NUMERICAL LINEAR ALGEBRA
WITH APPLICATIONS

Two-step Newton type methods for solving inverse eigenvalue problems

Journal:	*Numerical Linear Algebra with Applications*
Manuscript ID	NLA-17-0038.R1
Wiley - Manuscript type:	Research Article
Date Submitted by the Author:	03-Nov-2017
Complete List of Authors:	Xiao Shan, Chen; South China Normal University, School of Mathematical Sciences Chao Tao, Wen; South China Normal University Sun, Hai-wei; University of Macau, Mathematics;
Keyword:	Inverse eigenvalue problem, two-step Newton type method, super quadratically convergent

TWO-STEP NEWTON TYPE METHODS FOR SOLVING INVERSE EIGENVALUE PROBLEMS*

XIAO SHAN CHEN †, CHAO TAO WEN†, AND HAI-WEI SUN ‡

Abstract. In this paper, we apply the two-step Newton method to solve inverse eigenvalue problems, including exact Newton, Newton-like and inexact Newton-like versions. Our results show that both two-step Newton and two-step Newton-like methods converge cubically, and the two-step inexact Newton-like method is super quadratically convergent. Numerical implementations demonstrate the effectiveness of new algorithms.

Key words. Inverse eigenvalue problem, two-step Newton type method, super quadratically convergent

AMS subject classifications. 65F15, 65F99

1. Introduction. Inverse eigenvalue problems arise in various of scientific computing and engineering applications , see for instance the pole assignment problem[5], the inverse Toeplitz eigenvalue problem [6, 18], the inverse Sturm-Liouville problem [1, 2], and also problems in applied mechanics and structure design [10, 11]. In many applications, the problem size n can be large, for example, large Toeplitz eigenvalue problems and discrete inverse Sturm-Liouville problems. Our goal in this paper is to investigate two-step Newton type methods for solving inverse eigenvalue problems.

Let $\{A_i\}_{i=0}^n$ be $n+1$ real symmetric n-by-n matrices. For any $\mathbf{c} = (c_1, \ldots, c_n)^T \in \mathbb{R}^n$, let

$$A(\mathbf{c}) \equiv A_0 + \sum_{i=1}^{n} c_i A_i, \tag{1.1}$$

and denote the eigenvalues of $A(\mathbf{c})$ by $\{\lambda_i(\mathbf{c})\}_{i=1}^n$, where $\lambda_1(\mathbf{c}) \leq \cdots \leq \lambda_n(\mathbf{c})$. The inverse eigenvalue problem (IEP) is defined as follows: For given n real numbers $\{\lambda_i^*\}_{i=1}^n$ with $\lambda_1^* \leq \cdots \leq \lambda_n^*$, find a vector $\mathbf{c}^* = (c_1^*, \ldots, c_n^*)^T \in \mathbb{R}^n$ such that $\lambda_i(\mathbf{c}^*) = \lambda_i^*$ for $i = 1, \ldots, n$. There has been a lot of literature on various aspects of existence theory and numerical methods for the IEP; see, for example, [3, 4, 7, 8, 9, 13, 14] and the references therein.

Recall that solving the IEP is equivalent to solving the nonlinear equation

$$\mathbf{f}(\mathbf{c}) \equiv (\lambda_1(\mathbf{c}) - \lambda_1^*, \ \ldots, \lambda_n(\mathbf{c}) - \lambda_n^*)^T = \mathbf{0}. \tag{1.2}$$

Based on this equivalence, Newton method can be applied to solving the IEP, and it converges quadratically [9]. As it is well known, each iteration of Newton method involves solving a complete eigenproblem of the matrix $A(\mathbf{c})$. To overcome this drawback, different Newton-like methods have been proposed in [3, 7, 8, 9]. In this paper, we apply the two-step Newton method to solving the IEP. For a nonlinear system $\mathbf{g}(\mathbf{x}) = \mathbf{0}$, the two-step Newton iterative procedure is as follows:

$$\begin{aligned} \mathbf{y}^k &= \mathbf{x}^k - \left[\mathbf{g}'(\mathbf{x}^k)\right]^{-1} \mathbf{g}(\mathbf{x}^k), \\ \mathbf{x}^{k+1} &= \mathbf{y}^k - \left[\mathbf{g}'(\mathbf{x}^k)\right]^{-1} \mathbf{g}(\mathbf{y}^k), \quad k = 0, 1, 2, \ldots, \end{aligned} \tag{1.3}$$

*This work is supported by National Natural Science Foundations of China (11771159)

†School of Mathematical Sciences, South China Normal University, Guangzhou, 530631, People's Republic of China (E-mail: chenxs33@163.com, 1399367896@qq.com)

‡Department of Mathematics, University of Macau, Macau, Macao(E-mail: hsun@umac.mo)

in which a simplified Newton step is composed with a Newton step [12]. The iteration (1.3) has two advantages. (i) Two iterative steps in (1.3) have the same Jacobian matrix $\mathbf{g}'(\mathbf{x}^k)$, which saves the amount of calculation to form the Jacobian matrix; (ii) The iteration (1.3) has at least cubic convergence(see, [12, pp. 315]). Now applying the two-step Newton iteration (1.3) to the nonlinear equation (1.2), we propose the two-step Newton method, the two-step Newton-like method and the two-step inexact Newton-like method for solving the IEP. Our results show that both the two-step Newton method and two-step Newton-like method converge cubically, and the two-step inexact Newton-like method is super quadratically convergent. As showed in [3, 4], forming the Jacobian matrix of $\mathbf{f}(\mathbf{c})$ needs $O(n^4)$ flops. When n is large, forming the (approximate) Jacobian matrix will be costly. Comparing with two Newton iterative steps with convergence of order 4, the two-step Newton type methods only need forming one Jacobian matrix in each step, and the convergence order is cubic.

This paper is organized as follows. In section 2, we propose the two-step Newton method, two-step Newton-like method and two-step inexact Newton-like method for solving the IEP. In section 3, we give some useful basics needed in this paper. In section 4, we give the convergence analysis of our methods. In section 5, we present numerical tests to illustrate our results. Concluding remarks are given in section 6.

For convenience, we introduce the following notation. $\mathbb{R}^{n\times n}$ represent the set of all real $n \times n$ matrices. In particular, $\mathbb{R}^n = \mathbb{R}^{n\times 1}$. We use I to denote the identity matrix of suitable size. $\|\cdot\|_2$ represents the Euclidean vector norm or the induced matrix norm, $\|\cdot\|_\infty$ is the infinity vector norm and $\|\cdot\|_F$ is the Frobenius matrix norm. Throughout this paper, we assume that given eigenvalues $\lambda_1^*, \ldots, \lambda_n^*$ are distinct, i.e., $\lambda_i^* \neq \lambda_j^*$ for $i \neq j$, $1 \leq i, j \leq n$, and the Jacobian matrix $J(\mathbf{c}^*)$ defined by

$$[J(\mathbf{c}^*)]_{ij} = \mathbf{q}_i(\mathbf{c}^*)^T A_j \mathbf{q}_i(\mathbf{c}^*), \quad 1 \leq i, j \leq n,$$

is nonsingular, where $\{\mathbf{q}_i(\mathbf{c}^*)\}_{i=1}^n$ be normalized eigenvectors of $A(\mathbf{c}^*)$ corresponding to the eigenvalues $\{\lambda_i^*\}_{i=1}^n$.

2. The two-step Newton type methods. In this section, we apply the two-step Newton iteration (1.3) to the nonlinear equation (1.2) to obtain three methods solving the IEP. At first, we recall the Jacobian matrix of the nonlinear equation $\mathbf{f}(\mathbf{c}) = 0$. Since given eigenvalues $\{\lambda_i^*\}_{i=1}^n$ are distinct, then the eigenvalues $\{\lambda_i(\mathbf{c})\}_{i=1}^n$ of $A(\mathbf{c})$ are distinct too in some neighborhood of $\mathbf{c}^*$. Let $\mathbf{q}_i(\mathbf{c})$ be the normalized eigenvector of $A(\mathbf{c})$ corresponding to the eigenvalue $\lambda_i(\mathbf{c})$, i.e., $A(\mathbf{c})\mathbf{q}_i(\mathbf{c}) = \lambda_i(\mathbf{c})\mathbf{q}_i(\mathbf{c})$ and $\|\mathbf{q}_i(\mathbf{c})\|_2 = 1$. It follows that the function $\mathbf{f}(\mathbf{c})$ is analytic in the same neighborhood (see [17]) and that the Jacobian matrix $J(\mathbf{c})$ of $\mathbf{f}(\mathbf{c})$ is given by [7, 8]

$$[J(\mathbf{c})]_{ij} = \mathbf{q}_i(\mathbf{c})^T A_j \mathbf{q}_i(\mathbf{c}), \quad 1 \leq i, j \leq n.$$

It is easy from (1.1) to see that

$$[J(\mathbf{c})\mathbf{c}]_i = \mathbf{q}_i(\mathbf{c})^T \left[A(\mathbf{c}) - A_0\right] \mathbf{q}_i(\mathbf{c}) = \lambda_i(\mathbf{c}) - \mathbf{q}_i(\mathbf{c})^T A_0 \mathbf{q}_i(\mathbf{c}), \ 1 \leq i \leq n.$$

Thus we have

$$J(\mathbf{c})\mathbf{c} = (\lambda_1(\mathbf{c}), \ldots, \lambda_n(\mathbf{c}))^T - \left(\mathbf{q}_1(\mathbf{c})^T A_0 \mathbf{q}_1(\mathbf{c}), \ldots, \mathbf{q}_n(\mathbf{c})^T A_0 \mathbf{q}_n(\mathbf{c})\right)^T. \tag{2.1}$$

Now the two-step Newton method (1.3) is applied to the nonlinear equation (1.2) to give the following two-step Newton iterative procedure:

$$\begin{array}{rcl} \mathbf{y}^k & = & \mathbf{c}^k - \left[J(\mathbf{c}^k)\right]^{-1} \mathbf{f}(\mathbf{c}^k), \\ \mathbf{c}^{k+1} & = & \mathbf{y}^k - \left[J(\mathbf{c}^k)\right]^{-1} \mathbf{f}(\mathbf{y}^k), \quad k = 0, 1, 2, \ldots. \end{array} \tag{2.2}$$

By (1.2) and (2.1)-(2.2), we get the following two-step Newton iteration for solving the IEP:

$$\begin{array}{rcl} J(\mathbf{c}^k)\mathbf{y}^k & = & \lambda^* - \mathbf{b}^k, \\ J(\mathbf{c}^k)\mathbf{c}^{k+1} & = & J(\mathbf{c}^k)\mathbf{y}^k + \lambda^* - \lambda(\mathbf{y}^k), \quad k = 0, 1, \ldots, \end{array} \tag{2.3}$$

where λ^*, $\lambda(\mathbf{y}^k)$ and $\mathbf{b}^k$ are respectively defined by

$$\begin{array}{l} \lambda^* = (\lambda_1^*, \ldots, \lambda_n^*)^T, \quad \lambda(\mathbf{y}^k) = (\lambda_1(\mathbf{y}^k), \ldots, \lambda_n(\mathbf{y}^k))^T, \\ \mathbf{b}^k = \left(\mathbf{q}_1(\mathbf{c}^k)^T A_0 \mathbf{q}_1(\mathbf{c}^k), \ldots, \mathbf{q}_n(\mathbf{c}^k)^T A_0 \mathbf{q}_n(\mathbf{c}^k)\right)^T. \end{array} \tag{2.4}$$

To summarize, from (2.3) and (2.4) we obtain the following two-step Newton method for solving the IEP.

Algorithm 1: the two-step Newton method.

For $k = 0$ until convergence do

(a) Compute the eigen-decomposition of $A(\mathbf{c}^k)$:

$$Q(\mathbf{c}^k)^T A(\mathbf{c}^k) Q(\mathbf{c}^k) = \mathrm{diag}(\lambda_1(\mathbf{c}^k), \ldots, \lambda_n(\mathbf{c}^k)),$$

where $Q(\mathbf{c}^k) = [\mathbf{q}_1(\mathbf{c}^k), \ldots, \mathbf{q}_n(\mathbf{c}^k)]$ is orthogonal.

(b) Form the Jacobian matrix $J(\mathbf{c}^k)$ and $\mathbf{b}^k$:

$$\begin{array}{l} [J(\mathbf{c}^k)]_{ij} = \mathbf{q}_i(\mathbf{c}^k)^T A_j \mathbf{q}_i(\mathbf{c}^k), \quad 1 \le i, j \le n, \\ \left[\mathbf{b}^k\right]_i = \mathbf{q}_i(\mathbf{c}^k)^T A_0 \mathbf{q}_i(\mathbf{c}^k), \quad 1 \le i \le n. \end{array}$$

(c) Solve $\mathbf{y}^k$ from the Jacobian equation:

$$J(\mathbf{c}^k)\mathbf{y}^k = \lambda^* - \mathbf{b}^k.$$

(d) Compute the eigenvalues $\{\lambda_i(\mathbf{y}^k)\}_{i=1}^n$ of $A(\mathbf{y}^k)$.

(e) Solve $\mathbf{c}^{k+1}$ from the Jacobian equation:

$$J(\mathbf{c}^k)\mathbf{c}^{k+1} = J(\mathbf{c}^k)\mathbf{y}^k + \lambda^* - \lambda(\mathbf{y}^k).$$

We will prove that the two-step Newton method converges cubically in section 4. Comparing with two Newton iterative steps, which converge with four order, each iteration in Algorithm 1 only form the Jacobian matrix $J(\mathbf{c}^k)$ once, hence saves $O(n^4)$ flops. Notice that in (a) and (d) of Algorithm 1, we have to compute all the eigenvalues and eigenvectors of $A(\mathbf{c}^k)$ and all the eigenvalues of $A(\mathbf{y}^k)$ exactly. If we only compute them approximately, for example we perform the one step inverse power method to obtain approximate eigenvectors and use Rayleigh quotients to get approximate eigenvalues, this results in the following two-step Newton-like method. We will also prove in Section 4 that this replacement will retain cubical convergence.

Algorithm 2: The two-step Newton-like Method.

1. Given $\mathbf{c}^0$, iterate Algorithm 1 once to obtain $\mathbf{y}^0$ and $\mathbf{c}^1$. Let

$$P(\mathbf{y}^0) = [\mathbf{p}_1(\mathbf{y}^0), \ldots, \mathbf{p}_n(\mathbf{y}^0)] = Q(\mathbf{y}^0).$$

2. For $k = 1$ until convergence, do:

(a) Compute $\mathbf{v}_i^k$ by the one-step inverse power method:

$$(A(\mathbf{c}^k) - \lambda_i^* I)\mathbf{v}_i^k = \mathbf{p}_i(\mathbf{y}^{k-1}), \quad 1 \le i \le n. \tag{2.5}$$

(b) Normalize $\mathbf{v}_i^k$ to obtain an approximate eigenvector $\mathbf{p}_i(\mathbf{c}^k)$ of $A(\mathbf{c}^k)$:

$$\mathbf{p}_i(\mathbf{c}^k) = \mathbf{v}_i^k / \|\mathbf{v}_i^k\|_2, \quad 1 \le i \le n.$$

(c) Form the approximate Jacobian matrix J_k and $\hat{\mathbf{b}}^k$:

$$[J_k]_{ij} = \mathbf{p}_i(\mathbf{c}^k)^T A_j \mathbf{p}_i(\mathbf{c}^k), \quad 1 \le i, j \le n,$$
$$\left[\hat{\mathbf{b}}^k\right]_i = \mathbf{p}_i(\mathbf{c}^k)^T A_0 \mathbf{p}_i(\mathbf{c}^k), \quad 1 \le i \le n.$$

(d) Solve $\mathbf{y}^k$ from the approximate Jacobian equation:

$$J_k \mathbf{y}^k = \lambda^* - \hat{\mathbf{b}}^k. \tag{2.6}$$

(e) Compute $\mathbf{u}_i^k$ by the one-step inverse power method:

$$(A(\mathbf{y}^k) - \lambda_i^* I)\mathbf{u}_i^k = \mathbf{p}_i(\mathbf{c}^k), \quad 1 \le i \le n, \tag{2.7}$$

(f) Normalize $\mathbf{u}_i^k$ to obtain an approximate eigenvector $\mathbf{p}_i(\mathbf{y}^k)$ of $A(\mathbf{y}^k)$:

$$\mathbf{p}_i(\mathbf{y}^k) = \mathbf{u}_i^k / \|\mathbf{u}_i^k\|_2, \quad 1 \le i \le n.$$

(g) Form the approximate eigenvalues of $A(\mathbf{y}^k)$:

$$\hat{\lambda}_i(\mathbf{y}^k) = \mathbf{p}_i(\mathbf{y}^k)^T A(\mathbf{y}^k) \mathbf{p}_i(\mathbf{y}^k), \quad 1 \le i \le n.$$

(h) Solve $\mathbf{c}^{k+1}$ from the approximate Jacobian equation:

$$J_k \mathbf{c}^{k+1} = J_k \mathbf{y}^k + \lambda^* - \hat{\lambda}(\mathbf{y}^k), \tag{2.8}$$

where $\hat{\lambda}(\mathbf{y}^k) = (\hat{\lambda}_1(\mathbf{y}^k), \cdots, \hat{\lambda}_n(\mathbf{y}^k))^T$.

Note that in the two-step Newton-like method, we have to solve the four linear systems (2.5)-(2.8). If the systems are large, we may want to solve these systems by iterative methods. In this case, one may even be tempted to solve the systems only approximately to reduce the computational cost of the inner iterations. In fact, if the Jacobian equations (2.6) and (2.8) is to be solved by an iterative method, then the last few iterations before convergence are usually insignificant as far as the convergence of the outer iteration is concerned. This oversolving of the Jacobian equations will cause a waste of time and does not improve the efficiency of the whole method. In order to avoid the oversolving of the inner iterations, we have to look for suitable tolerances small enough to guarantee the convergence of the outer iterations. Below we give the following two-step inexact Newton-like method for solving the IEP.

Algorithm 3: The two-step inexact Newton-like method.

1. Same as the step 1 in Algorithm 2.
2. For $k = 1$ until convergence, do
   (a) Solve $\mathbf{v}_i^k$ inexactly in the one-step inverse power method:

$$(A(\mathbf{c}^k) - \lambda_i^* I)\mathbf{v}_i^k = \mathbf{p}_i(\mathbf{y}^{k-1}) + \mathbf{t}_i^k, \quad 1 \le i \le n, \tag{2.9}$$

   until the residual $\mathbf{t}_i$ satisfies

$$\|\mathbf{t}_i^k\|_2 \le 1/4. \tag{2.10}$$

   (b) Same as (b) in Algorithm 2.
   (c) Same as (c) in Algorithm 2.
   (d) Solve $\mathbf{y}^k$ inexactly from the approximate Jacobian equation

$$J_k\mathbf{y}^k = \left(\lambda^* - \hat{\mathbf{b}}^k\right) + \mathbf{r}(\mathbf{c}^k), \tag{2.11}$$

   until the residual $\mathbf{r}(\mathbf{c}^k)$ satisfies

$$\|\mathbf{r}(\mathbf{c}^k)\|_2 \le \left(\max_{1\le i\le n} \frac{1}{\| \mathbf{v}_i^k \|_2}\right)^{\beta_1}, \ 1 < \beta_1 \le 2. \tag{2.12}$$

   (e) Solve $\mathbf{u}_i^k$ inexactly in the one-step inverse power method

$$(A(\mathbf{y}^k) - \lambda_i^* I)\mathbf{u}_i^k = \mathbf{p}_i(\mathbf{c}^k) + \mathbf{s}_i^k, \quad 1 \le i \le n, \tag{2.13}$$

   until the residual $\mathbf{s}_i^k$ satisfies

$$\|\mathbf{s}_i^k\|_2 \le 1/4. \tag{2.14}$$

   (f) Same as (f) in Algorithm 2.
   (g) Same as (g) in Algorithm 2.
   (h) Solve $\mathbf{c}^{k+1}$ inexactly from the approximate Jacobian equation

$$J_k\mathbf{c}^{k+1} = \left[J_k\mathbf{y}^k + \lambda^* - \hat{\lambda}(\mathbf{y}^k)\right] + \mathbf{r}(\mathbf{y}^k), \tag{2.15}$$

   until the residual $\mathbf{r}(\mathbf{y}^k)$ satisfies

$$\|\mathbf{r}(\mathbf{y}^k)\|_2 \le \left(\max_{1\le i\le n} \frac{1}{\|\mathbf{u}_i^k\|}\right)^{\beta_2}, \ 2/\beta_1 < \beta_2 \le 1 + 1/\beta_1. \tag{2.16}$$

Note that the main difference between Algorithm 2 and Algorithm 3 is that we solve four linear systems approximately in Algorithm 3 rather than exactly as in Algorithm 2. We will prove in Section 4 that the convergence rate of Algorithm 3 is equal to $\beta_1\beta_2$. From (2.12) and (2.16) we get $2 < \beta_1\beta_2 \le 1+\beta_1 \le 3$. Hence Algorithm 3 is at least super quadratically convergent.

3. Preliminaries. In this section, we give some lemmas, which are used to prove convergence of Algorithms 1-3. Let $\{\lambda_i^*\}_{i=1}^n$ be given with $\lambda_1^* < \lambda_2^* < \cdots < \lambda_n^*$, and $\mathbf{c}^* \in \mathbb{R}^n$ be a solution of the IEP. Let $\{\lambda_i(\mathbf{c})\}_{i=1}^n$ be the eigenvalues $A(\mathbf{c})$ corresponding to the normalized eigenvectors $\{\mathbf{q}_i(\mathbf{c})\}_{i=1}^n$. Firstly we list some lemmas that are already proven in [7, 8].

LEMMA 3.1. *Let $\{\mathbf{w}_i\}_{i=1}^n$ be the normalized vector approximating $\{\mathbf{q}_i(\mathbf{c}^*)\}_{i=1}^n$. Define the approximate Jacobian matrix $J(\mathbf{w})$ and column vector $\mathbf{b}(\mathbf{w})$ by*

$$[J(\mathbf{w})]_{ij} = \mathbf{w}_i^T A_j \mathbf{w}_i, \quad 1 \le i, j \le n, \quad [\mathbf{b}(\mathbf{w})]_i = \mathbf{w}_i^T A_0 \mathbf{w}_i, \quad 1 \le i \le n.$$

Then we have

$$\|J(\mathbf{w})\mathbf{c}^* + \mathbf{b}(\mathbf{w}) - \lambda^*\|_2 \le 2n \|\lambda^*\|_\infty \max_{1\le i\le n} \|\mathbf{w}_i - \mathbf{q}_i(\mathbf{c}^*)\|_2^2, \tag{3.1}$$

$$\|J(\mathbf{w}) - J(\mathbf{c}^*)\|_F \le 2n \max_{1\le j\le n} \|A_j\|_2 \max_{1\le i\le n} \|\mathbf{w}_i - \mathbf{q}_i(\mathbf{c}^*)\|_2. \tag{3.2}$$

Hence if the Jacobian matrix $J(\mathbf{c}^*)$ be nonsingular, then there exist positive numbers δ_1 and τ such that if $\max_{1\le i\le n} \|\mathbf{w}_i - \mathbf{q}_i(\mathbf{c}^*)\|_2 \le \delta_1$, then $J(\mathbf{w})$ is nonsingular and

$$\left\| [J(\mathbf{w})]^{-1} \right\|_2 \le \tau. \tag{3.3}$$

Proof. See Lemma 4.3 and Lemma 4.4 in [8]. □

REMARK 3.1. Since $[J(\mathbf{w})\mathbf{c}^* + \mathbf{b}(\mathbf{w})]_i = \mathbf{w}_i^T A(\mathbf{c}^*)\mathbf{w}_i$ and $\lambda_i^* = \mathbf{q}_i(\mathbf{c}^*)^T A(\mathbf{c}^*)\mathbf{q}_i(\mathbf{c}^*)$, the inequality (3.1) shows that the precision of the Rayleigh quotient $\mathbf{w}_i^T A(\mathbf{c}^*)\mathbf{w}_i$ as an approximate eigenvalue of the symmetric matrix $A(\mathbf{c}^*)$ is higher than the one of $\mathbf{w}_i$ as its approximate eigenvector (also see [15]).

LEMMA 3.2. *Let the given eigenvalues $\{\lambda_i^*\}_{i=1}^n$ be distinct and $\{\mathbf{q}_i(\mathbf{c}^*)\}_{i=1}^n$ be the normalized eigenvectors of $A(\mathbf{c}^*)$ corresponding to $\{\lambda_i^*\}_{i=1}^n$. Then there exist positive numbers δ_0, ρ_0 and γ such that if $\|\mathbf{c} - \mathbf{c}^*\|_2 \le \delta_0$, then*

$$| \lambda_i(\mathbf{c}) - \lambda_i^* | \le \rho_0 \|\mathbf{c} - \mathbf{c}^*\|_2, \quad 1 \le i \le n, \tag{3.4}$$

$$\| \mathbf{q}_i(\mathbf{c}) - \mathbf{q}_i(c^*)\|_2 \le \rho_0 \|\mathbf{c} - \mathbf{c}^*\|_2, \quad 1 \le i \le n, \tag{3.5}$$

$$| \lambda_i(\mathbf{c}) - \lambda_j^* | \ge \gamma > 0, \quad 1 \le i \ne j \le n. \tag{3.6}$$

Proof. See Lemma 4.2 in [8]. □

LEMMA 3.3. *Let $\mathbf{p}_1, \ldots, \mathbf{p}_n \in \mathbb{R}^n$ be the normalized vectors. Consider solving $\mathbf{v}_i$ inexactly in the one-step inverse power method:*

$$(A(\mathbf{c}) - \lambda_i^* I)\mathbf{v}_i = \mathbf{p}_i + \mathbf{t}_i, \quad 1 \le i \le n.$$

If the residual $\mathbf{t}_i$ satisfy that $\|\mathbf{t}_i\|_2 \le 1/4$, $| \mathbf{q}_i(\mathbf{c})^T (\mathbf{p}_i + \mathbf{t}_i)) | \ge 1/4$, $1 \le i \le n$, and

$$| \lambda_j(\mathbf{c}) - \lambda_i^* | \ge \gamma > 0, \ 1 \le j \ne i \le n.$$

Then we have

$$\frac{1}{\|\mathbf{v}_i\|_2} \le 4 \, | \lambda_i(\mathbf{c}) - \lambda_i^* |, \ \|\mathbf{p}_i - \mathbf{q}_i(\mathbf{c})\|_2 \le (8/\gamma) \, | \lambda_i(\mathbf{c}) - \lambda_i^* |, \quad 1 \le i \le n. \tag{3.7}$$

Proof. Taking $\xi = \frac{1}{4}$ in [8, Lemma 4.1], we obtain the inequalities (3.7). □

LEMMA 3.4. *Let $\{\mathbf{p}_i(\mathbf{c})\}_{i=1}^n$ be the normalized vectors approximating the normalized eigenvectors $\{\mathbf{q}_i(\mathbf{c})\}_{i=1}^n$ of $A(\mathbf{c})$ and define the Rayleigh quotient of $\mathbf{p}_i(\mathbf{c})$ and column vector $\mathbf{b}$ as follows:*

$$\hat{\lambda}_i(\mathbf{c}) = \mathbf{p}_i(\mathbf{c})^T A(\mathbf{c})\mathbf{p}_i(\mathbf{c}), \quad [\mathbf{b}]_i = \mathbf{q}_i(\mathbf{c}_*)^T A_0 \mathbf{q}_i(\mathbf{c}_*), \quad 1 \le i \le n. \tag{3.8}$$

Let $\hat{\lambda}(\mathbf{c}) = (\hat{\lambda}_1(\mathbf{c}), \ldots, \hat{\lambda}_n(\mathbf{c}))^T$ and $\lambda(\mathbf{c}) = (\lambda_1(\mathbf{c}), \ldots, \lambda_n(\mathbf{c}))^T$. Then we have

$$\left\| J(\mathbf{c}^*)\mathbf{c} + \mathbf{b} - \hat{\lambda}(\mathbf{c}) \right\|_2 \le 2n \left\|\lambda(\mathbf{c})\right\|_\infty \times \left(\max_{1\le i\le n} \|\mathbf{q}_i(\mathbf{c}) - \mathbf{q}_i(\mathbf{c}^*)\|_2^2 + \max_{1\le i\le n} \|\mathbf{p}_i(\mathbf{c}) - \mathbf{q}_i(\mathbf{c})\|_2^2 \right). \tag{3.9}$$

Proof. Note that

$$\begin{aligned} \|J(\mathbf{c}^*)\mathbf{c} + \mathbf{b} - \hat{\lambda}(\mathbf{c})\|_2 &= \|J(\mathbf{c}^*)\mathbf{c} + \mathbf{b} - \lambda(\mathbf{c}) + \lambda(\mathbf{c}) - \hat{\lambda}(\mathbf{c})\|_2 \\ &\le \|J(\mathbf{c}^*)\mathbf{c} + \mathbf{b} - \lambda(\mathbf{c})\|_2 + \|\hat{\lambda}(\mathbf{c}) - \lambda(\mathbf{c})\|_2. \end{aligned} \tag{3.10}$$

By (3.1) and Remark 3.1, we have

$$\|J(\mathbf{c}^*)\mathbf{c} + \mathbf{b} - \lambda(\mathbf{c})\|_2 \le 2n\|\lambda(\mathbf{c})\|_\infty \max_{1\le i\le n} \|\mathbf{q}_i(\mathbf{c}^*) - \mathbf{q}_i(\mathbf{c})\|_2^2, \tag{3.11}$$

$$\|\hat{\lambda}(\mathbf{c}) - \lambda(\mathbf{c})\|_2 \le 2n\|\lambda(\mathbf{c})\|_\infty \max_{1\le i\le n} \|\mathbf{p}_i(\mathbf{c}) - \mathbf{q}_i(\mathbf{c})\|_2^2. \tag{3.12}$$

Combining (3.11)–(3.12) with (3.10), we get (3.9). □

The following lemma give an estimation of $\|\lambda(\mathbf{c})\|_\infty$ in Lemma 3.4.

LEMMA 3.5. *Let δ_0 and $\lambda(\mathbf{c})$ be defined in Lemma 3.2 and Lemma 3.4, respectively. If $\|\mathbf{c} - \mathbf{c}^*\|_2 \le \delta_0$, then*

$$\|\lambda(\mathbf{c})\|_\infty \le \rho_0 \|\mathbf{c} - \mathbf{c}^*\|_2 + \|\lambda^*\|_\infty. \tag{3.13}$$

Proof. From (3.4), we have

$$|\lambda_i(\mathbf{c})| \le |\lambda_i(\mathbf{c}) - \lambda_i^*| + |\lambda_i^*| \le \rho_0 \|\mathbf{c} - \mathbf{c}^*\|_2 + |\lambda_i^*|.$$

which implies that (3.13) holds. □

4. Convergence analysis. In this section we carry on convergence analysis of Algorithms 1–3. Let $\{\lambda_i(\mathbf{c}^k)\}_{i=1}^n$ and $\{\mathbf{q}_i(\mathbf{c}^k)\}_{i=1}^n$ be the eigenvalues and normalized eigenvectors of $A(\mathbf{c}^k)$, $\{\lambda_i(\mathbf{y}^k)\}_{i=1}^n$ and $\{\mathbf{q}_i(\mathbf{y}^k)\}_{i=1}^n$ be the eigenvalues and normalized eigenvectors of $A(\mathbf{y}^k)$. According to Lemmas 3.1–3.3, we define

$$\rho = (1 + 8/\gamma)\,\rho_0, \qquad \alpha_1 = 2n\rho_0^2\,(\rho_0 + \|\lambda^*\|_\infty)\,\left(1 + 64/\gamma^2\right), \tag{4.1}$$

$$\alpha_2 = 2n\rho \max\{\|\lambda^*\|_\infty, \max_{1\le j\le n} \|A_j\|_2\}, \quad \alpha_3 = 4^{\beta_1}\rho_0^{\beta_1}, \quad \alpha_4 = 4^{\beta_2}\rho_0^{\beta_2}, \tag{4.2}$$

$$\alpha = \alpha_1\tau^3(\alpha_2 + \alpha_3)^2 + \alpha_2\tau^2(\alpha_2 + \alpha_3) + \alpha_4\tau^{1+\beta_2}(\alpha_2 + \alpha_3)^{\beta_2}, \tag{4.3}$$

and

$$\delta = \min\left\{1, \delta_0, \delta_1, \frac{1}{\rho+\rho_0}, \frac{1}{\tau\alpha_1}, \frac{1}{2\tau(\alpha_1+\alpha_2)}, \frac{1}{[\tau(\alpha_2+\alpha_3)]^{1/(\beta_1-1)}}, \frac{1}{(2\tau\alpha_3)^{1/(\beta_2-1)}}\right\}. \tag{4.4}$$

Next we give the convergence rate of Algorithm 3 in terms of $\max_{1\le i\le n}\|\mathbf{p}_i(\mathbf{c}^k) - \mathbf{q}_i(\mathbf{c}^*)\|_2$ and $\max_{1\le i\le n}\|\mathbf{p}_i(\mathbf{y}^k) - \mathbf{q}_i(\mathbf{y}^k)\|_2$.

PROPOSITION 4.1. *Let the given eigenvalues $\{\lambda_i^*\}_{i=1}^n$ be distinct and the Jacobian matrix $J(\mathbf{c}^*)$ be nonsingular. Then in Algorithm 3, the conditions*

$$\|\mathbf{y}^k - \mathbf{c}^*\|_2 \le \|\mathbf{c}^k - \mathbf{c}^*\|_2 \le \|\mathbf{y}^{k-1} - \mathbf{c}^*\|_2 \le \|\mathbf{c}^{k-1} - \mathbf{c}^*\|_2 \le \delta, \tag{4.5}$$

$$\max_{1\le i\le n}\|\mathbf{p}_i(\mathbf{c}^{k-1}) - \mathbf{q}_i(\mathbf{c}^*)\|_2 \le \rho\|\mathbf{c}^{k-1} - \mathbf{c}^*\|_2, \tag{4.6}$$

$$\max_{1\le i\le n}\|\mathbf{p}_i(\mathbf{y}^{k-1}) - \mathbf{q}_i(\mathbf{y}^{k-1})\|_2 \le \rho\|\mathbf{y}^{k-1} - \mathbf{c}^*\|_2 \tag{4.7}$$

imply

$$\max_{1\le i\le n}\|\mathbf{p}_i(\mathbf{c}^k) - \mathbf{q}_i(\mathbf{c}^*)\|_2 \le \rho\|\mathbf{c}^k - \mathbf{c}^*\|_2, \tag{4.8}$$

$$\max_{1\le i\le n}\|\mathbf{p}_i(\mathbf{y}^k) - \mathbf{q}_i(\mathbf{y}^k)\|_2 \le \rho\|\mathbf{y}^k - \mathbf{c}^*\|_2, \tag{4.9}$$

$$\|\mathbf{c}^{k+1} - \mathbf{c}^*\|_2 \le \alpha\|\mathbf{c}^k - \mathbf{c}^*\|_2^{\beta_1\beta_2}, \tag{4.10}$$

where ρ, α and δ are defined by (4.1),(4.3) and (4.4), respectively.

Proof. Let us now prove (4.8) and (4.9). By conditions (2.10), (2.14) and (4.5)-(4.7), we apply the techniques described in[8, pp. 16] to getting

$$\left|\mathbf{q}_i\left(\mathbf{c}^k\right)^T(\mathbf{p}_i(\mathbf{y}^{k-1}) + \mathbf{t}_i^k)\right| \ge 1/4, \quad \left|\mathbf{q}_i\left(\mathbf{y}^k\right)^T(\mathbf{p}_i(\mathbf{c}^k) + \mathbf{s}_i^k)\right| \ge 1/4$$

for $i = 1, 2, \ldots, n$. Hence applying Lemma 3.3 to (2.9) and (2.13), we get

$$\|\mathbf{p}_i(\mathbf{c}^k) - \mathbf{q}_i(\mathbf{c}^k)\|_2 \le (8/\gamma) \mid \lambda_i(\mathbf{c}^k) - \lambda_i^* \mid, \quad 1 \le i \le n, \tag{4.11}$$

$$\|\mathbf{p}_i(\mathbf{y}^k) - \mathbf{q}_i(\mathbf{y}^k)\|_2 \le (8/\gamma) \mid \lambda_i(\mathbf{y}^k) - \lambda_i^* \mid, \quad 1 \le i \le n, \tag{4.12}$$

respectively. From (3.4), (3.5) and (4.11), we get

$$\begin{aligned}\|\mathbf{p}_i(\mathbf{c}^k) - \mathbf{q}_i(\mathbf{c}^*)\|_2 &\le \|\mathbf{p}_i(\mathbf{c}^k) - \mathbf{q}_i(\mathbf{c}^k)\|_2 + \|\mathbf{q}_i(\mathbf{c}^k) - \mathbf{q}_i(\mathbf{c}^*)\|_2 \\ &\le (8/\gamma) \mid \lambda_i(\mathbf{c}^k) - \lambda_i^* \mid + \rho_0\|\mathbf{c}^k - \mathbf{c}^*\|_2 \\ &\le (8/\gamma)\rho_0\|\mathbf{c}^k - \mathbf{c}^*\|_2 + \rho_0\|\mathbf{c}^k - \mathbf{c}^*\|_2 \\ &= \rho\|\mathbf{c}^k - \mathbf{c}^*\|_2.\end{aligned}$$

Hence (4.8) is proven. It follows from (3.4) and (4.12) that

$$\|\mathbf{p}_i(\mathbf{y}^k) - \mathbf{q}_i(\mathbf{y}^k)\|_2 \le (8/\gamma)\rho_0\|\mathbf{y}^k - \mathbf{c}^*\|_2 \le \rho\|\mathbf{y}^k - \mathbf{c}^*\|_2,$$

which implies that the inequality (4.9) holds.

Next we prove (4.10). From (2.11) we get

$$J_k\left(\mathbf{y}^k - \mathbf{c}^*\right) = \lambda^* - J_k\mathbf{c}^* - \hat{\mathbf{b}}^k + \mathbf{r}(\mathbf{c}^k). \tag{4.13}$$

By (2.15) and $J(\mathbf{c}^*)\mathbf{c}^* = \lambda^* - \mathbf{b}$ we get

$$J_k(\mathbf{c}^{k+1} - \mathbf{c}^*) = (J(\mathbf{c}^*)\mathbf{y}^k + \mathbf{b} - \hat{\lambda}(\mathbf{y}^k)) + (J_k - J(\mathbf{c}^*))(\mathbf{y}^k - \mathbf{c}^*) + \mathbf{r}(\mathbf{y}^k), \tag{4.14}$$

where $\mathbf{b}$ is defined by (3.8). So we get from (4.13) and (4.14)

$$\|\mathbf{y}^k - \mathbf{c}^*\|_2 \le \tau \left(\left\| J_k\mathbf{c}^* + \hat{\mathbf{b}}^k - \lambda^* \right\|_2 + \left\| \mathbf{r}(\mathbf{c}^k) \right\|_2 \right), \tag{4.15}$$

$$\begin{aligned} \left\| \mathbf{c}^{k+1} - \mathbf{c}^* \right\|_2 \le \tau \Big[& \|J(\mathbf{c}^*)\mathbf{y}^k + \mathbf{b} - \hat{\lambda}(\mathbf{y}^k)\|_2 + \|J_k - J(\mathbf{c}^*)\|_2 \times \\ & \|\mathbf{y}^k - \mathbf{c}^*\|_2 + \|\mathbf{r}(\mathbf{y}^k)\|_2 \Big], \end{aligned} \tag{4.16}$$

respectively, where τ is defined by (3.3).

At first, we estimate $\|J_k\mathbf{c}^* + \hat{\mathbf{b}}^k - \lambda^*\|_2$ in (4.15) and $\|J(\mathbf{c}^*)\mathbf{y}^k + \mathbf{b} - \hat{\lambda}(\mathbf{y}^k)\|_2$ in (4.16).

From (3.1) and (4.8), we have

$$\begin{aligned} \left\| J_k\mathbf{c}^* + \hat{\mathbf{b}}^k - \lambda^* \right\|_2 &\le 2n\|\lambda^*\|_\infty \max_{1\le i\le n} \|\mathbf{p}_i(\mathbf{c}^k) - \mathbf{q}_i(\mathbf{c}^*)\|_2^2 \\ &\le 2n\|\lambda^*\|_\infty \rho \|\mathbf{c}^k - \mathbf{c}^*\|_2^2 \le \alpha_2 \|\mathbf{c}^k - \mathbf{c}^*\|_2^2. \end{aligned} \tag{4.17}$$

From (3.4), (3.5), (3.7), (3.9) and (3.13), we obtain

$$\begin{aligned} \|J(\mathbf{c}^*)\mathbf{y}^k + \mathbf{b} - \hat{\lambda}(\mathbf{y}^k)\|_2 &\le 2n\|\lambda(\mathbf{y}^k)\|_\infty \left[\rho_0^2\|\mathbf{y}^k - \mathbf{c}^*\|_2^2 + 64|\lambda_i(\mathbf{y}^k) - \lambda_i^*|^2/\gamma^2\right] \\ &\le 2n\|\lambda(\mathbf{y}^k)\|_\infty \rho_0^2(1 + 64/\gamma^2)\|\mathbf{y}^k - \mathbf{c}^*\|_2^2 \\ &\le 2n\rho_0^2(\rho_0\|\mathbf{y}^k - \mathbf{c}^*\|_2 + \|\lambda^*\|_\infty)(1 + 64/\gamma^2)\|\mathbf{y}^k - \mathbf{c}^*\|_2^2 \\ &\le 2n\rho_0^2(\rho_0 + \|\lambda^*\|_\infty)(1 + 64/\gamma^2)\|\mathbf{y}^k - \mathbf{c}^*\|_2^2 \\ &= \alpha_1\|\mathbf{y}^k - \mathbf{c}^*\|_2^2. \end{aligned} \tag{4.18}$$

Next we estimate $\|J_k - J(\mathbf{c}^*)\|_2$ in (4.16).

It follows from (3.2) and (4.8) that

$$\begin{aligned} \|J_k - J(\mathbf{c}^*)\|_2 &\le 2n \max_{1\le j\le n} \|A_j\|_2 \max_{1\le i\le n} \|\mathbf{p}_i(\mathbf{c}^k) - \mathbf{q}_i(\mathbf{c}^*)\|_2 \\ &\le 2n\rho \max_{1\le j\le n} \|A_j\|_2 \|\mathbf{c}^k - \mathbf{c}^*\|_2 \le \alpha_2\|\mathbf{c}^k - \mathbf{c}^*\|_2. \end{aligned} \tag{4.19}$$

Finally, we estimate $\|\mathbf{r}(\mathbf{c}^k)\|_2$ in (4.15) and $\|\mathbf{r}(\mathbf{y}^k)\|_2$ in (4.16).

By (3.4) and (3.7), we have

$$\begin{aligned} \frac{1}{\|\mathbf{v}_i^k\|_2} &\le 4 \mid \lambda_i(\mathbf{c}^k) - \lambda_i^* \mid \le 4\rho_0\|\mathbf{c}^k - \mathbf{c}^*\|_2, \quad 1 \le i \le n, \\ \frac{1}{\left\|\mathbf{u}_i^k\right\|_2} &\le 4 \mid \lambda_i(\mathbf{y}^k) - \lambda_i^* \mid \le 4\rho_0 \left\|\mathbf{y}^k - \mathbf{c}^*\right\|_2, \quad 1 \le i \le n. \end{aligned}$$

Hence it follows from (2.12) and (2.16) that

$$\left\|\mathbf{r}(\mathbf{c}^k)\right\|_2 \le 4^{\beta_1}\rho_0^{\beta_1} \left\|\mathbf{c}^k - \mathbf{c}^*\right\|_2^{\beta_1} = \alpha_3 \left\|\mathbf{c}^k - \mathbf{c}^*\right\|_2^{\beta_1}, \tag{4.20}$$

$$\|\mathbf{r}(\mathbf{y}^k)\|_2 \le 4^{\beta_2}\rho_0^{\beta_2}\|\mathbf{y}^k - \mathbf{c}^*\|_2^{\beta_2} = \alpha_4\|\mathbf{y}^k - \mathbf{c}^*\|_2^{\beta_2}, \tag{4.21}$$

respectively. Combining (4.17), (4.20) with (4.15), we get

$$\begin{aligned}(4.22)\qquad \|\mathbf{y}^k-\mathbf{c}^*\|_2 &\le \tau\left[\alpha_2\|\mathbf{c}^k-\mathbf{c}^*\|_2^2+\alpha_3\|\mathbf{c}^k-\mathbf{c}^*\|_2^{\beta_1}\right]\\ &\le \tau\left[\alpha_2\|\mathbf{c}^k-\mathbf{c}^*\|_2^{2-\beta_1}+\alpha_3\right]\|\mathbf{c}^k-\mathbf{c}^*\|_2^{\beta_1}\\ &\le \tau\left(\alpha_2+\alpha_3\right)\|\mathbf{c}^k-\mathbf{c}^*\|_2^{\beta_1}.\end{aligned}$$

Similarly, combining (4.18), (4.19), (4.21) with (4.16)

$$\begin{aligned}(4.23)\qquad \|\mathbf{c}^{k+1}-\mathbf{c}^*\|_2 \le \tau\big(&\alpha_1\|\mathbf{y}^k-\mathbf{c}^*\|_2^2+\alpha_2\|\mathbf{c}^k-\mathbf{c}^*\|_2\|\mathbf{y}^k-\mathbf{c}^*\|_2\\ &+\alpha_4\|\mathbf{y}^k-\mathbf{c}^*\|_2^{\beta_2}\big).\end{aligned}$$

It follows from (4.22) and (4.23) that

$$\begin{aligned}\|\mathbf{c}^{k+1}-\mathbf{c}^*\|_2 &\le \alpha_1\tau^3(\alpha_2+\alpha_3)^2\|\mathbf{c}^k-\mathbf{c}^*\|_2^{2\beta_1}+\alpha_2\tau^2(\alpha_2+\alpha_3)\|\mathbf{c}^k-\mathbf{c}^*\|_2^{1+\beta_1}\\ &\quad+\alpha_4\tau^{1+\beta_2}(\alpha_2+\alpha_3)^{\beta_2}\|\mathbf{c}^k-\mathbf{c}^*\|_2^{\beta_1\beta_2}\\ &\le \Big[\alpha_1\tau^3(\alpha_2\alpha_3)^2\|\mathbf{c}^k-\mathbf{c}^*\|_2^{\beta_1(2-\beta_2)}+\alpha_2\tau^2(\alpha_2+\alpha_3)\times\\ &\quad\|\mathbf{c}^k-\mathbf{c}^*\|_2^{1+\beta_1-\beta_1\beta_2}+\alpha_4\tau^{1+\beta_2}(\alpha_2+\alpha_3)^{\beta_2}\Big]\|\mathbf{c}^k-\mathbf{c}^*\|_2^{\beta_1\beta_2}\\ &\le \left[\alpha_1\tau^3(\alpha_2+\alpha_3)^2+\alpha_2\tau^2(\alpha_2+\alpha_3)+\alpha_4\tau^{1+\beta_2}(\alpha_2+\alpha_3)^{\beta_2}\right]\times\\ &\quad\|\mathbf{c}^k-\mathbf{c}^*\|_2^{\beta_1\beta_2}=\alpha\|\mathbf{c}^k-\mathbf{c}^*\|_2^{\beta_1\beta_2},\end{aligned}$$

where the last inequality follows from

$$\|\mathbf{c}^k-\mathbf{c}^*\|_2\le\delta\le 1,\quad 1<\beta_1\le 2,\quad \text{and}\quad \beta_2\le 1+1/\beta_1<2.$$

The proof is complete. □

We are in the position to prove the local convergence of Algorithm 3 based on Proposition 4.1.

THEOREM 4.1. *Let given eigenvalues $\{\lambda_i^*\}_{i=1}^n$ be distinct and the Jacobian matrix $J(\mathbf{c}^*)$ be nonsingular. Then Algorithm 3 is locally convergent with convergence rate of $\beta_1\beta_2$. More precisely, if $\|\mathbf{c}^0-\mathbf{c}^*\|_2\le\delta$, then $\mathbf{c}^k$ converges to $\mathbf{c}^*$ with*

$$(4.24)\qquad \|\mathbf{c}^{k+1}-\mathbf{c}^*\|_2\le\alpha\|\mathbf{c}^k-\mathbf{c}^*\|_2^{\beta_1\beta_2},\quad k=0,1,2,\ldots,$$

where α and δ are defined by (4.3) and (4.4), respectively.

Proof. We first prove by mathematical induction that (4.5), (4.6) and (4.7) are true for all k.

For $k=1$, we recall that the step 1 of Algorithm 3 is the same as the first iteration of Algorithm 1. Hence $\mathbf{p}_i(\mathbf{c}^0),\mathbf{p}_i(\mathbf{y}^0)$ are the exact eigenvectors of $A(\mathbf{c}^0)$ and $A(\mathbf{y}^0)$, respectively, i.e. $\mathbf{p}_i(\mathbf{c}^0)=\mathbf{q}_i(\mathbf{c}^0)$ and $\mathbf{p}_i(\mathbf{y}^0)=\mathbf{q}_i(\mathbf{y}^0)$. It follows from (3.5) that

$$\begin{aligned}\|\mathbf{p}_i(\mathbf{c}^0)-\mathbf{q}_i(\mathbf{c}^*)\|_2&=\|\mathbf{q}_i(\mathbf{c}^0)-\mathbf{q}_i(\mathbf{c}^*)\|_2\le\rho_0\|\mathbf{c}^0-\mathbf{c}^*\|_2\le\rho\|\mathbf{c}^0-\mathbf{c}^*\|_2,\\ \|\mathbf{p}_i(\mathbf{y}^0)-\mathbf{q}_i(\mathbf{y}^0)\|_2&=\|\mathbf{q}_i(\mathbf{y}^0)-\mathbf{q}_i(\mathbf{y}^0)\|_2\le\rho_0\|\mathbf{y}^0-\mathbf{c}^*\|_2\le\rho\|\mathbf{y}^0-\mathbf{c}^*\|_2.\end{aligned}$$

Hence (4.6) and (4.7) are true for $k=1$. Since the Jacobian equation is solved exactly in the step 1 of Algorithm 3, we get from the step (c) of Algorithm 1

$$J(\mathbf{c}_0)(\mathbf{y}^0-\mathbf{c}^*)=\lambda^*-\mathbf{b}^0-J(\mathbf{c}_0)\mathbf{c}^*.$$

Hence we have

$$\|\mathbf{y}^0 - \mathbf{c}^*\|_2 \le \|J(\mathbf{c}_0)^{-1}\|_2 \|\lambda^* - \mathbf{b}^0 - J(\mathbf{c}_0)\mathbf{c}^*\|_2 \le \tau\|\lambda^* - \mathbf{b}^0 - J(\mathbf{c}_0)\mathbf{c}^*\|_2, \tag{4.25}$$

where τ is defined by (3.3). It follows from (3.1) and (3.5) that

$$\begin{aligned}\|\lambda^* - \mathbf{b}^0 - J(\mathbf{c}_0)\mathbf{c}^*\|_2 &\le 2n\|\lambda^*\|_\infty \max_{1\le i\le n} \|\mathbf{q}_i(\mathbf{c}^0) - \mathbf{q}_i(\mathbf{c}^*)\|_2^2 \\ &\le 2n\rho_0^2\|\lambda^*\|_\infty\|\mathbf{c}^0 - \mathbf{c}^*\|_2^2.\end{aligned} \tag{4.26}$$

Combining (4.25) with (4.26) yields

$$\begin{aligned}\|\mathbf{y}^0 - \mathbf{c}^*\|_2 &\le 2n\tau\rho_0^2\|\lambda^*\|_\infty\|\mathbf{c}^0 - \mathbf{c}^*\|_2^2 \le \tau\alpha_1\|\mathbf{c}^0 - \mathbf{c}^*\|_2^2 \\ &\le \tau\alpha_1\delta\|\mathbf{c}^0 - \mathbf{c}^*\|_2 \le \|\mathbf{c}^0 - \mathbf{c}^*\|_2,\end{aligned}$$

where the last inequality follows from (4.4).

Since the Jacobian equation is solved exactly in the step 1 of Algorithm 3, we have $J_0 = J(\mathbf{c}_0), \mathbf{r}(\mathbf{y}^0) = 0$ in (4.14) and $\hat{\lambda}(\mathbf{y}^0) = \lambda(\mathbf{y}^0)$. (4.14) reduces to

$$J(\mathbf{c}_0)(\mathbf{c}^1 - \mathbf{c}^*) = \left(J(\mathbf{c}^*)\mathbf{y}^0 + \mathbf{b} - \lambda(\mathbf{y}^0)\right) + (J(\mathbf{c}_0) - J(\mathbf{c}^*))(\mathbf{y}^0 - \mathbf{c}^*). \tag{4.27}$$

Applying Lemma 3.1, Lemma 3.2 and Lemma 3.5 to (4.27), we get

$$\begin{aligned}\|\mathbf{c}^1 - \mathbf{c}^*\|_2 &\le \tau\left[\left(\rho_0\|\mathbf{y}^0 - \mathbf{c}^*\|_2 + \|\lambda^*\|_\infty\right) 2n\rho_0^2\|\mathbf{y}^0 - \mathbf{c}^*\|_2\right. \\ &\quad \left. +2n \max_{1\le j\le n} \|A_j\|_2\rho_0\|\mathbf{c}^0 - \mathbf{c}^*\|_2\right]\|\mathbf{y}^0 - \mathbf{c}^*\|_2 \\ &\le \delta\tau\left[2n\rho_0^2\left(\rho_0 + \|\lambda^*\|_\infty\right) + 2n\rho_0 \max_{1\le j\le n} \|A_j\|_2\right]\|\mathbf{y}^0 - \mathbf{c}^*\|_2 \\ &\le \delta\tau(\alpha_1 + \alpha_2)\|\mathbf{y}^0 - \mathbf{c}^*\|_2 \le \|\mathbf{y}^0 - \mathbf{c}^*\|_2,\end{aligned}$$

where we use (4.4) and $\|\mathbf{y}^0 - \mathbf{c}^*\|_2 \le \|\mathbf{c}^0 - \mathbf{c}^*\|_2 \le \delta \le 1$.

From (4.22), we have

$$\begin{aligned}\|\mathbf{y}^1 - \mathbf{c}^*\|_2 &\le \tau(\alpha_2 + \alpha_3)\|\mathbf{c}^1 - \mathbf{c}^*\|_2^{\beta_1} \\ &\le \tau(\alpha_2 + \alpha_3)\|\mathbf{c}^1 - \mathbf{c}^*\|_2^{\beta_1 - 1}\|\mathbf{c}^1 - \mathbf{c}^*\|_2, \\ &\le \tau(\alpha_2 + \alpha_3)\delta^{\beta_1 - 1}\|\mathbf{c}^1 - \mathbf{c}^*\|_2,\end{aligned}$$

where the last inequality follows from (4.4). Thus (4.5) is also valid for $k = 1$.

Next we assume that (4.5), (4.6) and (4.7) are true for all positive integers less than or equal to k. Then by Proposition 4.1, we see that (4.6) and (4.7) are true for $k + 1$. Moreover from (4.23), we get

$$\begin{aligned}\|\mathbf{c}^{k+1} - \mathbf{c}^*\|_2 &\le \tau\left[\alpha_1\|\mathbf{y}^k - \mathbf{c}^*\|_2^2 + \alpha_2\|\mathbf{c}^k - \mathbf{c}^*\|_2\|\mathbf{y}^k - \mathbf{c}^*\|_2 + \alpha_4\|\mathbf{y}^k - \mathbf{c}^*\|_2^{\beta_2}\right] \\ &\le \left[\tau(\alpha_1 + \alpha_2)\delta + \tau\alpha_4\delta^{\beta_2 - 1}\right]\|\mathbf{y}^k - \mathbf{c}^*\|_2 \le \|\mathbf{y}^k - \mathbf{c}^*\|_2,\end{aligned}$$

where the last inequality follows from (4.4) and the fact that

$$\tau(\alpha_1 + \alpha_2)\delta \le 1/2, \quad \tau\alpha_4\delta^{\beta_2 - 1} \le 1/2.$$

By (4.22), we have

$$\|\mathbf{y}^{k+1} - \mathbf{c}^*\|_2 \leq \tau(\alpha_2 + \alpha_3)\delta^{\beta_1 - 1}\|\mathbf{c}^{k+1} - \mathbf{c}^*\|_2 \leq \|\mathbf{c}^{k+1} - \mathbf{c}^*\|_2.$$

So (4.5) is also true for $k+1$. Thus by mathematical induction, we have proved that (4.5), (4.6) and (4.7) are true for all positive integers k. Therefore by Proposition 4.1, Theorem 4.1 is established. □

REMARK 4.1. By (2.12) and (2.16), we have $2 < \beta_1\beta_2 \leq 3$. Hence (4.24) shows that Algorithm 3 is at least super quadratically convergent.

Next we carry on the convergence of Algorithm 2. Note that if the residuals $\mathbf{t}_i^k = \mathbf{s}_i^k = \mathbf{r}(\mathbf{c}^k) = \mathbf{r}(\mathbf{y}^k) = 0$ in Algorithm 3, then Algorithm 3 reduces to Algorithm 2. Thus for Algorithm 2, (4.22) and (4.23) reduce to

$$\|\mathbf{y}^k - \mathbf{c}^*\|_2 \leq \tau\alpha_1\|\mathbf{c}^k - \mathbf{c}^*\|_2^2, \tag{4.28}$$

$$\|\mathbf{c}^{k+1} - \mathbf{c}^*\|_2 \leq \tau\alpha_1\|\mathbf{y}^k - \mathbf{c}^*\|_2^2 + \tau\alpha_2\|\mathbf{c}^k - \mathbf{c}^*\|_2\|\mathbf{y}^k - \mathbf{c}^*\|_2, \tag{4.29}$$

respectively. It follows from (4.28) and (4.29) that

$$\|\mathbf{c}^{k+1} - \mathbf{c}^*\|_2 \leq \left[\tau^3\alpha_1^3\|\mathbf{c}^k - \mathbf{c}^*\|_2 + \tau^2\alpha_1\alpha_2)\right]\|\mathbf{c}^k - \mathbf{c}^*\|_2^3.$$

Thus we use the similar techniques in the proofs of Proposition 4.1 and Theorem 4.1 to get the following result, which shows that Algorithm 2 converges cubically.

THEOREM 4.2. *Let given eigenvalues $\{\lambda_i^*\}_{i=1}^n$ be distinct and the Jacobian matrix $J(\mathbf{c}^*)$ be nonsingular. Then Algorithm 2 is locally convergent with convergence rate of three order.*

If $\mathbf{p}_i(\mathbf{c}^k)$ and $\mathbf{p}_i(\mathbf{y}^k)$ are the eigenvectors of $A(\mathbf{c}^k)$ and $A(\mathbf{y}^k)$ in Algorithm 2, respectively, i.e., $\mathbf{p}_i(\mathbf{c}^k) = \mathbf{q}_i(\mathbf{c}^k)$, $\mathbf{p}_i(\mathbf{y}^k) = \mathbf{q}_i(\mathbf{y}^k)$, then Algorithm 2 reduces to Algorithm 1. Hence we also get the following result.

THEOREM 4.3. *Let given eigenvalues $\{\lambda_i^*\}_{i=1}^n$ be distinct and the Jacobian matrix $J(\mathbf{c}^*)$ be nonsingular. Then Algorithm 1 is locally convergent with convergence rate of three order.*

5. Numerical experiments. In this section, we report some numerical tests to illustrate the effectiveness of our proposed methods. We compare the numerical performance of Algorithms 1–3 with those of the Newton type methods in [7, 8, 9]. For the completeness of our presentation, we recall the Newton, Newton-like method and inexact Newton-like methods as follows.

Algorithm 4: the Newton method.

For $k = 0$ until convergence do

(a) Same as (a) in Algorithm 1.

(b) Same as (b) in Algorithm 1.

(c) Solve $\mathbf{c}^{k+1}$ from the Jacobian equation: $J(\mathbf{c}^k)\mathbf{c}^{k+1} = \lambda^* - \mathbf{b}^k$.

Algorithm 5: The Newton-like Method.

1. Given $\mathbf{c}^0$, iterate Algorithm 4 once to obtain $\mathbf{c}^1$. Let
$$P(\mathbf{c}^0) = [\mathbf{p}_1(\mathbf{c}^0), \cdots, \mathbf{p}_n(\mathbf{c}^0)] = Q(\mathbf{c}^0).$$
2. For $k = 1$ until convergence, do:
   (a) Compute $\mathbf{v}_i^k$ by the one-step inverse power method:
   $$(A(\mathbf{c}^k) - \lambda_i^* I)\mathbf{v}_i^k = \mathbf{p}_i(\mathbf{c}^{k-1}), \quad 1 \le i \le n.$$
   (b) Same as (b) in Algorithm 2.
   (c) Same as (c) in Algorithm 2.
   (d) Solve $\mathbf{c}^{k+1}$ from the approximate Jacobian equation: $J_k\mathbf{c}^{k+1} = \lambda^* - \hat{\mathbf{b}}^k$.

Algorithm 6: The inexact Newton-like method.

1. Same as the step 1 in Algorithm 5.
2. For $k = 1$ until convergence, do
   (a) Solve $\mathbf{v}_i^k$ inexactly in the one-step inverse power method
   $$(A(\mathbf{c}^k) - \lambda_i^* I)\mathbf{v}_i^k = \mathbf{p}_i(\mathbf{c}^{k-1}) + \mathbf{t}_i^k, \quad 1 \le i \le n,$$
   until the residual $\mathbf{t}_i$ satisfies $\|\mathbf{t}_i^k\|_2 \le 1/4$.
   (b) Same (b) as Algorithm 3.
   (c) Same (c) as Algorithm 3.
   (d) Solve $\mathbf{c}^{k+1}$ inexactly from the approximate Jacobian equation
   $$J_k\mathbf{c}^{k+1} = \lambda^* - \hat{\mathbf{b}}^k + \mathbf{r}^k,$$
   until the residual $\mathbf{r}^k$ satisfies
   $$\|\mathbf{r}^k\|_2 \le \left(\max_{1\le i\le n} \frac{1}{\| \mathbf{v}_i^k \|_2}\right)^{\beta}, \quad 1 < \beta \le 2.$$

EXAMPLE 5.1. Consider the Sturm-Liouville problem:

$$-u'' + q(x)u = \lambda u, \quad u(0) = u(\pi) = 0. \tag{5.1}$$

The inverse Sturm-Liouville problem is to determine $q(x)$ from λ. By the central difference scheme with uniform mesh $h = \pi/(n+1)$, the differential equation (5.1) is reduced to the matrix eigenproblem with tridiagonal structure:

$$(A_0 + h^2 X)\mathbf{u} = h^2\lambda\mathbf{u}, \tag{5.2}$$

where A_0 is the Laplacian matrix with zero boundary condition and X is a diagonal matrix representing the discretization of $q(x)$. The discrete analogue of the inverse Sturm-Liouville problem is an inverse eigenvalue problem. It is to determine the diagonal matrix X so that the matrix on the left hand side of (5.2) possesses a prescribed spectrum. Let $A_j = h^2\mathbf{e}_j\mathbf{e}_j^T$, for $j = 1, \ldots, n$, where $\mathbf{e}_j$ is the j-th unit n-vector. Thus we have the form (1.1) with $X = \mathrm{diag}(\mathbf{c})$. Given the exact solution $\mathbf{c}^*$ with the entries $[\mathbf{c}^*]_i = \mathrm{e}^{3ih}, 1 \le i \le n$, i.e. $q(x) = \mathrm{e}^{3x}$. We use the eigenvalues of

$A(\mathbf{c}^*)$ as the given eigenvalues $\{\lambda_i^*\}_{i=1}^n$. The initial guess $\mathbf{c}^0$ is formed by chopping the components of $\mathbf{c}^*$, i. e., $\mathbf{c}^0 = \mathbf{ceil}(10 \times \mathbf{c}^*)/10$.

In this test, the linear systems of Algorithms 1,2,4,5 and the step 1 in Algorithms 3 and 6 were all solved by the Matlab-provided **inv** function. The linear systems of the step 2 in Algorithms 3 and 6 were all solved by the Matlab-provided **qmr**(A, **b**, tol, 400,I,I,**x**0) function. For the inverse power equations (2.5) and (2.9), we use $\mathbf{u}_i^{k-1}$ and $\mathbf{v}_i^{k-1}$ as the initial guesses, respectively. For the approximate Jacobian equations (2.7) and (2.11), we use $\mathbf{c}^k$ and $\mathbf{y}^k$ as the initial guessses, respectively.

We report our experiment results. Tables 1-3 list the error values: error_$\mathbf{c}$ $= \|\mathbf{c}^k - \mathbf{c}^*\|_2$ and the residual values: error_λ $= \|P(\mathbf{c}^k)^T A(\mathbf{c}^k) P(\mathbf{c}^k) - \mathrm{diag}(\lambda^*)\|_F$ of Algorithms 1-6 for $n = 20$ in one test. From Tables 1-3, we can conclude that Algorithms 1-3 converges faster than Algorithms 4-6, respectively. Table 4 lists the CPU time of Algorithms 1-6 for $n = 30 : 5 : 50$ in one test. If the error value satisfies

$$\|\mathbf{c}^k - \mathbf{c}^*\|_2 \leq 10^{-10},$$

the outer iterations are stopped. Table 4 shows that Algorithms 1-3 consume less CPU time than Algorithms 4-6, respectively, when n is larger and larger, where the notation $\times$ denotes the corresponding algorithm fails to converge.

Table 1. Comparing Algorithm 1 and Algorithm 4

iterations		0	1	2	3	4
Algorithm 1	error_**c**	$2.50e-1$	$2.54e-6$	$6.34e-12$	$9.44e-12$	$3.27e-12$
	error_λ	$5.40e-3$	$1.77e-9$	$2.27e-13$	$1.46e-13$	$1.67e-13$
Algorithm 4	error_**c**	$2.50e-1$	$2.96e-4$	$1.00e-8$	$9.01e-12$	$1.22e-11$
	error_λ	$5.40e-3$	$2.43e-7$	$4.59e-12$	$1.26e-13$	$1.72e-13$

Table 2. Comparing Algorithm 2 and Algorithm 5

iterations		0	1	2	3	4
Algorithm 2	error_**c**	$2.50e-1$	$2.54e-6$	$1.07e-11$	$1.22e-11$	$1.18e-11$
	error_λ	$5.40e-3$	$1.77e-9$	$8.99e-14$	$1.17e-13$	$8.83e-14$
Algorithm 5	error_**c**	$2.50e-1$	$2.96e-4$	$1.00e-8$	$1.32e-11$	$1.17e-11$
	error_λ	$5.40e-3$	$2.43e-7$	$4.58e-12$	$6.23e-14$	$4.22e-14$

Table 3. Comparing Algorithm 3 with $(\beta_1, \beta_2) = (1.5, 1.6)$ and Algorithm 6 with $\beta = 1.8$

iterations		0	1	2	3	4
Algorithm 3	error_**c**	$2.50e-1$	$2.54e-6$	$1.31e-11$	$1.31e-11$	$1.31e-11$
	error_λ	$5.40e-3$	$1.79e-9$	$4.85e-7$	$2.00e-7$	$1.86e-7$
Algorithm 6	error_**c**	$2.50e-1$	$2.96e-4$	$9.99e-9$	$9.99e-9$	$9.90e-12$
	error_λ	$5.40e-3$	$2.49e-7$	$1.41e-6$	$3.05e-12$	$4.14e-11$

Table 4. CPU time of Algorithms 1–6 with $(\beta_1, \beta_2) = (1.5, 1.6)$ and $\beta = 1.8$

n	30	35	40	45	50
Algorithm 1	$1.04e-2$	$2.00e-2$	$2.46e-2$	$3.06e-2$	$4.14e-2$
Algorithm 4	$1.47e-2$	$2.37e-2$	$3.02e-2$	$3.98e-2$	$6.98e-2$
Algorithm 2	$1.64e-2$	$2.42e-2$	$3.09e-2$	$4.61e-2$	$5.86e-2$
Algorithm 5	$2.01e-2$	$2.98e-2$	$4.87e-2$	$6.25e-2$	$8.22e-2$
Algorithm 3	$9.12e-1$	3.08	4.09	1.77	5.63
Algorithm 6	$9.14e-1$	2.43	15.91	10.52	$\times$

EXAMPLE 5.2. In this tests, we compare the convergence rate of Algorithm 2 with that of Algorithm 3 to illustrate the oversolving problem. We use Toeplitz matrices

as our A_i in (1.1):

$$A_0 = O, A_1 = I, A_2 = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ 1 & 0 & 1 & \ddots & \vdots \\ 0 & 1 & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & 0 & 1 \\ 0 & \cdots & 0 & 1 & 0 \end{pmatrix}, \cdots, A_n = \begin{pmatrix} 0 & 0 & \cdots & 0 & 1 \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & \ddots & \ddots & 0 \\ 1 & 0 & \cdots & 0 & 0 \end{pmatrix}$$

In the tests, we tried Algorithms 2 and 3 on eleven 60×60 matrices. We first generate $\mathbf{c}^*$ with entries randomly chosen between 0 and 10, i.e., $\mathbf{c}^* = 10 \times \mathbf{rand}(n, 1)$. Then we compute the eigenvalues of $\{\lambda_i^*\}_{i=1}^n$ of $A(\mathbf{c}^*)$. The initial guess $\mathbf{c}^0$ is formed by chopping the components of $\mathbf{c}^*$ to two decimal places, i.e., $\mathbf{c}^0 = \mathbf{ceil}(100 \times \mathbf{c}^*)/100$. For Algorithm 2 and Algorithm 3, the stopping tolerance for the outer iterations is 10^{-10}. The inner systems in Algorithms 2 and 3 are all solved by the Matlab-provided **qmr**(A, **b**, tol, 400, I, I, **x**0) function. The stopping tolerances for the linear systems of Algorithm 2 is 10^{-13} and for those of Algorithm 3, they will be as given in the equations.

Table 5 gives the average numbers of outer iterations for the ten test matrices with respect to different choices (β_1, β_2), and give the average number of outer iterations for Algorithm 2. We see that the number of the outer iterations in Algorithm 3 is close to the one in Algorithm 2 when $\beta_1\beta_2$ tends to 3. In table 6, we give the total numbers of inner iterations, averaged over the ten test matrices. I_1 stands for the iterative numbers of the inverse power methods (2.5) and (2.9), and I_2 for the iterative numbers of the inverse power methods (2.7) and (2.13). J_1 stands for the iterative numbers of the approximate Jacobian equations (2.6) and (2.11), and J_2 for the iterative numbers of (2.8) and (2.15). We see that the linear systems in Algorithm 3 requires only less iterations than those in Algorithm 2.

Table 5. Average numbers of outer iterations for Algorithms 2 and 3.

(β_1, β_2)	(1.1, 1.9)	(1.2, 1.9)	(1.3, 1.9)	(1.4, 1.9)	(1.5, 1.9)	Algorithm 2
iterations	3.5	3.1	3.1	3.1	3.1	2.9

Table 6. Averaged total numbers of inner iterations in thousands for the inverse power methods and for the approximate Jacobian equations.

(β_1, β_2)	(1.1, 1.9)	(1.2, 1.9)	(1.3, 1.9)	(1.4, 1.9)	(1.5, 1.9)	Algorithm 2
I_1	11.87	10.68	8.56	8.13	9.09	12.37
I_2	11.80	12.94	12.14	12.74	12.86	14.43
J_1	0.24	0.24	0.23	0.24	0.27	0.37
J_2	0.36	0.34	0.26	0.27	0.22	0.26

6. Conclusions. In this paper, we have proposed two-step Newton type methods for solving the inverse eigenvalue problems. We show that the two-step Newton and two-step Newton-like methods converge cubically, and the inexact two-step Newton-like method converges super quadratically. Comparing with two Newton iterative steps with convergence of order 4, our methods only need forming one Jacobian matrix in each iterative step, which saves $O(n^4)$ flops. Numerical experiments is presented to illustrate that the two-step Newton type methods are effective.

Acknowledgement. We would like to thank Prof. Zheng-Jian Bai for his valuable discussion and to thank two anonymous referees for their valuable comments.

REFERENCES

[1] Andrew A L, *Some recent developments in inverse eigenvalue problems*, Computational Techniques and Applications CTAC93 ed D Stewart, H Gardner and D Singleton (Singapore: World Scientific), 1994, pp. 94–102.

[2] Andrew A L, *Numerical solution of inverse Sturm-Liouville problems,* ANZIAM J. 45(E)(2004) C326–C337.

[3] Bai Z J, *Inexact Newton methods for inverse eigenvalue problems*, Appl. Math. Comput., 172(2006), pp. 682–689.

[4] Bai Z J, Chan R H and Morini B, *An inexact Cayley transform methods for inverse eigenvalue problems,* Inverse Problems, 20(2004), pp. 1675–1689.

[5] Byrnes C I, *Pole placement by ouput feedback,* Three decades of Mathematics Systems Theory (Lecture Notes in Control and Inform. Sci. vol 135)(New York: Springer), 1989, pp. 31–78.

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

[7] Chan R H, Xu S F and Zhou H, *On the convergence rate of a quasi-Newton method for inverse eigenvalue problems.* SIAM J. Numer. Anal., 36(2)(1999), pp. 436–441.

[8] Chan R H, Chung H L and Xu S F, *The inexact Newton-like method for inverse eigenvalue problem.* BIT Numerical Mathematics, 43(1)(2003), pp. 7–20.

[9] Friedland S, Nocedal J and Overton M L, *The formulation and analysis of numerical methods for inverse eigenvalue problems.* SIAM J. Numer. Anal., 24(3)(1987), pp. 634–667.

[10] Gladwell G M L, *Inverse problems in vibration,* Appl. Mech. Rev., 39(1986), pp. 1013–1018.

[11] Gladwell G M L, *Inverse problems in vibration II,* Appl. Mech. Rev., 49(1996), pp. 25–34.

[12] Ortega J M and Rheinboldt W C, *Iterative Solution of Nonlinear Equations in Several Variables,* Academic Press, New York, 1970.

[13] Shapiro A, *On the unsolvability of inverse eigenvalue problems almost everywhere,* Linear Algebra Appl., 49(1983), pp. 27–31.

[14] Sun J G and Ye Q, *The unsolvability of inverse algebraic eigenvalue problems almost everywhere*, J. Comput. Math., 4(1986), pp. 212–226.

[15] Sun J G, *Eigenvalues of Rayleigh quotient matrices,* Numer. Math., 59(1991), pp. 603–614.

[16] Sun J G, *Backward errors for the inverse eigenvalue problem,* Numer. Math., 82 (1999), pp. 339–349.

[17] Stewart G W and Sun J G, *Matrix Perturbation Theory*, Academic, San Diego, 1990.

[18] Trench W F, *Numerical solution of the inverse eigenvalue problem for real symmetric Toeplitz matrices,* SIAM J. Sci. Comput., 18(6)(1997), pp. 1722–1736.

[19] Xu F S, *An introduction to inverse algebraic eigenvalue problems,* Peking University Press, 1998.