi.org/10.1137/S1052623494274374

20. P. Tao, L. An, A DC optimization algorithm for solving the trust-region subproblem, *SIAM J. Optim.*, **8** (1998), 476–505. https://doi.org/10.1137/S1052623494274313

21. G. H. Golub, C. F. Van Loan, *Matrix Computations*, 4^{th} edition, Johns Hopkins University Press, Baltimore, MD, 2013.

22. Y. Carmon, J. C. Duchi, First-order method for nonconvex quadratic minimization, *SIAM Rev.*, **62** (2020), 395–436. https://doi.org/10.1137/20M1321759

23. Y. Carmon, J. C. Duchi, Analysis of Krylov subspace solutions of regularized nonconvex quadratic problems, in *NIPS'18: Proceedings of the 32nd International Conference on Neural Information Processing Systems*, (2018), 10728–10738.

24. B. Feng, G. Wu, On convergence of the generalized Lanczos trust-region method for trust-region subproblems, *Adv. Comput. Math.*, **51** (2025), 4. https://doi.org/10.1007/s10444-024-10217-5

25. N. I. M. Gould, V. Simoncini, Error estimates for iterative algorithms for minimizing regularized quadratic subproblems, *Optim. Methods Software*, **35** (2020), 304–328. https://doi.org/10.1080/10556788.2019.1670177

26. Z. Jia, F. Wang, The convergence of the generalized Lanczos trust-region method for the trust-region subproblem, *SIAM J. Optim.*, **31** (2021), 887–914. https://doi.org/10.1137/19M1279691

27. L. Zhang, C. Shen, R. Li, On the generalized Lanczos trust-region method, *SIAM J. Optim.*, **27** (2017), 2110–2142. https://doi.org/10.1137/16M1095056

28. Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2^{nd} edition, SIAM, Philadelphia, PA, 2003. https://doi.org/10.1137/1.9780898718003

29. W. W. Hager, Minimizing a quadratic over a sphere, *SIAM J. Optim.*, **12** (2001), 188–208. https://doi.org/10.1137/S1052623499356071

30. N. J. Higham, *Functions of Matrices: Theory and Computation*, SIAM, Philadelphia, 2008. https://doi.org/10.1137/1.9780898717778

31. L. Lukšan, C. Matonoha, J. Vlček, On Lagrange multipliers of trust-region subproblems, *BIT Numer. Math.*, **48** (2008), 763–768. https://doi.org/10.1007/s10543-008-0197-5

32. G. W. Stewart, *Matrix Agorithms II: Eigensystems*, SIAM, Philadelphia, 2001. https://doi.org/10.1137/1.9780898718058

33. Z. Jia, Refined iterative algorithms based on Arnoldi's process for large unsymmetric eigenproblems, *Linear Algebra Appl.*, **259** (1997), 1–23. https://doi.org/10.1016/S0024-3795(96)00238-8

AIMS Press