CONTOUR INTEGRATION AND THE GENERALIZED EIGENVALUE PROBLEM

CHRISTOPHER TONG

ABSTRACT. The generalized eigenvalue problem $H\mathbf{x} = \epsilon S\mathbf{x}$ for Hermitian H and symmetric positive definite S plays a central role in computational physics and chemistry for modelling the spatial distribution of charge density in a molecule or extended solid. In this paper, we discuss the natural formulation of the generalized eigenvalue problem in terms of contour integration in the complex plane. We review two results [1, 2] that use this formulation to propose methods that solve the generalized eigenvalue problem.

1. INTRODUCTION

In the late 19$^{\text{th}}$ and early 20$^{\text{th}}$ centuries, advances in experimental physics allowed for the first observations of phenomena at atomic length and energy scales. These new phenomena were not well described by existing theories of classical mechanics, and necessitated developments of new theory. In 1926, Erwin Schrödinger proposed a quantum theory where particles in a time independent system can be described by the eigenfunctions $\{\psi_i : \mathbb{R}^3 \to \mathbb{C}\}$ of a Hermitian differential operator $\hat{H}$ in the position $\mathbf{r} \in \mathbb{R}^3$ [3]:

$$\hat{H}\psi_i(\mathbf{r}) = \epsilon_i \psi_i(\mathbf{r}) \tag{1.1}$$

In the Schrödinger picture, $\hat{H}$ acts on the Hilbert space of normalized square integrable functions ψ with inner product $\langle \psi_1, \psi_2 \rangle \equiv \int \mathrm{d}^3\mathbf{r}\, \psi_1^*(\mathbf{r})\psi_2(\mathbf{r})$. As a result, the amplitude $\left|\psi^2(\mathbf{r})\right| = \psi^*(\mathbf{r})\psi(\mathbf{r})$ at any point represents a corresponding probability of finding the particle in eigenstate ψ_i at position $\mathbf{r}$. The corresponding eigenvalue $\{\epsilon_i\}$ represents its respective energy. Schrödinger's theory is still generally accepted today as a powerful description of particles in non-relativistic systems such as an electron in a molecule of water or a copper wire.

However, the infinite-dimensional problem of computing eigenfunctions of a general differential operator is extremely expensive if not totally infeasible for $\hat{H}$ of almost any physical system. As a result, a common approach in computational physics and chemistry is to project the problem into the space spanned by a finite basis set of functions $\{\mu_j\}$. We define the matrices H and S according their matrix elements given by the previously defined inner product

$$\begin{aligned} H_{jj'} &= \int \mathrm{d}^3\mathbf{r}\, \mu_j^*(\mathbf{r})\hat{H}\mu_{j'}(\mathbf{r}) \\ S_{jj'} &= \int \mathrm{d}^3\mathbf{r}\, \mu_j^*(\mathbf{r})\mu_{j'}(\mathbf{r}) \end{aligned} \tag{1.2}$$

Since $\hat{H}$ is Hermitian, the matrix H is also Hermitian. The properties of S can be controlled by our choice of basis set $\{\mu_j\}$ - we note that if the $\{\mu_j\}$ are orthonormal, S is simply the identity matrix. We consider the more general problem where S is symmetric positive definite. As a result, the eigenvectors $\{\mathbf{x}_i\}$ to the now-discretized problem

$$H\mathbf{x} = \epsilon S\mathbf{x} \tag{1.3}$$

can be used to easily recover approximations to eigenfunctions of $\hat{H}$ via

$$\psi_i \approx \mathbf{x}_i^* \begin{pmatrix} \mu_1 \\ \mu_2 \\ \cdots \\ \mu_n \end{pmatrix} \tag{1.4}$$

The problem of constructing suitable basis sets for which we can efficiently compute the matrix elements of H and S and give good approximations for ψ_i at a minimal number for n is a key topic in modern computational quantum physics and chemistry. In this paper, we assume that a suitable basis set $\{\mu_j\}$ has already

been selected and focus on solving the resulting discretized eigenvalue problem.

To establish the connection to contour integration, we define the Green's function[1] $G : W \to \mathbb{C}^{nxn}$ on some region $W \subset \mathbb{C}$ as the unique solution to the matrix equation

$$G(z)(zS - H) = I \tag{1.5}$$

for any $z \in W$. We note that $zS-H$ is singular if and only if there exists $\mathbf{x} \in \mathbb{C}^n - \mathbf{0}$ such that $zS\mathbf{x}-H\mathbf{x} = 0$, ie. z is an eigenvalue of H. As a result, $G(z) = (zS-H)^{-1}$ is well defined on all of $\mathbb{C}$ except at the eigenvalues of H. Since H is a Hermitian $\mathbb{C}^{nxn}$ matrix, these values of z are simply n isolated points on the real line. We consider the case where H has n distinct eigenvalues[2].

The Green's function formulation provides the fundamental connection between eigenvalues of $zS - H$ and functions on the complex plane $G(z)$ [4]:

Theorem 1.6. *For eigenpairs $(\lambda_j, \mathbf{x}_j)$ of $zS-H$ for S symmetric positive definite and H Hermitian, where each eigenvector $\mathbf{x}_j$ is normalized, the corresponding Green's function is $G(z) = \frac{\mathbf{x}_j\mathbf{x}_j^*}{z-\lambda_j}$.*

Proof. Per the spectral theorem, H is diagonal in the eigenvectors $\mathbf{x}_j$. Therefore, for any i, j:

$$\mathbf{x}_i^* H \mathbf{x_j} = \lambda_i \delta_{ij} \tag{1.7}$$

Additionally, the assumed normalization of the solutions to our initial problem ψ and elements of the basis set μ implies that

$$\mathbf{x}_i^* S \mathbf{x_j} = \delta_{ij} \tag{1.8}$$

Therefore,

$$\mathbf{x_i}^* G(z)(zS - H)\mathbf{x_j} = \delta_{ij} \tag{1.9}$$

Then, performing left multiplication of $\mathbf{x_j}$ by $zS - H$ gives

$$\mathbf{x_i}^* G(z)(z - \lambda_j)\mathbf{x_j} = \delta_{ij} \to \mathbf{x_i}^* G(z)\mathbf{x_j} = \frac{\delta_{ij}}{(z - \lambda_j)} \tag{1.10}$$

It follows by direct evaluation that

$$G(z) = \sum_j \frac{\mathbf{x}_j\mathbf{x}_j^*}{z - \lambda_j} \tag{1.11}$$

is a solution to this matrix equation. Since $zS - H$ is non-singular except at isolated real λ_j, this solution for $G(z)$ is unique everywhere except at those points. □

Corollary 1.12. *For a closed, positively oriented, smooth curve $\Gamma : \mathbb{R} \to W \subset \mathbb{C}$ where $G(z)$ is uniquely defined everywhere along Γ, $\frac{1}{2\pi i}\oint_\Gamma G(z)\,\mathrm{d}z = \sum_{\lambda_j \in R} \mathbf{x_j}\mathbf{x_j}^*$, where R denotes the interior of Γ, and the integral is performed componentwise.*

Proof. Let us consider a specific component $G_{ab}(z)$ on the a^{th} row and b^{th} column of $G(z)$. By Thm. 1.6, we have that

$$G_{ab}(z) = \sum_j \frac{x_{ja}x_{jb}^*}{z - \lambda_j} \tag{1.13}$$

where x_{jk} is the k^{th} entry of eigenvector $\mathbf{x}_j$. Then, it is clear that $G_{ab} : \mathbb{C} \to \mathbb{C}$ has simple poles at λ_j whenever $x_{ja}^* x_{jb}$ is nonzero and is well defined everywhere else. Furthermore, since $G(z)$ is well defined

[1] We note that this is consistent with the general theory of Green's functions as the associated impulsive response of the differential operator $\hat{H}$ - note that for operator equation $(zI - \hat{H})(\mathbf{r})\hat{G}(\mathbf{r}) = \delta^3(\mathbf{r})$, computing the matrix representation of both sides of this equation in our chosen basis set $\{\mu_j\}$ gives the definition in Eqn. 1.5.

[2] We refer to the treatment in Theorem 2 of [1] for the case where H has possibly repeated eigenvalues, in which case we can use the Jordan decomposition of $zS - H$ to construct a similar expression for $G(z)$.

everywhere on Γ, Γ does not pass through any poles λ_j. Therefore, the Cauchy integral formula[3] implies that

$$\frac{1}{2\pi i}\oint_\Gamma G_{ab}(z)\,\mathrm{d}z = \sum_{\lambda_j \in R} x_{ja}^* x_{jb} \tag{1.14}$$

Therefore, the integral when taken over all components of G must satisfy

$$\frac{1}{2\pi i}\oint_\Gamma G(z)\,\mathrm{d}z = \sum_{\lambda_j \in R} \mathbf{x_j}\mathbf{x_j}^* \tag{1.15}$$

□

Importantly, Cor. 1.12 can be used to construct equations for the entries of the eigenvectors $\mathbf{x}_j$ given numerically evaluated values for the contour integral of $G_{ab}(z)$ along Γ. Furthermore, the value of the contour integral can be decomposed into discrete contributions from eigenvalues λ_j lying in the interval $[\lambda_{\min}, \lambda_{\max}]$ along the real line that is enclosed by Γ. Therefore, this equation captures all the necessary information to numerically determine the eigenvectors and eigenvalues of $zS - H$. In particular, there are two methods of interest that use this general approach to efficiently solve the generalized eigenvalue problem, which we review below.

2. Projection method

The projection method, published by Sakurai and Sugiura in 2003 [1], involves projecting a series expansion of the Green's function $G(z)$ of the target eigenvalue problem onto a smaller-dimensional space on which the target eigenvalue problem can be solved much more efficiently with traditional, exact methods. The projection method employs a scalarized $f : \mathbb{C} \to \mathbb{C}$ version of the Green's function $f(z) = u^* G(z) v$ for some $u, v \in \mathbb{C}^n$. Since $f(z)$ is simply an arbitrary linear combination of the matrix elements G_{ab}, $f(z)$ has a simple pole at every eigenvalue λ_j enclosed by Γ for generic choices of u and v by Thm. 1.6.

For convenience, let the contour Γ as defined in Sec. 1 enclose the interval $[\lambda_{\min}, \lambda_{\max}]$ along the real line which contains the origin[4]. Then, let us consider the k^{th} moments with respect to the origin of the Green's function f along Γ, where k ranges from $0, \ldots, 2K-1$ for some $K \geq J$, where J is the total number of distinct eigenvalues $\lambda_j \in R$. This can be computed numerically:

$$m_k = \frac{1}{2\pi i}\oint_\Gamma z^k f(z)\,\mathrm{d}z \tag{2.1}$$

Since 0 is not a pole of f, we note that the residue at any pole $\operatorname{Res}_{\lambda_j} z^k f(z)$ is simply $\lambda_j^k \operatorname{Res}_{\lambda_j} f(z)$ as λ_j remains a simple pole of $z^k f(z)$. Therefore,

$$m_k = \sum_{\lambda_j \in R} \lambda_j^k \operatorname{Res}_{\lambda_j} f(z) \tag{2.2}$$

Theorem 2.3. *The generalized eigenvalue problem $H_K \mathbf{y} = \lambda H_K^> \mathbf{y}$ defined by*

$$H_K = \begin{pmatrix} m_0 & m_1 & \cdots & m_{K-1} \\ m_1 & \ddots & & m_K \\ \vdots & & \ddots & \vdots \\ m_{K-1} & \cdots & \cdots & m_{2K-2} \end{pmatrix}, H_K^> = \begin{pmatrix} m_1 & m_2 & \cdots & m_K \\ m_2 & \ddots & & m_{K+1} \\ \vdots & & \ddots & \vdots \\ m_K & \cdots & \cdots & m_{2K-1} \end{pmatrix} \tag{2.4}$$

[3]Although a fundamental discussion of integrals of such form in the complex plane is beyond the scope of this paper, the fundamental intuition for those unfamiliar with complex integration is that for Γ closed, the only contributions to the integral of G_{ab} on Γ arise from discrete points at which $|G_{ab}|$ diverges in the interior of Γ. The value of the contribution from a specific λ_j scaled by $2\pi i$ is known as the residue at λ_j. See an introductory complex analysis textbook, such as the text by Stein and Shakarchi, for a comprehensive description.

[4]Without loss of generality, we can perform a global shift of $H \to H - cI$ for some constant $c \in \mathbf{R}$ in our problem in order to shift our region of interest to contain the origin.

for $H_K, H_K^> \in \mathbb{C}^{KxK}$ represents the projection of the original eigenvalue problem $H\mathbf{x} = \lambda S\mathbf{x}$ onto a K dimensional subspace containing the span of eigenvectors $\{\mathbf{x}_j\}$ with eigenvalues $\lambda_j \in R$.

Proof. The construction in the proof is due to [1], which takes inspriation from the method presented in [5] for the very similar problem of polynomial root-finding. By the previous expression for the moments m_k, we have that the entry in the a^{th} row and b^{th} column of H_K, where a and b are indexed starting from 0, is

$$H_{K,ab} = \sum_{\lambda_j \in R} \lambda_j^{a+b} \operatorname{Res}_{\lambda_j} f(z) = \sum_{\lambda_j \in R} \lambda_j^a \big(\operatorname{Res}_{\lambda_j} f(z)\big)\lambda_j^b \tag{2.5}$$

Similarly,

$$H^>_{K,ab} = \sum_{\lambda_j \in R} \lambda_j^{a+b+1} \operatorname{Res}_{\lambda_j} f(z) = \sum_{\lambda_j \in R} \lambda_j^a \big(\lambda_j \operatorname{Res}_{\lambda_j} f(z)\big)\lambda_j^b \tag{2.6}$$

Let us first restrict our consideration to the submatrix given by the first J rows and columns of H_K and $H_K^>$, which we denote H_K^{JJ} and $H_K^{>JJ}$. By inspection of the expression written in the suggestive manner in Eqns. 2.5, 2.6, H_K^{JJ} and $H_K^{>JJ}$ can be simultaneously factored in terms of

$$V = \begin{pmatrix} 1 & \lambda_1 & \dots & \lambda_1^{J-1} \\ 1 & \lambda_2 & \dots & \lambda_2^{J-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \lambda_J & \dots & \lambda_J^{J-1} \end{pmatrix} \tag{2.7}$$

Then,

$$\begin{aligned} H_K^{JJ} &= V \operatorname{diag}(\operatorname{Res}_{\lambda_1} f(z), \operatorname{Res}_{\lambda_2} f(z), \dots, \operatorname{Res}_{\lambda_J} f(z)) V^T \\ H_K^{>JJ} &= V \operatorname{diag}(\lambda_1 \operatorname{Res}_{\lambda_1} f(z), \lambda_2 \operatorname{Res}_{\lambda_2} f(z), \dots, \lambda_J \operatorname{Res}_{\lambda_J} f(z)) V^T \end{aligned} \tag{2.8}$$

so

$$\begin{aligned} H_K^{JJ}(V^T)^{-1}\mathbf{e}_j &= \operatorname{Res}_{\lambda_j} f(z) V \mathbf{e}_j \\ H_K^{>JJ}(V^T)^{-1}\mathbf{e}_j &= \lambda_j \operatorname{Res}_{\lambda_j} f(z) V \mathbf{e}_j \end{aligned} \tag{2.9}$$

Therefore, $(V^T)^{-1}\mathbf{e}_j$ is an eigenvector of $H_K^{JJ} - \lambda H_K^{>JJ}$ with eigenvalue λ_j for all $1 \le j \le J$. However, H_K^{JJ} and $H_K^{>JJ}$ are simply formed from truncation of the K dimensional matrices given by H_K and $H_K^>$, so the subspace onto which our origianl eigenvalue problem becomes projected onto must contain the span of eigenvectors associated with eigenvalues λ_j. In particular, when $J = K$, we will not obtain any spurious eigenvalues λ that are not eigenvalues of our original problem; however, the value of J is not generally known *a priori*; as a result, it is important to select some $K > J$ based on an intuitive understanding of the approximate density of eigenvalues in the spectrum of the original problem.

□

To emphasize the connection between the eigenvectors of $zS - H$ and the residue of f at poles λ_j, we explicitly note the expansion

$$\operatorname{Res}_{\lambda_j} f(z) = \sum_{1 \le a,b \le J} u_a^* x_{ja}^* x_{jb} v_b \tag{2.10}$$

As a result, computing $2K$ moments of $f(z)$ about Γ gives us a plethora of information regarding each component of each iegenvalue We refer to [1] for a detailed description of a concise and efficient implementation of this method for determining eigenvectors $\mathbf{x}_j$ via knowledge of the λ_j.

3. FEAST method

A main drawback of the projection method is the need to numerically calculate $\oint_\Gamma z^k f(z)\,\mathrm{d}z$ for $k = 0, 1, \dots, 2K-1$ for some $K \ge J$. As a result, for large contours Γ, not only does the number of integrals that need to be performed increase with $k \gtrapprox$ the number of eigenvalues we expect to enclose, but z^k also grows very quickly for $|z| > 1$. Therefore, the regime in which the projection method is robust and efficient is limited by both the expected density of eigenvalues of A as well as the exponential growth of z^k. As a result, the projection method can become difficult to use in practice. The FEAST method, published by Polizzi in 2009 [2], directly evaluates the integral of the Green's function instead of considering moments

due to scaling the integral of $f(z)$ by a polynomial factor. As a result, the FEAST method gives improved scalability of the contour integration-based eigensolver.

The FEAST method relies on the generation of K random, linearly independent vectors $\{\mathbf{y}_k\}$. Then, right multiplication of the result in Cor. 1.12 gives

$$Q = \frac{1}{2\pi i} \oint_\Gamma G(z) Y \, \mathrm{d}z = \left(\sum_{\lambda_j \in R} \mathbf{x}_j \mathbf{x}_j^* \right) Y \tag{3.1}$$

where

$$Y = \begin{pmatrix} \mathbf{y}_1 & \mathbf{y}_2 & \dots & \mathbf{y}_K \end{pmatrix} \tag{3.2}$$

Therefore, we can compute Q via numerical integration of $G(z)Y$. Additionally, expansion and factorization of the right hand side of Eqn. 3.1 gives

$$\left(\sum_{\sum_{\lambda_j \in R}} \mathbf{x}_j \mathbf{x}_j^* \right) Y = X \begin{pmatrix} \mathbf{x_1}^* \mathbf{y_1} & \mathbf{x_1}^* \mathbf{y_2} & \dots \mathbf{x_1}^* \mathbf{y_m} \\ \mathbf{x_2}^* \mathbf{y_1} & \mathbf{x_2}^* \mathbf{y_2} & \dots \mathbf{x_2}^* \mathbf{y_m} \\ \vdots & \ddots & \\ \mathbf{x_k}^* \mathbf{y_1} & \mathbf{x_k}^* \mathbf{y_2} & \dots \mathbf{x_k}^* \mathbf{y_m} \end{pmatrix} \tag{3.3}$$

where

$$X = \begin{pmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_J \end{pmatrix} \tag{3.4}$$

Therefore, the columns of Q are simply a linear combination of the k eigenvectors of $zS - H$ lying inside the contour Γ, which is what we desire to compute. Therefore, letting

$$S_Q = Q^T S Q, H_Q = Q^T H Q \tag{3.5}$$

gives an eigenvalue problem $zS_Q - H_Q$ representing the projection of our original eigenvalue problem $zS - H$ onto the subspace spanned by $\{\mathbf{y_m}\}$. Our eigenvalue problem is now much more computationally tractable, while maintaining that the eigenvalues $\lambda_{i,Q}$ correspond to eigenvalues λ_i of our original problem and eigenvectors $\mathbf{x}_{i,Q}$ are related to the eigenvectors of our original problem by $\mathbf{x}_i = Q\mathbf{x}_{i,Q}$.

We naively implement the FEAST method as described in [2] in Python and compare its performance to the built in generalized eigensolver *scipy.linalg.eig*. We consider a problem of $H \in \mathbb{C}^{1000\text{x}1000}$ that is real symmetric with random entries along its main diagonal selected independently and uniformly on $[0, 10]$ and random entries along the super- and sub-diagonals selected independently and uniformly on $[0, 0.1]$. S is similarly taken to be real symmetric with random entries along its main diagonal selected independently and uniformly on $[0.8, 1.2]$ and random entries along the super- and sub-diagonals selected independently and uniformly on $[0, 0.1]$. A task common to many applications is the problem of finding the eigenvalues of $\lambda S - H$ near a given value of λ_0.[5] Let us consider a representative value of $\lambda_0 = 2$ lying in the bulk of the spectrum.

We evaluate the performance of our implementation of the FEAST algorithm for a circular contour Γ centered at $\lambda_0 = 2$ with varying radius r and varying number of initial random vectors K. We generate m random vectors $\mathbf{y} \in \mathbb{R}^{1000}$. As H and S are real symmetric, we use the form of the integral given in Eqn. (11) of [2]:

$$\frac{1}{2\pi i} \oint_\Gamma G(z) Y \, \mathrm{d}z = -\frac{1}{\pi} \oint_{\Gamma^+} \mathrm{Im}\{G(z)\} Y \, \mathrm{d}z \tag{3.6}$$

where $\Gamma^+(\theta) = \lambda_0 + r\exp(i\theta)$ is parameterized by $\theta \in [0, \pi]$ for some fixed r represents the contour on the upper half complex plane. This integral is approximated using Gauss-Legendre quadrature with 8 values of θ,[6] where for any quadrature point $z \in \Gamma^+$, the matrix $G(z)Y$ is computed by solving the inverse problem $(zS - H)\mathbf{x_j} = \mathbf{y_j}$ at each point $\mathbf{y_j}$.

[5] For example, electrons in an extended solid fill energy states in order to minimize the total energy. As a result, there is a threshhold energy E_F below which electronic states are mostly filled and above which electronic states are mostly empty. Therefore, most physical phenomena involving electronic effects in the solid are mediated by electrons near E_F, so it is important to rigorously understand the spectrum near this threshhold energy.

[6] Points and weights given in Sec. IIIB of [2].

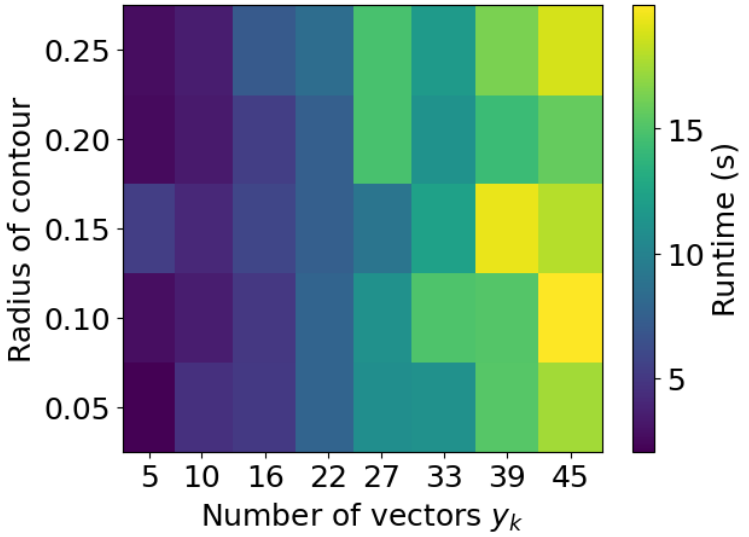

FIGURE 1. Runtime of the FEAST algorithm for a contour of given radius around $\lambda_0 = 2$ when K random vectors $\mathbf{y_k}$ are selected.

We consider the runtime of our implementation of the FEAST algorithm (Fig. 1). As expected, the the runtime of our algorithm increases linearly with K as K sets the number of 1000x1000 linear systems we need to solve to obtain suitable values of $G(z)Y$ for our quadrature method to integrate along the contour. Additionally, the runtime does not depend on the radius of the contour as we maintain an 8-point Gauss-Legendre quadrature scheme for all radius values. Furthermore, we note that the runtime of our naive implementation compares favorably to the built-in eigensolver for the problem $zS - H$, which computes all eigenvectors and eigenvalues of the original problem in ~ 11 seconds. Although this gives much more information about our system than the spectrum near $\lambda_0 = 2$, it is generally difficult to robustly calculate eigenvalues near a general point in the spectrum, even with state-of-the-art Krylov subspace methods. Not only does the FEAST algorithm provide a easily controllable handle on an interval on which we care about computing the eigenvalues of $zS - H$ accurately, we achieve runtimes comparable to the built-in Python eigensolver with a very naive implementation. This suggests that with a finite amount of software and hardware performance engineering, the FEAST approach would be much more efficient than the built-in method for this task.

Finally, we discuss the convergence of the results obtained via the FEAST algorithm to the values obtained via the Python built-in eigensolver (Fig. 2). We note that our implementation of the FEAST algorithm is most successful along a increasing function of contour radius versus number of $\mathbf{y_k}$ vectors selected. This matches our intuition about the performance of the FEAST algorithm, as we expect the largest success rate when we employ K random vectors $\mathbf{y_k}$ that is comparable to J, the number of eigenvalues enclosed by our contour. At an arbitrary point within the bulk of our spectrum of 1000x1000 random matrix, the spectrum near our arbitrary point can be reasonably treated as being distributed uniformly; thus, an increase in the radius linearly increases the number of enclosed eigenvalues in expectation.

If we have $K < J$, our chosen set $\mathbf{y_k}$ cannot possibly include the span of all eigenvectors associated with eigenvalues enclosed by Γ due to the dimensionality mismatch. As a result, the solution to the reduced eigenvalue $zS_Q - H_Q$ generated by FEAST represents a principal component analysis of some vector subspace that may contain some eigenvectors of our original problem, giving very poor convergence to our desired spectrum (Fig. 3, top panel). Therefore, the success rate above the main diagonal of radius-K space in Fig. 2 is very poor. In contrast, if $K > J$ by a significant amount, we perform much better on the front of guaranteeing that the eigenvectors enclosed by Γ are included in the subspace upon which we reduce our problem to. However, we restrict ourselves to a smaller subset of the spectrum to which we can have convergence of eigenvalues (Fig. 3, bottom panel), while incurring the same amount of runtime as if we had

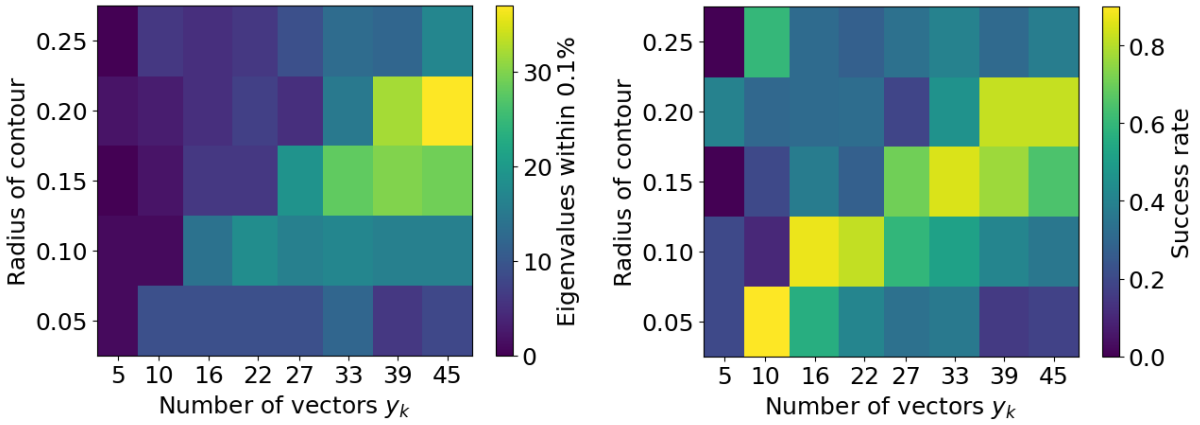

FIGURE 2. (L) Number of eigenvalues calculated with the FEAST algorithm within 0.1% from the corresponding eigenvalues obtained via the Python built-in eigensolver as a function of contour radius and number of random vectors $\mathbf{y_k}$ selected. (R) Number of eigenvalues calculated with the FEAST algorithm within 0.1% as a fraction of the number of random vectors $\mathbf{y_k}$ selected (ie. the dimension of the subspace in which we solve the generalized eigenvalue problem).

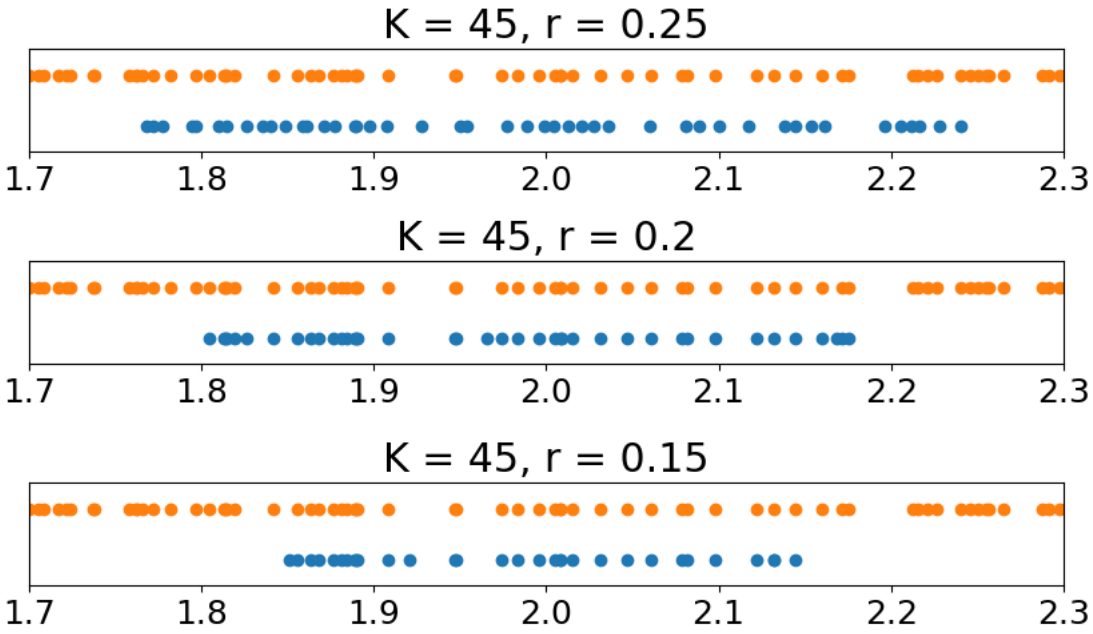

FIGURE 3. Example spectra of $zS - H$ calculated by the FEAST algorithm (blue) when 45 random vectors $\mathbf{y_k}$ are selected and the contour of integration about λ_0 is taken to have radius $0.25, 0.2, 0.15$ respectively. The spectrum of $zS - H$ calculated by the Python built-in eigensolver (orange) is shown for convenience of comparison of each case.

run the algorithm over a contour with larger radius. As a result, selecting the correct hyperparameters in the contour radius and the number of $\mathbf{y_k}$ is essential to successful usage of the FEAST algorithm.

4. Additional remarks on FEAST

The convergence of the FEAST algorithm can be enhanced by converting it to an iterative method, where the initial construction of Y as K random vectors $\{\mathbf{y_k}\}$ is replaced with SX, where X is made of the eigenvectors calculated by the previous iteration of the FEAST algorithm [2]. As a result, this iterative FEAST algorithm can be thought of as a subspace iteration method, for which the established theory related to subspace iteration holds. A convergence analysis of the FEAST algorithm under the subspace iteration formalism is present at [6]. It is important that the FEAST algorithm is fundamentally distinct from methods based upon Krylov subspace iteration as in each iteration of a Krylov method, another dimension in the direction best approximating the minimal polynomial of the problem matrix is added to the subspace. In contrast, in the FEAST algorithm, the dimension of the subspace is preserved. Instead, the iteration is performed such that the subspace initially generated by the span of K random vectors approaches the subspace spanned by the eigenvectors associated by eigenvalues enclosed by Γ by repeated application of an approximation of $\sum_{\lambda_j \in R} \mathbf{x_j x_j}^*$ constructed via a numerical contour integral followed by scaling with S.

The extension of the FEAST algorithm to eigenvalue problems for non-Hermitian problems is presented in [7]. Although the general intuition of constructing a transformation of the problem to a space spanned by the eigenvectors associated by eigenvalues enclosed by a contour, the problem is much more challenging due the loss of the restriction that eigenvalues must be real. As a result, in the most generic case, a contour with length scale r encloses $O(r^2)$ eigenvalues for a spectrum that appears uniformly distributed near some point $\lambda_0 \in \mathbb{C}$, instead of $O(r)$ in the Hermitian case. In this regime, not only does the already-sensitive problem become even more sensitive to initial conditions, we also require many more vectors $\{\mathbf{y}_k\}$ to capture the spectrum at a characteristic distance from a point λ_0.

References

[1] Tetsuya Sakurai and Hiroshi Sugiura, A projection method for generalized eigenvalue problems using numerical integration, *J. Comp. App. Math*, **159**, 119-128, (2003).

[2] Eric Polizzi, Density-matrix-based algorithm for solving eigenvalue problems, *Phys. Rev. B*, **79**, 115112, (2009).

[3] B. Zwiebach *Mastering Quantum Mechanics: Essentials, Theory, and Applications* (2022).

[4] A. R. Williams, Peter J. Feibelman, and N. D. Lang, Green's-function methods for electronic-structure calculations, *Phys. Rev. B*, **26**, 10, 5433-5444. (1982).

[5] P. Kravanja and M. Van Barel, A Derivative-Free Algorithm for Computing Zeros of Analytic Functions, *Computing*, **63**, 69-91, (1999).

[6] Ping Tak Peter Tang and Eric Polizzi, FEAST as a subspace iteration eigensolver accelerated by approximate spectral projection, *SIAM J. Matrix Anal. Appl.*, **35**, 2, 354-390 (2014).

[7] James Kestyn, Eric Polizzi, and Ping Tak Peter Tang, FEAST eigensolver for non-Hermitian problems, *SIAM J. Sci. Comput*, **38**, 5, S772-S799