A FAST CONTINUOUS METHOD FOR THE EXTREME EIGENVALUE PROBLEM

HUAN GAO\textsuperscript{1}, ZHIBAO LI\textsuperscript{2} and HAIBIN ZHANG\textsuperscript{1}

\textsuperscript{1} College of Applied Sciences, Beijing University of Technology
Beijing 100124, China

\textsuperscript{2} School of Mathematics and Statistics, Central South University
Changsha, Hunan 410083, China

(Communicated by Liqun Qi)

Abstract. Matrix eigenvalue problems play a significant role in many areas of computational science and engineering. In this paper, we propose a fast continuous method for the extreme eigenvalue problem. We first convert such a nonconvex optimization problem into a minimization problem with concave objective function and convex constraints based on the continuous methods developed by Golub and Liao. Then, we propose a continuous method for solving such a minimization problem. To accelerate the convergence of this method, a self-adaptive step length strategy is adopted. Under mild conditions, we prove the global convergence of this method. Some preliminary numerical results are presented to verify the effectiveness of the proposed method eventually.

1. Introduction. Let $A \in \mathbb{R}^{n \times n}$ be a symmetric matrix, the well-known Real Schur Theorem states that, all eigenvalues of $A$ are real and there exist an orthonormal matrix $U = (u_1, u_2, \ldots, u_n)$ and a diagonal matrix $\Lambda$ such that

$$A = U \Lambda U^T,$$

where

$$\Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n), \quad \lambda_1 = \cdots = \lambda_s < \lambda_{s+1} \leq \cdots \leq \lambda_n,$$

and $1 \leq s \leq n$. Then the extreme eigenvalue problem is to find the smallest eigenvalue $\lambda_1$ and an associated eigenvector $x \in \mathcal{X}$, where $\mathcal{X}$ is defined as

$$\mathcal{X} = \{ x \in \mathbb{R}^n \mid A x = \lambda_1 x, \quad x^T x = 1 \}.$$

The eigenvalue problem is a classical but very important problem in various disciplines of science and engineering \cite{21}. And the extreme eigenvalue and the associated eigenvector are used in many engineering and science applications such as control theory, vibration analysis, electric circuits, etc.

There are a lot of efficient numerical methods for the solving of extreme eigenvalue problem. For instance, the power method, QR iteration and Lanczos method
mentioned in the classical monograph by Golub and Van Loan [10], the conjugate gradient type methods [18], the filter Lanczos method [6], the Barzilai-Borwein-like methods [8], and so on. Besides these methods, some continuous methods are also powerful for solving the extreme eigenvalue problem. In [2, 3], some ordinary differential equation (ODE) systems were introduced for many iterative processes in solving application problems. After the successful application of the neural network approach of Hopfield in optimization [14, 15, 16], Cichocki and Unbehauen introduced a neural network model for computing particular eigenvalue and the corresponding eigenvector [4]. For the problem of finding the minimal (maximal) eigenvalue of a real symmetric matrix, they first converted it into a constrained optimization problem. Then a neural network model was applied to solve this constrained problem by using either the penalty method or the Lagrange multiplier method. The optimization problems formulated in [4] are not easy to solve, which, as a consequence, makes the application of their methods quite limited.

In [11], Golub and Liao proposed a new continuous method for the extreme and interior eigenvalue problems, where they also first converted the eigenvalue problem into some optimization problems. The main difference between their method and that of Cichocki and Unbehauen [4] is that the optimization problem in [11] is to minimize a strictly concave function over a unit ball, which is very easy to solve. Golub and Liao also proposed a continuous method for the resulting optimization problem in [11], which consists of a merit function and an ordinary differential equation, and proved the convergence of each ODE solution for any starting point. And in recent, this kind of continuous method has been introduced to solve symmetric generalized eigenvalue problems and linear programming successfully [9, 17].

In this paper, our first purpose is to simplify the proof of the works by Golub and Liao in [11], and our second purpose is to solve the resulting dynamic system by using a self-adaptive method. In detail, we introduce a parameter in the dynamic system, which plays a key role in the numerical methods. We then introduce a self-adaptive strategy to find the suitable parameter, along with the progress of the iterative method, such as Euler method. Our preliminary numerical results indicate that this strategy can enhance the performance of the proposed method greatly.

The rest of this paper is organized as follows. In Section 2, we present our problem formally and summarize some necessary preliminary results. In Section 3, we construct the dynamic system for solving the eigenvalue problem and prove the convergence of each ODE solution for any starting point. In Section 4, we propose a self-adaptive strategy in choosing the suitable parameter and prove the global convergence of our self-adaptive method. In Section 5, we conduct some numerical experiments to verify the propose method. Then we complete this paper by giving some concluding remarks in Section 6.

2. Problem formulation. Denote
\[ S = \{ (x, \lambda) \in \mathbb{R}^n \times \mathbb{R} \mid Ax = \lambda x, \ x^T x = 1 \}, \]  
(3)
then the dominant nonnegative eigenvalue problem is to find \((x^*, \lambda^*) \in S\), such that
\[ \lambda^* \leq \lambda, \ \forall (\lambda, x) \in S. \]

Therefore, this problem can be formulated as an optimization problem
\[
\min_{x \in \mathbb{R}^n} \begin{cases} 
 x^T Ax \\
 s.t. \ x^T x = 1. 
\end{cases}
\]  
(4)
A FAST CONTINUOUS METHOD FOR THE EXTREME EIGENVALUE PROBLEM 1589

From the constraint $x^T x = 1$, it is noted that the above problem is equivalent to

$$\begin{align*}
\min_{x \in \mathbb{R}^n} & \quad x^T A x + \alpha x^T x \\
\text{s.t.} & \quad x^T x = 1,
\end{align*}$$

(5)

where $\alpha$ can be an arbitrarily real constant. By denoting $B = A + \alpha I$, the problem (5) can be rewritten as

$$\begin{align*}
\min_{x \in \mathbb{R}^n} & \quad x^T B x \\
\text{s.t.} & \quad x^T x = 1,
\end{align*}$$

which has the same form as the original problem (1). Since $\alpha$ is arbitrary, we can always choose $\alpha$ big enough, such that the matrix $B$ is always a real symmetric positive definite matrix. Thus, in the following, without loss of generality, we assume that matrix $A$ is a real symmetric positive definite matrix.

The feasible region of problem (4) is a sphere, which is not a convex region. This makes the problem difficult to solve. However, it can be converted into a new optimization problem with a convex feasible set. In detail, we first choose a constant $c$ such that

$$c \geq \lambda_n + 1.$$  

(6)

Since $A$ is a symmetric matrix, from the Corollary 2.3.2 in [10], we have

$$\max_{1 \leq i \leq n} |\lambda_i| = \|A\|_2 \leq \|A\|_1.$$  

(7)

Thus, we can choose $c = \|A\|_1 + 1$ (which was used as the default value in the numerical experiments of Golub and Liao [10] and will also be adopted as the default value in our numerical computation). Then we can establish the following optimization problem

$$\begin{align*}
\min_{x \in \mathbb{R}^n} & \quad x^T A x - c x^T x \\
\text{s.t.} & \quad x^T x \leq 1,
\end{align*}$$

(8)

which is different from problem (1) in that the objective function is quadratic and strictly concave but the constraint is a simple ball constraint. Since the feasible region of problem (8) is a closed convex set. As a consequence, the problem (8) is easier to solve than (1).

Although the optimization problem (8) is not belong to the traditional convex optimization problems, there still has important property of the relationship between the local minimizer and global minimizer of the problem (8). The following lemma from [11] will play a key role in the next dynamic system and convergence analysis.

**Lemma 2.1.** Every local minimizer of the problem (8) is also the global minimizer of it.

3. **The dynamic system and convergence.** From the above extreme eigenvalue problem configuration, we have established an equivalent optimization problem (8) for our eigenvalue problem. However, that optimization problem is not a convex optimization problem. The objective function is strictly concave although the feasible region is convex, then lots of effective approaches for convex optimization problem can not be applied directly.

With the help of Lemma 2.1 it is guaranteed that the solution of problem (8) can be obtained by finding one local minimizer of it. Moreover, it is well known that
the first-order optimal condition holds at every local minimizer of a minimization problem:

$$\min \{ f(x) | x \in \Omega \},$$  \hspace{1cm} (9)

where \( \emptyset \neq \Omega \subset \mathbb{R}^n \) is a convex set, \( f : \mathbb{R}^n \to \mathbb{R} \) is a continuous differentiable function defined on \( \Omega \). And the first-order optimal condition of problem (9) at \( x^* \) is that any feasible direction is not the descent direction. By denoting

$$S_f(x) = \{ d \in \mathbb{R}^n | d = x' - x, x', x \in \Omega \}$$  \hspace{1cm} and

$$S_d(x) = \{ d \in \mathbb{R}^n | d^T \nabla f(x) < 0 \}$$

as the sets of feasible directions and descent directions at point \( x \in \Omega \), respectively.

The optimal condition of problem (9) at \( x^* \) in variational inequality form is

$$x^* \in \Omega \quad \text{and} \quad S_f(x^*) \cap S_d(x^*) = \emptyset,$$  \hspace{1cm} (10)

which is the variational inequality (VI(\( \Omega, \nabla f \))): Find \( x^* \in \Omega \), such that

$$(x - x^*)^T \nabla f(x^*) \geq 0, \quad \forall x \in \Omega.$$  \hspace{1cm} (11)

In the next, let us introduce the projection operator \( P_{\Omega}(\cdot) : \mathbb{R}^n \to \Omega \), which is defined as

$$P_{\Omega}(x) = \arg \min_{z \in \Omega} \| z - x \|_2, \quad x \in \mathbb{R}^n.$$  \hspace{1cm} (12)

One of the fundamental property of \( P_{\Omega}(\cdot) \) is that for each \( x \in \mathbb{R}^n \), \( P_{\Omega}(x) \) is the unique vector \( \bar{x} \in \Omega \) satisfying the inequality:

$$(y - \bar{x})^T (\bar{x} - x) \geq 0, \quad \forall y \in \Omega.$$  \hspace{1cm} (13)

Then we have the following proposition [5].

**Lemma 3.1.** The solution of VI(\( \Omega, \nabla f \)) defined in (11) is equivalent to find a zero point of

$$e(x) := x - P_{\Omega}(x - \nabla f(x)).$$  \hspace{1cm} (14)

**Proof.** The defining inequality for the (11) is

$$(x - x^*)^T \nabla f(x^*) \geq 0, \quad \forall x \in \Omega,$$

which can be rewritten as

$$(x - x^*)^T (x^* - (x^* - \nabla f(x^*))) \geq 0, \quad \forall x \in \Omega.$$  

From the property (13), the above inequality is equivalent to

$$x^* = P_{\Omega}(x^* - \nabla f(x^*)),$$

or equivalently \( e(x^*) = 0 \).

Moreover, it can be seen that the solution of the minimization problem (9) is invariant to the objective function \( f(x) \) scaled by any positive constant \( \beta > 0 \), which means that the zero point of \( e(x, \beta) := x - P_{\Omega}(x - \beta \nabla f(x)) \) is also invariant to the parameter \( \beta \).

Based on the above discussion, we can solve the optimization problem (8) by finding a zero point of the following equation

$$e(x, \beta) := x - P_{\Omega}(x - \beta \nabla f(x)),$$  \hspace{1cm} (15)

with \( \beta > 0 \) is an arbitrary constant and \( f(x) = x^T A x - c x^T x \).

In the following, we introduce the continuous method to find a zero point of (15) for solving the extreme eigenvalue problem based on the above preparation
works. Generally, a continuous method for an optimization problem consists of two components: a merit function (bounded below) and a dynamical system, where the merit function must be monotonically non-increasing along the dynamical system.

Taking the objective function as the merit function:

\[ f(x) = x^T Ax - cx^T x, \]  

the dynamical system can be constructed as

\[ \frac{dx(t)}{dt} = -(x - P_\Omega(x - \beta \nabla f(x))), \]  

where \( \beta > 0 \) is an arbitrary parameter.

Note that the dynamic system (17) is a simple generalization of that in [11] in the sense that we introduce a parameter \( \beta \), and the dynamic system in [11] is a special case for \( \beta = 1 \). From the numerical point of view, this simple modification enables us to use a self-adaptive strategy during solving the dynamic system (17), which is much faster than just using a fixed parameter.

Now, we give a simple proof of the proposition that any limit point of the dynamic system is a stationary point of the original problem. The following lemma shows that the limit point set of the dynamic system (17) is nonempty.

**Lemma 3.2.** Suppose that the merit function \( f(x) \) is differentiable, the gradient \( \nabla f(x) \) satisfies the Lipschitz condition, \( \Omega \) is a convex closed bounded set, and the initial condition point \( x_0 \in \Omega \). Then the trajectory of process (17) has a nonempty set of limit points.

**Proof.** Since the right-hand-side of (17) is continuous in \( \mathbb{R}^n \), the Cauchy-Peano theorem ensures that there exits a solution \( x(t) \) of the dynamical system (17) with \( x(t=t_0) = x_0 \).

**Theorem 3.3.** For any \( x(t=t_0) = x_0 \in \mathbb{R}^n \) there exits a unique solution \( x(t) \) of the dynamical system (17) with \( x(t=t_0) = x_0 \) in \( [t_0, +\infty] \).

**Proof.** From Lemma 3.2, we note that there exits a solution \( x(t) \) of the dynamical system (17) with \( x(t=t_0) = x_0 \). For this solution \( x(t) \), we define

\[ E(x(t)) = \|x(t) - P_\Omega(x(t))\|^2, \]

the square of the distance of \( x(t) \) to \( \Omega \). It follows from the Theorem 1.7 in [1] that

\[ \frac{dE(x(t))}{dt} = -2(x(t) - P_\Omega(x(t)))^T e(x(t), \beta). \]

Due to that \( \Omega \) is a closed convex set, from the fundamental inequality of projection [13], we have

\[ (z - P_\Omega(z))^T (y - P_\Omega(z)) \leq 0, \quad \forall y \in \Omega, \forall z \in \mathbb{R}^n. \]  

(18)

By setting \( z := x \) and \( y := P_\Omega(x - \beta \nabla f(x)) \) in the above inequality, we obtain

\[ (x - P_\Omega(x))^T (P_\Omega(x - \beta \nabla f(x)) - P_\Omega(x)) \leq 0, \]

i.e.,

\[ (x - P_\Omega(x))^T e(x, \beta) \geq \|x - P_\Omega(x)\|^2. \]

Thus,

\[ \frac{dE(x(t))}{dt} = -2(x(t) - P_\Omega(x(t)))^T e(x(t), \beta) \leq -2\|x - P_\Omega(x)\|^2 < 0, \text{ if } x \notin \Omega. \]
This means that $E(x(t))$ is monotonically nonincreasing in $t$. Therefore, we have
\[
\|e(x, \beta)\| = \|x - P_{\Omega}(x - \beta \nabla f(x))\| \\
\leq \|x - P_{\Omega}(x)\| + \|P_{\Omega}(x) - P_{\Omega}(x - \beta \nabla f(x))\| \\
\leq \|x_0 - P_{\Omega}(x_0)\| + \|P_{\Omega}(x) - P_{\Omega}(x - \beta \nabla f(x))\| \\
\leq \|x_0\| + \|P_{\Omega}(x_0)\| + \|P_{\Omega}(x)\| + \|P_{\Omega}(x - \beta \nabla f(x))\| \\
\leq \|x_0\| + 3,
\]
where the last inequality follows from the fact that
\[
\|P_{\Omega}(z)\| \leq 1, \quad \forall z \in R^n.
\]
Inequality (19) indicates that the right-hand-side of (17) is bounded for any given $x_0$. Again the Cauchy-Peano theorem ensures that the solution $x(t)$ exists in $[t_0, +\infty]$.

Since $\Omega$ is a closed convex set, from the nonexpansive property of the projection operator, we have
\[
\|P_{\Omega}(u) - P_{\Omega}(v)\| \leq \|u - v\|, \quad \forall u, v \in R^n.
\]
Then,
\[
\|e(x, \beta) - e(y, \beta)\| = \|x - P_{\Omega}(x - \beta \nabla f(x)) - y + P_{\Omega}(y - \beta \nabla f(y))\| \\
\leq \|x - y\| + \|P_{\Omega}(x - \beta \nabla f(x)) - P_{\Omega}(y - \beta \nabla f(y))\| \\
\leq 2\|x - y\| + \beta \|\nabla f(x) - \nabla f(y)\| \\
\leq (2 + 2\beta\|A\| + 2\beta c)\|x - y\|, \quad \forall x, y \in R^n,
\]
which implies that $e(x, \beta)$ in (14) is Lipschitz continuous in $R^n$. From the Picard-Lindelöf theorem, the solution of the dynamical system (17) is unique. \hfill \Box

The result of Theorem 3.3 indicates that the dynamical system (17) is well defined. In the proof of Theorem 3.3 we can see that if $x_0 \notin \Omega$, the solution $x(t)$ of (17) will move towards the feasible region. And if $x_0 \in \Omega$, then the solution $x(t)$ of (17) will stay in $\Omega$ from then on. Now we verify that the limit of the solution of (17) is the solution of the optimization problem (8). For this purpose, we need to introduce the following lemma beforehand.

**Lemma 3.4.** Suppose the scalar function $h(t)$ is differentiable on $[t_0, T]$ with $h(t_0) = 0$. If there exists an $M > 0$ such that
\[
\left|\frac{dh}{dt}\right| \leq M|h(t)|, \quad \forall t \in [t_0, T],
\]
then
\[
h(t) = 0, \quad \forall t \in [t_0, T].
\]

**Proof.** See Exercise 26, p. 119 in [19]. \hfill \Box

As noted in [11], it is easy to see that $T$ in the above lemma can be extended to $+\infty$. Now we prove the following convergence results for the solution of (17).

**Theorem 3.5.** For any $x_0 \in \Omega$, let $x(t)$ be the solution of dynamical system (17) with $x(t = t_0) = x_0$. Then, we have

(i) if $e(x_0, \beta) = 0$, $x(t) \equiv x_0$, $\forall t \geq t_0$;

(ii) if $e(x_0, \beta) \neq 0$, then $\lim_{t \to +\infty} e(x(t), \beta) = 0$. 

Proof. (i) Let
\[ h(t) = \| e(x(t), \beta) \|^2, \]
by using the Theorem 1.7 in [1] again, we have
\[ \left| \frac{dh}{dt} \right| = 2 \left| e(x(t), \beta)^T \frac{dx}{dt} \right| = 2|h(t)|. \]
Then the assumption of Lemma 3.4 is satisfied with \( M = 2 \) and the assertion follows immediately.

(ii) Since \( x_0 \in \Omega \), Theorem 3.1 ensures that \( x \in \Omega, \forall t \geq t_0 \). By taking \( y = x - \nabla f(x) \) in (18), we have
\[ (e(x, \beta) - \beta \nabla f(x))^T e(x, \beta) \leq 0. \]
From (14), (16), (17) and the above inequality, we have
\[ \frac{df(x)}{dt} = -(\nabla f(x))^T e(x, \beta) \leq -\|e(x, \beta)\|^2/\beta \leq 0, \quad \forall t \geq t_0. \]
From the LaSalle invariant set theorem (Theorem 3.4 in [20]) and the above inequality, we obtain
\[ \lim_{t \to \infty} e(x(t), \beta) = 0. \]
This completes the proof. \( \square \)

With the convergence results of the dynamical system, we can get the solution of this dynamical system by directly using some numerical tools. For instance, the ODE solvers provided by Matlab, ode45, as done by Golub and Liao [11]. Since we are going to incorporate a self-adaptive parameter selection strategy into the algorithm, we can solve this dynamical system adaptively by using some classical numerical methods for ODE, such as Euler method.

4. The fast algorithm. From the above discussion, in order to solve the original optimization problem [1], the continuous method can be used by setting the objective function [16] as the merit function and [17] as a dynamic system. The efficiency of numerical methods for tracking dynamic system [17] depends heavily on the choice of \( \beta \). However, in [11], Golub and Liao just fix \( \beta = 1 \). In fact, from our numerical experience, an optimal constant \( \beta \) is usually difficult to find: it varies from one problem to another. The freedom in setting \( \beta \) plays an important role in the numerical method for solving the dynamic system. We thus develop a fast algorithm by incorporating a self-adaptive strategy to select a suitable parameter \( \beta \). That is, at each iteration, we need to solve the dynamic system [17] adaptively.

In detail, we use the Euler method to solve the dynamical system. At time \( t \), we generate the next approximation to the optimal solution via the following iteration
\[ x(t + h) = x(t) - he(x(t), \beta(t)), \quad \text{(21)} \]
where \( h \) is the time step-length and \( \beta(t) \) is the suitable parameter selected self-adaptively.

The above iteration formula (21) is a simple but effect strategy, it has been successfully used by Fukushima [7] for solving variational inequality problems with strongly monotone mappings. It was also employed by He [13] for solving the monotone linear complementarity problems and convex quadratic problems. Moreover, it was extended by Han and Lo [12] to solve the nonlinear variational inequality problems with co-coercive mappings. We are now in the position to describe our algorithm for solving the dynamic system [17] formally.
Algorithm 4.1. A self-adaptive method.

**S0.** Given a nonnegative sequence $\tau(t)$ with $\sum_{t=t_0}^{\infty} \tau(t) < +\infty$, chosen an arbitrary initial point $x(t_0) \in \Omega$, and positive parameters $\epsilon > 0$, $h \in (0, 1)$, $\mu \in (0, 1)$ and $\gamma = \gamma^0 > 0$. Set $t := t_0$.

**S1.** If $\|e(x(t))\| \leq \epsilon$, stop; else, go to **S2**.

**S2.** (Self-adaptive procedure) Find the smallest nonnegative integer $l_k$, such that

$$x(t + h) = x(t) - he(x(t), \beta(t + h))$$

satisfies

$$f(x(t + h)) \leq f(x(t)) - h\beta(t + h)\nabla f(x(t))^T e(x(t), \beta(t + h)).$$

**S3.** (Selection of $\gamma$) If

$$f(x(t + h)) < f(x(t)) - 0.7h\beta(t + h)\nabla f(x(t))^T e(x(t), \beta(t + h)),$$

set $\gamma = (1 + \tau(t + h))\beta(t + h)$, otherwise, set $\gamma = \beta(t)$.

**S4.** Set $t := t + h$, and go to **S1**.

The following lemma shows that at each iteration, the self-adaptive procedure in selecting the suitable parameter terminates in finite steps. Consequently, the whole algorithm is well defined.

**Lemma 4.2.** Suppose that $\tau(t) \geq 0$ for all $t$, and

$$\sum_{t=t_0}^{\infty} \tau(t) < +\infty.$$  

For any $t$ in the above Algorithm 4.1, the procedure of searching $\beta(t + h)$ (self-adaptive procedure, **S2**) will be terminated in finite steps. Furthermore, there exists a positive real number denoted by $\beta_{\min}$, such that $\beta(t) \geq \beta_{\min} > 0$ for all $t \geq 0$.

**Proof.** By **S3** in the Algorithm 4.1, we derive that $\beta(t + h) \leq (1 + \tau(t))\beta(t)$ for all $t$. Since $\sum_{t=0}^{\infty} \tau(t) < +\infty$, it follows that $\prod_{t=0}^{\infty} (1 + \tau(t)) < +\infty$. This implies that there is a positive constant $K > 0$ such that $\gamma \leq K$ and $\beta(t) \leq K$ for all $t$. Because $f$ is uniformly concave with a constant modulus 1 on $\Omega$ ($c \geq \lambda_n + 1$), it yields

$$f(x - he(x(t), \beta(t + h))) \leq f(x(t)) - h\nabla f(x(t))^T e(x(t), \beta(t + h)).$$

By setting $y := x(t)$ and $z := x(t) - \beta(t + h)\nabla f(x(t + h))$ in (18), we have

$$(e(x(t), \beta(t + h)) - \beta(t + h)\nabla f(x(t + h)))^T e(x(t), \beta(t + h)) \leq 0,$$

i.e.,

$$\beta(t + h)\nabla f(x(t + h))^T e(x(t), \beta(t + h)) \geq \|e(x(t), \beta(t + h))\|^2 \geq 0.$$  

By combining the above inequality and (26) together, we obtain

$$f(x - he(x(t), \beta(t + h))) \leq f(x(t)) - h\beta(t + h)\nabla f(x(t))^T e(x(t), \beta(t + h)),$$

it provided that $\beta(t + h) \leq 1$. Based on the facts that $\gamma \leq K$ and $\lim_{n \to \infty} \mu^n = 0$, we derive that the self-adaptive procedure in each iteration will be terminated in finite
steps. The parameter \( l_k \) in the Algorithm 4.1 is the minimum nonnegative integer fulfilling the condition (23), and this means that

\[
\beta(t + h) \geq \beta_{\text{min}} := \min\{\gamma^0, \mu\} > 0, \quad \forall t.
\]

(27)

The proof is completed. \( \square \)

Lemma 4.2 indicates that the whole Algorithm 4.1 is well-defined. Thus, it either generates an approximate solution, or generates an infinite iterative sequence. In the following, we assume that the stopping checking parameter \( \epsilon \) is zero and the algorithm generates an infinite sequence of iterations.

Now, we prove that any cluster point of the Algorithm 4.1 is a local minimizer of the problem (8), which, by invoking Lemma 2.1, is a global minimizer of (4), a solution of the original problem.

**Theorem 4.3.** Suppose that the assumptions in Lemma 4.2 hold. Then, any cluster point of the generated sequence is a stationary point of (4).

**Proof.** Since \( x(t_0) \in \Omega \) and \( h \in (0, 1) \), we have that \( x(t) \in \Omega \) for all \( t \geq t_0 \). Due to the compact of \( \Omega \) and the continuity of \( f \), \( f(x(t)) \) is bounded for all \( t \geq t_0 \). That is, there exists a constant \( C \), such that

\[
|f(x(t))| \leq C, \quad \forall t \geq t_0.
\]

(28)

It then follows from (26) that

\[
h\nabla f(x(t))^T e(x(t), \beta(t + h)) \leq f(x(t)) - f(x(t) - he(x(t), \beta(t + h))).
\]

(29)

By using (18), we have

\[
\nabla f(x(t))^T e(x(t), \beta(t + h)) \geq \|e(x(t), \beta(t + h))\|^2.
\]

For any \( \beta' \geq \beta > 0 \) and for arbitrary but fixed \( x \), we get

\[
\|e(x, \beta')\| \geq \|e(x, \beta)\|
\]

and

\[
\frac{\|e(x, \beta')\|}{\beta} \leq \frac{\|e(x, \beta)\|}{\beta}.
\]

Then,

\[
\|e(x(t), \beta(t + h))\| \geq \frac{1}{1 + \tau(t + h)} \|e(x(t), \beta(t))\|.
\]

(30)

By combining (28)–(30) together, we obtain

\[
\frac{1}{1 + \tau(t + h)} \|e(x(t), \beta(t))\|^2 \leq f(x(t)) - f(x(t) - he(x(t), \beta(t + h))).
\]

Summing both sides of the above inequality, we have

\[
\sum_{t = t_0}^{+\infty} \frac{h}{(1 + \tau(t + h))^2} \|e(x(t), \beta(t))\|^2 \leq f(x(t_0)) + C \leq 2C.
\]

Since \( h \in (0, 1) \) and

\[
\sum_{t = t_0}^{+\infty} (1 + \tau(t + h))^2 < +\infty,
\]

we get

\[
\lim_{t \to \infty} e(x(t)) = 0.
\]

This completes the proof. \( \square \)
5. Numerical experiments. In this section, we conduct our proposed method on two examples to verify the effectiveness of the proposed method on the extreme eigenvalue problem, and compare it with the ODE solver used by Golub and Liao [11]. The stopping criterion used in our experiments is
\[ \| e(x) \|_\infty \leq \epsilon, \]
where \( \epsilon \) is a preset value, and we use \( \epsilon = 10^{-6} \) in all our tests. All codes were run in Matlab platform on a PC with Intel(R), Pentium(R) 4 CPU with 3.2GHz. The ODE solver used is ode45, a non-stiff medium order method, where we set RelTol=10^{-6} and RelTol=10^{-9} in all our runs. Our examples are constructed in the following ways.

**Example 1.** We construct the first example in a similar way as [11]. That is, we construct the matrix in the following three steps:

1. Select a diagonal matrix
   \[ \Lambda = \text{diag} (2, 3, 4, 1e-2, 1, \cdots, 1) \in \mathbb{R}^{n \times n}. \]
2. Generate a random matrix \( B \in \mathbb{R}^{n \times n} \), and get its QR decomposition by
   \[ [Q, R] = qr(B). \]
3. Set \( A = Q^T \Lambda Q \), which is a real symmetric positive definite matrix.

We select \( u=\text{ones}(n, 1) \), and \( x_0 = u/\|u\| \) as the initial value. We choose \( c = 5 \).

In our algorithm, we choose \( t_0 = 0, \gamma = \gamma^0 = 1, \mu = 0.6 \) and \( h = 0.05 \), and the nonnegative sequence \( \{ \tau(t) \} \) assignment is to be \( \tau(t) = 0.5, t \leq 720s \). We compare our method with that of Golub and Liao [11], with \( n \) varying from 100 to 5000.

Figure 1 shows the trajectories of some entries of the solutions of dynamic system for problem size \( n = 1000 \) and \( n = 5000 \), where the trajectories ‘x’ and ‘y’ denote the trajectories of solutions obtained by the Golub and Liao’s method and our proposed method, respectively.

From the results shown in Figure 1 we can see that all the three trajectories of ‘x’ and ‘y’ (the 200-th, 700-th and 800-th entries of solution for the case of \( n = 1000 \), the 1500-th, 3000-th and 4100-th entries of solution for the case of \( n = 5000 \)) are converged to the same values. This means that both the two methods can find the correct solution of dynamic system. Moreover, our method takes a relatively less time to make the dynamic system to be stable. In detail, we summarize the computation time and the number of iterations of our proposed method in Table 1 and do comparison with the ODE solver. From the results shown in Table 1 we can see that the proposed method is about 10 times faster than the ODE solver for all the testing scenarios.

**Example 2.** In the second example, we construct a new extreme eigenvalue problem with the eigenvalue different from each other, and the diagonal matrix is selected as
\[ \Lambda = \text{diag} (1, 1/2, 3/4, 4/5, 5/6, \cdots, n-1/n) \in \mathbb{R}^{n \times n}. \]

| n     | 100   | 500   | 1000  | 2000  | 3000  | 4000  | 5000  |
|-------|-------|-------|-------|-------|-------|-------|-------|
| ODE45(Sec.) | 0.328 | 5.297 | 17.2199 | 86.188 | 130.016 | 389.016 | 642.234 |
| Iterations | 31    | 55    | 43    | 51    | 39    | 55    | 60    |
| Time (Sec.) | 0.016 | 0.531 | 1.484 | 6.734 | 11.594 | 29.125 | 48.094 |

**Table 1. Results for Example 1.**
Figure 1. Sketch three trajectories of two different orders.

All parameters are set the same as the previous Example 1. We also test our proposed method and the ODE solver for comparison.

Illustration of the trajectories of some entries of solutions (the 200-th, 400-th and 900-th entries of solution for the case of $n = 1000$, the 1000-th, 2000-th and 4000-th entries of solution for the case of $n = 5000$) by using the two methods are plotted in Figure 2. It can be seen that both methods are effect, whereas the convergence speed of our proposed method is fast. We also summarize the detail results of the computation time and the number of iterations of the proposed method in Table 2. From the results, we can see that our proposed method is also about 10 times faster than the ODE solver for all the testing scenarios.

6. Conclusions. In this paper, we first simplify the proof of the works by Golub and Liao in [11] on the continuous method for the extreme eigenvalue problem. Then we propose a fast continuous method for solving the large-scale extreme eigenvalue problem. To accelerate the convergence speed, we introduce a self-adaptive strategy
in design our new method. The numerical results indicate that our proposed method is efficient, it is about 10 times faster than the ODE solver.

Acknowledgments. The authors would like to thank Prof. Deren Han and Dr. Hongjin He for the meaningful advice and discussions on this paper. The authors also take this opportunity to thank the editor and the anonymous referees for their valuable comments and suggestions, which led to a much better version of this paper.
REFERENCES

[1] A. Auslender, *Optimisation: Méthodes Numériques*, Masson, Paris, 1976.
[2] M. T. Chu, On the continuous realization of iterative processes *SIAM Review*, **30** (1988), 375–387.
[3] M. T. Chu, Matrix differential equations: a continuous realization process for linear algebra problems. *Nonlinear Analysis, Theory, Methods and Applications*, **18** (1992), 1125–1146.
[4] A. Cichocki and R. Unbehauen, Neural networks for computing eigenvalues and eigenvectors *Biological Cybernetics*, **68** (1992), 155–164.
[5] F. Facchinei and J. S. Pang, *Finite-Dimensional Variational Inequalities and Complementarity Problems*, Springer-Verlag, New York, 2003.
[6] H. R. Fang and Y. Saad, A filtered Lanczos procedure for extreme and interior eigenvalue problems, *SIAM Journal on Scientific Computing*, **34** (2012), A2220–A2246.
[7] M. Fukushima, Equivalent differentiable optimization problems and descent methods for asymmetric variational inequality problems *Mathematical Programming*, **53** (1992), 99–110.
[8] H. Gao, Y. H. Dai and X. J. Tong, Barzilai-Borwein-like methods for the extreme eigenvalue problem *Journal of Industrial and Management Optimization*, **11** (2015), 999–1019.
[9] X. B. Gao, G. H. Golub and L. Z. Liao, Continuous methods for symmetric generalized eigenvalue problems *Linear Algebra and its Applications*, **428** (2008), 676–696.
[10] G. H. Golub and C. F. Van Loan, *Matrix Computation*, 3rd ed., The John Hopkins University Press, 1996.
[11] G. H. Golub and L. Z. Liao, Continuous methods for extreme and interior eigenvalue problems *Linear Algebra and its Applications*, **415** (2006), 31–51.
[12] D. R. Han and H. K. Lo, Solving non-additive traffic assignment problems: A descent method for co-coercive variational inequalities *European Journal of Operational Research*, **159** (2004), 529–544.
[13] B. He, A projection and contraction method for a class of linear complementarity problems and its application in convex quadratic programming *Applied Mathematics and Optimization*, **25** (1992), 247–262.
[14] J. J. Hopfield, Neural networks and physical systems with emergent collective computational ability *Proc. National Academy of Sciences of the United States of America*, **79** (1982), 2554–2558.
[15] J. J. Hopfield, Neurons with graded response have collective computational properties like those of two-state neurons *Proc. National Academy of Sciences of the United States of America*, **81** (1984), 3088–3092.
[16] J. J. Hopfield and D. W. Tank, Neural computation of decisions in optimization problems, *Biological Cybernetics*, **52** (1985), 141–152.
[17] L. Z. Liao, A Study of the Dual Affine Scaling Continuous Trajectories for Linear Programming *Journal of Optimization Theory and Applications*, **163** (2014), 548–568.
[18] E. E. Ovtchinnikov, Jacobi correction equation, line search, and conjugate gradients in Hermitian eigenvalue computation I: Computing an extreme eigenvalue *SIAM Journal on Numerical Analysis*, **46** (2014), 2567–2592.
[19] W. Rudin, *Principles of Mathematical Analysis*, 3rd ed., McGraw-Hill Book Company, 1976.
[20] J. J. E. Slotine and W. Li, *Applied Nonlinear Control*, Prentice Hall, New Jersey, 1991.
[21] H. A. van der Vorst and G. H. Golub, 150 years old and still alive: Eigenproblems, In: I.S. Duff, Editor, *The State of the Art in Numerical Analysis*, **63** (1997), 93–119.
[22] T. Zhu and Z. G. Yu, A simple proof for some important properties of the projection mapping *Mathematical Inequalities and Applications*, **7** (2004), 453–456.

Received July 2015; revised April 2016.

E-mail address: huanhuan1382138163.com
E-mail address: zbl1@lsec.cc.ac.cn
E-mail address: zhanghaibin@bjut.edu.cn