



A Contour-Integral Based Method with Schur–Rayleigh–Ritz Procedure for Generalized Eigenvalue Problems

Guojian Yin¹

Received: 22 December 2018 / Revised: 10 July 2019 / Accepted: 15 July 2019 / Published online: 22 July 2019
© Springer Science+Business Media, LLC, part of Springer Nature 2019

Abstract

Recently, a class of eigensolvers based on contour integrals has been developed for computing the eigenvalues inside a given region in the complex plane. The CIRR method (a Rayleigh–Ritz type method with contour integrals) is a classic example among this kind of methods. It first constructs a subspace to contain the eigenspace of interest via a set of contour integrals, and then uses the standard Rayleigh–Ritz procedure to extract desired eigenpairs. However, it was shown that the CIRR method may fail to find the desired eigenpairs when the considered eigenproblem is non-Hermitian. This fact motivates us to develop a non-Hermitian scheme for the CIRR method. To this end, we formulate a Schur–Rayleigh–Ritz procedure to extract the desired eigenpairs. The theoretical analysis shows that our new extraction scheme can make the CIRR method also applicable for the non-Hermitian problems. Some implementation issues arising in practical applications are also studied. Numerical experiments are reported to illustrate the numerical performance of our new method.

Keywords Generalized eigenvalue problems · Contour integral · QZ method · Generalized Schur decomposition

Mathematics Subject Classification 15A18 · 58C40 · 65F15

1 Introduction

Let A and B be large $n \times n$ matrices. Assume that we have a generalized eigenvalue problem

$$A\mathbf{x} = \lambda B\mathbf{x}, \quad (1)$$

and want to compute the eigenvalues λ_i , along with their eigenvectors \mathbf{x}_i , of (1) inside a given region in the complex plane. This problem arises in various areas of scientific and engineering applications, for example in the model reduction of a linear dynamical system, one needs to know the response over a range of frequencies, see [4, 12, 21]. Computing a number of interior eigenvalues of a large problem remains one of the most difficult problems in computational

✉ Guojian Yin
guojianyin@gmail.com

¹ School of Mathematics, Sun Yat-sen University, Guangzhou, People's Republic of China

linear algebra today [10]. In practice, the methods of choice are based on subspace projection techniques, the key to the success of which is to construct an approximate invariant subspace containing the eigenspace of interest. Krylov subspace methods in conjunction with spectral transformation techniques, for example the shift-and-invert technique, are often used [22,27].

Recently, eigensolvers based on contour integration were developed to compute the eigenvalues inside a prescribed domain in the complex plane. The best-known methods of this kind are the Sakurai–Sugiura method [24] and the FEAST algorithm [20]. A major computational advantage of these contour-integral based methods is that they can be easily implemented in modern distributed parallel computers [3,16,18]. The FEAST algorithm works under the conditions that matrices A and B are Hermitian and B is positive definite, i.e., (1) is a Hermitian problem [4]. The rigorous convergence analysis was studied in [28]. In [33], the authors adapted the FEAST algorithm to non-Hermitian problems. Recently, Güttel et al. in [14] improved the convergence robustness of the FEAST algorithm based on the Zolotarev quadrature rules. In the Sakurai–Sugiura method, the original eigenproblem (1) is reduced to a small one with Hankel matrices, if the number of sought-after eigenvalues is small. However, since Hankel matrices are ill-conditioned [5], the Sakurai–Sugiura method suffers from numerical instability in practical applications [3,25]. By noticing this fact, in [25], Sakurai et al. later used the Rayleigh–Ritz procedure to replace the Hankel matrix approach to get a more stable algorithm called CIRR. In [6,29,30], the contour-integral based techniques were developed to solve the nonlinear eigenproblems.

The CIRR method is accurate and powerful and attracts great interest recently [3,15,17,26]. It first constructs a subspace containing the eigenspace of interest through a set of specially defined contour integrals. Then the orthogonal projection technique is used to extract desired eigenpairs from the constructed subspace. Originally, as with the FEAST algorithm, the CIRR method was formulated under the assumptions that matrices A and B are Hermitian and B is positive definite. Moreover, the CIRR method requires that the sought-after eigenvalues are distinct. In [15], the authors presented a block version for CIRR to make it also applicable for the degenerate system, that is, the considered eigenproblem has repeated eigenvalues. However, it was shown in [33] that the CIRR method (including its block version) as well as the FEAST algorithm may fail to find the desired eigenpairs for some non-Hermitian problems, due to using the orthogonal projection technique to extract sought-after eigenvalues. An example was also given in [33] to illustrate this fact. The authors in [33] presented a non-Hermitian scheme for the FEAST algorithm. In this work, we want to formulate a non-Hermitian scheme for the CIRR method. We resort to an oblique projection technique. The most important task is to seek an appropriate left subspace. To this end, we borrow ideas of the JDQZ method [11], and finally formulate a Schur–Rayleigh–Ritz procedure [27, Chapter 6] for extracting the desired eigenvalues, which is different from the extraction approach used in the non-Hermitian FEAST algorithm [33]. Theoretical analysis shows that our new method can deal with non-Hermitian problems. The only requirement is that the matrix pencil $zB - A$, $z \in \mathbb{C}$, is regular, which means our method can deal with the most common generalized eigenvalue problems [4]. The key difference between our new method and the CIRR method is the extraction procedure. Our method only slightly increases the computational work comparing to that required in the original CIRR method. Numerical experiments show that our method is more stable than the CIRR method when applied to the non-Hermitian problems that the CIRR method can solve.

The way to construct a subspace to contain the desired eigenspace in our method is the same with the one developed in CIRR [15,25]. As a result, our method also faces some implementation issues that arise in the CIRR method. For example, we have to know the number of desired eigenvalues in advance. This is because we need this knowledge (i) to

select an appropriate size for the starting vectors such that the constructed subspace contains the desired eigenspace, and (ii) to guarantee that all desired eigenvalues are found when the method stops. However, we generally do not have this knowledge in practical situations. For our method to work, we combine the ideas developed in [26,33] for choosing suitable starting vectors, and use the stopping criteria given in [33] to guarantee all sought-after eigenvalues are found. Similar to other contour-integral based eigensolvers, the contour integrals involved in our method have to be computed approximately by some numerical integration. In [25], the authors used the trapezoidal rule to compute the contour integrals involved in the CIRR method. Polizzi chose the Gauß–Legendre quadrature for the FEAST algorithm [20]. In our experiment, we also use Gauß–Legendre quadrature to compute the related contour integrals.

The rest of the paper is organized as follows. In Sect. 2, we briefly review the CIRR method [25]. To adapt the CIRR method to non-Hermitian cases, in Sect. 3 we formulate a Schur–Rayleigh–Ritz procedure for extracting the desired eigenpairs. After that we discuss some implementation issues and present the complete algorithm. Numerical experiments are reported in Sect. 4 to illustrate the numerical performance of our new method.

Throughout the paper, we use the following notation and terminology. The subspace spanned by the columns of matrix X is denoted by $\text{span}\{X\}$. The rank of matrix A is denoted by $\text{rank}(A)$. For any matrix S , we denote the submatrix that lies in the first i rows and the first j columns of S by $S_{(1:i, 1:j)}$, the submatrix consisting of the first j columns of S by $S_{(:, 1:j)}$, and the submatrix consisting of the first i rows of S by $S_{(1:i, :)}$. We denote by S^* the conjugate transpose of the matrix S . The imaginary unit is denoted by ι . The algorithms are presented in MATLAB style.

2 The CIRR Method

In [24], Sakurai and Sugiura used a moment-based technique to formulate a contour-integral based method, i.e., the Sakurai–Sugiura method, for finding the eigenvalues of (1) inside a given region. In order to improve the numerical stability of the Sakurai–Sugiura method, a variant of it used the Rayleigh–Ritz procedure to extract the desired eigenpairs. This leads to the so-called CIRR method [15,25]. Originally the CIRR method was derived in [25] under the assumptions that (i) matrices A and B are Hermitian with B being positive definite, and (ii) the eigenvalues inside the given region are distinct. In [15], the authors proposed a block version for the CIRR method such that it is able to deal with degenerate problems. In this section we briefly review the block CIRR method. Below, unless otherwise explicitly stated, the CIRR method refers to its block version.

The matrix pencil $zB - A$ is regular if $\det(zB - A)$ is not identically zero for all $z \in \mathbb{C}$ [2,9]. The Weierstrass canonical form of a regular matrix pencil $zB - A$ is defined as follows.

Theorem 1 ([19,33]) *Let $zB - A$ be a regular matrix pencil of order n . Then there exist nonsingular matrices S and $T \in \mathbb{C}^{n \times n}$ such that*

$$TAS = \begin{bmatrix} J_d & 0 \\ 0 & I_{n-d} \end{bmatrix} \quad \text{and} \quad TBS = \begin{bmatrix} I_d & 0 \\ 0 & N_{n-d} \end{bmatrix}, \quad (2)$$

where J_d is a $d \times d$ matrix in Jordan canonical form with its diagonal entries corresponding to the eigenvalues of $zB - A$, N_{n-d} is an $(n - d) \times (n - d)$ nilpotent matrix also in Jordan canonical form, and I_d denotes the identity matrix of order d .

Let J_d in (2) be of the form

$$J_d = \begin{bmatrix} J_{d_1}(\lambda_1) & & & \\ & J_{d_2}(\lambda_2) & & \\ & & \ddots & \\ & & & J_{d_m}(\lambda_m) \end{bmatrix}, \tag{3}$$

where $\sum_{i=1}^m d_i = d$, $1 \leq d_i \leq d$ for $i = 1, \dots, m$ and $J_{d_i}(\lambda_i)$ are $d_i \times d_i$ matrices of the form

$$J_{d_i}(\lambda_i) = \begin{bmatrix} \lambda_i & 1 & & & \\ & \lambda_i & 1 & & \\ & & \ddots & \ddots & \\ & & & \ddots & 1 \\ & & & & \lambda_i \end{bmatrix}, \quad i = 1, 2, \dots, m$$

with λ_i being the eigenvalues. Here the λ_i are not necessarily distinct and can be repeated according to their multiplicities.

Let us partition S into block form

$$S = [S_1, S_2, \dots, S_m, S_{m+1}], \tag{4}$$

where $S_i \in \mathbb{C}^{n \times d_i}$, $1 \leq i \leq m$, and $S_{m+1} \in \mathbb{C}^{n \times (n-d)}$. Then the first column in each S_i is an eigenvector associated with eigenvalue λ_i for $i = 1, \dots, m$ [4,15,16,33].

Let Γ be a given positively oriented simple closed curve in the complex plane. Below we show how to use the CIRR method to compute the eigenvalues of (1) inside Γ , along with their associated eigenvectors. Without loss of generality, let the set of eigenvalues of (1) enclosed by Γ be $\{\lambda_1, \dots, \lambda_l\}$, and $s := d_1 + d_2 + \dots + d_l$ be the number of eigenvalues inside Γ with multiplicity taken into account.

Define the contour integrals

$$F_k := \frac{1}{2\pi i} \oint_{\Gamma} z^k (zB - A)^{-1} B dz, \quad k = 0, 1, \dots \tag{5}$$

With the help of the residue theorem in complex analysis [1], it was shown in [16] that

$$F_k = S_{(:,1:s)} (J_{(1:s,1:s)})^k (S^{-1})_{(1:s,:)}, \quad k = 0, 1, \dots \tag{6}$$

Let h and g be two positive integers satisfying $hg \geq s$, and Y be an $n \times h$ random matrix, whose entries are independent and identically distributed (i.i.d.). Define

$$U := [U_0, U_1, \dots, U_{g-1}], \quad \text{where } U_k := F_k Y, \quad k = 0, \dots, g-1. \tag{7}$$

We have the following result for the CIRR method. It is a slightly modified version of Theorem 1 in [17], where eigenproblem (1) is assumed to be diagonalizable.

Theorem 2 *Let the eigenvalues inside Γ be $\lambda_1, \dots, \lambda_l$, then the number of eigenvalues of (1) inside Γ is s , counting multiplicity. If $\text{rank}(U) = s$, then we have*

$$\text{span}\{U\} = \text{span}\{S_{(:,1:s)}\}. \tag{8}$$

Proof By (6) and (7), we know that

$$U = S_{(:,1:s)} E, \tag{9}$$

where

$$E = [(S^{-1})_{(1:s,:)}Y, J_{(1:s,1:s)}(S^{-1})_{(1:s,:)}Y, \dots, (J_{(1:s,1:s)})^{g-1}(S^{-1})_{(1:s,:)}Y]. \tag{10}$$

Since the rank of U is s , we have that $\text{rank}(E) = s$, following from which (8) holds. \square

The assumption $\text{rank}(U) = s$ is also required during the formulation of the CIRR method [17]. It should be pointed out that $\text{rank}(E) = s$ may not be satisfied for $s \leq hg$ [17]. If $\text{rank}(E) < s$, then we have $\text{rank}(U) < s$. It was shown in [33, Lemma 1] the rank of $(S^{-1})_{(1:s,:)}Y$ is equal to s with probability 1 if Y is an $n \times h$ random matrix with i.i.d. entries and $h \geq s$. In this case $\text{rank}(U) = s$ holds almost surely. Note that $(S^{-1})_{(1:s,:)}Y$ is the first block column of E , thus it enables us to choose a value for h such that (8) holds. An idea was given in [26] to determine a suitable parameter h for starting the CIRR method.

According to Theorem 2, we know that $\text{span}\{U\}$ contains the eigenspace corresponding to the desired eigenvalues, if $\text{rank}(U) = s$. The CIRR method uses the well-known orthogonal projection technique to extract the eigenpairs inside Γ from $\text{span}\{U\}$, i.e., imposing the Ritz–Galerkin condition:

$$A\mathbf{x} - \lambda B\mathbf{x} \perp \text{span}\{U\}, \tag{11}$$

where $\lambda \in \mathbb{C}$ and $\mathbf{x} \in \text{span}\{U\}$.

The main task of the CIRR method is to evaluate U_k [cf. (7)]. In practice, the contour integrals U_k have to be computed approximately by numerical integration:

$$U_k \approx \tilde{U}_k = \frac{1}{2\pi i} \sum_{j=1}^q \omega_j z_j^k (z_j B - A)^{-1} B Y, \quad k = 0, 1, \dots, g - 1, \tag{12}$$

where z_j are the integration points and ω_j are the corresponding weights. From (12), we see that q generalized shifted linear systems

$$(z_j B - A)X_j = B Y, \quad j = 1, 2, \dots, q, \tag{13}$$

are needed to be solved. Solving these linear systems is the dominant work of the CIRR method. Noticing that the integration points z_j and the columns of right-hand sides in (13) are independent, the CIRR method can be easily implemented on a modern distributed parallel computer.

The complete CIRR method is summarized as follows [15].

Algorithm 1: The CIRR method

Input: The number of integration points q , the starting matrix $Y \in \mathbb{C}^{n \times h}$, and an integer g satisfying $hg \geq s$.

Output: Approximate eigenpairs $(\hat{\lambda}_i, \hat{\mathbf{x}}_i)$, $\hat{\lambda}_i$ inside Γ .

1. Compute $\tilde{U}_k, k = 0, 1, \dots, g - 1$, approximately by (12).
2. Compute the QR decomposition of $\tilde{U} = [\tilde{U}_0, \dots, \tilde{U}_{g-1}] : \tilde{U} = \hat{U} \hat{R}$.
3. Set $\hat{A} = \hat{U}^* A \hat{U}$ and $\hat{B} = \hat{U}^* B \hat{U}$.
4. Solve the generalized eigenproblem of size hg : $\hat{A} \mathbf{y} = \hat{\lambda} \hat{B} \mathbf{y}$, to obtain the eigenpairs $\{(\hat{\lambda}_i, \mathbf{y}_i)\}_{i=1}^{hg}$.
5. Compute $\hat{\mathbf{x}}_i = \hat{U} \mathbf{y}_i$, and select s approximate eigenpairs $(\hat{\lambda}_i, \hat{\mathbf{x}}_i)$.

The larger the value of h is, the faster the convergence rate will be. Especially, when we take $g = 1$ and $h \geq s$, the CIRR method becomes the FEAST algorithm [20]. It is shown that

to achieve the same accuracy the FEAST algorithm needs less iteration number comparing with the CIRR method [33], but the FEAST algorithm requires more computational cost than CIRR in each iteration because more linear systems need to be solved in the FEAST algorithm.

3 A Contour-Integral Based Eigensolver with a Schur–Rayleigh–Ritz Procedure

The contour-integral based methods are recent efforts for computing the interior eigenvalues of a generalized eigenproblem. The CIRR method is a typical example among the methods of this kind. According to the brief description in the previous section, the basic idea of the CIRR method can be summarized as follows: (i) constructing a subspace that contains the desired eigenspace by means of a set of contour integrals [cf. (7)], and (ii) using the orthogonal projection technique, with respect to the subspace $\text{span}\{U\}$, to extract the desired eigenpairs. However, the authors in [33] observed that the CIRR method may fail to find the desired eigenvalues when the eigenproblem under consideration is non-Hermitian, owing to using the orthogonal projection technique to extract the sought-after eigenvalues. They provided an example (Example 3.1 in [33]) for illustrating this fact.

In this section, we formulate a new extraction approach for adapting the CIRR method to the non-Hermitian cases. The idea stems from the motivation of using the oblique projection technique, instead of the orthogonal projection technique, to extract the desired eigenvalues for the CIRR method. To this end, we borrow the idea behind formulating the well-known JDQZ method [11], and finally develop a Schur–Rayleigh–Ritz extraction procedure. Some implementation issues that the new non-Hermitian CIRR scheme may encounter in practical applications will be discussed. After that, we will give the complete algorithm.

3.1 The Derivation of Our New Contour-Integral Based Eigensolver

The CIRR method uses the orthogonal projection technique to extract the sought-after eigenpairs from $\text{span}\{U\}$, which may result in failure when it comes to the non-Hermitian problems. Here we consider using the oblique projection technique [4,22], another class of projection method, to extract the desired eigenpairs.

Since $\text{span}\{U\}$ contains the eigenspace of interest [cf. (8)], as with the CIRR method, it is natural to choose $\text{span}\{U\}$ as the right subspace (or *search subspace*). The oblique projection technique extracts the desired eigenpairs from $\text{span}\{U\}$ by imposing the Petrov–Galerkin condition, which requires the residual of approximate eigenpair is orthogonal to some left subspace (or *test subspace*), say, $\text{span}\{W\}$:

$$A\mathbf{x} - \lambda B\mathbf{x} \perp \text{span}\{W\}, \quad (14)$$

where λ is located inside Γ , $\mathbf{x} \in \text{span}\{U\}$, and W is an $n \times s$ orthogonal matrix. Let V be an $n \times s$ matrix whose columns form an orthonormal basis of $\text{span}\{U\}$. The orthogonality condition (14) leads to the projected eigenproblem

$$(W^*AV)\mathbf{y} = \lambda(W^*BV)\mathbf{y}, \quad (15)$$

where $\mathbf{y} \in \mathbb{C}^s$ satisfies $\mathbf{x} = V\mathbf{y}$. As a result, computing the eigenvalues of (1) inside Γ now is transformed to solving the projected eigenproblem (15).

If the left subspace is taken to be the right subspace $\text{span}\{U\}$, the same as the right subspace, then the method described above is the orthogonal projection technique. The authors in [33] observed that the projected matrices V^*AV and V^*BV become zero matrices when using the orthogonal projection technique, consequently, it fails to find the desired eigenpairs. Now our task is to seek an appropriate left subspace $\text{span}\{W\}$, which is different from the right subspace. We start with a partial generalized Schur form for the matrix pair (A, B) .

Definition 1 ([11]) A partial generalized Schur form of dimension s for a matrix pair (A, B) is the decomposition

$$A Q_s = Z_s H_s, \quad B Q_s = Z_s G_s, \tag{16}$$

where Q_s and Z_s are orthogonal $n \times s$ matrices, and H_s and G_s are upper triangular $s \times s$ matrices. A column $(Q_s)_{(:,i)}$ is referred to as a right generalized Schur vector, and we refer to a pair $((H_s)_{(i,i)}/(G_s)_{(i,i)}, (Q_s)_{(:,i)})$ as a generalized Schur pair.

The formulation (16) is equivalent to

$$(Z_s)^* A Q_s = H_s, \quad (Z_s)^* B Q_s = G_s, \tag{17}$$

from which we know that $(H_s)_{(i,i)}/(G_s)_{(i,i)}$ are the eigenvalues of (H_s, G_s) . Let y_i be the eigenvectors of pair (H_s, G_s) associated with eigenvalues $(H_s)_{(i,i)}/(G_s)_{(i,i)}$, then we have that $((H_s)_{(i,i)}/(G_s)_{(i,i)}, Q_s y_i)$ are the eigenpairs of (A, B) [11, 19].

Applying the QZ factorization to the projected pair (W^*AV, W^*BV) to yield the generalized Schur form [13]

$$(P_L)^*(W^*AV)P_R = H_A \quad \text{and} \quad (P_L)^*(W^*BV)P_R = H_B, \tag{18}$$

where P_R and P_L are orthogonal $s \times s$ matrices, H_A and H_B are upper triangular $s \times s$ matrices. Then the eigenvalues of the pair (W^*AV, W^*BV) are $\{(H_A)_{(i,i)}/(H_B)_{(i,i)}\}_i^s$ [13, 19].

In view of (16), it was shown in [4, 11] that

$$\text{span}\{Z_s\} = \text{span}\{A Q_s\} = \text{span}\{B Q_s\}. \tag{19}$$

Comparing (17) with (18), it can readily be seen that we have constructed a partial generalized Schur form in (18) for the matrix pair (A, B) : $V P_R$ constructs a right generalized Schur vectors Q_s and $W P_L$ constructs a left generalized Schur vectors Z_s . It follows from (19) that

$$\text{span}\{W P_L\} = \text{span}\{A V P_R\} = \text{span}\{B V P_R\}. \tag{20}$$

Note that P_L and P_R are nonsingular, thus we have

$$\text{span}\{W\} = \text{span}\{A V\} = \text{span}\{B V\}. \tag{21}$$

According to (21), it makes sense to take the left subspace $\text{span}\{W\}$ to $\text{span}\{A V - \sigma B V\}$, where σ is a scalar and different from the desired eigenvalues. To justify this choice, we need the following result.

Theorem 3 ([15]) Let $L, D \in \mathbb{C}^{n \times t}$, $t \geq s$, be arbitrary matrices, and $R = F_0 D$. A projected matrix pencil $z \hat{B} - \hat{A}$ is defined by $\hat{B} = L^* B R$ and $\hat{A} = L^* A R$. If ranks of both $L^*(T^{-1})_{(:,1:s)}$ and $(S^{-1})_{(1:s,:)} D$ are s , the non-singular part of the projected matrix pencil is equivalent to $z I_s - (J_d)_{(1:s,1:s)}$.

Theorem 3 says that the desired eigenvalues $\{\lambda_i\}_{i=1}^l$ can be obtained via computing the eigenvalues of the projected eigenproblem $z\hat{B} - \hat{A}$, if the ranks of both $(S^{-1})_{(1:s,:)}D$ and $L^*(T^{-1})_{(:,1:s)}$ are s . Note that $F_0U = U$. Due to these, we want to show the following result.

Theorem 4 *If the rank of U is s , and the columns of V form an orthonormal basis of $\text{span}\{U\}$, then the ranks of $(S^{-1})_{(1:s,:)}V$ and $(AV - \sigma BV)^*(T^{-1})_{(:,1:s)}$ are s .*

Proof We first show that the rank of $(S^{-1})_{(1:s,:)}V$ is s . Since $\text{rank}(U) = s$, by Theorem 2 we know $\text{span}\{U\} = \text{span}\{S_{(:,1:s)}\}$. Note that the columns of V form an orthonormal basis of $\text{span}\{U\}$. Therefore, there exists an $s \times s$ nonsingular matrix Δ_1 such that

$$V = S_{(:,1:s)}\Delta_1. \tag{22}$$

Now we have

$$(S^{-1})_{(1:s,:)}V = (S^{-1})_{(1:s,:)}S_{(:,1:s)}\Delta_1 = \Delta_1, \tag{23}$$

from which we see that the rank of $(S^{-1})_{(1:s,:)}V$ is s .

Next we show that the rank of $(AV - \sigma BV)^*(T^{-1})_{(:,1:s)}$ is s . For convenience, we turn to show that the rank of $((T^{-1})_{(:,1:s)})^*(AV - \sigma BV)$, i.e., the conjugate transpose of $(AV - \sigma BV)^*(T^{-1})_{(:,1:s)}$, is s .

Since $\text{span}\{AV\} = \text{span}\{BV\}$ [cf. (21)], there exists an $s \times s$ nonsingular matrix Δ_2 such that

$$AV = BV\Delta_2. \tag{24}$$

According to (2) and (22), one can easily verify

$$((T^{-1})_{(:,1:s)})^*(AV - \sigma BV) = (BS_{(:,1:s)})^*BS_{(:,1:s)}\Delta_1(\Delta_2 - \sigma I_s). \tag{25}$$

Now we are in a position to show $\Delta_2 - \sigma I_s$ is nonsingular. Let $\Delta_2 = X_{\Delta_2}\Lambda_{\Delta_2}(X_{\Delta_2})^{-1}$ be the Jordan decomposition of Δ_2 . By (22) and (24) we have

$$A(S_{(:,1:s)}\Delta_1X_{\Delta_2}) = B(S_{(:,1:s)}\Delta_1X_{\Delta_2})\Lambda_{\Delta_2}, \tag{26}$$

from which we can conclude that the eigenvalues of Δ_2 are exactly the eigenvalues of (1) inside Γ . On the other hand, if $\Delta_2 - \sigma I_s$ is singular, then σ is an eigenvalue of Δ_2 , which is in contradiction with the assumption that σ is not an eigenvalue inside Γ .

In view of (2), we know $BS_{(:,1:s)}$ is full-rank, which means $(BS_{(:,1:s)})^*BS_{(:,1:s)}$ is nonsingular. By (25), now we can conclude that the rank of $(AV - \sigma BV)^*(T^{-1})_{(:,1:s)}$ is s . □

Based on Theorems 3 and 4, we know that the eigenvalues of pair $((AV - \sigma BV)^*AV, (AV - \sigma BV)^*BV)$ are the eigenvalues of (1) inside Γ , which justifies our choice of the left subspace, that is, taking $\text{span}\{W\}$ to $\text{span}\{AV - \sigma BV\}$. Let $((H_A)_{(i,i)}/(H_B)_{(i,i)}, \tilde{y}_i)$ be the eigenpairs of (H_A, H_B) , note that VP_R are the right generalized Schur vectors associated with H_A and H_B , we have that $((H_A)_{(i,i)}/(H_B)_{(i,i)}, VP_R\tilde{y}_i)$ are exactly the eigenpairs of (1) inside Γ . Now, we formulate a new contour-integral based eigensolver, which uses a Schur–Rayleigh–Ritz procedure [27, Chapter 6] to extract the desired eigenpairs. We summarize the new method as follows.

Algorithm 2: A contour-integral based method with Schur-Rayleigh-Ritz procedure.

Input: The number of quadrature nodes g , the starting matrix $Y \in \mathbb{C}^{n \times h}$, and an integer g satisfying $hg \geq s$, a scalar $\sigma \in \mathbb{C}$.

Output: Approximate eigenpairs $(\tilde{\lambda}_i, \tilde{\mathbf{x}}_i), i = 1, \dots, s, \tilde{\lambda}_i$ inside Γ .

1. Compute $\tilde{U}_k, k = 0, 1, \dots, g - 1$, by (12), and set $\tilde{U} = [\tilde{U}_0, \tilde{U}_1, \dots, \tilde{U}_{g-1}]$.
2. Compute QR decompositions: $\tilde{U} = VR_1$ and $AV - \sigma BV = WR_2$.
3. Compute $\tilde{A} = W^*AV$ and $\tilde{B} = W^*BV$.
4. Compute $[H_A, H_B, P_L, P_R, V_L, V_R] = \text{qz}(\tilde{A}, \tilde{B})$.
5. Compute $\tilde{\lambda}_i = (H_A)_{(i,i)}/(H_B)_{(i,i)}$ and $\tilde{\mathbf{x}}_i = VP_R(V_R)_{(:,i)}$.
6. Select the approximate eigenpairs $(\tilde{\lambda}_i, \tilde{\mathbf{x}}_i)$.

Our goal of our work is adapting the CIRR method to non-Hermitian problems. Algorithm 2 can be viewed as a non-Hermitian scheme of the CIRR method. It chooses the left subspace to $\text{span}\{AV - \sigma BV\}$, rather than $\text{span}\{U\}$ used in the CIRR method [15]. There are three typical types of choice for the parameter σ .

1. In view of the relationship (21), we know $\text{span}\{AV\} = \text{span}\{BV\} = \text{span}\{AV - \sigma BV\}$. However, in practical computations, we only can get the approximation, \tilde{U} , of U (in Step 1). Therefore, the first choice for σ is -1 , with the hope of the subspace $\text{span}\{W\}$ capturing the information of $\text{span}\{A\tilde{U}\}$ and $\text{span}\{B\tilde{U}\}$ simultaneously with equal weight. Here, it is required that -1 is not an eigenvalue inside the target region.
2. According to Theorem 4, for our method to work, the scalar σ must be different from the desired eigenvalues, which motivates the second type of choice for σ , that is, a value outside the target region.
3. The third type of choice is taking σ to a scalar that is located inside Γ , then our extraction procedure becomes the so-called harmonic Schur–Rayleigh–Ritz procedure [32]. The authors in [32] formulated a harmonic Schur–Rayleigh–Ritz procedure for computing the eigenvalues near a user-specified target value. When it comes to our method, it makes sense to choose the parameter σ to the center of the target region if the center is not an eigenvalue inside the target region.

In Step 4, we use the MATLAB built-in function `qz` to compute the QZ factorization of (\tilde{A}, \tilde{B}) , which produces two unitary matrices P_L, P_R of the left and right generalized Schur vectors, along with the upper triangular matrices H_A, H_B , and matrices V_L, V_R , whose columns are the left and right generalized eigenvectors, along with eigenvalues $(H_A)_{(i,i)}/(H_B)_{(i,i)}$, respectively. Specifically, the approximate eigenvectors are given by $VP_R(V_R)_{(:,i)}$. We have shown that VP_R and WP_L are the right and left partial generalized Schur vectors of pair (A, B) corresponding to upper triangular matrices H_A and H_B , the ratios of whose diagonal entries are the eigenvalues inside Γ . Therefore, Algorithm 2 can also be used to compute the partial generalized Schur factorization.

3.2 The Implementation Issues

If we apply Algorithm 2 to compute the eigenvalues inside Γ , we will encounter some issues in a practical implementation, just like other contour-integral based eigensolvers [20,24,25]. In this section, we discuss the implementation issues of Algorithm 2, and then present the complete algorithm.

The first issue we have to treat is about selecting a suitable parameter h for the starting matrix Y , with a prescribed parameter g . Since U (cf. 7) is expected to span a subspace that contains the eigenspace of interest, we have to choose a parameter h , the number of columns of Y , such that $hg \geq s$, the number of eigenvalues inside Γ . A strategy was proposed in [26] for finding a suitable parameter h for the CIR method. It starts with finding an estimation to s . Giving a positive integer h_0 , by “ $Y_{h_0} \sim N(0, 1)$ ”, we mean Y_{h_0} is an $n \times h_0$ matrix with i.i.d. entries drawn from the standard normal distribution $N(0, 1)$. By (6) and (7), one can easily verify that the mean

$$\mathbb{E}[\text{trace}((Y_{h_0})^* F_0 Y_{h_0})] = h_0 \cdot \text{trace}(F_0) = h_0 \cdot \text{trace}(S_{(:,1:s)}(S^{-1})_{(1:s,:)} = h_0 \cdot s. \tag{27}$$

Therefore,

$$s_0 := \frac{1}{h_0} \cdot \text{trace}((Y_{h_0})^* F_0 Y_{h_0}) \tag{28}$$

is an unbiased estimator of s , and the variance is $(\text{trace}(F_0^* F_0) + s)/h_0$ [33]. With this information on hand, the strategy in [26] works as follows: (i) set $h = \lceil \frac{s_0 \kappa}{g} \rceil$, where $\kappa > 1$, (ii) select the starting matrix $Y \in \mathbb{C}^{n \times h}$ and compute \tilde{U}_k by (12), (iii) if the minimum singular value σ_{\min} of $\tilde{U} = [\tilde{U}_0, \dots, \tilde{U}_{g-1}]$ is small enough, we find a suitable h ; otherwise, replace h with κh and repeat (ii) and (iii). We observe that the formula (28) can give a good estimation of s . However the computed s_0 may be much larger than s in some cases, such as when the matrices A and B are ill-conditioned. In these cases it is potentially expensive to compute the singular value decomposition of \tilde{U} . Due to this fact, we borrow some ideas proposed in [33]. The working mechanism of our method is as follows: use the rank-revealing QR factorization [7,13] to monitor the numerical rank of \tilde{U} , if \tilde{U} is numerically rank-deficient, then it means that the subspace spanned by \tilde{U} already contains the desired eigenspace sufficiently, as a result, we find a suitable parameter h .

Another issue we have to address is designing the stopping criteria. The stopping criteria here include two aspects: (i) all computed approximate eigenpairs attain the prescribed accuracy, and (ii) all eigenpairs inside the given region are found.

As for the first aspect of the stopping criteria, since we can only compute U approximately by some numerical integration [cf. (12)], the approximate eigenpairs computed by Algorithm 2 may be unable to attain the prescribed accuracy in practice. A natural solution is to refine \tilde{U} (step 2 in Algorithm 2) iteratively. A refinement scheme was suggested in [17]. Let $\tilde{U}_0^{(0)} = Y$ and l be a positive integer, the refinement scheme iteratively computes $U_k^{(l)} = F_k \tilde{U}_0^{(l-1)}$ by a q -point numerical integration scheme:

$$U_k^{(l)} \approx \tilde{U}_k^{(l)} = \frac{1}{2\pi i} \sum_{j=1}^q \omega_j z_j^k (z_j B - A)^{-1} B \tilde{U}_0^{(l-1)}, \quad k = 0, 1, \dots, g - 1, \tag{29}$$

and then constructs

$$\tilde{U}^{(l)} = \left[\tilde{U}_0^{(l)}, \tilde{U}_1^{(l)}, \dots, \tilde{U}_{g-1}^{(l)} \right]. \tag{30}$$

The refined $\tilde{U}^{(l)}$ is used to form the projected eigenproblem (15). The accuracy of approximate eigenpairs will be improved as the iterations proceed, see [26] for more details.

If all s approximate eigenpairs attain the prescribed accuracy after a certain iteration, we could stop the iteration process. However, in general we do not know the number of eigenvalues inside the target region in advance. This fact leads to the second aspect of the stopping criteria: how to guarantee that all desired eigenpairs are found when the iteration process stops. To fulfill this task, we take advantage of the idea proposed in [33]. The rationale

of the idea is that, as the iteration process proceeds, the accuracy of desired eigenpairs will be improved while the spurious ones do not, as a result, there will exist a gap of accuracy between the desired eigenpairs and the spurious ones [33]. Based on this observation, a test tolerance η , say 1.0×10^{-3} , is introduced to discriminate between the desired eigenpairs and the spurious ones. Specifically, for an approximate eigenpair $(\tilde{\lambda}_i, \tilde{\mathbf{x}}_i)$, define the corresponding residual norm as

$$r_i = \frac{\|A\tilde{\mathbf{x}}_i - \tilde{\lambda}_i B\tilde{\mathbf{x}}_i\|}{\|A\tilde{\mathbf{x}}_i\| + \|B\tilde{\mathbf{x}}_i\|}. \tag{31}$$

If $r_i < \eta$, then we view $(\tilde{\lambda}_i, \tilde{\mathbf{x}}_i)$ as an approximation to a sought-after eigenpair and refer to it as a filtered eigenpair by η . If the numbers of filtered eigenpairs are the same in two consecutive iterations, then we set them to be the number of eigenvalues inside Γ , see [33] for more details.

From (29) we can see that, in each iteration, the dominant work is to compute q generalized shifted linear systems of the form

$$(z_i B - A)X_i^{(l-1)} = BU_0^{(l-1)}, \quad i = 1, 2, \dots, q. \tag{32}$$

One can choose solvers to compute (32) based on the properties of matrices involved. In practical situations, when the matrices A and B are large-scale and sparse, the Krylov subspace based methods, such as GMRES [23] and BiCGSTAB [31], are natural choices. Since (32) are generalized shifted systems with multiple right-hand sides, the direct methods based on the sparse Gaussian LU factorization are also highly recommended. Especially, once the LU factors are obtained, they can be reused in the subsequent iterations. Since the quadrature nodes $z_j, j = 1, \dots, q$, are independent, and the columns of the right-hand sides are also independent, like other contour-integral based eigensolvers, our new method has a good potential to be parallelized.

Integrating the above strategies with Algorithm 2, we arrive at the resulting non-Hermitian CIRR method developed in this work. Algorithm 2 is the main body of our new method. Note that the key difference between Algorithm 2 and the CIRR method is the extraction approach. The CIRR method uses the widely used orthogonal projection technique, which leads to the potential failure when using the CIRR method to deal with non-Hermitian problems. Algorithm 2 uses the Schur–Rayleigh–Ritz procedure to extract the desired eigenvalues. Mathematical analysis shows that our method can deal with the non-Hermitian cases. Due to this, we refer to the resulting method as CISRR for ease of representation.

Algorithm 3: The CISRR method

Input: Matrices A and B , the size of sample vectors h_0 , a positive integer g , the number of integration points q , increasing factor κ , test tolerance η , convergence tolerance ϵ , a scalar $\sigma \in \mathbb{C}$, and maximum iteration number `max_iter`.

Output: Approximate eigenpairs $(\tilde{\lambda}_i, \tilde{\mathbf{x}}_i), i = 1, \dots, s, \tilde{\lambda}_i$ inside Γ .

1. Let $Y_{h_0} \sim N(0, 1)$, compute $\tilde{U}_k, k = 0, \dots, g - 1$, by (12).
2. Compute $s_0 = \lceil \frac{1}{h_0} \text{trace}((Y_{h_0})^* \tilde{U}_0) \rceil$, and set $h = \max\{\lceil \frac{s_0 \kappa}{g} \rceil, h_0\}$.
3. If $h > h_0$
4. Pick $\hat{Y}_{h-h_0} \sim N(0, 1)$ and compute \hat{U}_k by by (12). Augment \hat{U}_k to $\tilde{U}_k: \tilde{U}_k = \begin{bmatrix} \tilde{U}_k & \hat{U}_k \end{bmatrix} \in \mathbb{C}^{n \times h}$ and construct $\tilde{U} = \begin{bmatrix} \tilde{U}_0 & \tilde{U}_1 & \dots & \tilde{U}_{g-1} \end{bmatrix}$.
5. Else
6. Set $h = h_0$ and construct $\tilde{U} = \begin{bmatrix} \tilde{U}_0 & \tilde{U}_1 & \dots & \tilde{U}_{g-1} \end{bmatrix}$.

7. End
8. Compute the rank-revealing QR factorization: $\tilde{U} = VR_1T$. Set $s_1 = \text{rank}(R_1)$.
If $s_1 < hg$, stop; otherwise, set $h_0 = h$, $h = \kappa h$ and go to step 3.
9. Compute QR decomposition: $AV - \sigma BV = WR_2$.
10. Set $e(0) = 0$ and $c(0) = n$.
11. For $k = 1, 2, \dots, \text{max_iter}$
12. Compute $\tilde{A} = W^*AV$ and $\tilde{B} = W^*BV$. Set $s_1 = \text{rank}(\tilde{A})$.
13. Compute $[S_A, S_B, U_L, U_R, V_L, V_R] = \text{qz}(\tilde{A}, \tilde{B})$.
14. Compute $\tilde{\lambda}_i = (S_A)_{(i,i)} / (S_B)_{(i,i)}$ and $\tilde{\mathbf{x}}_i = VU_R(V_R)_{(:,i)}$, $i = 1, \dots, s_1$.
15. Set $r = []$, $\Lambda^{(k)} = []$, $X^{(k)} = []$, and $c(k) = 0$.
16. For $i = 1 : s_1$
17. If $\tilde{\lambda}_i$ inside Γ , compute $r_i = \|A\tilde{\mathbf{x}}_i - \tilde{\lambda}_i B\tilde{\mathbf{x}}_i\| / (\|A\tilde{\mathbf{x}}_i\| + \|B\tilde{\mathbf{x}}_i\|)$.
18. If $r_i < \eta$, set $c(k) = c(k) + 1$, $r = [r, r_i]$, $X^{(k)} = [X^{(k)}, \tilde{\mathbf{x}}_i]$,
and $\Lambda^{(k)} = [\Lambda^{(k)}, \tilde{\lambda}_i]$.
19. End
20. Set $e(k) = \max(r)$.
21. If $c(k) = c(k-1)$ and $e(k) < \varepsilon$, set $\tilde{\lambda}_i = (\Lambda^{(k)})_i$, $\tilde{\mathbf{x}}_i = (X^{(k)})_{(:,i)}$. Exit.
22. Set $Y = \tilde{U}_0$, and compute \tilde{U}_k by (12). Construct $\tilde{U} = [\tilde{U}_0, \tilde{U}_1, \dots, \tilde{U}_{g-1}]$.
23. Compute QR decompositions: $\tilde{U} = VR_1$ and $AV - \sigma BV = WR_2$.
24. End

Here we give some remarks on Algorithm 3.

1. Steps 1 to 8 are devoted to determining a suitable parameter h for the starting matrix Y . Meanwhile, an orthonormal matrix V , the subspace spanned by which contains the desired eigenspace approximately, is generated.
2. The for-loop, steps 16 to 19, is used to detect the spurious eigenvalues. Only the approximate eigenpairs whose residual norms are less than the test tolerance η are retained.
3. Step 21 refers to the stopping criteria, which contain two aspects: (i) the number of filtered eigenpairs is the same with the one in the previous iteration, and (ii) the residual norms of all filtered eigenpairs are less than the prescribed tolerance ε .

4 Numerical Experiments

In this section, we use some numerical experiments to illustrate the performance of our CISRR method (Algorithm 3). The test problems are from the Matrix Market collection.¹ The first six test problems are generalized eigenproblems and the last two test problems are standard ones. They are real-world problems from scientific and engineering applications. The descriptions of the related matrices are presented in Table 1, where `nnz` denotes the number of non-zero elements and `cond` denotes the condition numbers which are computed by Matlab function `condst`. All computations are carried out in MATLAB version R2014b on a MacBook with an Intel Core i5 2.5 GHz processor and 8 GB RAM.

The target region for each test problem is chosen to be a disk enclosed by circle Γ . In Table 2, γ and ρ represent the center and the radius of the target circle Γ , respectively, and s is the number of eigenvalues inside Γ . In our method, it is required to solve the

¹ <http://math.nist.gov/MatrixMarket/>.

Table 1 Test problems from Matrix Market that are used in our experiments

No.	Problem	Type	Size	Matrix	nnz	cond
1	BFW398	Generalized	398	A	3678	7.58×10^3
				B	2910	3.64×10^1
2	BFW782	Generalized	782	A	7514	4.63×10^3
				B	5982	3.05×10^1
3	DWG961	Generalized	961	A	3405	Inf
				B	10591	3.21×10^7
4	MHD1280	Generalized	1280	A	47906	9.97×10^{24}
				B	22778	5.99×10^{12}
5	MHD3200	Generalized	3200	A	68026	2.02×10^{44}
				B	18316	2.02×10^{13}
6	MHD4800	Generalized	4800	A	102252	2.54×10^{57}
				B	27520	1.03×10^{14}
7	DW8192	Standard	8192	A	41746	1.50×10^7
8	AF23560	Standard	23560	A	460598	3.50×10^5

Table 2 The target region and the number of eigenvalues inside

No.	γ	ρ	$\min_{\lambda, z_j} \left\{ \frac{ z_j - \lambda }{\rho} \right\}$	s
1	-5.0×10^5	1.0×10^5	2.17×10^{-2}	58
2	-6.0×10^5	2.0×10^5	5.48×10^{-2}	141
3	-5.0×10^5	3.0×10^5	2.08×10^{-2}	143
4	-1.0×10^1	8.0	2.21×10^{-2}	72
5	-5.0×10^1	3.0×10^1	2.32×10^{-2}	137
6	-5.0	2.0	2.17×10^{-2}	130
7	-5.0	0.5	9.41×10^{-3}	154
8	-1.0×10^2	1.5×10^1	3.44×10^{-2}	78

linear systems of the form $(z_j B - A)X_i = Y$ in each iteration. If an eigenvalue is close to the integration point z_i , then $z_i B - A$ will be ill-conditioned. In Table 2, we present $\min_{\lambda, z_j} \{|z_j - \lambda|/\rho\}$, which is used to measure the minimum distance between the integration points and the spectrum of the test problems. We use Gauß–Legendre quadrature [8] with $q = 16$ integration points on Γ to compute the contour integrals (29). To solve the generalized shifted linear systems of the form (32), we first use the MATLAB function `lu` to compute the LU decomposition of $A - z_j B$, $j = 1, 2, \dots, q$, and then perform the triangular substitutions to get the corresponding solutions. Define

$$\max_r = \max_{1 \leq i \leq s} r_i, \tag{33}$$

where r_i are the residual norms given by $\|A\tilde{x}_i - \tilde{\lambda}_i B\tilde{x}_i\| / (\|A\tilde{x}_i\| + \|B\tilde{x}_i\|)$ and $(\tilde{\lambda}_i, \tilde{x}_i)$ are the filtered eigenpairs. In our method (CISRR), we use test tolerance η to detect the spurious eigenvalues. In the experiments we take $\eta = 1.0 \times 10^{-3}$. In each iteration, only the solutions

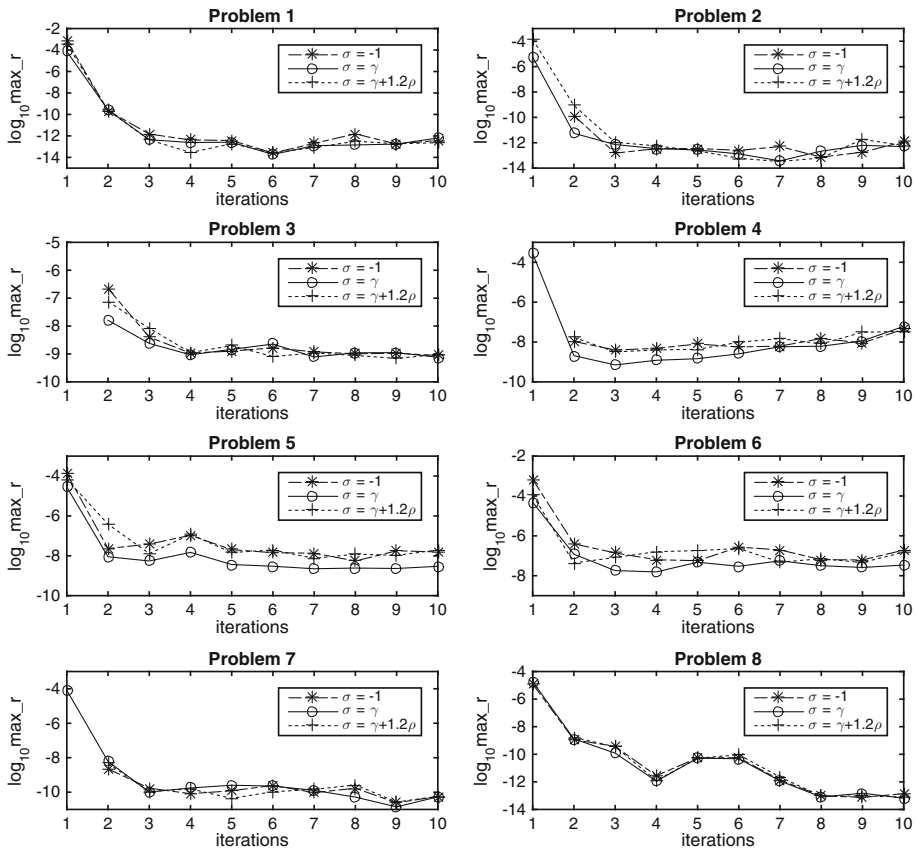


Fig. 1 The convergence behavior of Algorithm 2 corresponds to different σ s

whose residual norms are less than η are retained. The retained approximate eigenvalues are referred to as filtered eigenvalues.

Experiment 4.1 In the present work, we want to develop a non-Hermitian scheme for the CIR method [25]. To this end, we formulate a Schur–Rayleigh–Ritz procedure for extracting the desired eigenpairs, which leads to Algorithm 2, the main body of the resulting method (CISRR). The theoretical analysis shows that with the help of our new extraction approach, Algorithm 2 is applicable for non-Hermitian problems. There is an important parameter needed to select in Algorithm 2, that is, the scalar σ . We have introduced three types of choice for this parameter. This experiment is devoted to illustrating the numerical performance of Algorithm 2 when the parameter σ is taken to different values.

In the previous section, we introduced three typical types of choice for the parameter σ . In this experiment, we take σ to be -1 , γ , and $\gamma + 1.2\rho$, which correspond to the cases that (i) the left subspace $\text{span}\{W\}$ captures the information of $\text{span}\{AV\}$ and $\text{span}\{BV\}$ equally, (ii) σ is located inside Γ , and (iii) σ is outside the target region, respectively.

In the experiment, we set $g = 5$ and $h = \lceil 1.5s/5 \rceil$ such that $hg > s$, here we assume that the number of desired eigenvalues, s , is already known. In Fig. 1, we display the accuracy achieved by Algorithm 2 at the iterations that s filtered eigenvalues are found. It can be seen that Algorithm 2 has similar convergence behavior for the three different values of σ . We can

Table 3 Comparison of `eig` and CISRR

No.	<code>eig</code>	CISRR
1	2.57×10^{-13}	6.50×10^{-14}
2	5.59×10^{-12}	2.24×10^{-13}
3	6.81×10^{-10}	3.41×10^{-10}
4	1.15×10^{-8}	2.46×10^{-10}
5	4.94×10^{-7}	7.59×10^{-10}
6	2.59×10^{-7}	4.33×10^{-9}

observe that Algorithm 2 achieves the highest accuracy when the scalar σ is chosen as the center of the target region for the test problems 6 and 7, whose matrices are ill-conditioned. We have mentioned that in this case our extraction approach is actually the so-called harmonic Schur–Rayleigh–Ritz procedure [32]. In the following experiments, the parameter σ is taken to be the center of the target region for each test problem.

Experiment 4.2 The goal of this experiment is to compare our CISRR method (Algorithm 3) with the MATLAB built-in function `eig`, as well as two typical contour-integral based eigensolvers in terms of accuracy and computational time. We use `max_r` [see (33)] to measure the accuracy achieved by the test methods. In this experiment, the size of sampling vectors h_0 and the parameter g are taken to 30 and 5, respectively.

First, we compare our CISRR method with the MATLAB built-in function `eig` for the first six test problems. Applying `eig` to compute the eigenvalues inside the target region, we have to first compute all eigenvalues in dense format and then select the target eigenvalues according to the coordinates of computed eigenvalues. However, the matrices listed in Table 1 are sparse. Therefore, for the sake of fairness, we compare the two test methods only in terms of accuracy, measured by `max_r`, and will not show the amount of CPU time taken by these two methods.

In Table 3, we display the `max_rs` computed by `eig` in the second column. The smallest `max_rs` computed by the CISRR method within the first five iterations are presented in the last column. It is shown that our CISRR method is more accurate than its counterpart. Remarkably, for the test problems 4, 5 and 6, whose related matrices are ill-conditioned, the MATLAB built-in function `eig` cannot achieve high accuracy, we can see that our method outperforms its counterpart by two or three digits for these three problems.

We also compare our CISRR method with the FEAST algorithm and the CIRRR method. These three methods are contour-integral based eigensolvers. The FEAST algorithm originally was formulated for the Hermitian eigenproblems. The authors in [33] adapted it for non-Hermitian problems. Since our test problems are non-Hermitian, in the experiment we choose the non-Hermitian FEAST developed in [33] for comparison. As for the CIRRR method, in [26] the authors addressed some of its implementation problems, including how to select suitable starting vectors. In our work, we propose another way to choose an appropriate starting vectors. We find that our method is less costly, thus we use the method proposed in our work to determine the starting vectors for the CIRRR method in the tests. As a result, the main difference between our method and the CIRRR method used here is the extraction procedure.

Table 4 shows the accuracy achieved, as well as the CPU time required, by FEAST, CIRRR, and our CISRR method after four iterations. Firstly, we can see that the FEAST algorithm can obtain more accurate solutions comparing with the other two contour-integral based eigensolvers for the first three test problems and the last two standard eigenproblems; while

Table 4 Comparison of FEAST, CIRR, and CISRR

No.	FEAST		CIRR		CISRR	
	max_r	Time (s)	max_r	Time (s)	max_r	Time (s)
1	2.11×10^{-14}	1.14	2.52×10^{-13}	0.63	6.50×10^{-14}	0.67
2	2.29×10^{-14}	5.76	8.08×10^{-12}	3.54	4.49×10^{-13}	4.18
3	1.35×10^{-10}	8.35	5.50×10^{-10}	4.46	5.82×10^{-10}	5.43
4	2.29×10^{-9}	15.5	5.79×10^{-7}	8.91	2.46×10^{-10}	10.9
5	8.39×10^{-9}	27.7	6.93×10^{-7}	17.9	7.59×10^{-10}	19.4
6	1.08×10^{-8}	59.0	3.49×10^{-6}	20.6	4.33×10^{-9}	21.4
7	1.59×10^{-14}	105	8.70×10^{-12}	64.8	8.28×10^{-12}	67.2
8	6.71×10^{-14}	745	2.50×10^{-12}	372	2.18×10^{-12}	390

our method is most accurate in the other three problems, whose matrices are ill-conditioned. As for the CPU time, one can see that the FEAST algorithm is most costly of all three test methods. In each iteration, the dominant work of all three methods is solving $q = 16$ linear systems of the form (32). But the number of right-hand side vectors of the linear systems involved in the FEAST algorithm must be larger than the number, s , of eigenvalues inside Γ . But the numbers of right-hand side vectors in CIRR and CISRR are the parameter h , which is much smaller than the value of s . On the other hand, it is shown in Table 4 that the computational cost of our CISRR method is slightly larger than that of the CIRR method. As was mentioned, the difference between the version of CIRR used here and our CISRR method is just the way of extracting the desired eigenvalues. In both CIRR and CISRR, we need to compute the orthonormal basis V for the right subspace. In our method, the left subspace is different from the right subspace. Comparing to the CIRR method, the extra work required by our method is computing the orthonormal basis of the left subspace $\text{span}\{AV - \sigma BV\}$.

Experiment 4.3. Our work is motivated by the fact that the CIRR method [15] may fail to solve the desired eigenpairs when the problem under consideration is non-Hermitian. The key to the success of our CISRR method is formulating a (harmonic) Schur–Rayleigh–Ritz procedure, instead of the Rayleigh–Ritz procedure used in the CIRR method, to extract the eigenvalues of interest. This experiment is devoted to comparing our CISRR method with the CIRR method by their convergence behavior.

For all test problems, beginning with the second iteration, the number of filtered eigenvalues equals the number of eigenvalues inside Γ . Therefore, in Fig 2, we plot the max_rs [cf. (33)] from the second iteration to the 10th for each test problem. For comparison, we also plot the max_rs computed by the CIRR method in Fig 2. Here, the version of CIRR used is the same as the one described in the previous experiment, it can deal with all eight test problems we selected. We can see that the convergence behaviour of the two methods is quite similar for all test problems, except for the problems whose matrices are ill-conditioned. The max_rs decreases monotonically and dramatically in the first few iterations, and then maintain at almost the same level in the subsequent iterations. Remarkably, there exist obvious accuracy gaps between our method and the CIRR method for test problems 4, 5 and 6. The difference between the two test methods is only the way of extracting the desired eigenvalues, the projected matrix pairs in CIRR and our CISRR are (V^*AV, V^*BV) and $((AV)^*AV - \sigma(BV)^*AV, (AV)^*BV - \sigma(BV)^*BV)$, respectively. Note that both $(AV)^*AV$ and $(BV)^*BV$ are positive definite matrices, it can be expected that the con-

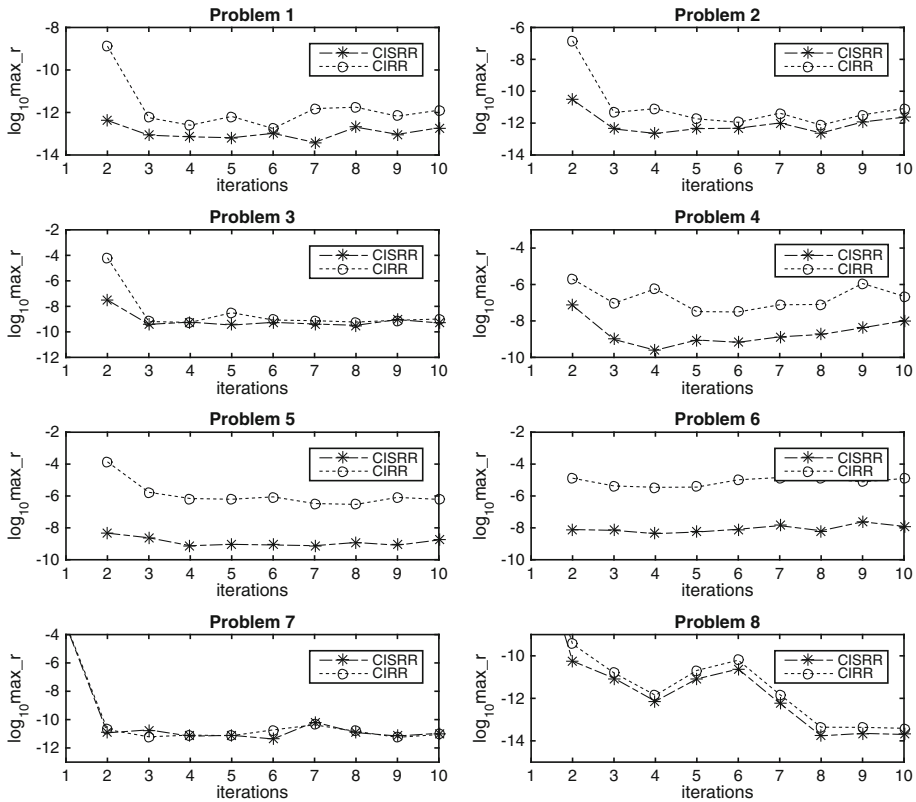


Fig. 2 The convergence behavior of CISRR and CIRR

dition numbers of the projected matrices in CISRR are smaller than those in CIRR. This may explain why our new extraction procedure generates more accurate solutions especially when the matrices involved are ill-conditioned, as is shown in the experiment.

5 Conclusions

In this paper, we presented a contour-integral method with Schur–Rayleigh–Ritz procedure for computing the eigenpairs inside a given region. The goal of our work is to make the CIRR method also applicable for non-Hermitian problems. Our method is based on the CIRR method. The main difference between the original CIRR method and our CISRR method is the way of extracting the desired eigenpairs. The theoretical analysis justified that our new method can deal with non-Hermitian eigenvalue problems. The CIRR method extracts the desired eigenpairs from the subspace spanned by the corresponding eigenbasis. In our method, the desired eigenpairs are extracted from the subspace spanned by the corresponding right generalized Schur vectors. Our extraction approach can circumvent the difficulties related to ill-conditioning or deficiency of the eigenbasis. The numerical experiments showed that our method was more reliable and accurate than the CIRR method. In the present work, some implementation issues arising in practical applications were also studied. The numerical

experiments showed that our method performs better if the parameter σ in our method is chosen as a value inside the target region, in which case, the extraction approach formulated in this work becomes the harmonic Schur–Rayleigh–Ritz procedure. It is easy to see that our method can be used to retrieve the partial generalized Schur vectors. The convergence analysis of the new method will be studied in our further work. The Matlab code is available on request.

Acknowledgements I would like to thank Professor Michiel E. Hochstenbach for his careful reading of the paper, and for comments which have helped to substantially improve the presentation. This work was supported by the National Natural Science Foundation of China (NSFC) under Grant 11701593.

References

1. Ahlfors, L.: Complex Analysis, 3rd edn. McGraw-Hill, Inc., New York (1979)
2. Anderson, A., Bai, Z., Bischof, C., Blackford, S., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A., Sorensen, D.: LAPACK Users' Guide, 3rd edn. SIAM, Philadelphia (1999)
3. Austin, A.P., Trefethen, L.N.: Computing eigenvalues of real symmetric matrices with rational filters in real arithmetic. *SIAM J. Sci. Comput.* **37**, A1365–A1387 (2015)
4. Bai, Z., Demmel, J., Dongarra, J., Ruhe, A., Van der Vorst, H.: Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide. SIAM, Philadelphia (2000)
5. Beckermann, B., Golub, G.H., Labahn, G.G.: On the numerical condition of a generalized Hankel eigenvalue problem. *Numer. Math.* **106**, 41–68 (2007)
6. Beyn, W.-J.: An integral method for solving nonlinear eigenvalue problems. *Linear Algebra Appl.* **436**, 3839–3863 (2012)
7. Chan, T.T.: Rank revealing QR factorizations. *Linear Algebra Appl.* **88–89**, 67–82 (1987)
8. Davis, P.J., Rabinowitz, P.: Methods of Numerical Integration, 2nd edn. Academic Press, Orlando (1984)
9. Demmel, J.: Applied Numerical Linear Algebra. SIAM, Philadelphia (1997)
10. Fang, H.-R., Saad, Y.: A filtered Lanczos procedure for extreme and interior eigenvalue problems. *SIAM J. Sci. Comput.* **34**, A2220–A2246 (2012)
11. Fokkema, D.R., Sleijpen, G.L.G., Van Der Vorst, H.A.: Jacobi–Davidson style QR and QZ algorithms for the reduction of matrix pencils. *SIAM J. Sci. Comput.* **20**, 94–125 (1998)
12. Gallivan, K., Grimme, E., Van Dooren, P.: A rational Lanczos algorithm for model reduction. *Numer. Algorithms.* **12**, 33–64 (1996)
13. Golub, G.H., Van Loan, C.F.: Matrix Computations, 3rd edn. Johns Hopkins University Press, Baltimore (1996)
14. Güttel, S., Polizzi, E., Tang, P., Viaud, G.: Zolotarev quadrature rules and load balancing for the FEAST eigensolver. *SIAM J. Matrix Anal. Appl.* **37**, A2100–A2122 (2015)
15. Ikegami, T., Sakurai, T.: Contour integral eigensolver for non-Hermitian systems: a Rayleigh–Ritz-type approach. *Taiwan. J. Math.* **14**, 825–837 (2010)
16. Ikegami, T., Sakurai, T., Nagashima, U.: A filter diagonalization for generalized eigenvalue problems based on the Sakurai–Sugiura projection method. *J. Comput. Appl. Math.* **233**, 1927–1936 (2010)
17. Imakura, A., Du, L., Sakurai, T.: Error bounds of Rayleigh–Ritz type contour integral-based eigensolver for solving generalized eigenvalue problems. *Numer. Algorithms* **68**, 103–120 (2015)
18. Krämer, L., Di Napoli, E., Galgon, M., Lang, B., Bientinesi, P.: Dissecting the FEAST algorithm for generalized eigenproblems. *J. Comput. Appl. Math.* **244**, 1–9 (2013)
19. Moler, C.B., Stewart, G.W.: An algorithm for generalized matrix eigenvalue problems. *SIAM J. Numer. Anal.* **10**, 241–256 (1973)
20. Polizzi, E.: Density-matrix-based algorithm for solving eigenvalue problems. *Phys. Rev. B* **79**, 115112 (2009)
21. Ruhe, A.: Rational Krylov: a practical algorithm for large sparse nonsymmetric matrix pencils. *SIAM J. Sci. Comput.* **19**, 1535–1551 (1998)
22. Saad, Y.: Numerical Methods for Large Eigenvalue Problems. SIAM, Philadelphia (2011)
23. Saad, Y., Schultz, M.H.: GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.* **7**, 856–869 (1986)
24. Sakurai, T., Sugiura, H.: A projection method for generalized eigenvalue problems using numerical integration. *J. Comput. Appl. Math.* **159**, 119–128 (2003)

25. Sakurai, T., Tadano, H.: CIRR: a Rayleigh–Ritz type method with contour integral for generalized eigenvalue problems. *Hokkaido Math. J.* **36**, 745–757 (2007)
26. Sakurai, T., Futamura, Y., Tadano, H.: Efficient parameter estimation and implementation of a contour integral-based eigensolver. *J. Algorithms Comput. Technol.* **7**, 249–269 (2013)
27. Stewart, G.W.: *Matrix Algorithms, Vol. II, Eigensystems*. SIAM, Philadelphia (2001)
28. Tang, P., Polizzi, E.: FEAST as a subspace iteration eigensolver accelerated by approximate spectral projection. *SIAM J. Matrix Anal. Appl.* **35**, 354–390 (2014)
29. Van Barel, M.: Designing rational filter functions for solving eigenvalue problems by contour integration. *Linear Algebra Appl.* **502**, 346–365 (2016)
30. Van Barel, M., Kravanja, P.: Nonlinear eigenvalue problems and contour integrals. *J. Comput. Appl. Math.* **292**, 526–540 (2015)
31. Van der Vorst, H.A.: Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.* **13**, 631–644 (1992)
32. Vecharynski, E., Yang, C., Xue, F.: Generalized preconditioned locally harmonic residual method for non-Hermitian eigenproblems. *SIAM J. Sci. Comput.* **38**, A500–A527 (2016)
33. Yin, G., Chan, R., Yueng, M.-C.: A FEAST algorithm with oblique projection for generalized eigenvalue problems. *Numer. Linear Algebra Appl.* **24**, e2092 (2017)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.