A Preconditioned Riemannian Gauss-Newton Method for Least Squares Inverse Eigenvalue Problems

Teng-Teng Yao∗, Zheng-Jian Bai†, Xiao-Qing Jin‡, Zhi Zhao§

June 19, 2018

Abstract

This paper is concerned with the least squares inverse eigenvalue problem of reconstructing a linear parameterized real symmetric matrix from the prescribed partial eigenvalues in the sense of least squares, which was originally proposed by Chen and Chu [SIAM J. Numer. Anal., 33 (1996), pp. 2417–2430]. We provide a Riemannian inexact Gauss-Newton method for solving the least squares inverse eigenvalue problem. The global and local convergence analysis of the proposed method is discussed. Also, a preconditioned conjugate gradient method with an efficient preconditioner is proposed for solving the Riemannian Gauss-Newton equation. Finally, some numerical tests, including an application in the inverse Sturm-Liouville problem, are reported to illustrate the efficiency of the proposed method.

Keywords. least squares inverse eigenvalue problem, Riemannian Gauss-Newton method, preconditioner

AMS subject classifications. 65F18, 65F15, 15A18, 58C15

1 Introduction

An inverse eigenvalue problem (IEP) aims to reconstruct a structured matrix from the prescribed spectral data. Inverse eigenvalue problems (IEPs) arise in various applications such as structural dynamics, vibration, inverse Sturm-Liouville problem, control design, geophysics, nuclear

∗Department of Mathematics, School of Sciences, Zhejiang University of Science and Technology, Hangzhou 310023, People’s Republic of China (yaotengteng718@163.com). The research of this author is supported by the National Natural Science Foundation of China (No. 11701514).

†Corresponding author. School of Mathematical Sciences and Fujian Provincial Key Laboratory on Mathematical Modeling & High Performance Scientific Computing, Xiamen University, Xiamen 361005, People’s Republic of China (zjbai@xmu.edu.cn). The research of this author is partially supported by the National Natural Science Foundation of Fujian Province of China (No. 2016J01035), and the Fundamental Research Funds for the Central Universities (No. 20720180008).

‡Department of Mathematics, University of Macau, Macao, People’s Republic of China (xqjin@umac.mo). The research of this author is supported by the research grant MYRG2016-00077-FST from University of Macau.

§Department of Mathematics, School of Sciences, Hangzhou Dianzi University, Hangzhou 310018, People’s Republic of China (zhaozhi231@163.com). The research of this author is supported by the National Natural Science Foundation of China (No. 11601112).
spectroscopy and molecular spectroscopy, etc. For the existence theory, numerical methods and applications of general IEPs, one may refer to [7, 9, 10, 13, 14, 15, 16, 26] and references therein.

In this paper, we consider the following least squares inverse eigenvalue problem, which was originally given by Chen and Chu [6].

**LSIEP I.** Given $l + 1$ real symmetric matrices $A_0, A_1, \ldots, A_l \in \mathbb{R}^{n \times n}$ and $m$ real numbers $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_m$ ($m \leq n$), find a vector $c = (c_1, \ldots, c_l)^T \in \mathbb{R}^l$ and a permutation $\sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_m\}$ with $1 \leq \sigma_1 < \sigma_2 < \cdots < \sigma_m \leq n$ to minimize the function

$$f(c, \sigma) := \frac{1}{2} \sum_{i=1}^{m} (\lambda_{\sigma_i}(c) - \lambda_i^*)^2,$$

where the real numbers $\lambda_1(c) \leq \lambda_2(c) \leq \cdots \leq \lambda_n(c)$ are the eigenvalues of the matrix $A(c)$ defined by

$$A(c) := A_0 + \sum_{i=1}^{l} c_i A_i.$$

This is a nonlinear least-squares problem, where the cost function $f(c, \sigma)$ is a function of a continuous variable $c$ and a discrete variable $\sigma$. This is a special kind of mixed optimization problem, where the function $f(c, \sigma)$ is nondifferentiable when the perturbation $\sigma$ is changed. For the LSIEP I, there exists an equivalent least-squares problem defined on a product manifold. Let $\mathcal{D}(m)$ and $\mathcal{O}(n)$ denote the set of all real diagonal matrices of order $m$ and the set of all real $n \times n$ orthogonal matrices, respectively. Define $\Lambda_m^* := \text{diag}(\lambda_1^*, \lambda_2^*, \ldots, \lambda_m^*)$, where $\text{diag}(a)$ denotes a diagonal matrix with $a$ on its diagonal. Given a matrix $\Lambda \in \mathcal{D}(n-m)$, $\text{blkdiag}(\Lambda_m^*, \Lambda)$ denotes the block diagonal matrix obtained from $\Lambda_m^*$ and $\Lambda$. Based on Theorem 3.1 in [6], the LSIEP I is equivalent to the following problem.

**LSIEP II.** Given $l + 1$ real symmetric matrices $A_0, A_1, \ldots, A_l \in \mathbb{R}^{n \times n}$ and $m$ real numbers $\lambda_1 \leq \lambda_2^* \leq \cdots \leq \lambda_m^*$ ($m \leq n$), find a vector $c \in \mathbb{R}^l$, an orthogonal matrix $Q \in \mathcal{O}(n)$, and a diagonal matrix $\Lambda \in \mathcal{D}(n-m)$ to minimize the function

$$h(c, Q, \Lambda) := \frac{1}{2} \|A(c) - Q \text{blkdiag}(\Lambda_m^*, \Lambda) Q^T\|_F^2,$$

where $\| \cdot \|_F$ denotes the Frobenius matrix norm.

The LSIEP II is a nonlinear least-squares problem defined on the product manifold $\mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)$. To solve the LSIEP II, Chen and Chu [6] proposed a lift and projection (LP) method. This method is a modification of the alternating projection method to an affine space and a Riemannian manifold. To solve the LSIEP I, Chen and Chu [6] proposed a hybrid method called the LP-Newton method. The idea is that an initial guess of $(c, \sigma)$ is obtained by using the LP method to the LSIEP II, then the Newton method is applied to the LSIEP I by fixing the value of $\sigma$. This method has fast local convergence while it requires an exact guess value of $\sigma$ via the LP method. We note that the LP-Newton method works only for small problems since the forming of the Hessian matrix of $f(c, \sigma)$ is very expensive. If $m = l$, then there is no permutation $\sigma$ in the function $f(c, \sigma)$. In this case, the LSIEP I becomes a continuous optimization problem. For this special case, Wang and Vong [25] proposed a Gauss-Newton-like method.
Optimization methods on smooth manifolds have been widely studied and applied to various kinds of areas such as numerical linear algebra and dynamical systems (see for instance [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25] and references therein). Recently, some Riemannian optimization methods were proposed for solving nonlinear eigenvalue problems and inverse eigenvalue problems [26, 27, 28, 29, 30]. In this paper, we propose a Riemannian inexact Gauss-Newton method for solving the LSIEP II. In [2], Absil et al. proposed a Riemannian Gauss-Newton method for solving nonlinear least squares problems defined between Riemannian manifold and Euclidean space, where the convergence analysis was not discussed. In [18], Gratton et al. gave some approximate Gauss-Newton methods for solving nonlinear least squares problems defined on Euclidean space. Sparked by [2, 18], we present an efficient Riemannian inexact Gauss-Newton method for solving the LSIEP II. The global convergence and local convergence rate are also discussed. An effective preconditioner is proposed for solving the Riemannian Gauss-Newton equation via the conjugate gradient (CG) method [17]. Finally, some numerical experiments, including an application in the inverse Sturm-Liouville problem, are reported to show the efficiency of the proposed method for solving the LSIEP II.

Throughout this paper, we use the following notation. The symbol $A^T$ denotes the transpose of a matrix $A$. The symbol $\text{Diag}(M) := \text{diag}(m_{11}, m_{22}, \ldots, m_{nn})$ denotes a diagonal matrix containing the diagonal elements of an $n \times n$ matrix $M = [m_{ij}]$. Let $0_{m \times n}$ be the $m \times n$ zero matrix and $e_k$ be the $k$-th column of the identity matrix $I_n$ of order $n$. Let $\mathbb{R}^{n \times n}$ and $\mathbb{S}\mathbb{R}^{n \times n}$ be the set of all $n$-by-$n$ real matrices and the set of all $n$-by-$n$ real symmetric matrices, respectively. Denote by $\text{tr}(A)$ the trace of a square matrix $A$. For two matrices $A, B \in \mathbb{R}^{n \times n}$, $[A, B] := AB - BA$ mean the Lie Bracket of $A$ and $B$. Let $\text{vec}(A)$ be the vectorization of a matrix $A$, i.e., a column vector obtained by stacking the columns of $A$ on top of one another. For two finite-dimensional vector spaces $\mathcal{X}$ and $\mathcal{Y}$ equipped with a scalar inner product $\langle \cdot, \cdot \rangle$ and its induced norm $\| \cdot \|$, let $A: \mathcal{X} \to \mathcal{Y}$ be a linear operator and the adjoint operator of $A$ be denoted by $A^*$. The operator norm of $A$ is defined by $\|A\| := \sup\{\|Ax\| : x \in \mathcal{X} \text{ with } \|x\| = 1\}$.

The remainder of this paper is organized as follows. In section 2 we propose a Riemannian inexact Gauss-Newton method for solving the LSIEP II. In section 3 we establish the global convergence and local convergence rate of the proposed approach under some conditions. A preconditioner is also proposed for solving the Riemannian Gauss-Newton equation. Finally, we report some numerical tests in section 4 and give some concluding remarks in section 5.

## 2 Riemannian inexact Gauss-Newton method

In this section, we present a Riemannian inexact Gauss-Newton method for solving the LSIEP II. Define an affine subspace and an isospectral manifold by

$$
\mathcal{A} := \{ A_0 + \sum_{i=1}^l c_i A_i \mid c_i \in \mathbb{R}, i = 1, 2, \ldots, l \},
$$

$$
\mathcal{M}(\Lambda_m^*) := \{ X = Q \text{blkdiag}(\Lambda_m^*, \Lambda) Q^T \in \mathbb{S}\mathbb{R}^{n \times n} \mid Q \in \mathcal{O}(n), \Lambda \in \mathcal{D}(n - m) \}.
$$

We see that $\mathcal{M}(\Lambda_m^*)$ is the set of all real $n \times n$ symmetric matrices whose spectrum contains the $m$ real numbers $\lambda_1^*, \lambda_2^*, \ldots, \lambda_m^*$. Thus, the LSIEP II has a solution such that $h(c, Q, \Lambda) = 0$ if and only if $\mathcal{A} \cap \mathcal{M}(\Lambda_m^*) \neq \emptyset$. 

3
Let $H$ be a nonlinear mapping between $\mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)$ and $\mathbb{S}^n$ defined by

$$H(c, Q, \Lambda) = A(c) - Q \text{blkdiag}(\Lambda^*_m, \Lambda) Q^T,$$

for all $(c, Q, \Lambda) \in \mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)$. Then, the LSIEP II can be written as the following minimization problem:

$$\min \quad h(c, Q, \Lambda) := \frac{1}{2} \|H(c, Q, \Lambda)\|^2_F,$$

subject to (s.t.) $(c, Q, \Lambda) \in \mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)$.

Sparked by the ideas in [2, 18], we propose a Riemannian inexact Gauss-Newton method for solving Problem [2]. We note that the dimension of the product manifold $\mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)$ is given by

$$\dim(\mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)) = l + \frac{n(n-1)}{2} + n - m.$$

If $l < m$, then

$$\dim(\mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)) < \dim(\mathbb{S}^n).$$

Therefore, the nonlinear equation $H(c, Q, \Lambda) = 0_{n \times n}$ is an over-determined matrix equation defined on the product manifold $\mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)$.

Notice that $\mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)$ is an embedded submanifold of $\mathbb{R}^l \times \mathbb{S}^n \times \mathcal{D}(n-m)$. One may equip $\mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)$ with the induced Riemannian metric:

$$g_{(c, Q, \Lambda)}((\xi_1, \eta_1, \tau_1), (\xi_2, \eta_2, \tau_2)) := \text{tr}(\xi_1^T \xi_2) + \text{tr}(\eta_1^T \eta_2) + \text{tr}(\tau_1^T \tau_2),$$

for any $(c, Q, \Lambda) \in \mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)$, and $(\xi_1, \eta_1, \tau_1), (\xi_2, \eta_2, \tau_2) \in T_{(c, Q, \Lambda)} \mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)$, and its induced norm $\| \cdot \|$. The tangent space $T_{(c, Q, \Lambda)} \mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)$ of $\mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)$ at $(c, Q, \Lambda) \in \mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)$, which is given by [2, p.42]

$$T_{(c, Q, \Lambda)} \mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m) = \{(\mathbf{r}, Q\Omega, U) \mid \Omega^T = -\Omega, \mathbf{r} \in \mathbb{R}^l, \Omega \in \mathbb{S}^n, U \in \mathcal{D}(n-m)\}.$$

Hence, $(\mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m), g)$ is a Riemannian product manifold.

A Riemannian Gauss-Newton method for solving Problem [2] can be stated as follows. Given the current iterate $X_k := (c_k, Q_k, \Lambda_k) \in \mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)$, solve the normal equation

$$(DH(X_k))^* \circ DH(X_k)[\Delta X_k] = -(DH(X_k))^*[H(X_k)],$$

for $\Delta X_k := (\Delta c_k, \Delta Q_k, \Delta \Lambda_k) \in T_{X_k} \mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)$. Here,

$$DH(X_k) : T_{(c, Q, \Lambda)} \mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m) \rightarrow T_{H(X_k)} \mathbb{S}^n \times \mathcal{D}(n-m)$$

is the Riemannian differential of $H$ at the point $X_k$, which is given by

$$DH(X_k)[\Delta X_k] = (A(\Delta c_k) - A_0) + [Q_k \bar{Q}_k Q_k^T, \Delta Q_k Q_k^T] - (Q_k P) \Delta \Lambda_k (Q_k P)^T,$$
With respect to the Riemannian metric $\xi$, for all $H$, \(\text{Hessian } \text{Hess}_h\) Step 1.

(A Riemannian inexact Gauss-Newton method) Algorithm 2.1

Problem (2) can be described as follows.

where

\[
\overline{A}_k := \text{blkdiag}(A^*_m, \Lambda_k) \quad \text{and} \quad P := [0_{(n-m)\times m}, I_{n-m}]^T.
\] (5)

Let \(\nabla\) denote the Riemannian connection of \(\mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)\). By using (8.31) in [2] p.185] we obtain

\[
\nabla^2 h(X)[\xi_X, \eta_X] = \langle DH(X)[\xi_X], DH(X)[\eta_X] \rangle + \langle H(X), \nabla^2 H(X)[\xi_X, \eta_X] \rangle,
\] (9)

for all \(X := (c, Q, \Lambda) \in \mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)\) and \(\xi_X, \eta_X \in T_X \mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)\), where \(\nabla^2 h\) is a \((0,2)\)-tensor field and \(\nabla^2 H(X)[\cdot, \cdot] = [\nabla^2 H_{ij}(X)[\cdot, \cdot]] \in \mathbb{S}^{n \times n}\) [2] p.109]. The Riemannian Hessian \(\text{Hess } h(X)\) at a point \(X \in \mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)\) is determined by

\[
\nabla^2 h(X)[\xi_X, \eta_X] = \langle \text{Hess } h(X)[\xi_X], \eta_X \rangle,
\] (10)

for all \(\xi_X, \eta_X \in T_X \mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)\). In particular, if \(X_*\) is a solution of the equation \(H(X) = 0_{n \times n}\), then we can obtain

\[
\text{Hess } h(X_*) = (DH(X_*)^* \circ DH(X_*))
\] (11)

Based on the above discussion, a Riemannian inexact Gauss-Newton method for solving Problem [2] can be described as follows.

**Algorithm 2.1** (A Riemannian inexact Gauss-Newton method)

**Step 0.** Choose an initial point \(X_0 \in \mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)\), \(\beta, \eta_{\text{max}} \in (0, 1), \sigma \in (0, \frac{1}{2})\). Let \(k := 0\).

**Step 1.** Apply the CG method to finding an approximate solution \(\Delta X_k \in T_{X_k} \mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m)\) of

\[
(DH(X_k))^* \circ DH(X_k)[\Delta X_k] = -\nabla h(X_k)
\] (12)

such that

\[
\| (DH(X_k))^* \circ DH(X_k)[\Delta X_k] + (DH(X_k))^*[H(X_k)] \| \leq \eta_k \|\nabla h(X_k)\| \] (13)
and
\[ \langle \text{grad} h(X_k), \Delta X_k \rangle \leq -\eta_k \langle \Delta X_k, \Delta X_k \rangle, \] (14)

where \( \eta_k := \min\{\eta_{\text{max}}, \|\text{grad} h(X_k)\|\} \). If (13) and (14) are not attainable, then let
\[ \Delta X_k := -\text{grad} h(X_k). \]

**Step 2.** Let \( l_k \) be the smallest nonnegative integer \( l \) such that
\[ h(R_{X_k}(\beta^l \Delta X_k)) - h(X_k) \leq \sigma \beta^l \langle \text{grad} h(X_k), \Delta X_k \rangle. \] (15)

Set
\[ X_{k+1} := R_{X_k}(\beta^l \Delta X_k). \]

**Step 3.** Replace \( k \) by \( k + 1 \) and go to **Step 1.**

We point out that, in **Step 2** of Algorithm 2.1, \( R \) is a retraction on \( \mathbb{R}^l \times O(n) \times D(n-m) \), which takes the form of
\[ R_{X_k}(\Delta X_k) = (c_k + \Delta c_k, R^o_{Q_k}(\Delta Q_k), \Lambda_k + \Delta \Lambda_k), \] (16)

where \( R^o \) is a retraction on \( O(n) \), which may be chosen as [2, p.58]:
\[ R^o_Q(\eta Q) = qf(Q + \eta Q), \quad \eta Q \in T_Q O(n). \]

Here, \( qf(A) \) denotes the \( Q \) factor of the QR decomposition of an invertible matrix \( A \in \mathbb{R}^{n \times n} \) as \( A = QR \), where \( Q \) belongs to \( O(n) \) and \( R \) is an upper triangular matrix with strictly positive diagonal elements. For the retraction \( R \) defined by (16), there exist two scalars \( \nu > 0 \) and \( \mu_\nu > 0 \) such that [2, p.149]
\[ \nu \|\Delta X\| \geq \text{dist}(X, R_X(\Delta X)), \] (17)

for all \( X \in \mathbb{R}^l \times O(n) \times D(n-m) \) and
\[ \Delta X \in T_X \mathbb{R}^l \times O(n) \times D(n-m) \] with \( \|\Delta X\| \leq \mu_\nu, \) (18)

where “dist” means the Riemannian distance on the Riemannian product manifold \( (\mathbb{R}^l \times O(n) \times D(n-m), g) \) [2, p.46]. Of course, one may choose other retractions on \( O(n) \) via polar decomposition, Givens rotation, Cayley transform, exponential mapping, or singular value decomposition (see for instance [2, p.58] and [28]).

### 3 Convergence Analysis

In this section, we establish the global convergence and local convergence rate of Algorithm 2.1.
3.1 Global Convergence

For the global convergence of Algorithm 2.1, we have the following result. The proof follows from Theorem 4.1 in [28]. Thus we omit it here.

**Theorem 3.1** Any accumulation point $X_*$ of the sequence $\{X_k\}$ generated by Algorithm 2.1 is a stationary point of the cost function $h$ defined in Problem 2.

The search directions $\{\Delta X_k\}$ generated by Algorithm 2.1 have the following property.

**Lemma 3.2** Let $X_*$ be an accumulation point of the sequence $\{X_k\}$ generated by Algorithm 2.1. If $DH(X_*) : T_{X_\ast} \mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m) \to T_{H(X_*)} \mathbb{R}^{n \times n}$ is surjective, then there exist three constants $\bar{\rho}, d_1, d_2 > 0$ such that for all $X_k \in B_{\bar{\rho}}(X_*)$,

$$d_1 \|\text{grad} h(X_k)\| \leq \|\Delta X_k\| \leq d_2 \|\text{grad} h(X_k)\|,$$

where $B_{\bar{\rho}}(X_*) := \{X \in \mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n-m) \mid \text{dist}(X, X_*) < \bar{\rho}\}$.

**Proof:** Since $DH(X_*)$ is surjective by hypothesis and $H$ is continuously differentiable, there exist three positive scalars $\bar{\rho}, \kappa_0, \kappa_1 > 0$ such that for all $X_k \in B_{\bar{\rho}}(X_*)$, $(DH(X_k))^* \circ DH(X_k)$ is nonsingular, and

$$\|(DH(X_k))^* \circ DH(X_k)\| \leq \kappa_0, \quad \|((DH(X_k))^* \circ DH(X_k))^{-1}\| \leq \kappa_1. \quad (19)$$

Let

$$T(X_k) := (DH(X_k))^* \circ DH(X_k)[\Delta X_k] + \text{grad} h(X_k).$$

It follows from (13), (14), and (19) that for all $X_k \in B_{\bar{\rho}}(X_*)$,

$$\|\Delta X_k\| = \|(DH(X_k))^* \circ DH(X_k)[\Delta X_k]\|^{-1} \|T(X_k) - \text{grad} h(X_k)\|$$

$$\leq \|T(X_k)\| + \|(DH(X_k))^* \circ DH(X_k)[\Delta X_k]\|$$

$$\leq \kappa_1(1 + \eta_k)\|\text{grad} h(X_k)\| \leq \kappa_1(1 + \eta_{\text{max}})\|\text{grad} h(X_k)\|$$

$$\equiv d_2\|\text{grad} h(X_k)\|$$

and

$$\|\text{grad} h(X_k)\| = \|T(X_k) - (DH(X_k))^* \circ DH(X_k)[\Delta X_k]\|$$

$$\leq \|T(X_k)\| + \|(DH(X_k))^* \circ DH(X_k)[\Delta X_k]\|$$

$$\leq \eta_k\|\text{grad} h(X_k)\| + \|(DH(X_k))^* \circ DH(X_k)[\Delta X_k]\| \cdot \|\Delta X_k\|$$

$$\leq \eta_{\text{max}}\|\text{grad} h(X_k)\| + \kappa_0\|\Delta X_k\|,$$

which implies that for all $X_k \in B_{\bar{\rho}}(X_*)$,

$$\|\Delta X_k\| \geq \frac{1 - \eta_{\text{max}}}{\kappa_0}\|\text{grad} h(X_k)\| \equiv d_1\|\text{grad} h(X_k)\|.$$
This completes the proof.

For the local convergence of Algorithm 2.1 related to an isolated local minima of \( h \), we have the following result. The proof follows from [5, Proposition 1.2.5].

**Lemma 3.3** Let \( X_* \) be an accumulation point of the sequence \( \{X_k\} \) generated by Algorithm 2.1. If \( DH(X_*) : T_{X_*} \mathbb{R}^l \times O(n) \times D(n-m) \rightarrow T_{H(X_*)} \mathbb{S}^R^{n \times n} \) is surjective and \( X_* \) is an isolated local minimizer of \( h \), then the sequence \( \{X_k\} \) converges to \( X_* \).

**Proof:** By assumption, there exists a parameter \( \hat{\rho} > 0 \) such that \( X_* \) is the only stationary point of \( h \) in the neighborhood \( B_{\hat{\rho}}(X_*) \) and

\[
 h(X) > h(X_*), \quad \forall X \neq X_*, \quad X \in B_{\hat{\rho}}(X_*). \tag{20}
\]

Since \( h \) is continuously differentiable and \( X_* \) is a stationary point of \( h \), i.e., \( \nabla h(X)_* = 0 \), we can obtain

\[
 \lim_{\text{dist}(X,X_*) \to 0} \nabla h(X) = 0_{X_*}. \tag{21}
\]

From (20) and (21), there exists a positive scalar \( 0 < \rho < \min\{\hat{\rho}, \bar{\rho}\} \) such that

\[
 h(X_*) \leq h(X) \quad \text{and} \quad \|\nabla h(X)\| < \mu_{\nu}, \quad \forall X \in B_{\rho}(X_*), \tag{22}
\]

where \( \mu_{\nu} \) is defined in (18) and \( \bar{\rho} \) is defined in Lemma 3.2.

Let

\[
 \phi(t) := \min\{h(X) - h(X_*) \mid X \in \{t \leq \text{dist}(X,X_*) \leq \rho\}\}, \quad \forall t \in [0, \rho].
\]

We note that \( \phi \) is a monotonically nondecreasing function of \( t \) and thus \( \phi(t) > 0 \) for all \( t \in (0, \rho] \). Using (21), for any \( \epsilon \in (0, \rho] \), there exists a constant \( r \in (0, \epsilon] \) such that

\[
 \text{dist}(X,X_*) < r \quad \implies \quad \text{dist}(X,X_*) + \nu d_2 \|\nabla h(X)\| < \epsilon, \tag{23}
\]

where \( \nu \) is defined in (17). Define the open set

\[
 S := \{X \mid \text{dist}(X,X_*) < \epsilon, \ h(X) < h(X_*) + \phi(r)\}.
\]

We claim that if \( X_k \in S \) for some \( k \), then \( X_{k+1} \in S \). Indeed, by using the definitions of \( \phi \) and \( S \), if \( X_k \in S \), then

\[
 \phi(\text{dist}(X_k,X_*)) \leq h(X_k) - h(X_*) < \phi(r), \tag{24}
\]

which implies that \( \text{dist}(X_k,X_*) < r \) since \( \phi \) is monotonically nondecreasing. From (23),

\[
 \text{dist}(X_k,X_*) + \nu d_2 \|\nabla h(X_k)\| < \epsilon. \tag{25}
\]

On the other hand, it follows from Lemma 3.2 (17), and (22) that

\[
 \text{dist}(X_{k+1},X_*) \leq \text{dist}(X_k,X_*) + \text{dist}(X_{k+1},X_k)
\]

\[
 = \text{dist}(X_k,X_*) + \text{dist}(R_{X_k}^{\rho_{k}}\Delta X_k),X_k)
\]

\[
 \leq \text{dist}(X_k,X_*) + \nu \rho_k \|\Delta X_k\| \leq \text{dist}(X_k,X_*) + \nu \|\Delta X_k\|
\]

\[
 \leq \text{dist}(X_k,X_*) + \nu d_2 \|\nabla h(X_k)\|. \tag{26}
\]
Since $h(X_{k+1}) \leq h(X_k)$, it follows from (25) and (26) that
\[
\text{dist}(X_{k+1}, X_*) < \epsilon, \quad h(X_{k+1}) - h(X_*) < \phi(r).
\]
Therefore, we have $X_{k+1} \in S$.

By induction, if $X_k \in S$ for some $k$, then $X_{k+1} \in S$ for all $k \geq k$. Since $X_*$ is an accumulation point of the sequence $\{X_k\}$, there exists a subsequence $\{X_{k_j}\}$ such that $\lim_{j \to \infty} X_{k_j} = X_*$. Then there exists an integer $k_l$ such that $X_{k_l} \in S$ and thus $X_k \in S$ for all $k \geq k_l$. Since $h(X_{k+1}) < h(X_k)$ for all $k \geq k_l$ and $\lim_{k \to \infty} h(X_{k_j}) = h(X_*)$, we can obtain
\[
\lim_{k \to \infty} h(X_k) = h(X_*).
\] (27)

Using (24) and (27) we have $\lim_{k \to \infty} \phi(\text{dist}(X_k, X_*)) = 0$. Since $\phi$ is monotone nondecreasing, it follows that $\lim_{k \to \infty} \text{dist}(X_k, X_*) = 0$ and thus $X_k \to X_*$. The proof is complete. \hfill \Box

3.2 Convergence rate

In this section, we discuss the local convergence rate of Algorithm 2.1. The pullbacks of $H$ and $h$ are defined as $\hat{H} := H \circ R$ and $\hat{h} := h \circ R$, where $R$ is the retraction defined in (16). In addition, we use $\hat{H}_X := H \circ R_X$ and $\hat{h}_X := h \circ R_X$ to denote the restrictions of $\hat{H}$ and $h$ to the tangent space $T_X \mathbb{R}^l \times \mathcal{O}(n) \times D(n-m)$. For the Riemannian gradient of $h$ and the gradient of its pull back $\hat{h}$, it holds that [21, p.56]
\[
\text{grad} h(X) = \text{grad} \hat{h}_X(0_X),
\] (28)
for all $X \in \mathbb{R}^l \times \mathcal{O}(n) \times D(n-m)$. For the differentials of $H$ and its pull back $\hat{H}$, we have
\[
DH(X) = D\hat{H}_X(0_X),
\] (29)
for all $X \in \mathbb{R}^l \times \mathcal{O}(n) \times D(n-m)$.

For the stepsizes $\beta^k$ in (15), we have the following result [22].

Lemma 3.4 Let $X_*$ be an accumulation point of the sequence $\{X_k\}$ generated by Algorithm 2.1. If $DH(X_*) : T_X \mathbb{R}^l \times \mathcal{O}(n) \times D(n-m) \to T_{H(X_*)} \mathbb{S}^n \times n \times n$ is surjective and $\|H(X_*)\|_F$ is sufficiently small, then for $k$ sufficiently large, $l_k = 0$ satisfies (15).

Proof: Let $\Delta X_k^{GN}$ denote the exact solution to (12). Then we have
\[
(DH(X_k))^* \circ DH(X_k)[\Delta X_k - \Delta X_k^{GN}] = \text{grad} h(X_k) + (DH(X_k))^* \circ DH(X_k)[\Delta X_k].
\] (30)

From (12) and (29) it follows that
\[
(D\hat{H}_X(0_X))^*[\hat{H}_{X_k}(0_X)] + (D\hat{H}_X(0_X))^* \circ D\hat{H}_X(0_X)[\Delta X_k^{GN}] = 0_X.
\] (31)
By hypothesis, $DH(X_*)$ is surjective. We know from (10) that if $\|H(X_*)\|_F$ is sufficiently small, then the Riemannian Hessian $\text{Hess} h(X_*)$ is positive definite. Thus $X_*$ is an isolated local minimizer of $h$. We have $X_k \rightarrow X_*$ by Lemma 3.3 Thus, by using Lemma 3.2, 13, 19, and (30), we have for all $k$ sufficiently large,

$$\frac{\|\Delta X_k - \Delta X_k^{GN}\|}{\|\Delta X_k\|^2} \leq \kappa_1 \eta_k \|\text{grad} h(X_k)\| \leq \kappa_1 \|\text{grad} h(X_k)\|^2$$

(32)

In addition, the differential operator $D\tilde{H}_X$ is Lipschitz-continuous at $0_X$ uniformly in a neighborhood of $X_*$. That is, there exist four scalars $\kappa_2, \kappa_3, \delta_1, \delta_2 > 0$, such that

$$\begin{align*}
\| (D\tilde{H}_X(\xi_X))^* - (D\tilde{H}_X(0_X))^* \| &\leq \kappa_2 \|\xi_X\|, \\
\| (D\tilde{H}_X(\xi_X))^* \circ D\tilde{H}_X(\xi_X) - (D\tilde{H}_X(0_X))^* \circ D\tilde{H}_X(0_X) \| &\leq \kappa_3 \|\xi_X\|,
\end{align*}$$

(33)

for all $X \in B_{\delta_1}(X_*)$ and $\xi_X \in B_{\delta_2}(0_X)$. Let

$$G(X_k) := \tilde{H}_{X_k}(\Delta X_k) - \tilde{H}_{X_k}(0_{X_k}) - D\tilde{H}_{X_k}(0_{X_k})[\Delta X_k].$$

(34)

By using Corollary 3.3 in [12], we have

$$\|G(X_k)\| \leq \sup_{\theta \in [0,1]} \|D\tilde{H}_{X_k}(\theta \Delta X_k) - D\tilde{H}_{X_k}(0_{X_k})\| \cdot \|\Delta X_k\|.$$  

(35)

From (34), we obtain

$$\begin{align*}
\tilde{h}_{X_k}(\Delta X_k) &= \frac{1}{2} \|\Delta X_k\|^2 + \|\tilde{H}_{X_k}(0_{X_k}) + D\tilde{H}_{X_k}(0_{X_k})[\Delta X_k] + G(X_k)\|^2 \\
&= \frac{1}{2} \|\tilde{H}_X(0_{X_k})\|^2 + \langle \tilde{H}_{X_k}(0_{X_k}), D\tilde{H}_{X_k}(0_{X_k})[\Delta X_k] \rangle + \frac{1}{2} \langle D\tilde{H}_{X_k}(0_{X_k})[\Delta X_k], D\tilde{H}_{X_k}(0_{X_k})[\Delta X_k] \rangle \\
&\quad + \langle \tilde{H}_{X_k}(0_{X_k}) + D\tilde{H}_{X_k}(0_{X_k})[\Delta X_k], G(X_k) \rangle + \frac{1}{2} \|G(X_k)\|^2 \\
&= \frac{1}{2} \|\tilde{H}_X(0_{X_k})\|^2 + \langle \tilde{H}_{X_k}(0_{X_k}), D\tilde{H}_{X_k}(0_{X_k})[\Delta X_k] \rangle + \frac{1}{2} \langle (D\tilde{H}_{X_k}(0_{X_k}))^* D\tilde{H}_{X_k}(0_{X_k})[\Delta X_k], \Delta X_k \rangle \\
&\quad + \langle \tilde{H}_{X_k}(0_{X_k}) + D\tilde{H}_{X_k}(0_{X_k})[\Delta X_k], G(X_k) \rangle + \frac{1}{2} \|G(X_k)\|^2.
\end{align*}$$

Using (19), (28), (29), (31), (32), (33), (35), and the above equality, we have for all $k$ sufficiently
large,
\[
\begin{align*}
& h(R_{X_k}(\Delta X_k)) - h(X_k) - \frac{1}{2}\langle \text{grad } h(X_k), \Delta X_k \rangle \\
= & \quad \hat{h}_{X_k}(\Delta X_k) - \hat{h}_{X_k}(0) - \frac{1}{2}\langle (D \hat{H}_{X_k}(0X_k))^*[\hat{H}_{X_k}(0X_k)], \Delta X_k \rangle \\
= & \quad \frac{1}{2}\langle \hat{H}_{X_k}(0X_k), D \hat{H}_{X_k}(0X_k)[\Delta X_k] \rangle + \frac{1}{2}\langle (D \hat{H}_{X_k}(0X_k))^*D \hat{H}_{X_k}(0X_k)[\Delta X_k], \Delta X_k \rangle \\
& + \langle \hat{H}_{X_k}(0X_k) + D \hat{H}_{X_k}(0X_k)[\Delta X_k], G(X_k) \rangle + \frac{1}{2}\|G(X_k)\|^2 \\
= & \quad \frac{1}{2}\langle (D \hat{H}_{X_k}(0X_k))^*[\hat{H}_{X_k}(0X_k)] + (D \hat{H}_{X_k}(0X_k))^* \circ D \hat{H}_{X_k}(0X_k)[\Delta X_k^{GN}], \Delta X_k \rangle \\
& + \frac{1}{2}\langle (D \hat{H}_{X_k}(0X_k))^* \circ D \hat{H}_{X_k}(0X_k)[\Delta X_k - \Delta X_k^{GN}], \Delta X_k \rangle \\
& + \langle \hat{H}_{X_k}(0X_k) + D \hat{H}_{X_k}(0X_k)[\Delta X_k^{GN}], G(X_k) \rangle + \frac{1}{2}\|G(X_k)\|^2 \\
& + \langle D \hat{H}_{X_k}(0X_k)[\Delta X_k - \Delta X_k^{GN}], G(X_k) \rangle \\
\leq & \quad 0 + \frac{1}{2}\|\hat{H}_{X_k}(0X_k)\| \cdot \|\Delta X_k - \Delta X_k^{GN}\| \cdot \|\Delta X_k\| \\
& + \|\hat{H}_{X_k}(0X_k) + D \hat{H}_{X_k}(0X_k)[\Delta X_k^{GN}]\| \cdot \|G(X_k)\| + \frac{1}{2}\|G(X_k)\|^2 \\
& + \|D \hat{H}_{X_k}(0X_k)\| \cdot \|\Delta X_k - \Delta X_k^{GN}\| \cdot \|G(X_k)\| \\
& \leq \frac{1}{2} \frac{\kappa_1 \kappa_0}{d_1^2} \|\Delta X_k\|^3 + \kappa_2 \|\hat{H}_{X_k}(0X_k)\|_F \cdot \|\Delta X_k\|^2 + \frac{1}{2} \frac{\kappa_2}{d_1^2} \|\Delta X_k\|^4 \\
& + \frac{\kappa_2}{d_1^2} \|D \hat{H}_{X_k}(0X_k)\| \cdot \|\Delta X_k\|^4 \\
= & \quad \kappa_2 \|H(X_k)\|_F \cdot \|\Delta X_k\|^2 + \frac{1}{2} \frac{\kappa_1 \kappa_0}{d_1^2} \|\Delta X_k\|^3 + \left( \frac{1}{2} \frac{\kappa_2^2}{d_1^2} + \frac{\kappa_1 \kappa_2}{d_1^2} \|D \hat{H}_{X_k}(0X_k)\| \right) \|\Delta X_k\|^4.
\end{align*}
\]
If \(\|H(X_k)\|_F\) is sufficiently small, then \(\|H(X_k)\|_F\) is small enough for all \(k\) sufficiently large. By using the above inequality, (15) holds with \(l_k = 0\) for all \(k\) sufficiently large. This completes the proof. □

We now establish the local convergence rate of Algorithm 2.1.

**Theorem 3.5** Let \(X_0\) be an accumulation point of the sequence \(\{X_k\}\) generated by Algorithm 2.1. If \(DH(X_*) : TX_0 \mathbb{R}^l \times O(n) \times D(n - m) \rightarrow T_{F(X_*)}\mathbb{S}^{n \times n}\) is surjective and \(\|H(X_*)\|_F\) is sufficiently small, then the whole sequence \(\{X_k\}\) converges to \(X_*\) linearly. Furthermore, if \(H(X_*) = 0_{n \times n}\), then the whole sequence \(\{X_k\}\) converges to \(X_*\) quadratically.

**Proof:** By hypothesis, \(DH(X_*)\) is surjective. From (9) and (10) it follows that if \(\|H(X_*)\|_F\) is sufficiently small, then Hess \(h(X_*)\) is positive definite. We have \(X_k \rightarrow X_*\) from Lemma 3.3. By Lemma 3.4, we have \(X_{k+1} = R_{X_k}(\Delta X_k)\) for all \(k\) sufficiently large. Using Lemma 7.4.8 and Lemma 7.4.9 in [2], there exist three scalars \(\tau_0, \tau_1, \tau_2 > 0\) such that for all \(k\) sufficiently large, \(X_k\) is large,

\[
\begin{align*}
\tau_0 \text{dist}(X_k, X_*) \leq \|\text{grad } h(X_k)\| & \leq \tau_1 \text{dist}(X_k, X_*) \\
\|\text{grad } h(X_{k+1})\| = \|\text{grad } h(R_{X_k}(\Delta X_k))\| & \leq \tau_2 \|\text{grad } \hat{h}_{X_k}(\Delta X_k)\|.
\end{align*}
\]

\[
(36)
\]
By using Taylor’s formula we have for all $k$ sufficiently large,

$$
\begin{align*}
\text{grad} \, \tilde{h}_{X_k}(\Delta X_k) &= \text{grad} \, h_{X_k}(0_{X_k}) + (D \tilde{H}_{X_k}(0_{X_k}))^* \circ D \tilde{H}_{X_k}(0_{X_k})[\Delta X_k] \\
&+ \text{Hess} \, \tilde{h}_{X_k}(0_{X_k})[\Delta X_k] - (D \tilde{H}_{X_k}(0_{X_k}))^* \circ D \tilde{H}_{X_k}(0_{X_k})[\Delta X_k] \\
&+ \int_0^1 (\text{Hess} \, \tilde{h}_{X_k}(t\Delta X_k) - \text{Hess} \, \tilde{h}_{X_k}(0_{X_k}))[\Delta X_k] dt.
\end{align*}
$$

(37)

Since $H$ is twice continuously differentiable, it follows from (38) and (39) that there exist two scalars $\kappa_4 > 0$ and $\delta_3 > 0$ such that for all $X \in B_{\delta_3}(X_*)$,

$$
\|\text{Hess} \, \tilde{h}_{X_k}(0_{X_k}) - (D \tilde{H}_{X_k}(0_{X_k}))^* \circ D \tilde{H}_{X_k}(0_{X_k})\| \leq \kappa_4 \|H(X_k)\|_F.
$$

(38)

Furthermore, the Hessian operator $\text{Hess} \, \tilde{h}_X$ is Lipschitz-continuous in a neighborhood of $X_*$, i.e., there exist three scalars $\kappa_5 > 0$, $\delta_4 > 0$, and $\delta_5 > 0$, such that for all $X \in B_{\delta_4}(X_*)$ and $\xi_X \in B_{\delta_5}(0_X)$, it holds that

$$
\|\text{Hess} \, \tilde{h}_X(\xi_X) - \text{Hess} \, \tilde{h}_X(0_X)\| \leq \kappa_5 \|\xi_X\|.
$$

(39)

In addition, $H$ is Lipschitz-continuous in a neighborhood of $X_*$, i.e., there exits two constants $L > 0$ and $\delta_6 > 0$ such that for all $X, Y \in B_{\delta_6}(X_*)$,

$$
\|H(X) - H(Y)\|_F \leq L \text{dist}(X, Y).
$$

(40)

From Lemma 3.2, 13, 36, 37, 38, and 39, we have for $k$ sufficiently large,

$$
\begin{align*}
\frac{\tau_0}{\tau_2} \text{dist}(X_{k+1}, X_*) &\leq \|\text{grad} \, \tilde{h}_{X_k}(\Delta X_k)\| \\
&\leq \|\text{grad} h(X_k) + (D \tilde{H}_{X_k}(0_{X_k}))^* \circ D \tilde{H}_{X_k}(0_{X_k})[\Delta X_k]\| \\
&+ \|\text{Hess} \, \tilde{h}_{X_k}(0_{X_k})[\Delta X_k] - (D \tilde{H}_{X_k}(0_{X_k}))^* \circ D \tilde{H}_{X_k}(0_{X_k})[\Delta X_k]\| \\
&+ \int_0^1 (\text{Hess} \, \tilde{h}_{X_k}(t\Delta X_k) - \text{Hess} \, \tilde{h}_{X_k}(0_{X_k}))[\Delta X_k] dt\|
\end{align*}
$$

$$
\leq \|\text{grad} h(X_k) + (D \tilde{H}_{X_k}(0_{X_k}))^* \circ D \tilde{H}_{X_k}(0_{X_k})[\Delta X_k]\| + \kappa_4 \|H(X_k)\|_F \cdot \|\Delta X_k\| + \kappa_5 \|\Delta X_k\|^2
$$

$$
\leq \eta_k \|\text{grad} h(X_k)\| + \kappa_4 d_2 \|H(X_k)\|_F \cdot \|\text{grad} h(X_k)\| + \kappa_5 d_2^2 \|\text{grad} h(X_k)\|^2
$$

$$
\leq \kappa_4 d_2 \tau_1 \|H(X_k)\|_F \text{dist}(X_k, X_*) + (1 + \kappa_5 d_2^2) \|\text{grad} h(X_k)\|^2
$$

$$
\leq \kappa_4 d_2 \tau_1 \|H(X_k)\|_F \text{dist}(X_k, X_*) + (1 + \kappa_5 d_2^2) \tau_1^2 \text{dist}(X_k, X_*)^2.
$$

(41)

Thus,

$$
\text{dist}(X_{k+1}, X_*) \leq \frac{\tau_1 \tau_2}{\tau_0} \kappa_4 d_2 \|H(X_k)\|_F \text{dist}(X_k, X_*) + \frac{\tau_1^2 \tau_2}{\tau_0} (1 + \kappa_5 d_2^2) \text{dist}(X_k, X_*)^2
$$

$$
= c_1 \|H(X_k)\|_F \text{dist}(X_k, X_*) + c_2 \text{dist}(X_k, X_*)^2,
$$

where $c_1$ and $c_2$ are constants.
where \( c_1 := \frac{\tau_1 \tau_2}{\tau_0} \kappa_4 d_2 \) and \( c_2 := \frac{\tau_1 \tau_2}{\tau_0} (1 + \kappa_5 d_2^2) \).
If \( \|H(X_*)\| \) is sufficiently small, then \( \|H(X_k)\| \) is small enough such that
\( c_1 \|H(X_k)\| < 1 \) for all \( k \) sufficiently large.
Thus if \( \|H(X_*)\| \) is sufficiently small, then \( \{X_k\} \) converges to \( X_* \) linearly.

If \( H(X_*) = 0_{n \times n} \), then we have from (40) for all \( k \) sufficiently large,
\[
\|H(X_k)\| = \|H(X_k) - H(X_*)\| \leq L \text{dist}(X_k, X_*).
\] (42)
Using (41) and (42), we have
\[
\text{dist}(X_{k+1}, X_*) \leq \frac{\tau_1 \tau_2}{\tau_0} \left( \kappa_4 d_2 L + (1 + \kappa_5 d_2^2) \tau_1 \right) \left( \text{dist}(X_k, X_*) \right)^2.
\]
Therefore, if \( H(X_*) = 0_{n \times n} \), then \( \{X_k\} \) converges to \( X_* \) quadratically. This completes the proof.

As a direct consequence of (36) and (41), we have the following result.

**Corollary 3.6** Let \( X_* \) be an accumulation point of the sequence \( \{X_k\} \) generated by Algorithm 2.1. Suppose the assumptions in Theorem 3.5 are satisfied. Then there exists two constants \( \mu_1, \mu_2 > 0 \) such that for all \( k \) sufficiently large,
\[
\|\text{grad } h(X_{k+1})\| \leq \mu_1 \|H(X_k)\| \|\text{grad } h(X_k)\| + \mu_2 \|\text{grad } h(X_k)\|^2.
\]
Furthermore, if \( H(X_*) = 0_{n \times n} \), then there exists a scalar \( \nu > 0 \) such that for all \( k \) sufficiently large,
\[
\|\text{grad } h(X_{k+1})\| \leq \nu \|\text{grad } h(X_k)\|^2.
\]

### 3.3 Surjectivity condition

In this section, we provide the surjectivity condition of \( DH(X_*) \), where \( X_* = (c_*, Q_*, \Lambda_*) \) is an accumulation point of the sequence \( \{X_k\} \) generated by Algorithm 2.1. Based on (4), \( DH(X_*) \) is surjective if and only if the following matrix equation
\[
(A \Delta c) - A_0 + Q_\Lambda Q_\Lambda^T \Delta Q \Delta Q^T - \Delta Q \Lambda \Delta Q^T = 0_{n \times n},
\] (43)
s.t. \( (\Delta c, \Delta Q, \Delta \Lambda) \in T_{X_*} \mathbb{R}^l \times \mathcal{O}(n) \times D(n-m) \)
has a unique solution \( (\Delta c, \Delta Q, \Delta \Lambda) = (0_m, 0_{n \times n}, 0_{(n-m) \times (n-m)}) \in T_{X_*} \mathbb{R}^l \times \mathcal{O}(n) \times D(n-m) \),
where \( \Lambda_* := \text{blkdiag}(\Lambda_m^*, \Lambda_*) \) and \( P \) is defined in (5).

For \( W \in \mathbb{R}^{n \times n} \), define \( \text{vec}(W) \in \mathbb{R}^{n(n+1)/2} \) by
\[
\text{vec}(W) \left( \frac{(j-1)(j-2)}{2} + i \right) := W_{ij}, \quad i < j, \quad j = 2, \ldots, n.
\]
This shows that \( \text{vec}(W) \) is a column vector obtained by stacking the strictly upper triangular part of \( W \). For \( \mathbf{w} \in \mathbb{R}^{n(n+1)/2} \), define \( \text{skew}(\mathbf{w}) \in \mathbb{R}^{n \times n} \) by
\[
\text{vec}(\text{skew}(\mathbf{w})) := \mathbf{w}, \quad \text{vec}\left( (\text{skew}(\mathbf{w}))^T \right) := -\mathbf{w}.
\]
and
\[
(\widehat{\text{skew}}(w))_{ii} = 0, \quad i = 1, 2, \ldots, n.
\]

We observe that \(\widehat{\text{skew}}(w)\) is a skew-symmetric matrix constructed from \(w\). Therefore, \(\text{vec}\) and \(\text{skew}\) are a pair of inverse operators. In addition, there exists a matrix \(\widehat{P} \in \mathbb{R}^{n^2 \times \frac{n(n-1)}{2}}\) such that
\[
\text{vec}(\widehat{\text{skew}}(w)) = \widehat{P}w
\] (44)

for all \(w \in \mathbb{R}^{\frac{n(n-1)}{2}}\). Since \(\Delta Q \in T_Q \mathcal{O}(n)\), there exists a skew-symmetric matrix \(\Delta \Omega \in \mathbb{R}^{n \times n}\) such that \(\Delta Q = Q \Delta \Omega\). For \(\Delta \Omega \in \mathbb{R}^{n \times n}\), it follows from (44) that there exists a vector \(\Delta v \in \mathbb{R}^{n-m}\) such that
\[
\text{vec}(\Delta \Omega) = \widehat{P} \Delta v.
\] (45)

Thus, we have
\[
\text{vec}(\Delta Q) = \text{vec}(Q \Delta \Omega) = (I_n \otimes Q) \text{vec}(\Delta \Omega) = (I_n \otimes Q) \widehat{P} \Delta v,
\]

where \(\otimes\) means the Kronecker product. Let \(\widehat{A}\) be an \(n^2 \times l\) matrix defined by
\[
\widehat{A} := [\text{vec}(A_1), \text{vec}(A_2), \ldots, \text{vec}(A_l)] \in \mathbb{R}^{n^2 \times l}.
\] (46)

Since \(\Delta \Lambda \in \mathcal{D}(n - m)\), there exists a matrix \(G \in \mathbb{R}^{(n-m)^2 \times (n-m)}\) and a vector \(\Delta w \in \mathbb{R}^{n-m}\) such that
\[
\text{vec}(\Delta \Lambda) = G \Delta w.
\] (47)

Based on (45), (46), and (47), the vectorization of the matrix equation (43) is given by
\[
\begin{bmatrix}
\hat{A}, (Q_\ast \otimes Q_\ast)(I_n \otimes \bar{X} - \bar{X} \otimes I_n) \hat{P}, (Q_\ast P) \otimes (Q_\ast P) G
\end{bmatrix}
\begin{bmatrix}
\Delta e \\
\Delta v \\
\Delta w
\end{bmatrix} = 0_{n^2}.
\] (48)

Based on the above analysis, we have the following surjectivity condition of \(DH(X_\ast)\).

**Theorem 3.7** Let \(X_\ast = (c_\ast, Q_\ast, \Lambda_\ast)\) be an accumulation point of the sequence \(\{X_k\}\) generated by Algorithm 2.1. Then \(DH(X_\ast)\) is surjective if and only if the following matrix
\[
\begin{bmatrix}
\hat{A}, (Q_\ast \otimes Q_\ast)(I_n \otimes \bar{X} - \bar{X} \otimes I_n) \hat{P}, (Q_\ast P) \otimes (Q_\ast P) G
\end{bmatrix}
\]
is of full rank.

### 3.4 Preconditioning technique

In this section, we propose a preconditioner for solving (12). Here we adapt a centered preconditioner [23, p.279]. For the CG method, instead of solving (12), we solve the following preconditioned linear system
\[
\left\{
\begin{aligned}
(DH(X_k))^* \circ M_k^{-1} \circ DH(X_k)[\Delta X_k] &= -(DH(X_k))^* \circ M_k^{-1}[H(X_k)], \\
\text{s.t.} \quad \Delta X_k &\in T_{(c,Q,\Lambda)} \mathbb{R}^l \times \mathcal{O}(n) \times \mathcal{D}(n - m),
\end{aligned}
\right.
\]

where \(M_k : T_{H(X_k)} \mathbb{S}^{n \times n} \to T_{H(X_k)} \mathbb{S}^{n \times n}\) is a self-adjoint and positive definite linear operator.
An efficient centered preconditioner $M_k$ may be defined by
\[
M_k[\Delta Z_k] := (A(\nu(\Delta Z_k)) - A_0) + [Q_k \overline{X}_k Q_k^T, [Q_k \overline{X}_k Q_k^T, \Delta Z_k]] + Q_k P P^T Q_k^T \Delta Z_k Q_k P P^T Q_k^T + i \Delta Z_k, \tag{49}
\]
for all $\Delta Z_k \in T_{H(X_k)} \mathbb{S}^n$, where $\hat{t} > 0$ is a given constant. Using (4) and (6) we have
\[
(DH(X_k) \circ (DH(X_k))^\ast + \hat{t} \text{Id}_{T_{H(X_k)} \mathbb{S}^n})[\Delta Z_k] = (A(\nu(\Delta Z_k)) - A_0) + [Q_k \overline{X}_k Q_k^T, [Q_k \overline{X}_k Q_k^T, \Delta Z_k]] + Q_k P \text{Diag}(P^T Q_k^T \Delta Z_k Q_k P) P^T Q_k^T + i \Delta Z_k,
\]
for all $\Delta Z_k \in T_{H(X_k)} \mathbb{S}^n$, where $\text{Id}_{T_{H(X_k)} \mathbb{S}^n}$ means the identity mapping on $T_{H(X_k)} \mathbb{S}^n$. This shows that
\[
M_k \approx DH(X_k) \circ (DH(X_k))^\ast + \hat{t} \text{Id}_{T_{H(X_k)} \mathbb{S}^n}.
\]
From (49) we see that for any $\Delta Z_k \in T_{H(X_k)} \mathbb{S}^n$,
\[
\text{vec}(M_k[\Delta Z_k]) = \text{vec}(A(\nu(\Delta Z_k)) - A_0) + (Q_k P P^T Q_k^T \otimes (Q_k P P^T Q_k^T)) \text{vec}(\Delta Z_k) + ((Q_k \otimes Q_k)(I_n \otimes \overline{X}_k - \overline{X}_k \otimes I_n)^2(Q_k^T \otimes Q_k^T) + i I_n^2) \text{vec}(\Delta Z_k) = (Q_k \otimes Q_k)((I_n \otimes \overline{X}_k - \overline{X}_k \otimes I_n)^2 + (P P^T) \otimes (P P^T) + i I_n^2)(Q_k^T \otimes Q_k^T) \text{vec}(\Delta Z_k) + \hat{A} \hat{A}^T \text{vec}(\Delta Z_k).
\]
Let
\[
\hat{B}_k := (Q_k \otimes Q_k)((I_n \otimes \overline{X}_k - \overline{X}_k \otimes I_n)^2 + (P P^T) \otimes (P P^T) + i I_n^2)(Q_k^T \otimes Q_k^T).
\]
It is clear that $\hat{B}_k$ is positive definite. Thus, for any $\Delta Z_k \in T_{H(X_k)} \mathbb{S}^n$,
\[
\begin{cases}
\text{vec}(M_k[\Delta Z_k]) = (\hat{B}_k + \hat{A}\hat{A}^T) \text{vec}(\Delta Z_k), \\
\text{vec}(M_k^{-1}[\Delta Z_k]) = (\hat{B}_k + \hat{A}\hat{A}^T)^{-1} \text{vec}(\Delta Z_k).
\end{cases}
\]
Note that
\[
\hat{A}\hat{A}^T = \sum_{i=1}^{l} \text{vec}(A_i) \text{vec}(A_i)^T.
\]
Thus the matrix $\hat{A}\hat{A}^T$ is a low rank matrix, i.e., rank($\hat{A}\hat{A}^T$) $\leq l$. Let
\[
\hat{M}_k := \hat{B}_k + \hat{A}\hat{A}^T.
\]
By assumption, $l < m \leq n < n^2$, $\hat{M}_k$ is a low rank perturbation of $\hat{B}_k$. By using the Sherman-Morrison-Woodbury formula [19], we can obtain
\[
\hat{M}_k^{-1} = (\hat{B}_k + \hat{A}\hat{A}^T)^{-1} = \hat{B}_k^{-1} - \hat{B}_k^{-1} \hat{A} (I_l + \hat{A}^T \hat{B}_k^{-1} \hat{A})^{-1} \hat{A}^T \hat{B}_k^{-1},
\]
where $\hat{A}$ is the matrix $\hat{A}$.
where
\[
\hat{B}^{-1}_k = (Q_k \otimes Q_k) \left( (I_n \otimes \Lambda_k - \Lambda_k \otimes I_n)^2 + (PP^T) \otimes (PP^T) + \hat{t} I_{n^2} \right)^{-1} (Q^T_k \otimes Q^T_k),
\]
which can be computed easily. For any vector \(x \in \mathbb{R}^{n^2}\), the matrix-vector products \((Q_k \otimes Q_k)x\) and \((Q^T_k \otimes Q^T_k)x\) can be computed via
\[
(Q_k \otimes Q_k)x = \text{vec}(Q_k \hat{X} Q^T_k) \quad \text{and} \quad (Q^T_k \otimes Q^T_k)x = \text{vec}(Q^T_k \hat{X} Q_k),
\]
where \(\hat{X} \in \mathbb{R}^{n \times n}\) is the matrix such that \(\text{vec}(\hat{X}) = x\). We conclude that the matrix-vector product \(\hat{M}^{-1}_k x\) can be computed efficiently, where the main computational cost is to calculate the inverse of \((I_l + \hat{A}^T \hat{B}^{-1}_k \hat{A}) \in \mathbb{R}^{l \times l}\).

4 Numerical Experiments

In this section we report the numerical performance of Algorithm 2.1 for solving Problem (2). All the numerical tests are carried out by using MATLAB 7.1 running on a workstation with an Intel Xeon CPU E5-2687W at 3.10 GHz and 32 GB of RAM. To illustrate the efficiency of our algorithm, we compare Algorithm 2.1 with the LP-Newton method (LP-N) in [6].

In our numerical tests, we set \(\beta = 0.5, \eta_{\max} = 0.01, \sigma = 10^{-4}, \text{ and } \hat{t} = 10^{-5}\). The largest number of iterations in Algorithm 2.1 and the LP-Newton method is set to be \(10^5\), and the largest number of iterations in the CG method is set to be \(n^3\). Let ‘CT.’, ‘IT.’, ‘LP.’, ‘NF.’, ‘NCG.’, ‘Res.’, ‘\text{grad}.’, and ‘err-c.’ denote the averaged total computing time in seconds, the averaged number of outer Newton or Gauss-Newton iterations, the averaged number of LP iterations, the averaged number of function evaluations, the averaged total number of inner CG iterations, the averaged residual \(\|H(X_k)\|_F\) or \(\sqrt{2f(c_k, \sigma^*)}\), the averaged residual \(\|\text{grad } h(X_k)\|\) or \(\|\text{grad } f(c_k, \sigma^*)\|\), and the averaged relative error \(\|c_k - \hat{c}\|_{\infty}/\|\hat{c}\|_{\infty}\) at the final iterates of the corresponding algorithms, accordingly.

For the LP-Newton method, the stopping criterion for the LP step is set to be
\[
\|c_k - c_{k-1}\|_F < 10^{-3}
\]
and the stopping criterion for the Newton step is set to be
\[
\|\text{grad } f(c_k, \sigma^*)\|_F < \zeta,
\]
and the stopping criterion for Algorithm 2.1 is set to be
\[
\|\text{grad } h(X_k)\|_F < \zeta,
\]
where \(\zeta > 0\) is the prescribed tolerance.

We consider the following three examples.
Example 4.1 \[6\] We consider the LSIEP with \( n = l = m = 5 \). Let
\[
A_0 = \begin{bmatrix}
0 & -1 & 0 & 0 & 0 \\
-1 & 0 & -1 & 0 & 0 \\
0 & -1 & 0 & -1 & 0 \\
0 & 0 & -1 & 0 & -1 \\
0 & 0 & 0 & -1 & 0 \\
\end{bmatrix}, \quad A_k = 4e_k e_k^T, \quad k = 1, 2, \ldots, 5.
\]
We choose \( \{1, 1, 2, 3, 4\} \) as the prescribed spectrum.

Example 4.2 We consider the Sturm-Liouville problem of the form:
\[
- \frac{d^2 y}{dx^2} + q(x)y = \lambda y, \quad 0 \leq x \leq \pi, \tag{50}
\]
where \( q \) is a real, square-integrable function and the following Dirichlet boundary conditions are imposed
\[y(0) = y(\pi) = 0.\]
By using the Rayleigh-Ritz method in \[20\], the Rayleigh quotient of (50) is given by
\[
R(y(x)) = \frac{\int_0^\pi ((y'(x))^2 + q(x)y(x)^2)dx}{\int_0^\pi y(x)^2dx}.
\]
Suppose that \( y(x) = \sum_{j=1}^n w_j \sin(jx) \). By simple calculation, we have
\[
R(y(x)) = \frac{\sum_{i=1}^n \sum_{j=1}^n i \cdot j \cdot w_i \cdot w_j \cdot \delta_{ij}^j + \frac{2}{\pi} \sum_{i=1}^n \sum_{j=1}^n w_i \cdot w_j \int_0^\pi q(x) \sin(ix) \sin(jx)dx}{\sum_{i=1}^n \sum_{j=1}^n w_i \cdot w_j \cdot \delta_{ij}^j},
\]
i.e.,
\[
R(y(x)) = \frac{w^T A w}{w^T w},
\]
where \( w := (w_1, w_2, \ldots, w_n)^T \) and the entries of the symmetric matrix \( A = [a_{ij}] \in \mathbb{R}^{n \times n} \) are given by
\[
a_{ij} = i \cdot j \cdot \delta_{ij}^j + \frac{2}{\pi} \int_0^\pi q(x) \cos((i-j)x) \cos((i+j)x) dx,
\]
for \( i, j = 1, 2, \ldots, n \). If \( q(x) = 2 \sum_{k=1}^l c_k \cos(2kx) \), then one has
\[
a_{ij} = i \cdot j \cdot \delta_{ij}^j + \sum_{k=1}^l c_k \cdot (\delta_{|i-j|}^{2k} - \delta_{i+j}^{2k}), \quad i, j = 1, 2, \ldots, n.
\]
Let $T_k (k = 1, 2, \ldots, n - 1)$ and $H_k (k = 1, 2, \ldots, 2n - 1)$ be $n \times n$ real matrices generated by the MATLAB built-in functions toeplitz and hankel:
\[
T_k = \text{toeplitz}(e_{k+1}), \quad k = 1, 2, \ldots, n - 1
\]
and
\[
H_k = \begin{cases}
\text{hankel}(e_k, 0_n), & k = 1, 2, \ldots, n, \\
\text{hankel}(0_n, e_{k-n+1}), & k = n + 1, n + 2, \ldots, 2n - 1.
\end{cases}
\]
Define
\[
A_0 = \text{diag}(1^2, 2^2, 3^2, \ldots, n^2), \quad A_k = \begin{cases}
T_{2k} - H_{2k-1}, & 1 \leq k \leq \min\{l, \frac{n-1}{2}\}, \\
-H_{2k-1}, & \min\{l, \frac{n-1}{2}\} < k \leq l.
\end{cases}
\]
Then
\[
A = A_0 + \sum_{k=1}^l c_k A_k \equiv A(c).
\]
To estimate the first $l$ Fourier coefficients of the potential $q(x)$ defined by [20]
\[
q(x) = \sum_{k=1}^{\infty} \frac{192}{\pi^4} \frac{1}{k^4} \cos(2kx),
\]
we consider the LSIEP with above $\{A_k\}$ and the $n$ eigenvalues of $A(\hat{c})$ as the prescribed spectrum for varying $n = m$ and $l$, where the entries of $\hat{c}$ are given by
\[
\hat{c}_k = \frac{192}{\pi^4} \frac{1}{k^4}, \quad k = 1, 2, \ldots, l.
\]

**Example 4.3** We consider the LSIEP with varying $n$, $l$, and $m$. Let $\hat{c} \in \mathbb{R}^l$ be a random vector and $A_0, A_1, \ldots, A_l$ be $n \times n$ random symmetric matrices, which are generated by the MATLAB built-in function randn:
\[
\hat{c} := \text{randn}(l, 1), \quad B_k := \text{randn}(n, n), \quad A_k = \frac{1}{2}(B_k + B_k^T), \quad k = 0, 1, \ldots, l.
\]
We choose the $m$ smallest eigenvalues of $A(\hat{c})$ as the prescribed partial spectrum.

For Algorithm 2.1 and the LP-Newton method in [6], the starting points are generated by the MATLAB built-in function eig:
\[
[Q_0, \bar{\Lambda}] = \text{eig}(A(c_0), 'real'), \quad \Lambda_0 = \bar{\Lambda}(m + 1 : n).
\]
For Example 4.1, $c_0$ is set to be
\[
c_0 = (0.6316, 0.2378, 0.9092, 0.9866, 0.5007)^T.
\]
For Example 4.2, \( c_0 \) is set to be a zero vector. For Example 4.3, \( c_0 \) is formed by chopping the components of \( \hat{c} \) to two decimal places for \( n < 100 \) and to three decimal places for \( n \geq 100 \).

We first apply the LP-Newton method and Algorithm 2.1 to Example 4.1 with \( \zeta = 10^{-7} \). Both methods converge to the same least squares solution:

\[
c_* = (0.4423, 0.6044, 0.6566, 0.6044, 0.4423)^T
\]

and the spectrum of \( A(c_*) \) is \{0.5888, 1.0422, 2.0742, 3.1446, 4.1501\}.

Table 1 lists numerical results for Example 4.1. We see from Table 1 that Algorithm 2.1 is not as effective as the LP-Newton method since \( \|H(X_*)\|_F = 0.4688 \) is not small enough.

| Alg. | CT. | LP. | IT. | NF. | NCG. | Res. | grad. |
|------|-----|-----|-----|-----|------|------|-------|
| LP-N | 0.0320 s | 29 | 2 | 3 | 4.5 | 0.4688 | 1.12 \times 10^{-8} |
| Alg. 2.1 with CG | 0.6880 s | 635 | 2776 | 18.3 | 0.4688 | 8.99 \times 10^{-8} |
| Alg. 2.1 with PCG | 0.3280 s | 655 | 2855 | 1.4 | 0.4688 | 9.83 \times 10^{-8} |

We now apply the LP-Newton method and Algorithm 2.1 to Example 4.2 with \( \zeta = 10^{-8} \). Table 2 displays numerical results for Example 4.2. We observe that Algorithm 2.1 works much better than the LP-Newton method in terms of computing time. We also see that the proposed preconditioner is very efficient.

| Alg. | \((n, l, m)\) | CT. | LP. | IT. | NF. | NCG. | Res. | grad. | err-c. |
|------|---------------|-----|-----|-----|-----|------|------|-------|-------|
| LP-N | (10, 6, 10) | 0.1560 s | 53 | 2 | 3 | 6 | 7.98 \times 10^{-11} | 1.05 \times 10^{-10} | 2.44 \times 10^{-11} |
| (20, 12, 20) | 1.4630 s | 101 | 2 | 3 | 7 | 1.14 \times 10^{-9} | 1.45 \times 10^{-9} | 3.31 \times 10^{-10} |
| (30, 18, 30) | 8.4830 s | 144 | 2 | 3 | 7 | 4.57 \times 10^{-9} | 5.81 \times 10^{-9} | 1.35 \times 10^{-9} |
| (40, 22, 40) | 30.109 s | 181 | 3 | 4 | 7 | 3.77 \times 10^{-12} | 2.56 \times 10^{-12} | 1.69 \times 10^{-12} |
| (50, 34, 50) | 02 m 01 s | 216 | 3 | 4 | 7 | 6.29 \times 10^{-12} | 5.35 \times 10^{-12} | 1.46 \times 10^{-12} |
| Alg. 2.1 with CG | (10, 6, 10) | 0.0450 s | 6 | 7 | 30.3 | 4.94 \times 10^{-14} | 1.41 \times 10^{-13} | 2.01 \times 10^{-14} |
| (20, 12, 20) | 0.0600 s | 6 | 7 | 133.7 | 1.16 \times 10^{-10} | 1.97 \times 10^{-10} | 4.44 \times 10^{-11} |
| (30, 18, 30) | 0.1830 s | 6 | 7 | 355.2 | 3.48 \times 10^{-11} | 1.99 \times 10^{-10} | 9.40 \times 10^{-12} |
| (40, 22, 40) | 0.4890 s | 6 | 7 | 700.8 | 3.22 \times 10^{-12} | 3.05 \times 10^{-11} | 1.57 \times 10^{-12} |
| (50, 34, 50) | 0.9690 s | 6 | 7 | 1164 | 4.68 \times 10^{-12} | 4.87 \times 10^{-11} | 1.26 \times 10^{-12} |
| Alg. 2.1 with PCG | (10, 6, 10) | 0.0420 s | 5 | 6 | 1.2 | 5.56 \times 10^{-14} | 1.69 \times 10^{-13} | 2.83 \times 10^{-14} |
| (20, 12, 20) | 0.0120 s | 5 | 6 | 1.2 | 2.92 \times 10^{-13} | 9.67 \times 10^{-13} | 7.19 \times 10^{-14} |
| (30, 18, 30) | 0.0080 s | 5 | 6 | 1.2 | 1.32 \times 10^{-12} | 2.02 \times 10^{-12} | 7.17 \times 10^{-13} |
| (40, 22, 40) | 0.0100 s | 5 | 6 | 1.2 | 2.81 \times 10^{-12} | 4.85 \times 10^{-12} | 1.35 \times 10^{-12} |
| (50, 34, 50) | 0.0180 s | 5 | 6 | 1.2 | 3.72 \times 10^{-12} | 1.13 \times 10^{-11} | 1.49 \times 10^{-12} |

Next, we apply the LP-Newton method and Algorithm 2.1 to Example 4.3 with \( \zeta = 10^{-8} \). For comparison purposes, we repeat our experiments over 10 different problems. Table 3 shows numerical results for Example 4.3. We observe from Table 3 that Algorithm 2.1 is more effec-
tive than the LP-Newton method in terms of computing time. We also see that the proposed preconditioner can reduce the number of inner CG iterations effectively.

To further illustrate the efficiency of Algorithm 2.1 we apply Algorithm 2.1 with the proposed preconditioner to Examples 4.2–4.3 for varying $n, l, m$. The corresponding numerical results are displayed in Tables 4–5. We see from Tables 4–5 that Algorithm 2.1 with the proposed preconditioner works very efficient for different values of $n, l, m$. Finally, the quadratic convergence of Algorithm 2.1 is observed from Figure 1 which agrees with our prediction.

Table 3: Comparison results for Example 4.3

| Alg. | $(n, l, m)$ | CT. | LP. | IT. | NF. | NCG. | Err. | Res. | err-c. |
|------|-------------|-----|-----|-----|-----|------|------|------|--------|
| LP-N | (10, 6, 8)  | 0.0794 s | 3.3 | 3.1 | 4.1 | 60 | $4.22 \times 10^{-11}$ | $6.15 \times 10^{-11}$ | $2.28 \times 10^{-11}$ |
|      | (20, 10, 18)| 1.1105 s | 3.5 | 3.0 | 4.0 | 40 | $4.89 \times 10^{-11}$ | $1.51 \times 10^{-10}$ | $1.14 \times 10^{-11}$ |
|      | (30, 16, 25)| 6.6156 s | 5.0 | 3.0 | 4.0 | 68 | $3.74 \times 10^{-10}$ | $1.31 \times 10^{-9}$ | $4.30 \times 10^{-11}$ |
|      | (40, 20, 32)| 22.379 s | 5.5 | 3.2 | 4.2 | 85 | $3.57 \times 10^{-10}$ | $1.92 \times 10^{-9}$ | $4.43 \times 10^{-11}$ |
|      | (50, 34, 42)| 42 m 29 s | 6.4 | 4.4 | 5.4 | 392 | $1.01 \times 10^{-10}$ | $5.23 \times 10^{-10}$ | $5.99 \times 10^{-12}$ |

| Alg. 2.1 with CG | $(n, l, m)$ | CT. | LP. | IT. | NF. | NCG. | Err. | Res. | err-c. |
|------------------|-------------|-----|-----|-----|-----|------|------|------|--------|
| (10, 6, 8)       | 0.0150 s    | 3.9 | 4.9 | 676 |      | $5.31 \times 10^{-11}$ | $5.78 \times 10^{-10}$ | $1.80 \times 10^{-12}$ |
| (20, 10, 18)     | 0.0890 s    | 4.2 | 5.2 | 919 |      | $1.16 \times 10^{-12}$ | $2.60 \times 10^{-12}$ | $9.49 \times 10^{-14}$ |
| (30, 16, 25)     | 0.2763 s    | 4.7 | 5.7 | 2130 | $2.75 \times 10^{-11}$ | $9.27 \times 10^{-11}$ | $1.75 \times 10^{-12}$ |
| (40, 20, 32)     | 1.0520 s    | 8.0 | 41.7 | 3799 | $1.98 \times 10^{-12}$ | $4.49 \times 10^{-11}$ | $3.27 \times 10^{-14}$ |
| (50, 34, 42)     | 1.8564 s    | 6.7 | 17.6 | 15517 | $7.69 \times 10^{-11}$ | $7.04 \times 10^{-10}$ | $5.76 \times 10^{-12}$ |

Table 4: Numerical results for Example 4.2

| $(n, l, m)$ | CT. | IT. | NF. | NCG. | Res. | grad. | err-c. |
|-------------|-----|-----|-----|------|------|-------|--------|
| (100, 50, 100) | 0.0920 s | 5 | 6 | 1.2 | $3.24 \times 10^{-11}$ | $5.79 \times 10^{-11}$ | $1.49 \times 10^{-11}$ |
| (200, 60, 200) | 0.3800 s | 5 | 6 | 1.2 | $2.46 \times 10^{-10}$ | $3.42 \times 10^{-10}$ | $6.45 \times 10^{-11}$ |
| (300, 70, 300) | 1.0400 s | 5 | 6 | 1.2 | $5.73 \times 10^{-10}$ | $9.52 \times 10^{-10}$ | $1.61 \times 10^{-10}$ |
| (400, 75, 400) | 2.3570 s | 5 | 6 | 1.2 | $1.78 \times 10^{-9}$ | $1.87 \times 10^{-9}$ | $4.00 \times 10^{-10}$ |
| (500, 80, 500) | 3.8170 s | 5 | 6 | 1.2 | $2.97 \times 10^{-9}$ | $3.14 \times 10^{-9}$ | $4.01 \times 10^{-10}$ |
| (600, 85, 600) | 5.8630 s | 5 | 6 | 1.2 | $4.76 \times 10^{-9}$ | $4.74 \times 10^{-9}$ | $9.18 \times 10^{-10}$ |
| (800, 90, 800) | 12.032 s | 5 | 6 | 1.2 | $1.12 \times 10^{-8}$ | $9.54 \times 10^{-9}$ | $2.29 \times 10^{-9}$ |

5 Conclusions

In this paper, we have proposed a preconditioned Riemannian inexact Gauss-Newton method for solving the least squares inverse eigenvalue problem. The global and local convergence analysis of the method is established under some conditions. Numerical experiments show the efficiency of the proposed method.
Table 5: Numerical results for Example 4.3.

| (n, l, m)      | CT.    | IT. | NF. | NCG. | Err.     | Res.    | err-c.    |
|----------------|--------|-----|-----|------|----------|---------|-----------|
| (100, 60, 80)  | 0.0905 s | 3.0 | 4.0 | 27   | 3.04 × 10⁻¹¹ | 4.17 × 10⁻¹⁰ | 3.73 × 10⁻¹⁴ |
| (200, 120, 160)| 0.6575 s | 3.0 | 4.0 | 33   | 6.86 × 10⁻¹¹ | 1.85 × 10⁻⁹  | 4.00 × 10⁻¹⁴ |
| (300, 160, 200)| 3.3588 s | 3.7 | 4.7 | 64   | 2.14 × 10⁻¹¹ | 1.43 × 10⁻⁹  | 3.71 × 10⁻¹⁴ |
| (400, 220, 280)| 10.702 s | 4.0 | 5.0 | 65   | 9.42 × 10⁻¹² | 2.51 × 10⁻⁹  | 4.19 × 10⁻¹⁴ |
| (500, 340, 400)| 22.921 s | 4.0 | 5.0 | 48   | 2.48 × 10⁻¹¹ | 5.64 × 10⁻⁹  | 6.29 × 10⁻¹⁴ |
| (600, 420, 480)| 46.968 s | 4.2 | 5.2 | 54   | 2.13 × 10⁻¹¹ | 9.58 × 10⁻⁹  | 7.95 × 10⁻¹⁴ |

Figure 1: Convergence history of two tests.

References

[1] P.-A. Absil, C. G. Baker, and K. A. Gallivan, Trust-region methods on Riemannian manifolds, Found. Comput. Math., 7 (2007), pp. 303–330.

[2] P.-A. Absil, R. Mahony, and R. Sepulchre, Optimization Algorithms on Matrix Manifolds, Princeton University Press, Princeton, NJ, 2008.

[3] P.-A. Absil and J. Malick, Projection-like retractions on matrix manifolds, SIAM J. Optim., 22 (2012), pp. 135–158.

[4] R. L. Adler, J.-P. Dedieu, J. Y. Margulies, M. Martens, and M. Shub, Newton’s method on Riemannian manifolds and a geometric model for the human spine, IMA J. Numer. Anal., 22 (2002), pp. 359–390.

[5] D. P. Bertsekas, Nonlinear Programming, 2nd edition, Athena Scientific, Belmont, 1999.

[6] X. Z. Chen and M. T. Chu, On the least-squares solution of inverse eigenvalue problems, SIAM J. Numer. Anal., 33 (1996), pp. 2417–2430.

[7] M. T. Chu, Inverse eigenvalue problems, SIAM Rev., 40 (1998), pp. 1–39.
M. T. Chu and K. R. Driessel, *Constructing symmetric nonnegative matrices with prescribed eigenvalues by differential equations*, SIAM J. Math. Anal., 22 (1991), pp. 1372–1387.

M. T. Chu and G. H. Golub, *Structured inverse eigenvalue problems*, Acta Numer., 11 (2002), pp. 1–71.

M. T. Chu and G. H. Golub, *Inverse Eigenvalue Problems: Theory, Algorithms, and Applications*, Oxford University Press, Oxford, UK, 2005.

M. T. Chu and Q. Guo, *A numerical method for the inverse stochastic spectrum problem*, SIAM J. Matrix Anal. Appl., 19 (1998), pp. 1027–1039.

R. Coleman, *Calculus on Normed Vector Spaces*, Springer-Verlag, New York, 2012.

S. J. Cox, M. Embree, and J. M. Hokanson, *One can hear the composition of a string: experiments with an inverse eigenvalue problem*, SIAM Rev., 54 (2012), pp. 157–178.

B. N. Datta, *Numerical Methods for Linear Control Systems: Design and Analysis*, Elsevier Academic Press, London, UK, 2003.

M. I. Friswell and J. E. Mottershead, *Finite Element Model Updating in Structural Dynamics*, Kluwer Academic Publishers, Dordrecht, NED, 1995.

G. M. L. Gladwell, *Inverse Problems in Vibration*, Kluwer Academic Publishers, Dordrecht, NED, 2004.

G. H. Golub and C. F. Van Loan, *Matrix Computations*, 4th edition, Johns Hopkins University Press, Baltimore, 2013.

S. Gratton, A. S. Lawless, and N. K. Nichols, *Approximate Gauss-Newton methods for nonlinear least squares problems*, SIAM J. Optim., 18 (2007), pp. 106–132.

W.W. Hager, *Updating the inverse of a matrix*, SIAM Rev., 31 (1989), pp. 221–239.

O. H. Hald, *The inverse Sturm-Liouville problem and the Rayleigh-Ritz method*, Math. Comp., 32 (1978), pp. 687–705.

U. Helmke and J. B. Moore, *Optimization and Dynamical Systems*, Springer-Verlag, London, UK, 1994.

W. Ring and B. Wirth, *Optimization methods on Riemannian manifolds and their application to shape space*, SIAM J. Optim., 22 (2012), pp. 596–627.

Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd Edition, SIAM, Philadelphia, 2003.

S. T. Smith, *Optimization techniques on Riemannian manifolds*, in Hamiltonian and Gradient Flows, Algorithms and Control, A. Bloch, ed., Fields Inst. Commun. 3, AMS, Providence, RI, 1994, pp. 113–136.
[25] Z. B. Wang and S. W. Vong, A Guass-Newton-like method for inverse eigenvalue problems, Inter. J. Comput. Math., 90 (2013), pp. 1435–1447.

[26] S. F. Xu, An Introduction to Inverse Algebraic Eigenvalue Problems, Peking University Press, Beijing; Friedr. Vieweg & Sohn, Braunschweig, 1998.

[27] T. T. Yao, Z. J. Bai, Z. Zhao, and W. K. Ching, A Riemannian Fletcher–Reeves conjugate gradient method for doubly stochastic inverse eigenvalue problems, SIAM J. Matrix Anal. Appl., 37 (2016), pp. 215–234.

[28] Z. Zhao, Z. J. Bai, and X. Q. Jin, A Riemannian Newton algorithm for nonlinear eigenvalue problems, SIAM J. Matrix Anal. Appl., 36 (2015), pp. 752–774.

[29] Z. Zhao, Z. J. Bai, and X. Q. Jin, A Riemannian Inexact Newton-CG Method for Nonnegative Inverse Eigenvalue Problems: Nonsymmetric Case, https://arxiv.org/abs/1706.03480, 2017.

[30] Z. Zhao, X. Q. Jin, and Z. J. Bai, A geometric nonlinear conjugate gradient method for stochastic inverse eigenvalue problems, SIAM J. Numer. Anal., 54 (2016), pp. 2015–2035.