SIAM J. Matrix Anal. Appl.
Vol. 46, No. 2, pp. 1037–1060
© 2025 Society for Industrial and Applied Mathematics

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

# INTERIOR EIGENSOLVER BASED ON RATIONAL FILTER WITH COMPOSITE RULE*

YUER CHEN† AND YINGZHOU LI‡

**Abstract.** The contour integral–based rational filter leads to interior eigensolvers for non-Hermitian generalized eigenvalue problems. Based on Zolotarev's third problem, this paper proves the asymptotic optimality of the trapezoidal quadrature of the contour integral in terms of the spectrum separation. A composite rule of the trapezoidal quadrature is derived, and two interior eigensolvers are proposed based on it. Both eigensolvers adopt direct factorization and the multishift generalized minimal residual method for the inner and outer rational functions, respectively. The first eigensolver fixes the order of the outer rational function and applies subspace iterations to achieve convergence, whereas the second eigensolver doubles the order of the outer rational function every iteration to achieve convergence without subspace iteration. The efficiency and stability of proposed eigensolvers are demonstrated on synthetic and practical sparse matrix pencils.

**Key words.** generalized eigenvalue problem, non-Hermitian matrix, contour integral, trapezoidal quadrature, rational optimality, Zolotarev's third problem

**MSC code.** 65F15

**DOI.** 10.1137/24M1654713

## 1. Introduction.

We aim to solve the large-scale interior eigenvalue problem for non-Hermitian matrices. Such problems arise from many fields including but not limited to electronic structure calculations, dynamic system simulations, and control theory. Most of these applications only require part of the eigenvalues of interest, and many of these are interior eigenvalues.

The interior non-Hermitian generalized eigenvalue problem we consider is

$$Ax_i = \lambda_i B x_i, \quad \lambda_i \in \mathcal{D}, \tag{1.1}$$

where $\mathcal{D}$ is the region of interest, matrix pencil $(A, B)$ is diagonalizable, and its eigenvalues are distributed on the complex plane. The goal is to find all eigenpairs $(\lambda_i, x_i)$ in the region $\mathcal{D}$. The interior eigensolver for this problem can be used to compute eigenpairs in several regions in parallel to obtain the partial or full eigendecomposition of the matrix of interest.

### 1.1. Related work.

Methods for non-Hermitian generalized eigenvalue problems have been developed for decades. The QZ method [9] is a popular one in practice for dense and small- to medium-scale matrices. When a sparse and large-scale matrix is considered, iterative methods [7, 17] are preferred.

*Received by the editors April 17, 2024; accepted for publication (in revised form) by P. Boito December 4, 2024; published electronically April 10, 2025.
https://doi.org/10.1137/24M1654713

**Funding:** This research was supported by the National Natural Science Foundation of China (NSFC) under grant 12271109, the Science and Technology Commission of Shanghai Municipality (STCSM) under grants 22TQ017 and 24DP2600100, and the Shanghai Institute for Mathematics and Interdisciplinary Sciences (SIMIS) under grant SIMIS-ID-2024-(CN).

†School of Mathematical Sciences, Fudan University, Shanghai, 200433 China (yechen20@fudan.edu.cn).

‡School of Mathematical Sciences, Fudan University, Shanghai, 200433 China, and Shanghai Key Laboratory for Contemporary Applied Mathematics, Shanghai, China (yingzhouli@fudan.edu.cn).

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Among iterative methods, many adopt the combination of a contour-based filter and the subspace iteration, e.g., the Sakurai–Sugiura (SS) method [15] and variants of the FEAST method [8, 13]. The original SS method suffers from numerical instability due to the ill-conditioned Hankel matrix. Then Sakurai and Sugiura propose CIRR [4], which uses Rayleigh–Ritz projection to avoid the explicit usage of the momentum and block version SS method [5]. The number of linear systems therein is reduced, and so is the order of the Hankel matrix. The FEAST method originally proposed for Hermitian matrices is extended to non-Hermitian matrices and results in many variants—dual FEAST [6], BFEAST [19], HFEAST [18], etc.

For all the contour-based filters or rational filters in the methods above, the convergence and convergence rate highly depend on the locations and weights of poles. Although the trapezoidal quadrature[1] leads to a good convergence behavior [6], its optimality remains unknown for non-Hermitian matrices. In this paper, we discuss the optimality of the trapezoidal quadrature and its composite rule property. Based on the composite rule, we propose two interior eigensolvers for non-Hermitian generalized eigenvalue problems.

**1.2. Contribution.** Our contribution in this paper has two parts, the theory part and the algorithm part. These two parts are later referred to as the filter design and filter implementation, respectively.

In filter design, we find the optimal rational filter in the sense of spectrum separation for fast convergence of subspace iteration. By making use of the connection with Zolotarev's third problem, we prove that when the contour is a circle, the rational filter used in the inverse power method leads to an optimal separation, while the trapezoidal quadrature leads to an asymptotically optimal separation.

In filter implementation, we focus on the flexibility of the trapezoidal quadrature implementation. The main cost of the rational filter $R_k(z)$ implementation comes from prefactorization, e.g., LU factorization, and solving phase, e.g., forward and backward substitution (triangular solve). In general, the cost of the prefactorization is of higher-order complexity than that of the solving phase. The composite rule of trapezoidal quadrature establishes a trade-off between the costs of the two.

Specifically, given a rational filter $R_k(z)$ from the trapezoidal quadrature, we derive a composite rule as $R_k(z) = R_{k_2}(M(R_{k_1}(z)))$ for $k = k_1 k_2$ and $M(\cdot)$ being a Möbius transform. Motivated by [8], two novel algorithms are proposed based on the composite rule, both of which implement $R_k(\cdot)$ with an inner-outer structure. The inner rational function $R_{k_1}(\cdot)$ is implemented with direct matrix factorizations, whereas the outer rational function $R_{k_2}(\cdot)$ is implemented via the multishift generalized minimal residual method (GMRES). The first algorithm adopts the subspace iteration framework. It substitutes matrix factorizations with the solves, which may reduce the total runtime for large cases. The second algorithm discards the framework of subspace iteration. It achieves target precision by dynamically increasing $k_2$ and reusing Krylov subspace to avoid the increase of the computational cost. These two algorithms hybridize the direct method and iterative method and enable us to use computational resources more effectively. Through the numerical experiments, we find that the second algorithm is more efficient and practical.

**1.3. Organization.** The rest of this paper is organized as follows. In section 2, we introduce the basic idea of the contour integral–based filter and discuss the computational cost and memory cost of the implementation based on the simple rule

[1]Trapezoidal rule instead of trapezoidal quadrature is a commonly used terminology.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

of trapezoidal quadrature. Later, we introduce our two novel algorithms based on the composite rule of trapezoidal quadrature in section 3. The optimality of rational functions is discussed in section 4, and the derivation of the composite rule is presented in section 5. In section 6, there are some numerical experiments to examine our results of optimality and demonstrate the efficiency of two proposed algorithms. Finally, section 7 concludes the paper.

## 2. Subspace iteration with rational filter.

Subspace iteration with rational filter is a class of eigensolvers for interior non-Hermitian generalized eigenvalue problems (1.1). All eigensolvers in this class use the subspace iteration framework and adopt various filters, i.e., rational functions with different choices of weights and poles. In this section, we will first review the framework of subspace iteration and introduce the contour integral formulation of the spectral projector. Then we demonstrate the process of deriving rational filters from numerical discretizations, alongside the cost of implementing this kind of rational filter.

### 2.1. Subspace iteration.

The general framework of the subspace iteration with filter iterates between two phases: (1) refining the subspace via filter, and (2) solving a reduced eigenvalue problem in the subspace.

In the first phase, the filter is applied to an approximate basis of the eigensubspace, and a refined representation of the eigensubspace is obtained. For non-Hermitian eigenvalue problems, left and right eigensubspaces are different. We can refine left and right eigensubspaces by applying the filter twice [6], or we can only refine the right eigensubspace and use an extra step to obtain an approximation of the left eigensubspace [18]. In the second phase, the original large-scale eigenvalue problem is projected to the left and right eigensubspaces and reduced to an eigenvalue problem of a much smaller scale. Then the small-scale eigenvalue problem is solved by classical dense eigensolvers, which results in the approximate eigenpairs of the original problem.

Due to the potentially ill-conditioned eigenbasis of non-Hermitian matrices, the generalized Schur vectors could be extracted to represent the eigensubspaces and lead to a more stable scheme. Such a subspace iteration idea has been combined with FEAST for non-Hermitian matrices and results in HFEAST [18]. Denote the approximate basis of the right eigensubspace as $U$. The orthonormal basis of $U$ is denoted as $V = \mathrm{orth}(U)$. As in HFEAST [18], the orthonormal basis of the left eigensubspace could be constructed as $W = \mathrm{orth}(AV - \sigma BV)$, where $\sigma$ is the shift away from the eigenvalues of $(A, B)$. After obtaining the approximate orthonormal basis of the left and right eigensubspaces, the reduced generalized eigenvalue problem $(W^*AV, W^*BV)$ is solved by the QZ algorithm and yields the generalized Schur form,

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

where $P_L$ and $P_R$ are orthogonal matrices, and $H_A$ and $H_B$ are upper triangular matrices. The approximate eigenvalues are

$$\tilde{\lambda}_i = (H_A)_{i,i}/(H_B)_{i,i}.$$

We further calculate the left and right eigenvectors of $(H_A, H_B)$ and denote them as $V_L$ and $V_R$, respectively. The approximate left and right eigenvectors of $(A, B)$ are, respectively,

$$WP_LV_L \quad \text{and} \quad VP_RV_R.$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

**Algorithm 2.1.** Subspace iteration with filter.

```
Input: matrix pencil $(A,B)$, region $\mathcal{D}$, number of columns $n_{\text{col}}$, shift $\sigma$.
Output: All approximate eigenpairs $(\widetilde{\lambda}_i, \widetilde{x}_i), \widetilde{\lambda}_i \in \mathcal{D}$.
1: Generate random $Y^{N\times n_{\text{col}}}$
2: while not converge do
3:     $U = \rho(B^{-1}A)Y$
4:     $V = \text{orth}(U)$
5:     $W = \text{orth}(AV - \sigma BV)$
6:     $[H_A, H_B, P_L, P_R, V_L, V_R] = \text{qz}(W^*AV, W^*BV)$
7:     $\widetilde{\lambda}_i = (H_A)_{i,i}/(H_B)_{i,i}$, $\widetilde{x}_i = VP_R(V_R)(:,i)$ and $Y = (\widetilde{x}_1, \ldots, \widetilde{x}_{n_{\text{col}}})$
8: end while
```

The overall framework of the subspace iteration in HFEAST [18] with filter $\rho(\cdot)$ is summarized in Algorithm 2.1. In the rest of the paper, we adopt the subspace iteration as in Algorithm 2.1 and focus on the design and implementation of $\rho(\cdot)$. In addition, we abbreviate lines 4–7 as $[Y, \widetilde{\Lambda}, \widetilde{X}] = \text{HSRR}(A, B, U, \sigma)$, where $\widetilde{\Lambda} = \text{diag}\{\widetilde{\lambda}_1, \ldots, \widetilde{\lambda}_{n_{\text{col}}}\}$ and $\widetilde{X} = (\widetilde{x}_1, \ldots, \widetilde{x}_{n_{\text{col}}})$.

**2.2. Contour-based filter and discretization.** The basic idea of designing a filter is to construct a matrix function whose eigenvalues are close to zero outside the region $\mathcal{D}$ and different from zero inside $\mathcal{D}$. One good choice of matrix function is the indicator function of $\mathcal{D}$, which could be constructed via a contour integral enclosing the region $\mathcal{D}$. The indicator function of $\mathcal{D}$ via contour integral admits

$$f(z) = \frac{1}{2\pi\imath}\oint_\Gamma \frac{1}{\zeta - z}\,\mathrm{d}\zeta = \begin{cases} 1, & z \in \mathcal{D}, \\ 0, & z \notin \bar{\mathcal{D}}, \end{cases} \tag{2.1}$$

where $\Gamma$ is the positively oriented Jordan curve that encloses the region $\mathcal{D}$.

For a diagonalizable matrix pencil $(A,B)$, i.e.,

$$AX = BX\Lambda,$$

with $X$ being the right eigenvectors and $\Lambda$ a diagonal matrix with eigenvalues on its diagonal, the indicator function $f(z)$ applied to $(A,B)$ admits

$$\begin{aligned} f(B^{-1}A) &= \frac{1}{2\pi\imath}\oint_\Gamma (\zeta B - A)^{-1} B\,\mathrm{d}\zeta \\ &= X\left[\frac{1}{2\pi\imath}\oint_\Gamma (\zeta I - \Lambda)^{-1}\,\mathrm{d}\zeta\right] X^{-1} = X\mathbb{1}_{\mathcal{D}}(\Lambda)X^{-1}, \end{aligned} \tag{2.2}$$

where $\mathbb{1}_{\mathcal{D}}(\cdot)$ denotes the indicator function for region $\mathcal{D}$.[2] In [19], a result similar to (2.2) is proved, which leads to the theoretical foundation that the contour integral works even if the non-Hermitian system is defective. Various numerical discretizations of the contour integral (2.2) lead to various filters. In many applications, especially non-Hermitian eigenvalue problems, the contour $\Gamma$ is circular. In other applications, the contour could be conformally mapped to a circle. Hence, in this paper, we focus on the case that $\Gamma$ is a circle.

[2] In (2.2), we implicitly assume that the eigenvalues of $(A,B)$ do not locate on the boundary of $\mathcal{D}$.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

We could reparameterize the circle by $e^{\imath\theta}$ for $\theta \in [0, 2\pi)$. The integral (2.1), then, is a one-dimensional integral and could be numerically evaluated by various quadrature rules. Generally, the contour integral (2.1) can be approximated by a numerical quadrature with $k$ quadrature points, written as

$$R_k(z) = \sum_{i=1}^{k} \frac{w_i^{(k)}}{p_i^{(k)} - z}, \tag{2.3}$$

where $\{w_i^{(k)}\}_{i=1}^k$ are weights and $\{p_i^{(k)}\}_{i=1}^k$ are poles. Let

$$\mathcal{R}_{n,m} = \{P(z)/Q(z) : \deg(P(z)) \le n, \deg(Q(z)) \le m\}$$

be the set of rational functions, where $P(z)$ and $Q(z)$ are polynomials and $\deg(\cdot)$ denotes the degree of the polynomial. It is easy to see that $R_k(z) \in \mathcal{R}_{k,k}$, i.e., it is a $k$th-order rational function.

When the discretized contour integral is applied to $(A, B)$, it yields a $k$th-order rational matrix function

$$R_k(B^{-1}A) = \sum_{i=1}^{k} w_i^{(k)} (p_i^{(k)} B - A)^{-1} B. \tag{2.4}$$

One of the common choices is the trapezoidal quadrature. When the contour is a circle, whose center is $c$ and radius is $r$, the integral (2.1) numerically discretized with $k$ quadrature points is denoted as $R_{c,r,k}(B^{-1}A)$, whose poles and weights are

$$p_i^{(k)} = re^{\imath\theta_i^{(k)}} + c, \quad w_i^{(k)} = re^{\imath\theta_i^{(k)}}/k, \quad \theta_i^{(k)} = (2i-1)\pi/k. \tag{2.5}$$

The matrix function $R_{c,r,k}(B^{-1}A)$ is used as the filter in this paper. We will call it the $k$th-order trapezoidal quadrature.

**2.3. Cost of implementation.** In (2.4), the considerable computational burden lies in solving the shifted linear systems, $(p_i^{(k)} B - A)^{-1}$ for $i = 1, \ldots, k$. When the eigengap between the interior eigenvalues and outer eigenvalues is small, the linear systems will be ill-conditioned, since the poles are on the contour and their distances to eigenvalues are bounded by the eigengap. Hence, in most contour-based filters, the shifted linear systems are solved by direct methods.

The overall computational cost is then divided into two parts: the offline factorization part, e.g., LU factorization, and the online solving part, e.g., triangular solve with the given LU factorization. The computational cost could be written as

$$C_{\text{factor}} \times k + C_{\text{apply}} \times k \times n_{\text{col}} \times T + o(C_{\text{apply}}),$$

where $C_{\text{factor}}$ is the cost of a factorization, $C_{\text{apply}}$ is the cost of a solve, $k$ is the number of poles, $n_{\text{col}}$ is the number of columns in $Y$, $T$ is the number of subspace iterations, and $o(C_{\text{apply}}) = o(C_{\text{apply}}(N))$ is the remaining lower-order cost. To extract the entire eigensubspace we are interested in, it is necessary that $n_{\text{col}} \ge s$ for $s$ being the number of eigenvalues inside. As for the memory cost, the solving complexity is the same as its memory cost for almost all dense and sparse linear system solvers. Hence, $C_{\text{apply}}$ is also used as the memory cost in storing a factorization. The cost of memory is $kC_{\text{apply}}$.

Throughout the subspace iterations, the tunable hyperparameters are $k$ and $n_{\text{col}}$. The dependence of $T$ on $k$ and $n_{\text{col}}$ could be reflected by the function value $R_k(\lambda_i)$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

since we are essentially applying a power method with $R_k(B^{-1}A)$. Let $\sigma$ be a permutation of $1, 2, \ldots, N$, such that

$$|R_k(\lambda_{\sigma_1})| \geq |R_k(\lambda_{\sigma_2})| \geq \cdots \geq |R_k(\lambda_{\sigma_N})|.$$

Then, the number of subspace iterations $T$ mainly depends on the ratio,

$$\max_{i > n_{\mathrm{col}}} |R_k(\lambda_{\sigma_i})| \Big/ \min_{\lambda_{\sigma_i} \in \mathcal{D}} |R_k(\lambda_{\sigma_i})|. \tag{2.6}$$

When the ratio is greater than or equal to one, the subspace iteration would suffer from a divergence issue. When the ratio is less than one, the smaller the ratio, the faster the convergence. In general, larger $k$ and $n_{\mathrm{col}}$ lead to smaller (2.6). However, $k$ is limited by the bottleneck of memory, especially for the large-scale problems. Then the implementation based on the simple rule can only increase $n_{\mathrm{col}}$ to make the ratio (2.6) smaller than one. When many outer eigenvalues are very close to the contour and $k$ is small, an extremely large $n_{\mathrm{col}}$ is needed for the convergence.

We refer to the formula (2.4) of $R_{c,r,k}(B^{-1}A)$ as the simple rule, which differs from the composite rule introduced in the next section.

## 3. Two eigensolvers based on the composite rule.

The implementation based on the simple rule (2.4) needs $k$ matrix factorizations, which brings a huge burden on computation and memory for large matrices. In this section, we will start by introducing the composite rule of trapezoidal quadrature without derivation. Then we propose two eigensolvers based on the composite rule, both of which implement $R_{c,r,k}(B^{-1}A)$ with an inner-outer structure.

### 3.1. The composite rule.

Given a positive integer $k$ and its integer factorization $k = k_1 k_2$ for $k_1 > 1$ and $k_2 > 1$, the composite rule of trapezoidal quadrature is shown as

$$R_{c,r,k}(z) = R_{0,1,k_2}(M(R_{c,r,k_1}(z))),$$

where $M(z) = (1-z)/z$ is a Möbius transform. That means the $k$th-order trapezoidal quadrature can be rewritten as a composition of a $k_2$th-order trapezoidal quadrature and a transformed $k_1$th-order trapezoidal quadrature.

When $k_2$ is even, the composite rule can be rewritten as

$$R_{c,r,k}(z) = \sum_{i=1}^{k_2} c_i^{(k_2)} (R_{c,r,k_1}(z) - s_i^{(k_2)})^{-1} R_{c,r,k_1}(z), \tag{3.1}$$
$$c_i^{(k_2)} = -\frac{1}{k_2} \frac{\sigma_i^{(k_2)}}{1+\sigma_i^{(k_2)}},\ s_i^{(k_2)} = \frac{1}{1+\sigma_i^{(k_2)}},$$

where $\{\sigma_i^{(k_2)}\}_{i=1}^{k_2}$ are roots of $z^{k_2} = -1$.

### 3.2. Interior eigensolver with subspace iteration.

Using $R_{c,r,k}(z)$ as the filter in subspace iteration requires the evaluation of $R_{c,r,k}(B^{-1}A)Y$ for $Y$ to be a matrix of size $N \times n_{\mathrm{col}}$. By the composite rule for $R_{c,r,k}(z)$ in (3.1), the evaluation of $R_{c,r,k}(B^{-1}A)Y$ could be rewritten as

$$R_{c,r,k}(B^{-1}A)Y = \left( \sum_{i=1}^{k_2} c_i^{(k_2)} (R_{c,r,k_1}(B^{-1}A) - s_i^{(k_2)} I)^{-1} \right) \left( R_{c,r,k_1}(B^{-1}A)Y \right), \tag{3.2}$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

where the operation $R_{c,r,k_1}(B^{-1}A)Y$ can be written as

$$R_{c,r,k_1}(B^{-1}A)Y = \sum_{i=1}^{k_1} w_i^{(k_1)}(p_i^{(k_1)}B - A)^{-1}BY, \tag{3.3}$$

where $\{w_i^{(k_1)}\}$ and $\{p_i^{(k_1)}\}$ are the weights and poles of $R_{c,r,k_1}(\cdot)$.

In (3.2), there are inner and outer parts of the evaluation. For the inner part, as in (3.3), we use direct solvers for all these linear systems for the same reason as the simple rule. We prefactorize all linear systems and denote them as $K_i = p_i^{(k_1)}B - A$ for $i = 1, \dots, k_1$, e.g., $K_i = L_iU_i$.[3] Once the factorizations are available, the inner part can be addressed efficiently. The inner part (3.3) essentially applies a rational filter of the matrix pencil $(A, B)$ to a set of vectors $Y$. Without loss of generality, we treat the inner part as an operator $G$ acting on $Y$.

For the outer part, we first rewrite (3.2) using the operator $G$:

$$R_{c,r,k}(B^{-1}A)Y = \sum_{i=1}^{k_2} c_i^{(k_2)}(G - s_i^{(k_2)}I)^{-1}\widetilde{Y} \tag{3.4}$$

for $\widetilde{Y} = G(Y)$. We notice that the eigenvalues of $G$ are clustered, i.e., the eigenvalues outside $\mathcal{D}$ cluster around zero, and the eigenvalues inside $\mathcal{D}$ cluster around one; see Figure 2. Iterative solvers, especially GMRES, are expected to converge quickly. Throughout this paper, we adopt GMRES [14] as the default iterative solver for (3.4) with $G$ being applied as an operator. Recall that GMRES is a Krylov subspace method. By the shift-invariant property of the Krylov subspace, all $k_2$ shifts in (3.4) could be addressed simultaneously in the same Krylov subspace, i.e.,

$$\begin{aligned}\mathcal{K}_n(G - s_i^{(k_2)}I, y) &= \mathcal{K}_n(G, y),\\ (G - s_i^{(k_2)}I)V_n &= V_n(H_{n,n+1} - s_i^{(k_2)}I_{n,n+1}),\end{aligned}$$

for $i = 1, \dots, k_2$ and $V_n$ denoting the basis of $\mathcal{K}_n(G, y)$. The multishift GMRES [1] applies the operator $G$ once per iteration. In all of our numerical experiments, the multishift GMRES converges in less than one hundred iterations, and no restarting is needed.

Using a direct solver and an iterative solver for the inner and outer part of (3.2), we obtain an effective algorithm for the rational matrix function filter. Combining this filter with subspace iteration leads to our first eigensolver. Algorithm 3.1 gives the overall pseudocode, where HSRR is an abbreviation for lines 4–7 of Algorithm 2.1.

The convergence criterion for the interior eigenvalue problem can be quite tricky. The relative error of the approximate eigenpair in our algorithms is defined as

$$e_i = e(\tilde{\lambda}_i, \tilde{x}_i) = \frac{\|A\tilde{x}_i - B\tilde{x}_i\tilde{\lambda}_i\|_2}{(|c| + r)\|B\tilde{x}_i\|_2}, \tag{3.5}$$

where $c$ and $r$ are the center and radius of the region $\mathcal{D}$. For the non-Hermitian interior eigenvalue problem, a phenomenon called ghost eigenvalue often appears. The ghost

[3]Throughout the numerical section of this paper, dense LU factorization is used by default for dense matrices $A$ and $B$. If $A$ and $B$ are sparse matrices, we adopt the default sparse LU factorization methods in MATLAB.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

**Algorithm 3.1.** Eigensolver: Composite rational function filter.

```
Input: Pencil $(A,B)$, center $c$, radius $r$, number of eigenvalues $s$, shift $\sigma$, number of
    poles $[k_1,k_2]$, tolerance $\tau_g$ and $\tau$.
Output: The approximate eigenpair $(\tilde{\lambda}_i,\tilde{x}_i)$ with $\tilde{\lambda}_i \in \mathcal{D}$.
 1: Compute $\{p_i^{(k_1)}, w_i^{(k_1)}\}_{i=1}^{k_1}$, $\{c_j^{(k_2)}, s_j^{(k_2)}\}_{j=1}^{k_2}$.
 2: for $i=1,\cdots,k_1$ do
 3:     Prefactorize $p_i^{(k_1)}B-A$ as $K_i$ (e.g., $K_i=L_iU_i$).
 4: end for
 5: Construct a function for applying $G$ to a set of vectors $V$.
        $$G(V)=\sum_{i=1}^{k_1} w_i^{(k_1)} K_i^{-1}BV \text{ (e.g., } K_i^{-1}=U_i^{-1}L_i^{-1}).$$
 6: Generate an orthonormal random matrix $Y^{N\times n_{\mathrm{col}}}$ with $n_{\mathrm{col}} \ge s$.
 7: while $p$ changes or any $e_i$ of the filtered eigenpair is larger than $\tau$ do
 8:     $\widetilde{Y}=G(Y)$.
 9:     Solve $U_j=(G-s_j^{(k_2)}I)^{-1}\widetilde{Y}$ for $j=1,\ldots,k_2$ via multishift GMRES.
10:     $U=\sum_{j=1}^{k_2} c_j^{(k_2)}U_j$.
11:     $[Y,\widetilde{\Lambda},\widetilde{X}]=\mathrm{HSRR}(A,B,U,\sigma)$.
12:     Distinguish ghost eigenvalues and filtered eigenvalues by $\tau_g$. Update the
        number of filtered eigenvalues $p$.
13: end while
```

eigenvalue is the one that appears in the region $\mathcal{D}$ but does not converge. The ghost eigenvalue would make it difficult to examine the convergence of subspace iterations.

One of the practical strategies is to set a tolerance $\tau_g$ as in [19], which is much larger than the target relative error $\tau$. As the iteration goes, the true eigenvalues will converge to a small relative error, while the ghost eigenvalues will not converge to the same precision. After a few steps, there is a gap in the relative errors between the true eigenvalues and the ghost eigenvalues. When the relative error of an approximate eigenpair $(\tilde{\lambda}_i,\tilde{x}_i)$ inside $\mathcal{D}$ is smaller than $\tau_g$, we treat it as a filtered eigenpair and denote the number of filtered eigenpairs as $p$. When $p$ is not changed and all relative errors of the filtered eigenpairs are smaller than $\tau$, we terminate the algorithm. That corresponds to lines 7 and 12 of Algorithm 3.1.

We now estimate the computational cost for Algorithm 3.1. In the preparation phase before subspace iteration, the weights and poles are computed in the computational cost of $O(1)$. For the prefactorizations of $k_1$ linear systems, the computational complexity is $k_1C_{\mathrm{factor}}$ and the memory required is $k_1C_{\mathrm{apply}}$. In the subspace iteration phase, the per-iteration computational cost is dominated by the multishift GMRES. If we denote $n_{\mathrm{iter}}^{(j,t)}$ as the GMRES iteration number for the $j$th column in the $t$th subspace iteration, the dominant computational cost in the GMRES is

$$\sum_{t=1}^{T}\sum_{j=1}^{n_{\mathrm{col}}} n_{\mathrm{iter}}^{(j,t)}\cdot k_1C_{\mathrm{apply}},$$

where $k_1C_{\mathrm{apply}}$ is the cost in applying $G(\cdot)$ to a vector.

The overall dominant computational and memory costs for Algorithm 3.1 are summarized in Table 1. In the same table, we also list the computational and memory costs for subspace iteration with $k_1k_2$th-order rational filter without using the com-

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

TABLE 1

*Computational and memory complexities of the subspace iteration with the simple rational filter and the composite rational filter. The simple rational filter is of order $k_1k_2$, and the composite rational filter is of inner and outer order $k_1$ and $k_2$, respectively. Here $C_{\text{factor}}$ and $C_{\text{apply}}$ are factorization and solving cost for a matrix of size $N \times N$.*

| Algorithm | Computation | | Memory | |
|---|---|---|---|---|
| | Prefact | Iteration | Prefact | Iteration |
| Simple | $k_1k_2C_{\text{factor}}$ | $Tn_{\text{col}}k_1k_2C_{\text{apply}}$ | $k_1k_2C_{\text{apply}}$ | $n_{\text{col}}N$ |
| Algorithm 3.1 | $k_1C_{\text{factor}}$ | $\sum_{t=1}^{T}\sum_{j=1}^{n_{\text{col}}} n_{\text{iter}}^{(j,t)}k_1C_{\text{apply}}$ | $k_1C_{\text{apply}}$ | $\max_{t=1}^{T}\sum_{j=1}^{n_{\text{col}}} n_{\text{iter}}^{(j,t)}N$ |
| Ratio | $k_2$ | $\frac{Tn_{\text{col}}k_2}{\sum_{t=1}^{T}\sum_{j=1}^{n_{\text{col}}} n_{\text{iter}}^{(j,t)}}$ | $k_2$ | $\frac{n_{\text{col}}}{\max_{t=1}^{T}\sum_{j=1}^{n_{\text{col}}} n_{\text{iter}}^{(j,t)}}$ |

posite rule. Another row of ratio is added to indicate the acceleration from Algorithm 3.1. Clearly, both the computation and memory costs in the prefactorization phase are reduced by a factor of $k_2$, while the comparison for the subspace iteration part is less clear. The ratio depends on the iteration numbers of both the subspace iteration and the multishift GMRES. Another interesting thing is that as the subspace iteration progresses, the columns of $Y$ become closer aligned with the eigenvectors, which means the Krylov subspaces will converge faster and so will the GMRES, as we numerically show in Appendix A.

**3.3. Composite rule eigensolver without subspace iteration.** Algorithm 3.1 still adopts the framework of subspace iteration. It implements the same order trapezoidal quadrature as the simple rule in a different way. We can take advantage of the composite rule from another point of view, i.e., achieving better approximation with the same number of factorizations as the simple rule. The better the rational approximation, the fewer subspace iterations are needed. In the limit of very accurate approximation, only one subspace iteration is enough. Making use of the shift-invariant property of Krylov subspace, we proposed Algorithm 3.2, which achieves better approximation with a fixed number of factorizations and discards the framework of subspace iteration.

More specifically, the first step of Algorithm 3.2 for the initial $[k_1, k_2]$ is the same as in Algorithm 3.1, which also constructs the operator $G$ via prefactorizations and generates Krylov subspaces of different vectors for the computation of $R_{c,r,k}(Y)$. When the approximate eigenpairs do not converge, Algorithm 3.2 will double $k_2$ and compute new shifts and weights, $s_j^{(2k_2)}$ and $c_j^{(2k_2)}$. Importantly, Algorithm 3.2 does not regenerate new Krylov subspaces from the approximate eigenvectors. Instead, it computes $R_{c,r,2k}(Y)$ in the existing Krylov subspaces used for $R_{c,r,k}(Y)$ and expands them when it is necessary. Algorithm 3.2 keeps enlarging $k_2$ until all the approximate eigenpairs converge. As we will show in section 6, the dimension of Krylov subspace is not sensitive to $k_2$ and increases mildly. In addition, we find that the shifts $s_j^{(k_2)}$ are parts of the shifts $s_j^{(2k_2)}$ and their weights satisfy $c_j^{(k_2)}/2 = c_j^{(2k_2)}$. This means we only need to compute $U_j = (G - s_j^{(2k_2)}I)^{-1}\widetilde{Y}$ for the new shifts, then $R_{c,r,2k}(Y)$ can be computed from $R_{c,r,2k}(Y) = \frac{1}{2}R_{c,r,k}(Y) + \sum_{j=k_2+1}^{2k_2} c_j^{(2k_2)}U_j$. It corresponds to lines 9 and 10 of Algorithm 3.2.

Compared to Algorithm 3.1, Algorithm 3.2 does not regenerate Krylov subspaces each time and enables adaptive selection of $k_2$, which makes Algorithm 3.2 more practical. The more interesting characteristic of Algorithm 3.2 is that it discards the framework of subspace iteration. We find that the idea of reusing the Krylov

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

**Algorithm 3.2.** Eigensolver: Composite rational function filter without subspace iteration.

```
Input: Pencil $(A,B)$, center $c$, radius $r$, number of eigenvalues $s$, shift $\sigma$, tolerance
    $\tau_g$ and $\tau$, number of poles $k_1$, initial $k_2$ ($k_2 = k_1$ in default), and $\hat{k}_2 = 0$.
Output: The approximate eigenpair $(\widetilde{\lambda}_i, \tilde{x}_i)$ with $\widetilde{\lambda}_i \in \mathcal{D}$.
 1: Compute $\{p_i^{(k_1)}, w_i^{(k_1)}\}_{i=1}^{k_1}$, $\{c_j^{(k_2)}, s_j^{(k_2)}\}_{j=1}^{k_2}$.
 2: for $i = 1, \ldots, k_1$ do
 3:     Prefactorize $p_i^{(k_1)} B - A$ as $K_i$ (e.g., $K_i = L_i U_i$).
 4: end for
 5: Construct a function for applying $G$ to a set of vectors $V$.
        $$G(V) = \sum_{i=1}^{k_1} w_i^{(k_1)} K_i^{-1} B V \text{ (e.g., } K_i^{-1} = U_i^{-1} L_i^{-1}).$$
 6: Generate an orthonormal random matrix $Y^{N \times n_{\text{col}}}$ with $n_{\text{col}} \geq s$.
 7: $\widetilde{Y} = G(Y)$, $U$ is a zero matrix of the same size.
 8: while $p$ changes or any $e_i$ of the filtered eigenpair is larger than $\tau$ do
 9:     Solve $U_j = (G - s_j^{(k_2)} I)^{-1} \widetilde{Y}$ for $j = \hat{k}_2 + 1, \ldots, k_2$ via multishift GMRES in the
        existing Krylov subspaces and expand it when necessary.
10:     $U = U/2 + \sum_{j=\hat{k}_2+1}^{k_2} c_j^{(k_2)} U_j$.
11:     $[Y, \widetilde{\Lambda}, \widetilde{X}] = \text{HSRR}(A, B, U, \sigma)$.
12:     Distinguish ghost eigenvalues and filtered eigenvalues by $\tau_g$. Update the
        number of filtered eigenvalues $p$.
13:     $\hat{k}_2 = k_2, k_2 = 2k_2$. Compute $\{c_j^{(k_2)}, s_j^{(k_2)}\}_{j=\hat{k}_2+1}^{k_2}$.
14: end while
```

subspace for algorithm design is also shown in [3], where the authors use a single Cayley transform for preconditioning. Instead, we use trapezoidal quadrature with $k_1$ poles for preconditioning. This means Algorithm 3.2 can enjoy the benefit of parallelization, as the solving of the $k_1$ shifted linear systems $(p_i^{(k_1)} B - A)x = y$ can be performed simultaneously.

Another feature of Algorithm 3.2 is that in the whole process of dynamically increasing $k_2$, we are essentially using the trapezoidal quadrature as a filter, while Algorithm 3.1 and the simple rule both use powers of the trapezoidal quadrature. As we show in the next section, trapezoidal quadrature is asymptotically optimal, which contributes part of the advantages of Algorithm 3.2.

## 4. Asymptotically optimal contour discretization.

In this section, we will start with the definition of spectrum separation, which is a continuous version of (2.6). Then we will study the optimal and asymptotically optimal rational filter in the sense of spectrum separation. Two main results will be proved, i.e., the rational function used in the inverse power method is optimal and the trapezoidal quadrature is asymptotically optimal with respect to $k$. The latter one shows that $R_{c,r,k}(z)$ is a reasonable good choice.

### 4.1. Spectrum separation.

Among various quadrature rules, the optimality of quadrature needs to be defined based on a criterion. The convergence rate mainly depends on (2.6). Since we do not know eigenvalues a priori, we could assume that

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

there is an annulus around the boundary of $\mathcal{D}$ as a generalized eigengap of the inner and outer eigenvalues. The inner part and the outer part are

$$\mathcal{I} = \{z : |z| \le a\} \quad \text{and} \quad \mathcal{O} = \{z : |z| \ge b\},$$

where $a$ and $b$ are the radii of the inner and outer parts of the annulus, and $\mathcal{I}$ contains all the eigenvalues inside $\mathcal{D}$. Then the criterion is defined as

$$\mathfrak{R} = \frac{\sup_{z\in\mathcal{O}} |R_k(z)|}{\inf_{z\in\mathcal{I}} |R_k(z)|}. \tag{4.1}$$

Hence, we would like to address the following optimization problem to obtain the optimal weights and poles for a given $k$:

$$\inf_{\{w_i^{(k)}\}_{i=1}^k, \{p_i^{(k)}\}_{i=1}^k} \frac{\sup_{z\in\mathcal{O}} |R_k(z)|}{\inf_{z\in\mathcal{I}} |R_k(z)|}. \tag{4.2}$$

We call (4.1) spectrum separation. One can see that as $b/a$ becomes larger, it is easier to separate the values inside and outside the annulus with rational functions. The drawback of using a larger $b$ is that more eigenvalues may fall into the annulus $\{z : a \le |z| \le b\}$ and we do not explicitly know the impact of these eigenvalues on the convergence of the subspace iteration.

**4.2. Zolotarev's third problem.** We introduce Zolotarev's third problem with its related theoretical results [12, 16]. Zolotarev's third problem is about the optimal separation of rational functions on two disjoint regions. Since contour discretization yields a rational function, it is natural to bridge the contour discretization and Zolotarev's third problem.

DEFINITION 4.1. *Let $\mathcal{E}$ and $\mathcal{G}$ be two disjoint regions of $\mathbb{C}$, i.e., $\mathcal{E} \cap \mathcal{G} = \emptyset$. Zolotarev's third problem is to solve the following optimal problem:*

$$Z_k(\mathcal{E},\mathcal{G}) = \inf_{r\in\mathcal{R}_{k,k}} \frac{\sup_{z\in\mathcal{E}} |r(z)|}{\inf_{z\in\mathcal{G}} |r(z)|}. \tag{4.3}$$

Zolotarev's third problem tends to find a rational function whose value on $\mathcal{E}$ is closest to zero and whose value on $\mathcal{G}$ is farthest from zero. When $\mathcal{E}$ and $\mathcal{G}$ are two symmetric disks with respect to the origin as in Figure 1, the solution to Zolotarev's third problem is explicitly given in Theorem 4.2. Theorem 4.2 as shown in this paper takes a different parameterized form from that in [16].

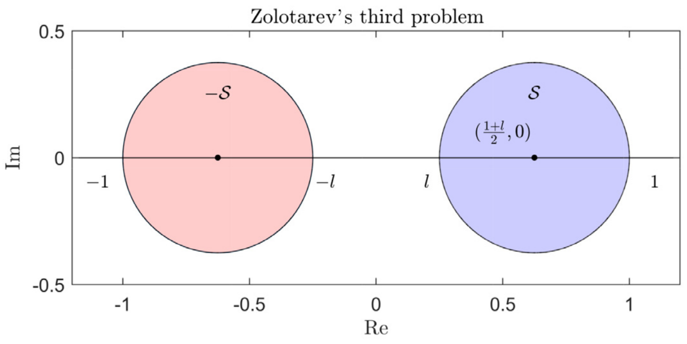

FIG. 1. *Regions in Zolotarev's third problem when $E$ and $G$ are symmetric disks.*

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

THEOREM 4.2. *Let $\mathcal{S} = \{z \in \mathbb{C} : |z - \frac{1+\ell}{2}| \le \frac{1-\ell}{2}\}, 0 < \ell < 1$. Then the rational function*

$$R_k^{(Z)}(z) = \left(\frac{z-\sqrt{\ell}}{z+\sqrt{\ell}}\right)^k$$

*attains the infimum of Zolotarev's third problem $Z_k(\mathcal{S}, -\mathcal{S})$ and the infimum is equal to $(\frac{1+\sqrt{\ell}}{1-\sqrt{\ell}})^{-2k}$.*

The explicit solution to Zolotarev's third problem as in Theorem 4.2 is the key to proving the asymptotical optimality in the sense of spectrum separation with respect to $k$ of the trapezoidal quadrature. The rational function in Theorem 4.2 is referred to as Zolotarev's third function in the rest of this paper.

**4.3. Compact form for $R_{0,1,k}(z)$.** In order to connect Zolotarev's function and the trapezoidal quadrature of the contour integral and derive the composite formula in section 5, we establish an identity that relates $R_{0,1,k}(z)$ and $R_{0,1,1}(z^k)$. The relation heavily relies on the symmetry of the trapezoidal quadrature on the circle.

Let us start with toy cases $k = 2, 4$. The trapezoidal quadrature of the unit circular contour with two poles could be rewritten as

$$R_{0,1,2}(z) = \frac{1}{2}\left(\frac{e^{\frac{\imath\pi}{2}}}{e^{\frac{\imath\pi}{2}} - z} + \frac{e^{\frac{3\imath\pi}{2}}}{e^{\frac{3\imath\pi}{2}} - z}\right) = \frac{1}{2}\frac{2e^{\imath\pi}}{e^{\imath\pi} - z^2} = \frac{1}{1+z^2} = R_{0,1,1}(z^2).$$

Here we use the symmetry of poles and weights with respect to the origin to derive the compact form of $R_{0,1,2}(z)$ and find that $R_{0,1,2}(z)$ is equivalent to $R_{0,1,1}(z^2)$. Let us further derive the compact form of $R_{0,1,4}(z)$,

$$\begin{aligned} R_{0,1,4}(z) &= \frac{1}{4}\left(\frac{e^{\frac{\imath\pi}{4}}}{e^{\frac{\imath\pi}{4}} - z} + \frac{e^{\frac{7\imath\pi}{4}}}{e^{\frac{7\imath\pi}{4}} - z} + \frac{e^{\frac{3\imath\pi}{4}}}{e^{\frac{3\imath\pi}{4}} - z} + \frac{e^{\frac{5\imath\pi}{4}}}{e^{\frac{5\imath\pi}{4}} - z}\right) \\ &= \frac{1}{2}\left(\frac{e^{\frac{\imath\pi}{2}}}{e^{\frac{\imath\pi}{2}} - z^2} + \frac{e^{\frac{3\imath\pi}{2}}}{e^{\frac{3\imath\pi}{2}} - z^2}\right) = R_{0,1,2}(z^2) = R_{0,1,1}(z^4), \end{aligned}$$

where, in the second equality, we combine the first two and last two terms, and in the last equality, we adopt the compact form of $R_{0,1,2}(z)$. From the derivation of the compact forms of $R_{0,1,2}(z)$ and $R_{0,1,4}(z)$, we could directly extend the derivation to obtain the compact form of $R_{0,1,k}(z) = R_{0,1,1}(z^k)$ for $k = 2^m$, $m \in \mathbb{N}_+$. Fortunately, the compact form holds for any $k \in \mathbb{N}_+$. The result is summarized in Lemma 4.3.

LEMMA 4.3. *For all $k \in \mathbb{N}_+$, let $k$ roots of $z^k = -1$ be $\sigma_i^{(k)}$ for $i = 1, \ldots, k$. Then the compact form of $R_{0,1,k}(z)$ admits*

$$R_{0,1,k}(z) = \frac{1}{k}\sum_{i=1}^{k} \frac{\sigma_i^{(k)}}{\sigma_i^{(k)} - z} = \frac{1}{1+z^k} = R_{0,1,1}(z^k). \tag{4.4}$$

*Proof.* We first prove two equalities, (4.5) and (4.6), and then derive the compact form of $R_{0,1,k}(z)$.

The $k$ roots of the $k$th degree polynomial $z^k + 1$ are denoted as $\sigma_i^{(k)}$ for $i = 1, 2, \ldots, k$. A $k$th- order polynomial with $k$ roots takes the form $a_k \prod_{i=1}^{k}(z - \sigma_i^{(k)})$, where $a_k$ is the coefficient in the leading order. Comparing with the leading-order coefficient in $z^k + 1$, we know $a_k = 1$ and have

$$z^k + 1 = \prod_{i=1}^{k}(z - \sigma_i^{(k)}). \tag{4.5}$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Then we prove the second equality,

$$-\frac{1}{k}\sum_{i=1}^{k}\sigma_i^{(k)}\prod_{j=1,j\neq i}^{k}(z-\sigma_j^{(k)})=1. \tag{4.6}$$

The left-hand side of (4.6) is a $(k-1)$th degree polynomial. For equality (4.6) to hold, we only need to make sure that the equality holds on $k$ different points. Specifically, we examine this on $\sigma_i^{(k)}$ for $i=1,\ldots,k$ and obtain

$$-\frac{\sigma_i^{(k)}}{k}\prod_{j=1,j\neq i}^{k}(\sigma_i^{(k)}-\sigma_j^{(k)})=-\frac{\sigma_i^{(k)}}{k}\lim_{z\to\sigma_i^{(k)}}\frac{z^k+1}{z-\sigma_i^{(k)}}=-\frac{\sigma_i^{(k)}}{k}\frac{k(\sigma_i^{(k)})^{k-1}}{1}=1,$$

where the first equality is due to (4.5) and the continuity of $(z^k+1)/(z-\sigma_i^{(k)})$, the second equality comes from the L'Hôpital rule of complex functions, and the last equality holds since $\sigma_i^{(k)}$ is a root of $z^k+1$.

Finally, we derive the compact form of $R_k(z)$ as in Lemma 4.3:

$$\begin{aligned}R_{0,1,k}(z)&=\frac{1}{k}\sum_{i=1}^{k}\frac{\sigma_i^{(k)}}{\sigma_i^{(k)}-z}=\frac{-\frac{1}{k}\sum_{i=1}^{k}\sigma_i^{(k)}\prod_{j=1,j\neq i}^{k}(z-\sigma_j^{(k)})}{\prod_{i=1}^{k}(z-\sigma_i^{(k)})}\\&=\frac{1}{\prod_{i=1}^{k}(z-\sigma_i^{(k)})}=\frac{1}{z^k+1}=R_{0,1,1}(z^k),\end{aligned}$$

where the second equality adopts (4.6) and the fourth equality adopts (4.5). ∎

A related compact form without detailed derivation can be found in [5]. The compact form Lemma 4.3 can be further generalized to $R_{c,r,k}(z)$ and results in the compact form

$$R_{c,r,k}(z)=\frac{1}{1+(\frac{z-c}{r})^k}.$$

## 4.4. Optimal solution and the asymptotic optimality of trapezoidal quadrature.

In this section, we prove that if we know the desired spectrum explicitly, the rational function used in the inverse power method achieves the infimum of (4.3) for $\mathcal{E}=\mathcal{O}$ and $\mathcal{G}=\mathcal{I}$. On the other hand, the rational function $R_{c,r,k}(z)$ from the trapezoidal quadrature discretization of the contour integral achieves asymptotic optimality of (4.3).

THEOREM 4.4. *The rational function $z^{-k}$ achieves the infimum of* (4.3) *for $\mathcal{E}=\mathcal{O}$ and $\mathcal{G}=\mathcal{I}$. And the infimum equals $(\frac{a}{b})^k$.*

*Proof.* We address Zolotarev's third problem with region $\mathcal{I}$ and $\mathcal{O}$, i.e., $Z_k(\mathcal{O},\mathcal{I})$. Define a Möbius transform $M(z)=\gamma\frac{z-\alpha}{z-\beta}$ such that

$$M(-b)=1,\quad M(-a)=-1,\quad M(a)=-\ell,\quad M(b)=\ell.$$

The parameters $\gamma$, $\alpha$, $\beta$, and $\ell$ are determined by $a$ and $b$. They satisfy

$$\alpha=\sqrt{ab},\quad \beta=-\sqrt{ab},\quad \gamma=\frac{\sqrt{b}-\sqrt{a}}{\sqrt{b}+\sqrt{a}},\quad \ell=\left(\frac{\sqrt{b}-\sqrt{a}}{\sqrt{b}+\sqrt{a}}\right)^2.$$

It can be verified that $M(\mathcal{I})=-\mathcal{S}$ and $M(\mathcal{O})=\mathcal{S}$ for $\mathcal{S}$ in Theorem 4.2. Then the composition of the Möbius transform and Zolotarev's function $R_k^{(Z)}(M(z))$ achieves the infimum of $Z_k(\mathcal{O},\mathcal{I})$ and is denoted as

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

$$R_k^{(A)}(z) = R_k^{(Z)}(M(z)) = z^{-k}. \tag{4.7}$$

The infimum of $\mathcal{I}$ is taken when $|z| = a$, and the supremum of $\mathcal{O}$ is taken when $|z| = b$. Then the infimum of the ratio is $(\frac{a}{b})^k$. ☐

Theorem 4.4 gives the optimal rational function in solving (4.3). The rational function $z^{-k}$ therein combined with subspace iteration corresponds to the well-known inverse power method. Further, the radius of $\mathcal{D}$ or the diameter of the annulus is not included in the optimal rational function. Hence, we conclude that, in the sense of convergence rate of subspace iteration, the optimal interior eigensolver is the inverse power method, assuming the center of the desired region $\mathcal{D}$ is explicitly known.

The optimal rational function $z^{-k}$, on the other hand, only has one pole and cannot be written as a sum of low-order rational functions in the form of (2.3). The inverse power method then has to be executed sequentially and cannot benefit from the parallelization of distinct poles. In the following, we argue that, although the trapezoidal quadrature of the contour integral is not the optimal rational function, it achieves asymptotic optimality.

We consider that the contour is the boundary of $\mathcal{I}$. By Lemma 4.3, the discretization can be rewritten as

$$R_{0,a,k}(z) = \frac{1}{1+(\frac{z}{a})^k}.$$

By the maximum modulus principle, the infimum of $\mathcal{I}$ and the supremum of $\mathcal{O}$ are taken when $|z| = a$ and $|z| = b$. In region $\mathcal{I}$, $|\frac{z}{a}|^k \leq 1$. The absolute value of the denominator can be viewed as the distance between $-1$ and $(\frac{z}{a})^k$. By simple computation, the infimum is achieved when $z = a$. Similarly, the supremum of $\mathcal{O}$ is achieved when $z = \sqrt[k]{-1}b$ from the fact that $|\frac{z}{a}|^k > 1$ in $\mathcal{O}$. The spectrum separation (4.1) is

$$\mathfrak{R} = \frac{2}{(\frac{b}{a})^k - 1} \sim 2\left(\frac{a}{b}\right)^k,$$

which asymptotically decays with respect to $k$ at the same rate as that in Theorem 4.4. The above discussion is summarized in the following corollary.

COROLLARY 4.5. *The trapezoidal quadrature discretization of the contour integral on the boundary of $\mathcal{G} = \mathcal{I}$ results in the rational function*

$$1\bigg/\left[1+\left(\frac{z}{a}\right)^k\right]. \tag{4.8}$$

*The spectrum separation* (4.1) *of the trapezoidal quadrature is* $2/[(\frac{b}{a})^k - 1]$, *which achieves the same decay rate as the infimum of* (4.3) *for $\mathcal{E} = \mathcal{O}$ and $\mathcal{G} = \mathcal{I}$.*

Although the trapezoidal quadrature used to approximate the contour integral is not the optimal rational function of (4.3), the ratio asymptotically achieves the optimal one up to a constant prefactor 2. Hence, we call $R_{c,r,k}(z)$ the nearly optimal rational function for (4.3). The advantage of the trapezoidal quadrature over $z^{-k}$ is that $R_{c,r,k}(z)$ can be efficiently parallelized. Specifically, the solving of the $k$ shifted linear systems $(p_i^{(k)}B - A)x = y$ can be performed simultaneously, while the solving of $z^{-k}$ has to be performed subsequently.

## 5. Composite rule of trapezoidal quadrature.

In this section, a composite rule of trapezoidal quadrature is derived. Let us start from $c = 0$ and $r = 1$. We repurpose $R_k(z)$ as $R_{0,1,k}(z)$ in the rest of this paper.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Given a positive integer $k$ and its integer factorization $k = k_1k_2$ for $k_1 > 1$ and $k_2 > 1$, we aim to rewrite the $k$th-order rational function $R_k(z)$ as a composition of two $k_1$th and $k_2$th rational functions, $R_{k_1}(z)$ and $\hat{R}_{k_2}(z) = R_{k_2}(M(z))$, where $M(\cdot)$ is a Möbius transform function. Precisely, the composite function admits $R_k(z) = \hat{R}_{k_2}(R_{k_1}(z)) = R_{k_2}(M(R_{k_1}(z))$.

According to Lemma 4.3, we have a natural composite expression as

$$R_{k_1k_2}(z) = R_1(z^{k_1k_2}) = R_{k_2}(z^{k_1}).$$

For the desired composite rule to hold, we should let $M(R_{k_1}(z)) = z^{k_1}$. Now we determine the coefficients of $M(z) = (az - b)/(cz - d)$ such that $M(R_{k_1}(z)) = z^{k_1}$ holds. For $|z| \neq \infty$, substituting $R_{k_1}(z) = 1/(1 + z^{k_1})$ into the expression of $M(z)$, we obtain

$$\begin{aligned} M(R_{k_1}(z)) &= \frac{a - b(1 + z^{k_1})}{c - d(1 + z^{k_1})} = z^{k_1} \\ \iff \quad & dz^{2k_1} + (d - c - b)z^{k_1} + (a - b) = 0. \end{aligned}$$

The above equality holds for all $z$. Hence we have solutions of coefficients satisfying $d = 0$ and $a = b = -c$. These solutions of coefficients lead to the unique Möbius transform function

$$M(z) = \frac{1 - z}{z}. \tag{5.1}$$

One can verify that $M(R_{k_1}(z)) = z^{k_1}$ holds for $|z| = \infty$. In Figure 2, the mapping of $R_{k_1}(z)$ and $M(R_{k_1}(z))$ are illustrated.

Throughout the above derivation, we conclude that $R_{k_1k_2}(z) = R_{k_2}(M(R_{k_1}(z)))$. A generalized composite rule is given in Theorem 5.1 for $\Gamma$ with center $c$ and radius $r$. In Theorem 5.1, we compose $R_{0,1,k_2}(\cdot)$ and $M(\cdot)$ together and rewrite it as the sum of first-order rational functions. Such a summation form could later be used directly in the algorithm design.

THEOREM 5.1. *Given a positive integer $k$ and its integer factorization $k = k_1k_2$, the rational function $R_{c,r,k}(z)$ admits the following composite rule:*

$$R_{c,r,k}(z) = R_{0,1,k_2}(M(R_{c,r,k_1}(z))),$$

*where $M(\cdot)$ is the Möbius transform* (5.1). *When $k_2$ is even, the rational function $R_{c,r,k}(z)$ further admits the summation form*

$$\begin{aligned} R_{c,r,k}(z) &= \sum_{i=1}^{k_2} c_i^{(k_2)} \left(R_{c,r,k_1}(z) - s_i^{(k_2)}\right)^{-1} R_{c,r,k_1}(z), \\ c_i^{(k_2)} &= -\frac{1}{k_2} \frac{\sigma_i^{(k_2)}}{1 + \sigma_i^{(k_2)}}, \; s_i^{(k_2)} = \frac{1}{1 + \sigma_i^{(k_2)}}, \end{aligned} \tag{5.2}$$

*where $\{\sigma_i^{(k_2)}\}_{i=1}^{k_2}$ are roots of $x^{k_2} = -1$. When $k_2$ is odd,*

$$R_{c,r,k}(z) = \sum_{i=1}^{k_2-1} c_i^{(k_2)} \left(R_{c,r,k_1}(z) - s_i^{(k_2)}\right)^{-1} R_{c,r,k_1}(z) + \frac{1}{k_2} R_{c,r,k_1}(z), \tag{5.3}$$

*where $\sigma_{k_2}^{(k_2)} = -1$.*

*Proof.* We can use the equation $z = ry + c$ to transfer the contour discretization on an arbitrary circle into the case of the unit circle around the origin. The rational function then admits

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

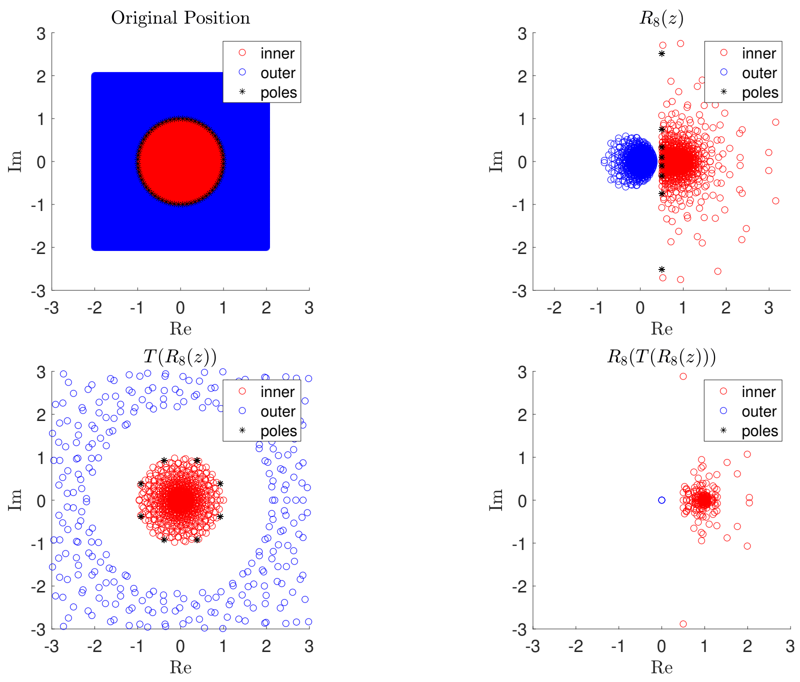

FIG. 2. *We plot the mapping on* $[-2,2]+[-2,2]*\imath$. *There are* 201 *equally spaced points in the direction of the real part and the imaginary part,* 40401 *points in total. The outer points are those* $|z| > h = 1.1$, *and the inner points are those* $|z| \leq 1$ *where the contour is* $|z| = 1$. *We fix the figure window at* $[-3,3]+[-3,3]*\imath$ *except for the top right figure which is shown at* $[-2.5,3.5]+[-3,3]*\imath$. *We let* $k_1 = k_2 = 8$, *and the poles in all figures are the poles of* $R_{k_1k_2}(z)$. *The original eigengap is almost invisible; see the top left figure. From the top right figure,* $R_8(z)$ *maps the inner part to be close to* 1 *and the outer part to be close to* 0, *and the poles are mapped to the line* $\mathrm{Real}(z) = 0.5$. *A clearer comparison of pre- and postmapping eigengaps is shown as the difference between the top left figure and bottom left figure. The composite mapping successfully maps the outer part close to* 0 *and the inner part close to* 1 *or modulus greater than* 1; *see bottom right figure.*

$$R_{c,r,k}(z) = R_{0,1,k}(y). \tag{5.4}$$

Combining this with $R_{0,1,k_1k_2}(z) = R_{0,1,k_2}(M(R_{c,r,k_1}(z)))$, we have

$$R_{c,r,k}(z) = R_{0,1,k_1k_2}(y) = R_{0,1,k_2}(M(R_{0,1,k_1}(y))) = R_{0,1,k_2}(M(R_{c,r,k_1}(z))). \tag{5.5}$$

Now we turn to prove the summation form. When $k_2$ is even, $\sigma_i^{(k_2)} \neq -1$ holds. With Lemma 4.3, the summation form is

$$\begin{aligned} R_{c,r,k_1k_2}(z) = R_{0,1,k_2}(M(R_{c,r,k_1}(y))) &= \frac{1}{k_2}\sum_{i=1}^{k_2} \frac{\sigma_i^{(k_2)}}{\sigma_i^{(k_2)} - \frac{1-R_{c,r,k_1}(y)}{R_{c,r,k_1}(y)}} \\ &= \frac{1}{k_2}\sum_{i=1}^{k_2} \frac{\sigma_i^{(k_2)} R_{c,r,k_1}(y)}{(1+\sigma_i^{(k_2)})R_{c,r,k_1}(y) - 1} \\ &= \frac{1}{k_2}\sum_{i=1}^{k_2} \frac{\sigma_i^{(k_2)}}{1+\sigma_i^{(k_2)}} \left( R_{c,r,k_1}(z) - \frac{1}{1+\sigma_i^{(k_2)}} \right)^{-1} R_{c,r,k_1}(x) \\ &= \sum_{i=1}^{k_2} c_i^{(k_2)} (s_i^{(k_2)} - R_{c,r,k_1}(z))^{-1} R_{c,r,k_1}(z), \end{aligned} \tag{5.6}$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

where

$$c_i^{(k_2)} = -\frac{1}{k_2}\frac{\sigma_i^{(k_2)}}{1+\sigma_i^{(k_2)}}, \quad s_i^{(k_2)} = \frac{1}{1+\sigma_i^{(k_2)}}. \tag{5.7}$$

When $k_2$ is odd, the term associated with $\sigma_{k_2}^{(k_2)} = -1$ in summation form is equal to $\frac{1}{k_1}R_{c,r,k_1}(z)$. ∎

The poles of the rational function $R_{c,r,k}(z)$ are transferred into the poles of $R_{0,1,k_2}(M(z))$ by the inner operator $R_{c,r,k_1}(z)$. This is detailed in Proposition 5.2.

PROPOSITION 5.2. *For any $p_i^{(k)}$ being a pole of $R_{c,r,k}(z)$, there exist $s_j^{(k_2)}$ for $1 \le j \le k_2$, such that*

$$R_{c,r,k_1}(p_i^{(k)}) = s_j^{(k_2)}, \tag{5.8}$$

*where $s_{k_2}^{(k_2)}$ could be infinite when $k_2$ is odd.*

*Proof.* By Lemma 4.3, we know

$$R_{c,r,k_1}(p_i^{(k)}) = R_{0,1,k_1}(\sigma_i^{(k)}) = \frac{1}{1+(\sigma_i^{(k)})^{k_1}} = \frac{1}{1+\sigma_j^{(k_2)}} = s_j^{(k_2)}. \tag{5.9}$$ ∎

## 6. Numerical experiment.

In this section, we will demonstrate the efficiency and stability of the two algorithms through three experiments. The first experiment shows the advantage of the trapezoidal quadrature over another quadrature, Gauss quadrature. The latter two experiments show the computational benefit of applying Algorithm 3.1 and Algorithm 3.2. This paper focuses on filter design rather than proposing a novel projection technique. Hence the projection techniques used in Algorithm 3.1, Algorithm 3.2, and the simple rule remain identical. Since the estimation of the number of eigenvalues is beyond the scope of this paper, we assume $s$ is known and set the number of columns $n_{\mathrm{col}} > s$ in all numerical experiments.

In our experiments, we set the parameters for the convergence criterion that have been discussed in section 3.2 to $\tau_g = 10^{-2}$ and $\tau = 10^{-8}$. The former one serves as the tolerance for distinguishing the ghost eigenvalues, while the latter one is the target precision of eigenpairs. The $\sigma$ in HSRR is set as $c$, which is the center of the circular contour $\Gamma$.

The direct solver is the `lu` function in MATLAB with four outputs under the default setting, which leads to a sparse LU factorization for sparse matrices. The triangular solves are performed by "\" in MATLAB, which can handle multiple right-hand sides simultaneously. All programs are implemented and executed with MATLAB R2022b and are performed on a server with Intel Xeon Gold 6226R CPU at 2.90 GHz and 1 TB memory. In performance experiments, we report the single-thread wall time.

### 6.1. Asymptotically optimal rational filter.

First, we show the spectrum separation (4.1) for the trapezoidal quadrature, Gauss quadrature, and the optimal one in Theorem 4.4. The numerical results are illustrated in Figure 3. Here, we set $a = 1$ and $b = 1.1$. The infimum of $\mathcal{I}$ and supremum of $\mathcal{O}$ for the Gauss quadrature are not known as a closed form, so we use the discretization of 1000 points in both directions of real and imaginary part on $[-1.5, 1.5] + [-1.5, 1.5] \cdot \imath$ to estimate (4.1) of the Gauss quadrature. Only even $k$ is adopted as we perform the Gauss quadrature on the upper semicircle and lower semicircle separately. Such a Gauss quadrature

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

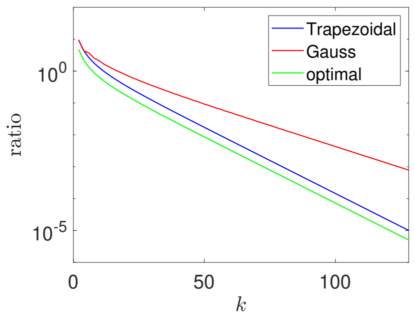

FIG. 3. *The separation ratio* (4.1) *for various quadrature rules with increasing numbers of poles. The number of poles $k$ ranges from 2 to 128. The trapezoidal quadrature shows the same slope as the optimal ratio, while the Gauss quadrature behaves differently.*

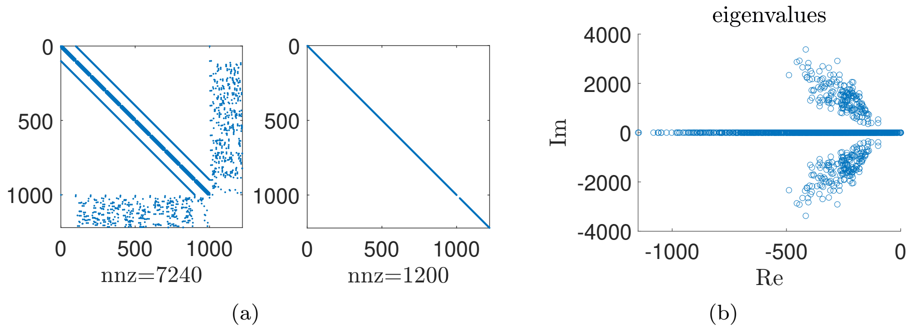

FIG. 4. (a) *Patterns of $G$ and $C$ when $n_x = 10$.* (b) *Eigenvalue distribution.*

discretization that preserves the symmetry will perform better than the one that performs the Gauss quadrature on the whole circle directly.

From Figure 3, we find that the trapezoidal quadrature always outperforms the Gauss quadrature. The figure also shows that the trapezoidal quadrature attains the same rate with respect to $k$ as the optimal one, as the slope of the straight line of the trapezoidal quadrature is the same as that of the optimal one.

We remark that the convergence behavior depends on the distribution of eigenvalues. Our analysis in section 4 views the desired spectrum and the undesired spectrum as a disk and the complement of a disk, while the eigenvalues of a matrix are discrete points in these regions. It is possible that the discrete eigenvalues avoid all bad areas in both the numerator and the denominator of (4.1) with Gauss quadrature and have a small ratio $\mathfrak{R}$ of (4.1). In such a case, the rational filter with Gauss quadrature could outperform the rational filter with trapezoidal quadrature for some matrices. Without prior knowledge of the distribution of eigenvalues, the trapezoidal-quadrature-based filter is a near-optimal choice.

**6.2. Composite rule with subspace iteration.** We compare Algorithm 3.1 against HFEAST with $k_1 = k_2 = 8$ and $k = k_1 \cdot k_2 = 64$.

The class of non-Hermitian generalized eigenvalue problems comes from the model order reduction tasks [2, 11] in the circuit simulation [10]. Matrices are constructed based on quasi-two-dimensional square power grids of size $n_x \times n_x \times 10$. The non-Hermitian matrix pencil is $(G, C)$, and the pattern and distribution of eigenvalues for $n_x = 10$ are shown in Figure 4. One can find the matrix construction details in

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

TABLE 2

*Matrix information. Columns show sizes and the number of nonzeros (nnz) of the $G+C$ matrix for various $n_x$. The centers and radii of target regions are included, and each encloses 20 eigenvalues. The last column includes the runtime ratio of one matrix factorization and one triangular solve.*

| $n_x$ | Size | nnz | $(c,r)$ | $C_{\text{factor}}/C_{\text{apply}}$ |
|---|---|---|---|---|
| 10 | 1,220 | 7,440 | $(-200+1000\imath,90)$ | 33.706 |
| 100 | 120,020 | 776,040 | $(-101+22\imath,3)$ | 47.903 |
| 200 | 480,020 | 3,112,040 | $(-24+4.7\imath,2.1)$ | 71.119 |
| 400 | 1,920,020 | 12,464,040 | $(-5.3+1\imath,0.9)$ | 118.037 |

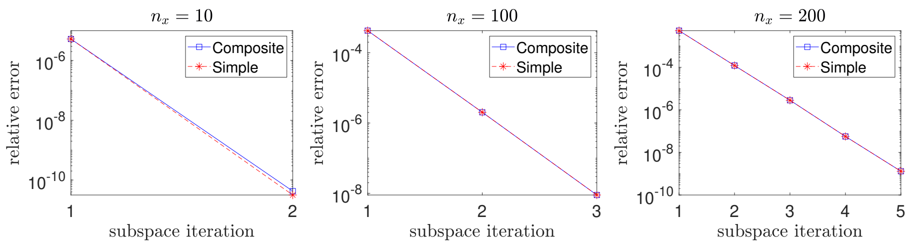

FIG. 5. *Convergence of the simple rule and the composite rule.*

Appendix B. Table 2 lists information about matrices used in our numerical experiments as well as their target regions. The last column of Table 2 includes the runtime ratio of one matrix factorization and one triangular solve. In all cases, there are 20 eigenvalues in their target regions and we adopt $n_{\text{col}} = 24$. Reference eigenvalues are calculated by `eigs` in MATLAB. The stopping criterion of GMRES is $10^{-9}$. The convergence behaviors are illustrated in Figure 5. Runtime is reported in Table 3. The italic values therein are estimated numbers since the simple rule runs out of memory for those settings.

Figure 5 shows that the composite rule converges in a similar fashion to the simple rule. This indicates that both the GMRES and the direct solver achieve sufficiently good accuracy. In most cases we have tested, the subspace iteration converges effectively when many poles are used, i.e., usually in a few iterations.

The composite rule establishes a trade-off between the number of matrix factorizations and the number of triangular solves in GMRES. Table 3 shows a comparison of the simple rule and the composite rule in two parts: runtime and memory. As shown in the last column of Table 2, the runtime ratio between the factorization and the triangular solve grows as the matrix size increases, which is because the matrix factorization is of higher-order complexity compared to that of the triangular solve. Hence, reducing the number of factorizations as in the composite rule would be beneficial for large matrices.

However, as shown in all cases of Table 3, the simple rule outperforms the composite rule in total runtime, because the solving time dominates. The domination comes from the increased number of subspace iterations, which is due to the denser spectrum of the larger case. It is hard to tell when the composite rule outperforms the simple rule for a specific case. The guidance is that when the factorization time dominates, the composite rule can help us substitute the solving time for the factorization time, which reduces the total runtime.

Regarding the memory cost, the simple rule costs about $k_2$ times more than the composite rule. In these examples, we find that the simple rule with $n_x = 400$ already

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

TABLE 3

*Runtime (second) of the simple rule and the composite rule for matrices in Table 2. Italic values are estimated due to the out-of-memory limit. "Comp" means the composite rule.*

| $n_x$ | Total | | Factorization | | Solving | |
|---|---|---|---|---|---|---|
| | Simple | Comp | Simple | Comp | Simple | Comp |
| 10 | $1.5 \times 10^0$ | $2.8 \times 10^0$ | $6.2 \times 10^{-1}$ | $7 \times 10^{-2}$ | $5.6 \times 10^{-1}$ | $1.9 \times 10^0$ |
| 100 | $1.0 \times 10^3$ | $2.5 \times 10^3$ | $4.1 \times 10^2$ | $5.1 \times 10^1$ | $6.0 \times 10^2$ | $2.3 \times 10^3$ |
| 200 | $9.0 \times 10^3$ | $2.8 \times 10^4$ | $3.4 \times 10^3$ | $4.2 \times 10^2$ | $5.6 \times 10^3$ | $2.6 \times 10^4$ |
| 400 | *$5.9 \times 10^4$* | $2.2 \times 10^5$ | *$2.9 \times 10^4$* | $3.6 \times 10^3$ | *$3.1 \times 10^4$* | $2.1 \times 10^5$ |

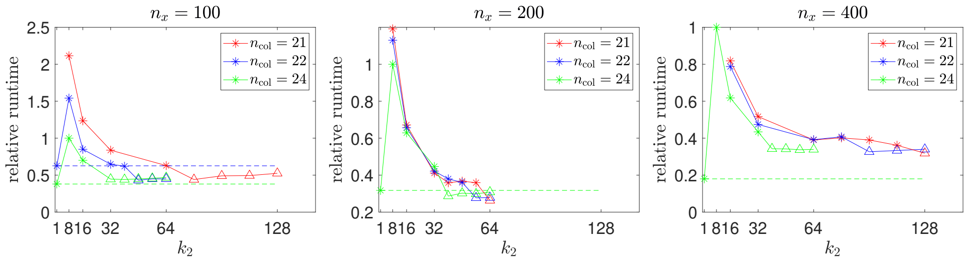

FIG. 6. *Relative runtime of Algorithm* 3.1 *with $k_1 = 8$ and various $k_2$. $k_2 = 1$ represents the simple rule. The runtime is scaled by the runtime of Algorithm* 3.1 *with $k_1 = k_2 = 8$ and $n_{\text{col}} = 24$. We do not plot the point that fails to converge. The star marks denote those subspace iterations that converge in more than one iteration, whereas the triangle marks denote those without subspace iteration. (Color figure available online.)*

exceeds our memory limit, whereas the composite rule can solve eigenvalue problems with $n_x = 400$ or even larger. Another benefit of the composite rule is that it allows us to utilize $R_{c,r,k}(B^{-1}A)$ for large $k$ with limited memory.

**6.3. Composite rule without subspace iteration.** This experiment aims to show that with large $k_2$, the composite rule will converge without subspace iteration, and the GMRES iteration number does not increase dramatically when $k_2$ increases. Such an observation means the strategy doubling $k_2$ each time in Algorithm 3.2 would be affordable compared to the case with optimal $k_2$. Throughout this section, we reuse matrix pencils in section 6.2. We perform three algorithms in this section: the simple rule with $k = 8$, the composite rule with $k_1 = 8$, and various choices of fixed $k_2$ (Algorithm 3.1) and Algorithm 3.2 with $k_1 = 8$. Also, various choices of the numbers of columns are explored.

We set the simple rule to have no more than 100 subspace iterations, while for the composite rule, the limitation is 10. We also terminate Algorithm 3.1 when in the first subspace iteration all eigenpairs converge to target tolerance $\tau$ for relative error. All 20 eigenpairs inside are filtered, and the relative errors are less than $\tau$ for the composite rule. Figure 6 illustrates the relative runtime of Algorithm 3.1 for various $k_2$, and Table 4 reports details of the simple rule and Algorithm 3.2. The relative runtime of Algorithm 3.2 could be read from Figure 6 from those first triangle marks at $k_2$ being a power of two. Table 4 shows more details of the simple rule and Algorithm 3.2. We can estimate the $n_{\text{iter}}$ for the cases of the simple rule that all eigenpairs of interest are filtered but the relative error cannot decrease to target $\tau$ within 100 subspace iterations, from the equation $e^{\lfloor n_{\text{iter}}/100 \rfloor} = \tau$. We use italic numbers to distinguish estimated $n_{\text{iter}}$ from the real one.

In Table 4, all three choices of $n_{\text{col}}$ overestimate the actual number of eigenvalues in the region. The simple rule with a fixed $k = 8$ fails to converge when $n_{\text{col}}$ is

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

TABLE 4

*Details of the simple rule and Algorithm 3.2. The column $p$ shows the number of filtered eigenpairs, i.e., the number of approximate eigenpairs inside the region whose relative error is less than $\tau_g$. The column $e$ shows the relative error when the algorithm converges or the limitation of subspace iteration is attained. The column $n_{\text{iter}}$ shows the times of applying $G$ to a set of vectors for the simple rule, while for Algorithm 3.2, the column $n_{\text{iter}}$ shows the maximum GMRES step of different vectors since the GMRES step will change with the vector. When not all eigenpairs are filtered, we use "-" for $e$ and $n_{\text{iter}}$, since the algorithm will fail to filter the eigenpairs of interest even if we run the algorithm with infinite $n_{\text{iter}}$, or $n_{\text{iter}}$ would be no less than 400 for the target precision $\tau$, which can be derived by two equations $e^{\lfloor n_{\text{iter}}/100 \rfloor} = \tau$ and $\sqrt[4]{\tau} = \tau_g < e$.*

| $(n_x, n_{\text{col}})$ | Simple | | | Composite | | |
|---|---|---|---|---|---|---|
| | $p$ | $e$ | $n_{\text{iter}}$ | $p$ | $e$ | $n_{\text{iter}}$ |
| (100,21) | 19 | - | - | 20 | $1.0 \times 10^{-10}$ | 39 |
| (100,22) | 20 | $7.7 \times 10^{-9}$ | 64 | 20 | $9.6 \times 10^{-11}$ | 39 |
| (100,24) | 20 | $7.0 \times 10^{-9}$ | 35 | 20 | $4.8 \times 10^{-11}$ | 39 |
| (200,21) | 20 | $2.0 \times 10^{-4}$ | *217* | 20 | $3.1 \times 10^{-9}$ | 51 |
| (200,22) | 20 | $4.2 \times 10^{-7}$ | *126* | 20 | $1.9 \times 10^{-10}$ | 51 |
| (200,24) | 20 | $8.2 \times 10^{-9}$ | 57 | 20 | $2.9 \times 10^{-11}$ | 51 |
| (400,21) | 19 | - | - | 20 | $2.3 \times 10^{-9}$ | 87 |
| (400,22) | 19 | - | - | 20 | $3.5 \times 10^{-9}$ | 87 |
| (400,24) | 20 | $7.0 \times 10^{-9}$ | 46 | 20 | $2.3 \times 10^{-9}$ | 87 |

not sufficiently large, e.g., $n_{\text{col}} = 21, 22$. In contrast, Algorithm 3.2 converges in all scenarios. Based on this experiment and other experiments we tried but did not list in the current paper, the convergence of the simple rule is sensitive to the choice of two hyperparameters, $k$ and $n_{\text{col}}$, while the convergence of Algorithm 3.2 is more robust. In the worst-case scenario, when the given region is enclosed by many unwanted eigenvalues, extremely large $n_{\text{col}}$ would be needed to resolve the convergence issue in the simple rule. When the simple rule and Algorithm 3.2 converge, the latter outperforms the former for small $n_{\text{col}}$; see all the red curves and blue curves in Figure 6. From green curves, we know that when $n_{\text{col}}$ increases, these two methods become comparable on runtime.

Figure 6 also explores the optimal choice of $k_2$ without subspace iteration, i.e., the first triangle marks on each curve. We find that the optimal $k_2$ is not necessarily $2^p k_1$ as in Algorithm 3.2. Besides the factorization cost, the dominant computational cost of the composite rule is the multishift GMRES iteration number, i.e., the number of applying $G$ (3.2). Increasing $k_2$ would add more shifts to the multishift GMRES but not necessarily increase the iteration number, and the extra cost of orthogonalization is negligible compared to that of applying $G$. In all curves in Figure 6, we observe that, after their first triangle marks, the relative runtime mostly stays flat and increases extremely slowly. Hence, even if Algorithm 3.2 is not using the optimal $k_2$, the runtime of Algorithm 3.2 is almost the same as that with optimal $k_2$. We conclude that Algorithm 3.2 is an efficient and robust eigensolver and is more preferred than Algorithm 3.1.

*Remark* 6.1. We remark on the hyperparameter choices in Algorithm 3.2. Given a matrix pencil and a region, an overestimation $n_{\text{col}}$ of the number of eigenvalues $s$ is required. If we perform factorizations and triangular solves sequentially, we may need to choose a proper $k_1$ depending on whether factorizations are more expensive than those of triangular solves. In the view of parallelization, the $k_1$ factorizations and the corresponding triangular solves are ideally parallelizable. Hence, we would set $k_1$ as large as possible to fully use the computation resource and reduce the GMRES iterations.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

## 7. Conclusion.

This paper finds the optimal rational function in the sense of spectrum separation via Zolotarev's third function. The optimal rational function leads to the traditional inverse power method in numerical linear algebra. Discretizing the contour integral with the standard trapezoidal quadrature results in an asymptotically optimal rational function. Further, we derive the composite rule of the trapezoidal quadrature, i.e., $R_{c,r,k}(z) = R_{0,1,k_2}(M(R_{c,r,k_1}(z)))$ for $k = k_1k_2$ being a positive integer factorization and $M$ being a Möbius transform.

Based on the composite rule, we propose two eigensolvers for the generalized non-Hermitian eigenvalue problems, Algorithm 3.1 and Algorithm 3.2. Both algorithms adopt direct matrix factorizations for the inner rational function evaluation and multishift GMRES for the outer rational function. Compared to the simple rule with the same number of poles, both composite rule–based algorithms reduce the number of factorizations and reduce the memory requirement. This is of fundamental importance when matrices are of large scale. The difference between the two composite algorithms is the subspace iteration. In Algorithm 3.1, both $k_1$ and $k_2$ are hyperparameters, and the algorithm adopts the subspace iteration to converge to desired eigenpairs. In contrast, Algorithm 3.2 is designed without subspace iteration. Algorithm 3.2 adopts $k_1$ as a hyperparameter and gradually increases $k_2$ until the rational function approximation is accurate enough, and the algorithm converges to desired eigenpairs without subspace iteration. As $k_2$ increases in Algorithm 3.2, by the property of multishift GMRES, the number of GMRES iterations, i.e., the number of applying $G$, increases very mildly. Hence, compared to the simple rule and Algorithm 3.1, Algorithm 3.2 is a robust and efficient eigensolver.

We demonstrate the efficiency of the proposed algorithms by synthetic and practical generalized non-Hermitian eigenvalue problems. Numerical results show that Algorithm 3.1 outperforms the simple rule only if the matrix factorization is much more expensive than the triangular solve. The convergence of Algorithm 3.2 is not sensitive to hyperparameter $n_{\text{col}}$. In terms of the runtime, Algorithm 3.2 either outperforms or is comparable to the simple rule. A suggestion for the hyperparameter choices of Algorithm 3.2 is also provided based on both the analysis and numerical results.

## Appendix A. GMRES iteration number.

As we mentioned in subsection 3.2, the convergence of the multishift GMRES method accelerates as the subspace iteration progresses. Table 5 reports the number of triangular solves in both the simple and the composite rules, and the GMRES iteration number in the composite rule. The normalized last column of Table 5 is visualized in Figure 7.

Table 5 shows that the number of triangular solves in each subspace iteration of the simple rule stays constant, whereas that for the composite rule decreases. Notice that the $n_{\text{iter}}$ decays much slower than the number of triangular solves in the composite rule. That is because different columns converge to eigenvectors with different rates.

## Appendix B. Construction of matrix.

Matrices are constructed based on quasi-two-dimensional square power grids of size $n_x \times n_x \times 10$. The non-Hermitian matrix pencil is $(G, C)$ taking the block form as

$$G = \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & 0 \end{bmatrix}, \quad C = \begin{bmatrix} C_c & 0 \\ 0 & L \end{bmatrix}.$$

In particular, $G_{11}$ represents the conductance matrix as $G_{11} = L_{n_x} \otimes I_{n_x} \otimes I_{10} + I_{n_x} \otimes L_{n_x} \otimes I_{10} + \frac{1}{10} I_{n_x} \otimes I_{n_x} \otimes L_{10}$, where $L_n$ is a weighted one-dimensional Laplacian matrix of size $n \times n$ as

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

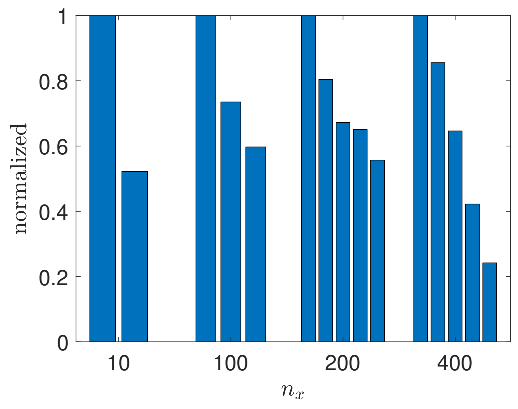

FIG. 7. *Normalized solving cost in the composite rule. In each test matrix, the bars show the number of triangular solves in each subspace iteration, which are normalized by the number of triangular solves in the first subspace iteration.*

TABLE 5

*Number of triangular solves of the simple rule and composite rule in each subspace iteration are shown in the "Solving" columns, respectively. The column "$n_{\text{iter}}$" shows the number of GMRES iterations in each subspace iteration. Italic values are estimated since the required memory is beyond our machine memory.*

| $n_x$ | Simple | Composite | |
|---|---|---|---|
| | Solving | $n_{\text{iter}}$ | Solving |
| 10 | [1536,1536] | [32,22] | [6128,3200] |
| 100 | [1536,1536,1536] | [39,32,31] | [7488,5504,4472] |
| 200 | [1536,1536,1536,1536,1536] | [51,43,38,37,37] | [9680,7784,6504,6296,5392] |
| 400 | $[\mathit{1536}, \mathit{1536}, \mathit{1536}, \mathit{1536}, \mathit{1536}]$ | [86,84,75,67,58] | [16328,13968,10552,6896,3952] |

$$L_n = \frac{n}{100}\begin{bmatrix} 1 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 1 \end{bmatrix}_{n\times n}$$

and $I_n$ is an identity matrix of size $n \times n$. The off-diagonal blocks of $G$ admit $G_{12} = -G_{21}^\top \in \mathbb{R}^{10n_x^2 \times (20+2n_x^2)}$ with entries being $\pm 1$ or zero. The first 20 columns of $G_{12}$ correspond to the 20 input ports located at the two edges, specifically at positions $(\cdot, 1, 1)$ and $(\cdot, n_x 10)$. The corresponding rows in these columns contain a positive one. The remaining $2n_x^2$ columns of $G_{12}$ correspond to inductors. We uniformly randomly pick $2n_x^2$ interior nodes from grid nodes and add an inductor with their neighbor nodes on the same layer. The corresponding $G_{12}$ part is the incidence matrix of the inductor graph. Matrix $L$ is a diagonal matrix of size $20 + 2n_x^2$. The first $20 \times 20$ block of $L$ is zero. The latter $2n_x^2 \times 2n_x^2$ block has diagonal entries uniformly randomly sampled from $[0.5, 1.5] \cdot n_x \cdot 10^{-4}$ being the inductance of inductors. The submatrix $C_c$ represents capacitors in the circuit. For each node, we add a grounded capacitor with capacitance uniformly randomly sampled from $[0.5, 1.5] \cdot 10^{-3}$, which means $C_c$ is a diagonal matrix whose elements are equal to the capacitances.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

**Acknowledgment.** We thank Chao Yang for the helpful discussions.

## REFERENCES

[1] T. Bakhos, P. K. Kitanidis, S. Ladenheim, A. K. Saibaba, and D. B. Szyld, *Multipreconditioned GMRES for shifted systems*, SIAM J. Sci. Comput., 39 (2017), pp. S222–S247, https://doi.org/10.1137/16M1068694.

[2] P. Gross, R. Arunachalam, K. Rajagopal, and L. Pileggi, *Determination of worst-case aggressor alignment for delay calculation*, in ICCAD'98: Proceedings of the 1998 IEEE/ACM International Conference on Computer-Aided Design, 1998, pp. 212–219, https://doi.org/10.1145/288548.288616.

[3] R. Huang, J. Sun, and C. Yang, *Recursive integral method with Cayley transformation*, Numer. Linear Algebra Appl., 25 (2018), e2199, https://doi.org/10.1002/nla.2199.

[4] 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, https://doi.org/10.11650/twjm/1500405869.

[5] T. Ikegami, T. Sakurai, and U. Nagashima, *A filter diagonalization for generalized eigenvalue problems based on the Sakurai–Sugiura projection method*, J. Comput. Appl. Math., 233 (2010), pp. 1927–1936, https://doi.org/10.1016/j.cam.2009.09.029.

[6] J. Kestyn, E. Polizzi, and P. T. P. Tang, *FEAST eigensolver for non-Hermitian problems*, SIAM J. Sci. Comput., 38 (2016), pp. S772–S799, https://doi.org/10.1137/15M1026572.

[7] R. B. Lehoucq, D. C. Sorensen, and C. Yang, *ARPACK Users' Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*, Software Environ. Tools 6, SIAM, Philadelphia, 1998, https://doi.org/10.1137/1.9780898719628.

[8] Y. Li and H. Yang, *Interior eigensolver for sparse Hermitian definite matrices based on Zolotarev's functions*, Commun. Math. Sci., 19 (2021), pp. 1113–1135, https://doi.org/10.4310/CMS.2021.v19.n4.a11.

[9] C. B. Moler and G. W. Stewart, *An algorithm for generalized matrix eigenvalue problems*, SIAM J. Numer. Anal., 10 (1973), pp. 241–256, https://doi.org/10.1137/0710024.

[10] F. N. Najm, *Circuit Simulation*, Wiley-IEEE Press, 2010.

[11] A. Odabasioglu, M. Celik, and L. T. Pileggi, *PRIMA: Passive reduced-order interconnect macromodeling algorithm*, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., 17 (1988), pp. 645–654, https://doi.org/10.1109/43.712097.

[12] P. P. Petrushev and V. A. Popov, *Rational Approximation of Real Functions*, Encyclopedia Math. Appl. 28, Cambridge University Press, 1988, https://doi.org/10.1017/CBO9781107340756.

[13] E. Polizzi, *Density-matrix-based algorithm for solving eigenvalue problems*, Phys. Rev. B, 79 (2009), 115112, https://doi.org/10.1103/PhysRevB.79.115112.

[14] Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd ed., SIAM, Philadelphia, 2003, https://doi.org/10.1137/1.9780898718003.

[15] T. Sakurai and H. Sugiura, *A projection method for generalized eigenvalue problems using numerical integration*, J. Comput. Appl. Math., 159 (2003), pp. 119–128, https://doi.org/10.1016/S0377-0427(03)00565-X.

[16] G. Starke, *Near-circularity for the rational Zolotarev problem in the complex plane*, J. Approx. Theory, 70 (1992), pp. 115–130, https://doi.org/10.1016/0021-9045(92)90059-W.

[17] G. W. Stewart, *A Krylov–Schur algorithm for large eigenproblems*, SIAM J. Matrix Anal. Appl., 23 (2001), pp. 601–614, https://doi.org/10.1137/S0895479800371529.

[18] G. Yin, *A harmonic FEAST algorithm for non-Hermitian generalized eigenvalue problems*, Linear Algebra Appl., 578 (2019), pp. 75–94, https://doi.org/10.1016/j.laa.2019.04.036.

[19] G. Yin, R. H. Chan, and M.-C. Yeung, *A FEAST algorithm with oblique projection for generalized eigenvalue problems*, Numer. Linear Algebra Appl., 24 (2017), e2092, https://doi.org/10.1002/nla.2092.

Downloaded 04/10/25 to 202.120.235.10 . Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/terms-privacy

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.