Journal of Computational Mathematics
Vol.32, No.1, 2014, 93–106.

http://www.global-sci.org/jcm
doi:10.4208/jcm.1310-m4225

A NATURAL GRADIENT ALGORITHM FOR THE SOLUTION OF LYAPUNOV EQUATIONS BASED ON THE GEODESIC DISTANCE[*]

Xiaomin Duan
School of Mathematics, Beijing Institute of Technology, Beijing 100081, China
School of Science, Dalian Jiaotong University, Dalian 116028, China
Email: dxmhope@gmail.com
Huafei Sun
School of Mathematics, Beijing Institute of Technology, Beijing 100081, China
Email: huaf.sun@gmail.com
Zhenning Zhang
College of Applied Sciences, Beijing University of Technology, Beijing 100124, China
Email: ningning0327@163.com

Abstract

A new framework based on the curved Riemannian manifold is proposed to calculate the numerical solution of the Lyapunov matrix equation by using a natural gradient descent algorithm and taking the geodesic distance as the objective function. Moreover, a gradient descent algorithm based on the classical Euclidean distance is provided to compare with this natural gradient descent algorithm. Furthermore, the behaviors of two proposed algorithms and the conventional modified conjugate gradient algorithm are compared and demonstrated by two simulation examples. By comparison, it is shown that the convergence speed of the natural gradient descent algorithm is faster than both of the gradient descent algorithm and the conventional modified conjugate gradient algorithm in solving the Lyapunov equation.

Mathematics subject classification: 65F10, 53B21, 90C26, 93C05.
Key words: Lyapunov equation, Geodesic distance, Natural gradient descent algorithm.

1. Introduction

Lyapunov matrix equation is widely used in many fields, such as linear control systems, stochastic analysis of dynamical systems and circuit simulations, see, e.g., [1–5]. The solution of this equation has been paid more and more attention in the computational mathematics field, see, e.g., [6–9]. In particular, Li and White presented a Cholesky factor-alternating direction implicit algorithm to generate a low-rank approximation solution of the Lyapunov equation [10]. Vandereycken and Vandewalle provided a Riemannian optimization approach to compute the low-rank solution of the Lyapunov equation [11]. Jbilou proposed a preconditioned Krylov method for solving Lyapunov matrix equations [12]. Deng, Bai and Gao designed iterative orthogonal direction methods according to the fundamental idea of the classical conjugate direction method for the standard system of linear equations to obtain the Hermitian solutions of the linear matrix equations $AXB = C$ and $(AX, XB) = (C, D)$ [13]. Recently, Su and Chen

[*] Received July 21, 2012 / Revised version received August 21, 2013 / Accepted October 9, 2013 / Published online January 22, 2014 /

proposed a modified conjugate gradient algorithm (MCGA) to solve Lyapunov matrix equations and some other linear matrix equations, which seemed to be the generalized result of [14].

Up to date, however, there has been few investigation on the solution problem of the Lyapunov matrix equation in the view of Riemannian manifolds. Note that this solution is a positive definite symmetric matrix in a global asymptotically stable linear system and the set of all the positive definite symmetric matrices can be taken as a manifold. Thus, it is more convenient to investigate the solution problem with the help of these attractive features on the manifold. To address such a need, we focus on a numerical method to solve the Lyapunov matrix equation on the manifold.

In the present paper, with the help of a natural gradient descent algorithm (NGDA), a new framework is proposed to calculate the numerical solution of the Lyapunov matrix equation on the manifold of positive definite matrices. Moreover, the geodesic distance on the curved Riemannian manifold is taken as the objective function of the NGDA. In order to compare with this NGDA, a gradient descent algorithm (GDA) based on the classical Euclidean distance is provided. Furthermore, the behaviors of NGDA, GDA and the conventional MCGA are compared and demonstrated by two simulation examples.

The remainder of the paper is organized as follows. The geometric structures on the manifold of positive definite matrices are introduced in Section 2. Then the GDA and NGDA are proposed to solve the Lyapunov matrix equation in Sections 3. Finally, two numerical examples are provided to compare the convergence speed and the computation consumption of the presented algorithms and the MCGA.

2. Preliminaries

Let $\mathrm{M}(n)$ be the set of $n \times n$ real matrices and $\mathrm{GL}(n)$ be its subset containing only nonsingular matrices. Then $\mathrm{GL}(n)$ is a Lie group, namely, a group which is also a differentiable manifold and on which the operations of the group multiplication and the inverse are smooth. The tangent space at the identity of $\mathrm{GL}(n)$ is called the corresponding Lie algebra, which is a space which consists of all linear transformations, namely $\mathrm{M}(n)$. The set of all the $n \times n$ positive definite symmetric matrices can be taken as a manifold, denoted by $\mathrm{PD}(n)$. Manifold $\mathrm{PD}(n)$ is not a Lie subgroup of $\mathrm{GL}(n)$, while it is a submanifold of $\mathrm{GL}(n)$. On manifold $\mathrm{PD}(n)$, we can define different metrics so that different geometry structures are established on this manifold. In this section, we briefly introduce these geometric structures which will be used in the following sections. More details can be found in [15–18].

2.1. Tangent space on manifold $\mathrm{PD}(n)$

The exponential map of a matrix $U \in \mathrm{M}(n)$ is given, as usual, by the convergent series

$$\exp(U) = \sum_{m=0}^{\infty} \frac{U^m}{m!}. \tag{2.1}$$

The inverse map, i.e., the logarithmic map is defined as follows

$$\log(V) = \sum_{m=1}^{\infty} (-1)^{m+1} \frac{(V-I)^m}{m}, \tag{2.2}$$

for V in a neighborhood of the identity I of $\mathrm{GL}(n)$.

Let us denote the space of all $n \times n$ symmetric matrices by $\mathrm{Sym}(n)$. We recall that the exponential map from $\mathrm{Sym}(n)$ to $\mathrm{PD}(n)$ is one-to-one and onto. In other words, the exponential map of any symmetric matrix is a positive definite symmetric matrix, and the logarithm map of any positive definite symmetric matrix is a symmetric matrix. As $\mathrm{PD}(n)$ is an open subset of $\mathrm{Sym}(n)$, for each $B \in \mathrm{PD}(n)$ we identify the tangent space $T_B\mathrm{PD}(n)$ at B with $\mathrm{Sym}(n)$.

2.2. Frobenius inner product and Euclidean distance

By regarding $\mathrm{PD}(n)$ as a subset of the Euclidean space $\mathrm{M}(n)$, the Frobenius inner product on it is defined by

$$\langle B_1, B_2\rangle = \mathrm{tr}(B_1^T B_2), \tag{2.3}$$

where $B_1, B_2 \in \mathrm{PD}(n)$ and tr denotes the trace of the matrix. The associated norm $\|B\|_F = \langle B, B\rangle^{\frac{1}{2}}$ is used to define the Euclidean distance on $\mathrm{M}(n)$ by

$$d_F(B_1, B_2) = \|B_1 - B_2\|_F. \tag{2.4}$$

Moreover, the geodesic passing through B in the direction of $W \in \mathrm{Sym}(n)$ with respect to the Frobenius inner product is given by

$$\gamma(t) = B + tW, \quad 0 \le t < \delta, \tag{2.5}$$

for some $\delta > 0$ and this geodesic stays within manifold $\mathrm{PD}(n)$ under the condition that t is not too large.

2.3. Riemannian metric and geodesic distance

On manifold $\mathrm{PD}(n)$, the Riemannian metric at the point B is defined by

$$g_B(X, Y) = g_I(B^{-1}X, B^{-1}Y) = \mathrm{tr}(B^{-1}XB^{-1}Y), \tag{2.6}$$

where I denotes the identity matrix, $X, Y \in \mathrm{Sym}(n)$. The positive definiteness of this metric is guaranteed by

$$g_B(X, X) = \mathrm{tr}(B^{-\frac{1}{2}}XB^{-\frac{1}{2}}B^{-\frac{1}{2}}XB^{-\frac{1}{2}}). \tag{2.7}$$

Then, the manifold $\mathrm{PD}(n)$ with the Riemannian metric (2.6) becomes a Riemannian manifold. Moreover, it is a curved manifold since this metric is invariant under the group action of $\mathrm{GL}(n)$.

Next, we discuss how to define the geodesic distance on manifold $\mathrm{PD}(n)$ with the Riemannian metric (2.6). Let $[0, 1]$ be a closed interval in $\mathbb{R}$, and $\phi : [0, 1] \to \mathrm{PD}(n)$ be a sufficiently smooth curve on manifold $\mathrm{PD}(n)$. We define the length of $\phi(t)$ by

$$\ell(\phi(t)) := \int_0^1 \sqrt{g_{\phi(t)}(\dot{\phi}(t), \dot{\phi}(t))}dt = \int_0^1 \sqrt{\mathrm{tr}(\phi(t)^{-1}\dot{\phi}(t))^2}dt. \tag{2.8}$$

The geodesic distance between two matrices B_1 and B_2 on manifold $\mathrm{PD}(n)$ is the infimum of lengths of curves connecting them, namely

$$d_g(B_1, B_2) := \inf\Big\{\ell(\phi) | \phi : [0, 1] \to \mathrm{PD}(n), \phi(0) = B_1, \phi(1) = B_2\Big\}. \tag{2.9}$$

It transpires that the length-minimizing smooth curves are geodesics, so the infimum of (2.9) is achieved by a geodesic. The Hopf-Rinow theorem implies that manifold $\mathrm{PD}(n)$ is geodesically complete [19]. This means that the geodesic for the interval $[0,1]$ can be extended to $(-\infty,+\infty)$. Therefore, for any given pair $B_1, B_2 \in \mathrm{PD}(n)$, we can find a geodesic $\gamma(t)$ such that $\gamma(0)=B_1$ and $\gamma(1)=B_2$, by taking the initial velocity as

$$\dot{\gamma}(0) = B_1^{\frac{1}{2}} \log(B_1^{-\frac{1}{2}} B_2 B_1^{-\frac{1}{2}}) B_1^{\frac{1}{2}} \in \mathrm{Sym}(n).$$

Moreover, the geodesic $\gamma(t)$ can be expressed as

$$\gamma(t) = B_1^{\frac{1}{2}} \exp(t \log(B_1^{-\frac{1}{2}} B_2 B_1^{-\frac{1}{2}})) B_1^{\frac{1}{2}} = B_1^{\frac{1}{2}} (B_1^{-\frac{1}{2}} B_2 B_1^{-\frac{1}{2}})^t B_1^{\frac{1}{2}}. \tag{2.10}$$

Obviously, the geodesic (2.10) is entirely contained in the manifold. Therefore, according to (2.8), the geodesic distance $d_g(B_1,B_2)$ can be computed explicitly by

$$d_g(B_1,B_2) = \| \log(B_1^{-\frac{1}{2}} B_2 B_1^{-\frac{1}{2}}) \|_F = \left(\sum_{i=1}^{n} \ln(\lambda_i)^2 \right)^{\frac{1}{2}}, \tag{2.11}$$

where λ_i are the eigenvalues of $B_1^{-\frac{1}{2}} B_2 B_1^{-\frac{1}{2}}$. Since λ_i are also the eigenvalues of $B_1^{-1}B_2$, we need not to invoke the matrix square root $B_1^{-\frac{1}{2}}$ in practice.

3. Algorithms for the Solution of Lyapunov Matrix Equation

It is well known that a linear time-invariant autonomous system

$$\dot{x} = Ax, \tag{3.1}$$

where $A \in \mathbb{R}^{n\times n}$, is global asymptotically stable, if for any given positive definite symmetric matrix Q, there exists a positive definite symmetric matrix P satisfying the Lyapunov matrix equation

$$A^T P + PA + Q = 0, \tag{3.2}$$

and the scalar function $f(t) = x(t)^T P x(t)$ is strictly decreasing. We assume that all the eigenvalues of the system matrix A have negative real parts. It ensures that system (3.1) is global asymptotically stable and the solution P for equation (3.2) is unique [20].

Firstly, we define the coordinate system of manifold PD(n) as follows. Let E_{pq} be the matrix with one as the (p,q)th element and the zero otherwise. Then the basis matrices of the vector space $\mathrm{Sym}(n)$ are defined by

$$E_u = E_{\tau(p,q)} = \begin{cases} E_{pq} & p = q, \\ E_{pq} + E_{qp} & p < q, \end{cases} \tag{3.3}$$

where τ is an appropriate rule to assign integers to the pair (p,q), namely, $\tau(p,q) = u$, for $1 \le u \le N := n(n+1)/2$. Thus we can represent any $P(\theta) \in \mathrm{PD}(n)$ as

$$P(\theta) = \sum_{i=1}^{N} \theta^i E_i, \theta^i \in \mathbb{R}, \tag{3.4}$$

uniquely, where $\theta = (\theta^1, \theta^2, ..., \theta^N)$ belongs to an open subset of $\mathbb{R}^N$ that satisfies the positive definiteness. Hence, θ can be considered as a global coordinate system of the N-dimensional manifold $\mathrm{PD}(n)$ [21].

Then, our purpose is to seek a matrix $P(\theta)$ on manifold $\mathrm{PD}(n)$ such that the matrix $-(A^T P(\theta) + P(\theta)A)$ is as close as possible to the given matrix Q. Therefore, the key points for designing an algorithm are as follows:

1. The difference between the given positive definite matrix Q and the matrix $-(A^TP(\theta)+P(\theta)A)$ is calculated, for an arbitrary matrix $P(\theta)$ on manifold $\mathrm{PD}(n)$.
2. A gradient descent algorithm is used to adjust the coordinate θ so that the difference can be as small as possible.

3.1. Gradient descent algorithm

In this subsection, we provide the traditional gradient descent algorithm(GDA) to solve equation (3.2). The difference between Q and $-(A^TP(\theta)+P(\theta)A)$ is expressed by the Euclidean distance

$$J_F(\theta) = d_F^2\Big(Q, -(A^TP(\theta)+P(\theta)A)\Big) = \mathrm{tr}\Big((Q+A^TP(\theta)+P(\theta)A)^2\Big), \tag{3.5}$$

which is taken as an objective function. Note that there is a unique solution for the Lyapunov equation (3.2), so the minimum of $J_F(\theta)$ must exist uniquely. Then the coordinate of the solution for the Lyapunov matrix equation (3.2) can be obtained by the closest point. That is the point θ_* such that

$$\theta_* = \arg\min_{\theta} J_F(\theta). \tag{3.6}$$

Moreover, the matrix $P(\theta_*) = \sum_{i=1}^N \theta_*^i E_i$ is the solution of the equation (3.2).

Lemma 3.1. ([22]) Let $X(t)$ be a $m\times n$ differentiable function-valued matrix of the real variable t for all t in its domain, then we have that

$$\frac{d}{dt}\mathrm{tr}(X(t)^TX(t)) = 2\,\mathrm{tr}(X(t)^T\frac{d}{dt}X(t)). \tag{3.7}$$

Furthermore, if $C_1 \in \mathbb{R}^{l\times m}, C_2 \in \mathbb{R}^{n\times l}$ are constant matrices, then

$$\frac{d}{dt}\mathrm{tr}(C_1X(t)C_2) = \mathrm{tr}(C_1\frac{d}{dt}X(t)C_2). \tag{3.8}$$

Moreover, if $m=n$ and $X(t)$ is an invertible matrix which does not have eigenvalues on the closed negative real line, then

$$\frac{d}{dt}\mathrm{tr}\Big(\log X(t)\Big) = \mathrm{tr}(X(t)^{-1}\frac{d}{dt}X(t)). \tag{3.9}$$

Now, we use the GDA to minimize $J_F(\theta)$, which is widely applied in Euclidean spaces.

Theorem 3.1. *The iterative process based on the Euclidean distance $J_F(\theta)$ is given by*

$$\theta_{t+1} = \theta_t - \varsigma_t\nabla J_F(\theta_t), \tag{3.10}$$

where ς_t denotes the step size and the components of the gradient $\nabla J_F(\theta_t)$ satisfy

$$\frac{\partial}{\partial\theta^i}J_F(\theta_t) = 2\,\mathrm{tr}\Big((Q+A^TP(\theta_t)+P(\theta_t)A)(A^TE_i+E_iA)\Big). \tag{3.11}$$

Proof. In this case, manifold $\mathrm{PD}(n)$ is regarded as a subset of $\mathrm{M}(n)$. Then, for the coordinate θ of the matrix $P(\theta)$, the squared length of a small incremental vector $d\theta$ connecting θ and $\theta+d\theta$ is given by

$$\|d\theta\|_F^2 = \sum_{i=1}^N (d\theta_i)^2, \tag{3.12}$$

where $d\theta_i$ are the components of $d\theta$. The gradient descent direction is defined by the vector $d\theta$ that minimizes $J_F(\theta + d\theta)$ under the constraint

$$\|d\theta\|_F^2 = \varepsilon^2, \tag{3.13}$$

for a sufficiently small constant ε. We put

$$d\theta = \varepsilon \mathbf{a}, \tag{3.14}$$

and search $\mathbf{a}$ that minimizes

$$J_F(\theta + d\theta) = J_F(\theta) + \varepsilon \nabla J_F(\theta) \cdot \mathbf{a}, \tag{3.15}$$

under the constraint

$$\|\mathbf{a}\|_F^2 = \sum_{i=1}^{N} a_i^2 = 1. \tag{3.16}$$

By the Lagrange method, we have

$$\frac{\partial}{\partial a_i}(\nabla J_F(\theta)^T \mathbf{a} - \lambda \mathbf{a}^T \mathbf{a}) = 0, \tag{3.17}$$

where λ is Lagrange multiplier, so it is obtained that

$$\mathbf{a} = \frac{1}{2\lambda} \nabla J_F(\theta). \tag{3.18}$$

From (3.14), we have the iterative formula of the GDA that

$$\theta_{t+1} = \theta_t - \varsigma_t \nabla J_F(\theta_t), \tag{3.19}$$

where ς_t is the step size. And, since the matrix $A^T P(\theta) + P(\theta)A$ is symmetric, we take advantage of (3.7) and (3.8) to get the components of the gradient $\nabla J_F(\theta_t)$

$$\begin{aligned} \frac{\partial}{\partial \theta^i} J_F(\theta_t) &= 2\,\mathrm{tr}\left((Q + A^T P(\theta_t) + P(\theta_t)A) \frac{\partial}{\partial \theta^i}(Q + A^T P(\theta_t) + P(\theta_t)A)\right) \\ &= 2\,\mathrm{tr}\left((Q + A^T P(\theta_t) + P(\theta_t)A)(A^T E_i + E_i A)\right). \end{aligned} \tag{3.20}$$

This completes the proof of Theorem 3.1. □

In Theorem 3.1, we use the negative gradient $-\nabla J_F(\theta)$ to determine a new search direction in each iteration, so that the objective function $J_F(\theta)$ reduces gradually. Now, we discuss the convergence of the sequence θ_t generated by (3.10).

Corollary 3.1. *If*

$$0 < \varsigma_t < \frac{2 J_F(\theta_t)}{\|\nabla J_F(\theta_t)\|^2}, \tag{3.21}$$

then the sequence θ_t generated by (3.10) *converges to the coordinate θ_* of the solution $P(\theta_*)$ for equation* (3.2). *Moreover, the gradient $\nabla J_F(\theta)$ exists the unique zero point θ_*.*

Proof. Subtracting θ_* from both sides of (3.10) gives

$$\theta_{t+1} - \theta_* = \theta_t - \theta_* - \varsigma_t \nabla J_F(\theta_t). \tag{3.22}$$

It can be verified that

$$\begin{aligned} \|\theta_{t+1} - \theta_*\|_F^2 &= \|\theta_t - \theta_*\|_F^2 - 2\varsigma_t(\theta_t - \theta_*)^T \nabla J_F(\theta_t) + 2\varsigma_t^2 \|\nabla J_F(\theta_t)\|_F^2 \\ &= \Big(1 - 2\varsigma_t \frac{(\theta_t - \theta_*)^T \nabla J_F(\theta_t)}{\|\theta_t - \theta_*\|_F^2} + 2\varsigma_t^2 \frac{\|\nabla J_F(\theta_t)\|_F^2}{\|\theta_t - \theta_*\|_F^2}\Big) \|\theta_t - \theta_*\|_F^2. \end{aligned} \tag{3.23}$$

It follows from (3.11) that

$$(\theta_t - \theta_*)^T \nabla J_F(\theta_t) = 2J_F(\theta_t). \tag{3.24}$$

Thus, we can rewrite (3.23) as

$$\|\theta_{t+1} - \theta_*\|_F^2 = \Big(1 - 2\varsigma_t \frac{2J_F(\theta_t)}{\|\theta_t - \theta_*\|_F^2} + 2\varsigma_t^2 \frac{\|\nabla J_F(\theta_t)\|_F^2}{\|\theta_t - \theta_*\|_F^2}\Big)(\|\theta_t - \theta_*\|_F^2). \tag{3.25}$$

If (3.21) is true, then we have

$$\|\theta_{t+1} - \theta_*\|_F < \|\theta_t - \theta_*\|_F, \tag{3.26}$$

which implies that the sequence θ_t converges to the coordinate θ_*.

In fact, if there exists $\theta'_* \neq \theta_*$ such that the iterative process stops, then it means $\nabla J_F(\theta'_*) = 0$. However, from (3.24), we get that

$$(\theta'_* - \theta_*)^T \nabla J_F(\theta'_*) = 2J_F(\theta'_*) = 0, \tag{3.27}$$

which shows that $P(\theta'_*)$ is another solution of Eq. (3.2). That is contradictory to the uniqueness of the solution for Eq. (3.2). Consequently, θ_* is the unique zero point of $\nabla J_F(\theta)$. This completes the proof of Corollary 3.1. □

From the above discussion, we formulate the GDA as follows and denote the spectral radius of the matrix by $\rho(\cdot)$.

Algorithm 3.1. (GDA) *For the coordinate $\theta = (\theta^1, \theta^2, \ldots, \theta^N)$ of manifold PD(n), the gradient descent algorithm is given by*

```
1. Set $\theta_0 = (\theta_0^1, \theta_0^2, \ldots, \theta_0^N)$ as the coordinate of an initial input matrix $P(\theta_0)$ and choose
   a desired tolerance $\varepsilon_0 > 0$.
2. Compute $\rho(A^T P(\theta_t) + P(\theta_t)A + Q)$.
3. If $\rho(A^T P(\theta_t) + P(\theta_t)A + Q) < \varepsilon_0$ then stop.
4. Update the coordinate vector $\theta$ by $\theta_{t+1} = \theta_t - \varsigma_t \nabla J_F(\theta_t)$ and go to step 2.
```

3.2. Natural gradient descent algorithm

On a curved Riemannian manifold, the GDA can not give the steepest descent direction of the objective function. In order to overcome this difficulty, Amari et. al. proposed a natural gradient descent algorithm(NGDA) to give the steepest descent direction on manifolds [23], as follows.

Lemma 3.2. *Let $\{u \in \mathbb{R}^n\}$ be a parameter space on which a function $L(u)$ is defined. The iterative process in the steepest descent direction on a Riemannian manifold is given by*

$$u_{t+1} = u_t - \eta_t G(u_t)^{-1} \nabla L(u_t), \tag{3.28}$$

where η_t denotes the step size, $G(u_t)^{-1} = (g^{ij}(u_t))$ is the inverse of the Riemannian metric $G(u_t) = (g_{ij}(u_t))$ and

$$\nabla L(u) = \Big(\frac{\partial}{\partial u_1} L(u), \frac{\partial}{\partial u_2} L(u), \cdots, \frac{\partial}{\partial u_n} L(u)\Big). \tag{3.29}$$

This algorithm has been well used into some fields, such as the neural network and the optimal control [24–27]. Here we apply it to solve the Lyapunov equation (3.2) on the manifold PD(n) with the Riemannian metric (2.6).

Note that the geodesic distance is the shortest one on a Riemannian manifold, so we take the geodesic distance between Q and $-(A^T P(\theta) + P(\theta)A)$ as the objective function $J_g(\theta)$, which is in the form of

$$\begin{aligned} J_g(\theta) &= d_g^2(Q, -(A^T P(\theta) + P(\theta)A)) \\ &= \Big\| \log(Q^{-\frac{1}{2}}(-A^T P(\theta) - P(\theta)A)Q^{-\frac{1}{2}})) \Big\|_F^2. \end{aligned} \tag{3.30}$$

Therefore, the coordinate θ_* of the solution $P(\theta_*)$ for the Lyapunov equation (3.2) is obtained by minimizing the objective function $J_g(\theta)$, namely

$$\theta_* = \arg\min_\theta J_g(\theta). \tag{3.31}$$

Now, the NGDA is used to minimize $J_g(\theta)$ and the iterative formula is as follows.

Theorem 3.2. *On manifold PD(n), the iterative process in the steepest descent direction is given by*

$$\theta_{t+1} = \theta_t - \eta_t G(\theta_t)^{-1} \nabla J_g(\theta_t), \tag{3.32}$$

where the components of gradient $\nabla J_g(\theta_t)$ satisfy

$$\begin{aligned} &\frac{\partial}{\partial \theta^i} J_g(\theta_t) \\ &= 2\,\mathrm{tr}\Big(\log(Q^{-1}(-A^T P(\theta_t) - P(\theta_t)A))(A^T P(\theta_t) + P(\theta_t)A)^{-1}(A^T E_i + E_i A)\Big). \end{aligned} \tag{3.33}$$

Proof. On the manifold PD(n) with the Riemannian metric (2.6), the squared length of a small incremental vector $d\theta$ connecting θ and $\theta + d\theta$ is

$$\|d\theta\|_g^2 = \sum_{i,j=1}^{N} (g_{ij}(\theta) d\theta_i d\theta_j)^2. \tag{3.34}$$

By the Lagrange method, we have

$$\frac{\partial}{\partial a_i}(\nabla J_g(\theta)^T \mathbf{a} - \lambda \mathbf{a}^T G(\theta_t)\mathbf{a}) = 0, \tag{3.35}$$

so it can be obtained that

$$\mathbf{a} = \frac{1}{2\lambda} G(\theta_t)^{-1} \nabla J_g(\theta). \tag{3.36}$$

Therefore, the iterative formula is

$$\theta_{t+1} = \theta_t - \eta_t G(\theta_t)^{-1} \nabla J_g(\theta_t). \tag{3.37}$$

Furthermore, combining Lemma 3.1 and the properties of the matrix trace, the components of $\nabla J_g(\theta_t)$ are

$$\begin{aligned}
&\frac{\partial}{\partial \theta^i} J_g(\theta_t) \\
&= 2\operatorname{tr}\Big(\log(Q^{-\frac{1}{2}}(-A^T P(\theta_t) - P(\theta_t)A)Q^{-\frac{1}{2}}))\frac{\partial}{\partial \theta^i}(\log(Q^{-\frac{1}{2}}(-A^T P(\theta_t) - P(\theta_t)A)Q^{-\frac{1}{2}}))\Big) \\
&= 2\operatorname{tr}\Big(\log(Q^{-1}(-A^T P(\theta_t) - P(\theta_t)A))(A^T P(\theta_t) + P(\theta_t)A)^{-1}(A^T E_i + E_i A)\Big).
\end{aligned} \tag{3.38}$$

This completes the proof of Theorem 3.2. □

Now, we discuss the convergence of the sequence θ_t generated by (3.32).

Corollary 3.2. *If*

$$0 < \eta_t < \frac{2\sum_{i=1}^n (\lambda_t^i - 1)\log \lambda_t^i}{\nabla J_g(\theta_t)^T G(\theta_t)^{-1} \nabla J_g(\theta_t)}, \tag{3.39}$$

then the sequence θ_t generated by (3.32) converges to the coordinate θ_* of the solution $P(\theta_*)$ for equation (3.2), where $\lambda_t^i (i = 1, \ldots, n)$ are the eigenvalues of $Q^{\frac{1}{2}}(-A^T P(\theta_t) - P(\theta_t)A)^{-1} Q^{\frac{1}{2}}$. Moreover, the gradient $\nabla J_g(\theta)$ exists the unique zero point θ_*.

Proof. From (3.32) and (3.34), we can obtain the equalities that

$$\begin{aligned}
&\|\theta_{t+1} - \theta_*\|_g^2 \\
&= \|\theta_t - \theta_*\|_g^2 - 2\eta_t(\theta_t - \theta_*)^T \nabla J_g(\theta_t) + 2\eta_t^2 \nabla J_g(\theta_t)^T G(\theta_t)^{-1} \nabla J_g(\theta_t) \\
&= \Big(1 - 2\eta_t \frac{(\theta_t - \theta_*)^T \nabla J_g(\theta_t)}{\|\theta_t - \theta_*\|_g^2} + 2\eta_t^2 \frac{\nabla J_g(\theta_t)^T G(\theta_t)^{-1} \nabla J_g(\theta_t)}{\|\theta_t - \theta_*\|_g^2}\Big)\|\theta_t - \theta_*\|_g^2.
\end{aligned} \tag{3.40}$$

Then, using the properties of the matrix trace, we have

$$\begin{aligned}
&(\theta_t - \theta_*)^T \nabla J_g(\theta_t) \\
&= 2\operatorname{tr}\big(\log(Q^{\frac{1}{2}}(-A^T P(\theta_t) - P(\theta_t)A)^{-1} Q^{\frac{1}{2}}))(Q^{\frac{1}{2}}(-A^T P(\theta_t) - P(\theta_t)A)^{-1} Q^{\frac{1}{2}} - I))\big).
\end{aligned} \tag{3.41}$$

Since $Q^{\frac{1}{2}}(-A^T P(\theta_t) - P(\theta_t)A)^{-1} Q^{\frac{1}{2}}$ is symmetric positive definite, its orthogonal similarity diagonalization can be represented as $U\Lambda U^T$, where U is the orthogonal matrix and $\Lambda = diag(\lambda_t^1, \ldots, \lambda_t^n)$ satisfying that $\lambda_t^i > 0, i = 1, \ldots, n$. Thus, the formula (3.41) can be rewritten as

$$(\theta_t - \theta_*)^T \nabla J_g(\theta_t) = 2\sum_{i=1}^n (\lambda_t^i - 1)\log \lambda_t^i, \tag{3.42}$$

where it is easy to verify that $(\lambda_t^i - 1)\log \lambda_t^i > 0$, for $i = 1, \ldots, n$. Combining (3.40), if (3.39) holds, then

$$\|\theta_{t+1} - \theta_*\|_g < \|\theta_t - \theta_*\|_g, \tag{3.43}$$

so the sequence θ_t generated by (3.32) converges to the coordinate θ_*.

In addition, we assume that $\nabla J_g(\theta'_*) = 0$ at the point $\theta'_*(\neq \theta_*)$ such that the iterative progress (3.32) terminates. Then, according to (3.42), the following equalities are valid

$$(\theta'_* - \theta_*)^T \nabla J_g(\theta'_*) = 2\sum_{i=1}^{n}(\lambda'^i_* - 1)\log \lambda'^i_* = 0, \tag{3.44}$$

where λ'_* are the eigenvalues of matrix $Q^{\frac{1}{2}}(-A^T P(\theta'_*) - P(\theta'_*)A)^{-1}Q^{\frac{1}{2}}$. Note that the second equality of (3.44) is equivalent to

$$\prod_{i=1}^{n} {\lambda'^i_*}^{\lambda'^i_* - 1} = 1, \tag{3.45}$$

so we have that $\lambda'^1_* = \lambda'^2_* = \ldots = \lambda'^n_* = 1$. That means $J_g(\theta'_*) = 0$, which is contrary to the uniqueness of the solution for the Lyapunov Eq. (3.2). Therefore, the gradient $\nabla J_g(\theta)$ exists the unique zero point θ_*. This completes the proof of Corollary 3.2. □

Based on the above discussion, we give the NGDA for the numerical solution of Eq. (3.2).

Algorithm 3.2. (NGDA) *For the coordinate* $\theta = (\theta^1, \theta^2, \ldots, \theta^N)$ *of manifold* $PD(n)$, *the natural gradient descent algorithm is given by*

1. *Set* $\theta_0 = (\theta_0^1, \theta_0^2, \ldots, \theta_0^N)$ *as the coordinate of an initial input matrix* $P(\theta_0)$ *and choose a desired tolerance* $\varepsilon_0 > 0$.
2. *Compute* $\rho(A^T P(\theta_t) + P(\theta_t)A + Q)$.
3. *If* $\rho(A^T P(\theta_t) + P(\theta_t)A + Q) < \varepsilon_0$ *then stop.*
4. *Update the coordinate vector* θ *by* $\theta_{t+1} = \theta_t - \eta_t G(\theta_t)^{-1}\nabla J_g(\theta_t)$ *and go to step* 2.

Comparing with the GDA, the NGDA needs less number of iterations since the steepest descent direction on manifold is used. However, the iterative procedure in the NGDA seems more complicated than that in the GDA, so the computational cost in the NGDA during each step may be more than that in the GDA. Especially, the Riemannian metric matrix G and its inverse are time-consuming in each iteration. Also the size of the matrix G is $N \times N$ in the NGDA, so the space for storing this large matrix reaches $O(N^2)$. In a word, the convergence speed of the NGDA is faster, at the same time the computational cost is higher.

4. Simulations

Two simulation examples are given to demonstrate the behaviors of the GDA, the NGDA and the traditional MCGA. An error criterion used in these simulations is $\rho(A^T P(\theta_t) + P(\theta_t)A + Q) < 10^{-10}$.

Example 4.1. First consider a fifth-order 2-input distillation column example with the system matrix A given by [28]

$$\begin{bmatrix} -0.1094 & 0.0628 & 0 & 0 & 0 \\ 1.3060 & -2.1320 & 0.9807 & 0 & 0 \\ 0 & 1.5950 & -3.1490 & 1.5470 & 0 \\ 0 & 0.0355 & 2.6320 & -4.2570 & 1.8550 \\ 0 & 0.0023 & 0 & 0.1636 & -0.1625 \end{bmatrix}.$$

With $Q = I$, we apply the GDA, the NGDA and the MCGA respectively to solve this equation by taking the coordinate of the 5-order identity matrix as the initial value θ_0.

Fig. 4.1 shows the step-sizes versus the iterations in the NGDA. It is found that, the iterations will reduce gradually as the step size changing from 0.1 to 0.24, while this algorithm will be divergent once the step size is bigger than 0.24. Thus this optimal step size is convenient to be obtained experimently. Also, the optimal step size corresponding to the best convergence speed in each algorithm is used in the following simulations.

Fig. 4.2 shows the comparison of the convergence speeds of the GDA, the NGDA and the MCGA. It can be found that the NGDA has the fastest convergent speed among three algorithms and needs 33 iterations to obtain the numerical solution as follows

$$\begin{bmatrix} 82.1354 & 6.4974 & 5.0137 & 3.4720 & 42.1676 \\ 6.4974 & 0.9097 & 0.6326 & 0.3871 & 3.8474 \\ 5.0137 & 0.6326 & 0.6853 & 0.3942 & 3.3242 \\ 3.4720 & 0.3871 & 0.3942 & 0.3557 & 2.4709 \\ 42.1676 & 3.8474 & 3.3242 & 2.4709 & 31.2834 \end{bmatrix}.$$

However, the MCGA and the GDA needs 53 and 104 steps to realize the same accuracy, respectively. Moreover, the three curves in Fig. 4.2 shows that this NGDA and the GDA possess the more steady convergence than the MCGA.

For this given error criterion, the elapsed time for the NGDA takes 0.57 second, and less time about 0.04 and 0.31 second are required in the case of the MCDA and GDA, respectively. We find that the NGDA is time-consuming in dealing with the metric matrix $G(\theta_t)$. However, it shows that an adaptive natural gradient algorithm [29, 30] can be attempted to reduce

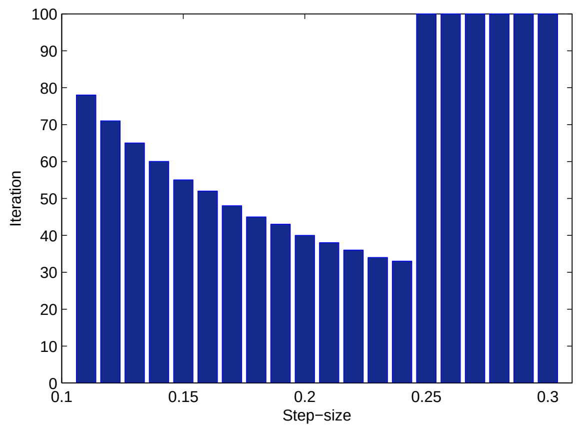

Fig. 4.1. Step-sizes versus iterations in the NGDA.

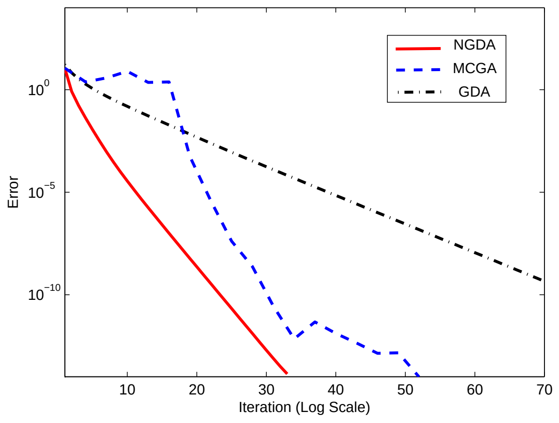

Fig. 4.2. Comparison of convergence speeds of the GDA, the NGDA and the MCGA.

computation time of this metric matrix, but such matters will not be discussed in the present paper.

Example 4.2. A 20-dimensional Lyapunov matrix equation with a small rank perturbation is also considered in our simulation. The system matrix $A \in \mathbb{R}^{20\times 20}$ satisfies $\mathrm{Re}\lambda_i(A) < 0 (1 \leq i \leq 20)$. Then we select a 2×10 random matrix B satisfying $rank(B) = 2$ and set the matrix $Q = I + B^T B$, this means that Q is a small rank perturbation of the identity matrix. Simulation indicates that the numerical solution of this equation can be obtained after 54 iterations using the NGDA, while the MGDA and the GDA need 87 and 299 steps, respectively. Therefore, the convergence speed of the NGDA is still the fastest one among three algorithms.

Also, the computing time take 24.89, 0.04 and 11.23 seconds in NGDA, the MGDA and the GDA respectively. As shown in the previous analysis, n=20 means the size of the metric matrix $G(\theta_t)$ in the NGDA is 210×210, this leads to the higher expenses in computation time.

In addition, when considering the effect with and without the small rank perturbation, simulation shows the iterative steps in the case without this perturbation are 52, 84 and 296 in the NGDA, the MCGA and the GDA respectively, and the computation time elapses 24.11, 0.03 and 10.03 seconds respectively. The convergence speeds and the computation time are no big difference, so the NGDA, the MCGA and the GDA are not sensitive to the small rank perturbation problem of the matrix Q.

5. Summary

Two algorithms, based on the gradient descent with Euclidean distance and the natural gradient descent with the geodesic distance on the manifold of positive definite matrices respectively, are developed to calculate the numerical solution for the Lyapunov matrix equation. And, the behaviors of two provided algorithms and the traditional modified conjugate gradient algorithm are compared and demonstrated by two simulation examples. It is shown that the convergence speed of the natural gradient descent algorithm is faster than both of the gradient descent algorithm and the modified conjugate gradient algorithm.

Acknowledgments. This project is supported by the National Natural Science Foundations of China (No. 61179031, No. 10932002). The authors would like to express their sincere thanks to the referees for their valuable suggestions, which greatly improved the presentation of this paper.

References

[1] H. Dai, Z.-Z. Bai, On eigenvalue bounds and iteration methods for discrete algebraic Riccati equations, *J. Comput. Math.*, **29** (2011), 341-366.

[2] Z.-Z. Bai, On Hermitian and skew-Hermitian splitting iteration methods for continuous Sylvester equations, *J. Comput. Math.*, **29** (2011), 185-198.

[3] P. Benner, Control Theory, Handbook of Linear Algebra, Chapman and Hall/CRC, Boca Raton, FL, 2006.

[4] N. Scheerlinck, P. Verboven and J. D. Stigter etc., A variance propagation algorithm for stochastic heat and mass transfer problems in food processes, *Internat. J. Numer. Methods Engrg.*, **51** (2001), 961-983.

[5] A. C. Antoulas, Approximation of Large-Scale Dynamical Systems, Advance in Design and Control 6, SIAM, Philadelphia, 2005.

[6] G. H. Golub, S. Nash and C. VanLoan, A Hessenberg-Schur method for the problem $AX + XB = C$, *Automatic Control, IEEE Transactions on*, **24** (1979), 909-913.

[7] B. N. Datta, Numerical Methods for Linear Control Systems, Elsevier Academic Press, 2004.

[8] L. Knizhnerman, V. Simoncini, Convergence analysis of the extended Krylov subspace method for the Lyapunov equation, *Numer. Math.*, **118** (2011), 567-586.

[9] B. Hashemi, M. Dehghan, The interval Lyapunov matrix equation: analytical results and an efficient numerical technique for outer estimation of the united solution set, *Mathematical and Computer Modelling*, **55** (2012), 622-633.

[10] J.-R. Li, J. White, Low-rank solution of Lyapunov equations, *SIAM J. Matrix Appl.*, **24** (2002), 260-280.

[11] B. Vandereycken, S. Vandewalle, A Riemannian optimization approach for computing low-rank solutions of Lyapunov equations, *SIAM J. Matrix Appl.*, **31** (2010), 2553-2579.

[12] K. Jbilou, ADI preconditioned Krylov methods for large Lyapunov matrix equations, *Linear. Alg. Appl.*, **432** (2010), 2473-2485.

[13] Y.-B. Deng, Z.-Z. Bai and Y.-H. Gao, Iterative orthogonal direction methods for Hermitian minimum norm solutions of two consistent matrix equations, *Numer. Linear. Alg. Appl.*, **13** (2006), 801-823.

[14] Y.-F. Su, G.-L. Chen, Iterative methods for solving linear matrix equation and linear matrix system, *Int. J. Comput. Math.*, **87** (2010), 763-774.

[15] M. Moakher, A differential geometric approach to the geometric mean of symmetric positive-definite matrices, *SIAM J. Matrix Appl.*, **26** (2005), 735-747.

[16] M. Moakher, On the averaging of symmetric positive-definite tensors, *Journal of Elasticity*, **82** (2006), 273-296.

[17] A. Schwartzman, Random ellipsoids and false discovery rates: Statistics for diffusion tensor imaging data, Ph.D Thesis, Stanford University, 2006.

[18] F. Barbaresco, Interactions between symmetric cones and information geometrics: Bruhat-tits and Siegel spaces models for high resolution autoregressive Doppler imagery, *Lecture Notes in Computer Science*, **5416** (2009), 124-163.

[19] J. Jost, Riemannian Geometry and Geometric Analysis, Third Edition. Springer, Berlin, 2002.

[20] D.-Z. Zheng, Linear System Theory, Second Edition, Tsinghua University Press, Beijing, 2005.

[21] A. Ohara, S. Amari, Differential geometric structures of stable state feedback systems with dual connections, *Kybernetika*, **30** (1994), 369-386.

[22] X.-D. Zhang, Matrix Analysis and Application, Springer, Beijing, 2004.

[23] S. Amari, S. C. Douglas, Why natural gradient, *Acoustics, Speech and Signal Processing*, **2** (1998), 1213-1216.

[24] S. Amari, Natural gradient works efficiently in learning, *Neural Computation*, **10** (1998), 251-276.

[25] Z.-Z. Zhang, H.-F. Sun and F.-W. Zhong, Natural gradient-projection algorithm for distribution control optimal control, *Applications and Methods*, **30** (2009), 495-504.

[26] Y. Tang, J. Li, Normalized natural gradient in independent component analysis, *Signal Processing*, **90** (2010), 2773-2777.

[27] S. Squartini, A. Arcangeli and F. Piazza, Stability analysis of natural gradient learning rules in overdetermined ICA, *Signal Processing*, **88** (2008), 761-766.

[28] J. Kautsky, N. K. Nichols, P. Van Dooren, Robust pole assignment in linear state feedback, *International Journal of Control*, **41** (1985), 1129-1155.

[29] H. Park, S. Amari and K. Fukumizu, Adaptive natural gradient learning algorithms for various stochastic models, *Neural Networks*, **13** (2000), 755-764.

[30] S. Amari, H. Park and K. Fukumizu, Adaptive method of realizing natural gradient learning for multilayer perceptrons, *Neural Computition*, **12** (2000), 1399-1409.