RIEMANNIAN NEWTON METHOD FOR THE MULTIVARIATE EIGENVALUE PROBLEM∗
LEI-HONG ZHANG†

Abstract. The multivariate eigenvalue problem (MEP) which originally arises from the canonical correlation analysis is an important generalization of the classical eigenvalue problem. Recently, the MEP also finds applications in many other areas and continues to receive interest. However, the existing algorithms for the MEP are the generalization of the power iteration for the classical eigenvalue problem and converge slowly. In this paper, we propose a Riemannian Newton method for the MEP, which is a generalization of the classical Rayleigh quotient iteration (RQI). Under a mild condition, the local quadratic convergence can be guaranteed. We also develop the inexact implementation by employing some Krylov subspace method and establishing the preconditioning technique to obtain an inexact Riemannian Newton step efficiently. Preliminary but promising numerical experiments are reported which show a good convergence performance in terms of the proposed method’s speed and global convergence.

Key words. multivariate statistics, canonical correlation, multivariate eigenvalue problem, Riemannian Newton method, Rayleigh quotient iteration, precondition, Krylov subspace methods, power method, quadratic convergence, global convergence

AMS subject classifications. 62H20, 15A12, 65F10, 65K05

DOI. 10.1137/100788975

1. Introduction. Given a symmetric matrix $A \in \mathbb{R}^{n \times n}$, a vector $x \in \mathbb{R}^n$, and a positive integers set

$$\mathcal{P}_m = \{n_1, n_2, \ldots, n_m\} \quad \text{with} \quad \sum_{i=1}^m n_i = n,$$

we can partition $A$ and $x$ in the form

$$A = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1m} \\ A_{21} & A_{22} & \cdots & A_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ A_{m1} & A_{m2} & \cdots & A_{mm} \end{bmatrix} \quad \text{and} \quad x = \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix},$$

where $A_{ij} \in \mathbb{R}^{n_i \times n_j}$ and $x_i \in \mathbb{R}^{n_i}$. Let $\lambda_1, \ldots, \lambda_m$ denote $m$ real scalars and

$$\Lambda := \text{diag}\{\lambda_1 I^{[n_1]}, \lambda_2 I^{[n_2]}, \ldots, \lambda_m I^{[n_m]}\},$$

where $I^{[n_i]} \in \mathbb{R}^{n_i \times n_i}$ stands for the $n_i \times n_i$ identity matrix. The so-called multivariate eigenvalue problem (MEP) with partition $\mathcal{P}_m$ is cast as finding a pair $(\Lambda, x)$ satisfying

$$\begin{cases} \quad Ax = \Lambda x, \\ \|x_i\|_2 = 1, \quad x_i \in \mathbb{R}^{n_i}, \quad i = 1, 2, \ldots, m. \end{cases}$$

As an interesting generalization of the classical eigenvalue problem (corresponding to $m = 1$), the MEP appears in many different areas and is of mathematical interest in

∗Received by the editors March 16, 2010; accepted for publication (in revised form) by P.-A. Absil October 18, 2010; published electronically December 16, 2010.
†Department of Applied Mathematics, Shanghai University of Finance and Economics, 777 Guoding Road, Shanghai 200433, People’s Republic of China (longzh@gmail.com).

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.
RIEMANNIAN NEWTON METHOD FOR THE MEP

2973

its own right. It first arises from the maximal correlation problem (MCP) proposed by Hotelling [20, 21], which is a very early development of canonical correlation analysis (CCA). CCA is an important statistical tool for assessing the relationship between sets of variables in areas such as cluster analysis, data classification, pattern recognition, principal component analysis, and bioinformatics. Considerable research efforts have been devoted to the development of theory and techniques for CCA; see, for example, the treatises in [16, 17, 18, 19, 22, 38, 40]. The MCP for two sets of variables aims at finding the linear combination of one set of variables that correlates maximally with the linear combination of the other set of variables. If this maximal correlation can be satisfactorily established, we then can take the advantage of using one set of variables to predict the other. A concise discussion on the statistical background for m sets of variables can be found in [14], and the corresponding optimization problem is formulated as

\[
\begin{align*}
\text{Maximize} \quad & r(x) := x^\top A x \\
\text{subject to} \quad & \|x_i\|_2 = 1, \quad x_i \in \mathbb{R}^{n_i}, \quad i = 1, 2, \ldots, m.
\end{align*}
\]

Note that the constraint of (1.5)

\[
M := \{x \in \mathbb{R}^n \mid \|x_i\|_2 = 1, \quad x_i \in \mathbb{R}^{n_i}\}
\]

is simply the topology product of m unit spheres; i.e., \(M = \prod_{i=1}^{m} S^{n_i-1}\). Upon employing the Lagrange multiplier theory, it is easy to see that the first-order optimality condition for (1.5) is the existence of the Lagrange multipliers, namely, \(\lambda_1, \ldots, \lambda_m\) such that the system of equations (1.4) is satisfied. This gives the MEP. It is clear that maximizing \(r(x)\) on \(M\) is equivalent to maximizing \(x^\top (A + cI^{[n]}) x\) on \(M\) for any \(c \in \mathbb{R}\), and thereby no generality is lost when \(A\) is assumed to be positive definite. Besides serving as a necessary condition for the MCP, the MEP arises also in entirely different nonstatistical settings and continues to receive interest. For example, the MEP finds applications in the perturbation analysis of linear dynamical systems subject to additive bounded noises [42]. A special bivariate eigenvalue problem arises from identification of finite impulse response systems [25]. See also [8] for a related linear complementary problem.

We mention in passing some basic properties related with the MEP and the MCP. By observing (1.4), it is easy to see that if \((\Lambda, x)\) is a solution pair of (1.4), then \(\Lambda\) and \(x\) are related via

\[
\lambda_i = x_i^\top \mathcal{A}_i x = \|\mathcal{A}_i x\|_2, \quad i = 1, \ldots, m,
\]

where

\[
\mathcal{A}_i := [A_{i1}, A_{i2}, \ldots, A_{im}] \in \mathbb{R}^{n_i \times n}.
\]

Therefore, for a given \(x \in M\), if we denote

\[
\lambda_i(x) := x_i^\top \mathcal{A}_i x, \quad i = 1, \ldots, m, \quad \text{and}
\]

\[
\Lambda(x) := \text{diag} \left\{ \lambda_1(x)I^{[n_1]}, \ldots, \lambda_m(x)I^{[n_m]} \right\},
\]

then \(x \in M\) is a solution to the MEP if and only if the residual

\[
\delta x := Ax - \Lambda(x)x
\]
is zero, or $Ax = \Lambda(x)x$. It is interesting to note that whenever the symmetric matrix $A$ has $n$ distinct eigenvalues, then there are precisely $\prod_{i=1}^{m} (2n_i)$ solutions [14]. Among these solutions, the one associated with the MCP has the largest value of $\sum_{i=1}^{m} \lambda_i(x)$. Finding the global maximizer of the MCP (1.5) in general is a hard problem, and some efforts have been made along this line, for example, in [16, 42, 45, 46].

Similar to the classical eigenvalue problem, finding a solution to the MEP ($m > 1$) can be realized by means of iteration, but current algorithms converge slowly. The first algorithm, Algorithm 1, proposed by Horst [18] is a generalization of the classical power iteration whose recurrence structure is of Jacobi-type (we call it the Horst–Jacobi algorithm). Its convergence is established later in [14], where an improved algorithm, Algorithm 2, that takes the recurrence structure of the Gauss–Seidel-type iteration is also proposed (we call it as the Gauss–Seidel algorithm). The convergence of the improved algorithm is analyzed in [45] and an extensive numerical comparison between these two algorithms is carried out. Although the Gauss–Seidel algorithm for the MEP clearly shows the superior behavior to the Horst–Jacobi algorithm, numerical experiments seem to indicate that its convergence basically is linear. An algorithm with higher convergence rate is quite desirable.

**Algorithm 1:** The Horst–Jacobi algorithm for the MEP [18].

Given $x^{(0)} \in \mathbb{R}^n$;

for $k = 0, 1, \ldots$ do

for $i = 1, 2, \ldots, m$ do

$y_i^{(k)} := \sum_{j=1}^{m} A_{ij} x_j^{(k)}$;

$\lambda_i^{(k)} := \|y_i^{(k)}\|_2$;

$x_i^{(k+1)} := \frac{y_i^{(k)}}{\lambda_i^{(k)}}$;

end

end

**Algorithm 2:** The Gauss–Seidel algorithm for the MEP [14].

Given $x^{(0)} \in \mathbb{R}^n$;

for $k = 0, 1, \ldots$ do

for $i = 1, 2, \ldots, m$ do

$y_i^{(k)} := \sum_{j=1}^{i-1} A_{ij} x_j^{(k+1)} + \sum_{j=i}^{m} A_{ij} x_j^{(k)}$;

$\lambda_i^{(k)} := \|y_i^{(k)}\|_2$;

$x_i^{(k+1)} := \frac{y_i^{(k)}}{\lambda_i^{(k)}}$;

end

end

Viewing the MEP as a generalization of the classical eigenvalue problem, it is a natural idea to generalize some state-of-the-art eigenvalue computation methods to the MEP. This motivates us to consider the generalization of the classical Rayleigh quotient iteration (RQI). On the other hand, since the MEP serves as the KKT condition for (1.5), applying some optimization methods is another natural idea to
solve the MEP. From this point of view, in order to achieve a high convergence speed, second-order information should be used (both the Horst–Jacobi algorithm and the Gauss–Seidel algorithm are based on the first-order information). This leads us to a Newton type iteration for the MEP.

In this paper, we generalize the classical RQI to the MEP. It is known that a classical Rayleigh quotient step is equivalent to the Riemannian Newton step (we shall discuss briefly the Riemannian Newton iteration in section 2; see also [4, 5, 15, 30, 34]), and it is of cubic convergence rate for the symmetric eigenvalue problem. Our proposed algorithm for the MEP is exactly the Riemannian Newton method based on the corresponding optimization problem (1.5), and the quadratic convergence rate can be guaranteed under a mild assumption. Krylov subspace based solvers can be directly employed to obtain an inexact Riemannian Newton step, and preconditioning techniques can also be easily incorporated to improve the efficiency. Our numerical experiments show that the proposed method has a good convergence performance in terms of its speed and global convergence.

The paper is organized as follows. In section 2, some preliminaries are prepared. In section 3, we develop the complete Riemannian Newton method for the MEP and present some related properties. The local and global convergence are also addressed in this section. Section 4 is devoted to the practical implementations by discussing an inexact Riemannian Newton iteration. Some Krylov subspace methods are employed to obtain an inexact Riemannian Newton direction and special preconditioning techniques are proposed. In section 5, we report some numerical experiences to show the efficiency of our preconditioning techniques in improving the convergence behavior of the method. Finally, some concluding remarks are drawn in section 6.

2. Preliminaries. For the ease of proceeding discussion, we first review some basic geometric concepts involved in the Riemannian Newton method. For a relatively complete and comprehensive discussion, the reader is recommended to refer to [4, 6] and the many references therein.

Denote

\[
C(x) = \frac{1}{2} \begin{bmatrix}
\|x_1\|_2^2 - 1 \\
\vdots \\
\|x_m\|_2^2 - 1
\end{bmatrix},
\]

(2.1)

and then one has

\[
\nabla C(x) = \begin{bmatrix}
x_1 \\
\vdots \\
x_m
\end{bmatrix} \in \mathbb{R}^{n \times m}.
\]

(2.2)

Since for any \(x \in \mathcal{M}\), or equivalently \(C(x) = 0\), it follows that \(\text{rank}(\nabla C(x)) = m\), which implies that \(0\) is a regular value of \(C(x)\). The well-known result [1] in differential geometry then ensures that the constraint \(\mathcal{M} = C^{-1}(0)\) of the MCP is an embedded submanifold of \(\mathbb{R}^n\) with dimension \(n - m\). Moreover, the tangent space \(T_x \mathcal{M}\) to \(\mathcal{M}\) at \(x\) is \(\text{Ker}(\nabla C(x)^\top)\), which is a linear subspace of \(\mathbb{R}^n\) with the orthogonal projection onto \(\text{Ker}(\nabla C(x)^\top)\) given by

\[
P_x = I^{[n]} - \nabla C(x)(\nabla C(x)^\top \nabla C(x))^{-1} \nabla C(x)^\top
\]

(2.3)

\[
= \text{diag} \left\{ I^{[n_1]} - x_1 x_1^\top, \ldots, I^{[n_m]} - x_m x_m^\top \right\}.
\]
It is clear that $\mathcal{T}_x \mathcal{M} = \{P_x y | y \in \mathbb{R}^n\}$ and is simply the topology product of the tangent spaces of the unit spheres with dimensions $n_1 - 1, \ldots, n_m - 1$; that is, for any $N \in \mathcal{T}_x \mathcal{M}$, one has that

$$N_i^\top x_i = 0, \quad \forall i = 1, \ldots, m,$$

where $N_i \in \mathbb{R}^{n_i}$ is from the partition of $N$ according to $\mathcal{P}_m$.

Viewing $\mathcal{M}$ as a submanifold of $\mathbb{R}^n$, the natural inner product

$$\langle y, z \rangle = y^\top z, \quad \forall \ y, z \in \mathbb{R}^n$$

is induced on $\mathcal{T}_x \mathcal{M}$. Endowed with the natural inner product, $\mathcal{M}$ is then called a Riemannian submanifold of $\mathbb{R}^n$.

The constrained optimization MCP now can be viewed simply as maximizing the cost function

$$(2.4) \quad r_{|\mathcal{M}}(x) : \mathcal{M} \to \mathbb{R},$$

where the real-valued function $r_{|\mathcal{M}}(x)$ is the restriction of $r(x) = x^\top Ax$ on the Riemannian submanifold $\mathcal{M}$. The gradient of $r_{|\mathcal{M}}(x)$ at $x \in \mathcal{M}$ could be expressed as

$$(2.5) \quad g(x) := \text{grad} \ r_{|\mathcal{M}}(x) = P_x (\nabla r(x)) = 2(A - \Lambda(x))x,$$

and a stationary point $x \in \mathcal{M}$ of $r_{|\mathcal{M}}(x)$ is defined as the point satisfying $g(x) = 0$; i.e., $Ax = \Lambda(x)x$ which is exactly the MEP (1.4).

The classical Newton iteration for real-value functions uses the information of Hessian. For the cost function $r_{|\mathcal{M}}(x)$, on the other hand, defining the (symmetric) Hessian operator at $x \in \mathcal{M}$

$$\text{Hess} \ r_{|\mathcal{M}}(x) : \mathcal{T}_x \mathcal{M} \to \mathcal{T}_x \mathcal{M} : N \mapsto \nabla N \text{grad} \ r_{|\mathcal{M}}(x)$$

involves the so-called affine connection $\nabla$ (see [4]). The Riemannian connection (or Levi–Civita connection) is obviously a natural affine connection as it possesses distinctive properties (see section 5.5 of [4]). It also significantly simplifies our computations since the action of Hessian of $r_{|\mathcal{M}}(x)$ on a tangent vector $N \in \mathcal{T}_x \mathcal{M}$ can be expressed by

$$(2.6) \quad \text{Hess} \ r_{|\mathcal{M}}(x)[N] = P_x (Dg(x)[N]) = P_x (Dg(x)[N]), \quad \forall N \in \mathcal{T}_x \mathcal{M},$$

where $Dg(x)[N]$ stands for the derivative at $x \in \mathcal{M}$ along $N$ of the function that maps $x$ to $g(x)$. By a simple manipulation, it is not difficult to find that

$$(2.7) \quad Dg(x)[N] = 2 \begin{bmatrix} \mathcal{A}_1 N - N_1 x_1^\top \mathcal{A}_1 x - x_1 N_1^\top \mathcal{A}_1 x - x_1 x_1^\top \mathcal{A}_1 N \\ \vdots \\ \mathcal{A}_m N - N_m x_m^\top \mathcal{A}_m x - x_m N_m^\top \mathcal{A}_m x - x_m x_m^\top \mathcal{A}_m N \end{bmatrix}.$$ 

Thus, we have

$$\text{Hess} \ r_{|\mathcal{M}}(x)[N] = P_x (Dg(x)[N]) = 2 \begin{bmatrix} \mathcal{A}_1 N - N_1 x_1^\top \mathcal{A}_1 x - x_1 x_1^\top \mathcal{A}_1 N \\ \vdots \\ \mathcal{A}_m N - N_m x_m^\top \mathcal{A}_m x - x_m x_m^\top \mathcal{A}_m N \end{bmatrix},$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.
where the last equality follows from the fact that $x_i^T N_i = 0$ for $i = 1, 2, \ldots, m$. Note that (2.8) can be further simplified into the following matrix form:

$$\text{Hess } r_{|_{\mathcal{M}}}(x)[N] = 2(P_x A - \Lambda(x))N, \quad \forall N \in T_x \mathcal{M}.$$ 

As $N \in T_x \mathcal{M}$, it follows that $P_x N = N$, and moreover, by noting that $P_x^2 = P_x$, we finally get

$$\text{Hess } r_{|_{\mathcal{M}}}(x)[N] = 2P_x(A - \Lambda(x))P_x N, \quad \forall N \in T_x \mathcal{M}. \tag{2.8}$$

This expression is important for it explicitly presents the action of (symmetric) Hessian on a tangent vector $N \in T_x \mathcal{M}$ as a matrix-vector product $2P_x(A - \Lambda(x))P_x N$. Related with the Hessian, if $x$ is a local maximizer of $r_{|_{\mathcal{M}}}(x)$, then the matrix $P_x(A - \Lambda(x))P_x$ restricted to the tangent space is negative semidefinite, whereas if $g(x) = 0$ and

$$P_x(A - \Lambda(x))P_x : T_x \mathcal{M} \to T_x \mathcal{M}$$

is negative definite, then $x$ is a local maximizer [45].

The Newton direction $N \in T_x \mathcal{M}$ at $x \in \mathcal{M}$ of $r_{|_{\mathcal{M}}}(x)$ is just the solution to the following system:

$$\text{Hess } r_{|_{\mathcal{M}}}(x)[N] = -g(x), \quad N \in T_x \mathcal{M},$$

or equivalently,

$$P_x(A - \Lambda(x))P_x N = -\frac{g(x)}{2}, \quad N \in T_x \mathcal{M}. \tag{2.9}$$

We will discuss the solution of (2.9) in more detail in sections 3 and 4.

The last geometric concept to prepare for the Riemannian Newton method is the so-called retraction (see section 4.1 of [4]). For a given general manifold $\mathcal{M}$, a retraction, namely $R$, is defined as a smooth mapping from the tangent bundle

$$T \mathcal{M} := \bigcup_{x \in \mathcal{M}} T_x \mathcal{M}$$

onto $\mathcal{M}$ that satisfies the following properties: Let $0_x$ stand for the zero element in $T_x \mathcal{M}$ and let $R_x$ denote the restriction of $R$ to $T_x \mathcal{M}$; then

$$R_x(0_x) = x, \quad \text{and } \frac{d}{dt} R_x(tz)|_{t=0} = z, \quad \forall z \in T_x \mathcal{M}. \tag{2.10}$$

For our discussed Riemannian manifold $\mathcal{M} = C^{-1}(0)$, the projection $\Pi_x : T_x \mathcal{M} \to \mathcal{M}$ at $x \in \mathcal{M}$ onto $\mathcal{M}$ defined by

$$\Pi_x(y) := \begin{bmatrix} x_1 + y_1 \|x_1 + y_1\|_2 \\ \vdots \\ x_m + y_m \|x_m + y_m\|_2 \\ \end{bmatrix} \in \mathcal{M}, \quad \forall y \in T_x \mathcal{M} \tag{2.11}$$

is a simple and computable retraction. For a given vector $y \in T_x \mathcal{M}$, $\Pi_x(y)$ returns the nearest point onto $\mathcal{M}$ to $x + y$; moreover, it is proved that the projection to a Riemannian manifold is a second-order retraction (see [4, 6]), which means that in addition to the properties (2.10), the following condition holds:

$$P_x \frac{d^2}{dt^2} R_x(tz)|_{t=0} = 0, \quad \forall z \in T_x \mathcal{M}.$$
3. Riemannian Newton method for the MEP.

3.1. Riemannian Newton iteration for the MEP. With the preparations of section 2, we can state a general Riemannian Newton iteration for finding a stationary point of a real-valued function \( f_{|M}(x) \) defined on a general Riemannian manifold \( M \) [4, 7]. Let \( x \in M \) be the current point, and \( R \) be a retraction on \( M \); then the next iteration \( x_+ \) can be computed as

\[
\begin{align*}
\text{Hess } f_{|M}(x)[N] &= -\text{grad } f_{|M}(x), \quad N \in T_x M, \\
x_+ &= R_x(N).
\end{align*}
\]

In particular, by taking the special structure of the manifold \( M = C^{-1}(0) \), the real-valued function \( f(x) = r(x) \), and the retraction \( R_x(N) = \Pi_x(N) \) given by (2.11), the Riemannian Newton method for the MEP proceeds as Algorithm 3.

**Algorithm 3:** Riemannian Newton method for the MEP.

1. Choose the tolerance \( \epsilon > 0 \) and an \( x^{(0)} \in M \); set \( k = 0 \).
2. Solve the Newton equation

\[
P_{x^{(k)}}(A - \Lambda(x^{(k)}))P_{x^{(k)}}N = -(A - \Lambda(x^{(k)}))x^{(k)}
\]

for the unknown Newton direction \( N \in T_{x^{(k)}} M \), where \( P_{x^{(k)}} \) is given by (2.3).
3. Set \( x^{(k+1)} = \Pi_{x^{(k)}}(N) \), where \( \Pi_{x^{(k)}}(N) \) is computed by (2.11).
4. Stop if

\[
\|\delta x^{(k+1)}\|_2 = \|Ax^{(k+1)} - \Lambda(x^{(k+1)})x^{(k+1)}\|_2 < \epsilon;
\]

otherwise, set \( k = k + 1 \) and goto Step 2.

The key step of Algorithm 3 is to solve the Newton direction \( N \). To simplify the following presentation, superscripts and subscripts are dropped when the meaning is clear from the context; hence \( x^{(k)} \) is denoted by \( x \) and \( x^{(k+1)} \) by \( x_+ \). It has already been known that if \( m = 1 \), the Riemannian Newton iteration is equivalent to the classical RQI (see, e.g., [4, 15, 30, 34]). In this case, indeed, if \( A - r(x)[I]^{(n)} \) is nonsingular, then from \( P_x = I^{[n]} - xx^\top \) and \( P_x N = N \), the Newton equation (2.9) leads to

\[
N = (A - r(x)[I]^{[n]})^{-1}x d - (A - r(x)[I]^{[n]})^{-1}g(x) = (A - r(x)[I]^{[n]})^{-1}x d - x,
\]

where \( d \in \mathbb{R} \) is a scalar to ensure \( N^\top x = 0 \), and consequently it gives rise to

\[
N = -x + \frac{(A - r(x)[I]^{[n]})^{-1}x}{x^\top (A - r(x)[I]^{[n]})^{-1}x}.
\]

This derivation is based on the technique of Olsen, Jørgensen, and Simons [27]; see also [35, 36]. Interestingly, for \( m > 1 \), the technique of [27] can be extended to give a closed-form for the Newton direction defined by (2.9). Assuming \( A - \Lambda(x) \) is nonsingular (this assumption could be relaxed as we shall see later), by \( P_x N = N \) and the fact that \( N \in T_x M \) if and only if

\[
N_i^\top x_i = 0, \quad i = 1, 2, \ldots, m,
\]
one has from (2.9) that
\[
N = -x + (A - \Lambda(x))^{-1}Dx,
\]
where
\[
D = \text{diag}\{d_1 I^{[n_1]}, \ldots, d_m I^{[n_m]}\}
\]
and \(d_1, \ldots, d_m\) are real scalars to ensure \(N \in T_x M\). Denoting \(d = [d_1, \ldots, d_m]^T \in \mathbb{R}^m\), we next describe the procedure to obtain the vector \(d\).

Denote \((A - \Lambda(x))^{-1}\) by \(B\). According to the set \(P_m(1.1)\), the matrix \(B\) can also be partitioned into block form \(B = [B_{ij}]\), where \(B_{ij} \in \mathbb{R}^{n_i \times n_j}\).

Since \(N_i = -x_i + \sum_{j=1}^m B_{ij}x_jd_j, \quad i = 1, 2, \ldots, m\), and \(\|x_i\|_2 = 1\) for \(i = 1, 2, \ldots, m\), the condition \(N_i^T x_i = 0\) leads to
\[
\sum_{j=1}^m x_i^T B_{ij}x_jd_j = 1, \quad i = 1, 2, \ldots, m.
\]

Let \(S \in \mathbb{R}^{m \times m}\), where \(S_{ij} := x_i^T B_{ij}x_j\). Then we need to solve for \(d\) from the system
\[
\sum_{j=1}^m S_{ij}d_j = 1, \quad i = 1, 2, \ldots, m,
\]
which can be further formulated as
\[
Sd = e := [1, \ldots, 1]^T \in \mathbb{R}^m.
\]

The equation (3.4) provides a closed-form for \(d\), and if \(S\) is nonsingular, then \(d = S^{-1}e\).

We point out that the closed-form of the Newton direction can be derived from an augmented system of (2.9). Similar to the case \(m = 1\) (see, e.g., [4, 30]), the system (2.9) can be equivalently transformed to a saddle-point problem (3.5).

**Theorem 3.1.** Let \(x \in M\). Then any solution of the saddle-point problem
\[
G_x \begin{bmatrix} N \\ w \end{bmatrix} := \begin{bmatrix}
-A - \Lambda(x) & -x_1 \\
-x_1 & \ddots & \ddots \\
& \ddots & -x_m \\
-x_1 & \ddots & 0
\end{bmatrix} \begin{bmatrix} -x_1 \\ \vdots \\ -x_m \\ 0
\end{bmatrix}
\]

\[
\begin{bmatrix} N \\ w \end{bmatrix} = \begin{bmatrix} (A - \Lambda(x))^T x \\ 0
\end{bmatrix}
\]
is of the form
\[
[N^T, w^T]^T = [N^T, x_1 \otimes_1 N, \ldots, x_m \otimes_m N]^T \in \mathbb{R}^{n+m}.
\]
Moreover, $N$ is a solution to (2.9) if and only if $[N^T, x^+_1 \mathcal{A}_1 N, \ldots, x^+_m \mathcal{A}_m N]^T$ is a solution to (3.5).

Proof. First, we show the structure of the solution of (3.5). Suppose $[N^T, w^T]^T$ is an arbitrary solution of (3.5), then we have

\[
\begin{align*}
(A - \Lambda(x))N - \begin{bmatrix} x_1 & \cdots & x_m \end{bmatrix} w &= -(A - \Lambda(x))x, \\
\quad x^+_i N_i &= 0, \quad \forall i = 1, 2, \ldots, m.
\end{align*}
\]

The second relation implies $N \in T_x \mathcal{M}$. By premultiplying $\text{diag} \{x^+_1, \ldots, x^+_m\}$ to both sides of the first equation in (3.6) and noting $\|x_i\|_2 = 1$ ($i = 1, \ldots, m$), and $\lambda_i(x) = x^+_i \mathcal{A}_i x$, we have

\[
w_i = x^+_i \mathcal{A}_i N, \quad i = 1, \ldots, m.
\]

Now, suppose $N$ satisfies (2.9), then one has $x^+_i N_i = 0$ ($i = 1, 2, \ldots, m$), $P_x N = N$, and

\[
-(A - \Lambda(x))x = P_x(A - \Lambda(x))P_x N = P_x(A - \Lambda(x))N
\]

\[
= (A - \Lambda(x))N - \begin{bmatrix} x_1 & \cdots & x_m \end{bmatrix} \begin{bmatrix} x^+_1 \mathcal{A}_1 N \\ \vdots \\ x^+_m \mathcal{A}_m N \end{bmatrix}
\]

which leads to the saddle-point system (3.5).

Conversely, if $[N^T, x^+_1 \mathcal{A}_1 N, \ldots, x^+_m \mathcal{A}_m N]^T$ is a solution for (3.5), then we have

$N \in T_x \mathcal{M}$ and

\[
(A - \Lambda(x))N - \begin{bmatrix} x_1 & \cdots & x_m \end{bmatrix} \begin{bmatrix} x^+_1 \mathcal{A}_1 N \\ \vdots \\ x^+_m \mathcal{A}_m N \end{bmatrix} = -(A - \Lambda(x))x.
\]

Premultiplying $P_x$ on both sides and noting $P_x N = N$, $P_x(A - \Lambda(x))x = (A - \Lambda(x))x$, and $(F^{[x]} - x_i x_i^T)x_i = 0$, (2.9) follows. \hfill \square

Theorem 3.1 also leads to the following straightforward properties:

**Corollary 3.2.** Let $x \in \mathcal{M}$. Then

(i) the Newton direction $N$ of (2.9) is unique if and only if the saddle-point matrix $G_x$ given by (3.5) is nonsingular;

(ii) the operator

\[
P_x(A - \Lambda(x)) : T_x \mathcal{M} \to T_x \mathcal{M}
\]

is invertible if and only if $G_x$ is nonsingular.

From Corollary 3.2, it is clear that the uniqueness of the Newton direction of (2.9) does not necessarily require the nonsingularity of $A - \Lambda(x)$; however, if $G_x$ and $A - \Lambda(x)$ are both nonsingular, we can express the Newton direction explicitly from (3.5). In fact, under these assumptions, the solution of (3.5) is

\[
\begin{bmatrix} N \\ w \end{bmatrix} = -G_x^{-1} \begin{bmatrix} (A - \Lambda(x))x \\ 0 \end{bmatrix}.
\]
Note that $G_x^{-1}$ is a 2-by-2 block matrix and its inverse can be expressed by the Bakerchiewicz inversion formula ([44, pp. 11]). As a result, by denoting $B = (A - \Lambda(x))^{-1}$, the (1,1) block of $G_x^{-1}$ is

$$B - B[x_1 \ldots x_m] \left( \begin{bmatrix} x_1^T & \ldots & x_m^T \end{bmatrix} B \begin{bmatrix} x_1 \ldots x_m \end{bmatrix} \right)^{-1} \begin{bmatrix} x_1^T \ldots x_m^T \end{bmatrix} B;$$

consequently, we have

$$\Sigma = -x + B[x_1 \ldots x_m] \left( \begin{bmatrix} x_1^T & \ldots & x_m^T \end{bmatrix} B \begin{bmatrix} x_1 \ldots x_m \end{bmatrix} \right)^{-1} \begin{bmatrix} x_1^T \ldots x_m^T \end{bmatrix} x$$

where

$$S = S^{-1} e$$

(3.9)

$$N = -x + B[x_1 \ldots x_m] \left( \begin{bmatrix} x_1^T & \ldots & x_m^T \end{bmatrix} B \begin{bmatrix} x_1 \ldots x_m \end{bmatrix} \right)^{-1} \begin{bmatrix} x_1^T \ldots x_m^T \end{bmatrix} d$$

(3.9)

where $d := S^{-1} e$

$$= -x + (A - \Lambda(x))^{-1} D x$$

(3.9)

which is precisely (3.3). Moreover, from this relation and the first equation of (3.5), we know that $d = w$.

A computational scheme based on (3.9) involves solving a series of linear systems. First, to obtain the matrix $S$, $m$ linear systems should be addressed instead of directly forming $(A - \Lambda(x))^{-1}$. Since

$$S_{ij} = x^T_i B_{ij} x_j = [0, \ldots, x_i^T, \ldots, 0] B [0, \ldots, x_j^T, \ldots, 0]^T,$$

a linear system

$$(A - \Lambda(x)) v = [0, \ldots, x_j^T, \ldots, 0]^T$$

must be solved, where $v = B [0, \ldots, x_j^T, \ldots, 0]^T$; therefore, $S_{ij} = x^T_i v$, where $v_i \in \mathbb{R}^n$: is a subvector of $v$ corresponding to the set $P_m$. Note that these $m$ linear systems have the same coefficient matrix and hence they form a system with multiple right-hand sides. Another smaller linear system $Sd = e$ is to generate the vector $d = [d_1, \ldots, d_m]^T$, and the last step is to obtain $\hat{x} := (A - \Lambda(x))^{-1} D x$ by solving the linear system

$$(A - \Lambda(x)) \hat{x} = [d_1 x_1^T, \ldots, d_m x_m^T]^T.$$

These steps finally yield an algorithm for the MEP and the pseudocode is presented in Algorithm 4. Because this algorithm reduces to the classical RQI when $m = 1$, we also call it the RQI for the MEP (RQI-MEP).

We next briefly discuss the computational complexity of the RQI-MEP. Note that for a single outer iteration in Algorithm 4, there are totally $m + 1$ linear systems of order $n$ involved in (A) and (C). Since these $m + 1$ systems all have the coefficient
matrix $A - \Lambda(x^{(k)})$, an LU decomposition could then be used for solving these $m+1$ systems, and therefore, the flops used in (A) and (C) are $\frac{2m^3}{3} + 2n^2(m+1)$. Moreover, the $m$ linear systems in (A) could also be implemented in parallel, and if a good preconditioner for $A - \Lambda(x^{(k)})$ can be designed, the computation for (A) can be more efficient. An additional linear system in (B) in Algorithm 4 is of order $m$ and can be solved with flops $\frac{2m^3}{3} + 2m^2$. Consequently, we know that an outer iteration roughly needs flops

$$\frac{2n^3}{3} + 2n^2(m + 1) + \frac{2m^3}{3} + 2m^2.$$

We notice that the dominant order of flops is the same as solving a single linear system of order $n$ and therefore, for a given $m > 1$, the RQI-MEP (Algorithm 4) roughly has the same computational complexity as the classical RQI for the standard eigenvalue problem.

We finish this subsection by briefly discussing the situation when the operator (3.8) is singular. When this happens, $G_x$ is singular and the uniqueness of the Newton direction defined by (2.9) is not guaranteed. Fortunately, from a computational point of view, this is not a tragedy because a Krylov subspace method, for example, the minimal residual method (MINRES) which is able to handle the singular linear system, can be employed to solve the linear systems involved in Algorithm 4; moreover, as we shall see in section 4, a Krylov subspace method can also directly be used to solve (2.9). Another possibility is to dismiss the singularity by slightly perturbing $\Lambda(x)$ by a shift $\Delta \Lambda = \text{diag}\{\Delta \lambda_1I^{[n_1]}, \ldots, \Delta \lambda_mI^{[n_m]}\}$. Discussions for $m = 1$ can be found, for example, in [30].

### 3.2. Local convergence

Recall that for solving nonlinear equations, the classical Newton method is of local quadratic convergence rate if the related Jacobi is nonsingular. This result has been extended to the Riemannian Newton method (see [4, Theorem 6.3.2, p. 114]). In particular, to apply this local convergence result
to Algorithm 3, we require the assumption that the Hessian operator (3.8) is invertible at a solution \( x^* \in \mathcal{M} \) of the MEP (1.4). Note that Corollary 3.2 gives an equivalent condition for the invertibility of the Hessian, and as a result, we have the following.

**Theorem 3.3.** Let \( x^* \in \mathcal{M} \) be a solution to the MEP (1.4). If the matrix \( G_{x^*} \), defined by (3.5) is nonsingular, then there exists a neighborhood \( \mathcal{U} \) of \( x^* \) in \( \mathcal{M} \) such that, for all \( x^{(0)} \in \mathcal{U} \), Algorithm 3 (or Algorithm 4) generates an infinite sequence \( \{x^{(k)}\} \) converging at least quadratically to \( x^* \).

We make some remarks for Theorem 3.3. First, it deserves to be pointed out that the local quadratic convergence in Theorem 3.3 is only based on the result of the Riemannian Newton method for a general problem. On the other hand, for some special eigenvalue problems, the convergence of the Riemannian Newton iteration is always cubic. For example, when applying the Riemannian Newton to the Grassmann manifold for solving the invariant subspace of \( A \), Absil, Mahony, and Sepulchre [3] show that the method converges cubically if and only if the targeted invariant subspace is a left and right spectral invariant subspace of \( A \); other applications of the Riemannian Newton where the cubic convergence takes place can be found, e.g., in [2, 4, 5, 15]. Moreover, when \( m = 1 \), Algorithm 4 is equivalent to the classical RQI for the symmetric eigenvalue problem and converges cubically to a simple eigenvalue. This seems to imply that a higher convergence rate may exist in Algorithm 4. Unfortunately, our preliminary numerical experiments give no clue for the cubic convergence in general. Second, when \( m = 1 \), an equivalent statement for the nonsingularity of \( G_{x^*} \) is that the eigenvalue \( (x^*)^\top Ax^* \) corresponding to \( x^* \) is a simple eigenvalue of \( A \); see, e.g., [4, 30]. For \( m > 1 \), we do not have an equivalent algebraic statement thus far. Nevertheless, in the following special cases, Theorem 3.3 can be applied to the global maximizer of the MCP.

**Theorem 3.4.** Let \( x^* \) be a global maximizer of the MCP (1.5). If either of the conditions

(a) \( A \) is nonnegative irreducible,
(b) \( m = 2 \) and zero is a simple eigenvalue of \( A - \Lambda(x^*) \),

is satisfied, then there exists a neighborhood \( \mathcal{U} \) of \( x^* \) in \( \mathcal{M} \) such that, for all \( x^{(0)} \in \mathcal{U} \), Algorithm 3 (or Algorithm 4) generates an infinite sequence \( \{x^{(k)}\} \) converging at least quadratically to \( x^* \).

**Proof.** According to Theorem 3.3, it suffices to proves that the corresponding matrix \( G_{x^*} \) is nonsingular. To this end, we suppose there is a vector \( [f^\top, h^\top]^\top \in \mathbb{R}^{n+m} \) such that \( G_{x^*}[f^\top, h^\top]^\top = 0 \). Thus

\[
(A - \Lambda(x^*))f = \begin{bmatrix} x^*_1 & \cdots & x^*_m \end{bmatrix} h \quad \text{and} \quad f \in T_{x^*} \mathcal{M}.
\]

Therefore, one has \( f^\top (A - \Lambda(x^*))f = 0 \). On the other hand, whenever the condition (a) or (b) is true, then the facts that \( A - \Lambda(x^*) \) is negative semidefinite and the largest eigenvalue \( \sigma_1(A - \Lambda(x^*)) = 0 \) of \( A - \Lambda(x^*) \) is a simple eigenvalue have been proved in [46] and [16], respectively. This implies that \( f \) is parallel to \( x^* \) but the conclusion \( f \in T_{x^*} \mathcal{M} \) leads to \( f = 0 \) and consequently \( h = 0 \). This proves that \( G_{x^*} \) is nonsingular and Theorem 3.3 applies. \( \square \)

### 3.3. Global convergence

For the global convergence, Algorithm 4 with \( m = 1 \) (the RQI) has been well understood [10, 28, 29]: (i) the generated sequence \( \{x^{(k)}\} \) either converges to an eigenvector of \( A \) or to the bisectors of a pair of eigenvectors...
Proposition 3.5. Given \( x \neq 0 \), then for any scalar \( \mu \),
\[
\|(A - \mu I^n)x\|_2 \geq \|(A - r(x)I^n)x\|_2
\]
with equality only when \( \mu = r(x) \). That is,
\[
r(x) = \arg\min_\mu \|(A - \mu I^n)x\|_2.
\]

Interestingly, a similar minimal residual property holds for the MEP as we shall see below.

Proposition 3.6. Given a positive integers set \( P_m \) (1.1) and \( x \neq 0 \), then for any \( \Lambda \) defined by (1.3)
\[
\|(A - \Lambda)x\|_2 \geq \|(A - \Lambda(x))x\|_2
\]
with equality only when \( \Lambda = \Lambda(x) \). That is,
\[
\Lambda(x) = \arg\min_\Lambda \|(A - \Lambda)x\|_2.
\]

Proof. Note that
\[
\|(A - \Lambda)x\|_2^2 = \|(A - \Lambda(x))x + \Delta \Lambda x\|_2^2
\]
\[
= \|(A - \Lambda(x))x\|_2^2 + \|
\Delta \Lambda x\|_2^2 + 2x^\top \Delta \Lambda (A - \Lambda(x))x,
\]
where \( \Delta \Lambda := \Lambda(x) - \Lambda \). Based on the definition of \( \Lambda(x) \) in (1.8) and (1.9), it is easy to see that \( x^\top \Delta \Lambda (A - \Lambda(x))x = 0 \) and hence
\[
\|(A - \Lambda)x\|_2^2 \geq \|(A - \Lambda(x))x\|_2^2 + \|
\Delta \Lambda x\|_2^2 \geq \|(A - \Lambda(x))x\|_2^2
\]
with equality only when \( \Lambda = \Lambda(x) \). \( \Box \)

The minimal residual property Proposition 3.6 motivates us to investigate the monotone decrease of the residual \( \{\|
\delta x^{(k)}\|_2\} \). Unfortunately, we observed that the monotone decrease property of the residual for \( m = 1 \) could not be generalized to the case \( m > 1 \). A numerical example is presented as an illustration.

Example 1. The matrix \( A \) is given by
\[
A = \begin{bmatrix}
4.3229 & 2.3230 & -1.3711 & -0.0084 & -0.7414 \\
2.3230 & 3.1181 & 1.0959 & 0.1285 & 0.0727 \\
-1.3711 & 1.0959 & 6.4920 & -1.9883 & -0.1878 \\
-0.0084 & 0.1285 & -1.9883 & 2.4591 & 1.8463 \\
-0.7414 & 0.0727 & -0.1878 & 1.8463 & 5.8875
\end{bmatrix} \in \mathbb{R}^{5 \times 5}
\]

with \( m = 2 \) and \( P_m = \{2, 3\} \). This example is from [14].

We apply Algorithm 4, the Gauss–Seidel algorithm, and the Horst–Jacobi algorithm to solve this problem. Starting with \( x^{(0)} = [0.9413, 0.3376, -0.9636, -0.2340, 0.1295]^\top \) and terminating whenever \( \|
\delta x^{(k)}\|_2 < \epsilon = 10^{-6} \) or \( k > 500 \), the
residuals of these algorithms are recorded and the sequence \(\{\log_{10} \|\delta x^{(k)}\|_2\}\) is plotted in the left subfigure of Figure 3.1. All algorithms converge to the global maximizer and the RQI-MEP algorithm converges very fast. We observed that the related matrix \(G_{x^*}\) is nonsingular and the fast convergence of RQI-MEP is an illustration of the local quadratic convergence. However, the monotone decrease in \(\{\|\delta x^{(k)}\|_2\}\) does not appear.\(^1\)

Besides the residual, the monotone increase of the objective function value is also a possible term to guarantee the global convergence. In fact, the proof of the global convergence of either the Horst–Jacobi algorithm [14] or the Gauss–Seidel algorithm [45] exploits this property. Unfortunately again, the monotone increase of \(\{r(x^{(k)})\}\) was not observed in Example 1. See the right subfigure in Figure 3.1.

Nevertheless, though now we are not able to ensure the global convergence theoretically, an inexact version of Riemannian Newton method Algorithm 3 (see details in subsection 4.1) in which the Newton equation (3.1) is solved inexactly by MINRES (with maximal number of iterations 20 and tolerance \(10^{-6}\) as the stopping criterion for MINRES) shows a good performance in terms of the global convergence. In Example 1, we randomly generate \(10^5\) starting points. The algorithm converges from all these starting points with the average number of outer loop iterations 5.78. To extend our testing of the global convergence, we test random MEPs in Example 2.

**Example 2.** This example contains many MEPs. For a given pair \(n \text{ and } m\), we first randomly generate the testing matrices \(A\)'s and the positive integers sets \(P_m\)'s for \(10^4\) times. Therefore, for a given pair \(n \text{ and } m\), we have \(10^4\) testing problems.

For each problem, we then run the algorithm starting from a randomly generated starting point and terminate the iteration whenever \(\|\delta x^{(k)}\|_2 < \epsilon = 10^{-6}\) or \(k > 500\). We last count the cases for which the algorithm converges successfully to a solution to the MEP within the stopping criterion. For each given pair \(n \text{ and } m\),

\(^1\)To avoid the effect of the rounding errors, we employed three different ways to implement the Riemannian Newton method for Example 1. The result shown in the left of Figure 3.1 is from Algorithm 4, where the LU decomposition is used to solve the linear systems involved. We also tried the Krylov subspace solver, MINRES, to solve these systems with maximum number of iterations 20 and tolerance \(10^{-8}\) as the stopping criterion. Moreover, a computation based on Algorithm 3, where the Newton equation (3.1) is solved by MINRES (with the same stopping criterion), is also implemented. All computations yield the same curve as that in the left of Figure 3.1.
the percent of convergent cases as well as the average number of outer iterations (over $10^4$ testing MEPs) are summarized in Table 3.1. More numerical testing on the global convergence, with some ill-conditioned matrices $A$’s, will also be carried out in section 4.

### Table 3.1

Performance of the global convergence of an inexact Riemannian Newton iteration on Example 2.

<table>
<thead>
<tr>
<th>$n$</th>
<th>10</th>
<th>10</th>
<th>50</th>
<th>50</th>
<th>100</th>
<th>100</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m$</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>10</td>
<td>20</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td># of convergent cases</td>
<td>$10^4$ (%)</td>
<td>98.31</td>
<td>98.76</td>
<td>99.11</td>
<td>99.78</td>
<td>99.78</td>
<td>99.12</td>
</tr>
<tr>
<td>Avg. Iter. #</td>
<td>17.88</td>
<td>15.92</td>
<td>34.03</td>
<td>36.05</td>
<td>64.71</td>
<td>46.16</td>
<td>61.31</td>
</tr>
</tbody>
</table>

#### 3.4. Further discussion.

In this subsection, we discuss some strategies to improve the convergence of the Riemannian Newton iteration for solving the MEP (1.4). From the optimization point of view, several techniques developed on the Euclidean space have been extended to Riemannian manifolds, and they can be borrowed to solve the MCP (1.5) globally. Most of these techniques fall into two categories: line search methods and trust-region methods. In [43], the Armijo-line search is extended to general smooth Riemannian manifolds (see also [4]), and an Armijo–Newton algorithm on Riemannian manifolds is proved to converge globally under certain conditions (see [43, Theorem 4.3]). The trust-region method, on the other hand, can be regarded as an enhancement of the Newton method, and [2, 4] have made substantial efforts on developing the trust-region method on Riemannian manifolds. We find that the MCP (1.5) particularly fits within the framework of the trust-region method of [2, 4], since the trust-region subproblem at the current point $x(k)$ ∈ $\mathcal{M}$ can be simply formulated as

\[
\begin{align*}
\max_{y \in T_{x(k)} \mathcal{M}, \|y\|_2 \leq \Delta_k} & \left\{ r(x(k)) + 2y^T(A - \Lambda(x(k)))x(k) + y^TP_{x(k)}(A - \Lambda(x(k)))P_{x(k)}y \right\},
\end{align*}
\]

where $\Delta_k > 0$ is the trust-region radius that will be updated according to the quality of the model (3.10) (see Algorithm 1 in [2] for the details). While the subproblem (3.10) can be solved by many classical methods for the trust-region subproblem [26], the truncated conjugate-gradient method of Steihaug [37] and Toint [39] is particularly appropriate in this case. Once the solution $y(k)$ of (3.10) is at hand, the next iterate $x(k+1) \in \mathcal{M}$ then is either $x(k+1) = \Pi_{x(k)}(y(k))$ or $x(k+1) = x(k)$, depending on the quality of the model (3.10). The global convergence (see [2, Theorem 4.4]) and the order of convergence (see [2, Theorems 4.13 and 4.14]) have been well established under appropriate assumptions. For the MCP (1.5), because the cost function $r|_{\mathcal{M}}(x)$ and the retraction (2.11) are smooth, and the manifold $\mathcal{M}$ is compact, these assumptions are satisfied (see [2, Theorem 4.4 and Corollary 4.6]), which therefore guarantees the global convergence.

Another useful idea to globalize the Riemannian Newton iteration is to generalize the classical Jacobi–Davidson iteration [35, 41] to the MEP (1.4). The Jacobi–Davidson approach, as well as the Arnoldi method (or the Lanczos method for the symmetric eigenvalue problem), belong to the Davidson-type subspace methods [35, 41] in which the principal is to sequentially project the original large problem.
to small eigenvalue problems via a sequence of properly generated search subspaces \( \{ \mathcal{V}(k) \} \). The Arnoldi method differs from the Jacobi–Davidson approach in that the former employs the Arnoldi’s procedure (i.e., the subspace \( \mathcal{V}(k) \) is a Krylov subspace) to expand the subspace \( \mathcal{V}(k) \), while the latter uses the solution of the Jacobi–Davidson correction equation to expand \( \mathcal{V}(k) \). It is known that (see [4, section 6.4.5]) the Jacobi–Davidson correction equation is exactly the Riemannian Newton equation for the Rayleigh quotient on the sphere; the Jacobi–Davidson approach, therefore, can be viewed as a Newton method with subspace acceleration. In [42], the Arnoldi method for the standard eigenvalue problem has been generalized to solve the MEP (1.4), where the iterate from the Horst–Jacobi method is used to expand the related search subspaces. Based on the framework of the Davidson-type subspace methods and the generalization of the Arnoldi method [42], it is not difficult to generalize the classical Jacobi–Davidson iteration to the MEP, as we can employ the Riemannian Newton equation (3.1) to expand the related search subspaces. The issue of the detailed implementation and comparison of these globalization techniques in practice is outside the scope of this paper and will be part of our future work.

4. Inexact Riemannian Newton iteration for the MEP. As an alternative to the direct method Algorithm 4, in this section, we will discuss inexact implementations to realize the Riemannian Newton iteration (Algorithm 3). We will employ some Krylov subspace method to solve the Newton equation (3.1) and also consider some special preconditioning technique for (3.1). These techniques could make the computation more efficient.

4.1. Inexact Riemannian Newton step. Recall that the generalized minimal residual (GMRES) [33] algorithm is a widely used projection method for solving the general linear system

\[
FF = h, \quad F \in \mathbb{R}^{n \times n}.
\] (4.1)

GMRES belongs to the class of Petrov–Galerkin methods, and MINRES is its variant which is mathematically equivalent to GMRES but handles the symmetric and indefinite linear system in an efficient way (see [32, 41]). Starting from an initial guess \( f^{(0)} \), the \( l \)-th iteration of a projection method finds an approximation \( f^{(l)} \) of (4.1) satisfying the Petrov–Galerkin conditions

\[
f^{(l)} \in f^{(0)} + \mathcal{K}_l \quad \text{such that} \quad h - FF^{(l)} \perp \mathcal{L}_l,
\] (4.2)

where \( \mathcal{K}_l \) and \( \mathcal{L}_l \) are two \( l \)-dimensional subspaces of \( \mathbb{R}^n \). In GMRES, \( \mathcal{L}_l = FK_l \) and \( \mathcal{K}_l \) is chosen as the \( l \)-th Krylov subspace

\[
\mathcal{K}_l = \mathcal{K}_l(F, s^{(0)}) := \text{span}\{s^{(0)}, Fs^{(0)}, \ldots, F^{l-1}s^{(0)}\},
\]

where \( s^{(0)} := h - FF^{(0)} \) is the residual of the linear system (4.1). Equivalently, \( f^{(l)} \) is defined by

\[
f^{(l)} := \arg\min_{f \in f^{(0)} + \mathcal{K}_l(F, s^{(0)})} \|h - FF\|_2.
\] (4.3)

Noting that (2.9) defines a solution \( N \) of the linear system

\[
H_N = -\frac{g(x)}{2}, \quad \text{where} \quad H_N = P_N(A - \Lambda(x))P_N,
\] (4.4)
in the tangent space $T_xM$, we know that for any $y \in \mathbb{R}^n$, if we take $N^{(0)} = P_x y \in T_xM$ as the initial guess, then the residual $s^{(0)}$ of the linear system (4.4) satisfies
\[ s^{(0)} = H_x N^{(0)} + \frac{g(x)}{2}, \quad \text{and} \quad H_x s^{(0)} \in T_xM, \quad \forall i = 0, 1, \ldots, \]
which implies that
\[ K_l(H_x, s^{(0)}) \subseteq T_xM, \quad \forall l = 1, 2, \ldots. \]
Consequently, according to (4.3), we conclude that directly applying MINRES to the linear system (4.4) with initial guess $N^{(0)} \in T_xM$ will produce an approximation of the Newton direction of (2.9) in $T_xM$. A natural choice for the initial guess is $N^{(0)} = 0$.

Indeed, we point out that to obtain an approximation to (2.9), the initial guess $N^{(0)}$ for MINRES can be an arbitrary vector in $\mathbb{R}^n$. This is because if $\tilde{N}$ is an approximation to the linear system (4.4), then the new vector $N = P_x \tilde{N} \in T_xM$ satisfies
\[ H_x N = H_x (P_x \tilde{N}) = H_x \tilde{N}, \quad \text{and} \quad \| \frac{g(x)}{2} + H_x N \|_2 = \| \frac{g(x)}{2} + H_x \tilde{N} \|_2, \]
which implies that $N$ is also an approximation to the Newton direction defined by (2.9). Furthermore, the relation (4.5) also implies
\[ \min_{N \in \mathbb{R}^n} \| \frac{g(x)}{2} + H_x N \|_2 = \min_{N \in T_xM} \| \frac{g(x)}{2} + H_x N \|_2, \]
and the following property holds.

**Proposition 4.1.** Let $x \in M$. Then (2.9) has a solution if and only if the singular linear system (4.4) is consistent.

These relations suggest that we can directly apply MINRES to the singular linear system (4.4). We notice that for singular linear systems, the behavior of any residual minimizing Krylov subspace method that is mathematically equivalent to GMRES has been discussed, e.g., in [13]. As the singular matrix $H_x$ is symmetric, one of the main results (Theorem 2.4 of [13]) reveals that the MINRES iterates converge safely to a least-squares solution of (4.4). Refer to [13] for more discussions on the behavior of GMRES for singular systems.

**Theorem 4.2.** Let $x \in M$. For any initial guess $N^{(0)} \in \mathbb{R}^n$, the MINRES algorithm for the linear system (4.4) determines a solution $\tilde{N}$ for the least-squares problem
\[ \min_{N \in \mathbb{R}^n} \| \frac{g(x)}{2} + H_x N \|_2 \]
without breakdown at some step and breaks down at the next step. The vector $N = P_x \tilde{N}$ is a solution for
\[ \min_{N \in T_xM} \| \frac{g(x)}{2} + H_x N \|_2. \]
Furthermore, if (4.4) is consistent and $N^{(0)} = P_x y$ for some $y \in \mathbb{R}^n$, then the solution $\tilde{N}$ reached is in $T_xM$ and is the pseudoinverse solution (i.e., a solution for (4.7) and (4.8) with minimal-length).
Proof. The proof is straightforward based on Theorem 2.4 in [13]. \qed

Theorem 4.2 implies that an arbitrary \( N(0) \in \mathbb{R}^n \) can be the initial guess for MINRES, but we need an additional step to project the resulted solution onto \( T_xM \) to generate a (approximation) solution for (2.9). Theorem 4.2 also gives the evidence that \( N(0) = 0 \) is a good initial guess for MINRES, and in all our numerical testings, we used this strategy.

Based on Theorem 4.2, we know that in exact arithmetic, MINRES requires at most \( n - m \) steps to obtain a solution for (2.9) if \( N(0) = 0 \). In floating point arithmetic, however, due to the rounding errors during the computation, MINRES requires a few more and stagnation may occur; moreover ill conditioning can arise, and MINRES may even produce misleading results [13]. This may cause Algorithm 3 to diverge and is a possible reason for the failure cases in our testing examples. On the other hand, similar to the traditional inexact Newton method (see [23]), we observed that the accuracy of the approximation to the Newton direction affects the number of outer iterations of Algorithm 3. This can be observed in the following example.

Example 3. In this example, the matrix \( A \) is

\[
A = Q^T \Theta Q \in \mathbb{R}^{100 \times 100},
\]

where \( \Theta = \text{diag}\{10^{-2}, 10^{-2}, 3, 4, \ldots, 98, 10^2, 10^2\} \) and \( Q \) is an orthogonal matrix from the QR decomposition of a random 100-by-100 matrix. The matrix \( A \) then has repeated eigenvalues and its condition number is \( 10^4 \). We set \( m = 4 \) and \( \mathcal{P}_m = \{25, 25, 25, 25\} \).

We randomly generate \( 10^4 \) MEPs according to Example 3, and implement Algorithm 3 with different maximum numbers of iterations (maxit) for MINRES in solving (2.9). We set the starting point \( x(0) = \left[ \frac{1}{\sqrt{25}}, \ldots, \frac{1}{\sqrt{25}} \right]^T \) for all cases, and terminate the iteration whenever \( ||\delta x(k)||_2 < 10^{-6} \) or \( k > 500 \). We counted the cases for which the algorithm converges successfully to a solution of the MEP within the stopping criterion. The numerical results are reported in Table 4.1, where the percent of convergent cases and the average number of outer iterations (over \( 10^4 \) testing MEPs) are presented. In Figure 4.1, we also plotted the histories of \( \{\log_{10} ||\delta x(k)||_2\} \) for a particular problem of Example 3 with maxit=20, 30, 50, 90 and 120.

<table>
<thead>
<tr>
<th>Maxit for MINRES</th>
<th>20</th>
<th>30</th>
<th>50</th>
<th>90</th>
<th>120</th>
</tr>
</thead>
<tbody>
<tr>
<td># of convergent cases (%)</td>
<td>64.93</td>
<td>76.12</td>
<td>81.24</td>
<td>90.81</td>
<td>99.60</td>
</tr>
<tr>
<td>Avg. Iter. #</td>
<td>179.61</td>
<td>106.21</td>
<td>51.86</td>
<td>33.25</td>
<td>24.41</td>
</tr>
</tbody>
</table>

For Example 3, we observed that stagnation in outer iteration will occur if the maximum number of iterations for MINRES is small, and furthermore, Table 4.1 seems to suggest that high accuracy of the approximation to the solution for (2.9) is preferable. However, in general, it is not clear how accurate an inexact Newton direction should be in order to reduce the number of outer iterations and to ensure the global convergence of Algorithm 3. Our numerical experiences seem to indicate that even a modest error control for MINRES could possibly lead to the convergence. In
some cases, furthermore, we even observed that a small maximum number of iterations for MINRES could lead to fast convergence.\footnote{This may happen when some $x^{(k)}$ during the iterations falls into the basin of attraction of another MEP solution and fast convergence takes place onwards.}

We last point out that an inexact Newton direction $N$ can be alternatively computed from the equivalent saddle-point system (3.5). Many methods have been developed for a saddle-point problem

\begin{equation}
\begin{bmatrix}
F & J \\
J^T & 0
\end{bmatrix}
\begin{bmatrix}
f \\
h
\end{bmatrix} = \begin{bmatrix}
q \\
u
\end{bmatrix},
\end{equation}

and basically, they can be distinguished as three main approaches [24]. The first is to eliminate the variable $f$ from the second equation of (4.9) and solve for $h$ from the linear system

$$J^T F^{-1} J h = J^T F^{-1} q - u.$$

The vector $f$ then can be obtained from back substitution via solving

$$F f = q - J h.$$

This is a condensation direct approach which, by taking our special structure in (3.5), leads exactly to our RQI-MEP (Algorithm 4). The second approach is the Uzawa algorithm [9]; see also [12, 32]. Different from the first approach, the Uzawa algorithm is an iterative method and involves the inner and outer loops. The last approach is to directly solve (4.9) as a whole but using preconditioning techniques; see, e.g., [11, 24, 31, 32] and many references therein. Preconditioning techniques have been developed when $F$ is positive definite and $J$ has full column rank. For our problem (3.5), however, the $(1, 1)$ block $A - \Lambda(x)$ is not necessarily definite and hence designing a good preconditioner for (3.5) is an interesting research topic in the future.

\section{4.2. Preconditioning the Riemannian Newton equation.}

This subsection is dedicated to the preconditioning technique for the system (2.9). We will put our

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig4.1}
\caption{$\log_{10} \| \delta x^{(k)} \|_2$ from Algorithm 3 for different scenarios.}
\end{figure}
main efforts to developing the preconditioned versions (left-preconditioning and two-sided preconditioning) of the iterative methods for obtaining an inexact Newton direction. In section 5, we will provide and test some simple particular preconditioners.

To begin with, we assume that an approximation preconditioner $K \in \mathbb{R}^{n \times n}$ is available for which, in some sense, $K^{-1}(A - \Lambda(x)) \approx I^{[n]}$, and the linear system $Kf = q$ related to $K$ is inexpensive to solve. As the diagonal matrix $\Lambda(x)$ varies with the outer iteration count $k$, the preconditioner $K$ may also vary accordingly, although it is often practical to use the same $K$ during the iteration. Considering the special structure in (2.9), the preconditioner $K$ should also be restricted to the tangent space $T_x M$. (This procedure is the same as that for solving the correction equation for the Jacobi–Davidson algorithm [35, 36].) This means that we should work with

$$K = PxKP : T_x M \rightarrow T_x M$$

for which, in some sense, $K^{-1}H_x$ approximates the identity operator from $T_x M$ to itself. Therefore, with left-preconditioning, we should solve for $N$ from

$$K^{-1}H_x N = -K^{-1} g(\bar{x}) / 2, \quad N \in T_x M.$$

As $K^{-1}H_x$ is not necessarily symmetric, MINRES is not suitable for this system, and one should use the iterative solver like GMRES or the biconjugate gradient stabilized (BICGSTAB) algorithm [41].

At first glance, one may think that the system (4.11) is similar to (2.9) and the Krylov subspace solver GMRES could be directly used with the initial guess $N^{(0)} = 0$. But the difficulty in (4.11) is how to compute the vector $z := K^{-1}H_x v$ for a vector $v$ (in the tangent space $T_x M$ because $N^{(0)} = 0$) supplied by the Krylov subspace solver. Even though a linear system $Kf = q$ is not expensive to solve, it does not imply that computing $z$ from $Kz = H_x v$ is inexpensive, due to the fact that the special structure in $K$ may be destroyed in $K$. Considering the structure in $K$, however, we note that if $K$ is a block diagonal matrix with

$$K = \text{diag}\{K_1, \ldots, K_m\}, \quad K_i \in \mathbb{R}^{n_i \times n_i},$$

then so is $K$; moreover, thanks to the technique of Olsen, Jøgensen, and Simons [27] (see also [35, 36]), in this case, we will see that computing $z = K^{-1}H_x v$ can be implemented in a simple way.

To this end, we denote $y := H_x v$. Note from $Kz = y$ that

$$Kz = \begin{bmatrix} K_1 z_1 \\ \vdots \\ K_m z_m \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix},$$

where $K_i = (I^{[n_i]} - x_i x_i^T)K_i(I^{[n_i]} - x_i x_i^T)$ and $z_i, y_i \in \mathbb{R}^{n_i}$ are the subvectors of $z$ and $y$, respectively. Since $z \in T_x M$, we know that $z_i \perp x_i$, and from $K_i z_i = y_i$, it follows that

$$K_i z_i = y_i - \alpha_i x_i \quad \text{or} \quad z_i = K_i^{-1} y_i - \alpha_i K_i^{-1} x_i,$$

where $\alpha_i \in \mathbb{R}$. The condition $z_i \perp x_i$ allows us to determine

$$\alpha_i = \frac{x_i^T K_i^{-1} y_i}{x_i^T K_i^{-1} x_i}, \quad i = 1, \ldots, m.$$
Consequently, the computation scheme for \( z = \tilde{K}^{-1}H_x v \) in this case can be summarized in Algorithm 5. It should be noted that the vector \( q = [q_1^T, \ldots, q_m^T]^T \) in Algorithm 5 has to be solved only once in an iteration process for solving (4.11), and the right-hand side \( -\tilde{K}^{-1}\frac{g(x)}{2} \) can also be computed according to Algorithm 5 with a slight modification (i.e., set \( y = -\frac{g(x)}{2} \)).

\[
\text{Algorithm 5:} \quad \text{The matrix-vector product for } (\tilde{K}^{-1}H_x)v \text{ with } K = \text{diag}\{K_1, \ldots, K_m\}.
\]

\begin{verbatim}
compute \( y = H_x v \)
for \( i = 1, 2, \ldots, m \) do
  solve \( u_i \) from \( K_i u_i = x_i \);
  solve \( q_i \) from \( K_i q_i = y_i \);
  set \( z_i = y_i - \frac{1}{x_i} u_i \); 
end
\end{verbatim}

We conclude this section by mentioning that the technique in Algorithm 5 can also be used to form the two-sided preconditioning for (2.9). Suppose that \( K = \text{diag}\{K_1, \ldots, K_m\} \) and \( F = \text{diag}\{F_1, \ldots, F_m\} \) are two preconditioners such that \( \tilde{K}^{-1}H_x F^{-1} \) approximates the identity operator from \( \mathcal{T}_x M \) to itself, where \( \tilde{F} = P_x F P_x : \mathcal{T}_x M \to \mathcal{T}_x M \). To solve the preconditioned system

\[
(4.12) \quad \tilde{K}^{-1}H_x \tilde{F}^{-1} p = -\frac{1}{2} \tilde{K}^{-1}\frac{g(x)}{2} \quad (p = \tilde{F}N \in \mathcal{T}_x M),
\]

it is sufficient to provide the scheme for the matrix-vector product \((\tilde{K}^{-1}H_x \tilde{F}^{-1})v\). It is now clear that the computation procedure in Algorithm 5 allows us to compute \( t := \tilde{F}^{-1}v \) first and then \( \tilde{K}^{-1}H_x t \); furthermore, when an approximation \( \tilde{p} \) is obtained, the corresponding approximation \( \tilde{N} \) to (2.9) can also be recovered from solving \( \tilde{F}N = \tilde{p} \).

5. Numerical experiment. In this section, we implement the Riemannian Newton method with the preconditioning technique described in subsection 4.2. Many particular preconditioners can be used in solving (3.1), but a good preconditioner is always problem-dependent. Finding a good preconditioner is often viewed as a combination of art and science [32] which is beyond the scope of this paper. Instead, we attempt to show that the convergence behavior of Algorithm 3 could be significantly improved, if a proper preconditioner is used. For this purpose, we try (i) the left-preconditioning with a diagonal preconditioner \( K^D := \text{diag}(A - \Lambda(x)) \), and (ii) the two-sided preconditioning with \( K^L := \text{diag}\{K^L_1, \ldots, K^L_m\} \) and \( K^U := \text{diag}\{K^U_1, \ldots, K^U_m\} \) as the left and right preconditioners, respectively, where \([K^L_1, K^U_1] := lu(A_{ii} - \lambda_i(x)I^{[m,1]})\) is the LU factorization of \( A_{ii} - \lambda_i(x)I^{[m,1]} \). (In practice, it is not necessary to update the preconditioners \( K^L \) and \( K^U \) in every outer iteration; instead, we can use the same \( K^L \) and \( K^U \) in a fixed number of outer iterations to improve the efficiency; see [36] for more detailed discussions on the preconditioning techniques.) These preconditioners are tested on the following MEPs.

Example 4. We select two matrices for the MEP from the Matrix Market\(^3\) in the set BCSSTRUC1 (from the Harwell–Boeing collection). Both matrices are positive definite and their sizes and condition numbers are summarized in Table 5.1. For each

\[^3\text{http://math.nist.gov/MatrixMarket/}\]
matrix, we choose different m’s and define the corresponding positive integers set \( P_m \) as
\[
P_m = \left\{ \frac{n}{m}, \frac{n}{m}, \ldots, \frac{n}{m} \right\} \quad \text{(i.e., } n_i = \frac{n}{m}) \]
This results to different MEPs.

For each generated MEP, we choose \( x^{(0)} = [\sqrt{\frac{n}{m}}, \ldots, \sqrt{\frac{n}{m}}]^{\top} \in \mathbb{R}^n \) and terminate the iteration whenever \( \|\delta x^{(k)}\|_2 < 10^{-6} \) or \( k > 500 \). In solving (3.1), we employ MINRES when no preconditioner is used (i.e., \( K = I^n \)), while we employ GMRES for the other two left-preconditioning versions. The performance of Algorithm 3 with various maxit for the Krylov subspace solvers is compared. In Tables 5.2 and 5.3, we reported both the number of outer iterations and the total numbers used in the Krylov subspace solver for different scenarios. For each matrix \( A \), we observed that for a fixed pair \((m, \maxit)\), the convergent points from different preconditioners \((I^n, K_D)\) and \((K_L, K_U)\) are the same, except for five cases marked by “∗.”

These numerical results indicate that the left-preconditioning with the diagonal precondioner \( K_D \) and the two-sided preconditioning using the block diagonal with triangular blocks \((K_L, K_U)\) are very helpful in solving these MEPs. It also shows that a crude approximation to the Newton direction may cause stagnation or even

<table>
<thead>
<tr>
<th>Table 5.1</th>
<th>Summary of the testing matrices in Example 4.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matrix name</td>
<td>Size ((n \times n))</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>BCSSTK02</td>
<td>66 \times 66</td>
</tr>
<tr>
<td>BCSSTK04</td>
<td>132 \times 132</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 5.2</th>
<th>The number of iterations for different scenarios for BCSSTK02.</th>
</tr>
</thead>
<tbody>
<tr>
<td>BCSSTK02</td>
<td>(K = I^n)</td>
</tr>
<tr>
<td>m</td>
<td>maxit</td>
</tr>
<tr>
<td>-----</td>
<td>-------</td>
</tr>
<tr>
<td>m=3</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>128</td>
</tr>
<tr>
<td>30</td>
<td>49</td>
</tr>
<tr>
<td>50</td>
<td>15</td>
</tr>
<tr>
<td>60</td>
<td>11</td>
</tr>
<tr>
<td>m=6</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>67</td>
</tr>
<tr>
<td>30</td>
<td>27</td>
</tr>
<tr>
<td>50</td>
<td>14*</td>
</tr>
<tr>
<td>60</td>
<td>12</td>
</tr>
<tr>
<td>m=11</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>370</td>
</tr>
<tr>
<td>30</td>
<td>25</td>
</tr>
<tr>
<td>50</td>
<td>21</td>
</tr>
<tr>
<td>60</td>
<td>19</td>
</tr>
</tbody>
</table>

These numerical results indicate that the left-preconditioning with the diagonal precondioner \( K_D \) and the two-sided preconditioning using the block diagonal with triangular blocks \((K_L, K_U)\) are very helpful in solving these MEPs. It also shows that a crude approximation to the Newton direction may cause stagnation or even
Table 5.3
The number of iterations for different scenarios for BCSSTK04.

<table>
<thead>
<tr>
<th>BCSSTK04</th>
<th>$K = I^{[n]}$</th>
<th>$K^{[I]}$</th>
<th>$(K^{[L]}, K^{[U]})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>m</td>
<td>maxit</td>
<td>outer iter.#</td>
<td>MINRES</td>
</tr>
<tr>
<td>15</td>
<td>275</td>
<td>4125</td>
<td>14</td>
</tr>
<tr>
<td>25</td>
<td>98</td>
<td>2450</td>
<td>8</td>
</tr>
<tr>
<td>40</td>
<td>32*</td>
<td>1280</td>
<td>6</td>
</tr>
<tr>
<td>70</td>
<td>8</td>
<td>560</td>
<td>6</td>
</tr>
<tr>
<td>90</td>
<td>7</td>
<td>630</td>
<td>6</td>
</tr>
<tr>
<td>10</td>
<td>fail</td>
<td>fail</td>
<td>fail</td>
</tr>
<tr>
<td>20</td>
<td>fail</td>
<td>fail</td>
<td>fail</td>
</tr>
<tr>
<td>30</td>
<td>fail</td>
<td>fail</td>
<td>10</td>
</tr>
<tr>
<td>70</td>
<td>19</td>
<td>1330</td>
<td>10</td>
</tr>
<tr>
<td>90</td>
<td>9</td>
<td>810</td>
<td>10</td>
</tr>
</tbody>
</table>

divergence, and a proper preconditioner, on the other hand, is a possible remedy for the convergence.

6. Concluding remarks. Viewing the MEP as a special generalization of the classical eigenvalue problem, we have developed, from an optimization point of view, a Rayleigh quotient type iteration. The method takes advantage of the special properties in the related optimization problem (1.5) and uses the second-order information. The local and global convergence has also been investigated theoretically and numerically. The key step for this method is to solve the Newton equation, and we developed a direct method and an inexact implementation by employing a Krylov subspace solver with specially designed preconditioning techniques. Our preliminary numerical testing indicates that the method has a good convergence behavior in terms of its speed and global convergence.

Acknowledgments. Special thanks to Professor Moody T. Chu for introducing and suggesting the MCP and MEP to the author during his visit to North Carolina State University. The author is also grateful to the Associate Editor and two anonymous referees for their very helpful comments and suggestions that improved this paper significantly. One of the referees drew our attention to the references [2, 3, 27, 43] and the technique of Olsen, Jøgensen, and Simons [27], which leads us to a more clear way (in subsection 3.1) to derive the closed-form of the Newton direction defined by (3.1).

REFERENCES

RIEMANNIAN NEWTON METHOD FOR THE MEP


