SIAM J. Matrix Anal. Appl.
Vol. 38, No. 4, pp. 1075–1099

FAST ESTIMATION OF tr($f(A)$) VIA STOCHASTIC LANCZOS QUADRATURE[*]

SHASHANKA UBARU[†], JIE CHEN[‡], AND YOUSEF SAAD[†]

Abstract. The problem of estimating the trace of matrix functions appears in applications ranging from machine learning and scientific computing, to computational biology. This paper presents an *inexpensive* method to estimate the trace of $f(A)$ for cases where f is analytic inside a closed interval and A is a symmetric positive definite matrix. The method combines three key ingredients, namely, the stochastic trace estimator, Gaussian quadrature, and the Lanczos algorithm. As examples, we consider the problems of estimating the log-determinant ($f(t) = \log(t)$), the Schatten p-norms ($f(t) = t^{p/2}$), the Estrada index ($f(t) = e^t$), and the trace of the matrix inverse ($f(t) = t^{-1}$). We establish multiplicative and additive error bounds for the approximations obtained by this method. In addition, we present error bounds for other useful tools such as approximating the log-likelihood function in the context of maximum likelihood estimation of Gaussian processes. Numerical experiments illustrate the performance of the proposed method on different problems arising from various applications.

Key words. trace estimation, Lanczos algorithm, log-determinants, fast approximate algorithms, Schatten norms, Gaussian processes

AMS subject classifications. 65C05, 65D32, 65F15, 65F60, 65F30, 65G99, 65Y20

DOI. 10.1137/16M1104974

1. Introduction.

The problem of estimating the trace of matrix functions appears frequently in applications of machine learning, signal processing, scientific computing, statistics, computational biology, and computational physics [6, 17, 39, 37, 20, 30, 33, 2, 26]. Developing fast and scalable algorithms to perform this task has long been a primary focus of research in these fields. An important instance of the trace estimation problem is that of approximating $\log(\det(A))$, the log-determinant of a positive definite matrix A. Log-determinants of covariance and precision matrices play an important role in Gaussian processes and Gaussian graphical models [37, 39]. Log-determinant computations also appear in applications such as kernel learning [14], discrete probabilistic models [1], Bayesian learning [35], spatial statistics [4], and Markov field models [45, 26, 9].

Another instance of the trace estimation problem in applications is that of estimating Schatten p-norms, particularly the nuclear norm, since this norm is used as the convex surrogate of the matrix rank. The Schatten p-norms appear in convex optimization problems, e.g., in the context of matrix completion [10], in differential privacy problems [27], and in sketching and streaming models [33, 2]. On the other hand, in uncertainty quantification and in lattice quantum chromodynamics [30, 46], it is necessary to estimate the trace of the inverse of covariance matrices. Moreover, estimating the Estrada index (trace of the exponential function) is another illustration

[*] Received by the editors November 28, 2016; accepted for publication (in revised form) by Michele Benzi July 27, 2017; published electronically October 3, 2017.

http://www.siam.org/journals/simax/38-4/M110497.html

Funding: The work of the first and third authors was supported by NSF under grant NSF/CCF-1318597. The second author's work was supported in part by the XDATA program of the Defense Advanced Research Projects Agency (DARPA), administered through Air Force Research Laboratory contract FA8750-12-C-0323.

[†] Computer Science & Engineering, University of Minnesota, Minneapolis, MN 55455 (ubaru001@umn.edu, saad@umn.edu).

[‡] IBM Thomas J. Watson Research Center, Yorktown Heights, NY 10598 (chenjie@us.ibm.com).

of the problem. Other applications include protein indexing [17], statistical thermodynamics [18], and information theory [11].

For a symmetric matrix $A \in \mathbb{R}^{n\times n}$ with an eigendecomposition $A = U\Lambda U^T$ with $\Lambda = diag(\lambda_1, \ldots, \lambda_n)$, where λ_i, $i = 1, \ldots, n$ are the eigenvalues of A, the matrix function $f(A)$ is defined as $f(A) = Uf(\Lambda)U^T$ with $f(\Lambda) = diag(f(\lambda_1), \ldots, f(\lambda_n))$ [28]. Then the trace estimation problems mentioned above can be formulated as follows: given a symmetric matrix $A \in \mathbb{R}^{n\times n}$, compute an approximation of the trace of the matrix function $f(A)$, i.e.,

$$\texttt{tr}(f(A)) = \sum_{i=1}^{n} f(\lambda_i), \tag{1}$$

where λ_i, $i = 1, \ldots, n$, are the eigenvalues of A, and f is the desired function. A naive approach for estimating the trace of matrix functions is to compute this trace from the eigenvalues of the matrix. A popular approach to computing the log-determinant is to exploit the Cholesky decomposition [22]. Given the decomposition $A = LL^\top$, the log-determinant of A is $\log\det(A) = 2\sum_i \log(L_{ii})$. Computing the Schatten norms in a standard way would typically require the singular value decomposition (SVD) of the matrix. These methods have cubic computational complexity (in terms of the matrix dimension, i.e., $O(n^3)$ cost) in general, and are not viable for large scale applications. In this paper, we study inexpensive methods for accurately estimating these traces for large matrices.

Our contribution. This paper is a study of the method we call the stochastic Lanczos quadrature (SLQ) for approximating the trace of functions of large matrices [6, 7, 20]. The method combines three key ingredients. First, the stochastic trace estimator, also called the *Hutchinson method* [29], is considered for approximating the trace. Next, the bilinear form that appears in the trace estimator is expressed as a Riemann–Stieltjes integral, and the Gauss quadrature rule is used to approximate this integral. Finally, the Lanczos algorithm is used to obtain the weights and the nodes of the quadrature rule (see section 3 for details). We establish multiplicative and additive approximation error bounds for the trace obtained by using the method. To the best of our knowledge, such error bounds for SLQ have not appeared in the prior literature. We show that the Lanczos quadrature approximation has a faster convergence rate compared to popular methods such as those based on Chebyshev or Taylor series expansions. The analysis can be extended to any matrix functions that are analytic inside a closed interval and are analytically continuable to an open Bernstein ellipse [42].

We consider several important trace estimation problems and their applications. We discuss the log-determinant computation, estimation of the Estrada index, and the trace of the matrix inverse, and show how the SLQ method can be used to approximately estimate these quantities rapidly. We also adapt our method for fast estimation of the nuclear norm and Schatten-p norms of large matrices. In addition, we establish error bounds for the approximation of log-likelihoods in the context of maximum likelihood estimation of Gaussian processes. Several numerical experiments are presented to demonstrate the superiority of the proposed method over existing methods in practice.

Related works and comparison. A plethora of methods have been developed in the literature to deal with trace estimation problems. In the following, we discuss some of the works that are closely related to SLQ, particularly those that invoke the stochastic trace estimator. The stochastic trace estimator has been employed for a number of applications in the literature, for example, for estimating the diagonal of a matrix [8], for counting eigenvalues inside an interval [16], for approximating the score function of Gaussian processes [41], and for estimating the numerical rank [43, 44]. For the log-determinant computation, a few methods have been proposed, which also invoke the stochastic trace estimator. These methods differ in the approach used to approximate the log function. [26] used the Chebyshev polynomial approximations for the log function. The log function was approximated using the Taylor series expansions in [47]. [9] provided an improved analysis for the log-determinant computations using these Taylor series expansions. Aune, Simpson, and Eidsvik [4] adopted the method proposed in [24] to estimate the log function. Here, the Cauchy integral formula of the log function is considered and the trapezoidal rule is invoked to approximate the integral. This method is equivalent to using a rational approximation for the function. The method requires solving a series of linear systems and is generally expensive. The functions can also be approximated by means of least squares polynomials as proposed in [13].

Not many fast algorithms are available in the literature to approximate the nuclear norm and Schatten-p norms; see [33, 2] for discussions. [25] extends the idea of using Chebyshev expansions developed in [16, 26] to approximate the trace of various matrix functions including Schatten norms, the Estrada index, and the trace of the matrix inverse. Related articles on estimating the trace of the matrix inverse and other matrix functions are [46, 12].

A key objective of this work is to demonstrate how the powerful Lanczos algorithm can be employed to solve trace estimation problems for matrix functions. The Lanczos method has clear advantages over the above-mentioned methods such as Chebyshev expansions, Taylor series expansions, and rational function approximations. To understand the pros and cons of the Lanczos method, let us first examine the three classes of techniques that are commonly used, namely, the Lanczos method, polynomial approximation methods, and rational approximation methods. Most of the polynomial and rational approximation methods require as input an interval containing the spectrum of the matrix. One advantage of the Lanczos method is that there exists no such requirement. In fact, the Lanczos algorithm itself is often used to estimate the spectrum interval. On the other hand, a disadvantage of the Lanczos method is that it requires one to store the Lanczos vectors and to reorthogonalize these vectors in practice. Polynomial approximation methods are more economical in terms of storage. In terms of convergence, in section 4, we show that the convergence rate obtained by the Lanczos method is better than those reached by any polynomial (Chebyshev or Taylor series) approximations. Such a faster convergence comes from the fact that the Lanczos method applied to computing a bilinear form of a matrix admits a quadrature interpretation, where the weight function in the quadrature is matrix dependent. On the other hand, the convergence of polynomial approximation methods does not depend on the matrix. As a result it is easy to estimate a posteriori errors by analyzing only the function. For example, all that is needed to get the error for the exponential function is to have an idea of the error made in approximating the exponential by the given polynomial in an interval containing the spectrum of A. Such a posteriori error estimates do not require any computations with the matrix A. This is in contrast with the Lanczos approach for which such errors are generally not as straightforward. There are no known good extensions of the a posteriori error estimates given in [40] for the Lanczos approach to more general functions than the exponential. Finally, rational approximations (see, e.g., [24]) usually converge the fastest. However, a major disadvantage of this approach is that we need to solve a number of shifted linear systems. This is expensive, in general, and prohibitive in many realistic cases.

The polynomial approximation methods mentioned earlier may use several different strategies to obtain a good polynomial: Taylor series expansions [47], Chebyshev expansions [25], and least squares approximations [13]. The Taylor series approach converges too slowly and is usually not appealing. Chebyshev is a good choice in many scenarios but if the function has a steep derivative, then the expansion may need an extremely large number of terms to achieve a good approximation. In the extreme case, if there is a discontinuity (e.g., the sign/step function), then Chebyshev expansions exhibit the Gibbs phenomenon. The least squares approach [13, 12] addresses this issue by first approximating the function by using a spline, where more knots are placed around the areas with larger derivatives, and then in turn approximating the spline by a least squares polynomial. However, we show that the Lanczos method converges faster than any polynomial methods. Section 5 also illustrates the superior performance of the Lanczos method compared to the methods presented in [26, 47] via several numerical experiments.

Outline. The outline of the paper is as follows: section 2 is a discussion of the various applications that lead to estimating the trace of matrix functions. Section 3 describes the SLQ method in detail. A modified approach of the SLQ method that is more suitable for the Schatten norm estimation is also given. This alternate approach is appropriate when the input matrix has a large number of singular values close to zero, is nonsymmetric, or even rectangular. Section 4 lays out the theoretical analysis for the SLQ method. The analysis is applicable for any function that is analytic inside a closed interval and analytically continuable to an open Bernstein ellipse. We establish the approximation error bounds for the computation of different matrix function traces mentioned in section 2 using the SLQ method. Section 5 presents numerical experiments.

2. Applications.

This section is a brief survey of applications that require the computation of the trace of matrix functions. Such calculations arise in different ways in many disciplines and what follows is just a small set of representative applications. Much more information can be obtained by following the cited references.

2.1. Log-determinant.

As previously mentioned, the log-determinants have numerous applications in machine learning and related fields. 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 is equivalent to estimating the trace of the matrix function $f(A) = \log(A)$.

Suppose the positive definite matrix A has its eigenvalues inside the interval $[\lambda_{\min}, \lambda_{\max}]$, then the logarithm function $f(t) = \log(t)$ is analytic over this interval. When computing the log-determinant of a matrix, the case $\lambda_{\min} = 0$ is obviously excluded, where the function has its singularity. The Lanczos algorithm requires the input matrix to be symmetric. If A is nonsymmetric, we can either consider the matrix[1]

[1] The matrix product need not be formed explicitly since the Lanczos algorithm requires only matrix vector products.

$A^\top A$, since $\log\det(A^TA) = 2\log|\det(A)|$ or use the Golub–Kahan-bidiagonalization algorithm described later.

2.2. Log-likelihood. The problem of computing the likelihood function occurs in applications related to Gaussian processes [37]. Maximum likelihood estimation is a popular approach used for parameter estimation when high dimensional Gaussian models are used, especially in statistical machine learning. The objective in parameter estimation is to maximize the log-likelihood function with respect to a hyperparameter vector ξ:

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

where z is the data vector and $S(\xi)$ is the covariance matrix parameterized by ξ. The second term (log-determinant) in (2) can be computed by using the SLQ method. We observe that the first term in (2) resembles the quadratic form that appears in the trace estimator, and it can be also computed by using the Lanczos quadrature method. That is, we can estimate the term $z^\top S(\xi)^{-1}z$ using m steps of the Lanczos algorithm applied to $z/\|z\|$ as the starting vector, then compute the quadrature rule for the inverse function $f(t) = t^{-1}$, and rescale the result by $\|z\|^2$. In section 4, we give further details on this and present the error bounds for the log-likelihood function estimation by the SLQ method.

2.3. Computing the Schatten *p*-norms. Another important problem that arises in applications is the estimation of the nuclear norm and the Schatten p-norms of large matrices (a few applications were mentioned earlier). Given an input matrix $X \in \mathbb{R}^{d\times n}$, the nuclear norm of X is defined as $\|X\|_* = \sum_{i=1}^r \sigma_i$, where σ_i are the singular values of X and r is its rank. Suppose we define a positive semidefinite matrix A as $A = X^\top X$ or $A = XX^\top$. Then, the nuclear norm of X can be expressed as

$$\|X\|_* = \sum_{i=1}^r \sigma_i = \sum_{i=1}^r \sqrt{\lambda_i},$$

where the λ_i's are the eigenvalues of A. Hence, we can consider the symmetric positive semidefinite matrix $A = X^\top X$, and compute the nuclear norm of X as

$$\|X\|_* = \texttt{tr}(f(A)),\ f(t) = \sqrt{t}.$$

To estimate the above trace, we can invoke the SLQ method described in this work. Generally, the Schatten p-norm of a general matrix X is defined as

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

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 the SLQ method. Note that the functions $f(t) = t^{p/2}$ have a singularity at zero. Input matrices whose Schatten norms we seek are likely to have singular values equal or close to zero (low rank or numerically low rank). However, we explain in section 4.5 that such input matrices can be easily handled with a simple modification before applying SLQ.

2.4. Trace of a matrix inverse and the Estrada index. Other frequent matrix function trace estimation problems include estimating the trace of the matrix inverse and the Estrada index. As the name indicates, 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 $\lambda_{\min} > 0$.

Estimation of the Estrada index of graphs is popular in computational biology. This problem accounts for estimating the trace of the exponential function, i.e., $f(t) = \exp(t)$. Note that, here, the matrix A is the adjacency matrix of a graph, which need not be positive definite in general. However, the matrix $\exp(A)$ is always positive definite and our method and theory are applicable in this case. In addition, resolvent-based centrality measures (see [31, 3]) involve a resolvent matrix of the form $R(\alpha) = (I - \alpha A)^{-1}$, where α is a (small) parameter and in this context the matrix $R(\alpha)$ involved is always positive definite. The inverse, exponential, and resolvent functions, are analytic in the appropriate intervals of interest. Therefore, we can extend the analysis presented in this paper to obtain approximation error bounds.

2.5. Other applications. The SLQ method has been employed in the literature for a few related trace estimation problems before. One of the methods proposed by Ubaru, Saad, and Seghouane [44] for estimating the numerical rank of large matrices is equivalent to the SLQ method discussed here. The function f for this numerical rank estimation problem turns out to be a step function with a value of one above an appropriately chosen threshold. That is, the numerical rank of a matrix is the trace of an appropriate step function of the matrix. The article also proposes an approach to choosing this threshold based on the spectral density of the matrix.

An interesting related problem, which is mentioned in [25], is testing the positive definiteness of a matrix. This problem is also equivalent to estimating the trace of a step function of the matrix, with a value of one in a different interval. However, note that the step function has a discontinuity at the point of inflection (the point where it goes from zero to one) and hence we cannot directly apply the analysis developed in this paper. Also, the degree or the number of Lanczos steps required might be very high in practice. A workaround of this issue, proposed in [25] (also mentioned in [44]), is to first approximate the step function by a shifted and scaled hyperbolic tangent function of the form $\tilde{f}(t) = \frac{1}{2}(1 + \tanh(\alpha t))$, where α is an appropriately chosen constant, and then approximate the trace of this surrogate function $\tilde{f}(t)$.

Another problem where SLQ was previously used was in approximating the spectral density of a matrix [34]. The spectral density, also known as density of states of a matrix, is a probability density distribution that measures the likelihood of finding eigenvalues of the matrix at a given point on the real line. Being a distribution, the spectral density of a matrix can we written as a sum of delta functions of the eigenvalues of the matrix. That is, the spectral density is defined as

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

where δ is the Dirac distribution or Dirac δ-function. Lin, Saad, and Yang [34] demonstrated how the Lanczos algorithm can be used to approximately estimate the spectral density (equivalent to the SLQ method). The idea is to replace the delta function by a surrogate Gaussian blurring function. Then, the spectral density is approximated by estimating the trace of this blurring function using the Lanczos algorithm.

3. Stochastic Lanczos quadrature.

The Lanczos quadrature method was developed by Golub and his collaborators in a series of articles [21, 6, 7, 20]. The idea of combining the stochastic trace estimator with the Lanczos quadrature method appeared in [6, 7] for estimating the trace of the inverse and the determinant of matrices. Given a symmetric positive definite (SPD) matrix[2] $A \in \mathbb{R}^{n\times n}$, we wish to compute the trace of the matrix function $f(A)$, i.e., the expression given by (1), where we assume that the function f is analytic inside a closed interval containing the spectrum of A. To estimate the trace, we invoke the stochastic trace estimator [29], which is a Monte Carlo-type method that uses only matrix vector products. The attractiveness of this method is that it is inexpensive compared to the methods based on the computing of all eigenvalues of the matrix. The method estimates the trace $\mathtt{tr}(f(A))$ by generating random vectors $u_l, l = 1, \dots, \mathrm{n_v}$, with Rademacher distribution (vectors with ±1 entries of equal probability), forming unit vectors $v_l = u_l/\|u_l\|_2$, and then computing the average over the samples $v_l^\top f(A) v_l$:

$$\mathtt{tr}(f(A)) \approx \frac{n}{\mathrm{n_v}} \sum_{l=1}^{\mathrm{n_v}} v_l^\top f(A) v_l. \tag{3}$$

Hutchinson originally proposed to use vectors with ±1 entries of equal probability (Rademacher distribution) without scaling. It has since been shown that vectors from any other random distributions of zero mean and unit covariance also work [8, 5]. Strictly speaking, the prior results [5, 38] on which our bounds in section 4 are based, compute the approximation as $\|u_l\|_2^2 \cdot v_l^\top f(A) v_l$, rather than $n \cdot v_l^\top f(A) v_l$. However, for Rademacher vectors, $\|u_l\|_2^2 = n$. For other random vectors, in expectation, the two approaches are the same, as long as $\mathbb{E}[u_l u_l^\top] = I$. Hence, for computing the trace we only need to estimate the scalars of the form $v^\top f(A) v$, and the explicit computation of $f(A)$ is never needed.

The scalar (quadratic form) quantities $v^\top f(A) v$ are computed by transforming them to a Riemann–Stieltjes integral, and then employing the Gauss quadrature rule to approximate this integral. Consider the eigendecomposition of A as $A = Q\Lambda Q^\top$. Then, we can write the scalar product as

$$v^\top f(A) v = v^\top Q f(\Lambda) Q^\top v = \sum_{i=1}^{n} f(\lambda_i)\mu_i^2, \tag{4}$$

where μ_i are the components of the vector $Q^\top v$. The above sum can be considered as a Riemann–Stieltjes integral given by

$$I = v^\top f(A) v = \sum_{i=1}^{n} f(\lambda_i)\mu_i^2 = \int_a^b f(t) d\mu(t), \tag{5}$$

where the measure $\mu(t)$ is a piecewise constant function defined as

$$\mu(t) = \begin{cases} 0 \ \text{ if } t < a = \lambda_1, \\ \sum_{j=1}^{i-1} \mu_j^2 \ \text{ if } \lambda_{i-1} \le t < \lambda_i, \ i = 2, \dots, n, \\ \sum_{j=1}^{n} \mu_j^2 \ \text{ if } b = \lambda_n \le t, \end{cases} \tag{6}$$

[2] This matrix may be the sample covariance matrix of the input data matrix X, or may also be the form $X^\top X$ or $XX^\top$ for the given general rectangular matrix X.

assuming that the eigenvalues λ_i are ordered nondecreasingly. Next, the integral can be estimated using the Gauss quadrature rule [23]

$$\int_a^b f(t) d\mu(t) \approx \sum_{k=0}^{m} \omega_k f(\theta_k), \tag{7}$$

where $\{\omega_k\}$ are the weights and $\{\theta_k\}$ are the nodes of the $(m+1)$-point Gauss quadrature, which are unknowns and need to be determined. We wish to remark that the Riemann–Stieltjes integral, as well as the Gauss quadrature considered here, do not require A to be positive definite; see [20].

An elegant way to compute the nodes and the weights of the quadrature rule is to use the Lanczos algorithm [20]. For a given real 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* $\text{Span}\{w_0, Aw_0, \ldots, A^m w_0\}$ 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. For details, see [22]. 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 vectors w_k are orthonormal, and we can show that the Lanczos polynomials are orthogonal with respect to the measure $\mu(t)$ in (6); see [20, Theorem 4.2]. Therefore, the nodes and the weights of the quadrature rule in (7) can be computed as the eigenvalues and the squares of the first entries of the eigenvectors of T_{m+1}. Then, we can approximate the quadratic form (4) as

$$v^\top f(A) v \approx \sum_{k=0}^{m} \tau_k^2 f(\theta_k) \quad \text{with} \quad \tau_k^2 = \left[e_1^\top y_k\right]^2, \tag{8}$$

where $(\theta_k, y_k), k = 0, 1, \ldots, m$, are eigenpairs of T_{m+1} by using v as the starting vector w_0. Note that the above quadrature formula in (8) is equal to $e_1^T f(T_{m+1}) e_1$, i.e., $\sum_{k=0}^{m} \tau_k^2 f(\theta_k) = (f(T_{m+1}))_{1,1,}$. Using this expression we can compute the quadrature by other methods (depending on f) than the eigendecomposition; see, e.g., [28]. Thus, the trace of matrix function $f(A)$ can be computed as

$$\texttt{tr}(f(A)) \approx \frac{n}{\mathrm{n_v}} \sum_{l=1}^{\mathrm{n_v}} \left(\sum_{k=0}^{m} (\tau_k^{(l)})^2 f(\theta_k^{(l)}) \right) = \frac{n}{\mathrm{n_v}} \sum_{l=1}^{\mathrm{n_v}} \left(f(T_{m+1}^{(l)}) \right)_{1,1}, \tag{9}$$

where $(\theta_k^{(l)}, \tau_k^{(l)}), k = 0, 1, \ldots, m$, are eigenvalues and the first entries of the eigenvectors of the tridiagonal matrix $T_{m+1}^{(l)}$ corresponding to the starting vectors $v_l, l = 1, \ldots, \mathrm{n_v}$. This method is far less costly than computing the eigenvalues of the matrix A for the purpose of computing the trace via (1). The SLQ algorithm corresponding to this procedure is summarized in Algorithm 1.

Algorithm 1 Trace of a matrix function by SLQ using the Lanczos algorithm.

```
Input: SPD matrix $A \in \mathbb{R}^{n\times n}$, function $f$, degree $m$, and $\mathrm{n_v}$.
Output: Approximate trace $\Gamma$ of $f(A)$.
for $l = 1$ to $\mathrm{n_v}$ do
    1. Generate a Rademacher random vector $u_l$ and form unit vector $v_l = u_l/\|u_l\|_2$
    2. $T = \mathrm{Lanczos}(A, v_l, m+1)$; that is, apply $m+1$ steps of Lanczos to $A$ with $v_l$
    as the starting vector.
    3. $[Y, \Theta] = \mathtt{eig}(T)$ and compute $\tau_k = [e_1^\top y_k]$ for $k = 0, \dots, m$
    4. $\Gamma \leftarrow \Gamma + \sum_{k=0}^{m} \tau_k^2 f(\theta_k)$.
end for
Output $\Gamma = \frac{n}{\mathrm{n_v}}\Gamma$.
```

In section 4, we establish error bounds for this approach for functions analytic inside a closed interval. We show that the convergence rate of quadrature methods is faster than other polynomial expansion methods, e.g., Chebyshev approximation.

Golub–Kahan bidiagonalization. In computing Schatten p-norms, when the input matrix X has a large number of singular values close to zero, the Lanczos algorithm might encounter numerical issues. In such scenarios, it is advantageous to use the Golub–Kahan bidiagonalization (G-K-B) algorithm [19] on X in place of the Lanczos algorithm on $A = X^\top X$ or $A = XX^\top$. For the connections between the two algorithms, see, e.g., [20]. Suppose B_{m+1} is the bidiagonal matrix obtained by the G-K-B algorithm, then the matrix $T_{m+1} = B_{m+1}^\top B_{m+1}$ will be the Lanczos Jacobi matrix corresponding to $X^\top X$ [20]. The singular values ϕ_k of B_{m+1} are such that $\phi_k = \sqrt{\theta_k}$ for $k = 0, \dots, m$, where θ_k are the eigenvalues of T_{m+1}. Thus, the Shatten p-norms (corresponding to the pth power of the square root function) can be computed using the singular values of the bidiagonal matrix B_{m+1} obtained from m steps of the G-K-B algorithm. Similarly, traces of functions of non-Hermitian matrices can also be computed using this algorithm. Algorithm 2 presents a version of the SLQ method that uses the G-K-B bidiagonalization.

Algorithm 2 Trace of a matrix function by SLQ using the G-K-B algorithm.

```
Input: $X \in \mathbb{R}^{d\times n}$, function $f$ (with $A = X^\top X, \tilde{f} : \tilde{f}(t) = f(t^2)$), $m$, and $\mathrm{n_v}$.
Output: Approximate trace $\Gamma$ of $f(A)$.
for $l = 1$ to $\mathrm{n_v}$ do
    1. Generate a Rademacher random vector $u_l$ and form unit vector $v_l = u_l/\|u_l\|_2$
    2. $B = \mathrm{GKB}(X, v_l, m+1)$; that is, apply $m+1$ steps of GKB to $X$ with $v_l$ as
    the starting vector.
    3. $[U, \Phi] = \mathtt{svd}(B)$ and compute $\tau_k = [e_1^\top u_k]$ for $k = 0, \dots, m$
    4. $\Gamma \leftarrow \Gamma + \sum_{k=0}^{m} \tau_k^2 \tilde{f}(\phi_k)$.
end for
Output $\Gamma = \frac{n}{\mathrm{n_v}}\Gamma$.
```

Computational cost. Since we apply m steps of Lanczos or the G-K-B algorithm for $\mathrm{n_v}$ different starting vectors, the cost of the SLQ method will be $O((\mathrm{nnz}(A)m + nm^2)\mathrm{n_v})$, where $\mathrm{nnz}(A)$ is the number of nonzeros in A. The additional cost $O(nm^2)$ is the orthogonalization cost inside the Lanczos algorithm. Assuming full reorthogonalization, if we choose degree m, then we need to reorthogonalize m vectors of length

n. Typically both m and $\mathrm{n_v}$ are much smaller than the matrix dimension n. Hence, the method will be very inexpensive for large sparse matrices.

The Lanczos algorithm has an additional storage cost compared to polynomial approximation methods. We need to store the orthogonalized vectors of the Krylov subspace inside the Lanczos algorithm. This storage depends on whether partial or full reorthogonalization is used. However, since the degree m is very small, we can use full orthogonalization inside the Lanczos algorithm and this additional storage cost will be negligible. That is, at each step of the algorithm, the new vector is orthogonalized with respect to all the previous Lanczos vectors, which requires storing of these vectors for orthogonalization. Note that, under exact arithmetic there is no need for reorthogonalization, but due to numerical issues a partial or full reorthogonalization is needed in practice.

The computations for both the Chebyshev and the Lanczos methods can be done in parallel across the different starting vectors. This is an obvious coarse-grained parallelism but it also is the most effective in practice. The use of MPI will be helpful here and communication is minimal. Finer-grained parallelism can also happen within each starting vector (such as in a threaded implementation). In this case, Chebyshev is advantageous, as it requires no (or very few) global reductions. Communication takes place only for matrix-vector multiplications and it is mostly local for sparse matrices. Chebyshev does not need vector norm computation, and hence there is no global synchronization. Lanczos, on the other hand, needs global synchronization for computing inner products, norms, and for the reorthogonalization. However, since the number of Lanczos steps required (degree m in both cases) is small, such finer-grained parallelism is typically not necessary.

4. Analysis.

In this section, we present multiplicative error bounds for approximating the trace of a matrix function using SLQ. Additive error bounds are also established for the log-determinant approximation of a positive definite matrix and the log-likelihood function estimation. The nuclear norm and Schatten-p norms estimation of a general matrix is discussed in the latter part of the section. First, we give the following definition: a Bernstein ellipse E_ρ is an ellipse on the complex plane with focii at $-1, 1$ and major semiaxis $(\rho + \rho^{-1})/2$ with $\rho > 1$ [42]. It can be viewed as a mapping of the circle $C(0, \rho)$ (center at zero and radius ρ) using the Joukowsky transform $(z + z^{-1})/2$. Hence we can have two values of ρ that are inverses of each other, which give the same ellipse. Following is our main result.

THEOREM 4.1. *Consider an SPD matrix $A \in \mathbb{R}^{n\times n}$ with eigenvalues in $[\lambda_{\min}, \lambda_{\max}]$ and condition number $\kappa = \lambda_{\max}/\lambda_{\min}$. Let f be a function analytic in $[\lambda_{\min}, \lambda_{\max}]$ and be either positive or negative (i.e., does not cross zero) inside this interval. Denote by m_f the absolute minimum value of f in the interval. Assume that f is analytically continuable in an open Bernstein ellipse E_ρ encompassing the interval, with foci $\lambda_{\min}, \lambda_{\max}$ and sum of the two semiaxes ρ, such that $|f(z)| \leq M_\rho$ for all $z \in E_\rho$. Let ε, η be constants in $(0, 1)$. Then for SLQ parameters satisfying*

- $m \geq \frac{1}{2}\log(\frac{4M_\rho(\lambda_{\max}-\lambda_{\min})}{\varepsilon m_\rho(\rho^2-1)})/\log(\rho)$ *number of Lanczos steps, and*
- $\mathrm{n_v} \geq (24/\varepsilon^2)\log(2/\eta)$ *number of starting Rademacher vectors,*

the output Γ of the SLQ method is such that

$$\Pr\Bigg[|\mathtt{tr}(f(A)) - \Gamma| \leq \varepsilon\,|\mathtt{tr}(f(A))|\Bigg] \geq 1 - \eta. \tag{10}$$

In particular for $\rho = (\sqrt{\kappa}+1)/(\sqrt{\kappa}-1)$ for which the function of interest is analytic inside E_ρ, we have $m \geq (\sqrt{\kappa}/4)\log(K/\varepsilon)$ with $K = (\lambda_{\max}-\lambda_{\min})(\sqrt{\kappa}-1)^2M_\rho/(\sqrt{\kappa}m_f)$.

To prove the theorem, we first derive error bounds for the Lanczos quadrature approximation (which gives the convergence rate), using the facts that an $(m+1)$-point Gauss quadrature rule is exact for any $2m+1$ degree polynomial and that the function is analytic inside an interval and is analytically continuable in a Bernstein ellipse. We then combine this bound with the error bounds for the stochastic trace estimator to obtain the above result.

4.1. Convergence rate for the Lanczos quadrature.

In order to prove Theorem 4.1, we first establish the convergence rate for the Lanczos quadrature approximation of the quadratic form. Recall that the quadratic form $v^\top f(A)v$ can be written as a Riemann–Stieltjes integral I, as given in (5). Let I_m denote the $(m+1)$-point Gauss quadrature rule that approximates the integral I, given by

$$I_m = \sum_{k=0}^{m} \omega_k f(\theta_k),$$

where $\{\omega_k\}$ are the weights and $\{\theta_k\}$ are the nodes, computed by using $m+1$ steps of the Lanczos algorithm. The well-known error analysis for the Gauss quadrature rule is given by [20],

$$|I - I_m| = \frac{f^{(2m+2)}(\zeta)}{(2m+2)!} \int_a^b \left[\prod_{k=0}^{m} (t-\theta_k) \right]^2 d\mu(t) \tag{11}$$

for some $a < \zeta < b$. However, this analysis might not be useful for our purpose, since the higher derivatives of both the logarithm and the square root function become excessively large in the interval of interest. Hence, in this work, we establish improved error analysis for the Lanczos quadrature approximations, using some classical results developed in the literature, with the fact that functions of interest are analytic over a certain interval. We begin with the following result.

THEOREM 4.2. *Let a function g be analytic in $[-1,1]$ and analytically continuable in the open Bernstein ellipse E_ρ with foci ±1 and sum of major and minor axis equal to $\rho > 1$, where it satisfies $|g(z)| \le M_\rho$. Then the $(m+1)$-step Lanczos quadrature approximation satisfies*

$$|I - I_m| \le \frac{4M_\rho}{(\rho^2-1)\rho^{2m}}. \tag{12}$$

Proof. We follow a similar argument to that developed in [36] that estimates the error of Gaussian quadratures for a Riemann integral. The result and the proof strategy are usually covered in standard textbooks, e.g., [42, Thm. 19.3]. In our case, the integral is a Riemann–Stieltjes integral with respect to a specific measure given in (6). As a result, the bound admits the same rate but with a different constant.

For the given function g that is analytic over the interval $[-1,1]$, consider the $2m+1$ degree Chebyshev polynomial approximation of $g(t)$, i.e.,

$$P_{2m} = \sum_{j=0}^{2m+1} a_j T_j(t) \approx g(t).$$

We know that the $(m+1)$-point Gauss quadrature rule is exact for any polynomial of degree up to $2m+1$; see [20, Thm. 6.3] or [42, Thm. 19.1]. This can also be deduced from the error term in (11). Hence, the error in integrating g is the same as the error in integrating $g - P_{2m}$. Thus, we have

$$\begin{aligned}|I - I_m| &= |I(g - P_{2m}) - I_m(g - P_{2m})| \leq |I(g - P_{2m})| + |I_m(g - P_{2m})| \\ &= \left| I\left(\sum_{j=2m+2}^{\infty} a_j T_j(t)\right)\right| + \left| I_m\left(\sum_{j=2m+2}^{\infty} a_j T_j(t)\right)\right| \\ &\leq \sum_{j=2m+2}^{\infty} |a_j| \Big[|I(T_j)| + |I_m(T_j)| \Big].\end{aligned}$$

Next, we obtain bounds for the three terms inside the summation above.

If the function g is analytic in $[-1, 1]$ and analytically continuable in the Bernstein ellipse E_ρ, then for the Chebyshev coefficients we have from [42, Theorem 8.1] and [36, (14)],

$$|a_j| \leq \frac{2M_\rho}{\rho^j}.$$

Next, for the quadrature rule $I_m(T_j)$, we have

$$I_m(T_j) = \sum_{k=0}^{m} \tau_k^2 T_j(\theta_k) \leq \sum_{k=0}^{m} |\tau_k^2||T_j(\theta_k)| \leq 1.$$

The last inequality results from the fact that, for $f(t) = 1$, the quadrature rule is exact, and the thus the integral is equal to 1 ($v_l^\top f(A) v_l = v_l^\top v_l = 1$). Therefore, the weights τ_k^2 must sum to 1. The maximum value of T_j inside the interval is 1. Finally, in order to bound the Riemann–Stieltjes integral $I(T_j)$, we use the following:

$$I(T_j) = v^\top T_j(A) v \leq \lambda_{\max}(T_j(A)) = 1$$

by the min-max theorem and $\|v\| = 1$. Therefore,

$$|I - I_m| \leq \sum_{j=2m+2}^{\infty} \frac{2M_\rho}{\rho^j}[1 + 1].$$

Since the Gauss quadrature rule is a symmetric rule [36], the error in integration of $T_j(t)$ for any odd j will be equal to zero. Thus, we get the result in the theorem

$$|I - I_m| \leq \frac{4M_\rho}{(\rho^2 - 1)\rho^{2m}}.$$

□

Remark 1. The convergence rate for the Chebyshev polynomial approximation of an analytic function is $O(1/\rho^m)$; see [42, Theorem 8.2]. Hence, the Lanczos quadrature approximation is twice as fast as the Chebyshev approximation. Moreover, it is known that the Gauss quadrature has the maximal polynomial order of accuracy [42].

Theorem 4.2 holds for functions that are analytic over $[-1, 1]$. The functions considered in this paper such as logarithm, exponential, and square root functions are analytic over $[\lambda_{\min}, \lambda_{\max}]$ for $\lambda_{\min} > 0$. Hence, we need to use the following transform to get the right interval.

If $f(x)$ is analytic on $[\lambda_{\min}, \lambda_{\max}]$, then

$$g(t) = f\left[\left(\frac{\lambda_{\max} - \lambda_{\min}}{2}\right) t + \left(\frac{\lambda_{\max} + \lambda_{\min}}{2}\right)\right]$$

is analytic on $[-1, 1]$. If we denote the error in the quadrature rule for approximating the integral of function f as $E(f)$, then we have

$$E(f) = \left(\frac{\lambda_{\max} - \lambda_{\min}}{2}\right) E(g).$$

The function g will have its singularity at $t_0 = \alpha = -\frac{\kappa+1}{\kappa-1}$. Hence, we choose the ellipse E_ρ with the semimajor axis length of $|\alpha|$, where g is analytic inside. Then, the convergence rate ρ will be

$$\rho = \alpha \pm \sqrt{\alpha^2 - 1} = \frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1} > 1.$$

The sign is chosen such that $\rho > 1$. From Theorem 4.2, the error $E(g) \le 4M_\rho/[(\rho^2-1)\rho^{2m}]$, where $|g(z)| < M_\rho$ inside E_ρ. Hence, the error $E(f)$ will be

$$E(f) = \left(\frac{\lambda_{\max} - \lambda_{\min}}{2}\right) \frac{4M_\rho}{(\rho^2-1)\rho^{2m}} = \frac{(\lambda_{\max} - \lambda_{\min})(\sqrt{\kappa}-1)^2 M_\rho}{2\sqrt{\kappa}\rho^{2m}}$$

with ρ defined as above. Thus, for a function f that is analytic on $[\lambda_{\min}, \lambda_{\max}]$ and $C_\rho = 2M_\rho(\lambda_{\max} - \lambda_{\min})/(\rho^2-1) = (\lambda_{\max} - \lambda_{\min})(\sqrt{\kappa}-1)^2 M_\rho/(2\sqrt{\kappa})$, we have

$$\left| v^\top f(A) v - \sum_{k=0}^{m} \tau_k^2 f(\theta_k) \right| \le \frac{C_\rho}{\rho^{2m}}. \tag{13}$$

4.2. Approximation error of the trace estimator. The quadratic form $v^\top f(A) v$ for which we derived the error bounds in the previous section comes from the Hutchinson trace estimator. Let us denote this estimator as $\mathtt{tr}_{\mathrm{n_v}}(A) = \frac{n}{\mathrm{n_v}} \sum_{l=1}^{\mathrm{n_v}} v_l^\top A v_l$. The convergence analysis for the stochastic trace estimator was developed in [5], and improved in [38] for sample vectors with different probability distributions. We state the following theorem which is proved in [38].

THEOREM 4.3. *Let A be an $n \times n$ symmetric positive semidefinite matrix and $v_l, l = 1, \dots, \mathrm{n_v}$, be random starting vectors sampled from the Rademacher distribution and scaled to a unit 2-norm. Then, with $\mathrm{n_v} \ge (6/\varepsilon^2)\log(2/\eta)$, we have*

$$\Pr\left[|\mathtt{tr}_{\mathrm{n_v}}(A) - \mathtt{tr}(A)| \le \varepsilon |\mathtt{tr}(A)|\right] \ge 1 - \eta.$$

The above theorem can be used to bound the trace of any matrix function $f(A)$, if the function is either positive or negative inside the spectrum interval. Therefore, the theorem holds for the square root function, its powers, and the exponential. However, for the logarithm function, different scenarios occur depending on the spectrum, which will be discussed later. Let Γ be the output of the SLQ method to estimate the trace of such functions, given by

$$\Gamma = \frac{n}{\mathrm{n_v}} \sum_{l=1}^{\mathrm{n_v}} \left(\sum_{k=0}^{m} (\tau_k^{(l)})^2 f(\theta_k^{(l)}) \right). \tag{14}$$

We need the following lemma.

LEMMA 4.4. *Let $A \in \mathbb{R}^{n \times n}$ be an SPD matrix with eigenvalues in $[\lambda_{\min}, \lambda_{\max}]$ and condition number $\kappa = \lambda_{\max}/\lambda_{\min}$, and f be an analytic function in this interval* *with $|f(z)| \leq M_\rho$ for all z inside a Bernstein ellipse E_ρ that encompasses the interval. Then, the following inequality holds:*

$$|\mathtt{tr}_{\mathrm{n_v}}(f(A)) - \Gamma| \leq \frac{nC_\rho}{\rho^{2m}},$$

where $\rho = (\sqrt{\kappa}+1)/(\sqrt{\kappa}-1)$ and $C_\rho = 2M_\rho(\lambda_{\max} - \lambda_{\min})/(\rho^2-1) = (\lambda_{\max} - \lambda_{\min})(\sqrt{\kappa}-1)^2 M_\rho/(2\sqrt{\kappa})$.

Proof. The lemma follows from (13). We have

$$\begin{aligned}|\mathtt{tr}_{\mathrm{n_v}}(f(A)) - \Gamma| &= \frac{n}{\mathrm{n_v}} \left| \sum_{l=1}^{\mathrm{n_v}} v_l^\top f(A) v_l - \sum_{l=1}^{\mathrm{n_v}} I_m^{(l)} \right| \\ &\leq \frac{n}{\mathrm{n_v}} \sum_{l=1}^{\mathrm{n_v}} \left| v_l^\top f(A) v_l - I_m^{(l)} \right| \\ &\leq \frac{n}{\mathrm{n_v}} \sum_{l=1}^{\mathrm{n_v}} \frac{C_\rho}{\rho^{2m}} = \frac{nC_\rho}{\rho^{2m}}. \qquad \square\end{aligned}$$

Now, we are ready to prove Theorem 4.1. Based on the condition of m,

$$\log\frac{K}{\varepsilon} \leq \frac{4m}{\sqrt{\kappa}} \leq 2m \log\left(\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1}\right).$$

Therefore,

$$\frac{K}{\varepsilon} \leq \rho^{2m} \quad \text{and hence} \quad \frac{C_\rho}{\rho^{2m}} \leq \frac{\varepsilon}{2} f_{\min}(\lambda),$$

where $f_{\min}(\lambda) \equiv m_f$ is the absolute minimum of the function in the interval $[\lambda_{\min}, \lambda_{\max}]$. This gives us the lower bound on the degree m in the theorem. Then, from Lemma 4.4 we have

$$|\mathtt{tr}_{\mathrm{n_v}}(f(A)) - \Gamma| \leq \frac{\varepsilon n}{2} f_{\min}(\lambda) \leq \frac{\varepsilon}{2}|\mathtt{tr}(f(A))|. \tag{15}$$

From Theorem 4.3, we have

$$\Pr\left[|\mathtt{tr}(f(A)) - \mathtt{tr}_{\mathrm{n_v}}(f(A))| \leq \frac{\varepsilon}{2}|\mathtt{tr}(f(A))|\right] \geq 1-\eta. \tag{16}$$

Combining the above two inequalities (15) and (16) leads to the result in Theorem 4.1:

$$\begin{aligned}1-\eta &\leq \Pr\left[|\mathtt{tr}(f(A)) - \mathtt{tr}_{\mathrm{n_v}}(f(A))| \leq \frac{\varepsilon}{2}|\mathtt{tr}(f(A))|\right] \\ &\leq \Pr\left[|\mathtt{tr}(f(A)) - \mathtt{tr}_{\mathrm{n_v}}(f(A))| + |\mathtt{tr}_{\mathrm{n_v}}(f(A)) - \Gamma| \leq \frac{\varepsilon}{2}|\mathtt{tr}(f(A))| + \frac{\varepsilon}{2}|\mathtt{tr}(f(A))|\right] \\ &\leq \Pr\left[|\mathtt{tr}(f(A)) - \Gamma| \leq \varepsilon\,|\mathtt{tr}(f(A))|\right].\end{aligned}$$

For comparison, note that for Chebyshev approximations [26], the required degree of the polynomial is $m = \Theta(\sqrt{\kappa}\log\frac{\kappa}{\varepsilon})$ and for Taylor approximations [9], $m = O(\kappa\log\frac{\kappa}{\varepsilon})$. Recall from Remark 1, the Lanczos algorithm is superior to the Chebyshev expansions because the former approximation converges twice as fast as does the latter. Clearly, the Lanczos approximation also converges faster than the Taylor approximation. Theorem 4.1 can be used to establish the error bounds for approximating the log-determinants and the Schatten p-norms. The quality and the complexity of the algorithms depend on the condition number κ, since matrix function approximations become harder when matrices become more ill-conditioned, which requires higher degree approximations.

4.3. Bounds for log-determinant. For the logarithm function, we encounter three different scenarios depending on the spectrum of the matrix. The first case is when $\lambda_{\max} < 1$, $\log(A)$ is negative definite, and the log-determinant will always be negative. Thus, the conditions of Theorem 4.1 are satisfied. Similarly, Theorem 4.1 holds in the second case when $\lambda_{\min} > 1$, since $\log(A)$ is positive definite. In the third case when $\lambda_{\min} < 1$ and $\lambda_{\max} > 1$, however, we cannot obtain multiplicative error bounds of the form given in Theorem 4.1, since the log function will cross zero inside the interval. In the worst case, the log-determinant can be zero. One simple workaround to avoid this case is to scale the matrix such that its eigenvalues are either all smaller than 1 or all greater than 1; however, such an approach requires the computation of the extreme eigenvalues of A. The following corollary gives additive error bounds without scaling; it holds for any SPD matrix.

COROLLARY 4.5. *Given $\varepsilon, \eta \in (0,1)$, an SPD matrix $A \in \mathbb{R}^{n\times n}$ with its eigenvalues in $[\lambda_{\min}, \lambda_{\max}]$, and condition number $\kappa = \lambda_{\max}/\lambda_{\min}$ for SLQ parameters*

- *$m \geq (\sqrt{3\kappa}/4)\log(K_1/\varepsilon)$ number of Lanczos steps, and*
- *$\mathrm{n_v} \geq (24/\varepsilon^2)\log(1+\kappa))^2\log(2/\eta)$ number of starting vectors,*

where $K_1 = 5\kappa\log(2(\kappa+1))/\sqrt{2\kappa+1}$, we have

$$\Pr\left[|\log\det(A) - \Gamma| \leq \varepsilon n\right] \geq 1-\eta, \tag{17}$$

where Γ is the output of the SLQ method for log-determinant computation.

Proof. The proof of the corollary is on similar lines to the proof of Theorem 4.1. In the logarithm case, Theorem 4.2 still holds, however, we need to choose a smaller ellipse (smaller α) since the log function goes to infinity near the singularity. We choose $\alpha = (\kappa+1)/\kappa$, then $\rho = (\sqrt{2\kappa+1}+1)/(\sqrt{2\kappa+1}-1)$. For Theorem 4.3, we consider the fact that, if $B = \frac{A}{\lambda_{\max}+\lambda_{\min}}$, then

$$\log\det A = \log\det B + n\log(\lambda_{\max}+\lambda_{\min}).$$

Since the matrix B has its eigenvalues inside $(0,1)$, the logarithm function is negative and we hence can apply Theorem 4.3 with $f(A) = \log(\frac{A}{\lambda_{\max}+\lambda_{\min}})$, and then add and subtract $n\log(\lambda_{\max}+\lambda_{\min})$ to get an inequality of the form (16). To compute the parameters in Theorem 4.2, we consider this function $f(t) = \log(\frac{t}{\lambda_{\max}+\lambda_{\min}})$, and the ellipse E_ρ where the function is analytic with ρ as defined above. Then, we have $\rho^2 - 1 = (4\sqrt{2\kappa+1})/(\sqrt{2\kappa+1}-1)^2$ and M_ρ is computed as

$$\begin{aligned}\max_{z\in E_\rho}|\log(z)| &\leq \max_{z\in E_\rho}\sqrt{(\log|z|)^2+\pi^2}\\ &= \sqrt{(\log|1/2\kappa|)^2+\pi^2} \leq 5\log(2(\kappa+1)) = M_\rho.\end{aligned}$$

The first inequality comes from the fact $|\log(z)| = |\log|z| + i\arg(z)| \leq \sqrt{(\log|z|)^2+\pi^2}$. The ellipse E_ρ is defined with foci at $1/(\kappa+1)$ and $\kappa/(\kappa+1)$. The maximum occurs at endpoint $z_0 = 1/(2\kappa)$. As in the proof of Theorem 4.1, we have

$$E(f) = \left(\frac{\kappa-1}{\kappa+1}\right)\frac{2M_\rho}{(\rho^2-1)\rho^{2m}} \leq \frac{5\kappa\log(2(\kappa+1))}{2\sqrt{2\kappa+1}\rho^{2m}}.$$

The K_1 value is obtained by setting

$$\frac{n5\kappa\log(2(\kappa+1))}{2\sqrt{2\kappa+1}\rho^{2m}} \leq \frac{\varepsilon n}{2}.$$

The lower bound for m_f is simplified using the fact $\sqrt{2\kappa+1} \leq \sqrt{3\kappa}$. We can then conclude the corollary using $|\log\det B| \leq n\log(1+\kappa)$ and choosing $\varepsilon = \varepsilon/\log(1+\kappa)$ in Theorem 4.3. ∎

4.4. Bounds for log-likelihood function. Recall the log-likelihood function defined in (2). The log-determinant term in it can be bounded as above. The first term $z^\top S(\xi)^{-1}z$ is computed using the Lanczos quadrature method with $z/\|z\|$ as the starting vector for the Lanczos algorithm. The following corollary gives the error bound for the log-likelihood function estimation by SLQ, which follows from Theorem 4.1 and Corollary 4.5.

COROLLARY 4.6. *Given a data vector $z \in \mathbb{R}^n$, a covariance matrix $S(\xi) \in \mathbb{R}^{n\times n}$ with hyperparameter ξ and its eigenvalues in $[\lambda_{\min}, \lambda_{\max}]$, and constants $\varepsilon, \eta \in (0,1)$ for SLQ parameters*

- $m_1 \geq (\sqrt{3\kappa}/4)\log(K_1/\varepsilon)$, $m_2 \geq (\sqrt{3\kappa}/4)\log(K_2/\varepsilon)$ *and*
- $\mathrm{n_v} \geq (24/\varepsilon^2)(\log(1+\kappa))^2\log(2/\eta)$,

where K_1 is defined in Corollary 4.5 *and* $K_2 = \|z\|^2(\kappa-1)(\sqrt{2\kappa-1}-1)^2/\sqrt{2\kappa-1}$, *we have*

$$\Pr\Big[|\log p(z \mid \xi) - \Gamma| \leq \varepsilon(n+1)\Big] \geq 1-\eta, \tag{18}$$

where $\Gamma = -\Gamma_1 - \Gamma_2 - \frac{n}{2}\log(2\pi)$, Γ_1 *is the output of SLQ with parameters* m_1 *and* $\mathrm{n_v}$, *and* Γ_2 *is the output of the Lanczos quadrature method for approximating* $z^\top S(\xi)^{-1}z$ *with* m_2 *steps of Lanczos and scaled by* $\|z\|^2$.

Proof. To prove the corollary, we obtain bounds for the two quantities Γ_1 and Γ_2. We bound the log-determinant term Γ_1 obtained by SLQ using Corollary 4.5. Bounds of Γ_2, the Lanczos quadrature approximation of $z^\top S(\xi)^{-1}z$, can be computed using Theorem 4.2 as follows. We again need to choose a smaller ellipse E_ρ where the function is analytic, since the function $f(t) = t^{-1}$ also goes to infinity near singularity. We set $\alpha = \kappa/(\kappa-1)$, then $\rho = (\kappa + \sqrt{2\kappa-1})/(\kappa-1)$ and $\rho^2 - 1 = (4\sqrt{2\kappa-1})/(\sqrt{2\kappa-1}-1)^2$. For the inverse function, the maximum must occur on the real line, particularly at $-\alpha$ for $g(z)$ or at $\lambda_{\min}/2$ for $f(z)$, so, $M_\rho = 2/\lambda_{\min}$. Then,

$$E(f) = \frac{(\kappa-1)(\sqrt{2\kappa-1}-1)^2}{\sqrt{2\kappa-1}\rho^{2m}}.$$

We will have a scaling $\|z\|^2$. We get the bounds by setting $\|z\|^2E(f) \leq \varepsilon$. ∎

We can also compute the error bounds for approximating the trace of matrix inverse by SLQ using the above proof.

4.5. Schatten *p*-norms estimation. When estimating the nuclear and Schatten p-norms, we encounter the following issue when approximating the square root function. *In order to obtain strong theoretical results (exponential convergence) for a given function $f(t)$, the function must be analytic in the spectrum interval. However, the square root function is nondifferentiable at $t = 0$.* This will be a major stumbling block for rank-deficient matrices since the interval of eigenvalues now contains zero.

Shifting the spectrum. To overcome the issue, we propose the following remedy, which is based on the key observation, proper to the computation of the nuclear norm, that the small and zero singular values do not contribute much to the norm itself. In other words, the nuclear norm of a matrix depends mainly on the top singular values.

The idea is then to shift the spectrum of the matrix by a small $\delta > 0$ such that no eigenvalues of the matrix A are equal to zero. That is, we replace A by $A + \delta I$, such that the eigenvalues of the new shifted matrix are $\lambda_i + \delta$. For the square root function, the error is given by,

$$\sqrt{\lambda_i + \delta} - \sqrt{\lambda_i} = \frac{\delta}{\sqrt{\lambda_i + \delta} + \sqrt{\lambda_i}}.$$

Hence, the error in the large eigenvalues will be small. The error in the nuclear norm will be

$$\sum_{i=1}^{n} \sqrt{\lambda_i + \delta} - \sum_{i=1}^{n} \sqrt{\lambda_i} = \sum_{i=1}^{n} \frac{\delta}{\sqrt{\lambda_i + \delta} + \sqrt{\lambda_i}}. \tag{19}$$

For the shifted matrix, the eigenvalues will be in the interval $[\delta, \lambda_{\max} + \delta]$. Now, Theorem 4.1 holds in this interval (square root function will be positive) and we can obtain the approximation error bounds. The error due to shifting is small, and can also be corrected using the Taylor series expansion of the square root function (details omitted). Algorithm 2 will be better suited for nuclear norm estimation.

Bounds for Schatten p-norms. To summarize, if the input matrix has full rank ($\lambda_{\min} > 0$), then Theorem 4.1 is directly applicable, since the square root function is positive and analytic in the interval. For rank deficient matrices (has zero singular values), we will encounter the above problem, and we need to shift the spectrum by δ. From (19), we can upper bound the error due to shifting by $n\sqrt{\delta}$. Thus, the shift δ is chosen such that this error due to shifting is at most $\varepsilon \|X\|_p^p$. Here, the value of $\|X\|_p$ can be taken to be roughly poly(n). We can then compute $\|X\|_p$ of the shifted matrix using SLQ. We have the following general result.

COROLLARY 4.7. *Given $\varepsilon, \eta \in (0,1)$, a matrix $X \in \mathbb{R}^{d \times n}$ with its singular values in $[\sigma_{\min}, \sigma_{\max}]$, we consider the SPD matrix $A = X^T X$ (not formed explicitly) with its condition number $\kappa = \sigma_{\max}^2/\sigma_{\min}^2$ for SLQ parameters*

- *$m \geq (\sqrt{\kappa}/4)\log(K_3/\varepsilon)$ number of Lanczos steps, and*
- *$\mathrm{n_v} \geq (24/\varepsilon^2)\log(2/\eta)$ number of starting vectors,*

where $K_3 = \frac{\sigma_{\min}^2(\kappa+1)^{p/2}(\kappa^2-1)}{\sqrt{\kappa}}$, we have

$$\Pr\left[\left|\|X\|_p^p - \Gamma^p\right| \leq \varepsilon \|X\|_p^p\right] \geq 1 - \eta, \tag{20}$$

where Γ is the output of the SLQ method for Schatten p norm computation.

Proof. Theorem 4.1 gives the above error bound. We consider the function $f(t) = t^{p/2}$ applied to $A = X^T X$ (not formed explicitly). We can also consider the G-K-B algorithm. We choose $\rho = (\sqrt{\kappa} + 1)/(\sqrt{\kappa} - 1)$ as in the theorem since $g(t)$ is analytic inside the ellipse E_ρ. Then, $m_f = \sigma_{\min}^p$ and $M_\rho = (\sigma_{\max}^2 + \sigma_{\min}^2)^{p/2}$, since the maximum occurs at the right end of the ellipse. Substituting these values in the theorem, we get the above result. □

Comparison of bounds. Here we compare the theoretical results of our SLQ method with the Chebyshev and the Taylor methods. Table 1 lists the theoretical worst case degree m, the computational costs, and the memory requirements for the three methods along with the Cholesky factorization method. Here T_{ext} refers to the cost to compute the extreme eigenvalues of the matrix.

Table 1
Computational cost and memory requirements.

Method	Degree m	Cost	Memory
SLQ	$\geq (\frac{\sqrt{\kappa}}{4} \log \frac{K}{\varepsilon})$	$O((\mathrm{nnz}(A)m + nm^2)\mathrm{n_v})$	$O(\mathrm{nnz}(A) + nm^2)$
Chebyshev	$\geq (\sqrt{\kappa} \log \frac{K}{\varepsilon})$	$O(\mathrm{nnz}(A)m\mathrm{n_v}) + T_{ext}$	$O(\mathrm{nnz}(A) + n)$
Taylor	$O(\kappa \log \frac{\kappa}{\varepsilon})$	$O(\mathrm{nnz}(A)m\mathrm{n_v}) + T_{ext}$	$O(\mathrm{nnz}(A) + n)$
Cholesky	-	$O(n^3)$	$O(n^2)$

As mentioned earlier, SLQ has an improved dependency on the condition number of the matrix compared to other methods, i.e., the theoretical worst case degree m required is the smallest. However, the SLQ method requires an additional cost of nm^2 compared to the Chebyshev method. The degree m required in practice for SLQ is typically small, hence, this additional cost is usually negligible. But, there might be cases where the additional cost might not be negligible, e.g., when $\mathrm{nnz}(A)$ is small and the function f to be approximated has large derivatives requiring a larger m. However, in most cases in practice, SLQ yields more accurate results and requires a much smaller degree m compared to the other methods as illustrated in the following section. Both the Chebyshev and the Taylor methods require computation of the extreme eigenvalues of the matrix which requires an additional cost, and this cost depends on the spectrum of the matrix. Note that computing the smallest eigenvalue accurately is usually difficult for data matrices that have very small spectral gap.

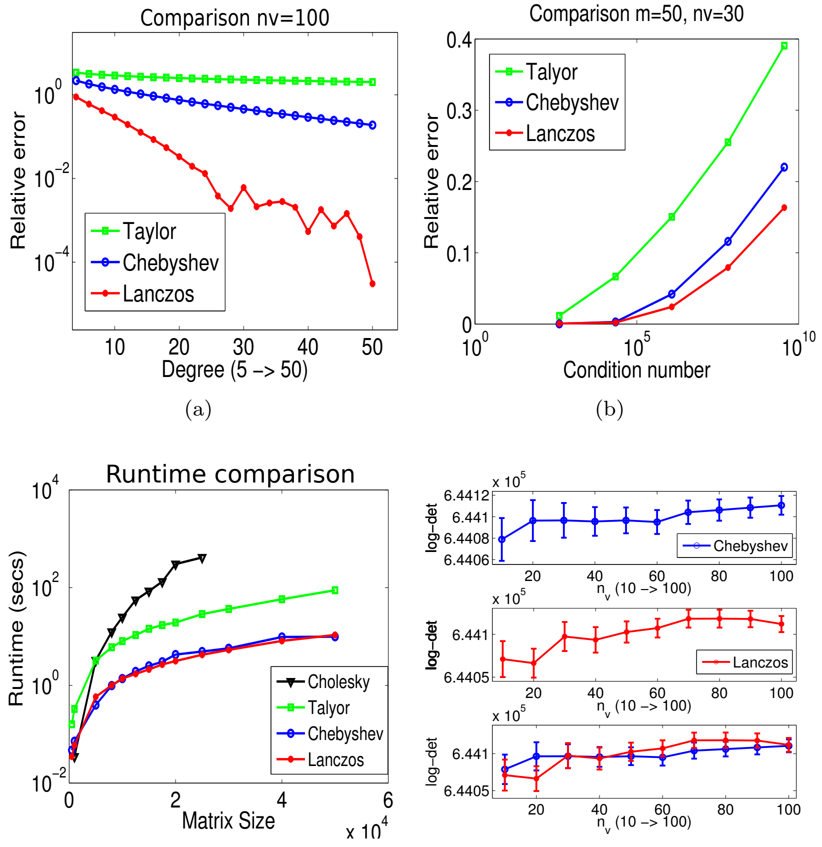
5. Numerical experiments. In this section, we present several examples to illustrate the performance of the SLQ method in various applications. First, we evaluate its performance for log-determinant computation of large matrices, and compare the performance against other related stochastic methods.

In the first experiment (Figure 1(a)), we compare the relative errors obtained by the SLQ method for different degrees chosen, and compare it against the stochastic Chebyshev [26] (implemented by the authors) and the stochastic Taylor series expansions method [47]. We consider the sparse matrix `california` (a graph Laplacian matrix) of size 9664×9664, $\mathrm{nnz} \approx 10^5$, and $\kappa \approx 5 \times 10^4$ from the University of Florida (UFL) sparse matrix collection [15]. The number of starting vectors $\mathrm{n_v} = 100$ in all three cases. The figure shows that our method is superior in accuracy compared to the other two methods. With just a degree of around 50, we get 4 digits of accuracy, while Chebyshev expansions give only 1–2 digits of accuracy and Taylor series expansions are very inaccurate for such low degrees.

In the second experiment, we evaluate the performance of our method with respect to the condition number of the matrix. We consider a Hadamard[3] matrix H of size 8192 and form the test matrix as $HDH^\top$, where D is a diagonal matrix with entries such that the desired condition number is obtained. Figure 1(b) plots the relative errors obtained by the three stochastic methods for the log-determinant estimations of the matrices with different condition numbers. The degree and the number of starting vectors used in all three cases were $m = 50$ and $\mathrm{n_v} = 30$. Again, we observe the superior accuracy of SLQ.

In the third experiment, we compare the runtime of the three algorithms for log-determinant estimation of large sparse matrices. The matrices have used 10% nonzeros in each row. An example MATLAB code is the following: `N=20000; rho = 10/N;`

[3] A Hadamard matrix is chosen since its eigenvalues are known a priori and is easy to generate. Reproducing the experiment will be easier.



(a)

(b)

(c)

(d)

FIG. 1. *Performance comparison between SLQ, Chebyshev, and Taylor series expansions:* (a) *relative error versus degree m,* (b) *relative error versus condition number of the matrices,* (c) *runtime comparison against Cholesky decomposition, and* (d) *estimation and standard error versus number of starting vectors.*

`A = sprand(N,N,rho); A = A'*A + lmin*speye(N)`. These are the same matrices used in [25, Fig. 1]. We also include the runtime for the Cholesky decomposition. For a fair comparison, we chose $m = \sqrt{\kappa}$ for the Chebyshev method, $m = \sqrt{\kappa}/2$ for SLQ, and $m = 4\sqrt{\kappa}$ for Taylor series (will be less accurate since we need $m \approx \kappa$ for similar accuracy). Figure 1(c) plots the runtime of the four algorithms for different matrix sizes. We observe that the runtime of the SLQ method is equal to or less than the runtime of the Chebyshev method. Note also that both Chebyshev and Taylor methods require computation of the extreme eigenvalues. The relative errors we obtained by SLQ in practice are also lower than that obtained by the Chebyshev method. These two methods are both significantly faster than the one based on Cholesky. All experiments were conducted using MATLAB on an Intel core i-5 3.3 GHz machine. All timings are reported using `cputime` function. Comparisons with Schur complement methods and rational approximations can be seen in [25, Fig. 1], where it is shown that the Chebyshev method is superior to these two methods. Hence, we compare SLQ with only the Chebyshev method in the following experiments.

Table 2
Log-determinant computation of real datasets from UFL matrix collection with $\mathrm{n_v} = 30$.

Matrices	Exact logdet	Chebyshev Expansions			Lanczos Quadrature		
		m	Estimate	time	m	Estimate	time
California	-35163	150	-31657.9	1.02	55	-35112.3	1.55
qpband	5521	70	5480.1	0.95	30	5517.0	0.28
thermo.	-546787	75	-546640.3	7.76	25	-546793.9	7.34
boneS01	$1.1093e6$	150	$4.119e6$	26.15	35	$1.104e6$	17.59
ecology2	$3.3943e6$	60	$3.3946e6$	70.8	30	$3.3949e6$	75.24

Table 3
Description of matrices used for the experiments.

Matrices	Applications	Size
California	Web search	9664
qpband	Optimization	20000
thermomechTC	Thermal	102158
boneS01	Model reduction	127224
ecology2	2 dimensional/3 dimensional	999999
Erdos992	undirected graph	6100
deter3	linear programming	7047
FA	Pajek network graph	10617

For very large matrices ($\sim 10^6$ and above), it is impractical to compute the exact log-determinants. To gauge the approximation quality, we approximate the estimator variance by using sample variance and show the standard errors. Figure 1(d) plots the log-determinants estimated and the error bars obtained for different number of starting vectors for the matrix `webbase-1M` (web connectivity matrix) of size $10^6 \times 10^6$ obtained from the UFL database [15]. For Lanczos quadrature, we chose degree $m = 30$, and for Chebyshev, $m = 60$. The width of the error bars gives us a rough idea of how close the estimation might be to the trace of $f(A)$ approximated by the respective methods. The theoretical results for the four methods were listed in Table 1.

Table 2 gives some additional comparison results between Chebyshev expansions and SLQ methods on some large real datasets. All matrices were obtained from the UFL sparse matrix collection [15] and are sparse. A description of the matrices from the UFL collection that are used in the following experiments is given in Table 3. Some of these matrices were also used in [9] as test matrices. The exact log-determinmants of the matrices are listed in the third column. For the first two matrices, their singular values are also available in the UFL database (logdet were computed using them). For the remaining matrices, the exact log-determinants are reported in [9], where the authors used Cholesky decomposition to obtain these values. For the Chebyshev method, we increment the degree m until either we achieve 2-3 digits of accuracy or $m = 150$. For SLQ, we increment the degree m (number of Lanczos steps) until we achieve 3–4 digits of accuracy. The degrees used and the log-determinants estimated by these two methods are listed in the table along with the time taken (averaged over 5 trials) by these algorithms. In all experiments, the number of starting vectors $\mathrm{n_v} = 30$.

We observe that, in all cases, results obtained by SLQ are way more accurate than the Chebyshev method. Also, SLQ requires at least 2–3 times lower degree m than the Chevbyshev method to achieve more accurate results. In addition, we note that the stochastic trace estimator, in general, performs much better than what the worst

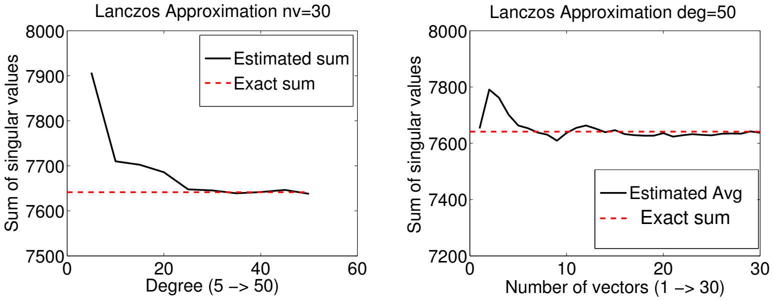


FIG. 2. *The nuclear norm estimated by SLQ for the example* `ukerbe1` *matrix (left) as a function of degree m and (right) as a function of number of starting vectors* $\mathrm{n_v}$.

TABLE 4
Estimation of the sum of singular values of various matrices.

Matrices	m	Exact sum	Estimated sum	Time (secs)	SVD time
Erdos992	40	3292.06	3294.5	1.05	876.2 secs
deter3	30	16518.08	16508.46	1.62	1.3 hrs
California	100	3803.74	3803.86	8.32	4.17 mins
FA	150	1306.79	1312.8	23.13	1.5 hrs
qpband	60	26708.14	26710.1	0.35	2.9 hrs

case analysis in Theorem 4.3 suggests. We get reasonably accurate trace estimation for $\mathrm{n_v} \approx 30$–50. Also, it is important to note that the stochastic Chebyshev and Taylor series methods require computation of the largest and the smallest eigenvalues of the matrix. The computational time reported in the table does not include this additional cost of computing the extreme eigenvalues.

Nuclear norm. Next, let us consider the estimation of the nuclear norm of a matrix for examining the effects of the parameters m and $\mathrm{n_v}$ in the SLQ performance. We consider the matrix `ukerbe1` of size 5981 × 5981 from the UFL database. The performance of the SLQ method in approximately estimating the sum of singular values of this matrix is given in Figure 2.

The left figure plots the estimated nuclear norm for different numbers of Lanczos steps m used, with the number of starting vectors $\mathrm{n_v} = 30$ (black solid line). The right figure plots the approximate nuclear norm computed using Lanczos quadrature obtained for different starting vectors v_l and the cumulative average (black solid line) for Lanczos quadrature of degree $m = 50$. The nuclear norm estimated for degree $m = 50$ and $\mathrm{n_v} = 30$ was 7640.62. The exact sum of singular values is 7641.44.

Finally, we employ our SLQ algorithm 2 with G-K-B for the nuclear norm estimation of real datasets. Table 4 lists the approximate nuclear norm estimated by our method for a set of matrices from various applications. All matrices were obtained from the UFL database [15] and are sparse (listed in Table 3). We increment the degree m (number of G-K-B steps) until we achieve 3–4 digit accuracy. The degree used and the approximate sum obtained are listed in the table along with the exact sum and the time taken (averaged over 5 trials) by our algorithm. In all experiments, the number of starting vectors $\mathrm{n_v} = 30$. In addition, we also list the time taken to compute only the top 2000 singular values of each matrix (computed using the MATLAB `svds` function which relies on ARPACK) in order to provide a rough illustration of the potential computational gain of our algorithm over partial SVD.

Stability. In general, we found that the stability of the Lanczos algorithm will not be an issue here, as long as full reorthogonalization is done (since m is small). If partial orthogonalization is used, we might encounter some stability issues. For a matrix which has many eigenvalues close to zero (possibly ill-conditioned), particularly for the Schatten p-norm applications, the Lanczos algorithm might encounter numerical issues. In these scenarios, it is advantageous to use the G-K-B algorithm which is a numerically safer approach. For example, for the matrix California (Table 2, row 3) of size 9664 which has rank $= 1647$ (has 8017 zero singular values), the Lanczos algorithm with $m = 50$ applied to $X^T X$ gives a tridiagonal matrix with a few (2 or 3) negative eigenvalues. The G-K-B algorithm will not have this issue, and gives a more accurate estimation of the nuclear norm than the Lanczos algorithm.

Maximum likelihood estimation for GRFs. We now test our method for maximum likelihood estimation of Gaussian Random Fields (GRFs). To illustrate the use of log-determinant calculation in GRFs, we simulate one such field by using the Wendland covariance function [37] with smoothness $q = 0$ on a 900×1200 grid ($n = 1.08 \times 10^6$); see Figure 3(a). To better demonstrate the fine details of this highly nonsmooth data, we have zoomed into the middle 300×400 grid and shown only this part. Next, we randomly sampled ten percent of the data and used them to estimate the length scale of the function. These training data are the nonwhite pixels in Figure 3(b). We compute a local log-likelihood curve (as in (2)) shown in Figure 3(c) using SLQ with different values for the hyperparameter, which suggests a peak at 50. That is, maximum likelihood estimation estimates using SLQ suggests the hyperparameter value to be 50. This coincides with the true value used for simulation. The log-determinants therein were computed using 100 Lanczos steps and 100 random vectors. Because the covariance matrix is multilevel Toeplitz, the matrix-vector multiplications were carried out through circulant embedding followed by the FFT, which resulted in an $O(n \log n)$ cost [22]. With the estimated length scale, we perform a prediction calculation for the rest of the data (white pixels in Figure 3(b)) and show the predicted values, together with the ten percent used for training, in Figure 3(d). We observe that the pattern obtained from the predicted values appears quite similar to the original data pattern. The relative difference between Figures 3(a) and 3(d) is 0.27.

Spatial analysis using GMRF for CO_2 data. We consider the Gaussian Markov random field (GMRF) [39] parameter estimation problem for real spatial data with missing entries. We use a global dataset of column-integrated CO_2 obtained from http://niasra.uow.edu.au/cei/webprojects/UOW175995.html. The values of column-integrated CO_2 are on a grid of 1.25° longitude by 1° latitude, which results in a total of $288 \times 181 = 52128$ grid cells (matrix size) on the globe [32]. The dataset has 26633 observations. We assume the GRMF model for the data and use maximum likelihood estimation to predict the remaining (missing) values. For the GMRF field, we considered the spatial autoregressive model, i.e., the precision matrix is defined as $G(\xi) = \xi^4 C + \xi^2 G_1 + G_2$, where matrices C, G_1, and G_2 define the neighborhood (four, eight, and 16 neighbors, respectively) and are sparse [39]. We obtain maximum likelihood estimates using the SLQ method to choose the optimal parameter ξ. That is, we sweep through a set of values for ξ and estimate the log-likelihood for the data given by $\log p(z \mid \xi) = \log\det G(\xi) - z^\top G(\xi) z - \frac{n}{2}\log(2\pi)$, and determine the parameter ξ that maximizes the log-likelihood. Figure 4(top) shows the sparse observations of the CO_2 data across the globe. The GMRF interpolation with the parameter $\xi = 0.2$ is given in Figure 4(bottom).

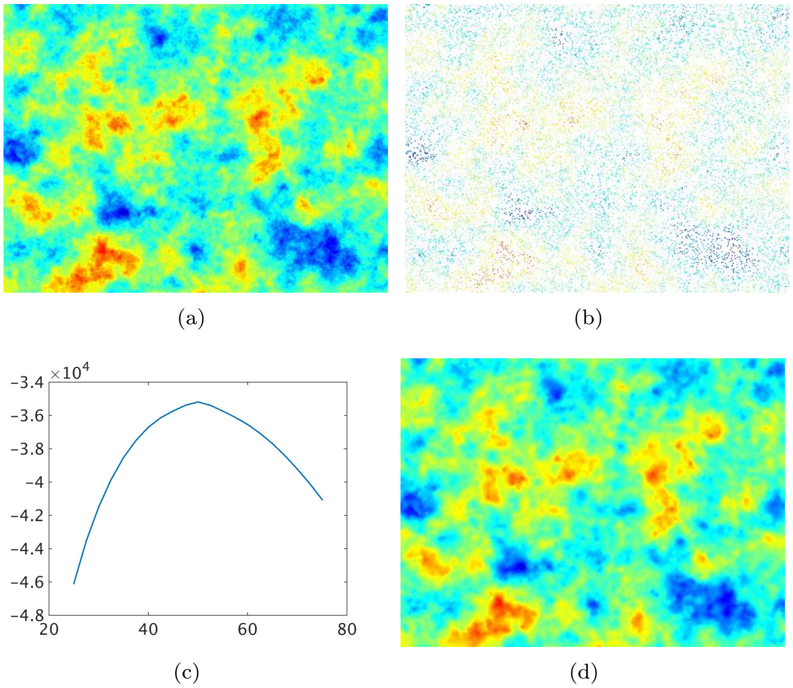


Fig. 3. *Estimation and prediction for a GRF.* (a) *The random field.* (b) *Training data (non-white pixels) for parameter estimation.* (c) *Log-likelihood; the horizontal axis denotes the length-scale parameter.* (d) *Prediction by using the estimated parameter.*

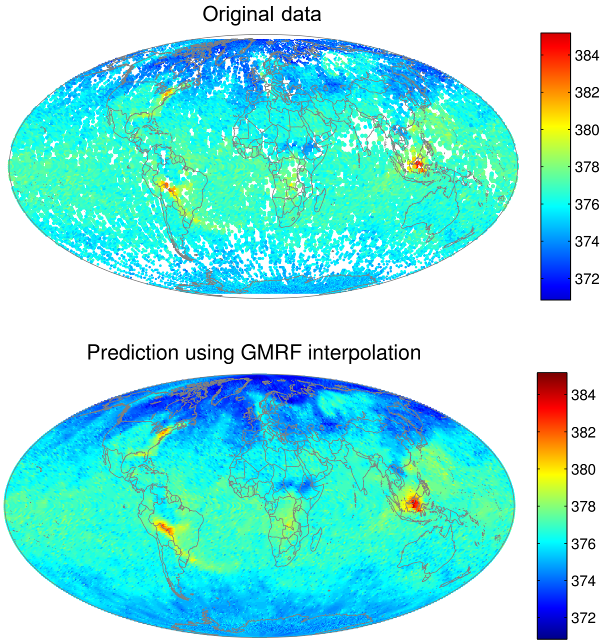


Fig. 4. *GMRF interpolation for CO_2 data. Top: Original data with missing values. Bottom: GMRF interpolated values with parameter* $\xi = 0.2$.

6. Conclusions.

In this paper, we studied an inexpensive technique which we called the SLQ to approximately compute the trace of matrix functions $\mathtt{tr}(f(A))$. We derived approximation error bounds for the method, and showed that it converges faster than any polynomial approximation method. We also established error bounds for approximating useful quantities such as the log-likelihood function. Numerical experiments demonstrated the superior performance of SLQ in practice.

REFERENCES

[1] A. Anandkumar, F. Huang, D. J. Hsu, and S. M. Kakade, *Learning mixtures of tree graphical models*, in Advances in Neural Information Processing Systems, Curran Associates, Red Hood, NY, 2012, pp. 1052–1060.

[2] A. Andoni, R. Krauthgamer, and I. Razenshteyn, *Sketching and embedding are equivalent for norms*, in Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, ACM, New York, 2015, pp. 479–488.

[3] M. Aprahamian, D. J. Higham, and N. J. Higham, *Matching exponential-based and resolvent-based centrality measures*, J. Complex Netw., 4 (2016), pp. 157–176.

[4] E. Aune, D. P. Simpson, and J. Eidsvik, *Parameter estimation in high dimensional Gaussian distributions*, Stat. Comput., 24 (2014), pp. 247–263.

[5] H. Avron and S. Toledo, *Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix*, J. ACM, 58 (2011), 8.

[6] Z. Bai, G. Fahey, and G. Golub, *Some large-scale matrix computation problems*, J. Comput. Appl. Math., 74 (1996), pp. 71–89.

[7] Z. Bai and G. H. Golub, *Bounds for the trace of the inverse and the determinant of symmetric positive definite matrices*, Ann. Numer. Math., 4 (1996), pp. 29–38.

[8] C. Bekas, E. Kokiopoulou, and Y. Saad, *An estimator for the diagonal of a matrix*, Appl. Numer. Math., 57 (2007), pp. 1214–1229.

[9] C. Boutsidis, P. Drineas, P. Kambadur, E. Kontopoulou, and A. Zouzias, *A randomized Algorithm for approximating the log determinant of a symmetric positive definite matrix*, Linear Algebra Appl., 533 (2017), pp. 95–119.

[10] E. J. Candès and B. Recht, *Exact matrix completion via convex optimization*, Found. Comput. Math., 9 (2009), pp. 717–772.

[11] R. Carbó-Dorca, *Smooth function topological structure descriptors based on graph-spectra*, J. Math. Chem., 44 (2008), pp. 373–378.

[12] J. Chen, *How accurately should I compute implicit matrix-vector products when applying the Hutchinson trace estimator?*, SIAM J. Sci. Comput., 38 (2016), pp. A3515–A3539.

[13] J. Chen, M. Anitescu, and Y. Saad, *Computing f(a)b via least squares polynomial approximations*, SIAM J. Sci. Comput., 33 (2011), pp. 195–222.

[14] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon, *Information-theoretic metric learning*, in Proceedings of the 24th International Conference on Machine Learning, ACM, New York, 2007, pp. 209–216.

[15] T. A. Davis and Y. Hu, *The University of Florida sparse matrix collection*, ACM Trans. Math. Software, 38 (2011), 1.

[16] E. Di Napoli, E. Polizzi, and Y. Saad, *Efficient estimation of eigenvalue counts in an interval*, Numer. Linear Algebra Appl., 23 (2016), pp. 674–692.

[17] E. Estrada, *Characterization of 3d molecular structure*, Chem. Phys. Lett., 319 (2000), pp. 713–718.

[18] E. Estrada and N. Hatano, *Statistical-mechanical approach to subgraph centrality in complex networks*, Chem. Phys. Lett., 439 (2007), pp. 247–251.

[19] G. Golub and W. Kahan, *Calculating the singular values and pseudo-inverse of a matrix*, J. SIAM Ser. B Numer. Anal., 2 (1965), pp. 205–224.

[20] G. H. Golub and G. Meurant, *Matrices, Moments and Quadrature with Applications*, Princeton University Press, Princeton, NJ, 2009.

[21] G. H. Golub and Z. Strakoš, *Estimates in quadratic formulas*, Numer. Algorithms, 8 (1994), pp. 241–268.

[22] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Vol. 3. John Hopkins University Press, Baltimore, MD, 2012.

[23] G. H. Golub and J. H. Welsch, *Calculation of Gauss quadrature rules*, Math. Comp., 23 (1969), pp. 221–230.

[24] N. Hale, N. J. Higham, and L. N. Trefethen, *Computing $A^\alpha, \log(A)$, and related matrix functions by contour integrals*, SIAM J. Numer. Anal., 46 (2008), pp. 2505–2523.

[25] I. Han, D. Malioutov, H. Avron, and J. Shin, *Approximating spectral sums of large-scale matrices using stochastic Chebyshev approximations*, SIAM J. Sci. Comput., 39 (2017), pp. A1558–A1585.

[26] I. Han, D. Malioutov, and J. Shin, *Large-scale log-determinant computation through stochastic Chebyshev expansions*, in Proceedings of the 32nd International Conference on Machine Learning, Curran Asociates, Red Hook, NY, 2015, pp. 908–917.

[27] M. Hardt, K. Ligett, and F. McSherry, *A simple and practical algorithm for differentially private data release*, in Advances in Neural Information Processing Systems, Curran Associates, Red Hook, NY, 2012, pp. 2339–2347.

[28] N. J. Higham, *Functions of Matrices: Theory and Computation*, SIAM, Philadelphia, 2008.

[29] M. F. Hutchinson, *A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines*, Comm. Statist. Simulation Comput., 19 (1990), pp. 433–450.

[30] V. Kalantzis, C. Bekas, A. Curioni, and E. Gallopoulos, *Accelerating data uncertainty quantification by solving linear systems with multiple right-hand sides*, Numer. Algorithms, 62 (2013), pp. 637–653.

[31] L. Katz, *A new status index derived from sociometric analysis*, Psychometrika, 18 (Mar. 1953), pp. 39–43.

[32] M. Katzfuss and N. Cressie, *Tutorial on Fixed Rank Kriging (FRK) of $C0_2$ Data*, Technical report 858, Department of Statistics, The Ohio State Unversity, Colorado, OH, 2011.

[33] Y. Li, H. L. Nguyên, and D. P. Woodruff, *On sketching matrix norms and the top singular vector*, in Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, Philadelphia, 2014, pp. 1562–1581.

[34] L. Lin, Y. Saad, and C. Yang, *Approximating spectral densities of large matrices*, SIAM Rev., 58 (2016), pp. 34–65.

[35] D. J. MacKay, *Information Theory, Inference and Learning Algorithms*, Cambridge University Press, Cambridge, 2003.

[36] P. Rabinowitz, *Rough and ready error estimates in Gaussian integration of analytic functions*, Comm. ACM, 12 (1969), pp. 268–270.

[37] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*, MIT Press, Cambridge, MA, 2006.

[38] F. Roosta-Khorasani and U. Ascher, *Improved bounds on sample size for implicit matrix trace estimators*, Found. Comput. Math., 15 (2014), pp. 1187–1212.

[39] H. Rue and L. Held, *Gaussian Markov Random Fields: Theory and Applications*, CRC Press, Boca Raton, FL, 2005.

[40] Y. Saad, *Analysis of some Krylov subspace approximations to the matrix exponential operator*, SIAM J. Numer. Anal., 29 (1992), pp. 209–228.

[41] M. L. Stein, J. Chen, M. Anitescu, et al., *Stochastic approximation of score functions for Gaussian processes*, Ann. Appl. Stat., 7 (2013), pp. 1162–1191.

[42] L. N. Trefethen, *Approximation Theory and Approximation Practice*, SIAM, Philadelphia, 2013.

[43] S. Ubaru and Y. Saad, *Fast methods for estimating the numerical rank of large matrices*, in Proceedings of the 33rd International Conference on Machine Learning, 2016, pp. 468–477.

[44] S. Ubaru, Y. Saad, and A.-K. Seghouane, *Fast estimation of approximate matrix ranks using spectral densities*, Neural Comput., 29 (2017), pp. 1317–1351.

[45] M. J. Wainwright and M. I. Jordan, *Log-determinant relaxation for approximate inference in discrete Markov random fields*, IEEE Trans. Signal Process., 54 (2006), pp. 2099–2109.

[46] L. Wu, J. Laeuchli, V. Kalantzis, A. Stathopoulos, and E. Gallopoulos, *Estimating the trace of the matrix inverse by interpolating from the diagonal of an approximate inverse*, J. Comput. Phys., 326 (2016), pp. 828–844.

[47] Y. Zhang and W. E. Leithead, *Approximate implementation of the logarithm of the matrix determinant in Gaussian process regression*, J. Stat. Comput. Simul., 77 (2007), pp. 329–348.