Applications of Trace Estimation Techniques

Shashanka Ubaru(✉) and Yousef Saad

University of Minnesota at Twin Cities, Minneapolis, MN 55455, USA
{ubaru001,saad}@umn.edu

Abstract. We discuss various applications of trace estimation techniques for evaluating functions of the form $\mathtt{tr}(f(A))$ where f is certain function. The first problem we consider that can be cast in this form is that of approximating the *Spectral density* or *Density of States* (DOS) of a matrix. The DOS is a probability density distribution that measures the likelihood of finding eigenvalues of the matrix at a given point on the real line, and it is an important function in solid state physics. We also present a few non-standard applications of spectral densities. Other trace estimation problems we discuss include estimating the trace of a matrix inverse $\mathtt{tr}(A^{-1})$, the problem of counting eigenvalues and estimating the rank, and approximating the log-determinant (trace of log function). We also discuss a few similar computations that arise in machine learning applications. We review two computationally inexpensive methods to compute traces of matrix functions, namely, the Chebyshev expansion and the Lanczos Quadrature methods. A few numerical examples are presented to illustrate the performances of these methods in different applications.

1 Introduction

Let $A \in \mathbb{R}^{n\times n}$ be a symmetric (real) matrix with an eigen-decomposition $A = U\Lambda U^\top$, where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$, and λ_i, $i = 1, \ldots, n$ are the eigenvalues of A and where the columns $u_i, i = 1, \ldots, n$ of U are the associated eigenvectors. For the matrix function $f(A)$, defined as $f(A) = Uf(\Lambda)U^\top$, where $f(\Lambda) = \mathrm{diag}(f(\lambda_1), \ldots, f(\lambda_n))$ [18], the trace estimation problems consist of computing an approximation of the trace of the matrix function $f(A)$, i.e.,

$$\mathtt{tr}(f(A)) = \sum_{i=1}^{n} f(\lambda_i). \quad (1)$$

The problem of estimating the trace of a matrix function appears in a very broad range of applications that include machine learning, signal processing, scientific computing, statistics, computational biology and computational physics [2,5,12,13,17,20,22,27,29]. Clearly, a trace of a matrix function can be trivially computed from the eigen-decomposition of A. However, matrices in most of the applications just mentioned are typically very large and so computing the

T. Kozubek et al. (Eds.): HPCSE 2017, LNCS 11087, pp. 19–33, 2018.
https://doi.org/10.1007/978-3-319-97136-0_2

complete eigen-decomposition can be expensive and sometimes even infeasible. Hence, the problem is to develop fast and scalable algorithms to perform such tasks without requiring the eigen-decomposition. This specific problem, which has been at the forefront of research in many distinct areas whether in physics or data-related applications, is the primary focus of this paper.

2 Trace Estimation Problems

We begin by first discussing a few trace estimation problems that arise in certain application areas.

2.1 Spectral Density

The first problem that we consider is that of computing the spectral density of a matrix [23], a very common problem in solid state physics. The spectral density of matrix, also known as Density of States (DOS) in physics, is a probability density distribution that measures the likelihood of finding eigenvalues of the matrix at a given point on the real line. Formally, the spectral density of a matrix is expressed as a sum of delta functions of the eigenvalues of the matrix. That is,

$$\phi(t) = \frac{1}{n}\sum_{i=1}^{n} \delta(t-\lambda_i),$$

where δ is the Dirac distribution or Dirac δ-function. This is not a proper function but a distribution and it is clearly not practically computable as it is defined. What is important is to compute a smoothed or approximate version of it that does not require computing eigenvalues, and several inexpensive methods have been proposed for this purpose, [23,31,35]. Recently, the DOS has been used in applications such as eigenvalue problem, e.g., for spectral slicing [38], for counting eigenvalues in intervals ('eigencounts') [11], and for estimating ranks [33,34]. In Sect. 4, we present a few other (new) applications where the DOS can be exploited.

Article [23] reviews a set of inexpensive methods for computing the DOS. Here we briefly discuss two of these methods, namely the Kernel Polynomial method [35] and the Lanczos approximation method [23]. As was already mentioned the DOS is not a proper function. At best it can practically be viewed as a highly discontinuous function, which is difficult to handle numerically. The idea then is to replace the delta function by a surrogate Gaussian blurring function as is often done. A "blurred" version of the DOS is given by:

$$\phi_\sigma(t) = \frac{1}{n}\sum_{i=1}^{n} h_\sigma(t-\lambda_i),$$

where $h_\sigma(t) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{t^2}{2\sigma^2}}$. The spectral density can then approximated by estimating this trace of this blurred function, e.g., using the Lanczos algorithm. An older method consists of just expanding (formally) the DOS into Chebyshev polynomials. These two methods will be discussed later in Sect. 3.

2.2 Eigencount and Numerical Rank

The second trace estimation application we consider is that of counting eigenvalues located in a given interval (eigencount) and the related problem of estimating the numerical rank of a matrix. Estimating the number of eigenvalues $\eta_{[a,b]}$ located in a given interval $[a, b]$ (including the possible multiplicities) of a large sparse symmetric matrix is a key ingredient of effective eigensolvers [11], because these eigensolvers require an estimate of the dimension of the eigenspace to compute to allocate resources and tune the method under consideration. Estimating the numerical rank $r_\epsilon = \eta_{[\epsilon,\lambda_1]}$ is another closely related problem that occurs in machine learning and data analysis applications such as Principal Component Analysis (PCA), low rank approximations, and reduced rank regression [33,34]. Both of these problems can be viewed from the angle of estimating the trace of a certain eigen-projector, i.e., the number of eigenvalues $\eta_{[a,\ b]}$ in $[a,\ b]$ satisfies:

$$\eta_{[a,b]} = \mathtt{tr}(P), \text{ where } P = \sum_{\lambda_i\ \in\ [a,\ b]} u_i u_i^\top.$$

We can interpret P as a step function of A given by

$$P = h(A), \text{ where } h(t) = \begin{cases} 1 & \text{if } \ t\ \in\ [a,\ b] \\ 0 & \text{otherwise} \end{cases}. \tag{2}$$

The problem then is to find an estimate of the trace of $h(A)$. A few inexpensive methods are proposed in [11,33,34,40] to approximately compute this trace. We can also compute the eigencount from the spectral density since

$$\eta_{[a,\ b]} = \int_a^b \sum_j \delta(t-\lambda_j)dt \equiv \int_a^b n\phi(t)dt. \tag{3}$$

2.3 Log-Determinant

Log-determinants have numerous applications in machine learning and related fields [17,27,29]. The logarithm of the determinant of a given positive definite matrix $A \in \mathbb{R}^{n\times n}$, is equal to the trace of the logarithm of the matrix, i.e.,

$$\log\det(A) = \mathtt{tr}(\log(A)) = \sum_{i=1}^{n} \log(\lambda_i).$$

So, estimating the log-determinant of a matrix leads once more to the estimation of the trace of a matrix function in this case the logarithm function.

Various methods have been proposed for inexpensively computing logdeterminants. These methods differ in the approach used to approximate the log function. For example, the article [17] uses Chebyshev polynomial approximations of the log function, while [8,39] uses Taylor series expansions. On the other hand, Aune et al. [3] developed a method based on rational approximations of the logarithm. More recently, a method based on Lanczos Quadrature has been advocated for computing log determinants [32].

Log-Likelihood Estimation: One of the applications of log-determinants computation is in the in likelihood estimation problems that arise in Gaussian processes [27]. Maximum Likelihood Estimation (MLE) is a popular approach used for parameter estimation in high dimensional Gaussian models. The objective is to maximize the log-likelihood function with respect to a hyperparameter vector ξ:

$$\log p(z \mid \xi) = -\frac{1}{2} z^\top S(\xi)^{-1} z - \frac{1}{2} \log\det S(\xi) - \frac{n}{2}\log(2\pi), \tag{4}$$

where z is the data vector and $S(\xi)$ is the covariance matrix parameterized by ξ. As seen from the above expression, a log-determinant must be computed to obtain the log-likelihood, see [32].

2.4 Other Applications

Other frequent matrix function trace estimation problems include estimating the trace of a matrix inverse, the Schatten norms, and the Estrada index. These are discussed in turn.

Trace of a Matrix Inverse: The matrix inverse trace estimation problem amounts to computing the trace of the inverse function $f(t) = t^{-1}$ of a positive definite matrix $A \in \mathbb{R}^{n\times n}$, whose eigenvalues lie in the interval $[\lambda_{\min}, \lambda_{\max}]$ with $\lambda_{\min} > 0$. This problem appears in uncertainty quantification and in lattice quantum chromodynamics [20,37], where it is necessary to estimate the trace of the inverse of covariance matrices.

Schatten p-Norms: Given an input matrix $X \in \mathbb{R}^{d\times n}$, the Schatten p-norm of X is defined as

$$\|X\|_p = \left(\sum_{i=1}^{r} \sigma_i^p\right)^{1/p},$$

where the σ_i's are the singular values of X and r its rank. The nuclear norm is the Schatten 1-norm so it is just the sum of the singular values. Estimating the nuclear norm and the Schatten p-norms of large matrices appears in matrix completion and in rank-constrained optimization problems, differential privacy and theoretical chemistry [25,32]. It is also used in SVD entropy computations [1] which has many applications [25]. Suppose we define a positive semidefinite matrix A as[1] $A = X^\top X$ or $A = XX^\top$. Then, the Schatten p-norm of X is defined as

$$\|X\|_p = \left(\sum_{i=1}^{r} \lambda_i^{p/2}\right)^{1/p} = \left(\mathtt{tr}(A^{p/2})\right)^{1/p}.$$

[1] The matrix product is not formed explicitly since the methods involved typically require only matrix vector products.

Hence, Schatten p-norms (the nuclear norm being a special case with $p = 1$) are the traces of matrix functions of A with $f(t) = t^{p/2}$, and they can be computed inexpensively using methods such as Lanczos Quadrature [32], Chebyshev expansions [16] and others [25].

Estrada Index: The Estrada index of graphs is a common tool in computational biology, and has applications that include protein indexing [12], statistical thermodynamics and information theory [9]. Estimating the Estrada index amounts to finding an approximation to the trace of the exponential function, i.e., we need to estimate $\texttt{tr}(\exp(A))$, where A is the adjacency matrix of the graph. Articles [16,25,32] discuss methods for a fast estimation of the Estrada index of graphs.

3 Methods

We now discuss two inexpensive techniques for estimating traces of matrix functions. The first approach is well-known among solid-state physicists. Termed 'Kernel Polynomial Method' (KPM) [34–36] or 'Chebyshev expansion method' [16,17,33], it consists simply of expanding the spectral density into Chebyshev polynomials. The second approach is based on the Lanczos algorithm [23,32], where the relation between the Lanczos procedure and Gaussian quadrature formulas is exploited to construct a good approximation for the matrix function. We first discuss a standard tool known as the 'stochastic trace estimator', which is a key ingredient used in the methods to be discussed.

3.1 Stochastic Trace Estimator

The stochastic trace estimator [4,19,28] approximates the trace of a matrix function $f(A)$ by means of matrix-vector products with $f(A)$. This method takes a sequence of random vectors $v_l, l = 1, .., \mathrm{n_v}$ whose components have zero mean $\mathbb{E}[v_l] = 0$ and whose 2-norm is one, $\|v_l\|_2 = 1$ and it then computes the average over the samples of $v_l^\top f(A) v_l$ to approximate the trace,

$$\texttt{tr}(f(A)) \approx \frac{n}{\mathrm{n_v}} \sum_{l=1}^{\mathrm{n_v}} v_l^\top f(A) v_l. \tag{5}$$

Convergence has been analyzed in [4,28]. A variant of this method can be used to estimate the diagonal entires of $f(A)$, see [7]. Note that $f(A)$ need not be explicitly formed since we only need to efficiently compute the vectors $f(A)v_l$ for any v_l. This can be accomplished in a number of effective ways.

3.2 Chebyshev (Kernel) Polynomial Method

The Kernel Polynomial Method (KPM) proposed in [30,35] computes an approximate DOS of a matrix using Chebyshev polynomial expansions, see, [23] for a discussion. In KPM, the matrix is linearly transformed so as to map its eigenvalues from the initial interval $[\lambda_n,\ \lambda_1]$ into the interval $[-1,\ 1]$. This requires estimating the extreme eigenvalues, see [34]. KPM seek the expansion of:

$$\hat{\phi}(t) = \sqrt{1-t^2}\phi(t) = \sqrt{1-t^2} \cdot \frac{1}{n}\sum_{j=1}^{n}\delta(t-\lambda_j),$$

instead of the original $\phi(t)$ since the Chebyshev polynomials are orthogonal with respect to the weight function $(1-t^2)^{-1/2}$. Then, we write the partial expansion of $\hat{\phi}(t)$ as

$$\hat{\phi}(t) \approx \sum_{k=0}^{m}\mu_k T_k(t),$$

where $T_k(t)$ is the Chebyshev polynomial of degree k. A little calculation reveals that, formally at least, each expansion coefficient μ_k is given by $\mu_k = \frac{(2-\delta_{k0})}{\pi}\mathtt{tr}(T_k(A))$. Here δ_{ij} is the Kronecker symbol, so $2-\delta_{k0}$ is 1 when $k=0$ and 2 otherwise. The trace of $T_k(A)$ can now be estimated with the help of the expansion coefficients the corresponding expansion coefficient μ_k are approximated as,

$$\mu_k \approx \frac{2-\delta_{k0}}{\pi \mathrm{n_v}}\sum_{l=1}^{\mathrm{n_v}}(v_l)^\top T_k(A)v_l.$$

Scaling back by the weight function $(1-t^2)^{-1/2}$, we obtain the approximation for the spectral density function in terms of Chebyshev polynomial of degree m.

General Function $f(A)$: For a general function $f:[-1,1]\to\mathbb{R}$, it is possible to obtain an approximation of the form

$$f(t) \approx \sum_{k=0}^{m}\gamma_k T_k(t).$$

using Chebyshev polynomial expansions or interpolations, see [24] for details. Here $T_k(t)$ is the Chebyshev polynomial of degree k and γ_k are corresponding coefficients specific to expanding the function $f(t)$. Hence, we can approximate the trace of $f(A)$ as

$$\mathtt{tr}(f(A)) \approx \frac{n}{\mathrm{n_v}}\sum_{l=1}^{\mathrm{n_v}}\left[\sum_{k=0}^{m}\gamma_k(v_l)^T T_k(A)v_l\right], \tag{6}$$

using the Chebyshev expansions and the stochastic trace estimator (5). Articles [11,33] discussed the expansion of step functions using Chebyshev polynomials to compute eigencounts and numerical ranks of matrices, respectively.

Han et al. [16] discussed the above Chebyshev expansion method to approximately estimate the traces $\mathtt{tr}(f(A))$, when the function f is analytic over an interval. They proposed using Chebyshev interpolations to obtain the coefficients γ_k. Problems such as estimating log-determinants, Estrada index, trace of matrix inverse and Shcatten p norms were discussed. The functions $\log(t), \exp(t), t^{-1}$ and $t^{p/2}$ are all analytic in the spectrum interval $[\lambda_{\min}, \lambda_{\max}]$, with $\lambda_{\min} > 0$.

When expanding discontinuous functions including the DOS and step functions using Chebyshev polynomials, oscillations known as *Gibbs Oscillations* appear near the discontinuities [11]. To reduce or suppress these oscillations, damping multipliers are often used, see [11,23] for details. An important practical consideration is that we can economically compute vectors of the form $T_k(A)v$ using the three term recurrence of Chebyshev polynomials, see [16,34]. The recent article [16] analyzed the convergence of methods for approximating $\mathtt{tr}(f(A))$ with the Chebyshev method when the function $f(A)$ is analytic over the interval of interest.

3.3 Lanczos Quadrature

The Lanczos Quadrature method was developed in [14], and the idea of combining the stochastic trace estimator with the Lanczos Quadrature method appeared in [5,6]. This method was recently analyzed and applied to matrix function trace estimation in [32]. In the Stochastic Lanczos Quadrature (SLQ) method, the scalar quantities $v^\top f(A)v$ in the trace estimator (5) are computed by treating them to Riemann-Stieltjes integral, and then using the Gauss quadrature rule to approximate this integral. Given the eigen-decomposition $A = Q\Lambda Q^\top$, we can write the scalar product as Riemann-Stieltjes integral given by,

$$v^\top f(A)v = v^\top Q f(\Lambda) Q^\top v = \sum_{i=1}^{n} f(\lambda_i)\mu_i^2 = \int_a^b f(t)d\mu(t), \tag{7}$$

where μ_i are the components of the vector $Q^\top v$ and the measure $\mu(t)$ is a piecewise constant function defined as

$$\mu(t) = \begin{cases} 0, & \text{if } t < a = \lambda_1, \\ \sum_{j=1}^{i-1} \mu_j^2, & \text{if } \lambda_{i-1} \le t < \lambda_i,\ i = 2, \ldots, n, \\ \sum_{j=1}^{n} \mu_j^2, & \text{if } b = \lambda_n \le t, \end{cases} \tag{8}$$

with λ_i ordered nondecreasingly. However, the complete eigen-decomposition of A will not be available, and will be very expensive to compute for large matrices. So, we consider estimating the integral using the Gauss quadrature rule [15]

$$\int_a^b f(t)d\mu(t) \approx \sum_{k=0}^{m} \omega_k f(\theta_k), \tag{9}$$

where $\{\omega_k\}$ are the weights and $\{\theta_k\}$ are the nodes of the $(m+1)$-point Gauss quadrature. We can then compute these weights and nodes using the Lanczos algorithm [13].

Given symmetric matrix $A \in \mathbb{R}^{n\times n}$ and a starting vector w_0 of unit 2-norm, the Lanczos algorithm generates an orthonormal basis W_{m+1} for the *Krylov subspace* span$\{w_0, Aw_0, \ldots, A^m w_0\}$ such that $W_{m+1}^\top A W_{m+1} = T_{m+1}$, where T_{m+1} is an $(m{+}1)\times(m{+}1)$ tridiagonal matrix. The columns w_k of W_{m+1} are related as

$$w_k = p_{k-1}(A)w_0,\ k = 1, \ldots, m,$$

where p_k are the Lanczos polynomials. The Lanczos polynomials are orthogonal with respect to the measure $\mu(t)$ in (8); see Theorem 4.2 in [13]. The nodes and the weights of the quadrature rule in (9) can be computed as the eigenvalues and the squares of the first entries of the eigenvectors of T_{m+1}. Thus, we have

$$v^\top f(A)v \approx \sum_{k=0}^{m} \tau_k^2 f(\theta_k) \quad \text{with} \quad \tau_k^2 = \left(e_1^\top y_k\right)^2, \tag{10}$$

where $(\theta_k, y_k), k = 0, 1, ..., m$ are eigenpairs (eigenvalues and eigenvectors) of T_{m+1} by using v as the starting vector w_0. Note that eigen-decomposition of T_{m+1} for small m is inexpensive to compute. Then, the trace of matrix function $f(A)$ can be computed as,

$$\mathtt{tr}(f(A)) \approx \frac{n}{\mathrm{n_v}} \sum_{l=1}^{\mathrm{n_v}} \left(\sum_{k=0}^{m} (\tau_k^{(l)})^2 f(\theta_k^{(l)}) \right), \tag{11}$$

where $(\theta_k^{(l)}, \tau_k^{(l)}), k = 0, 1, ..., m$ are eigenvalues and the first entries of the eigenvectors of the tridiagonal matrix $T_{m+1}^{(l)}$ corresponding to the starting vectors $v_l, l = 1, \ldots, \mathrm{n_v}$. A convergence analysis for this SLQ method was proposed in the recent article [32]. For analytic functions, it has been shown that the convergence of the Lanczos quadrature approximation is twice as fast as that of Chebyshev approximation methods. This stems from the fact that an m-point quadrature rule is exact for any polynomial of degree $2m$, see [32] for details.

Lanczos Approximation for the DOS. The Lanczos approximation technique for estimating spectral densities discussed in [23] is based on the above Lanczos Quadrature framework. Given a polynomial $p(t)$, we can use the Lanczos quadrature formula in Eq. (10), to compute the (Riemann-Stieltjes) integral $v^\top p(A)v$, see [13] for details. Since this is a Gaussian quadrature formula, it is exact when p is a polynomial of degree $\le 2m + 1$.

As seen earlier, for an initial vector w_0 of the Lanczos sequence, expanded in the eigenbasis $\{u_i\}_{i=1}^n$ of A as $w_0 = \sum_{i=1}^n \beta_i u_i$ and we consider the discrete (Stieltjes) integral:

$$\int p(t)d\mu(t) = (p(A)w_0, w_0) = \sum_{i=1}^{n} \beta_i^2 p(\lambda_i). \tag{12}$$

This integral is a distribution ϕ_{w_0} applied to p, written as $(p(A)w_0, w_0) \equiv \langle \phi_{w_0}, p \rangle$. If we assume an idealistic situation where $\beta_i^2 = 1/n$ for all i, then ϕ_{w_0} will be exactly the distribution, the DOS function. In the sense of distributions,

$$\langle \phi_{w_0}, p \rangle \equiv (p(A)w_0, w_0) = \sum_{i=1}^{n} \beta_i^2 p(\lambda_i) = \sum_{i=1}^{n} \beta_i^2 \langle \delta_{\lambda_i}, p \rangle = \frac{1}{n} \sum_{i=1}^{n} \langle \delta_{\lambda_i}, p \rangle ,$$

where δ_{λ_i} is a δ-function at λ_i. Then, from the Gaussian quadrature rule (10), we have: $\langle \phi_{w_0}, p \rangle \approx \sum_{k=1}^{m} \tau_k^2 p(\theta_k) = \sum_{k=1}^{m} \tau_k^2 \langle \delta_{\theta_k}, p \rangle$ and

$$\phi_{w_0} \approx \sum_{k=1}^{m} \tau_k^2 \delta_{\theta_k} .$$

Since the β_i's are not equal in practice, we will need to use several starting vectors v_l and average the result of the above formula over them. This is the Lanczos approximation method for computing an approximate DOS [23].

If $(\theta_k^{(l)}, y_k^{(l)}), k = 1, 2, ..., m$ are eigenpairs of the tridiagonal matrix T_m corresponding to the starting vector $v_l, l = 1, \ldots, \mathrm{n_v}$ and $\tau_k^{(l)}$ is the first entry of $y_k^{(l)}$, then the DOS function by Lanczos approximation is given by

$$\tilde{\phi}(t) = \frac{1}{\mathrm{n_v}} \sum_{l=1}^{\mathrm{n_v}} \left(\sum_{k=1}^{m} (\tau_k^{(l)})^2 \delta(t - \theta_k^{(l)}) \right) . \tag{13}$$

The above function is a weighted spectral distribution of T_m, where τ_k^2 is the weight for the corresponding θ_k and it approximates the spectral density of A.

Computational Cost: The most expensive step in KPM is when computing the scalars $(v_l)^\top T_k(A) v_l$ for $l = 1, \ldots, \mathrm{n_v}, k = 0, \ldots, m$. Hence, the computational cost for estimating matrix function traces by KPM will be $O(\mathrm{nnz}(A) m \mathrm{n_v})$ for sparse matrices, where $\mathrm{nnz}(A)$ is the number of nonzero entries of A. Similarly, the most expensive part of the Lanczos Quadrature procedure is to perform the m Lanczos steps with the different starting vectors. The computational cost for matrix function trace estimation by SLQ will be $O((\mathrm{nnz}(A)m + nm^2)\mathrm{n_v})$ for sparse matrices where there is an assumed cost for reorthogonalizing the Lanczos vectors. As can be seen, these algorithms are inexpensive relative to methods that require matrix factorizations such as the QR or SVD.

4 Applications of the DOS

Among the applications of the DOS that were mentioned earlier, we will discuss two that are somewhat related. The first is a tool employed for estimating the rank of a matrix. This is only briefly sketched as it has been discussed in details earlier in [33,34]. The second application, is in clustering and the problem of community detection in social graphs.

Threshold Selection for Rank Estimation: The *numerical rank* of a general matrix $X \in \mathbb{R}^{d\times n}$ is defined with respect to a positive tolerance ϵ as follows:

$$r_\epsilon = \min\{\text{rank}(B) : B \in \mathbb{R}^{d\times n}, \|X - B\|_2 \le \epsilon\}. \qquad (14)$$

To estimate r_ϵ we need to provide a good value for the threshold ϵ to be used to determine the rank. Recently, references [33,34] proposed a method for determining this threshold ϵ based on the plot of DOS. The idea is to detect a gap between the noisy and relevant eigenvalues (we are interested in the count of these relevant eigenvalues) by locating a local minima near zero in the DOS plot. The cutoff point is chosen to be where the derivative of the spectral density function becomes close to zero (local minimum) for the first time. Thus, the threshold ϵ can be selected as

$$\epsilon = \min\{t : \phi'(t) \ge tol, \lambda_n \le t \le \lambda_1\}, \qquad (15)$$

for a small tolerance, e.g., $tol = -0.01$ and not zero for practical reasons.

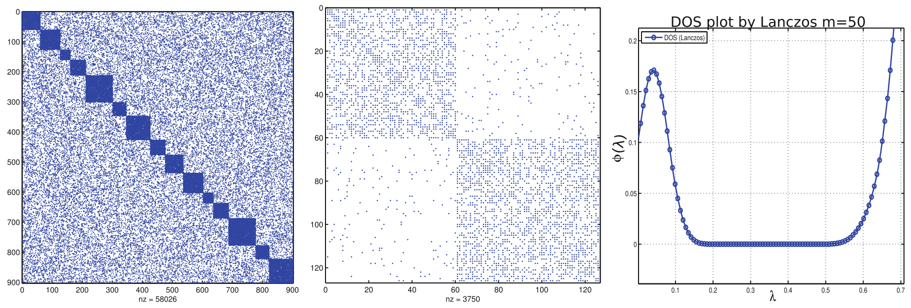

Fig. 1. Matrix of size $n = 902$ showing a block structure (left) and a zoom at the pattern focusing in the first 2 blocks (center). The DOS plot zooms in on the part between 0 and ≈ 0.7 (right).

Community Detection in Graphs: A problem of primary importance in various scientific and engineering disciplines is to determine dense subgraphs or *communities* within a given graph [21,26] Here we exploit the idea of spectral densities to determine the number of communities in a graph.

Let W denote the adjacency graph of a given (undirected) graph and let $L = D - W$ be the graph Laplacian with $D = \text{diag}(We)$ the diagonal matrix with diagonal entries as the sums of the entries of the corresponding row in W. In the ideal case of perfect clusters, where each cluster is a clique, the matrix L will have an eigenvalue of zero for each block, leading to a multiple zero eigenvalue, one for each block. By way of illustration consider left plot of Fig. 1. This represents the sparsity pattern of a small synthetic example made up of $k = 15$ sparse diagonal blocks that have a good density (about half dense) completed by some sparse matrix that has a lower density. The sizes of the blocks vary from 105 to 190. The right side of the figure shows a zoom on the first two blocks to provide an illustration.

In an ideal scenario, each of the diagonal blocks is dense and there are no elements outside of these blocks. Then, the graph Laplacian will have exactly k zero eigenvalues where k is the number of blocks, which is 15 here. This is still true when the matrix is block diagonal but each diagonal block is a small sparse Laplacian, that contributes one zero eigenvalue. When we have certain off-block diagonal entries (equivalent to noise), the zero eigenvalues are perturbed, and we get a cluster of eigenvalues close to the origin. This is illustrated in the DOS plot corresponding to this matrix (zoomed in), see rightmost plot of Fig. 1. If we count the number of eigenvalues included in that cluster we will find that it is equal to the number of blocks. It is not too difficult to devise small subroutines that will consider the DOS curve and spot the point where the curve levels off after descending from its peak value to the left. A short matlab script we wrote finds the value $\tau = 0.136$ and the number of blocks detected with the help of spectral densities is close to the correct number of $k = 15$. Thus, this technique can be used to find an effective way to estimate the number of dense subgraphs.

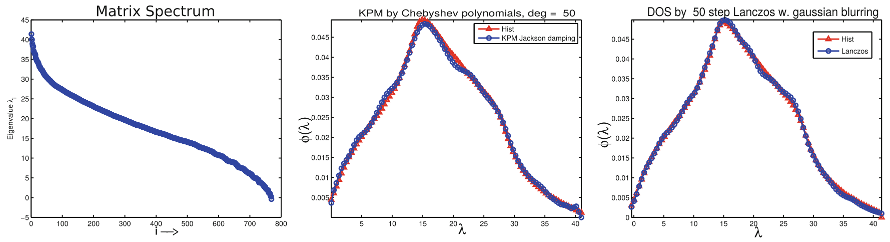

Fig. 2. The spectrum (Left), and the approximate DOS found by KPM (Middle), and by SLQ for the example `Si2`.

Spectral densities can be used to extract a wealth of additional information. For example, the graphs may have multi-level clustering or sub-communities within the larger communities. Such situations are common in social networks, document and product categorization, etc. For such matrices, the multi-level clustering will result in a matrix with multiple clusters of eigenvalues in the DOS curve. It is possible to identify such clusters using the DOS and count the eigenvalues within these clusters, corresponding to the number of sub-communities.

5 Numerical Examples

In this section, we present a few examples to illustrate the performances of the Kernel polynomial and the Lanczos Quadrature methods for different problems of estimating traces of matrix functions.

Spectral Density. In the first experiment, to illustrate the performances of KPM and the Lanczos Approximation for approximating the DOS using a small example with matrix[2] `Si2` of size 769. Figure 2 plots the matrix spectrum (left), and the DOS plots obtained using KPM (Middle) and SLQ (right) with degree $m = 50$ and a number of samples $\mathrm{n_v} = 30$, respectively. The red triangular lines in both plots represent the actual histogram (spectral density) of the matrix. The blue circle lines are the estimated DOS. Jackson damping [23] was used for KPM and Gaussian blurring with $\sigma = 0.25$ was used for Lanczos approximation plot. We note that the Lanczos algorithm gives slightly more accurate results for the same degree m compared to KPM. However,the Lanczos method is slightly more expensive due to the orthogonalization step.

Numerical Rank Estimation. The following experiment will illustrate the performances of the two techniques, KPM and SLQ, for estimating the numerical rank, see Fig. 3. We consider a 1961×1961 matrix named `netz4504` from the SuiteSparse collection (see footnote 2), see [10]. The matrix spectrum and the DOS plot obtained using KPM with degree $m = 50$ and a number of samples $\mathrm{n_v} = 30$ are given in the left plots of Fig. 3. The threshold ϵ (the gap) estimated using this DOS plot was $\epsilon = 0.12$. The middle figure plots the estimated approximate ranks for different degrees m (top) and different number of starting vector $\mathrm{n_v}$ (bottom) using KPM (black solid line). Similarly, the right plots in Fig. 3 shows the estimated approximate ranks by the Lanczos approximation method using different degrees m (or the dimension of the Krylov subspace for the Lanczos method) and different number of sample vectors $\mathrm{n_v}$. $\mathrm{n_v} = 30$ in the top plot

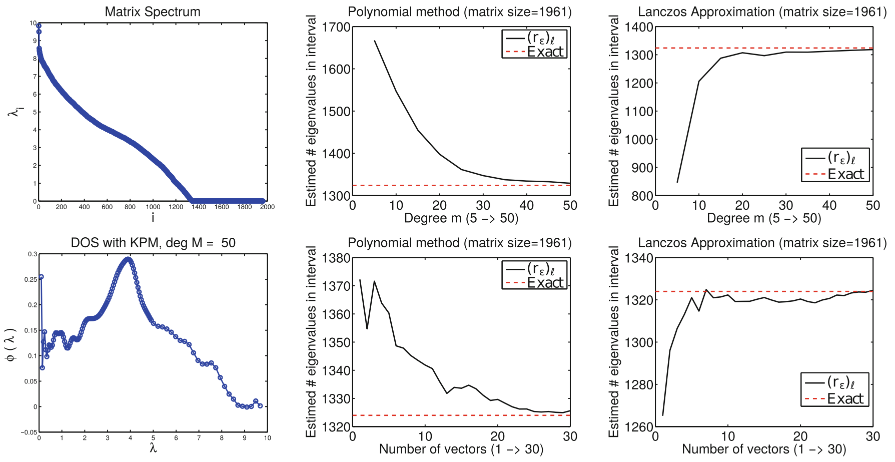

Fig. 3. The spectrum, the DOS found by KPM, and the approximate ranks estimation by KPM and by SLQ for the example `netz4504`.

[2] The matrices used in the experiments can be obtained from the SuiteSparse matrix collection: https://sparse.tamu.edu/.

and $m = 50$ in the bottom in both cases. The average approximate rank estimated over 30 sample vectors is equal to 1323.12 by KPM and by SLQ is equal to 1325.68. The exact number of eigenvalues in the interval is 1324, (indicated by the dash line in the plot). In this case ($m = 50, \mathrm{n_v} = 30$), the number of matrix-vector multiplications required for both the rank estimator techniques is 1500.

6 Conclusion

The aim of this article, and the related presentation at HPCSE17, has been to provide a brief overview of the applications of traces of matrix functions and of effective related techniques for approximating them. In particular, we strived to highlight the broad applicability and the importance of spectral densities. These provide vital information in certain disciplines in physics and for this reason physicists were the first to develop effective methods such as KPM for computing them. On the other hand, the use of spectral densities, and more generally traces of matrix functions, in other areas such as machine learning and computational statistics, is more recent. There is no doubt that the interest in this topic will gain in importance given the rapid progress of disciplines that exploit large datasets.

Acknowledgments. This work was supported by NSF under grant CCF-1318597.

References

1. Alter, O., Brown, P.O., Botstein, D.: Singular value decomposition for genome-wide expression data processing and modeling. Proc. Nat. Acad. Sci. **97**(18), 10101–10106 (2000)
2. Andoni, A., Krauthgamer, R., Razenshteyn, I.: Sketching and embedding are equivalent for norms. In: Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, pp. 479–488. ACM (2015)
3. Aune, E., Simpson, D.P., Eidsvik, J.: Parameter estimation in high dimensional Gaussian distributions. Stat. Comput. **24**(2), 247–263 (2014)
4. Avron, H., Toledo, S.: Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix. J. ACM **58**(2), 8 (2011)
5. Bai, Z., Fahey, G., Golub, G.: Some large-scale matrix computation problems. J. Comput. Appl. Math. **74**(1), 71–89 (1996)
6. Bai, Z., Golub, G.H.: Bounds for the trace of the inverse and the determinant of symmetric positive definite matrices. Annal. Numer. Math. **4**, 29–38 (1996)
7. Bekas, C., Kokiopoulou, E., Saad, Y.: An estimator for the diagonal of a matrix. Appl. Numer. Math. **57**(11), 1214–1229 (2007)
8. Boutsidis, C., Drineas, P., Kambadur, P., Kontopoulou, E.-M., Zouzias, A.: A randomized algorithm for approximating the log determinant of a symmetric positive definite matrix. Linear Algebra Appl. **533**, 95–119 (2017)
9. Carbó-Dorca, R.: Smooth function topological structure descriptors based on graph-spectra. J. Math. Chem. **44**(2), 373–378 (2008)

10. Davis, T.A., Hu, Y.: The University of Florida sparse matrix collection. ACM Trans. Math. Softw. (TOMS) **38**(1), 1 (2011)
11. Di Napoli, E., Polizzi, E., Saad, Y.: Efficient estimation of eigenvalue counts in an interval. ArXiv preprint arXiv:1308.4275 (2013)
12. Estrada, E.: Characterization of 3D molecular structure. Chem. Phys. Lett. **319**(5), 713–718 (2000)
13. Golub, G.H., Meurant, G.: Matrices, Moments and Quadrature with Applications. Princeton University Press, Princeton (2009)
14. Golub, G.H., Strakoš, Z.: Estimates in quadratic formulas. Numer. Algorithms **8**(2), 241–268 (1994)
15. Golub, G.H., Welsch, J.H.: Calculation of gauss quadrature rules. Math. Comput. **23**(106), 221–230 (1969)
16. Han, I., Malioutov, D., Avron, H., Shin, J.: Approximating spectral sums of large-scale matrices using stochastic Chebyshev approximations. SIAM J. Sci. Comput. **39**(4), A1558–A1585 (2017)
17. Han, I., Malioutov, D., Shin, J.: Large-scale log-determinant computation through stochastic Chebyshev expansions. In: Proceedings of the 32nd International Conference on Machine Learning, pp. 908–917 (2015)
18. Higham, N.J.: Functions of Matrices: Theory and Computation. SIAM, University City (2008)
19. Hutchinson, M.F.: A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. Commun. Stat.-Simul. Comput. **19**(2), 433–450 (1990)
20. Kalantzis, V., Bekas, C., Curioni, A., Gallopoulos, E.: Accelerating data uncertainty quantification by solving linear systems with multiple right-hand sides. Numer. Algorithms **62**(4), 637–653 (2013)
21. Kaufman, L., Rousseeuw, P.J.: Finding Groups in Data: An Introduction to Cluster Analysis. Wiley, Hoboken (1990)
22. Li, Y., Nguyên, H.L., Woodruff, D.P.: On sketching matrix norms and the top singular vector. In: Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1562–1581. SIAM (2014)
23. Lin, L., Saad, Y., Yang, C.: Approximating spectral densities of large matrices. SIAM Rev. **58**(1), 34–65 (2016)
24. Mason, J.C., Handscomb, D.C.: Chebyshev Polynomials. CRC Press, Boca Raton (2002)
25. Musco, C., Netrapalli, P., Sidford, A., Ubaru, S., Woodruff, D.P.: Spectrum approximation beyond fast matrix multiplication: algorithms and hardness. arXiv preprint arXiv:1704.04163 (2017)
26. Newman, M.E.J.: Finding community structure in networks using the eigenvectors of matrices. Phys. Rev. E **74**(3), 036104 (2006)
27. Rasmussen, C., Williams, C.: Gaussian Processes for Machine Learning. MIT Press, Cambridge (2006)
28. Roosta-Khorasani, F., Ascher, U.: Improved bounds on sample size for implicit matrix trace estimators. Found. Comput. Math. **15**(5), 1187–1212 (2015)
29. Rue, H., Held, L.: Gaussian Markov Random Fields: Theory and Applications. CRC Press, Boca Raton (2005)
30. Silver, R., Röder, H.: Densities of states of mega-dimensional Hamiltonian matrices. Int. J. Mod. Phys. C **5**(04), 735–753 (1994)
31. Turek, I.: A maximum-entropy approach to the density of states within the recursion method. J. Phys. C: Solid State Phys. **21**(17), 3251 (1988)

32. Ubaru, S., Chen, J., Saad, Y.: Fast estimation of tr(f(A)) via stochastic Lanczos quadrature. SIAM J. Matrix Anal. Appl. **38**, 1075–1099 (2017)
33. Ubaru, S., Saad, Y.: Fast methods for estimating the numerical rank of large matrices. In: Proceedings of the 33rd International Conference on Machine Learning, pp. 468–477 (2016)
34. Ubaru, S., Saad, Y., Seghouane, A.-K.: Fast estimation of approximate matrix ranks using spectral densities. Neural Comput. **29**(5), 1317–1351 (2017)
35. 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**(15), 10154 (1994)
36. Weiße, A., Wellein, G., Alvermann, A., Fehske, H.: The kernel polynomial method. Rev. Mod. Phys. **78**(1), 275 (2006)
37. Wu, L., Laeuchli, J., Kalantzis, V., Stathopoulos, A., Gallopoulos, E.: Estimating the trace of the matrix inverse by interpolating from the diagonal of an approximate inverse. J. Comput. Phys. **326**, 828–844 (2016)
38. Xi, Y., Li, R., Saad, Y.: Fast computation of spectral densities for generalized eigenvalue problems. arXiv preprint arXiv:1706.06610 (2017)
39. Zhang, Y., Leithead, W.E.: Approximate implementation of the logarithm of the matrix determinant in Gaussian process regression. J. Stat. Comput. Simul. **77**(4), 329–348 (2007)
40. Zhang, Y., Wainwright, M.J., Jordan, M.I.: Distributed estimation of generalized matrix rank: efficient algorithms and lower bounds. In: Proceedings of The 32nd International Conference on Machine Learning, pp. 457–465 (2015)