A RANDOMIZED FEAST ALGORITHM FOR GENERALIZED EIGENVALUE PROBLEMS

GUOJIAN YIN*

Abstract. The FEAST algorithm, due to Polizzi, is a typical contour-integral based eigensolver for computing the eigenvalues, along with their eigenvectors, inside a given region in the complex plane. It was formulated under the circumstance that the considered eigenproblem is Hermitian. The FEAST algorithm is stable and accurate, and has attracted much attention in recent years. However, it was observed that the FEAST algorithm may fail to find the target eigenpairs when applying it to the non-Hermitian problems. Efforts have been made to adapt the FEAST algorithm to non-Hermitian cases. In this work, we develop a new non-Hermitian scheme for the FEAST algorithm. The mathematical framework will be established, and the convergence analysis of our new method will be studied. Numerical experiments are reported to demonstrate the effectiveness of our method and to validate the convergence properties.

Key words. generalized eigenvalue problems, contour integral, spectral projection

AMS subject classifications. 15A18, 58C40, 65F15

1. Introduction. Large-scale non-Hermitian eigenvalue problems arise in various areas of science and engineering, such as dynamic analysis of structures [14], linear stability analysis of the Navier-Stokes equation in fluid dynamics [6], the electron energy and position problems in quantum chemistry [10], and resonant state calculation [22]. In most practical applications, it is not the whole spectrum but rather a significant part of it is of interest to the users [20]. For example, in the model reduction of a linear dynamical system, one only needs to know the response over a range of frequencies, see [3, 11].

Consider the generalized eigenvalue problem

$$A\mathbf{x} = \lambda B\mathbf{x}, \tag{1.1}$$

where $A, B \in \mathbb{C}^{n\times n}$. The scalars $\lambda \in \mathbb{C}$ and the associated vectors $\mathbf{x} \in \mathbb{C}^n, \mathbf{x} \neq 0$, are called the eigenvalues and their associated (right) eigenvectors, respectively [3, 8, 13]. When B is the identity matrix, then (1.1) becomes a standard eigenvalue problem. In this work, our objective is to compute the eigenvalues of (1.1) inside a given region in the complex plane, along with the corresponding eigenvectors.

Computing the partial spectrum of a large-scale problem is very difficult in practice. Maybe the most straightforward method is first using the well-known QZ method [19] to compute the whole spectrum and then selecting the target eigenvalues. This direct method costs about $\mathcal{O}(n^3)$ [13], consequently, it is prohibitively expensive when the size of considered problem is large. In the past decades, the most successful methods for solving the partial spectrum of a large eigenproblem are based on the projection techniques [3, 4, 29], of which perhaps the Krylov subspace approaches are the most wildly used [23, 24]. However, the existing projection methods mainly focus on computing the extreme eigenvalues [28] or the eigenvalues close to a given shift [14].

Recently, a class of eigensolvers based on contour integrals were proposed for computing the eigenvalues inside a given region in the complex plane [2, 5, 9, 21, 25, 26, 27]. Unlike the well-known Krylov subspace methods, these new methods use

*School of Mathematics, Sun Yat-sen University, Guangzhou, P. R. China (guojianyin@gmail.com).

specifically defined contour integrals to generate subspaces to contain the eigenspace corresponding to the target eigenvalues. Then the projection techniques are used to extract the target eigenpairs. Two typical examples of these contour-integral based eigensolvers are the Sakurai-Sugiura (SS) method [26] and the FEAST algorithm developed by Polizzi in [21]. By noticing that the SS method always suffers from numerical instability [2, 17], Sakurai *et al.* turned to use the Rayleigh-Ritz procedure to extract the target eigenpairs, and leaded to a more stable contour-integral based eigensolver, called CIRR [16, 27].

The derivation of both CIRR and FEAST is under the assumptions that A and B are Hermitian matrices and B is positive definite, i.e., (1.1) is a Hermitian problem. It was shown in [32] that CIRR and FEAST may fail to find the target eigenpairs when (1.1) is a non-Hermitian problem. Motivated by this observation, the authors in [32] developed a non-Hermitian FEAST algorithm to make the FEAST algorithm also applicable for the non-Hermitian problems. Instead of the orthogonal projection technique used in the FEAST algorithm, the non-Hermitian FEAST algorithm proposed in [32] uses to the oblique projection technique with appropriately chosen left subspace to extract desired eigenpairs.

In this work, we would like to formulate another non-Hermitian scheme for the FEAST algorithm. We find that the FEAST algorithm can deal with the non-Hermitian problems if the left subspace spanned by a random matrix. The theoretical analysis will be given to justify our findings. The convergence properties also will be studied to show the effectiveness of our method.

The paper is organised as follows. In Section 2, we briefly describe the FEAST algorithm [21]. In Section 3, we review the non-Hermitian variant of the FEAST algorithm proposed in [32]. We formulate our new non-Hermitian FEAST algorithm and give convergence analysis in Section 4. In Section 5, numerical experiments are reported to illustrate the numerical performance of our method.

Throughout the paper, we use the following notation and terminology. The subspace spanned by the columns of a matrix X is denoted by $\mathrm{span}\{X\}$. The rank and conjugate transpose of X are denoted by $\mathrm{rank}(X)$ and X^* respectively. The algorithms are presented in MATLAB style.

2. Introduction to FEAST. In this section, we provide a brief review of the FEAST algorithm [21]. The algorithm was formulated under the assumptions that A and B are Hermitian and B is positive definite, in which case the eigenvalues of (1.1) are real-valued [8]. The FEAST algorithm was developed for finding all eigenvalues of (1.1) within a specified interval, say $[\sigma_1, \sigma_2]$, and their associated eigenvectors. Without loss of generality, assume that the eigenvalues inside $[\sigma_1, \sigma_2]$ are $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_s$. Therefore, there are s eigenvalues inside $[\sigma_1, \sigma_2]$.

Essentially, the FEAST algorithm belongs to the family of subspace iteration with orthogonal projection [30]. Unlike the better known Krylov subspace methods, the FEAST algorithm constructs a subspace that envelops the desired eigenspace via the contour integral defined as

$$V := \frac{1}{2\pi\sqrt{-1}} \oint_{\Gamma} (zB - A)^{-1} dz Y, \tag{2.1}$$

where Γ is any contour that contains $\{\lambda_i\}_{i=1}^{s}$ inside, and Y is an $n \times s$ random matrix. To formulate the FEAST algorithm, we need the following theorem.

THEOREM 2.1 ([29]). *Let A and B be $n \times n$ Hermitian matrices and that B is*

positive definite. Then there exists an $n \times n$ matrix $X = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]$ for which

$$X^*BX = I_n \quad \textit{and} \quad X^*AX = \Lambda = \mathrm{diag}(\lambda_1, \lambda_2 \cdots, \lambda_n), \tag{2.2}$$

where I_n is the $n \times n$ identity matrix, $\{\lambda_i\}_{i=1}^n$ are the eigenvalues of the matrix pencil $zB - A$, and the columns $\{\mathbf{x}_i\}_{i=1}^n$ of X are their associated eigenvectors.

By (2.2) and the residue theorem in complex analysis [1], we have

$$V = \frac{1}{2\pi\sqrt{-1}} X \oint_\Gamma (zI_n - \Lambda)^{-1} dz Y (BX)^{-1} = X_{(:,1:s)}(X_{(:,1:s)})^* Y. \tag{2.3}$$

Then the columns of V form a basis for the eigenspace $\mathrm{span}\{X_{(:,1:s)}\}$, if $(X_{(:,1:s)})^*Y$ is full-rank. Forming the $s \times s$ matrices $\hat{A} = V^*AV$ and $\hat{B} = V^*BV$, solving the problem (1.1) now is reduced to computing the eigenpairs of the projected eigenvalue problem

$$\hat{A}\mathbf{y} = \lambda \hat{B}\mathbf{y}, \tag{2.4}$$

according to the Rayleigh-Ritz procedure [21, 29].

To generate the projected eigenproblem (2.4), the most important task is to compute the basis vectors V. In view of (2.1) and (2.3), we know that V has to be computed numerically by a quadrature scheme. Let Γ be the circle with center at $\gamma = (\sigma_1 + \sigma_2)/2$ and radius $\rho = (\sigma_2 - \sigma_1)/2$, applying the q-point Gauss-Legendre quadrature [7] to compute V numerically yields

$$V = \frac{1}{2\pi\sqrt{-1}} \oint_\Gamma (zB - A)^{-1} dz Y \approx \frac{1}{2} \sum_{j=1}^{q} \omega_j (z_j - \gamma)(z_j B - A)^{-1} Y, \tag{2.5}$$

where $z_j = \gamma + \rho e^{\sqrt{-1}\theta_j}$, $\theta_j = (1 + t_j)\pi$, and t_j is the jth Gaussian node with associated weight ω_j. From (2.5), one can see that the dominant computational work of the FEAST algorithm is solving the linear systems of the form

$$(z_j B - A)X_j = Y, \quad j = 1, \ldots, q. \tag{2.6}$$

The complete FEAST algorithm is given as follows.

ALGORITHM 1. *Input Hermitian matrices A and B with B being positive definite, a uniformly-distributed random matrix $Y \in \mathbb{R}^{n \times t}$, where $t \geq s$, the circle Γ enclosing the interval $[\sigma_1, \sigma_2]$, and a convergence tolerance ϵ. The function "*FEAST*" computes eigenpairs $(\hat{\lambda}_i, \hat{\mathbf{x}}_i)$ of (1.1) that satisfy*

$$\hat{\lambda}_i \in [\sigma_1, \sigma_2] \quad \text{and} \quad \sum_{i=1}^{s} \hat{\lambda}_i < \epsilon, \tag{2.7}$$

and they are output in the vector Λ_s and the matrix X_s.

Function $[\Lambda_s, X_s] = \mathrm{FEAST}(A, B, Y, \Gamma, \epsilon)$

1. Compute V approximately by (2.5).
2. Set $\hat{A} = V^*AV$ and $\hat{B} = V^*BV$.
3. Solve the generalized eigenproblem of size t: $\hat{A}\mathbf{y} = \hat{\lambda}\hat{B}\mathbf{y}$, to obtain the eigenpairs $\{(\hat{\lambda}_i, \mathbf{y}_i)\}_{i=1}^t$.
4. Compute $\hat{\mathbf{x}}_i = V\mathbf{y}_i, i = 1, 2, \ldots t$.
5. Check if $\{(\hat{\lambda}_i, \hat{\mathbf{x}}_i)\}_{i=1}^t$ satisfy the convergence criteria (2.7). If s eigenpairs satisfy (2.7), stop. Otherwise, set $X_t = [\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \ldots, \hat{\mathbf{x}}_t]$ and $Y = BX_t$, then go back to Step 1.

The FEAST algorithm is an accurate and reliable technique [18, 30]. It transforms the difficulty of solving the eigenproblem (1.1) to that of solving linear systems (2.6). Since the quadrature nodes z_j and the columns of the right-hand sides in (2.6) are independent, the FEAST algorithm can be easily implemented on parallel machines. Due to these appealing features, the FEAST algorithm attracts much attention recently.

3. A non-Hermitian FEAST algorithm. The FEAST algorithm was formulated when (1.1) is a Hermitian problem. However, when it comes to the non-Hermitian problem, it was found in [32] that the FEAST algorithm may fail to compute the desired eigenpairs; a simple example was given to illustrate this fact. Motivated by this observation, the authors in [32] developed a non-Hermitian FEAST algorithm so as to adapt FEAST to the non-Hermitian cases. The key to the success of their non-Hermitian FEAST algorithm is that the oblique projection technique, instead of the orthogonal projection technique used in FEAST, with appropriately chosen left subspace is used to extract the desired eigenpairs.

The only requirement for the non-Hermitian FEAST algorithm proposed in [32] is that the matrix pencil $zB-A$ is regular, which means the method is able to deal with the most common generalized eigenproblems [3]. Recall that a matrix pencil $zB-A$ is regular if $\det(zB-A)$ is not identically zero for all $z \in \mathbb{C}$. As with the Jordan canonical form for a matrix, there exists a canonical form for the regular matrix pencil $zB-A$.

THEOREM 3.1 (The Weierstrass canonical form [12, 32]). *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*

$$TAS = \begin{bmatrix} J_d & 0 \\ 0 & I_{n-d} \end{bmatrix} \quad \text{and} \quad TBS = \begin{bmatrix} I_d & 0 \\ 0 & N_{n-d} \end{bmatrix}, \tag{3.1}$$

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 be of the form

$$J_d = \begin{bmatrix} J_{d_1}(\lambda_1) & 0 & \cdots & 0 \\ 0 & J_{d_2}(\lambda_2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & J_{d_m}(\lambda_m) \end{bmatrix} \tag{3.2}$$

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

$$J_{d_i}(\lambda_i) = \begin{bmatrix} \lambda_i & 1 & 0 & \cdots & 0 \\ 0 & \lambda_i & 1 & & \vdots \\ & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & 1 \\ 0 & \cdots & & 0 & \lambda_i \end{bmatrix}, \qquad i = 1, 2, \ldots, m$$

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

Let N_{n-d} be of the form

$$N_{n-d} = \begin{bmatrix} N_{d_1'} & 0 & \cdots & 0 \\ 0 & N_{d_2'} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & N_{d_{m'}'} \end{bmatrix},$$

where $\sum_{i=1}^{m'} d_i' = n - d$ and $N_{d_i'}$ are $d_i' \times d_i'$ matrices of the form

$$N_{d_i'} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & & \vdots \\ & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & 1 \\ 0 & \cdots & & 0 & 0 \end{bmatrix}, \qquad i = 1, 2, \ldots, m'.$$

Partition S into block form $S = [S_1, S_2, \ldots, S_m, S_{m+1}]$, where each $S_i \in \mathbb{C}^{n \times d_i}$, $1 \le i \le m$, and S_i into $S_i = [\mathbf{s}_1^i, \mathbf{s}_2^i, \ldots, \mathbf{s}_{d_i}^i]$ with $\mathbf{s}_j^i \in \mathbb{C}^n$, $1 \leqslant j \leqslant d_i$. It was verified in [32] that for any eigenvalue λ_i, $1 \le i \le m$,

$$(\lambda_i B - A) S \begin{bmatrix} I_d & 0 \\ 0 & N_{n-d} \end{bmatrix} = BS \begin{bmatrix} \lambda_i I_d - J_d & 0 \\ 0 & \lambda_i N_{n-d} - I_{n-d} \end{bmatrix}. \tag{3.3}$$

By comparing the first d columns on both sides above, we get

$$(\lambda_i B - A)\mathbf{s}_j^i = B\mathbf{s}_{j-1}^i, \quad 1 \le j \le d_i, \quad 1 \le i \le m, \tag{3.4}$$

with $\mathbf{s}_0^i \equiv \mathbf{0}$. We can see that $\mathbf{s}_1^i$ are the eigenvectors corresponding to the eigenvalues λ_i for all $1 \le i \le m$.

Let Γ be a positively oriented simple closed curve enclosing the desired eigenvalues. Again without loss of generality, we let the eigenvalues of (1.1) enclosed by Γ be $\{\lambda_1, \ldots, \lambda_l\}$, and $s := d_1 + d_2 + \cdots + d_l$ be the number of eigenvalues inside Γ with multiplicity taken into account. Define the contour integral

$$Q := \frac{1}{2\pi\sqrt{-1}} \oint_\Gamma (zB - A)^{-1} B dz. \tag{3.5}$$

According to the residue theorem in complex analysis [1], it was verified in [32] that

$$Q = \frac{1}{2\pi\sqrt{-1}} \oint_\Gamma (zB - A)^{-1} B dz = S \begin{bmatrix} I_s & 0 \\ 0 & 0 \end{bmatrix} S^{-1} = S_{(:,1:s)} (S^{-1})_{(1:s,:)}. \tag{3.6}$$

One can show that $Q^2 = Q$, which means Q is a projector onto subspace $\mathcal{K} = \text{span}\{S_{(:,1:s)}\}$. Define

$$U := QY = S_{(:,1:s)} (S^{-1})_{(1:s,:)} Y, \tag{3.7}$$

where Y is an $n \times s$ random matrix. Therefore, U is the projection of Y onto the subspace $\mathcal{K}$. Now we would like to show that the columns of U form a basis for the subspace $\mathcal{K}$. We begin with

LEMMA 3.2 ([32]). *Let $Y \in \mathbb{R}^{n\times s}$. If the entries of Y are random numbers from a continuous distribution and that they are independent and identically distributed (i.i.d.), then with probability 1, the matrix $(S^{-1})_{(1:s,:)}Y$ is nonsingular.*

According to (3.7) and Lemma 3.2, we can conclude that the columns of U form a basis for the subspace $\mathcal{K}$. Note that $\mathcal{K}$ contains the eigenspace corresponding to the desired eigenvalues (see (3.4) for details), it is natural to take $\mathcal{K}$ as the right subspace. The FEAST algorithm takes advantage of the often used orthogonal projection technique to extract desired eigenpairs. The authors in [32] found that this extraction approach may fail to compute the desired eigenpairs for the FEAST algorithm when (1.1) is a non-Hermitian problem. To address this deficiency, they resorted to the oblique projection method, and developed a non-Hermitian FEAST algorithm. In their method, the left subspace is taken as $B\mathcal{K}$; the approximate eigenpairs $(\bar{\lambda}, \bar{\mathbf{x}})$ are obtained by imposing the Petrov-Galerkin condition [3, 24]:

$$(A\bar{\mathbf{x}} - \bar{\lambda}B\bar{\mathbf{x}}) \perp B\mathcal{K}, \tag{3.8}$$

where $\bar{\lambda} \in \mathbb{C}$ and $\bar{\mathbf{x}} \in \mathcal{K}$. It was shown in [32] that the columns of BU form a basis for $B\mathcal{K}$. Therefore (3.8) can be written in matrix form

$$(BU)^*(AU\mathbf{y} - \bar{\lambda}BU\mathbf{y}) = 0, \tag{3.9}$$

where $\mathbf{y} \in \mathbb{C}^s$ satisfying $\bar{\mathbf{x}} = U\mathbf{y}$. Accordingly, solving the eigenvalues of (1.1) inside Γ now is reduced to solve the projected eigenproblem

$$\overline{A}\mathbf{y} = \bar{\lambda}\overline{B}\mathbf{y}, \tag{3.10}$$

with

$$\overline{A} = (BU)^*AU \quad \text{and} \quad \overline{B} = (BU)^*BU. \tag{3.11}$$

The key to the success of the non-Hermitian FEAST algorithm proposed in [32] is that the left subspace is taken as $B\mathcal{K}$, instead of $\mathcal{K}$ used in the FEAST algorithm. Due to this, below we call this non-Hermitian FEAST algorithm BFEAST for the ease of reference. The following theorem justifies their choice of the left subspace.

THEOREM 3.3. *Let $\{(\bar{\lambda}_i, \mathbf{y}_i)\}_{i=1}^s$ be the eigenpairs of the projected eigenproblem (3.10). Then $\{(\bar{\lambda}_i, U\mathbf{y}_i)\}_{i=1}^s$ are the eigenpairs of (1.1) located inside Γ.*

In order to generate the projected eigenproblem (3.10), the most important task is to compute the projection U (see (3.7)). In practice, we have to compute U approximately by a quadrature rule:

$$U = QY = \frac{1}{2\pi\sqrt{-1}}\oint_\Gamma (zB - A)^{-1}BdzY \approx \frac{1}{2\pi\sqrt{-1}}\sum_{j=1}^{q}\omega_j(z_jB - A)^{-1}BY, \tag{3.12}$$

where z_j are the quadrature nodes on Γ associated with weights ω_j. From (3.12), we know that the dominant work is solving q linear systems of the form

$$(z_iB - A)X_i = BY. \tag{3.13}$$

The non-Hermitian FEAST algorithm (BFEAST) can be described as follows.

ALGORITHM 2. *Input $A, B \in \mathbb{C}^{n\times n}$, an i.i.d. random matrix $Y \in \mathbb{R}^{n\times t}$ where $t \geq s$, a closed curve Γ, a convergence tolerance ϵ, and "`max_iter`" to control the maximum*

*number of iterations. The function "*BFEAST*" computes eigenpairs* $(\bar{\lambda}_i, \bar{\mathbf{x}}_i)$ *of (1.1) that satisfies*

$$\bar{\lambda}_i \text{ inside } \Gamma \quad \text{and} \quad \frac{\|A\bar{\mathbf{x}}_i - \bar{\lambda}_i B\bar{\mathbf{x}}_i\|_2}{\|A\bar{\mathbf{x}}_i\|_2 + \|B\bar{\mathbf{x}}_i\|_2} < \epsilon. \tag{3.14}$$

The results are stored in the vector Λ_s *and the matrix* X_s*.*

Function $[\Lambda_s, X_s] = \text{BFEAST}(A, B, Y, \Gamma, \epsilon, \texttt{max\_iter})$
1. For $k = 1, \cdots, \texttt{max\_iter}$
2. Compute U approximately by the quadrature rule (3.12).
3. Compute QR decompositions: $U = U_1 R_1$ and $BU = U_2 R_2$.
4. Form $\overline{A} = U_2^* A U_1$ and $\overline{B} = U_2^* B U_1$.
5. Solve the projected eigenproblem $\overline{A}\mathbf{y} = \bar{\lambda}\overline{B}\mathbf{y}$ of size t to obtain eigenpairs $\{(\bar{\lambda}_i, \mathbf{y}_i)\}_{i=1}^t$. Set $\bar{\mathbf{x}}_i = U_1\mathbf{y}_i, i = 1, 2, \ldots, t$.
6. Set $\Lambda_s = [\,]$ and $X_s = [\,]$.
7. For $i = 1 : t$
8. If $(\bar{\lambda}_i, \bar{\mathbf{x}}_i)$ satisfies (3.14), then $\Lambda_s = [\Lambda_s, \bar{\lambda}_i]$ and $X_s = [X_s, \bar{\mathbf{x}}_i]$.
9. End
10. If there are s eigenpairs satisfying (3.14), stop. Otherwise, set $Y = U_1$.
11. End.

4. A randomized FEAST algorithms. In the previous two sections, we reviewed the FEAST algorithm, as well as its non-Hermitian variation, i.e., the BFEAST algorithm. In this part, we first formulate another non-Hermitian scheme for the FEAST algorithm. After that, the convergence analysis will be given to illustrate the effectiveness of our new method.

4.1. The derivation of our method. In [32], the authors used the oblique projection technique, rather than the wildly used orthogonal projection technique, to extend FEAST to the non-Hermitian problems. The key step is that they take the left subspace to $B\mathcal{K}$ instead of $\mathcal{K}$ used in the original FEAST algorithm. Here we present another scheme for the non-Hermitian FEAST algorithm. The intuition behind our new method is inspired by Lemma 3.2. The following theorem validates our intuition.

THEOREM 4.1. *Let* R *be an* $n \times s$ *random matrix, whose entries are independent and identically distributed (i.i.d.). Define*

$$\widetilde{A} = R^* A U \quad \textit{and} \quad \widetilde{B} = R^* B U. \tag{4.1}$$

Let $\{(\tilde{\lambda}_i, \mathbf{y}_i)\}_{i=1}^s$ *be the eigenpairs of the projected eigenproblem*

$$\widetilde{A}\mathbf{y} = \tilde{\lambda}\widetilde{B}\mathbf{y}. \tag{4.2}$$

Then $\{(\tilde{\lambda}_i, U\mathbf{y}_i)\}_{i=1}^s$ *are the eigenpairs of (1.1) located inside* Γ*.*

Proof. By (3.1), one can verify that

$$(\tilde{\lambda}B - A)S_{(:,1:s)} = (T^{-1})_{(:,1:s)}(\tilde{\lambda}I_{(1:s,1:s)} - J_{(1:s,1:s)}). \tag{4.3}$$

By (3.7), (4.1) and (4.3), we have

$$\begin{aligned} \tilde{\lambda}\widetilde{B} - \widetilde{A} &= R^*(\tilde{\lambda}B - A)U \\ &= R^*(\tilde{\lambda}B - A)S_{(:,1:s)}(S^{-1})_{(1:s,:)}Y \\ &= R^*(T^{-1})_{(:,1:s)}(\tilde{\lambda}I_{(1:s,1:s)} - J_{(1:s,1:s)})(S^{-1})_{(1:s,:)}Y. \end{aligned} \tag{4.4}$$

Therefore, the characteristic polynomial of the eigenproblem (4.2) is

$$\det(\tilde{\lambda}\widetilde{B} - \widetilde{A}) = \det(R^*(T^{-1})_{(:,1:s)}) \det(\tilde{\lambda} I_{(1:s,1:s)} - J_{(1:s,1:s)}) \det((S^{-1})_{(1:s,:)} Y).$$

By Lemma 3.2, we know that $R^*(T^{-1})_{(:,1:s)}$ and $(S^{-1})_{(1:s,:)}Y$ are nonsingular. As a result, due to the special structure of $J_{(1:s,1:s)}$, the roots of the characteristic polynomial $\det(\tilde{\lambda}\widetilde{B} - \widetilde{A})$ are $\lambda_1, \lambda_2, \ldots, \lambda_l$ with multiplicities $d_1, d_2, \ldots, d_l$ respectively.

Recall that $\lambda_1, \ldots, \lambda_l$ are not necessary distinct. Without loss of generality, let us consider the case where $\tilde{\lambda} = \lambda_1 = \lambda_2$ and $\tilde{\lambda} \neq \lambda_i$ for $3 \leqslant i \leqslant l$. Since $(\tilde{\lambda}\widetilde{B} - \widetilde{A})\mathbf{y} = 0$ and $R^*(T^{-1})_{(:,1:s)}$ is nonsingular, by (4.4) we have

$$(\tilde{\lambda} I_{(1:s,1:s)} - J_{(1:s,1:s)})(S^{-1})_{(1:s,:)} Y\mathbf{y} = 0.$$

The above equation in turn implies that $(S^{-1})_{(1:s,:)}Y\mathbf{y} = \alpha\mathbf{e}_1 + \beta\mathbf{e}_{d_1+1}$ for some scalars α and β not both zero due to the special structure of J, where $\mathbf{e}_1$ and $\mathbf{e}_{d_1+1}$ are the first and the (d_1+1)th columns of the $s \times s$ identity matrix, respectively. Therefore,

$$U\mathbf{y} = S_{(:,1:s)}(S^{-1})_{(1:s,:)}Y\mathbf{y} = \alpha S_{(:,1:s)}\mathbf{e}_1 + \beta S_{(:,1:s)}\mathbf{e}_{d_1+1}.$$

Note that the first and the (d_1+1)th columns of S are the eigenvector of (1.1) corresponding to the eigenvalues λ_1 and λ_2 respectively by (3.4). Thus, their linear combinations are the eigenvectors associated with $\tilde{\lambda}$. The proof is completed.
□

Theorem 4.1 tells us that the FEAST algorithm can deal with the non-Hermitian eigenproblems if we take the left subspace spanned by a random matrix. Due to the usage of random matrix, we call our new non-Hermitian FEAST algorithm RFEAST for ease of reference.

ALGORITHM 3. *Input $A, B \in \mathbb{C}^{n\times n}$, an i.i.d. random matrix $Y \in \mathbb{R}^{n\times t}$ where $t \geq s$, a closed curve Γ, a convergence tolerance ϵ, and "`max_iter`" to control the maximum number of iterations. The function "*RFEAST*" computes eigenpairs $(\tilde{\lambda}_i, \tilde{\mathbf{x}}_i)$ of (1.1) that satisfies*

$$\tilde{\lambda}_i \text{ inside } \Gamma \quad \text{and} \quad \frac{\|A\tilde{\mathbf{x}}_i - \tilde{\lambda}_i B\tilde{\mathbf{x}}_i\|_2}{\|A\tilde{\mathbf{x}}_i\|_2 + \|B\tilde{\mathbf{x}}_i\|_2} < \epsilon. \tag{4.5}$$

The results are stored in the vector Λ_s and the matrix X_s.

Function $[\Lambda_s, X_s] = \text{RFEAST}(A, B, Y, \Gamma, \epsilon, \texttt{max\_iter})$
1. For $k = 1, \cdots, \texttt{max\_iter}$
2. Compute U approximately by the quadrature rule (3.12).
3. Generate an $n \times t$ random matrix R, and compute QR decompositions: $U = U_1 R_1$ and $R = U_2 R_2$.
4. Form $\tilde{A} = U_2^* A U_1$ and $\tilde{B} = U_2^* B U_1$.
5. Solve the projected eigenproblem $\tilde{A}\mathbf{y} = \tilde{\lambda}\tilde{B}\mathbf{y}$ of size t to obtain eigenpairs $\{(\tilde{\lambda}_i, \mathbf{y}_i)\}_{i=1}^t$. Set $\tilde{\mathbf{x}}_i = U_1\mathbf{y}_i, i = 1, 2, \ldots, t$.
6. Set $\Lambda_s = [\,]$ and $X_s = [\,]$.
7. For $i = 1 : t$
8. If $(\tilde{\lambda}_i, \tilde{\mathbf{x}}_i)$ satisfies (4.5), then $\Lambda_s = [\Lambda_s, \tilde{\lambda}_i]$ and $X_s = [X_s, \tilde{\mathbf{x}}_i]$.
9. End
10. If there are s eigenpairs satisfying (4.5), stop. Otherwise, set $Y = U_1$.
11. End.

Our method can make the FEAST algorithm applicable to the non-Hermitian problems. Obviously, in each iteration, the dominant work in our method is to compute the projection U by a quadrature scheme, see (3.12) for details. As with other contour-integral based methods, our algorithm replaces the difficulty of solving the eigenvalue problem (1.1) by the difficulty of solving the linear systems (3.13), and has a good potential to be parallelized.

4.2. Convergence Analysis. In this part, we study the convergence properties of our new method (Algorithm 3) to show its effectiveness.

An important quantity for the convergence properties of projection methods is the distance of the exact eigenvector from the search subspace [24]. We begin the convergence analysis of our method from this perspective. For notational convenience, we represent the approximate projection computed in the kth iteration in Algorithm 3 by $U^{(k)}$.

Compute the contour integral

$$f(\mu) = \frac{1}{2\pi\sqrt{-1}} \oint_{\Gamma} \frac{1}{z-\mu} dz \tag{4.6}$$

by a q-point quadrature rule on Γ:

$$f(\mu) \approx \tilde{f}(\mu) = \frac{1}{2\pi\sqrt{-1}} \sum_{i=1}^{q} \frac{\omega_i}{z_i - \mu}, \tag{4.7}$$

where z_j are the quadrature nodes on Γ associated with weights ω_j. Let Γ be an unit circle with center at γ and $\mu = \gamma + re^{\sqrt{-1}\theta}, \theta \in (-\pi, \pi]$. Therefore, μ is located inside Γ when $0 \le r < 1$ and is located outside Γ when $r > 1$. In theory, according to the residue theorem, we have that $f(\mu) = 1$ if μ is located inside Γ and $f(\mu) = 0$ if μ is located outside Γ [1]. Compute the approximation $\tilde{f}(\mu)$ of $f(\mu)$ by the Gauss-Legendre quadrature with 16 integration points on Γ. FIG 4.1 depicts the magnitude of $|\tilde{f}(\mu)|$. We can see that $|\tilde{f}(\mu)|$ is close 1 when μ is contained inside Γ and is close to 0 when μ is outside Γ. Without loss of generality, assume that

$$|\tilde{f}(\lambda_1)| \ge |\tilde{f}(\lambda_2)| \ge \cdots \ge |\tilde{f}(\lambda_l)| > |\tilde{f}(\lambda_{l+1})| \ge \cdots \ge |\tilde{f}(\lambda_m)|. \tag{4.8}$$

THEOREM 4.2. *Let $d_0 = 0$, then $S_{(:,1+d_0+\ldots+d_{j-1})}$ is an eigenvector corresponding to λ_j. Let t be the size of starting vectors Y satisfying $t > s$, then there exists an integer $l', l' > l$, such that $\sum_i^{l'-1} d_i < t \le \sum_i^{l'} d_i$. Suppose the eigenvalues outside Γ are simple, which implies that $d_i = 1$ for $i = l+1, \ldots, m$. There exists a vector $v_j^{(k)} \in \mathrm{span}\{U^{(k)}\}$, $j = 1, \ldots, l$, such that*

$$\|S_{(:,1+d_0+\ldots+d_{j-1})} - v_j^{(k)}\|_2 \le \tau_j \big(\frac{|\tilde{f}(\lambda_{l'})|}{|\tilde{f}(\lambda_j)|}\big)^k, \tag{4.9}$$

where τ_j is a constant. In particular,

$$\|(I_n - Q_{(k)})S_{(:,1+d_0+\ldots+d_{j-1})}\|_2 \le \tau_j \big(\frac{|\tilde{f}(\lambda_{l'})|}{|\tilde{f}(\lambda_j)|}\big)^k, \tag{4.10}$$

where $Q_{(k)}$ is the orthogonal projector onto the subspace $\mathrm{span}\{U^{(k)}\}$.

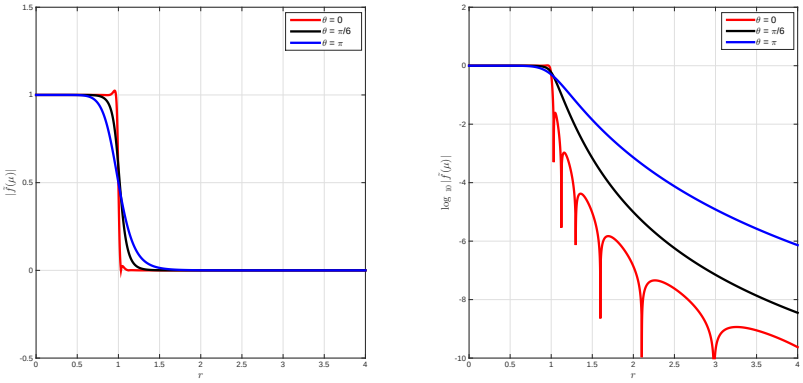

FIG. 4.1. *Here Γ is a unit circle with center at γ. The approximation $\tilde{f}(\mu)$, where $\mu = \gamma + re^{\sqrt{-1}\theta}, \theta \in (-\pi, \pi]$, is computed by the Gauss-Legendre quadrature with 16 quadrature nodes. Therefore, μ is located inside Γ when $0 \leq r < 1$ and is located outside Γ when $r > 1$. The left picture shows the general shape of $|\tilde{f}(\mu)|$, while the right one shows the logarithmic scale shape of the function.*

Proof. Let $U^{(0)} = Y$ be an $n \times t$ random matrix and $Z = (S^{-1})_{(1:t,:)}U^{(0)}$. By Lemma 3.2, we know Z is nonsingular, then

$$\begin{aligned} U^{(0)} &= SS^{-1}U^{(0)} = \left[S_{(:,1:t)}(S^{-1})_{(1:t,:)} + S_{(:,t+1:n)}(S^{-1})_{(t+1:n,:)}\right] U^{(0)} \\ &= \left[S_{(:,1:t)} + S_{(:,t+1:n)}(S^{-1})_{(t+1:n,:)}U^{(0)}Z^{-1}\right] Z \\ &= \left[(S_{(:,1:s)} + S_{(:,t+1:n)}E_{(0)}), V_{(0)}\right] Z, \end{aligned} \tag{4.11}$$

where $E_{(0)}$ is the first s columns of $(S^{-1})_{(t+1:n,:)}U^{(0)}Z^{-1}$, and $V_{(0)}$ is the last $(t-s)$ columns of matrix $S_{(:,1:t)} + S_{(:,t+1:n)}(S^{-1})_{(t+1:n,:)}U^{(0)}Z^{-1}$.

Under the assumption that the eigenvalues outside Γ are simple, the matrix N_{n-d} in (3.1) is a zero matrix. Let

$$D = \frac{1}{2\pi\sqrt{-1}} \sum_{j=1}^{q} \omega_j \begin{bmatrix} (z_j I_d - J_d)^{-1} & 0 \\ 0 & 0 \end{bmatrix}. \tag{4.12}$$

It was shown in [15, 31] that $|D_{(i,i)}| > 0$, for $i = 1, \ldots, s$. According to (2.5), (3.12), (4.11) and (4.12), we have

$$\begin{aligned} U^{(1)} &= \frac{1}{2\pi\sqrt{-1}} \sum_{j=1}^{q} \omega_j (z_j B - A)^{-1} B U^{(0)} = SDS^{-1}U^{(0)} \\ &= \left[(S_{(:,1:s)} + S_{(:,t+1:n)}E_{(1)})D_{(1:s,1:s)}, V_{(1)}\right] Z, \end{aligned} \tag{4.13}$$

where $E_{(1)} = D_{(t+1:n,t+1:n)}E_{(0)}(D_{(1:s,1:s)})^{-1}$ and $V_{(1)} = SDS^{-1}V_{(0)}$. Denote the QR decomposition of $U^{(k)}$ by $U^{(k)} = U_k R_k$. By induction, we have the following relationship

$$\begin{aligned} U^{(k)} &= SDS^{-1}U_{k-1} = SDS^{-1}U^{(k-1)}(R_{k-1})^{-1} \\ &= \left[(S_{(:,1:s)} + S_{(:,t+1:n)}E_{(k)})(D_{(1:s,1:s)})^k, V_{(k)}\right] Z(R_{k-1} \ldots R_1)^{-1}, \end{aligned} \tag{4.14}$$

where $E_{(k)} = (D_{(t+1:n,t+1:n)})^k E_{(0)}((D_{(1:s,1:s)})^{-1})^k$ and $V_{(k)} = (SDS^{-1})^k V_{(0)}$.

Since $Z(R_{k-1}\dots R_1)^{-1}$ is nonsingular, from (4.14) we conclude that the columns of $S_{(:,1:s)} + S_{(:,t+1:n)}E_{(k)}$ are in the subspace $\text{span}\{U^{(k)}\}$. In particular, vectors $v_j^{(k)} = S_{(:,1+d_0+\dots+d_{j-1})} + S_{(:,t+1:n)}(E_{(k)})_{(:,1+d_0+\dots+d_{j-1})} \in \text{span}\{U^{(k)}\}$ for $j = 1,\dots,l$. By the special structure of $(D_{(1:s,1:s)})^{-1})^k$, we have

$$\begin{aligned}\|S_{(:,1+d_0+\dots+d_{j-1})} - v_j^{(k)}\|_2 &= \|S_{(:,t+1:n)}(E_{(k)})_{(:,1+d_0+\dots+d_{j-1})}\|_2 \\ &= \big(\frac{1}{|\tilde{f}(\lambda_j)|}\big)^k \|S_{(:,t+1:n)}(D_{(t+1:n,t+1:n)})^k (E_{(0)})_{(:,1+d_0+\dots+d_{j-1})}\|_2 \\ &\le \tau_j \big(\frac{|\tilde{f}(\lambda_{l'})|}{|\tilde{f}(\lambda_j)|}\big)^k, \end{aligned} \tag{4.15}$$

where $\tau_j = \|S_{(:,t+1:n)}\|_2 \|(E_{(0)})_{(:,1+d_0+\dots+d_{j-1})}\|_2$.

Moreover,

$$\begin{aligned}\|(I_n - Q_{(k)})S_{(:,1+d_0+\dots+d_{j-1})}\|_2 &= \min_{v\in\text{span}\{U^{(k)}\}} \|S_{(:,1+d_0+\dots+d_{j-1})} - v\|_2 \\ &\le \tau_j \big(\frac{|\tilde{f}(\lambda_{l'})|}{|\tilde{f}(\lambda_j)|}\big)^k. \end{aligned} \tag{4.16}$$

□

Note that $l' > l$, which means $\lambda_{l'}$ is located outside Γ. Suppose the $|\tilde{f}(\lambda_{l'})|$ is about 1.0×10^{-3}, we can expect that there exists a vector $v_j^{(k)}$ in $\text{span}\{U^{(k)}\}$ such that $\|S_{(:,1+d_0+\dots+d_{j-1})} - v_j^{(k)}\|_2 \to 0$ at a rate of 10^{-3k}.

Let $P_{(k)}$ be the oblique projector onto the subspace $\text{span}\{U^{(k)}\}$ and orthogonal to the left subspace generated by a random matrix in the kth iteration in Algorithm 3. Define approximate operators $A_k = P_{(k)}AQ_{(k)}$ and $B_k = P_{(k)}BQ_{(k)}$. The following theorem gives an upper bound for the residual norm of the exact eigenpair with respect to the approximate operator pair (A_k, B_k).

THEOREM 4.3. *Let $\sigma^{(k)} = \|P_{(k)}(A-\lambda B)(I_n - Q_{(k)})\|_2$. Then the residual norms of $(\lambda_j, S_{(:,1+d_0+\dots+d_{j-1})}), j = 1,\dots,l$, for the approximate operator pair (A_k, B_k) saftisfy*

$$\|(A_k - \lambda_j B_k)S_{(:,(:,1+d_0+\dots+d_{j-1}))}\|_2 \le \sigma^{(k)}\tau_j \big(\frac{|\tilde{f}(\lambda_{l'})|}{|\tilde{f}(\lambda_j)|}\big)^k \tag{4.17}$$

Proof. Similar to the proof of Lemma 2 in [15] and Theorem 4.7 in [24], we have

$$\begin{aligned}\|(A_k - \lambda_j B_k)S_{(:,1+d_0+\dots+d_{j-1})}\|_2 &= \|P_{(k)}(A - \lambda_j B)(I_n - Q_{(k)})S_{(:,1+d_0+\dots+d_{j-1})}\|_2 \\ &= \|P_{(k)}(A - \lambda_j B)(I_n - Q_{(k)})(I_n - Q_{(k)})S_{(:,1+d_0+\dots+d_{j-1})}\|_2 \\ &\le \sigma^{(k)}\|(I_n - Q_{(k)})S_{(:,1+d_0+\dots+d_{j-1})}\|_2. \end{aligned}$$

By (4.10), we establish our result (4.17). □

Theorem 4.3 says that the residual norm associated with the exact eigenpair $(\lambda_j, S_{(:,1+d_0+\dots+d_{j-1})})$ converges at a rate of $\big(|\tilde{f}(\lambda_{l'})|/|\tilde{f}(\lambda_j)|\big)$ with respect to the iteration counts.

TABLE 5.1
Test problems from Matrix Market that are used in our experiments.

No.	Problem	Type	n	Region: (γ, ρ)	s
1	BFW782	gen.	782	$(-6.0 \times 10^5, 3.0 \times 10^5)$	230
2	DWG961	gen.	961	$(5.0 \times 10^2, 2.0 \times 10^2)$	157
3	UTM1700	gen.	1700	$(4.0, 1.0)$	96
4	MHD4800	gen.	4800	$(-6.0, 3.0)$	169
5	OLM5000	stand.	5000	$(-1.0 \times 10^4, 0.6 \times 10^4)$	204
6	DW8192	stand.	8192	$(1.0, 0.2)$	270

5. Numerical Experiments. In this section, we present some numerical experiments to demonstrate the numerical performance of our new non-Hermitian FEAST algorithm (RFEAST). The experiments are organized into threes sets. The first set aims at demonstrating the convergence behavior of our new method. The second set is devoted to comparing our technique with another non-Hermitian variant of FEAST, that is the BFEAST algorithm (Algorithm 2). In the last set, we would like to compare our RFEAST method with the MATLAB built-in function `eig`. For the approximation eigenpairs $(\tilde{\lambda}_i, \tilde{\mathbf{x}}_i)$, define the relative residual norms

$$r_i = \frac{\|A\tilde{\mathbf{x}}_i - \tilde{\lambda}_i B\tilde{\mathbf{x}}_i\|_2}{\|A\tilde{\mathbf{x}}_i\|_2 + \|B\tilde{\mathbf{x}}_i\|_2}. \tag{5.1}$$

We use the maximum relative residual norm defined as $\texttt{Res} = \max_{1\le i\le s} r_i$ to assess the accuracy achieved by the test methods. 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 test matrices presented in TABLE 5.1 are available from the Matrix Market collection[1]. They are the real-world problems from scientific and engineering applications. All test problems are non-Hermitian. The first four test problems are generalized eigenvalue problems and the last two test problems are standard. The region of interest for each test problem is a circle with center at γ and radius ρ. The value of s is the number of eigenvalues inside the target region. In all experiments, we use the Gauss-Legendre quadrature with 16 quadrature nodes to compute the approximate projection U (see (3.12)). The generalized shifted linear systems (see (3.13)) involved are computed by direct method. We first use the MATLAB function `lu` to compute the LU decomposition of $A - z_j B, j = 1, 2, \ldots, q$, and then perform the triangular substitutions to get the corresponding solutions.

5.1. The convergence behaviour. The FEAST algorithm is a stable and fast technique [18, 30]. It was formulated for the Hermitian problems [21]. The goal of our work is to adapt FEAST to the non-Hermitian cases. Meanwhile, we hope that our method retains the effectiveness of the FEAST algorithm. The objective of this experiment is two-fold. First, we would like to validate the convergence properties analysed in Section 4. Second, we would like to demonstrate the influence of the size of starting vectors on our new method.

In each iteration there are $t - s$ spurious eigenvalues. The spurious eigenvalues outside the target region can be easily detected according to their coordinates. For the spurious eigenvalues inside the target region, in [32] the authors introduced a tolerance η to filter them. The idea behind is that the spurious eigenvalues can not

[1] http://math.nist.gov/MatrixMarket/

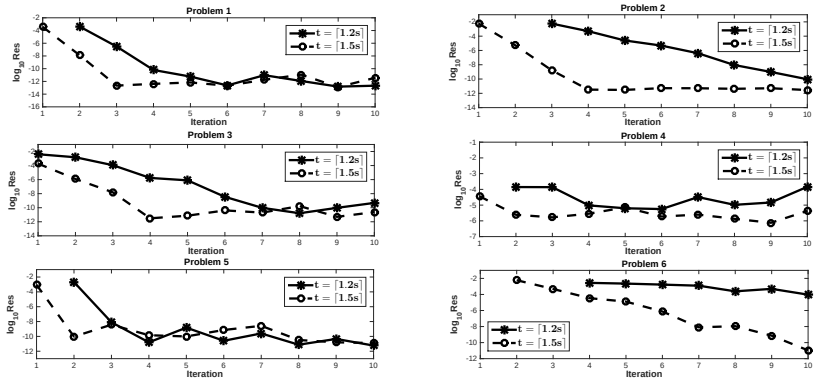

FIG. 5.1. *The maximum relative residual norms in different iterations.*

achieve high accuracy; as the iteration process proceeds, there will be a gap in accuracy between the desired eigenpairs and the spurious ones. If the relative residual norm of an eigenpair is less that η, then the eigenpair is viewed as desired one and referred as filtered eigenpair. In the experiment, we set the filtering tolerance $\eta = 1.0 \times 10^{-2}$. In FIG 5.1, we plot the `Res`'s from the iteration that the number of filtered eigenpairs attains s for the first time to the 10th iteration for the cases $t = \lceil 1.2s \rceil$ and $t = \lceil 1.5s \rceil$, respectively. Here, we assume that the number s of eigenvalues inside the region of interest is known. Theorem 4.3 tells us the residual norm will converge with the factor $|\tilde{f}(\lambda_{l'})|/|\tilde{f}(\lambda_i)|$ for the exact eigenpair $(\lambda_i, \mathbf{x}_i)$ with respect to the iteration counts. FIG 5.1 shows the maximum relative residual norm `Res` decreases monotonically, as expected, until the accuracy can not be further improved. On the other hand, a larger subspace size t leads to a smaller $|\tilde{f}(\lambda_{l'})|$, and then leads to faster convergence. Taking the Problem 2 as an example, it is clear to see that our method converges almost linearly with a factor for both $t = \lceil 1.5s \rceil$ and $t = \lceil 1.2s \rceil$. Precisely, the convergence rate is ablout 1.0×10^{-4} for the former case and is about 1.0×10^{-2} for the latter. To converge to the minimum residual norm, which is about 1.0×10^{-13}, it needs 4 iterations when we take the size t to $\lceil 1.5s \rceil$, but 10 iterations are required for the case $t = \lceil 1.2s \rceil$.

Increasing the value of t will lead to a faster convergence rate, however, it also results in a considerable increase in computational cost in each iteration since t represents the number of the right-hand sides in each shifted linear system involved (see (3.13)).

5.2. Comparisons with BFEAST. Both RFEAST and BFEAST [32] aim to make the FEAST algorithm applicable for the non-Hermitian problems. The only difference between the two non-Hermitian FEAST methods is the choice of the left subspace. In BFEAST, the left subspace is spanned by BU (see (3.7)), while in our method the left subspace is spanned by a random matrix. The dominant work of both methods is computing the approximate projection U (see (3.12)). Thus the computational cost required by both non-Hermitian FEAST algorithms in each iteration is almost the same. Due to this, in this experiment we compare the numerical performance of the two methods through the accuracy achieved in each iteration.

In [32], the authors presented a technique to select a suitable size of the starting

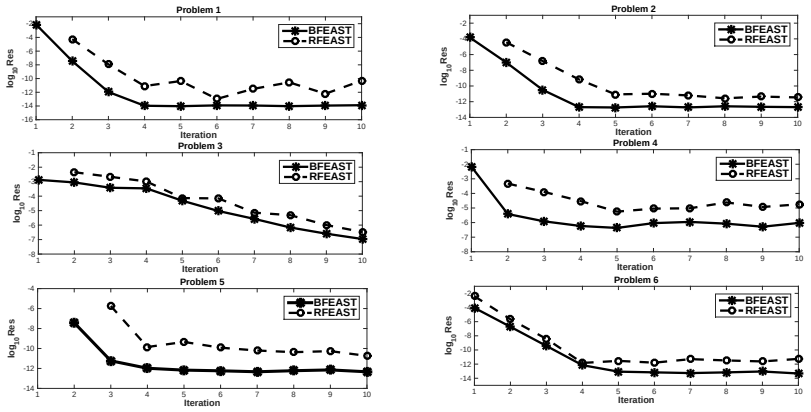

FIG. 5.2. *The convergence behavior of two non-Hermitian FEAST algorithms.*

vectors Y for the BFEAST algorithm. To facilitate the comparisons, here we also use this technique to start our method. We depict `Res`'s computed by the two test methods from the iteration that the number of filtered eigenpairs attains s for the first time to the 10th iteration in FIG 5.2. As with the previous experiment, the filtering tolerance η is also taken to 1.0×10^{-2}. The convergence curves of two non-Hermitian FEAST methods are almost parallel, which means the two methods converge with almost the same rate. We have shown in Theorem 4.3 that the upper bound for the residual norms of exact eigenpairs $(\lambda_j, \mathbf{x}_j)$ are $\sigma^{(k)}\tau_j\big(|\tilde{f}(\lambda_{l'})|/|\tilde{f}(\lambda_j)|\big)^k$ in the kth iteration (see (4.17)). Recall that our method shares the same right subspace with the BFEAST algorithm. In view of the proof of Theorem 4.3, we are able to establish a similar upper bound for the BFEAST algorithm simply via replacing the oblique projector $P_{(k)}$ in the expression of $\sigma^{(k)}$ with $Z_{(k)}$, where $Z_{(k)}$ is the oblique projector onto $\mathrm{span}\{U^{(k)}\}$ and orthogonal to the left subspace $B\mathcal{K}$. More precisely, we can write the upper bound for BFEAST as $\kappa^{(k)}\tau_j\big(|\tilde{f}(\lambda_{l'})|/|\tilde{f}(\lambda_j)|\big)^k$, where $\kappa^{(k)} = \|Z_{(k)}(A - \lambda B)(I_n - Q_{(k)})\|_2$. Therefore, the two methods have almost the same convergence rate, which is $|\tilde{f}(\lambda_{l'})|/|\tilde{f}(\lambda_j)|$ for the eigenpair $(\lambda_j, \mathbf{x}_j)$. This can interpret why two non-Hermitian FEAST algorithms exhibit essentially the same convergence behavior.

On the other hand, it can be seen from FIG 5.2 that BFEAST performs better than our method in all test problems in terms of accuracy. The only difference in the upper bounds between the two methods is the constants $\kappa^{(k)}$ and $\sigma^{(k)}$, due to the different choices of the left subspaces. The BFEAST algorithm works better than our method possibly because the constant $\kappa^{(k)}$ in the BFEAST algorithm is smaller than $\sigma^{(k)}$ in our method, and therefore the upper bound in BFEAST is sharper than the one in our our method.

5.3. Comparisons with Matlab's `eig` function. In this experiment, we compare our method with the MATLAB built-in function `eig` in terms of timing. Since the target eigenvalues are the interior ones of non-Hermitian problems, when using `eig` to compute the eigenvalues inside the regions presented in TAB 5.1, we have to first compute all eigenvalues in dense format and then select the target eigenvalues according to their coordinates. In our method, we set the convergence tolerance ϵ to

TABLE 5.2
Comparison of `eig` *and our method in terms of timing.*

No.	`eig`	Our method
1	13.87	3.22
2	14.13	10.94
3	68.40	35.96
4	2719.36	180.07
5	2397.71	27.58
6	77653.86	398.82

1.0×10^{-8} and take the parameter `max_iter` $= 10$.

The amount of time, which is measured in seconds, required by `eig` and our RFEAST algorithm is reported in TABLE 5.2. It is clear to see that our method is much faster than the MATLAB function `eig`, although the parallelism offered by our method is not used in the tests. The difference in CPU times is more obvious when the size of test problem grows larger. Therefore, our method is much more efficient than the MATLAB's `eig` function.

6. Conclusions. In this work, we have developed a new scheme to make the FEAST algorithm applicable for the non-Hermitian problems. The key step is that the left subspace used to extract the desired eigenpairs in our method is spanned by a random matrix. Theoretical analysis shown that our method can deal with the non-Hermitian cases. The resulting method retains the feature of parallelism offered by the original FEAST algorithm and does not increase the computational cost. The convergence properties of our new method were also investigated. Numerical experiments were reported to demonstrate the numerical performance of our new method and to validated the convergence analysis.

7. Acknowledgment. I would like to thank Professor Raymond H. Chan, my thesis advisor, at The Chinese University of Hong Kong and Professor Man-Chung Yeung at University of Wyoming for their help and fruitful discussions in the preparation of this paper.

REFERENCES

[1] L. AHLFORS, *Complex Analysis*, 3rd Edition, McGraw-Hill, Inc., 1979.
[2] A. P. AUSTIN AND L. N. TREFETHEN, *Computing eigenvalues of real real symmetric matrices with rational filters in real arithmetic*, preprint.
[3] Z. BAI, J. DEMMEL, J. DONGARRA, A. RUHE, AND H. VAN DER VORST, *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide* , SIAM, Philadelphia, 2000.
[4] Z. BAI, J. DEMMEL, AND M. GU, *An inverse free parallel spectral divide and conquer algorithm for nonsymmetric eigenproblem*, Numer. Math., 76 (1997), pp. 279–308.
[5] W.-J. BEYN, *An integral method for solving nonlinear eigenvalue problems*, Lin. Alg. Appl., 436 (2012), pp. 3839–3863
[6] K. A. CLIFFE, A. SPENCE, AND S. J. TAVENER, *The numerical analysis of bifurcation problems with application to fluid mechanics*, Acta Numer., 9 (2000), pp. 39–131.
[7] P. J. DAVIS AND P. RABINOWITZ, *Methods of numerical integration*, 2nd Edition, Academic Press, Orlando, FL, 1984.
[8] J. DEMMEL, *Applied Numerical Linear Algebra*, SIAM, Philadelphia, 1997.
[9] E. DI NAPOLI, E. POLIZZI, AND Y. SAAD, *Efficient estimation of eigenvalue counts in an interval*, http://arxiv.org/abs/1308.4275.
[10] B. FORD AND G. HALL, *The generalized eigenvalue problem in quantum chemistry*, Comput. Phys. Commun., 8 (1974), pp. 337–348.
[11] K. GALLIVAN, E. GRIMME, AND P. VAN DOOREN, *A rational Lanczos algorithm for model reduction*, Numer. Algorithms, 12 (1996), pp. 33–64.

[12] F. R. Gantmacher, *The Theory of Matrices*, Chelsea, New York, 1959.
[13] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd Edition, Johns Hopkins University Press, Baltimore, MD, 1996.
[14] R. G. Grimes, J. D. Lewis, and H. D. Simon, *A shifted block Lanczos algorithm for solving sparse symmetric generalized eigenproblems*, SIAM J. Matrix Anal. Appl., 15 (1994), pp. 228–272.
[15] A. Imakura, L. Du, and T. Sakurai, *Error bounds of Rayleigh-Ritz type contour integral-based eigensolver for solving generalized eigenvalue problems*, Numer. Algor., (accepted).
[16] T. Ikegami and T. Sakurai, *Contour integral eigensolver for non-Hermitian systems: a Rayleigh-Ritz-type approach*, Taiwanese J. Math., 14 (2010), pp. 825–837.
[17] T. Ikegami, T. Sakurai, and U. Nagashima, *A filter diagonalization for generalized eigenvalue problems based on the Sakurai-Sugiura projection method*, J. Comp. Appl. Math., 233 (2010), pp. 1927–1936.
[18] L. Krämer, E. Di Napoli, M. Galgon, B. Lang, and P. Bientinesi, *Dissecting the FEAST algorithm for generalized eigenproblems*, J. Comput. Appl. Math., 244 (2013), pp. 1–9.
[19] C. B. Moler and G. W. Stewart, *An algorithm for generalized matrix eigenvalue problems*, SIAM J. Numer. Anal., 10 (1973), pp. 241–256.
[20] Y. Nakatsukasa and N. J. Higham, *Stable and efficient spectral divide and conquer algorithms for the symmetric eigenvalue decomposition and the SVD*, SIAM J. Sci. Comput., 35 (2013), pp. A1325–A1349.
[21] E. Polizzi, *Density-matrix-based algorithm for solving eigenvalue problems*, Phys. Rev. B 79 (2009), 115112.
[22] W. P. Reinhardt, *Complex coordinates in the theory of atomic and molecular structure and dynamics*, Annu. Rev. Phys. Chem., 33 (1982), pp. 223–255.
[23] A. Ruhe, *Rational Krylov: A practical algorithm for large sparse nonsymmetric matrix pencils*, SIAM J. Sci. Comput., 19 (1998), pp. 1535–1551.
[24] Y. Saad, *Numerical Methods for Large Eigenvalue Problems*, SIAM, Philadelphia, 2011.
[25] T. Sakurai, Y. Futamura, and H. Tadano, *Efficient parameter estimation and implementation of a contour integral-based eigensolver*, J. Alg. Comput. Tech., 7 (2013), pp. 249–269.
[26] T. Sakurai and H. Sugiura, *A projection method for generalized eigenvalue problems using numerical integration*, J. comput. Appl. Math., 159 (2003), pp. 119–128.
[27] T. Sakurai and H. Tadano, *CIRR: A Rayleigh–Ritz type method with contour integral for generalized eigenvalue problems*, Hokkaido Math. J. 36 (2007), pp. 745–757.
[28] B. K. Sriperumbudur, D. A. Torres, and G. R. G. Lanckriet, *A majorization-minimization approach to the sparse generalized eigenvalue problem*, Mach. Learn., 85 (2011), pp. 3–39.
[29] G. W. Stewart, *Matrix Algorithms, Vol. II, Eigensystems*, SIAM, Philadelphia, 2001.
[30] P. T. P. Tang and E. Polizzi, *FEAST as a subspace iteration eigensolver accelerated by approximate spectral projection*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 354–390.
[31] G. Yin, *A Contour-integral Based Method for Counting the Eigenvalues Inside a Region in the Complex Plane*, https://arxiv.org/abs/1503.05035.
[32] G. Yin, R. H. Chan, and M-C. Yeung, *A FEAST Algorithm with oblique projection for generalized eigenvalue problems*, http://arxiv.org/abs/1404.1768.