An Inexact Cayley Transform Method For Inverse Eigenvalue Problems

Zheng-Jian Bai* Raymond H. Chan* Benedetta Morini†

Abstract

The Cayley transform method is a Newton-like method for solving inverse eigenvalue problems. If the problem is large, one can solve the Jacobian equation by iterative methods. However, iterative methods usually oversolve the problem in the sense that they require far more (inner) iterations than is required for the convergence of the Newton (outer) iterations. In this paper, we develop an inexact version of the Cayley transform method. Our method can reduce the oversolving problem and improves the efficiency with respect to the exact version. We show that the convergence rate of our method is superlinear and that a good tradeoff between the required inner and outer iterations can be obtained.

Keywords. Nonlinear equation, inverse eigenvalue problem, Cayley transform
AMS subject classifications. 65F18, 65F10, 65F15.

1 Introduction

Inverse eigenvalue problems arise in a variety of applications, see for instances the pole assignment problem [5, 32], the inverse Toeplitz eigenvalue problem [8, 31, 35], the inverse Sturm-Liouville problem [1, 21], and also problems in applied mechanics and structure design [18, 19, 22], applied geophysics [30], applied physics [23], numerical analysis [27], and dynamics systems [14]. A good reference for these applications is the recent survey paper on structured inverse eigenvalue problems by Chu and Golub [10]. In many of these applications, the problem size n can be large. For example, large Toeplitz eigenvalue problems have been considered in [31]. Moreover, in the discrete inverse Sturm-Liouville problem, n is the number of grid-points, see Chu and Golub [10, p. 10]. Our goal in this paper is to derive an efficient algorithm for solving inverse eigenvalue problems when n is large.

Let us first define the notations. Let $\{A_k\}_{k=0}^n$ be $n+1$ real symmetric n-by-n matrices. For any $\mathbf{c} = (c_1, \ldots, c_n)^T \in \mathbb{R}^n$, let

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

*(zjbai, rchan@math.cuhk.edu.hk) Department of Mathematics, Chinese University of Hong Kong, Shatin, NT, Hong Kong. The research of the second author was partially supported by the Hong Kong Research Grant Council Grant CUHK4243/01P and CUHK DAG 2060257.

†(benedetta.morini@unifi.it) Dipartimento di Energetica 'S. Stecco' Università di Firenze, Via C. Lombroso 6/17, 50134 Firenze. Research was partially supported by MIUR, Rome, Italy, through "Cofinanziamenti Programmi di Ricerca Scientifica di Interesse Nazionale".

and denote the eigenvalues of $A(\mathbf{c})$ by $\{\lambda_i(\mathbf{c})\}_{i=1}^n$, where $\lambda_1(\mathbf{c}) \le \lambda_2(\mathbf{c}) \le \cdots \le \lambda_n(\mathbf{c})$. An inverse eigenvalue problem (IEP) is defined as follows: Given n real numbers $\lambda_1^* \le \cdots \le \lambda_n^*$, find $\mathbf{c} \in \mathbb{R}^n$ such that $\lambda_i(\mathbf{c}) = \lambda_i^*$ for $i = 1, \ldots, n$.

We note that the IEP can be formulated as a system of nonlinear equations

$$\mathbf{f}(\mathbf{c}) \equiv (\lambda_1(\mathbf{c}) - \lambda_1^*, \ldots, \lambda_n(\mathbf{c}) - \lambda_n^*)^T = \mathbf{0}. \tag{2}$$

It is easy to see that a direct application of Newton method to (2) requires the computation of $\lambda_i(\mathbf{c})$ at each iteration. To overcome the drawback, different Newton-like methods for solving (2) are given in [17]. One of these methods, Method III, forms an approximate Jacobian equation by applying matrix exponentials and Cayley transforms. As noted in [7], the method is particularly interesting and it has been used or cited in [8, 9, 25, 33] for instances.

If (2) is solved by Newton-like methods, then in each Newton iteration (the outer iteration), we need to solve the approximate Jacobian equation. When n is large, solving such a linear system will be costly. The cost can be reduced by using iterative methods (the inner iterations). Although iterative methods can reduce the complexity, they may oversolve the approximate Jacobian equation in the sense that the last tens or hundreds inner iterations before convergence may not improve the convergence of the outer Newton iterations [13]. In order to alleviate the oversolving problem, we propose in this paper an inexact Newton-like method for solving the nonlinear system (2). The inexact Newton-like method is a method that stops the inner iterations before convergence. By choosing suitable stopping criteria, we can minimize the oversolving problem and therefore reduce the total cost of the whole inner-outer iterations. In essence, one does not need to solve the approximate Jacobian equation exactly in order that the Newton method converges fast.

In this paper, we give an inexact version of Method III where the approximate Jacobian equation is solved inexactly by stopping the inner iterations before convergence. We propose a new criterion to stop the inner iterations at each Newton step and provide theoretical and experimental results for the procedure. First, we will show that the convergence rate of our method is superlinear. Then, we illustrate by numerical examples that it can avoid the oversolving problem and thereby reduce the total cost of the inner-outer iterations.

We remark that our proposed method is locally convergent. Thus, how to select the initial guess becomes a crucial problem. However, global continuous methods such as the homotopy method can be used in conjunction with our procedure. In these continuous methods, our inexact method can be used as the corrector step where a valid starting point is provided by the globalization strategy, see for examples [3] and [37, pp. 256–262].

This paper is organized as follows. In §2, we recall Method III for solving the IEP. In §3, we introduce our inexact method. In §4, we give the convergence analysis of our method. In §5, we present numerical tests to illustrate our results.

2 The Cayley Transform Method

Method III in [17] is based on Cayley transforms. In this section, we briefly recall this method. Let $\mathbf{c}^*$ be a solution to the IEP. Then there exists an orthogonal matrix Q_* satisfying

$$Q_*^T A(\mathbf{c}^*) Q_* = \Lambda_*, \quad \Lambda_* = \operatorname{diag}(\lambda_1^*, \ldots, \lambda_n^*). \tag{3}$$

Suppose that $\mathbf{c}^k$ and Q_k are the current approximations of $\mathbf{c}^*$ and Q_* in (3) respectively and that Q_k is an orthogonal matrix. Define $e^{Z_k} \equiv Q_k^T Q_*$. Then Z_k is a skew-symmetric matrix and (3) can be written as

$$Q_k^T A(\mathbf{c}^*)Q_k = e^{Z_k}\Lambda_* e^{-Z_k} = (I + Z_k + \frac{1}{2}(Z_k)^2 + \cdots)\Lambda_*(I - Z_k + \frac{1}{2}(Z_k)^2 + \cdots).$$

Thus $Q_k^T A(\mathbf{c}^*)Q_k = \Lambda_* + Z_k\Lambda_* - \Lambda_* Z_k + O(\|Z_k\|^2)$, where $\|\cdot\|$ denotes the 2-norm.

In Method III, $\mathbf{c}^k$ is updated by neglecting the second order terms in Z_k, i.e.

$$Q_k^T A(\mathbf{c}^{k+1})Q_k = \Lambda_* + Z_k\Lambda_* - \Lambda_* Z_k. \tag{4}$$

We find $\mathbf{c}^{k+1}$ by equating the diagonal elements in (4), i.e. $\mathbf{c}^{k+1}$ is given by

$$(\mathbf{q}_i^k)^T A(\mathbf{c}^{k+1})\mathbf{q}_i^k = \lambda_i^*, \quad i = 1, \ldots, n, \tag{5}$$

where $\{\mathbf{q}_i^k\}_{i=1}^n$ are the column vectors of Q_k. By (1), (5) can be rewritten as a linear system

$$J^{(k)}\mathbf{c}^{k+1} = \boldsymbol{\lambda}^* - \mathbf{b}^{(k)}, \tag{6}$$

where $\boldsymbol{\lambda}^* \equiv (\lambda_1^*, \ldots, \lambda_n^*)^T$, and

$$\left[J^{(k)}\right]_{ij} = (\mathbf{q}_i^k)^T A_j \mathbf{q}_i^k, \quad i, j = 1, \ldots, n, \tag{7}$$

$$[\mathbf{b}^{(k)}]_i = (\mathbf{q}_i^k)^T A_0 \mathbf{q}_i^k, \quad i = 1, \ldots, n. \tag{8}$$

Once we get $\mathbf{c}^{k+1}$ from (6), we obtain Z_k by equating the off-diagonal elements in (4), i.e.

$$[Z_k]_{ij} = \frac{(\mathbf{q}_i^k)^T A(\mathbf{c}^{k+1})\mathbf{q}_j^k}{\lambda_j^* - \lambda_i^*}, \qquad 1 \le i \ne j \le n. \tag{9}$$

Finally we update Q_k by setting $Q_{k+1} = Q_k U_k$, where U_k is an orthogonal matrix constructed by the Cayley transform for e^{Z_k}, i.e.

$$U_k = (I + \frac{1}{2}Z_k)(I - \frac{1}{2}Z_k)^{-1}.$$

We summarize the algorithm here.

Algorithm I: Cayley Transform Method

1. *Given* $\mathbf{c}^0$, *compute the orthonormal eigenvectors* $\{\mathbf{q}_i(\mathbf{c}^0)\}_{i=1}^n$ *of* $A(\mathbf{c}^0)$. *Let* $Q_0 = [\mathbf{q}_1^0, \ldots, \mathbf{q}_n^0] = [\mathbf{q}_1(\mathbf{c}^0), \ldots, \mathbf{q}_n(\mathbf{c}^0)]$.
2. *For* $k = 0, 1, 2, \ldots$, *until convergence, do:*
   - *(a) Form the approximate Jacobian matrix* $J^{(k)}$ *by (7) and* $\mathbf{b}^{(k)}$ *by (8).*
   - *(b) Solve* $\mathbf{c}^{k+1}$ *from the approximate Jacobian equation (6).*
   - *(c) Form the skew-symmetric matrix* Z_k *by (9).*
   - *(d) Compute* $Q_{k+1} = [\mathbf{q}_1^{k+1}, \ldots, \mathbf{q}_n^{k+1}] = [\mathbf{w}_1^{k+1}, \ldots, \mathbf{w}_n^{k+1}]^T$ *by solving*

$$(I + \frac{1}{2}Z_k)\mathbf{w}_j^{k+1} = \mathbf{g}_j^k, \quad j = 1, \cdots, n, \tag{10}$$

where $\mathbf{g}_j^k$ *is the* j*th column of* $G_k = (I - \frac{1}{2}Z_k)Q_k^T$.

This method was proved to converge quadratically in [17]. Note that in each outer iteration (i.e. Step 2), we have to solve the linear systems (6) and (10). When the systems are large, we may reduce the computational cost by solving both systems iteratively. One could expect that it requires only a few iterations to solve (10) iteratively. This is due to the fact that, as $\{\mathbf{c}^k\}$ converges to $\mathbf{c}^*$, $\|Z_k\|$ converges to zero, see [17, Equation (3.64)]. Consequently, the coefficient matrix on the left hand side of (10) approaches the identity matrix in the limit, and therefore (10) can be solved efficiently by iterative methods. On the other hand, iterative methods may oversolve the approximate Jacobian equation (6), in the sense that for each outer Newton iteration, the last few inner iterations may not contribute much to the convergence of the outer iterations. How to stop the inner iterations efficiently is the focus of our next section.

3 The Inexact Cayley Transform Method

The main aim of this paper is to propose an efficient version of Algorithm I for large problems. To reduce the computational cost, we solve both (6) and (10) iteratively with (6) being solved inexactly. First, we derive a computable stopping criterion for (6), then we establish the convergence rate of the resulting procedure.

For general nonlinear equation $\mathbf{f}(\mathbf{c}) = \mathbf{0}$, the stopping criterion of inexact Newton methods is usually given in terms of $\mathbf{f}(\mathbf{c})$, see for instances [13, 15, 26]. By (2), this will involve computing $\lambda_i(\mathbf{c}^k)$ of $A(\mathbf{c}^k)$ which are costly to compute. Our idea is to replace them by the Rayleigh quotients, see (14) and (16) below. We will prove in §4 that this replacement will retain superlinear convergence.

Algorithm II: Inexact Cayley Transform Method

1. *Given $\mathbf{c}^0$, compute the orthonormal eigenvectors $\{\mathbf{q}_i(\mathbf{c}^0)\}_{i=1}^n$ and the eigenvalues $\{\lambda_i(\mathbf{c}^0)\}_{i=1}^n$ of $A(\mathbf{c}^0)$. Let $P_0 = [\mathbf{p}_1^0, \dots, \mathbf{p}_n^0] = [\mathbf{q}_1(\mathbf{c}^0), \dots, \mathbf{q}_n(\mathbf{c}^0)]$, and*

$$\boldsymbol{\rho}^0 = (\lambda_1(\mathbf{c}^0), \dots, \lambda_n(\mathbf{c}^0))^T.$$

2. *For $k = 0, 1, 2, \dots,$ until convergence, do:*

   (a) *Form the approximate Jacobian matrix J_k and $\mathbf{b}^k$ as follows:*

$$[J_k]_{ij} = (\mathbf{p}_i^k)^T A_j \mathbf{p}_i^k, \quad 1 \le i, j \le n, \tag{11}$$

$$[\mathbf{b}^k]_i = (\mathbf{p}_i^k)^T A_0 \mathbf{p}_i^k, \quad 1 \le i \le n. \tag{12}$$

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

$$J_k \mathbf{c}^{k+1} = \boldsymbol{\lambda}^* - \mathbf{b}^k + \mathbf{r}^k, \tag{13}$$

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

$$\|\mathbf{r}^k\| \le \frac{\|\boldsymbol{\rho}^k - \boldsymbol{\lambda}^*\|^\beta}{\|\boldsymbol{\lambda}^*\|^\beta}, \quad \beta \in (1, 2]. \tag{14}$$

   (c) *Form the skew-symmetric matrix Y_k:*

$$[Y_k]_{ij} = \frac{(\mathbf{p}_i^k)^T A(\mathbf{c}^{k+1})\, \mathbf{p}_j^k}{\lambda_j^* - \lambda_i^*}, \quad 1 \le i \ne j \le n.$$

(d) *Compute* $P_{k+1} = [\mathbf{p}_1^{k+1}, \ldots, \mathbf{p}_n^{k+1}] = [\mathbf{v}_1^{k+1}, \ldots, \mathbf{v}_n^{k+1}]^T$ *by solving*

$$(I + \frac{1}{2}Y_k)\mathbf{v}_j^{k+1} = \mathbf{h}_j^k, \quad j = 1, \cdots, n, \tag{15}$$

where $\mathbf{h}_j^k$ *is the* j*th column of* $H_k = (I - \frac{1}{2}Y_k)P_k^T$.

(e) *Compute* $\boldsymbol{\rho}^{k+1} = (\rho_1^{k+1}, \ldots, \rho_n^{k+1})^T$ *by*

$$\rho_i^{k+1} = (\mathbf{p}_i^{k+1})^T A(\mathbf{c}^{k+1})\, \mathbf{p}_i^{k+1}, \quad i = 1, \ldots, n. \tag{16}$$

Since P_0 is an orthogonal matrix and Y_k are skew-symmetric matrices, we see that P_k so generated by the Cayley transform in (15) must be orthogonal, i.e.

$$P_k^T P_k = I, \quad k = 0, 1, \ldots. \tag{17}$$

To maintain the orthogonality of P_k, that would mean that (15) cannot be solved inexactly. However, we will see in §4 that $\|Y_k\|$ converges to zero as $\mathbf{c}^k$ converges to $\mathbf{c}^*$ (see (35) and (44)). Consequently, the matrix on the left hand side of (15) approaches the identity matrix in the limit. Therefore we can expect to solve (15) accurately by iterative methods using just a few iterations.

The expensive step in Algorithm II will be the solution of (13). The aim of our next section is to show that with our stopping criterion in (14), the convergence rate of Algorithm II is equal to β given in (14).

4 Convergence Analysis

In the following, we let $\mathbf{c}^k$ be the kth iterate produced by Algorithm II, and $\{\lambda_i(\mathbf{c}^k)\}_{i=1}^n$ and $\{\mathbf{q}_i(\mathbf{c}^k)\}_{i=1}^n$ be the eigenvalues and normalized eigenvectors of $A(\mathbf{c}^k)$. We let $Q_* = [\mathbf{q}_1(\mathbf{c}^*), \ldots, \mathbf{q}_n(\mathbf{c}^*)]$ be the orthogonal matrix of the eigenvectors of $A(\mathbf{c}^*)$. Moreover, we define

$$E_k \equiv P_k - Q_*, \tag{18}$$

the error matrix at the kth outer iteration. As in [17], we assume that the given eigenvalues $\{\lambda_i^*\}_{i=1}^n$ are distinct and that the Jacobian $J(\mathbf{c}^*)$ defined by

$$\Big[J(\mathbf{c}^*)\Big]_{ij} \equiv \mathbf{q}_i(\mathbf{c}^*)^T A_j \mathbf{q}_i(\mathbf{c}^*), \quad 1 \le i, j \le n, \tag{19}$$

is nonsingular.

4.1 Preliminary Lemmas

In this subsection, we prove some preliminary results which are necessary for the convergence analysis of our method. First we list three lemmas that are already proven in other papers.

Lemma 1 *Let the given eigenvalues* $\{\lambda_i^*\}_{i=1}^n$ *be distinct and* $\mathbf{q}_i(\mathbf{c}^*)$ *be the normalized eigenvectors of* $A(\mathbf{c}^*)$ *corresponding to* λ_i^* *for* $i = 1, \ldots, n$. *Then there exist positive numbers* δ_0 *and* τ_0 *such that, if* $\|\mathbf{c}^k - \mathbf{c}^*\| \le \delta_0$, *we get*

$$\|\mathbf{q}_i(\mathbf{c}^k) - \mathbf{q}_i(\mathbf{c}^*)\| \le \tau_0 \|\mathbf{c}^k - \mathbf{c}^*\|, \quad 1 \le i \le n. \tag{20}$$

Proof: It follows from the analyticity of eigenvectors corresponding to simple eigenvalues, see for instances [37, p. 249, Equation (4.6.13)]. □

Lemma 2 *Let J_k, $J(\mathbf{c}^*)$ and E_k be defined as in (11), (19) and (18) respectively. Then $\|J_k - J(\mathbf{c}^*)\| = O(\|E_k\|)$. Hence if $J(\mathbf{c}^*)$ is nonsingular, then there exist positive numbers ϵ_0 and τ_1 such that if $\|E_k\| \leq \epsilon_0$, then J_k is nonsingular and*

$$\|J_k^{-1}\| \leq \tau_1. \tag{21}$$

Proof: The first part follows easily from the formula of J_k and $J(\mathbf{c}^*)$, and the second part follows from the continuity of matrix inverses, cf. [6] or [37, p. 249, Equation (4.6.11)]. □

Lemma 3 [17, Corollary 3.1] *There exist two positive numbers ϵ_1 and τ_2 such that, if $\|E_k\| \leq \epsilon_1$, the skew-symmetric matrix X_k defined by $e^{X_k} \equiv P_k^T Q_*$ satisfies $\|X_k\| \leq \tau_2 \|E_k\|$.*

We now express our stopping criteria (14) in terms of $\|\mathbf{c}^k - \mathbf{c}^*\|$ and $\|E_k\|$.

Lemma 4 *Let the given eigenvalues $\{\lambda_i^*\}_{i=1}^n$ be distinct and $\boldsymbol{\rho}^k$ be given by (16). Then for $k \geq 0$,*

$$\|\boldsymbol{\rho}^k - \boldsymbol{\lambda}^*\| = O(\|\mathbf{c}^k - \mathbf{c}^*\| + \|E_k\|). \tag{22}$$

Proof: By (16), $\rho_i^k = (\mathbf{p}_i^k)^T A(\mathbf{c}^k)\mathbf{p}_i^k$. For $1 \leq i \leq n$, we write

$$|\rho_i^k - \lambda_i^*| \leq |(\mathbf{p}_i^k)^T A(\mathbf{c}^k)\mathbf{p}_i^k - (\mathbf{p}_i^k)^T A(\mathbf{c}^*)\mathbf{p}_i^k| + |(\mathbf{p}_i^k)^T A(\mathbf{c}^*)\mathbf{p}_i^k - \lambda_i^*|. \tag{23}$$

We claim that each term in the right hand side of (23) is bounded by $O(\|\mathbf{c}^k - \mathbf{c}^*\| + \|E_k\|)$. For the first term, by (1) and (17), we have

$$|(\mathbf{p}_i^k)^T A(\mathbf{c}^k)\mathbf{p}_i^k - (\mathbf{p}_i^k)^T A(\mathbf{c}^*)\mathbf{p}_i^k| = |(\mathbf{p}_i^k)^T \sum_{j=1}^n (c_j^k - c_j^*) A_j \mathbf{p}_i^k| = O(\|\mathbf{c}^k - \mathbf{c}^*\|). \tag{24}$$

For the second term, we have

$$\begin{aligned}
& |(\mathbf{p}_i^k)^T A(\mathbf{c}^*)\mathbf{p}_i^k - \lambda_i^*| \\
= \; & |(\mathbf{p}_i^k)^T A(\mathbf{c}^*)\mathbf{p}_i^k - (\mathbf{q}_i(\mathbf{c}^*))^T A(\mathbf{c}^*)\mathbf{q}_i(\mathbf{c}^*)| \\
\leq \; & |(\mathbf{p}_i^k)^T A(\mathbf{c}^*)\mathbf{p}_i^k - (\mathbf{q}_i(\mathbf{c}^*))^T A(\mathbf{c}^*)\mathbf{p}_i^k| + |(\mathbf{q}_i(\mathbf{c}^*))^T A(\mathbf{c}^*)\mathbf{p}_i^k - (\mathbf{q}_i(\mathbf{c}^*))^T A(\mathbf{c}^*)\mathbf{q}_i(\mathbf{c}^*)| \\
\leq \; & (\|\mathbf{p}_i^k\| + \|\mathbf{q}_i(\mathbf{c}^*)\|)\|A(\mathbf{c}^*)\|\|\mathbf{q}_i(\mathbf{c}^*) - \mathbf{p}_i^k\| \leq O(\|\mathbf{p}_i^k - \mathbf{q}_i(\mathbf{c}^*)\|).
\end{aligned}$$

Since $[\mathbf{p}_i^k - \mathbf{q}_i(\mathbf{c}^*)]$ is the ith column of E_k, $\|\mathbf{p}_i^k - \mathbf{q}_i(\mathbf{c}^*)\| \leq \|E_k\|$, and we have

$$|(\mathbf{p}_i^k)^T A(\mathbf{c}^*)\mathbf{p}_i^k - \lambda_i^*| = O(\|E_k\|), \quad 1 \leq i \leq n. \tag{25}$$

Putting (24) and (25) into (23), we have (22). □

As remarked already, the main difference between Algorithm II and Algorithm I is that we solve (13) approximately rather than exactly as in (6). Thus by comparing with (4), we see that the matrix Y_k and vector $\mathbf{c}^{k+1}$ of Algorithm II are defined by

$$\Lambda_* + Y_k\Lambda_* - \Lambda_* Y_k = P_k^T A(\mathbf{c}^{k+1})P_k - R_k, \tag{26}$$

where $R_k = \text{diag}([\mathbf{r}^k]_1, \ldots, [\mathbf{r}^k]_n)$ and $[\mathbf{r}^k]_i$ is the ith entry of the residual vector $\mathbf{r}^k$ given in (13). Using (26), we can estimate $\|\mathbf{c}^{k+1} - \mathbf{c}^*\|$ and $\|E_{k+1}\|$ in terms of $\|\mathbf{c}^k - \mathbf{c}^*\|$ and $\|E_k\|$.

Lemma 5 *Let the given eigenvalues $\{\lambda_i^*\}_{i=1}^n$ be distinct and the Jacobian $J(\mathbf{c}^*)$ defined in (19) be nonsingular. Then there exist two positive numbers δ_1 and ϵ_2 such that the conditions $\|\mathbf{c}^k - \mathbf{c}^*\| \le \delta_1$ and $\|E_k\| \le \epsilon_2$ imply*

$$\begin{aligned} \|\mathbf{c}^{k+1} - \mathbf{c}^*\| &= O(\|\boldsymbol{\rho}^k - \boldsymbol{\lambda}^*\|^\beta + \|E_k\|^2), & (27)\\ \|E_{k+1}\| &= O(\|\mathbf{c}^{k+1} - \mathbf{c}^*\| + \|E_k\|^2). & (28)\end{aligned}$$

Proof: Let X_k be defined by $e^{X_k} \equiv P_k^T Q_*$. By Lemma 3, if $\|E_k\| \le \epsilon_1$, then

$$\|X_k\| = O(\|E_k\|). \tag{29}$$

By (3), $e^{X_k}\Lambda_* e^{-X_k} = P_k^T A(\mathbf{c}^*)P_k$. Hence, if $\|E_k\|$ is small enough, we have

$$\Lambda_* + X_k\Lambda_* - \Lambda_* X_k = P_k^T A(\mathbf{c}^*)P_k + O(\|E_k\|^2). \tag{30}$$

Subtracting (26) from (30), we have

$$(X_k - Y_k)\Lambda_* - \Lambda_*(X_k - Y_k) = P_k^T(A(\mathbf{c}^*) - A(\mathbf{c}^{k+1}))P_k + R_k + O(\|E_k\|^2). \tag{31}$$

Equating the diagonal elements yields

$$J_k(\mathbf{c}^{k+1} - \mathbf{c}^*) = \mathbf{r}^k + O(\|E_k\|^2),$$

where J_k is defined by (11). Thus if $\|E_k\|$ is sufficiently small, then by (21) and (14), we get (27).

To get (28), we note from (15) that

$$\begin{aligned} E_{k+1} &= P_{k+1} - Q^* \\ &= P_k\left[(I + \frac{1}{2}Y_k)\,(I - \frac{1}{2}Y_k)^{-1} - e^{X_k}\right] \\ &= P_k\left[(I + \frac{1}{2}Y_k) - \big(I + X_k + O(\|X_k\|^2)\big)\,(I - \frac{1}{2}Y_k)\right](I - \frac{1}{2}Y_k)^{-1} \\ &= P_k\left[Y_k - X_k + O(X_kY_k + \|X_k\|^2)\right](I - \frac{1}{2}Y_k)^{-1}. \end{aligned}$$

Therefore by (17) and (29), we have

$$\|E_{k+1}\| \le \left[\|Y_k - X_k\| + O(\|E_k\|\|Y_k\| + \|E_k\|^2)\right]\|(I - \frac{1}{2}Y_k)^{-1}\|. \tag{32}$$

We now estimate the norms in the right hand side of (32) one by one. For $1 \le i \ne j \le n$, the off-diagonal equations of (31) give

$$[X_k]_{ij} - [Y_k]_{ij} = \frac{1}{\lambda_j^* - \lambda_i^*}(\mathbf{p}_i^k)^T(A(\mathbf{c}^*) - A(\mathbf{c}^{k+1}))\mathbf{p}_j^k + O(\|E_k\|^2).$$

It follows that

$$|[X_k]_{ij} - [Y_k]_{ij}| = O(\|\mathbf{c}^{k+1} - \mathbf{c}^*\| + \|E_k\|^2),$$

and hence

$$\|X_k - Y_k\| \le \|X_k - Y_k\|_F = O(\|\mathbf{c}^{k+1} - \mathbf{c}^*\| + \|E_k\|^2), \tag{33}$$

where $\|\cdot\|_F$ denotes the Frobenius norm. By (29) and (33),

$$\|Y_k\| = O(\|\mathbf{c}^{k+1} - \mathbf{c}^*\| + \|E_k\| + \|E_k\|^2). \tag{34}$$

By (27) and (22), we have

$$\|Y_k\| = O(\|\boldsymbol{\rho}^k - \boldsymbol{\lambda}^*\|^\beta + \|E_k\|) = O((\|\mathbf{c}^k - \mathbf{c}^*\| + \|E_k\|)^\beta + \|E_k\|). \tag{35}$$

Thus if $\|\mathbf{c}^k - \mathbf{c}^*\|$ and $\|E_k\|$ are sufficiently small, we have $\|Y_k\| \le 1$, and therefore

$$\|(I - \frac{1}{2}Y_k)^{-1}\| \le \frac{1}{1 - \frac{1}{2}\|Y_k\|} \le 2. \tag{36}$$

Finally, by putting (33), (34) and (36) into (32), we have (28). □

4.2 Convergence Rate of Algorithm II

In the following, we show that the root-convergence rate of our method is at least β. Here, we recall the definition of root-convergence, see [29, Chap. 9].

Definition 1 *Let $\{\mathbf{x}^k\}$ be a sequence with limit $\mathbf{x}^*$. Then the numbers*

$$R_p\{\mathbf{x}^k\} = \begin{cases} \limsup_{k\to\infty} \|\mathbf{x}^k - \mathbf{x}^*\|^{1/k}, & \text{if } p = 1, \\ \limsup_{k\to\infty} \|\mathbf{x}^k - \mathbf{x}^*\|^{1/p^k}, & \text{if } p > 1, \end{cases} \tag{37}$$

are the root-convergence factors of $\{\mathbf{x}^k\}$. The quantity

$$O_R(\mathbf{x}^*) = \begin{cases} \infty, & \text{if } R_p\{\mathbf{x}^k\} = 0, \forall p \in [1, \infty), \\ \inf\{p \in [1,\infty) | R_p\{\mathbf{x}^k\} = 1\}, & \text{otherwise}, \end{cases} \tag{38}$$

is called the root-convergence rate of $\{\mathbf{x}^k\}$.

We begin by proving that our method is locally convergent.

Theorem 1 *Let the given eigenvalues $\{\lambda_i^*\}_{i=1}^n$ be distinct and $J(\mathbf{c}^*)$ defined in (19) be nonsingular. Then there exists $\delta > 0$ such that if $\|\mathbf{c}^0 - \mathbf{c}^*\| \le \delta$, the sequence $\{\mathbf{c}^k\}$ generated by Algorithm II converges to $\mathbf{c}^*$.*

Proof: Suppose that $\|\mathbf{c}^k - \mathbf{c}^*\| \le \delta_1$, and $\|E_k\| \le \epsilon = \min\{1, \epsilon_2\}$, where δ_1 and ϵ_2 are given in Lemma 5. By Lemmas 4 and 5, there exists a constant $\mu > 1$ such that for any $k \ge 0$,

$$\begin{aligned} \|\boldsymbol{\rho}^k - \boldsymbol{\lambda}^*\| &\le \mu(\|\mathbf{c}^k - \mathbf{c}^*\| + \|E_k\|), & (39) \\ \|\mathbf{c}^{k+1} - \mathbf{c}^*\| &\le \mu(\|\boldsymbol{\rho}^k - \boldsymbol{\lambda}^*\|^\beta + \|E_k\|^2), & (40) \\ \|E_{k+1}\| &\le \mu(\|\mathbf{c}^{k+1} - \mathbf{c}^*\| + \|E_k\|^2). & (41) \end{aligned}$$

Putting (39) into (40), we have

$$\begin{aligned} \|\mathbf{c}^{k+1} - \mathbf{c}^*\| &\le \mu[\mu^\beta(\|\mathbf{c}^k - \mathbf{c}^*\| + \|E_k\|)^\beta + \|E_k\|^2] \\ &\le \mu[(2\mu)^\beta + 1] \max\left\{\|\mathbf{c}^k - \mathbf{c}\|^\beta, \|E_k\|^\beta\right\}. & (42) \end{aligned}$$

Putting (42) into (41), and using the fact that $\mu > 1$, we have

$$\begin{aligned} \|E_{k+1}\| &\leq 2\mu \max\left\{\|\mathbf{c}^{k+1} - \mathbf{c}\|, \|E_k\|^2\right\} \\ &\leq 2\mu^2[(2\mu)^\beta + 1] \max\left\{\|\mathbf{c}^k - \mathbf{c}\|^\beta, \|E_k\|^\beta\right\}. \qquad (43) \end{aligned}$$

Let $\varphi \equiv \max\{\tau_0\sqrt{n}, 2\mu^2[(2\mu)^\beta + 1]\} > 1$. Then by (42) and (43), we have

$$\max\left\{\|\mathbf{c}^{k+1} - \mathbf{c}\|, \|E_{k+1}\|\right\} \leq \varphi \max\left\{\|\mathbf{c}^k - \mathbf{c}\|^\beta, \|E_k\|^\beta\right\}, \quad k = 0, 1, \ldots. \qquad (44)$$

We now prove the theorem by using the mathematical induction. In particular, we show that if $\|\mathbf{c}^0 - \mathbf{c}^*\| \leq \delta$ where

$$\delta \equiv \min\left\{1, \delta_0, \delta_1, \frac{\epsilon}{\varphi}, \frac{1}{\varphi^{\beta^2/(\beta-1)^2}}\right\} < \epsilon, \qquad (45)$$

then for each $k \geq 1$, the following inequalities hold:

$$\begin{aligned} \max\{\|\mathbf{c}^k - \mathbf{c}^*\|, \|E_k\|\} &\leq \delta, \qquad (46) \\ \max\{\|\mathbf{c}^k - \mathbf{c}^*\|, \|E_k\|\} &\leq \varphi^{1+\beta+\cdots+\beta^k}\|\mathbf{c}^0 - \mathbf{c}^*\|^{\beta^k}. \qquad (47) \end{aligned}$$

We first note that from (20), we have

$$\|E_0\| \leq \sqrt{n}\max_i \|\mathbf{q}_i(\mathbf{c}^0) - \mathbf{q}_i(\mathbf{c}^*)\| \leq \tau_0\sqrt{n}\|\mathbf{c}^0 - \mathbf{c}^*\| \leq \varphi\|\mathbf{c}^0 - \mathbf{c}^*\|. \qquad (48)$$

Hence by using (45), $\|E_0\| \leq \varphi\|\mathbf{c}^0 - \mathbf{c}^*\| \leq \varphi\delta \leq \epsilon$.

We now verify (47) for $k = 1$. By (44) and (48),

$$\begin{aligned} \max\{\|\mathbf{c}^1 - \mathbf{c}^*\|, \|E_1\|\} &\leq \varphi \max\left\{\|\mathbf{c}^0 - \mathbf{c}\|^\beta, \|E_0\|^\beta\right\} \\ &\leq \varphi\|\mathbf{c}^0 - \mathbf{c}\|^\beta \max\left\{1, \varphi^\beta\right\} \leq \varphi^{1+\beta}\|\mathbf{c}^0 - \mathbf{c}\|^\beta. \qquad (49) \end{aligned}$$

Moreover, if we define $\zeta \equiv \varphi^{\frac{\beta}{\beta-1}}\delta$, then by (45),

$$\zeta^\beta \leq \delta. \qquad (50)$$

Hence by (49),

$$\max\{\|\mathbf{c}^1 - \mathbf{c}^*\|, \|E_1\|\} \leq \varphi^{1+\beta}\delta^\beta = (\varphi^{\frac{1+\beta}{\beta}}\delta)^\beta \leq (\varphi^{\frac{\beta}{\beta-1}}\delta)^\beta = \zeta^\beta \leq \delta.$$

Thus (46) holds for $k = 1$.

Next we assume that at the kth iteration, (46) and (47) hold. We first prove that (47) holds for $k + 1$. In fact, by (44) and (47) for k, we have

$$\begin{aligned} \max\{\|\mathbf{c}^{k+1} - \mathbf{c}^*\|, \|E_{k+1}\|\} &\leq \varphi \cdot \left(\varphi^{1+\beta+\cdots+\beta^k}\|\mathbf{c}^0 - \mathbf{c}^*\|^{\beta^k}\right)^\beta \\ &= \varphi^{1+\beta+\cdots+\beta^{k+1}}\|\mathbf{c}^0 - \mathbf{c}^*\|^{\beta^{k+1}}. \qquad (51) \end{aligned}$$

To prove that (46) holds for $k+1$, we use (51):

$$\begin{aligned}
\max\{\|\mathbf{c}^{k+1}-\mathbf{c}^*\|, \|E_{k+1}\|\} &\leq \left(\varphi^{\frac{1+\beta+\cdots+\beta^k+\beta^{k+1}}{\beta^{k+1}}}\|\mathbf{c}^0-\mathbf{c}^*\|\right)^{\beta^{k+1}} \\
&= \left(\varphi^{\left(\frac{1}{\beta^{k+1}}+\frac{1}{\beta^k}+\cdots+1\right)}\|\mathbf{c}^0-\mathbf{c}^*\|\right)^{\beta^{k+1}} \\
&\leq \left(\varphi^{\frac{\beta}{\beta-1}}\|\mathbf{c}^0-\mathbf{c}^*\|\right)^{\beta^{k+1}} \leq \zeta^{\beta^{k+1}}. \qquad (52)
\end{aligned}$$

By (50), we have $\zeta \leq \delta^{1/\beta} \leq 1$. Hence

$$\max\{\|\mathbf{c}^{k+1}-\mathbf{c}^*\|, \|E_{k+1}\|\} \leq \zeta^{\beta^{k+1}} \leq \zeta^{\beta} \leq \delta.$$

Thus we have proved that (46) and (47) hold for any $k \geq 1$. Moreover, from (52), we see that $\{\mathbf{c}^k\}$ converges to $\mathbf{c}^*$. □

We end this section by establishing the root convergence of our method.

Theorem 2 *Under the same conditions as in Theorem 1, the iterates $\{\mathbf{c}^k\}$ converges to $\mathbf{c}^*$ with root-convergence rate at least equal to β.*

Proof: By Theorem 1, we know that $\{\mathbf{c}^k\}$ converges to $\mathbf{c}^*$. From (52), we have for any $k \geq 1$, $\|\mathbf{c}^k-\mathbf{c}^*\| \leq \zeta^{\beta^k}$, where $\zeta < 1$. We now estimate the root-convergence factors of $\{\mathbf{c}^k\}$ defined in (37) for different values of p:

1. If $p=1$, then
$$R_1\{\mathbf{c}^k\} = \limsup_{k\to\infty} \|\mathbf{c}^k-\mathbf{c}^*\|^{1/k} \leq \limsup_{k\to\infty}\ \zeta^{\beta^k/k} = 0.$$

2. If $1<p<\beta$, then
$$R_p\{\mathbf{c}^k\} = \limsup_{k\to\infty} \|\mathbf{c}^k-\mathbf{c}^*\|^{1/p^k} \leq \limsup_{k\to\infty}\ \zeta^{(\beta/p)^k} = 0.$$

3. If $p=\beta$, then
$$R_\beta\{\mathbf{c}^k\} = \limsup_{k\to\infty} \|\mathbf{c}^k-\mathbf{c}^*\|^{1/\beta^k} \leq \zeta < 1.$$

4. If $p>\beta$, then
$$R_p\{\mathbf{c}^k\} = \limsup_{k\to\infty} \|\mathbf{c}^k-\mathbf{c}^*\|^{1/p^k} \leq \limsup_{k\to\infty}\ \zeta^{(\beta/p)^k} = 1.$$

Therefore, $R_p\{\mathbf{c}^k\}=0$ for any $p \in [1,\beta)$ and $R_p\{\mathbf{c}^k\} \leq 1$ for any $p \in [\beta,\infty)$. Thus according to (38), $O_R(\mathbf{c}^*) \geq \beta$. □

5 Numerical Experiments

In this section, we compare the numerical performance of Algorithm I with that of Algorithm II on two problems. The first one is the inverse Toeplitz eigenvalue problem, see [8, 31, 35], and the second one is the inverse Sturm-Liouville problem, see [17] and [10, p. 10]. Our aim is to illustrate the advantage of our method over Algorithm I in terms of minimizing the oversolving problem and the overall computational complexity.

Example 1. In this example, we use Toeplitz matrices as our A_i in (1):

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

Thus $A(\mathbf{c})$ is a symmetric Toeplitz matrix with the first column equals to $\mathbf{c}$.

In [31], very large inverse Toeplitz eigenvalue problem were solved on parallel architectures. Here we consider three problem sizes: $n = 100$, 200, and 300. For each value of n, we constructed ten n-by-n test problems where the exact solutions $\mathbf{c}^*$ are chosen randomly. Then we computed the eigenvalues $\{\lambda_i^*\}_{i=1}^n$ of $A(\mathbf{c}^*)$ as the prescribed eigenvalues. Since both algorithms are locally convergent, $\mathbf{c}^0$ was formed by chopping the components of $\mathbf{c}^*$ to four decimal places for $n = 100$ and to five decimal places for $n = 200$ and 300.

The linear systems (6), (10), (13), and (15) are solved iteratively by the QMR method [16] using the `Matlab`-provided `QMR` function. To guarantee the orthogonality of Q_k in (10) and P_k in (15), both systems are solved up to machine precision `eps` (which is $\approx 2.2\times10^{-16}$). We use the right-hand side vector as the initial guess for these two systems.

For the Jacobian systems (6) and (13), we use $\mathbf{c}^k$, the iterant at the kth iteration, as the initial guess for the iterative method at the $(k+1)$th iteration. We note that both systems are difficult to solve and one can use preconditioning to speed up the convergence. Here we have used the `Matlab`-provided Modified ILU (MILU) preconditioner: `LUINC(A,[drop-tolerance,1,1,1])` since the MILU preconditioner is one of the most versatile preconditioners for unstructured matrices [12, 20]. The drop tolerance we used is 0.05 for all the three problem sizes. We emphasize that, we are not attempting to find the best preconditioners for these systems, but trying to illustrate that preconditioning can be incorporated into both systems easily.

The inner loop stopping tolerance for (13) is given by (14). For (6) in Algorithm I, we are supposed to solve it up to machine precision `eps`. Here however, we first try to solve (6) with a larger stopping tolerance of 10^{-13} and compare the two algorithms. Later we will vary this and see how it affects the performance of Algorithm I. The outer iterations of Algorithms I and II are stopped when

$$\|Q_k^T A(\mathbf{c}^k) Q_k - \Lambda_*\|_F \le 10^{-10}, \quad \text{and} \quad \|P_k^T A(\mathbf{c}^k) P_k - \Lambda_*\|_F \le 10^{-10}. \tag{53}$$

In Table 1, we give the total numbers of outer iterations N_o averaged over the ten tests and the average total numbers of inner iterations N_i required for solving the approximate Jacobian equations. In the table, "I" and "P" respectively mean no preconditioner or the MILU preconditioner is used. We can see from Table 1 that N_o is small for Algorithm I and also for Algorithm II when $\beta \geq 1.5$. This confirms the theoretical convergence rate of the two algorithms. In terms of N_i, Algorithm II is more effective than Algorithm I for $\beta \approx 1.5$. We also note that the MILU preconditioner is quite effective for the Jacobian equations.

n			Alg. I	β in Alg. II									
				1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.0
100	I	N_o	3.2	12	5.2	4	3.3	3.2	3.2	3.2	3.2	3.2	3.2
		N_i	397	755	445	379	327	323	325	329	336	339	349
	P	N_o	3.2	7.7	5.3	4.2	3.7	3.2	3.2	3.2	3.2	3.2	3.2
		N_i	37.7	15.3	15.7	15.6	19	17.9	20.8	24	27.4	28.1	30.9
200	I	N_o	3	10.9	6	4	3	3	3	3	3	3	3
		N_i	818	1444	1144	855	684	719	725	732	738	747	763
	P	N_o	3	7.4	5.1	4	3	3	3	3	3	3	3
		N_i	49.8	22.5	24.4	27.6	24.5	29.6	35.8	41	42.2	44	48.7
300	I	N_o	3	11	6	4	3	3	3	3	3	3	3
		N_i	1329	2086	1729	1348	1106	1171	1207	1241	1257	1259	1286
	P	N_o	3	7.8	5.2	4	3	3	3	3	3	3	3
		N_i	74.2	35	35.7	37.3	32.9	40.2	48.4	56.1	62.3	65.4	67.5

Table 1: Averaged total numbers of outer and inner iterations.

To further illustrate the oversolving problem, we give the convergence history of Algorithms I and II for one of the test matrices with $n = 100$ in Figure 1. The figure depicts the logarithm of the error versus the number of inner iterations for solving the Jacobian systems (6) and (13) by the preconditioned QMR method. We have labeled the error at the outer iterations with special symbols. We can see that for Algorithm I, the oversolving problem is very significant (see the horizontal lines between iteration numbers 5 to 15, and 20 to 28), whereas there are virtually no oversolving for Algorithm II with $\beta = 1.5$.

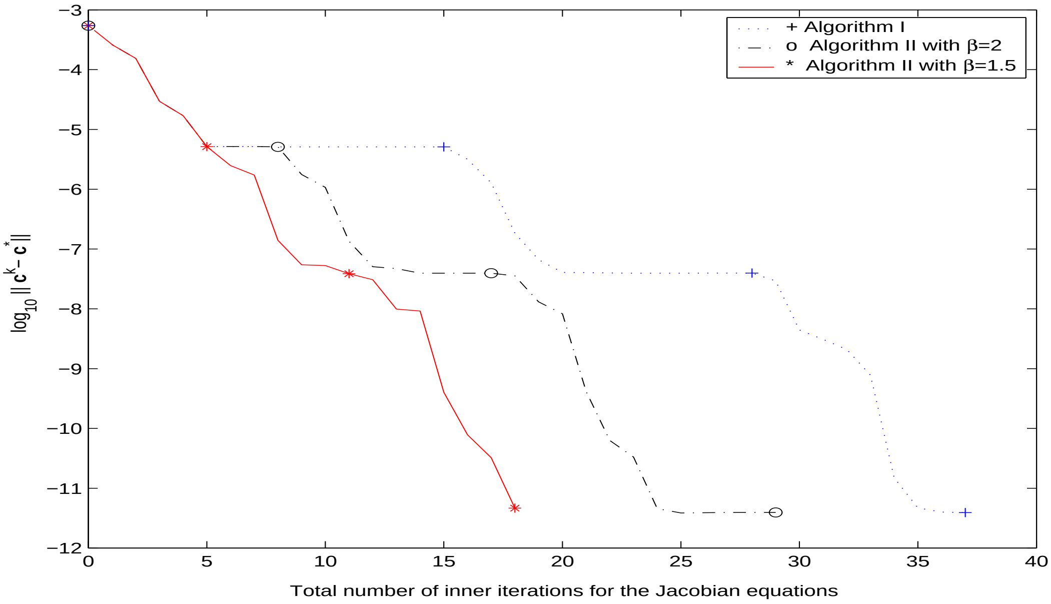

Figure 1: Convergence history of one of the test matrices.

In Table 1, (6) is solved with stopping tolerance $\eta = 10^{-13}$. One may expect that by increasing this stopping tolerance η, i.e. by solving (6) more inexactly, one can obtain an inexact method that may be better than our Algorithm II. To illustrate that it is not the case, we tried solving (6) with different η for ten matrices with $n = 100$ and 200, and compare their results with our Algorithm II with $\beta = 1.5$. We also repeated the experiments with four different iterative methods: the BiCG [36] and the CGS [34] methods together with their MILU-preconditioned versions. From Table 2, we see that our method is better than just solving (6) with increasing η. In fact, if η is big, the outer iteration does not converge within 20 iterations; and if η is small, the number of inner iterations will be bigger than that of our method. We also see that when n is larger, η should be smaller in order that the outer iteration converges.

Also from Table 2, we see that CGS performs better than the other two iterative solvers if preconditioning is used, but is worse if not. Since in general, we do not have any information regarding the structure of the Jacobian matrix in (6) and (13), choosing a good iterative solver for these systems will not be an easy problem, not to mention the choice of an effective preconditioner for them. However, the results in Table 2 show that the oversolving problem is independent of the solvers we choose. Our method is always better than Algorithm I if the same iterative solver is used. Clearly, a greater gain can be made if a better preconditioner is available. But since the Jacobian matrices (see (7) and (11)) are in general nonsymmetric and dense, how to choose a good preconditioner needs a further study, see for instance a survey paper [4].

		$n = 100$					$n = 200$			
		Alg. II $\beta = 1.5$	Alg. I Stopping tolerance η for (6)				Alg. II $\beta = 1.5$	Alg. I Stopping tolerance η for (6)		
			10^{-13}	10^{-12}	10^{-11}	10^{-10}		10^{-13}	10^{-12}	10^{-11}
N_o		3.2	3.2	3.2	3.2	> 20	3	3	3	> 20
QMR	N_i	323	397	356	344	*	719	818	738	*
BiCG	N_i	322	371	359	347	*	715	783	745	*
CGS	N_i	372	446	425	392	*	825	943	874	*
PQMR	N_i	17.9	37.7	32.7	28.2	*	29.6	49.8	41.8	*
PBiCG	N_i	18.3	37.7	33.1	28.5	*	30.5	49.5	42	*
PCGS	N_i	10.6	21.3	19	15.1	*	18.2	28.4	24.4	*

Table 2: Averaged total numbers of inner iterations.

As mentioned in §§2–3, solving the linear systems (10) and (15) iteratively will require only a few iterations since the coefficient matrices of these systems converge to the identity matrix as $\mathbf{c}^k$ converges to $\mathbf{c}^*$. We report in Table 3 the numbers of iterations required for convergence for these systems, averaged over the ten test problems with $n = 100$ and 200. From the table, we see that the number of inner iterations required is small and decreases as the outer iteration progresses. Thus it is reasonable to solve these systems by iterative solvers without any preconditioning.

Example 2. Consider the Sturm-Liouville problem:

$$-u'' + q(x)u = \lambda u, \quad u(0) = u(\pi) = 0. \tag{54}$$

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 (54) is reduced to the

	$n = 100$			$n = 200$		
Outer iteration	1st	2nd	3rd	1st	2nd	3rd
Alg. I	9.7	5.4	2.6	8.6	4.8	2.0
Alg. II with $\beta = 2.0$	9.8	5.3	2.6	8.6	4.8	2.0
Alg. II with $\beta = 1.5$	9.9	5.3	2.6	8.5	4.7	2.0

Table 3: Averaged numbers of inner iterations required by Step (d) of Algorithms I and II.

matrix eigenvalue problem with tridiagonal structure:

$$\left(A_0 + h^2 X\right) \mathbf{u} = h^2 \lambda \mathbf{u}, \tag{55}$$

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 (55) possesses a prescribed spectrum. Let $A_j = h^2 \mathbf{e}_j \mathbf{e}_j^T$, for $j = 1, \cdots, n$, where $\mathbf{e}_j$ is the jth unit n-vector. Thus we have the form (1) with $X = \mathrm{diag}(\mathbf{c})$.

In [2], inverse Sturm-Liouville problem of size $n = 50$ was considered. Here for demonstration purposes, we consider $n = 100$. Given the exact solution $\mathbf{c}^*$ with entries $[\mathbf{c}^*]_i = e^{3ih}$, $1 \le i \le n$, i.e. $q(x) = e^{3x}$, we use the eigenvalues $\{h^2 \lambda_i^*\}_{i=1}^n$ of $A(\mathbf{c}^*)$ as the prescribed spectrum. We perturb each entry of $\mathbf{c}^*$ by a random number uniformly distributed between -1 and 1, and then use the perturbed vector as the initial guess $\mathbf{c}^0$ for both Algorithms I and II. In practice, the available data will be eigenvalues of (54), not eigenvalues of (55). Before our methods can be implemented, it is necessary to use the given eigenvalues of (54) to obtain adequate estimates of the eigenvalues of (55). A simple method of doing this is described in [1, 2], which also contain some important references giving further details.

We also note that though the coefficient matrix in (55) is tridiagonal and sparse, the Jacobian matrices in (6) and (13) are still nonsymmetric and dense. Therefore, for large n, iterative methods with appropriate preconditioner will be better than direct methods for solving the corresponding Jacobian equations. Here, the linear systems (6), (10), (13), and (15) are solved by the MILU-preconditioned QMR method as in Example 1. Table 4 gives the total numbers of outer and inner iterations N_0 and N_i averaged over ten different initial guesses. From the table, we can see again that our method with $\beta \approx 1.5$ is better than Algorithm I.

	Alg. I	β in Alg. II									
		1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.0
N_o	3	7.8	5.3	4.4	4	3	3	3	3	3	3
N_i	71.6	65.1	60.3	60	62.8	48.6	56.6	60.4	65.7	68.1	73.2

Table 4: Averaged total numbers of outer and inner iterations for Example 2.

We end by remarking that in practice, the number of eigenvalues which can be measured is strictly limited. However, if we are to obtain the coefficient function $q(x)$ with sufficient accuracy, then we have to solve the inverse problem on more grid points and hence we have to supplement the observed spectrum with eigenvalues that are assigned in a systematical way. One way to do it is to use the ideas from finite element model updating [11, 24]. For example, if we are only given m measured eigenvalues $\{\lambda_i^*\}_{i=1}^m$, then we can compute $q(x)$ on m uniform grid-points by using our method above. To obtain $q(x)$ on n uniform grid-points, where $n > m$, one can interpolate the obtained $q(x)$ on n grid-points. But the spectrum of the resulting system, denoted by $\{\lambda_i(q)\}_{i=1}^n$, may not contain $\{\lambda_i^*\}_{i=1}^m$. Following Equation (17) in [24], we can replace those $\lambda_i(q)$ that are closest to λ_i^* by λ_i^*, and keep the remaining $(n-m)$ eigenvalues. Then we will have n prescribed eigenvalues. The $q(x)$ thus obtained by our method will be defined on n grid-points and its corresponding system will have the measured eigenvalues $\{\lambda_i^*\}_{i=1}^m$ as part of its spectrum.

Acknowledgment: We would like to thank the referees and Prof. S.F. Xu for their insightful and valuable comments.

References

[1] A. L. Andrew, *Some Recent Developments in Inverse Eigenvalue Problems*, in Computational Techniques and Applications, CTAC93, D. Stewart, H. Gardner, and D. Singleton, eds., World Scientific, Singapore, 1994, pp. 94–102.

[2] A. L. Andrew, *Numerical Solution of Inverse Sturm–Liouville Problems*, ANZIAM J., 45 (E) (2004), pp. C326–C337.

[3] E. L. Allgower and K. Georg, *Continuation and Path Following*, Acta Numer., (1993), pp. 1–64.

[4] M. Benzi, *Preconditioning Techniques for Large Linear Systems: A Survey*, J. Comput. Phys., 182 (2002), pp. 418-477.

[5] C. I. Byrnes, *Pole Placement by Output Feedback*, in Three Decades of Mathematics Systems Theory, Lecture Notes in Control and Inform. Sci. 135, Springer-Verlag, New York, 1989, pp. 31–78.

[6] R. H. Chan, H. L. Chung, and S. F. Xu, *The Inexact Newton-Like Method for Inverse Eigenvalue Problem*, BIT, 43 (2003), pp. 7–20.

[7] M. T. Chu, *Inverse Eigenvalue Problems*, SIAM Rev., 40 (1998), pp. 1–39.

[8] M. T. Chu, *On a Newton Method for the Inverse Toeplitz Eigenvalue Problem*, preprint available at http://www4.ncsu.edu/˜mtchu/ Research/Papers/itep.ps.

[9] M. T. Chu, *Numerical Methods for Inverse Singular Value Problems*, SIAM J. Numer. Anal., 29 (1992), pp. 885–903.

[10] M. T. Chu and G. H. Golub, *Structured Inverse Eigenvalue Problems*, Acta Numer., 11 (2002), pp. 1–71.

[11] B. N. Datta, *Finite-Element Model Updating, Eigenstructure Assignment and Eigenvalue Embedding Techniques for Vibrating Systems*, Mech. Syst. Signal Proc., 16 (2002), pp. 83–96.

[12] T. Dupont, R. P. Kendall, and H. H. Rachford JR., *An Approximate Factorization Procedure for Solving Self-adjoint Elliptic Difference Equations*, SIAM J. Numer. Anal., 5 (1968), pp. 559–573.

[13] S. C. Eisenstat and H. F. Walker, *Choosing the Forcing Terms in an Inexact Newton Method*, SIAM J. Sci. Comput., 17 (1996), pp. 16–32.

[14] S. Elhay and Y. M. Ram, *An Affine Inverse Eigenvalue Problem*, Inverse Problems, 18 (2002), pp. 455–466.

[15] D. R. Fokkema, G. L. G. Sleijpen, and H. A. van der Vorst, *Accelerated Inexact Newton Schemes for Large Systems of Nonlinear Equations*, SIAM J. Sci. Comput., 19 (1998), pp. 657–674.

[16] R. W. Freund and N. M. Nachtigal, *QMR: A Quasi-Minimal Residual Method for Non-Hermitian Linear Systems*, Numer. Math., 60 (1991), pp. 315–339.

[17] S. Friedland, J. Nocedal, and M. L. Overton, *The Formulation and Analysis of Numerical Methods for Inverse Eigenvalue Problems*, SIAM J. Numer. Anal., 24 (1987), pp. 634–667.

[18] G. M. L. Gladwell, *Inverse Problems in Vibration*, Appl. Mech. Rev., 39 (1986), pp. 1013–1018.

[19] G. M. L. Gladwell, *Inverse Problems in Vibration, II*, Appl. Mech. Rev., 49 (1996), pp. 25–34.

[20] I. Gustafsson, *A Class of First Order Factorizations*, BIT, 18 (1978), pp. 142–156.

[21] O. H. Hald, *On Discrete and Numerical Inverse Sturm-Liouville Problems*, Ph.D. thesis, New York University, New York, 1972.

[22] K. T. Joseph, *Inverse Eigenvalue Problem in Structural Design*, AIAA Ed. Ser., 30 (1992), pp. 2890–2896.

[23] N. Li, *A Matrix Inverse Eigenvalue Problem and Its Application*, Linear Algebra Appl., 266 (1997), pp. 143–152.

[24] C. Mares, M. I. Friswell, and J. E. Mottershead, *Model Updating Using Robust Estimation*, Mech. Syst. Signal Proc., 16 (2002), pp. 169–183.

[25] C. M. McCarthy, *Recovery of a Density From the Eigenvalues of a Nonhomogeneous Membrane*, Proceedings of the Third International Conference on Inverse Problems in Engineering: Theory and Practice, Port Ludlow, Washington, June 13–18, 1999.

[26] B. Morini, *Convergence Behaviour of Inexact Newton Methods*, Math. Comput., 68 (1999), pp. 1605–1613.

[27] M. Müller, *An Inverse Eigenvalue Problem: Computing B-Stable Runge-Kutta Methods Having Real Poles*, BIT, 32 (1992), pp. 676–688.

[28] M. Neher, *Ein Einschließungsverfahren für das Inverse Dirichletproblem*, Doctoral thesis, University of Karlsruhe, Karlsruhe, Germany, 1993.

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

[30] R. L. Parker and K. A. Whaler, *Numerical Methods for Establishing Solutions to the Inverse Problem of Electromagnetic Induction*, J. Geophys. Res., 86 (1981), pp. 9574–9584.

[31] J. Peinado and A. M. Vidal, *A New Parallel Approach to the Toeplitz Inverse Eigenproblem Using Newton-like Methods*, Lecture Notes in Computer Science, 1981/2001, Springer-Verlag, 2003, pp. 355–368.

[32] M. S. Ravi, J. Rosenthal, and X. A. Wang, *On Decentralized Dynamic Pole Placement and Feedback Stabilization*, IEEE Trans. Automat. Control, 40 (1995), pp. 1603–1614.

[33] V. Scholtyssek, *Solving Inverse Eigenvalue Problems by a Projected Newton Method*, Numer. Funct. Anal. Optim., 17 (1996), pp. 925–944.

[34] P. Sonneveld, *CGS: A Fast Lanczos-Type Solver for Nonsymmetric Linear Systems*, SIAM J. Sci. Stat. Comput., 10 (1989), pp. 36–52.

[35] W. F. Trench, *Numerical Solution of the Inverse Eigenvalue Problem for Real Symmmetric Toeplitz Matrices*, SIAM J. Sci. Comput., 18 (1997), pp. 1722–1736.

[36] H. A. van der Vorst, *BiCGSTAB: A Fast and Smoothly Converging Variant of the Bi-CG for the Solution of Nonsymmetric Linear Systems*, SIAM J. Sci. and Stat. Comp., 13 (1992), pp. 631–644.

[37] S. F. Xu, *An Introduction to Inverse Algebraic Eigenvalue Problems*, Peking University Press and Vieweg Publishing, 1998.