Numer. Math.
DOI 10.1007/s00211-016-0837-7

Numerische Mathematik

Randomized estimation of spectral densities of large matrices made accurate

Lin Lin[1,2]

Received: 29 April 2015 / Revised: 15 July 2016

Abstract For a large Hermitian matrix $A \in \mathbb{C}^{N\times N}$, it is often the case that the only affordable operation is matrix–vector multiplication. In such case, randomized method is a powerful way to estimate the spectral density (or density of states) of A. However, randomized methods developed so far for estimating spectral densities only extract information from different random vectors independently, and the accuracy is therefore inherently limited to $\mathcal{O}(1/\sqrt{N_v})$ where N_v is the number of random vectors. In this paper we demonstrate that the "$\mathcal{O}(1/\sqrt{N_v})$ barrier" can be overcome by taking advantage of the correlated information of random vectors when properly filtered by polynomials of A. Our method uses the fact that the estimation of the spectral density essentially requires the computation of the trace of a series of matrix functions that are numerically low rank. By repeatedly applying A to the same set of random vectors and taking different linear combination of the results, we can sweep through the entire spectrum of A by building such low rank decomposition at different parts of the spectrum. Under some assumptions, we demonstrate that a robust and efficient implementation of such spectrum sweeping method can compute the spectral density accurately with $\mathcal{O}(N^2)$ computational cost and $\mathcal{O}(N)$ memory cost. Numerical results indicate that the new method can significantly outperform existing randomized methods in terms of accuracy. As an application, we demonstrate a way to accurately compute a trace of a smooth matrix function, by carefully balancing the smoothness of the integrand and the regularized density of states using a deconvolution procedure.

Mathematics Subject Classification 15A18 · 65F15

✉ Lin Lin
linlin@math.berkeley.edu

1 Department of Mathematics, University of California, Berkeley, Berkeley, CA 94720, USA

2 Computational Research Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA

1 Introduction

Given an $N \times N$ Hermitian matrix A, the *spectral density*, also commonly referred to as the *density of states* (DOS), is formally defined as

$$\phi(t) = \frac{1}{N} \sum_{i=1}^{N} \delta(t - \lambda_i). \tag{1}$$

Here δ is the Dirac distribution commonly referred to as the Dirac δ-"function" (see e.g. [1–3]), and the λ_i's are the eigenvalues of A, assumed here to be labeled non-decreasingly.

The DOS is an important quantity in many physics problems, in particular in quantum physics, and a large volume of numerical methods were developed by physicists and chemists [4–15] for this purpose. Besides being used as a qualitative visualization tool for understanding spectral characteristics of the matrix, the DOS can also be used to quantitatively compute the trace of a matrix function, as given in the formal formulation below

$$\mathrm{Tr}[f(A)] = \sum_{i=1}^{N} f(\lambda_i) \equiv N \int_{-\infty}^{\infty} f(t)\phi(t)\,\mathrm{d}t. \tag{2}$$

Here $f(t)$ is a smooth function, and the formal integral in Eq. (2) should be interpreted in the sense of distribution.

If one had access to all the eigenvalues of A, the task of computing the DOS would become a trivial one. However, in many applications, the dimension of A is large. The computation of its entire spectrum is prohibitively expensive, and a procedure that relies entirely on multiplications of A with vectors is the only viable approach. Fortunately, in many applications A only has $\mathcal{O}(N)$ nonzero entries, and therefore the cost of matrix–vector multiplication, denoted by c_{matvec}, is $\mathcal{O}(N)$. In some other cases the matrix is a dense matrix but fast matrix–vector multiplication method still exists with $\mathcal{O}(N \log^p N)$ cost, where p is "an integer" that is not too large. This is the case when the matrix–vector multiplication can be carried out effectively with fast algorithms, such as the fast Fourier transform (FFT), the fast multipole method (FMM) [16], the hierarchical matrix [17], and the fast butterfly algorithm [18], to name a few.

Rigorously speaking, the DOS is a distribution and cannot be directly approximated by smooth functions. In order to assess the accuracy of a given numerical scheme for estimating the DOS, the DOS must be properly *regularized*. The basic idea for estimating the DOS is to first expand the regularized DOS using simple functions such as polynomials. Then it can be shown that the estimation of the DOS can be obtained by computing the trace of a polynomial of A, which can then be estimated by repeatedly applying A to a set of random vectors. This procedure has been discovered more or less independently by statisticians [19] and by physicists and chemists [8,9], and will be referred to as Hutchinson's method in the following. In physics such method is often referred to as the kernel polynomial method (KPM) [10] with a few different variants. A recent review on the choice of regularization and different numerical methods for estimating the DOS is given in [20]. There are also a variety of randomized estimators that can be used in Hutchinson's method, and the quality of different estimators is analyzed in [21].

Contribution To the extent of our knowledge, all randomized methods so far for estimating the DOS are based on different variants of Hutchinson's method. These methods estimate the DOS by averaging the information obtained from N_v random vectors directly. The numerical error, when properly defined, decays asymptotically as $\mathcal{O}(1/\sqrt{N_v})$. As a result, high accuracy is difficult to achieve: every extra digit of accuracy requires increasing the number of random vectors by 100 fold.

In this work, we demonstrate that the accuracy for estimating the regularized DOS can be significantly improved by making use of the *correlated information* obtained among different random vectors. We use the fact that each point of the DOS can be evaluated as the trace of a numerically low rank matrix, and such trace can be evaluated by repeatedly applying A to a small number of random vectors, and by taking certain linear combination of the resulting vectors. If different set of random vectors were needed for different points on the spectrum the method will be prohibitively expensive. However, we demonstrate that it is possible to use *the same set of random vectors* to "sweep through" in principle the entire spectrum. Therefore we call our method a "spectrum sweeping method".

Our numerical results indicate that the spectrum sweeping method can significantly outperform Hutchinson type methods in terms of accuracy, as the number of random vectors N_v becomes large. However, the computational cost and the storage cost can still be large when the DOS needs to be evaluated at a large number of points. Furthermore, the accuracy of the spectrum sweeping method may be compromised when the right number of randomized vectors is not known a priori. We develop a robust and efficient implementation of the spectrum sweeping method to overcome these two problems. Under certain assumption on the distribution of eigenvalues of the matrix A, and the cost of the matrix–vector multiplication is $\mathcal{O}(N)$, we demonstrate that the computational cost of the new method scales as $\mathcal{O}(N^2)$ and the storage cost scales as $\mathcal{O}(N)$ for increasingly large matrix dimension N. We also demonstrate that the new method for evaluating the DOS can be useful for accurate trace estimation as in Eq. (2).

Other related works The spectrum sweeping method is not to be confused with another set of methods under the name of "spectrum slicing" methods [22–27]. The idea of the spectrum slicing methods is still to obtain a partial diagonalization of the matrix A. The main advantage of spectrum slicing methods is enhanced parallelism compared to conventional diagonalization methods. Due to the natural orthogonality of eigenvectors corresponding to distinct eigenvalues of a Hermitian matrix, the computational cost for each set of processors handling different parts of the spectrum can be reduced compared to direct diagonalization methods. However, the overall scaling for spectrum slicing methods is still $\mathcal{O}(N^3)$ when a large number of eigenvalues and eigenvectors are to be computed.

Notation In linear algebra notation, a vector $w \in \mathbb{C}^m$ is always treated as a column vector, and its conjugate transpose is denoted by w^*. For a randomized matrix $B \in \mathbb{C}^{m\times n}$, its entry-wise expectation value is denoted by $\mathbb{E}[B]$ and its entry-wise variance is denoted by $\mathrm{Var}[B]$. We call $B \in \mathbb{C}^{m\times n}$ a (real) random Gaussian matrix, if each entry of B is real, and follows independently the normal distribution $\mathcal{N}(0, 1)$. In the case when $n = 1$, B is called a (real) random Gaussian vector. The imaginary unit is denoted by $\imath$.

The paper is organized as follows. In Sect. 2 we introduce the DOS estimation problem. We also demonstrate the Delta–Gauss–Chebyshev (DGC) method, which is a variant of the kernel polynomial method, to estimate the DOS. We develop in Sect. 3 the spectrum sweeping method based on the randomized estimation of the trace of numerically low rank matrices, and demonstrate a robust and efficient implementation of the spectrum sweeping method in Sect. 4. We show how the DOS estimation method can be used to effectively compute the trace of a matrix function in Sect. 5. Following the numerical results in Sect. 6, we conclude and discuss the future work in Sect. 7.

2 Density of state estimation for large matrices

Without loss of generality, we shall assume that the spectrum of A is contained in the interval $(-1, 1)$. For a general matrix with spectrum contained in the interval (a, b), we can apply a spectral transformation

$$\widetilde{A} = \frac{2A - (a + b)I}{b - a}.$$

The spectrum of $\widetilde{A}$ is contained in $(-1, 1)$, and all the discussion below can be applied to $\widetilde{A}$. The fact that the spectral density $\phi(t)$ is defined in terms of Dirac δ-functions suggests that no smooth function can converge to the spectral density in the limit of high resolution, in the usual L^p norms $(p \geq 1)$ [20]. In order to compare different numerical approximations to the spectral density in a meaningful way, the DOS should be regularized. One simple method is to employ a Gaussian regularization

$$\phi_\sigma(t) = \sum_{i=1}^{N} g_\sigma(t - \lambda_i) = \mathrm{Tr}[g_\sigma(tI - A)]. \tag{3}$$

Here

$$g_\sigma(s) = \frac{1}{N\sqrt{2\pi\sigma^2}} e^{-\frac{s^2}{2\sigma^2}} \tag{4}$$

is a Gaussian function. In the following our goal is to compute the smeared DOS ϕ_σ.

The key of computing the DOS is the estimation of the trace of a matrix function without diagonalizing the matrix. To the extent of our knowledge, randomized methods developed so far are based on Hutchinson's method or its variants [19,21]. The following simple and yet useful theorem is a simple variant of Hutchinson's method and explains how the method works.

Theorem 1 (Hutchinson's method) *Let $A \in \mathbb{C}^{N\times N}$ be a Hermitian matrix, and $w \in \mathbb{R}^N$ be a random Gaussian vector, then*

$$\mathbb{E}[w^* A w] = \mathrm{Tr}[A], \quad \mathrm{Var}[w^* A w] = 2\sum_{i\neq j}(\mathrm{Re} A_{ij})^2. \tag{5}$$

Proof First

$$\mathbb{E}[w^* A w] = \mathbb{E}\left[\sum_{ij} w_i w_j A_{ij}\right] = \sum_i A_{ii} = \mathrm{Tr}[A].$$

Here we used that $\mathbb{E}[w] = 0, \quad \mathbb{E}[ww^*] = I$ and w is a real vector. This follows directly from that each entry of w is independently distributed and follows the Gaussian distribution $\mathcal{N}(0, 1)$.

Second,

$$\mathrm{Var}[w^* A w] = \mathbb{E}\left[(w^* A w)^2 - (\mathrm{Tr}[A])^2\right] = \mathbb{E}\left[\sum_{ijkl} w_i w_j w_k w_l A_{ij} A_{kl} - (\mathrm{Tr}[A])^2\right]$$
$$= \sum_{ik} A_{ii} A_{kk} + 2\sum_{i\neq j} (\mathrm{Re} A_{ij})^2 - \left(\sum_i A_{ii}\right)^2 = 2\sum_{i\neq j} (\mathrm{Re} A_{ij})^2.$$

□

Using Theorem 1, if we choose $W \in \mathbb{R}^{N\times N_v}$ to be a random Gaussian matrix, then

$$\mathrm{Tr}[g_\sigma(tI - A)] \approx \frac{1}{N_v} \mathrm{Tr}[W^* g_\sigma(tI - A) W].$$

In practice in order to compute $g_\sigma(A - tI)W$, we can expand $g_\sigma(A - tI)$ into polynomials of A for each t, and then evaluate the trace of polynomial of A. A stable and efficient implementation can be obtained by using Chebyshev polynomials. Other choices of polynomials such as Legendre polynomials can be constructed similarly. Using the Chebyshev polynomial, $g_\sigma(tI - A)$ is approximated by a polynomial of degree M as

$$g_\sigma(tI - A) \approx g_\sigma^M(tI - A) := \sum_{l=0}^{M} \mu_l(t) T_l(A). \tag{6}$$

The coefficients $\{\mu_l(t)\}$ need to be evaluated for each t. Since the Dirac δ-function is regularized using a Gaussian function, following the notion in [20] we refer to Eq. (6) as the "Delta–Gauss–Chebyshev" (DGC) expansion. Since $g_\sigma(t - \cdot)$ is a smooth function on $(-1, 1)$, the coefficient $\mu_l(t)$ in the DGC expansion can be computed as

$$\mu_l(t) = \frac{2 - \delta_{l0}}{\pi} \int_{-1}^{1} \frac{1}{\sqrt{1 - s^2}} g_\sigma(t - s) T_l(s)\, \mathrm{d}s. \tag{7}$$

Here δ_{l0} is the Kronecker δ symbol. With change of coordinate $s = \cos\theta$, and use the fact that $T_l(s) = \cos(l \arccos(s))$, we have

$$\mu_l(t) = \frac{2 - \delta_{l0}}{2\pi} \int_{0}^{2\pi} g_\sigma(t - \cos\theta) \cos(l\theta)\, \mathrm{d}\theta. \tag{8}$$

Eq. (8) can be evaluated by discretizing the interval $[0, 2\pi]$ using a uniform grid, and the resulting quadrature can be efficiently computed by the Fast Fourier Transform (FFT). This procedure is given in Algorithm 1, and this procedure is usually inexpensive.

Algorithm 1 Computing the Delta-Gauss-Chebyshev (DGC) polynomial expansion at a given point t.

```
Input:   Chebyshev polynomial degree $M$;
         Number of integration points $2N_\theta$, with $N_\theta > M$;

         Smooth function $g_\sigma(t - \cdot)$.
Output:  Chebyshev expansion coefficients $\{\mu_l(t)\}_{l=0}^{M}$.
1: Let $\theta_j = \frac{j\pi}{N_\theta}$,   $j = 0, \ldots, 2N_\theta - 1$.
2: $g_j = g_\sigma(t - \cos\theta_j)$.
3: Compute $\hat{g} = \mathcal{F}[g]$, where $\mathcal{F}$ is the discrete Fourier transform. Specifically
       $$\hat{g}_l = \sum_{j=0}^{2N_\theta-1} e^{-\frac{\imath 2\pi j l}{2N_\theta}} g_j.$$
4: $\mu_l(t) = \frac{2-\delta_{l0}}{2N_\theta} \mathrm{Re}\hat{g}_l$,   $l = 0, \ldots, M$.
```

Using the DGC expansion and Hutchinson's method, $\phi_\sigma(t)$ can be approximated by

$$\widetilde{\phi}_\sigma(t) := \mathrm{Tr}[g_\sigma^M(tI - A)] \approx \sum_{l=0}^{M} \mu_l(t) \frac{1}{N_v} \mathrm{Tr}[W^* T_l(A) W] \equiv \sum_{l=0}^{M} \mu_l(t) \zeta_l.$$

The resulting algorithm, referred to as the DGC algorithm in the following, is given in Algorithm 2.

We remark that the DGC algorithm can be viewed as a variant of the kernel polynomial method (KPM) [10]. The difference is that KPM *formally* expands the Dirac δ-function, which is not a well defined function but only a distribution. Therefore the accuracy of KPM cannot be properly measured until regularization is introduced [20]. On the other hand, DGC introduces a Gaussian regularization from the beginning, and the DGC expansion (6) is not a formal expansion, and its accuracy can be relatively easily analyzed.

Theorem 2 *Let $A \in \mathbb{C}^{N\times N}$ be a Hermitian matrix with spectrum in $(-1, 1)$. For any $t \in \mathbb{R}$, the error of a M-term DGC expansion (6) is*

$$\left|\mathrm{Tr}[g_\sigma(tI - A)] - \mathrm{Tr}[g_\sigma^M(tI - A)]\right| \le \frac{C_1}{\sigma}(1 + C_2\sigma)^{-M}, \tag{9}$$

where C_1, C_2 are constants independent of A as well as σ, M, t.

Algorithm 2 The Delta-Gauss-Chebyshev (DGC) method for estimating the DOS.

```
Input:   Hermitian matrix $A$ with eigenvalues between $(-1, 1)$;
         A set of points $\{t_i\}_{i=1}^{N_t}$ at which the DOS is to be evaluated;
         Polynomial degree $M$; Smearing parameter $\sigma$;
         Number of random vectors $N_v$.

Output:  Approximate DOS $\{\widetilde{\phi}_\sigma(t_i)\}_{i=1}^{N_t}$.

1: for each $t_i$ do
2:     Compute the coefficient $\{\mu_l(t_i)\}_{l=0}^{M}$ for each $t_i$, $i = 1, \ldots, N_t$ using Algorithm 1.
3: end for
4: Initialize $\zeta_k = 0$ for $k = 0, \cdots, M$.
5: Generate a random Gaussian matrix $W \in \mathbb{R}^{N \times N_v}$.
6: Initialize the three term recurrence matrices $V_m, V_p \leftarrow \mathbf{0} \in \mathbb{C}^{N \times N_v}$, $V_c \leftarrow W$.
7: for $l = 0, \ldots, M$ do
8:     Accumulate $\zeta_l \leftarrow \zeta_l + \frac{1}{N_v} \mathrm{Tr}[W^* V_c]$.
9:     $V_p \leftarrow (2 - \delta_{l0}) A V_c - V_m$.
10:    $V_m \leftarrow V_c$, $V_c \leftarrow V_p$.
11: end for
12: for $i = 1, \ldots, N_t$ do
13:    Compute $\widetilde{\phi}_\sigma(t_i) \leftarrow \sum_{l=0}^{M} \mu_l(t_i) \zeta_l$.
14: end for
```

Proof Note that

$$\begin{aligned}\left|\mathrm{Tr}[g_\sigma(tI - A)] - \mathrm{Tr}[g_\sigma^M(tI - A)]\right| &\le N \left\| g_\sigma(tI - A) - g_\sigma^M(tI - A) \right\|_2 \\ &\le \max_{-1 \le s \le 1} \left| N g_\sigma(t - s) - N g_\sigma^M(t - s) \right|.\end{aligned}$$

The estimate then follows from standard approximation theory with Chebyshev polynomials (see e.g. [28, Theorem 4.3]). □

We remark that Theorem 2 only gives an upper bound of the decay rate. Since the Gaussian function is analytic in the entire complex plane, the actual decay rate is super-exponential. Using the error bound in Theorem 2, the error of the DGC algorithm can be split into two parts: the error of the Chebyshev expansion (approximation error) and the error due to random sampling (sampling error). According to Theorem 2, it is sufficient to choose M to be $\mathcal{O}(\sigma^{-1} |\log \sigma|)$ to ensure that the error of the Chebyshev expansion is negligible. Therefore the error of the DGC mainly comes from the sampling error, which decays slowly as $\frac{1}{\sqrt{N_v}}$.

3 Spectrum sweeping method for estimating spectral densities

In this section we present an alternative randomized algorithm called the spectrum sweeping method for estimating spectral densities. Our numerical results indicate that the spectrum sweeping method can significantly outperform Hutchinson type methods in terms of accuracy, as the number of random vectors N_v becomes large. The main tool is randomized methods for low rank matrix decomposition which is briefly introduced in Sect. 3.1. The spectrum sweeping method is given in Sect. 3.2, and its complexity is analyzed in Sect. 3.3.

3.1 Randomized method for low rank decomposition of a numerically low rank matrix

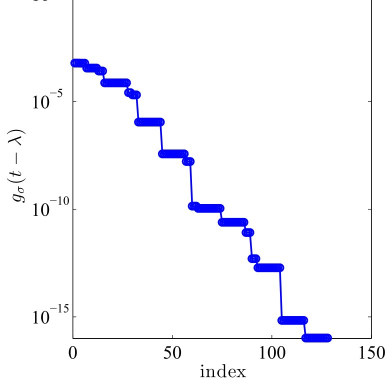
Consider a square matrix $P \in \mathbb{C}^{N\times N}$, and denote by r the rank of P. If $r \ll N$, then P is called a low rank matrix. Many matrices from scientific and engineering computations may not be exactly low rank but are close to be a low rank matrix. For such matrices, the concept of *numerical rank* or *approximate rank* can be introduced, defined by the closest matrix to P in the sense of the matrix 2-norm. More specifically, the numerical ε-rank of a matrix P, denoted by r_ε with respect to the tolerance $\varepsilon > 0$ is (see e.g. [29, section 5.4])

$$r_\varepsilon = \min\{\mathrm{rank}(Q) : Q \in \mathbb{C}^{N\times N}, \|P - Q\|_2 \le \varepsilon\}.$$

In the following discussion, we simply refer to a matrix P with numerical ε-rank r_ε as a matrix with numerical rank r. For instance, this applies to the function $g_\sigma(tI - A)$ with small σ, since the value $g_\sigma(t - \lambda_j)$ decays fast to 0 when λ_j is away from t, and the corresponding contribution to the rank of $g_\sigma(tI - A)$ can be neglected up to ε level. As an example, for the ModES3D_4 matrix to be detailed in Sect. 6, if we set $\sigma = 0.01$ and $t = 1.0$, then the values $\{g_\sigma(tI - \lambda)\}$ sorted in non-increasing order is given in Fig. 1, where each λ is an eigenvalue of A. If we set $\varepsilon = 10^{-8}$, then the ε-rank of $g_\sigma(tI - A)$ is 59, much smaller than the dimension of the matrix A which is 64000.

For a numerically low rank matrix, its approximate singular value decomposition can be efficiently evaluated using randomized algorithms (see e.g. [30–32]). The idea is briefly reviewed as below, though presented in a slightly non-standard way. Let $P \in \mathbb{C}^{N\times N}$ be a square matrix with numerical rank $r \ll N$, and $W \in \mathbb{R}^{N\times N_v}$ be a random Gaussian matrix. If N_v is larger than r by a small constant, then with high probability, P projected to the column space of PW is very close to P in matrix

Fig. 1 For the ModES3D_4 matrix, the values $g_\sigma(tI - \lambda)$ sorted in non-increasing order plotted in log scale with $\sigma = 0.01, t = 1.0$. Only the first 128 values larger than 10^{-16} are shown



2-norm. Similarly with high probability, P projected to the row space of W^*P is very close to P in matrix 2-norm. In the case when P is a Hermitian matrix, only the matrix–vector multiplication PW is needed.

In order to construct an approximate low rank decomposition of a Hermitian matrix P, let us denote by $Z = PW$, then an approximate low rank decomposition of P is given by

$$P \approx ZBZ^*. \tag{10}$$

The matrix B is to be determined and can be computed in several ways. One choice of B can be obtained by requiring Eq. (10) to hold when applying W^* and W to the both sides of the equation, i.e.

$$K_W := W^*PW = W^*Z \approx (W^*Z)B(W^*Z)^* = (W^*Z)B(W^*Z).$$

In the last equality we used that (W^*Z) is Hermitian. Hence one can choose

$$B = (W^*Z)^\dagger \equiv K_W^\dagger. \tag{11}$$

Here $K_W^\dagger$ is the Moore–Penrose pseudo-inverse (see e.g. [29, section 5.5]) of the matrix K_W. In Algorithm 3 we summarize the algorithm for constructing such a low rank decomposition.

Remark In the case when K_W is singular, the pseudo-inverse should be handled with care. This will be discussed in Sect. 3.2. Besides the choice in Eq. (11), another possible choice of the matrix B is given by requiring that Eq. (10) holds when applying Z^* and Z to the both sides of the equation, i.e.

$$Z^*PZ \approx (Z^*Z)B(Z^*Z),$$

and hence one can choose

$$B = (Z^*Z)^\dagger(Z^*PZ)(Z^*Z)^\dagger. \tag{12}$$

Note that Eq. (12) can also be derived from a minimization problem

$$\min_B \left\| ZBZ^* - P \right\|_F^2 .$$

However, the evaluation of Z^*PZ is slightly more difficult to compute, since the matrix–vector multiplication PZ needs to be further computed. In the discussion below we will adopt the choice of Eq. (11).

3.2 Spectrum sweeping method

The approximate low rank decomposition method can be used to estimate the DOS. Note that for each t, when the regularization parameter σ is small enough, the column

Algorithm 3 Randomized low rank decomposition algorithm.

```
Input:    Hermitian matrix $P \in \mathbb{C}^{N\times N}$ with approximate rank $r$;
Output:   Approximate low rank decomposition $P \approx ZBZ^*$.
1: Generate a random Gaussian matrix $W \in \mathbb{R}^{N\times N_v}$ where $N_v = r + c$ and $c$ is a small constant.
2: Compute $Z \leftarrow PW$.
3: Form $B = (W^*Z)^\dagger$.
```

space of $g_\sigma(tI - A)$ is approximately only spanned by eigenvectors of A corresponding to eigenvalues near t. Therefore for each t, $g_\sigma(tI - A)$ is approximately a low rank matrix. Algorithm 3 can be used to construct a low rank decomposition. Motivated from the DGC method, we can use the same random matrix W for all t.

$$g_\sigma(tI - A) \approx Z(t)(W^*Z(t))^\dagger Z^*(t),$$

and its trace can be accurately estimated as

$$\mathrm{Tr}[g_\sigma(tI - A)] \approx \mathrm{Tr}[(W^*Z(t))^\dagger (Z^*(t)Z(t))]. \tag{13}$$

Here $Z(t) = g_\sigma^M(tI - A)W$. We can use a Chebyshev expansion in Eq. (6) and compute $g_\sigma^M(tI - A)$.

The Chebyshev expansion requires the calculation of $T_l(A)W, \quad l = 0, \ldots, M$. Note that this does not mean that all $T_l(A)W$ need to be stored for all l. Instead we only need to accumulate $Z(t)$ for each point t that the DOS is to be evaluated. $T_l(A)$ only need to be applied to one random W matrix, and we can sweep through the spectrum of A just via different linear combination of all $T_l(A)W$ for each t. Therefore we refer to the algorithm a "spectrum sweeping" method.

As remarked earlier, the pseudo-inverse should be handled with care. There are two difficulties associated with the evaluation of $K_W^\dagger$. First, it is difficult to know a priori the exact number of vectors N_v that should be used at each t, and N_v should be chosen to be large enough to achieve an accurate estimation of the DOS. Hence the columns of Z are likely to be nearly linearly dependent, and K_W becomes singular. Second, although $g_\sigma(tI - A)$ is by definition a positive semidefinite matrix, the finite term Chebyshev approximation $g_\sigma^M(tI - A)$ may not be positive semidefinite due to the oscillating tail of the Chebyshev polynomial. Figure 2a gives an example of such possible failure. The test matrix is the ModES3D_1 matrix to be detailed in Sect. 6. The parameters are $\sigma = 0.05$, $N_v = 50$. When computing the pseudo-inverse, all negative eigenvalues and positive eigenvalues with magnitude less than 10^{-7} times the largest eigenvalue of K_W are discarded. Figure 2a demonstrates that when a relatively small number of degrees of polynomials $M = 400$ is used, the treatment of the pseudo-inverse may have large error near $t = 0.9$. This happens mainly when the degrees of Chebyshev polynomials M is not large enough. Figure 2b shows that when M is increased to 800, the accuracy of the pseudo-inverse treatment is much improved.

The possible difficulty of the direct evaluation of $K_W^\dagger$ can be understood as follows. First, Theorem 3 suggests that for a strictly low rank matrix P, there is correspondence

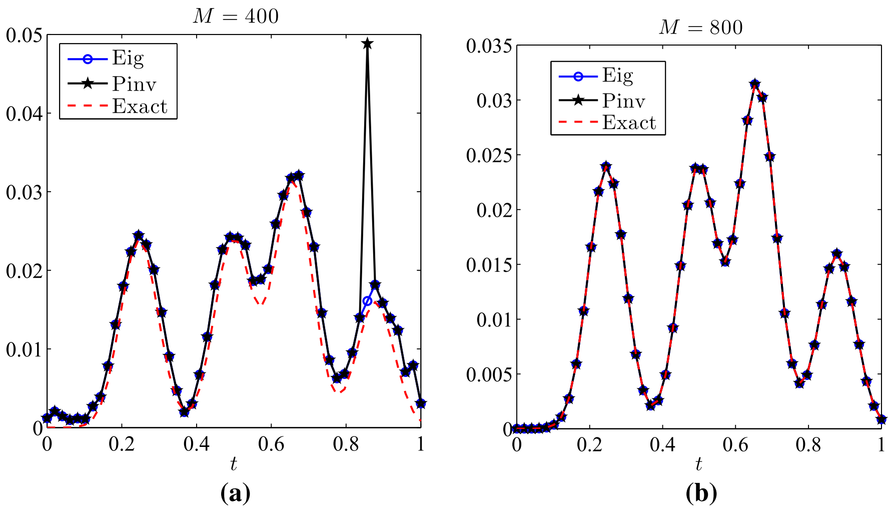


Fig. 2 For the ModES3D_1 matrix, compute the DOS using the spectrum sweeping method by computing the pseudo-inverse ("Pinv") and by computing the generalized eigenvalue problem ("Eig") using **a** a low degree Chebyshev polynomial $M = 400$ **b** a high degree Chebyshev polynomial $M = 800$

between the trace of the form in Eq. (13) and the solution of a generalized eigenvalue problem.

Theorem 3 *Let $P \in \mathbb{C}^{N\times N}$ be a Hermitian matrix with rank $r \ll N$ and with eigen decomposition*

$$P = USU^*.$$

Here $U \in \mathbb{C}^{N\times r}$ and $U^*U = I$. $S \in \mathbb{R}^{r\times r}$ is a real diagonal matrix containing the nonzero eigenvalues of P. For $W \in \mathbb{C}^{N\times p}$ $(p > r)$ and assume W^*U is a matrix with linearly independent columns, then S can be recovered through the generalized eigenvalue problem

$$(W^*P^2W)C = (W^*PW)C\Xi. \tag{14}$$

Here $C \in \mathbb{C}^{p\times r}$ and $\Xi \in \mathbb{R}^{r\times r}$ is a diagonal matrix with diagonal entries equal to those of S up to reordering. Furthermore,

$$\mathrm{Tr}\left[(W^*PW)^{\dagger}(W^*P^2W)\right] = \mathrm{Tr}[S] = \mathrm{Tr}[P]. \tag{15}$$

Proof Using the eigen decomposition of P,

$$(W^*P^2W)C = (W^*U)S^2(U^*WC), \quad (W^*PW)C\Xi = (W^*U)S(U^*WC)\Xi.$$

Since $W^*U \in \mathbb{C}^{p\times r}$ is a matrix with linearly independent columns, we have

$$S^2(U^*WC) = S(U^*WC)\Xi,$$

or equivalently

$$S(U^*WC) = (U^*WC)\Xi.$$

Since S is a diagonal matrix we have $S = \Xi$ up to reordering of diagonal entries.

To prove Eq. (15), note that

$$(W^*PW)^\dagger = (U^*W)^\dagger S^{-1}(W^*U)^\dagger.$$

Therefore

$$\begin{aligned}\mathrm{Tr}[(W^*PW)^\dagger(W^*P^2W)] &= \mathrm{Tr}[(U^*W)^\dagger S^{-1}(W^*U)^\dagger(W^*U)S^2(U^*W)]\\ &= \mathrm{Tr}[S] = \mathrm{Tr}[USU^*] = \mathrm{Tr}[P].\end{aligned}$$

□

Although Theorem 3 is stated for exactly low rank matrices, it provides an practical criterion for removing some of the large, spurious contribution to the DOS such as in Fig. 2a. For $g_\sigma(tI - A)$ which is numerically low rank, only the generalized eigenvalues within the range of g, i.e. the interval $[0, \frac{1}{N\sqrt{2\pi\sigma^2}}]$ should be selected. This motivated the use of Algorithm 4 to solve the generalized eigenvalue problem

$$Z^*Z\widetilde{C} = (W^*Z)\widetilde{C}\widetilde{\Xi}. \tag{16}$$

where $\widetilde{\Xi}$ is a diagonal matrix only containing the generalized eigenvalues in the interval $[0, \frac{1}{N\sqrt{2\pi\sigma^2}}]$. Algorithm 4 only keeps the generalized eigenvalues within the possible range of g_σ. Our numerical experience indicates that this procedure is more stable than the direct treatment of the pseudo-inverse. The algorithm of the spectrum sweeping method is given in Algorithm 5.

Now consider the problematic point when using the pseudo-inverse in Fig. 2a. We find that the generalized eigenvalues Ξ at that problematic point has one generalized eigenvalue 0.033, exceeding the maximally allowed range $\frac{1}{N\sqrt{2\pi\sigma^2}} = 0.008$. After removing this generalized eigenpair, the error of the DOS obtained by solving the generalized eigenvalue problem becomes smaller. Again when the degree of the Chebyshev polynomial M increases sufficiently large to 800, the accuracy of the generalized eigenvalue formulation also improves, and the result obtained by using the pseudo-inverse and by using the generalized eigenvalue problem agrees with each other, as illustrated in Fig. 2b. In such case, all generalized eigenvalues fall into the range $[0, \frac{1}{N\sqrt{2\pi\sigma^2}}]$.

The SS-DGC method can be significantly more accurate compared to the DGC method. This is because the spectrum sweeping method takes advantage that different columns of $Z \equiv g_\sigma(tI - A)W$ are correlated: The information in different columns of Z saturates as N_v increases beyond the numerical rank of $g_\sigma(tI - A)$, and the columns of Z become increasingly linearly dependent. Comparatively Hutchinson's method neglects such correlated information, and the asymptotic convergence rate is only $\mathcal{O}(N_v^{-\frac{1}{2}})$.

Algorithm 4 Solve the generalized eigenvalue problem for the spectrum sweeping method.

```
Input:   Matrices $Z, W \in \mathbb{C}^{N \times N_v}$;
         Smearing parameter $\sigma$;
         Truncation parameter $\tau$.
Output:  Generalized eigenvalues $\widetilde{\widetilde{\Xi}}$ and generalized eigenvectors $\widetilde{C}$.

1: Compute the eigenvalue decomposition of the matrix
       $$W^* Z = USU^*.$$
   $S$ is a diagonal matrix with diagonal entries $\{s_i\}$.
2: Let $\widetilde{S}$ be a diagonal matrix with all eigenvalues $s_j \geq \tau \max_i s_i$, and $\widetilde{U}$ be the corresponding eigenvectors.
3: Solve the standard eigenvalue problem
       $$\widetilde{S}^{-\frac{1}{2}} \widetilde{U}^* Z^* Z \widetilde{U} \widetilde{S}^{-\frac{1}{2}} X = X \Xi.$$
   $\Xi$ is a diagonal matrix with diagonal entries $\{\xi_i\}$.
4: Let $\widetilde{\widetilde{\Xi}}$ be a diagonal matrix containing the generalized eigenvalues $\xi_i \in [0, \frac{1}{N\sqrt{2\pi\sigma^2}}]$, and $\widetilde{X}$ be the corresponding eigenvectors.
5: Compute the generalized eigenvectors $\widetilde{C} = \widetilde{U} \widetilde{S}^{-\frac{1}{2}} \widetilde{X}$.
```

Algorithm 5 Spectrum sweeping method using the Delta-Gauss-Chebyshev expansion (SS-DGC) for estimating the DOS.

```
Input:   Hermitian matrix $A$ with eigenvalues between $(-1, 1)$;
         A set of points $\{t_i\}_{i=1}^{N_t}$ at which the DOS is to be evaluated;
         Polynomial degree $M$; Smearing parameter $\sigma$;
         Number of random vectors $N_v$.
Output:  Approximate DOS $\{\widetilde{\phi}_\sigma(t_i)\}$.

1: Compute the coefficient $\{\mu_l(t_i)\}_{l=0}^{M}$ for each $t_i$, $i = 1, \ldots, N_t$ using Algorithm 1 with $f(x) = g_\sigma(x - t_i)$.
2: Generate a random Gaussian matrix $W \in \mathbb{R}^{N \times N_v}$.
3: Initialize the three term recurrence matrices $V_m, V_p \leftarrow \mathbf{0} \in \mathbb{C}^{N \times N_v}$, $V_c \leftarrow W$.
4: Initialize $Z(t_i) \leftarrow \mathbf{0} \in \mathbb{C}^{N \times N_v}$, $i = 1, \ldots, N_t$.
5: for $l = 0, \ldots, M$ do
6:     for $i = 1, \ldots, N_t$ do
7:         $Z(t_i) \leftarrow Z(t_i) + \mu_l(t_i) V_c$.
8:     end for
9:     $V_p \leftarrow (2 - \delta_{l0}) A V_c - V_m$.
10:    $V_m \leftarrow V_c$, $V_c \leftarrow V_p$.
11: end for
12: for $i = 1, \ldots, N_t$ do
13:     Compute $\widetilde{\phi}_\sigma(t_i) = \mathrm{Tr}[\widetilde{\widetilde{\Xi}}(t_i)]$ where $\widetilde{\widetilde{\Xi}}(t_i)$ is a diagonal matrix obtained by solving the generalized eigenvalue problem
        $$Z^*(t_i) Z(t_i) \widetilde{C}(t_i) = W^*(t_i) Z(t_i) \widetilde{C}(t_i) \widetilde{\widetilde{\Xi}}(t_i)$$
    using Algorithm 4.
14: end for
```

3.3 Complexity

The complexity of the DOS estimation is certainly problem dependent. In order to measure the asymptotic complexity of Algorithm 5 for a series of matrices with increasing value of N, we consider a series of matrices are *spectrally uniformly distributed*, i.e. the spectral width of each matrix is bounded between $(-1, 1)$, and the number of eigenvalues in any interval (t_1, t_2) is proportional to N. In other words, we do not consider the case when the eigenvalues can asymptotically be concentrated into one point. In the complexity analysis below, we neglect any contribution on the order of $\log N$. In Sect. 4 we will give a detailed example for which the assumption is approximately satisfied.

Algorithm 5 scales as $\mathcal{O}(N_v^3)$ with respect to the number of random vectors N_v. Therefore it can be less efficient to let N_v grow proportionally to the matrix size N. Instead it is possible to choose N_v to be a constant, and to choose the regularization parameter σ to be $\mathcal{O}(N^{-1})$ so that $g_\sigma(tI - A)$ is a matrix of bounded numerical rank as N increases. Equation (9) then states that the Chebyshev polynomial degree M should be $\mathcal{O}(N)$. We denote by c_{matvec} the cost of each matrix–vector multiplication (matvec). We assume $c_{\text{matvec}} \sim \mathcal{O}(N)$. We also assume that N_v is kept to be a constant and is omitted in the asymptotic complexity count with respect to N and N_t.

Under the assumption of spectrally uniformly distributed matrices, the computational cost for applying the Chebyshev polynomial to the random matrix W is $c_{\text{matvec}} M N_v \sim \mathcal{O}(N^2)$. The cost for updating all $\{Z(t_i)\}$ is $N_t M N N_v \sim \mathcal{O}(N^2 N_t)$. The cost for computing the DOS by trace estimation is $\mathcal{O}(N_t N N_v^2 + N_t N_v^3) \sim \mathcal{O}(N_t N)$. So the total cost is $\mathcal{O}(N^2 N_t)$.

The memory cost is dominated by the storage of the matrices $\{Z(t_i)\}$, which scales as $\mathcal{O}(N N_t N_v) \sim \mathcal{O}(N N_t)$.

If the DOS is evaluated at a few points with N_t being small, the spectrum sweeping method is very efficient. However, in some cases such as the trace estimation as discussed in Sect. 5, N_t should be chosen to be $\mathcal{O}(N)$. Therefore the computational cost of SS-DGC is $\mathcal{O}(N^3)$ and the memory cost is $\mathcal{O}(N^2)$. This is undesirable and can be improved as in the next section with a more efficient implementation.

4 A robust and efficient implementation of the spectrum sweeping method

The SS-DGC method can give very accurate estimation of the DOS. However, it also has two notable disadvantages:

1. The SS-DGC method requires a rough estimate of the random vectors N_v. If the number of random vectors N_v is less than the numerical rank of $g_\sigma(tI - A)$, then the estimated DOS will has $\mathcal{O}(1)$ error.
2. The SS-DGC method requires the formation of the $Z(t)$ matrix for each point t. The computational cost scales as $\mathcal{O}(N^2 N_t)$ and the memory cost is $\mathcal{O}(N N_t)$. This is expensive if the number of points N_t is large.

In this section we provide a more robust and efficient implementation of the spectrum sweeping (RESS) method to overcome the two problems above. The main idea of the RESS-DGC method is to have

1. a hybrid strategy for robust estimation of the DOS in the case when the number of vectors N_v is insufficient at some points with at least $\mathcal{O}(1/\sqrt{N_v})$ accuracy.
2. a consistent method for directly computing of the matrix $Z^*(t_i)Z(t_i)$ for each point t_i and avoiding the computation and storage of $Z(t_i)$.

4.1 A robust and efficient implementation for estimating the trace of a numerically low rank matrix

Given a numerically low rank matrix P, we may apply Algorithm 3 to compute its low rank approximation using a random matrix of size $N \times N_v$. Let us denote the residual by

$$R := P - ZBZ^*. \tag{17}$$

If N_v is large enough, then $\|R\|_F$ should be very close to zero. Otherwise, Hutchinson's method can be used to estimate the trace of R as the correction for the trace of P. According to Theorem 1, if ZBZ^* is relatively a good approximation to P, the variance for estimating $\mathrm{Tr}[R]$ can be significantly reduced.

To do this, we use another set of random vectors $\widetilde{W} \in \mathbb{C}^{N\times\widetilde{N}_v}$, and

$$\mathrm{Tr}[R] \approx \frac{1}{\widetilde{N}_v}\mathrm{Tr}[\widetilde{W}^* R \widetilde{W}] = \frac{1}{\widetilde{N}_v}\left(\mathrm{Tr}[\widetilde{W}^* P \widetilde{W}] - \mathrm{Tr}[(\widetilde{W}^* Z)B(Z^*\widetilde{W})]\right). \tag{18}$$

Equation (18) still requires the storage of Z. As explained above, storing Z can become expensive if we use the same set of random vectors W and $\widetilde{W}$ but evaluate the DOS at a large number of points $\{t_i\}$. In order to reduce such cost, note that

$$Z^*Z = W^*P^*PZ = W^*(P^2W).$$

If we can compute both PW and P^2W, then Z does not need to be explicitly stored. Instead we only need to store

$$K_W = W^*(PW), \quad K_Z = W^*(P^2W).$$

This observation is used in Sect. 4.2, where PW and P^2W are computed separately with Chebyshev expansion.

Similarly, for the computation of the correction term (18), we can directly compute the cross term due to $\widetilde{W}$ as

$$K_C = \widetilde{W}^*Z = \widetilde{W}^*(PW), \quad K_{\widetilde{W}} = \widetilde{W}^*(P\widetilde{W}).$$

Algorithm 3 involves the computation of the pseudo-inverse of K_W, which is in practice computed by solving a generalized eigenvalue problem using Algorithm 4.

The hybrid strategy in Eq. (18) is consistent with the generalized eigenvalue problem, in the sense that

$$\mathbb{E}_{\widetilde{w}}[\widetilde{w}^* Z\widetilde{C}\widetilde{C}^* Z^* \widetilde{w}] = \mathrm{Tr}[Z\widetilde{C}\widetilde{C}^* Z^*] = \mathrm{Tr}[\widetilde{C}^*(Z^* Z)\widetilde{C}] = \mathrm{Tr}[\widetilde{\Xi}]. \tag{19}$$

The last equality of Eq. (19) follows from that $\widetilde{\Xi}$, $\widetilde{C}$ solve the generalized eigenvalue problem as in Algorithm 4.

In summary, a robust and efficient randomized method for estimating the trace of a low rank matrix is given in Algorithm 6.

Algorithm 6 Robust and efficient randomized method for estimating the trace of a numerically low rank matrix.

```
Input:   Hermitian matrix $P \in \mathbb{C}^{N\times N}$; Number of randomized vectors $N_v$, $\widetilde{N}_v$
Output:  Estimated $\mathrm{Tr}[P]$.
1: Generate random Gaussian matrices $W \in \mathbb{R}^{N\times N_v}$ and $\widetilde{W} \in \mathbb{C}^{N\times \widetilde{N}_v}$.
2: Compute $K_W = W^*(PW)$,   $K_Z = W^*(P^2 W)$.
3: Compute $K_C = \widetilde{W}^*(PW)$, $K_{\widetilde{W}} = \widetilde{W}^*(P\widetilde{W})$.
4: Solve the generalized eigenvalue problem
       $K_Z\widetilde{C} = K_W\widetilde{C}\widetilde{\Xi}$,
   using Algorithm 4.
5: Compute the trace
       $\mathrm{Tr}[P] \approx \mathrm{Tr}[\widetilde{\Xi}] + \frac{1}{\widetilde{N}_v}\left(\mathrm{Tr}[K_{\widetilde{W}}] - \mathrm{Tr}[K_C\widetilde{C}\widetilde{C}^* K_C^*]\right)$.
```

4.2 Robust and efficient implementation of the spectrum sweeping method

In order to combine Algorithm 6 and Algorithm 5 to obtain a robust and efficient implementation of the spectrum sweeping method, it is necessary to evaluate $(g_\sigma(tI - A))^2 W$, where

$$(g_\sigma(tI - A))^2 \equiv \frac{1}{N^2 2\pi\sigma^2} e^{-\frac{(tI-A)^2}{\sigma^2}}.$$

In order to do this, one can directly compute $(g_\sigma(tI - A))^2$ using an auxiliary Chebyshev expansion as follows

$$(g_\sigma(tI - A))^2 \approx \sum_{l=0}^{M} \nu_l(t) T_l(A). \tag{20}$$

Proposition 4 states that if the expansion is chosen carefully, the Chebyshev expansion (6) and (20) are fully consistent, i.e. if g_σ is expanded by a Chebyshev polynomial expansion of degree $M/2$ denoted by $g_\sigma^{M/2}$, we can expand $(g_\sigma)^2$ using a Chebyshev polynomial expansion of degree M denoted by $\widetilde{g}_\sigma^M$. These two expansions are consistent in the sense that $(g_\sigma^{M/2})^2 = \widetilde{g}_\sigma^M$.

Proposition 4 *Let M be an even integer, and P is a matrix polynomial function of A*

$$P = \sum_{l=0}^{M/2} \mu_l T_l(A).$$

Then

$$P^2 = \sum_{l=0}^{M} \nu_l T_l(A), \tag{21}$$

where

$$\nu_l = \frac{2-\delta_{l0}}{\pi} \int_{-1}^{1} \frac{1}{\sqrt{1-x^2}} T_l(x) \left(\sum_{k=0}^{M/2} \mu_k T_k(x) \right)^2 \mathrm{d}x, \quad l = 0, \ldots, M.$$

Proof Using the definition of P,

$$P^2 = \sum_{p,q=0}^{M/2} \mu_p \mu_q T_p(A) T_q(A).$$

Since $0 \le p+q \le M$, P^2 is a polynomial of A up to degree M, and can be expanded using a Chebyshev polynomial of the form (21). □

The expansion coefficient ν_l's can be obtained using Algorithm 1 with numerical integration, and we have a consistent and efficient method for estimating the DOS.

Theorem 5 *Let $A \in \mathbb{C}^{N\times N}$ be a Hermitian matrix, and $W \in \mathbb{R}^{N\times N_v}$ be a random Gaussian matrix. For any $t \in (-1, 1)$, let $g_\sigma^{M/2}(tI - A)$ be the $M/2$ degree Chebyshev expansion*

$$g_\sigma^{M/2}(tI - A) = \sum_{l=0}^{M/2} \mu_l(t) T_l(A), \tag{22}$$

and the coefficients $\{\nu_l\}_{l=0}^M$ are defined according to Proposition 4. Define $Z(t) = g_\sigma^{M/2}(tI - A)W$, and

$$K_W(t) = W^* Z(t), \quad K_Z(t) = W^* \left(\sum_{l=0}^{M} \nu_l(t) T_l(A) W \right). \tag{23}$$

Let $\widetilde{w}$ be a random Gaussian vector, and

$$K_C(t) = \widetilde{w}^* Z(t), \quad K_{\widetilde{W}}(t) = \widetilde{w}^* g_\sigma^{M/2}(tI - A)\widetilde{w}. \tag{24}$$

Then

$$\mathrm{Tr}\left[g_\sigma^{M/2}(tI - A)\right] = \mathbb{E}_{\widetilde{w}}\left(K_{\widetilde{W}}(t) - K_C(t)\widetilde{C}(t)\widetilde{C}^*(t)K_C^*(t)\right) + \mathrm{Tr}[\widetilde{\Xi}(t)]. \quad (25)$$

Here $\widetilde{C}(t)$, $\widetilde{\Xi}(t)$ *solves the generalized eigenvalue problem*

$$K_Z(t)\widetilde{C}(t) = K_W(t)\widetilde{C}(t)\widetilde{\Xi}(t), \quad (26)$$

using Algorithm 4.

Proof By Proposition 4, $(g_\sigma^{M/2}(tI - A))^2$ can be exactly computed using a Chebyshev of degree M with coefficients $\{\nu_l\}_{l=0}^M$. Therefore

$$K_Z(t) = Z^*(t)Z(t) = W^*\left(\sum_{l=0}^{M} \nu_l(t)T_l(A)W\right).$$

The consistency between Hutchinson's method and the estimation of $\mathrm{Tr}[\widetilde{\Xi}(t)]$ follows from Eq. (19). $\square$

Following Theorem 5, the RESS-DGC algorithm is given in Algorithm 7. Compared to DGC and SS-DGC method, the RESS-DGC method introduces an additional parameter $\widetilde{N}_v$. In the numerical experiments, in order to carry out a fair in the sense the *total* number of random vectors used are the same, i.e. for RESS-DGC, we always choose $N_v + \widetilde{N}_v$ to be the same number of random vectors $N_v^{\mathrm{DGC}} = N_v^{\mathrm{SS-DGC}}$ used in DGC and SS-DGC, respectively. For instance, when N_v^{DGC} is relatively small, in RESS-DGC we can choose $N_v = \widetilde{N}_v = \frac{1}{2}N_v^{\mathrm{DGC}}$, i.e. half of the random vectors are used for low rank approximation, and half for hybrid correction. When we are certain that N_v is large enough, we can eliminate the usage of the hybrid correction and take $N_v = N_v^{\mathrm{DGC}}$ and $\widetilde{N}_v = 0$.

4.3 Complexity

Following the same setup in Sect. 4.3, we assume that a series of matrices are spectrally uniformly distributed. We assume the Chebyshev polynomial degree $M \sim \mathcal{O}(N)$, and $c_{\mathrm{matvec}} \sim \mathcal{O}(N)$. In the complexity analysis below, we neglect any contribution on the order of $\log N$. We also assume that N_v is kept as a constant and is omitted in the asymptotic complexity count with respect to N and N_t.

In terms of the computational cost, the computational cost for applying the Chebyshev polynomial to the random matrix W is $c_{\mathrm{matvec}}MN_v \sim \mathcal{O}(N^2)$. The cost for updating $K_W(t_i)$ and $K_Z(t_i)$ is $N_tMN_v^2 \sim \mathcal{O}(NN_t)$. The cost for computing the DOS by trace estimation for each t_i is $\mathcal{O}(N_tN_v^3) \sim \mathcal{O}(N_t)$. So the total cost is $\mathcal{O}(NN_t+N^2)$.

In terms of the memory cost, the advantage of using Algorithm 7 also becomes clear here. The matrices $\{Z(t_i)\}$ do not need to be computed or stored. Using the three-term recurrence for Chebyshev polynomials, the matrices $K_W(t)$, $K_Z(t)$, $K_C(t)$, $K_{\widetilde{W}}(t)$ can be updated gradually, and the cost for storing these matrices are $\mathcal{O}(N_v^2N_t \sim N_t)$.

Algorithm 7 Robust and efficient spectrum sweeping with Delta-Gauss-Chebyshev (RESS-DGC) method for estimating the DOS.

```
Input:    Hermitian matrix $A$ with eigenvalues between $(-1, 1)$;
          A set of points $\{t_i\}_{i=1}^{N_t}$ at which the DOS is to be evaluated;
          Polynomial degree $M$;
          Smearing parameter $\sigma$;
          Number of random vectors $N_v, \widetilde{N}_v$.
Output:   Approximate DOS $\{\widetilde{\phi}_\sigma(t_i)\}$.

1: Compute the coefficient $\{\mu_l(t_i)\}_{l=0}^{\frac{M}{2}}$ for each $t_i, i = 1, \ldots, N_t$ using Algorithm 1 for Eq. (6), and let
   $\mu_l(t_i) = 0, \quad l = \frac{M}{2} + 1, \ldots, M$.
2: Compute the coefficient $\{\nu_l(t_i)\}_{l=0}^{M}$ for each $t_i, i = 1, \ldots, N_t$ using Algorithm 1 for Eq. (20).
3: Generate random Gaussian matrices $W \in \mathbb{R}^{N \times N_v}$ and $\widetilde{W} \in \mathbb{R}^{N \times \widetilde{N}_v}$.
4: Initialize the three term recurrence matrices $V_m$,
   $V_p \leftarrow \mathbf{0} \in \mathbb{C}^{N \times N_v}, V_c \leftarrow W; \widetilde{V}_m, \widetilde{V}_p \leftarrow \mathbf{0} \in \mathbb{C}^{N \times \widetilde{N}_v}, \widetilde{V}_c \leftarrow \widetilde{W}$.
5: Initialize matrices $K_W(t_i), K_Z(t_i) \leftarrow \mathbf{0} \in \mathbb{C}^{N_v \times N_v}, \quad i = 1, \ldots, N_t; K_C(t_i) \leftarrow \mathbf{0} \in \mathbb{C}^{\widetilde{N}_v \times N_v}$ and
   $K_{\widetilde{W}} \leftarrow \mathbf{0} \in \mathbb{C}^{\widetilde{N}_v \times \widetilde{N}_v}, \quad i = 1, \ldots, N_t$.
6: for $l = 0, \ldots, M$ do
7:    Compute $X_W \leftarrow W^* V_c, \quad X_C \leftarrow \widetilde{W}^* V_c, \quad X_{\widetilde{W}} \leftarrow \widetilde{W}^* \widetilde{V}_c$.
8:    for $i = 1, \ldots, N_t$ do
9:       Accumulate $K_W(t_i) \leftarrow K_W(t_i) + \mu_l(t_i) X_W, \quad K_Z(t_i) \leftarrow K_Z(t_i) + \nu_l(t_i) X_W$.
10:      Accumulate $K_C(t_i) \leftarrow K_C(t_i) + \mu_l(t_i) X_C, \quad K_{\widetilde{W}}(t_i) \leftarrow K_{\widetilde{W}}(t_i) + \mu_l(t_i) X_{\widetilde{W}}$.
11:   end for
12:   $V_p \leftarrow (2 - \delta_{l0}) A V_c - V_m$.
13:   $V_m \leftarrow V_c, V_c \leftarrow V_p$.
14:   $\widetilde{V}_p \leftarrow (2 - \delta_{l0}) A \widetilde{V}_c - \widetilde{V}_m$.
15:   $\widetilde{V}_m \leftarrow \widetilde{V}_c, \widetilde{V}_c \leftarrow \widetilde{V}_p$.
16: end for
17: for $i = 1, \ldots, N_t$ do
18:   Compute $\widetilde{\phi}_\sigma(t_i) \leftarrow \mathrm{Tr}\left[g_\sigma^{\frac{M}{2}}(t_i I - A)\right] \approx \frac{1}{\widetilde{N}_v} \mathrm{Tr}\left[K_{\widetilde{W}}(t_i) - K_C(t_i)\widetilde{C}(t_i)\widetilde{C}^*(t_i)K_C^*(t_i)\right] +$
      $\mathrm{Tr}[\widetilde{\Xi}(t_i)]$, where $\widetilde{\Xi}(t_i)$,$\widetilde{C}(t_i)$ are computed from Algorithm 4.
19: end for
```

The cost for storing the matrix W is $\mathcal{O}(NN_v) \sim \mathcal{O}(N)$. So the total storage cost is $\mathcal{O}(N + N_t)$.

If we also assume $N_t \sim \mathcal{O}(N)$ due to the vanishing choice of σ, then the computational cost scales as $\mathcal{O}(N^2)$ and the storage cost scales as $\mathcal{O}(N)$.

5 Application to trace estimation of general matrix functions

As an application for the accurate estimation of the DOS, we consider the problem of estimating the trace of a smooth matrix function as in Eq. (2). In general $f(t)$ is not a localized function on the real axis, and Algorithm 6 based on low rank decomposition cannot be directly used to estimate $\mathrm{Tr}[f(A)]$.

However, if we assume that there exists $\sigma > 0$ so that the Fourier transform of $f(t)$ decays faster than the Fourier transform of $g_\sigma(t)$, where g_σ is a Gaussian function, then we can find a smooth function $\widetilde{f}(t)$ satisfying

$$(\widetilde{f} * g_\sigma)(t) = \int_{-\infty}^{\infty} \widetilde{f}(s) g_\sigma(t-s)\,\mathrm{d}s = f(t). \tag{27}$$

The function $\widetilde{f}(t)$ can be obtained via a deconvolution procedure. Formally

$$\begin{aligned}\frac{1}{N}\operatorname{Tr}[f(A)] &= \int_{-\infty}^{\infty} f(t)\phi(t)\,\mathrm{d}t = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \widetilde{f}(s) g_\sigma(t-s)\phi(t)\,\mathrm{d}t\,\mathrm{d}s\\ &= \int_{-\infty}^{\infty} \widetilde{f}(s)\phi_\sigma(s)\,\mathrm{d}s.\end{aligned} \tag{28}$$

Equation (28) states that the trace of the matrix function $f(A)$ can be accurately computed from the integral of $\widetilde{f}(s)\phi_\sigma(s)$, which is a now smooth function. Since the spectrum of A is assumed to be in the interval $(-1, 1)$, the integration range in Eq. (28) can be replaced by a finite interval. The integral can be evaluated accurately via a trapezoidal rule, and we only need the value of the DOS ϕ_σ evaluated on the points requested by the quadrature. In such a way, we "transfer" the smoothness of $f(t)$ to the regularized DOS without losing accuracy.

The deconvolution procedure (27) can be performed via a Fourier transform. Assume that the eigenvalues of A are further contained in the interval $(-a, a) \subset (-1, 1)$ with $0 < a < 1$. Then the Fourier transform requires that the function $f(t)$ is a periodic function on a interval containing $(-a, a)$, which is in general not satisfied in practice. However, note that the interval in Eq. (2) does not require the exact function $f(t)$. In fact

$$\operatorname{Tr}[f(A)] = \int_{-a}^{a} f(t)\phi(t)\,\mathrm{d}t = \int_{-a}^{a} h(t)\phi(t)\,\mathrm{d}t$$

for any smooth function $h(t)$ satisfying

$$h(t) = f(t), \quad t \in (-a, a). \tag{29}$$

In particular, $h(t)$ can be extended to be a periodic function on the interval $[-1, 1]$. In this work, we construct $h(t)$ as follows.

$$h(t) = f(t)\pi(t) + \frac{f(-1) + f(1)}{2}\left(1 - \pi(t)\right). \tag{30}$$

We remark that the constant $\frac{f(-1)+f(1)}{2}$ in Eq. (30) is not important and can be in principle any real number. Here $\pi(t)$ is a function with $\pi(t) = 1$ for $t \in (-a, a)$, and smoothly goes to 0 outside $(-a, a)$. It is easy to verify that such choice of $h(t)$ satisfies Eq. (29) and is periodic on $(-1, 1)$. There are many choice of $\pi(t)$, and here we use

$$\pi(t) = \frac{1}{2}\left[\operatorname{erf}\left(\frac{1 + a - 2t}{\widetilde{\sigma}}\right) - \operatorname{erf}\left(\frac{-1 - a - 2t}{\widetilde{\sigma}}\right)\right]. \tag{31}$$

Here erf is the error function. If $\widetilde{\sigma}$ is chosen to be small enough, then $\pi(t)$ as defined in Eq. (31) satisfies the requirement.

Algorithm 8 describes the procedure for computing the trace of a matrix function.

Algorithm 8 Spectrum sweeping method for estimating the trace of a matrix function.

```
Input:   Hermitian matrix $A$ with eigenvalues between $(-a, a)$ where $0 < a < 1$;
         Smooth function $f(t)$;
         Smearing parameter $\sigma, \widetilde{\sigma}$.
Output:  Estimated value of $\mathrm{Tr}[f(A)]$.
1: Compute auxiliary function $h(t)$ according to Eq. (30).
2: Compute $\{\widetilde{f}(t_i)\}_{i=1}^{N_t}$ satisfying $(\widetilde{f} * g_\sigma)(t) = h(t)$ on $(-1, 1)$ through the Fourier transform on a
   uniform set of points $\{t_i\}_{i=1}^{N_t}$ with spacing $\Delta t = t_2 - t_1$.
3: Use one of the algorithms to compute $\{\widetilde{\phi}_\sigma(t_i)\}$.
4: Compute $\mathrm{Tr}[f(A)] \approx N\Delta t \sum_{i=1}^{N_t} \widetilde{f}(t_i)\widetilde{\phi}_\sigma(t_i)$.
```

6 Numerical examples

In this section we demonstrate the accuracy and efficiency of the SS-DGC and RESS-DGC methods for computing the spectral densities and for estimating the trace. For the asymptotic scaling of the method, we need a series of matrices that are approximately spectrally uniformly distributed. These are given by the ModES3D_X matrices to be detailed below. In order to demonstrate that the methods are also applicable to general matrices, we also give test results for two matrices obtained from the University of Florida matrix collection [33]. All the computation is performed on a single computational thread of an Intel i7 CPU processor with 64 gigabytes (GB) of memory using MATLAB.

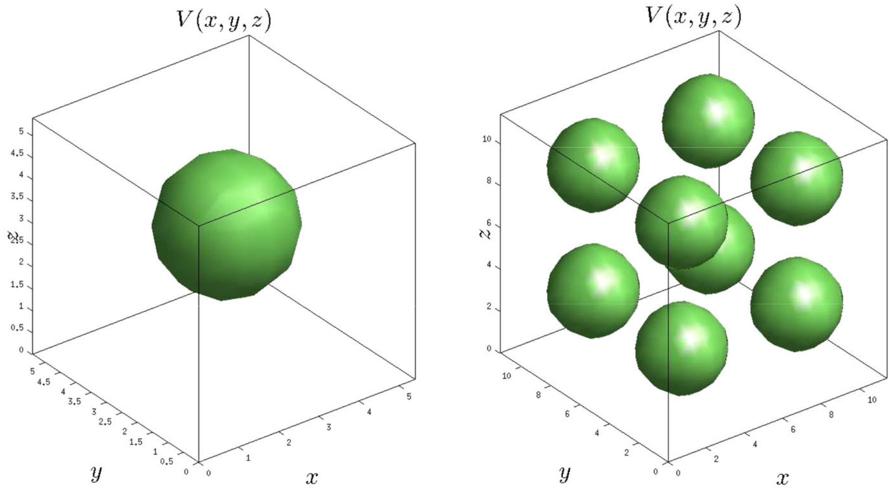
As a model problem, we consider a second order partial differential operator $\hat{A}$ in a three-dimensional (3D) cubic domain with periodic boundary conditions. For a smooth function $u(x, y, z)$, $\hat{A}u$ is given by

$$(\hat{A}u)(x, y, z) = -\Delta u(x, y, z) + V(x, y, z)u(x, y, z).$$

The matrix A is obtained from a 7-point finite difference discretization of $\hat{A}$. In order to create a series of matrices, first we consider one cubic domain $\Omega = [0, L]^3$ and $V(x, y, z)$ is taken to be a Gaussian function centered at $(L/2, L/2, L/2)^T$. This is called a "unit cell". The unit cell is then extended by n times along the x, y, z directions, respectively, and the resulting domain is $[0, nL]^3$ and $V(x, y, z)$ is the linear combination of n^3 Gaussian functions. Such matrix can be interpreted as a model matrix for electronic structure calculation. Each dimension of the domain is uniformly discretized with grid spacing h, and the resulting matrix A is denoted by ModES3D_X where X is the total number of unit cells. Here we take $L = 6$ and $h = 0.6$. Some characteristics of the matrices, including the dimension, the smallest

Table 1 Characteristics of the test matrices

Matrix	N	Min(ev)	Max(ev)
ModES3D_1	1000	-2.22	32.23
ModES3D_8	8000	-2.71	31.31
ModES3D_27	27000	-2.75	31.30
ModES3D_64	64000	-2.76	32.30
pe3k	9000	8×10^{-6}	127.60
shwater	81920	5.79	20.30



(a) ModES3D_1 **(b)** ModES3D_8

Fig. 3 Isosurface for $V(x, y, z)$

and the largest eigenvalue are given in Table 1. Figure 3a, b shows the isosurface of one example of such potential for the matrix ModES3D_1, and ModES3D_8, respectively. Figure 4 shows the DOS corresponding to low lying eigenvalues for the matrices with $X = 1, 8, 27, 64$ for a fixed regularization parameter $\sigma = 0.02$, which indicate that the spectral densities is roughly uniform.

6.1 Spectral densities

In order to compare with the accuracy of the DOS, the exact DOS is obtained by solving eigenvalues corresponding to the region of interest for computing the DOS. We use the locally optimal block preconditioned conjugate gradient (LOBPCG) [34] method. The LOBPCG method is advantageous for solving a large number of eigenvalues and eigenvectors since it can effectively take advantage of the BLAS3 operations by solving all eigenvectors simultaneously. The number of eigenvectors to be computed is set to be $35X$ for the test matrices ModES3D_X with $X = 1, 8, 27, 64$, respectively,

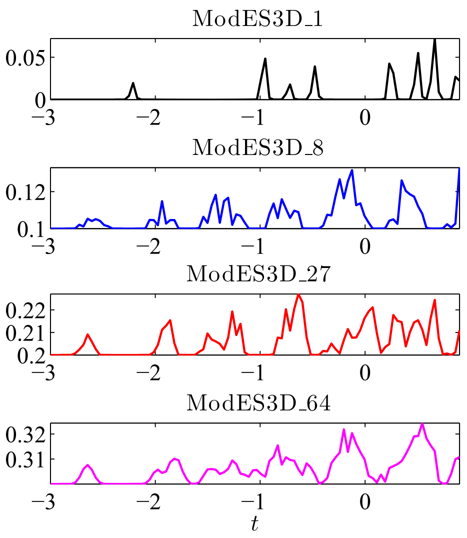


Fig. 4 Shape of the DOS for a series of matrices with $\sigma = 0.02$

and the highest eigenvalue obtained with such number of eigenvectors is slightly larger than 1.0. The tolerance of LOBPCG is set to be 10^{-8} and the maximum number of iterations is set to be 1000.

We measure the error of the approximate DOS using the relative L^1 error defined as follows. Denote by $\widetilde{\phi}_\sigma(t_i)$ the approximate DOS evaluated at a series of uniformly distributed points t_i, and by $\phi_\sigma(t_i)$ the exact DOS obtained from the eigenvalues. Then the error is defined as

$$\text{Error} = \frac{\sum_i \left|\widetilde{\phi}_\sigma(t_i) - \phi_\sigma(t_i)\right|}{\sum_i |\phi_\sigma(t_i)|}. \tag{32}$$

For the DGC and SS-DGC method, an M-th order Chebyshev polynomial is used to evaluate $Z(t_i)$. For the RESS-DGC method, an M-th order Chebyshev polynomial should be used to expand $Z^*(t_i)Z(t_i)$. Following Theorem 5, only an $M/2$-th order Chebyshev polynomial can be used to evaluate $Z(t_i)$. Similarly when N_v random vectors are used for DGC and SS-DGC, RESS-DGC is a hybrid method containing two terms. As discussed in Sect. 4.2, the number of random vectors used in DGC and SS-DGC are the same i.e. $N_v^{\text{DGC}} = N_v^{\text{SS-DGC}}$. For RESS-DGC, we use $N_v^{\text{RESS-DG}} = \frac{1}{2}N_v^{\text{DGC}}$ for the low rank approximation, and $\widetilde{N}_v^{\text{RESS-DGC}} = \frac{1}{2}N_v^{\text{DGC}}$ for the hybrid correction. This setup makes sure that all methods perform *exactly the same* number of matrix–vector multiplications.

Figure 5a shows the error of the three methods for the ModES3D_8 matrix when a very small number of random vectors $N_v = 40$ is used, with increasing polynomial degrees M from 200 to 3200. Here we use $\sigma = 0.05$. Note the relatively large polynomial degree is mainly due to the relatively large spectral radius compared to the desired resolution as in Table 1. This can be typical in practical applications. DGC is slightly more accurate for low degree of polynomials, but as M increases, RESS-DGC becomes more accurate. Figure 5b shows the error when a relatively large number of random vectors $N_v = 160$ is used. When M is large enough, both SS-DGC and RESS-

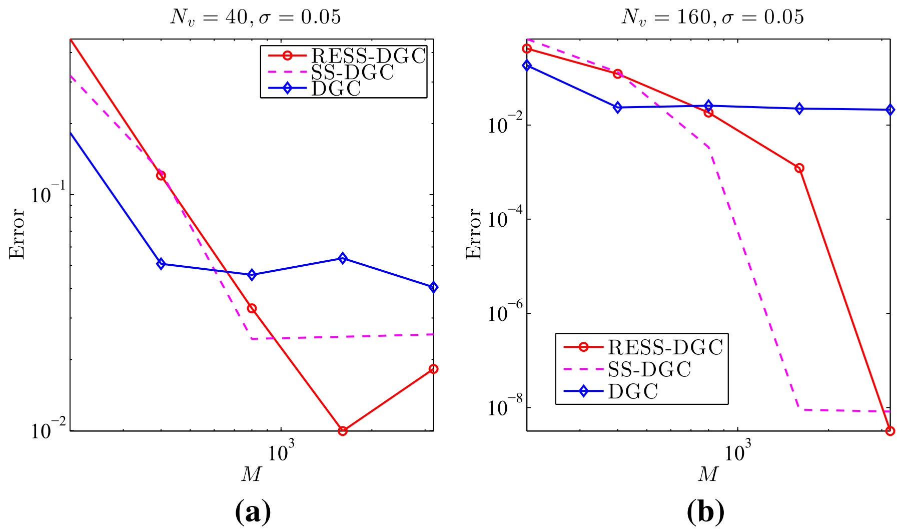


Fig. 5 For the ModES3D_8 matrix, the error of the DOS with respect to increasing polynomial degrees for **a** a small number of random vectors $N_v = 40$, **b** a large number of random vectors $N_v = 160$

DGC can be significantly more accurate than DGC. SS-DGC is slightly more accurate here, because it uses the Chebyshev polynomials and random vectors more optimally than RESS-DGC, though the computational cost can be higher when spectral densities at a large number of points N_t need to be evaluated.

Figure 6a shows the comparison of the accuracy of three methods for a relatively low degree of polynomials $M = 800$ and with increasing number of random vectors N_v. When N_v is small, SS-DGC has $\mathcal{O}(1)$ error, and this error is much suppressed in RESS-DGC thanks to the hybrid correction scheme. It is interesting to see that RESS-DGC outperforms DGC for all choices of N_v, but its accuracy is eventually limited by the insufficient number of Chebyshev polynomials to expand the Gaussian function. SS-DGC is more accurate than RESS-DGC when N_v is large enough. This is because in such case the low rank decomposition captures the correlation between the results obtained among different random vectors more efficiently. On the contrary, Hutchinson's method for which DGC relies on only reduces the error only through direct Monte Carlo sampling. Figure 6b shows the case when a relatively large number of polynomials $M = 2400$ is used. Again for small N_v, RESS-DGC reduces the large error compared to the SS-DGC method, while for large enough N_v both SS-DGC and RESS-DGC can be very accurate.

In both SS-DGC and RESS-DGC methods, the parameter σ is important since it determines both the degrees of Chebyshev polynomial to accurately expand g_σ, and the number of random vectors needed for accurate low rank approximation. Figure 7 shows the error of DGC, SS-DGC and RESS-DGC with σ varying from 0.02 to 0.1. Here we choose $M = 120/\sigma$ and $N_v = 3200\sigma$. This corresponds to the case when a relatively high degrees of Chebyshev polynomial and a relatively large number of random vectors are used in the previous discussion. We observe that the scaling

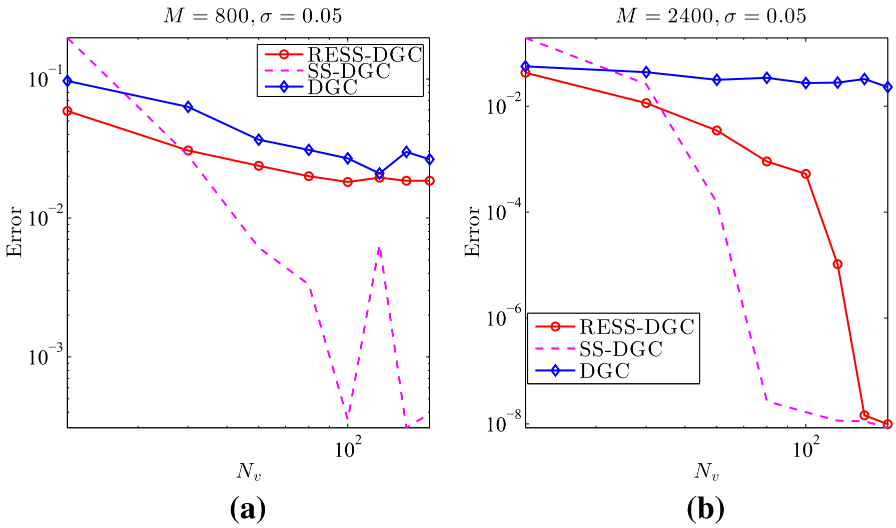


Fig. 6 For the ModES3D_8 matrix, the error of the DOS with respect to increasing random vectors for **a** a low polynomial degree $M = 800$, **b** a high polynomial degree $M = 2400$

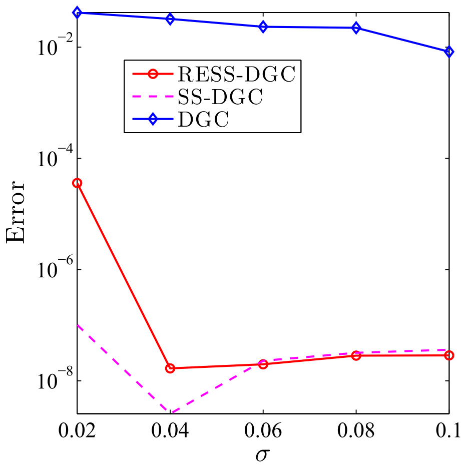


Fig. 7 For the ModES3D_8 matrix, the error of the DOS with respect to different choices of σ

$M \sim \mathcal{O}(\sigma^{-1})$ and $N_v \sim \mathcal{O}(\sigma)$ is important for spectrum sweeping type methods to be accurate, and SS-DGC and RESS-DGC can significantly outperform DGC type methods in terms of accuracy.

In order to study the weak scalability of the methods using the ModES3D_X matrices, as given in the complexity analysis, the polynomial degrees M should be chosen to be proportional to X. Here $M = 300X$ for $X = 1, 8, 27, 64$, respectively. Correspondingly $\sigma = 0.4/X$ and $N_t = 5X$. This allows us to use the same number of random vectors $N_v = 150$ for all matrices. Figure 8 shows the wall clock time of the three methods. Both SS-DGC and LOBPCG are asymptotically $\mathcal{O}(N^3)$ methods with respect to the increase of the matrix size, and the cubic scaling becomes apparent from

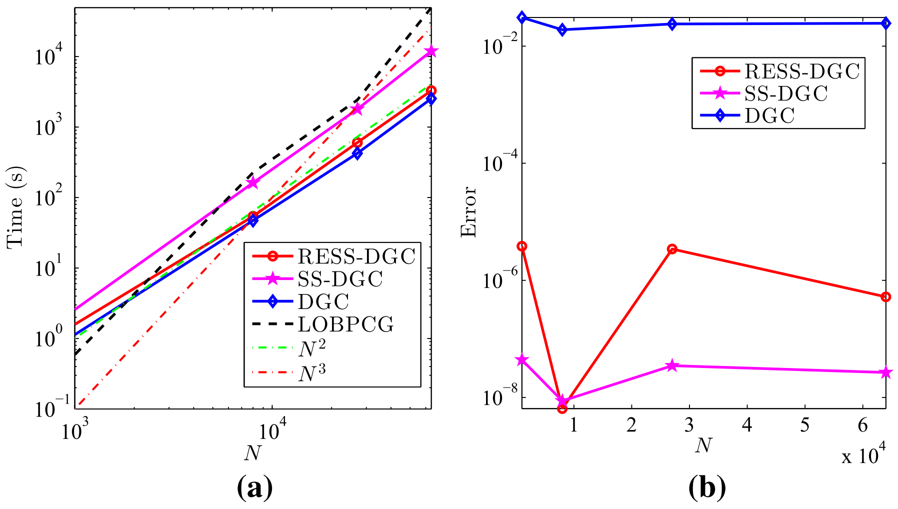


Fig. 8 **a** Wall clock time of the DGC, SS-DGC and RESS-DGC methods compared with diagonalization method using LOBPCG. **b** The error of the DOS computed at RESS-DGC and the DGC method

$X = 27$ to $X = 64$. RESS-DGC is only slightly more expensive than DGC and scales as $\mathcal{O}(N^2)$. For ModES3D_64, the wall clock time for DGC, RESS-DGC, SS-DGC and LOBPCG is 2535, 3293, 11979, 49389 seconds, respectively. Here RESS-DGC is 15 times faster than LOBPCG and is the most effective method. Figure 8b shows the accuracy in terms of the relative L^1 error. Both SS-DGC and RESS-DGC can be significantly more accurate than DGC. We remark that since N_v is large enough in all cases here, the efficiency of RESS-DGC can be further improved by noting that it only effectively uses half of the random vectors here, due to the small contribution from the other half of random vectors used for the hybrid correction.

6.2 Trace estimation

As discussed in Sect. 5, the accurate calculation of the regularized DOS can be used for trace estimation. To demonstrate this, we use the same ModES3D_X matrices, and let $f(A)$ be the Fermi–Dirac function, i.e.

$$\mathrm{Tr}[f(A)] = \mathrm{Tr}\left[\frac{1}{1+\exp(\beta(A-\mu I)}\right]$$

is to be computed. In electronic structure calculation, μ has the physical meaning of chemical potential, and β is the inverse temperature. The trace of the Fermi–Dirac distribution has the physical meaning of the number of electrons at chemical potential μ. Here $\beta = 10.0$, $\mu = -1.0$. The value of σ that can be used for the deconvolution procedure in Eq. (28) should be chosen such that after deconvolution $\widetilde{f}(s)$ is still a smooth function. Here we use $\sigma = 0.05$ for $X = 1$, and $\sigma = 0.4/X$ for $X = 8, 27, 64$, respectively. The value of $\widetilde{\sigma}$ for the smearing function in Eq. (31) is 0.016.

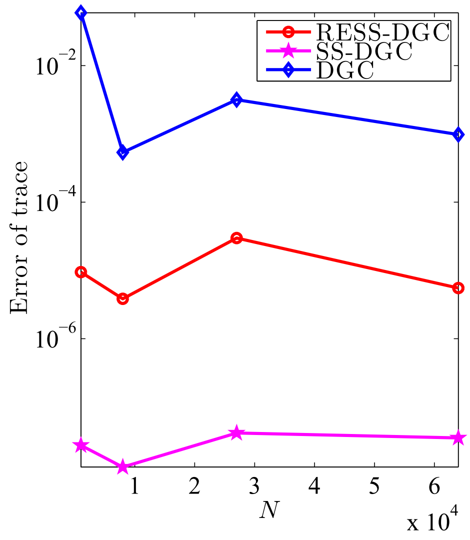


Fig. 9 Relative error for estimating the trace of Fermi–Dirac functions applied to ModES3D_X matrices

Correspondingly the polynomial degree $M = 300X$. The number of random vectors N_v is kept to be 100 for all calculations.

Figure 9 shows the relative error of the trace for DGC, SS-DGC and RESS-DGC. We observe that SS-DGC and RESS-DGC can be significantly more accurate compared to DGC, due to the better use of the correlated information obtained among different random vectors. Again when N_v is sufficiently large, SS-DGC is more accurate since the hybrid strategy in RESS-DGC is no longer needed here.

6.3 Other matrices

In Sect. 6.1, we verified that both SS-DGC and RESS-DGC can obtain very accurate estimation of the DOS when the degrees of Chebyshev polynomial and the number of random vectors are large enough, and RESS-DGC can lead to more efficient implementation. In this section we further verify that RESS-DGC can achieve high accuracy for other test matrices, when the degrees of polynomial and number of vectors N_v is large enough. The test matrices pe3k and shwater are obtained from the University of Florida matrix collection [33], and are used as test matrices in [20]. The character of the matrices is given in Table 1.

Figure 10 shows the DOS obtained from RESS-DGC for the pe3k matrix, compared to the DOS obtained by diagonalizing the matrix directly ("Exact"). The parameters are $\sigma = 0.25$, $M = 4084$, $N_v = 300$, $N_t = 100$. Since the goal is to demonstrate high accuracy, we turn off the hybrid mode in the RESS-DGC method by setting $\widetilde{N}_v = 0$. Figure 10 shows that the error of RESS-DGC is less than 10^{-7} everywhere. The relatively large error occurs at the two peaks of the DOS, which agrees with the theoretical estimate that RESS-DGC needs more random vectors when the spectral density is large. Similarly Fig. 11 shows the same comparison for the shwater matrix, with $\sigma = 0.005$, $M = 16240$, $N_v = 640$, $\widetilde{N}_v = 0$, and $N_t = 100$. More detailed

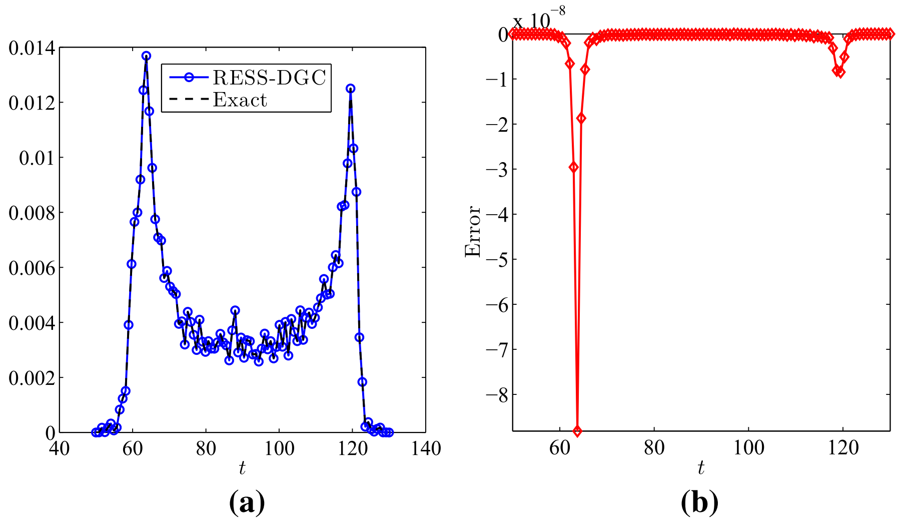


Fig. 10 For the pe3k matrix, **a** numerically computed DOS by RESS-DGC, compared to the exact DOS, and **b** the error of the DOS computed by RESS-DGC

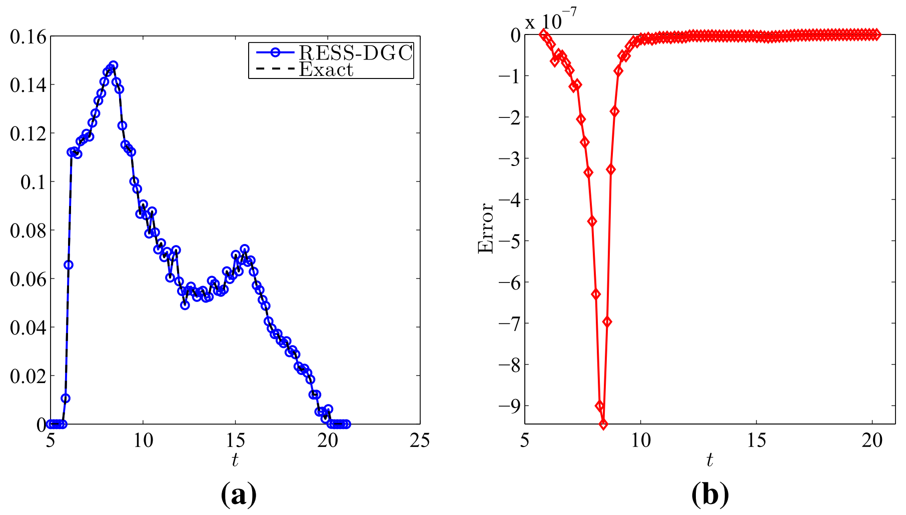


Fig. 11 For the shwater matrix, **a** numerically computed DOS by RESS-DGC, compared to the exact DOS, and **b** the error of the DOS computed by RESS-DGC

comparison of the error and running time for the two matrices with increasing number of random vectors N_v is given in Tables 2 and 3, respectively. We observe that the error of RESS-DGC rapidly decreases with respect to the increase of the number of random vectors, while the error of DGC only decreases marginally. We find that RESS-DGC only introduces marginally extra cost compared to the cost of the DGC method.

Table 2 Error of the DOS estimation for RESS-DGC and DGC with different numbers of random vectors N_v

Matrix	N_v	Error RESS-DGC	Error DGC
pe3k	100	4.2×10^{-2}	1.7×10^{-2}
pe3k	200	4.0×10^{-4}	1.3×10^{-2}
pe3k	300	4.8×10^{-7}	1.1×10^{-2}
shwater	320	9.6×10^{-3}	5.7×10^{-3}
shwater	480	1.2×10^{-4}	4.8×10^{-3}
shwater	640	9.8×10^{-7}	3.7×10^{-3}

Table 3 Running time of the DOS estimation for RESS-DGC and DGC with different numbers of random vectors N_v

Matrix	N_v	Time RESS-DGC (s)	Time DGC (s)
pe3k	100	780	764
pe3k	200	1691	1576
pe3k	300	2825	2720
shwater	320	7371	5513
shwater	480	12310	9487
shwater	640	23479	18495

7 Conclusion and future work

For large Hermitian matrices that the only affordable operation is matrix–vector multiplication, randomized algorithms can be an effective way for obtaining a rough estimate the DOS. However, so far randomized algorithms are based on Hutchinson's method, which does not use the correlated information among different random vectors. The accuracy is inherently limited to $\mathcal{O}(1/\sqrt{N_v})$ where N_v is the number of random vectors.

We demonstrate that randomized low rank decomposition can be used as a different mechanism to estimate the DOS. By properly taking into account the correlated information among the random vectors, we develop a spectrum sweeping (SS) method that can sweep through the spectrum with a reasonably small number of random vectors and the accuracy can be substantially improved compared to $\mathcal{O}(1/\sqrt{N_v})$. However, For spectrally uniformly distributed matrices with a large number of points to evaluate the DOS, the direct implementation of the spectrum sweeping method can have $\mathcal{O}(N^3)$ complexity. We also present a robust and efficient implementation of the spectrum sweeping method (RESS). For spectrally uniformly distributed matrices, the complexity for obtaining the DOS can be improved to $\mathcal{O}(N^2)$, and the extra robustness comes from a hybridization with Hutchinson's method for estimating the residual.

We demonstrate how the regularized DOS can be used for estimating the trace of a smooth matrix function. This is based on a careful balance between the smoothness of the function and that of the DOS. Such balance is implemented through a deconvolution procedure. Numerical results indicate that this allows the accurate estimate of the trace with again $\mathcal{O}(N^2)$ scaling.

The current implementation of the spectrum sweeping method is based on Chebyshev polynomials. Motivated from the discussion in [20], Lanczos method would be more efficient than Chebyshev polynomials for estimating the DOS, and it would be interesting to extend the idea of spectrum sweeping to Lanczos method and compare with Chebyshev polynomials. We also remark that the spectrum sweeping method effectively builds a low rank decomposition near each point on the spectrum for which the DOS is to be evaluated. Combining the deconvolution procedure as in the trace estimation and the low rank decomposition could be potentially useful in some other applications to directly estimate the whole or part of a matrix function. For instance, in electronic structure calculation, the diagonal entries of the Fermi–Dirac function is needed to evaluate the electron density. These directions will be explored in the future.

Acknowledgments This work is partially supported by the Scientific Discovery through Advanced Computing (SciDAC) program and the Center for Applied Mathematics for Energy Research Applications (CAMERA) funded by U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research and Basic Energy Sciences, and by the Alfred P. Sloan fellowship. We thank the anonymous referees for their comments that greatly improved the paper.

References

1. Schwartz, L.: Mathematics for the Physical Sciences. Dover, New York (1966)
2. Byron, F.W., Fuller, R.W.: Mathematics of Classical and Quantum Physics. Dover, New York (1992)
3. Richtmyer, R.D., Beiglböck, W.: Principles of Advanced Mathematical Physics, vol. 1. Springer, New York (1981)
4. Ducastelle, F., Cyrot-Lackmann, F.: Moments developments and their application to the electronic charge distribution of d bands. J. Phys. Chem. Solids **31**, 1295–1306 (1970)
5. Turek, I.: A maximum-entropy approach to the density of states within the recursion method. J. Phys. C **21**, 3251 (1988)
6. Drabold, D.A., Sankey, O.F.: Maximum entropy approach for linear scaling in the electronic structure problem. Phys. Rev. Lett. **70**, 3631–3634 (1993)
7. Wheeler, J.C., Blumstein, C.: Modified moments for harmonic solids. Phys. Rev. B **6**, 4380–4382 (1972)
8. Silver, R.N., Röder, H.: Densities of states of mega-dimensional Hamiltonian matrices. Int. J. Mod. Phys. C **5**, 735–753 (1994)
9. Wang, L.-W.: Calculating the density of states and optical-absorption spectra of large quantum systems by the plane-wave moments method. Phys. Rev. B **49**, 10154 (1994)
10. Weiße, A., Wellein, G., Alvermann, A., Fehske, H.: The kernel polynomial method. Rev. Mod. Phys. **78**, 275–306 (2006)
11. Covaci, L., Peeters, F.M., Berciu, M.: Efficient numerical approach to inhomogeneous superconductivity: the Chebyshev-Bogoliubov–de Gennes method. Phys. Rev. Lett. **105**, 167006 (2010)
12. Jung, D., Czycholl, G., Kettemann, S.: Finite size scaling of the typical density of states of disordered systems within the kernel polynomial method. Int. J. Mod. Phys. Conf. Ser. **11**, 108 (2012)
13. Seiser, B., Pettifor, D.G., Drautz, R.: Analytic bond-order potential expansion of recursion-based methods. Phys. Rev. B **87**, 094105 (2013)
14. Haydock, R., Heine, V., Kelly, M.J.: Electronic structure based on the local atomic environment for tight-binding bands. J. Phys. C: Solid State Phys. **5**, 2845 (1972)
15. Parker, G.A., Zhu, W., Huang, Y., Hoffman, D., Kouri, D.J.: Matrix pseudo-spectroscopy: iterative calculation of matrix eigenvalues and eigenvectors of large matrices using a polynomial expansion of the Dirac delta function. Comput. Phys. Commun. **96**, 27–35 (1996)
16. Greengard, L., Rokhlin, V.: A fast algorithm for particle simulations. J. Comput. Phys. **73**, 325–348 (1987)
17. Hackbusch, W.: A sparse matrix arithmetic based on $\mathcal{H}$-matrices. Part I: Introduction to $\mathcal{H}$-matrices. Computing **62**, 89–108 (1999)

18. Candès, E., Demanet, L., Ying, L.: A fast butterfly algorithm for the computation of fourier integral operators. SIAM Multiscale Model. Simul. **7**(4), 1727–1750 (2009)
19. Hutchinson, M.F.: A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. Commun. Stat. Simul. Comput. **18**, 1059–1076 (1989)
20. Lin, L., Saad, Y., Yang, C.: Approximating spectral densities of large matrices. SIAM Rev. **58**, 34 (2016)
21. Avron, H., Toledo, S.: Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix. J. ACM **58**, 8 (2011)
22. Parlett, B.N.: The Symmetric Eigenvalue Problem, vol. 7. SIAM, Englewood Cliffs (1980)
23. Sakurai, T., Sugiura, H.: A projection method for generalized eigenvalue problems. J. Comput. Appl. Math. **159**, 119–128 (2003)
24. Polizzi, E.: Density-matrix-based algorithm for solving eigenvalue problems. Phys. Rev. B **79**, 115112–115117 (2009)
25. Schofield, G., Chelikowsky, J.R., Saad, Y.: A spectrum slicing method for the Kohn–Sham problem. Comput. Phys. Commun. **183**, 497–505 (2012)
26. Fang, H.-R., Saad, Y.: A filtered Lanczos procedure for extreme and interior eigenvalue problems. SIAM J. Sci. Comput. **34**, A2220–A2246 (2012)
27. Aktulga, H.M., Lin, L., Haine, C., Ng, E.G., Yang, C.: Parallel eigenvalue calculation based on multiple shift-invert Lanczos and contour integral based spectral projection method. Parallel Comput. **40**, 195–212 (2014)
28. Trefethen, L.N.: Is Gauss quadrature better than Clenshaw–Curtis? SIAM Rev. **50**, 67–87 (2008)
29. Golub, G.H., Van Loan, C.F.: Matrix Computations, 4th edn. Johns Hopkins University Press, Baltimore (2013)
30. Liberty, E., Woolfe, F., Martinsson, P., Rokhlin, V., Tygert, M.: Randomized algorithms for the low-rank approximation of matrices. Proc. Natl. Acad. Sci. USA **104**, 20167–20172 (2007)
31. Woolfe, F., Liberty, E., Rokhlin, V., Tygert, M.: A fast randomized algorithm for the approximation of matrices. Appl. Comput. Harmon. Anal. **25**(3), 335–366 (2008)
32. Halko, N., Martinsson, P.-G., Tropp, J.A.: Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. SIAM Rev. **53**(2), 217–288 (2011)
33. Davis, T.A., Hu, Y.: The University of Florida sparse matrix collection. ACM Trans. Math. Softw. **38**, 1 (2011)
34. Knyazev, A.V.: Toward the optimal preconditioned eigensolver: locally optimal block preconditioned conjugate gradient method. SIAM J. Sci. Comput. **23**, 517–541 (2001)