



Estimating the trace of matrix functions with application to complex networks

Rafael Díaz Fuentes¹ · Marco Donatelli²  · Caterina Fenu¹ · Giorgio Mantica^{2,3,4,5}

Received: 2 May 2022 / Accepted: 14 September 2022 / Published online: 5 November 2022
© The Author(s) 2022

Abstract

The approximation of $\text{trace}(f(\Omega))$, where f is a function of a symmetric matrix Ω , can be challenging when Ω is exceedingly large. In such a case even the partial Lanczos decomposition of Ω is computationally demanding and the stochastic method investigated by Bai et al. (*J. Comput. Appl. Math.* 74:71–89, 1996) is preferred. Moreover, in the last years, a partial global Lanczos method has been shown to reduce CPU time with respect to partial Lanczos decomposition. In this paper we review these techniques, treating them under the unifying theory of measure theory and Gaussian integration. This allows generalizing the stochastic approach, proposing a block version that collects a set of random vectors in a rectangular matrix, in

Dedicated to Claude Brezinski on his eightieth birthday

Rafael Díaz Fuentes, Marco Donatelli, Caterina Fenu and Giorgio Mantica contributed equally to this work.

✉ Marco Donatelli
marco.donatelli@uninsubria.it

Rafael Díaz Fuentes
rafael.diazfuentes@unica.it

Caterina Fenu
kate.fenu@unica.it

Giorgio Mantica
giorgio.mantica@uninsubria.it

- ¹ Department of Mathematics and Computer Science, University of Cagliari, Via Ospedale 72, Cagliari, 09124, Italy
- ² Dipartimento di Scienza e Alta Tecnologia, Università dell'Insubria, Via Valleggio 11, Como, 22100, Italy
- ³ Center for Complex and Nonlinear Systems, Università dell'Insubria, Via Valleggio 11, Como, 22100, Italy
- ⁴ INFN, Sezione di Milano, Milan, Italy
- ⁵ INdAM - GNFM, Firenze, Italy

a similar fashion to the partial global Lanczos method. We show that the results of this technique converge quickly to the same approximation provided by Bai et al. (*J. Comput. Appl. Math.* 74:71–89, 1996), while the block approach can leverage the same computational advantages as the partial global Lanczos. Numerical results for the computation of the Von Neumann entropy of complex networks prove the robustness and efficiency of the proposed block stochastic method.

Keywords Gauss quadrature · Lanczos algorithm · Monte Carlo method · Trace computation · Network analysis

1 Introduction

We are concerned with the computation of the trace of a matrix function, that is:

$$\mathcal{T}(\Omega; f) = \text{trace}(f(\Omega)), \quad (1)$$

where $\Omega \in \mathbb{R}^{n \times n}$ is a symmetric matrix and $f(x)$ is a real function of sufficient regularity, yet not trivially simple. We are interested in applications where the large size of the matrix Ω prevents usage of full matrix operations and only matrix-vector products are available. A paradigmatic example is obtained by choosing $f(x) = -x \log(x)$ and Ω a positive semidefinite matrix, albeit large, truncation of a quantum-mechanical operator. In this case $\mathcal{T}(\Omega; f)$ becomes the von Neumann entropy, a fundamental physical quantity. The same function $f(x)$ is also of relevance to information theory and dynamical systems [2]. A relatively newer and equally important application has recently arisen in relation to graphs, where Ω is the normalized graph Laplacian operator. The von Neumann entropy of graphs has been introduced in [3] and its importance as a statistical indicator in random graphs has been exposed in [4, 5]. From a numerical point of view, its computation is notably challenging, see, e.g., [6] and references therein, and for a broader perspective on matrix functions see [7–9].

While a vast literature has dealt with the problem of approximating the trace of a matrix function (too large to review here, but see the early works of Claude Brezinski [10, 11]), in this paper we follow an approach originally developed in the physics literature [12–14] and later exposed by Golub and Meurant [15]. In this approach matrix elements of $f(\Omega)$ can be reduced to Stieltjes integrals of $f(x)$ with respect to suitable positive measures, as we discuss below. Evaluation of these integrals (at times also providing upper and lower bounds, when $f(x)$ is a completely monotonic function) can be efficiently performed by Gaussian quadrature, which in turn can be profitably derived by the Jacobi matrix of the measure, computed via the Lanczos algorithm. The trace of the matrix function $f(\Omega)$ can finally be obtained by summation over all diagonal matrix elements in the canonical basis, cf. Section 2.

It is an interesting generalization to extend the formalism beyond the evaluation of matrix elements. The quantity $\mathcal{T}(\Omega; f)$ can be computed as the integral

$$\text{trace}(f(\Omega)) = \int_0^\infty f(x) d\nu_\Omega(x), \quad (2)$$

in which ν_Ω is a fundamental spectral measure associated with a compact operator, the counting measure of the eigenvalues,

$$\nu_\Omega = \sum_{j=1}^n \delta_{\lambda_j}, \tag{3}$$

where λ_j , with $j = 1, \dots, n$, are the eigenvalues of Ω and δ_{λ_j} is the atomic measure of unit weight at position λ_j . This measure has crucial importance in many physical properties, from classical and quantum partition functions in statistical mechanics [12] to random matrix theory [16]. In this context, when normalized to unity, it is frequently called the *density of states* (DOS). It plays a prominent role also in mathematics, notably in constructive approximations: consider a positive Borel measure supported on a compact subset K of the real line, and let $\lambda_j^{(n)}$ be the zeros of the orthogonal polynomials of degree n associated with such measure. Remarkable properties are enjoyed by this measure if the normalized counting measure of the zeros has a weak limit, which is the equilibrium measure of the support K in logarithmic potential theory [17].

Numerical algorithms to deal with ν_Ω are therefore of paramount importance. In [18] it was shown that it is possible to adapt the Lanczos algorithm to construct a sequence of orthogonal matrix polynomials, whose associated Jacobi matrix exactly corresponds to the measure ν_Ω , and hence the trace in (1) can be computed by direct integration of (2), without passing via matrix elements. The same procedure was later discussed in [19] (also see the references therein), where it was termed the *global Lanczos method*, a nomenclature that we follow in this work, as opposed to the *scalar Lanczos method* for matrix elements. The global approach was also employed in a related context in [20, 21]. Formal equivalence of the scalar and global technique will be apparent in our presentation: Claude Brezinski, to whom this paper is dedicated, recognized this equivalence early on [22, 23].

When applied to estimate the trace of a large matrix, these techniques are computationally intensive, even if they only require matrix-vector products, a fact that limits their applicability in the case of huge problems. To overcome this limitation a stochastic approach is then advisable, to provide an estimate of the desired quantity $\mathcal{T}(\Omega; f)$. A technique to combine random sampling with the Stieltjes approach for matrix elements has been originally developed by Bai, Fahey, and Golub in [1]. The stochastic approach was extended to estimate the Jacobi matrix of the DOS measure in [18]. A further extension of [1] for the computation of the trace of a matrix function in the case of rank-one vectors is proposed in [24]. Moreover, [25] also deals with the computation in the case of indefinite matrices.

In this paper, we apply and extend the stochastic approach in [1] to a block version that collects $1 \leq k \leq n$ random vectors in an $n \times k$ matrix, similar to what is done in the global Lanczos method in [19]. The new block method reduces to what was proposed in [1] and in [18] in the case of $k = 1$ and $k = n$, respectively. We proved that the mean of the bounds obtained by applying our new block version converges to $\text{trace}(f(\Omega))$. Moreover, we show that the average of k approximations obtained by the scalar stochastic method is different from what is obtained by the block stochastic algorithm applied to the same random vectors. Nevertheless, the results obtained by

the two methods rapidly converge to each other already for small values of k and small Jacobi matrices. From a theoretical point of view, the two methods compute the same number of matrix-vector products, but the block version can take advantage of the modern computer architectures as it occurs for global Lanczos with respect to scalar Lanczos.

While the proposed block stochastic method can be applied to solve the problem (2) for a generic function f , in the numerical experiments we focus on the von Neumann entropy of graphs. Several tests on large real networks reveal results of good accuracy with a relatively low CPU time of the proposed block stochastic method.

This paper is organized as follows: In Section 2 we review the relations that connect the trace of a matrix function to Stieltjes integrals with respect to positive Borel measures and provide bounds by Gaussian and Gauss-Radau formulae. The Lanczos technique is employed in Section 3 to compute the Jacobi matrices of these measures. Our presentation aims to show the formal equivalence of the so-called scalar and global Lanczos technique, the latter described in Section 3.1. In Section 4 we review the Monte Carlo approach in [1], for which we prove theoretically a useful convergence property. In Section 5, following the same formal equivalence that links scalar and global Lanczos, the Monte Carlo approach is generalized to random “vectors” in $\mathbb{R}^{n \times k}$, also proving its convergence. The performance of this method is compared in Section 6 with the other reviewed algorithms on the computation of the von Neumann entropy of a sample of complex networks. Section 7 contains concluding remarks.

2 The trace of a matrix function as a quadratic functional

In this section, mainly to fix the notation, we write the trace of a matrix function as a quadratic functional, acting in spaces of vectors or matrices, in a similar fashion. These functionals are then reduced to Stieltjes integrals with respect to corresponding positive Borel measures. The standard theory is applied in Section 2.1 to bound these integrals by Gaussian and Gauss-Radau formulae.

The trace in (1) can be evaluated as a sum of quadratic forms:

$$\text{trace}(f(\Omega)) = \sum_{i=1}^n \mathcal{Q}(\mathbf{e}_i; f). \quad (4)$$

In this equation, \mathbf{e}_i is the i -th vector in the canonical basis of \mathbb{R}^n and $\mathcal{Q}(\mathbf{u}; f)$ is

$$\mathcal{Q}(\mathbf{u}; f) = \mathbf{u}^T f(\Omega) \mathbf{u} = \langle \mathbf{u}, f(\Omega) \mathbf{u} \rangle, \quad (5)$$

where $\mathbf{u} \in \mathbb{R}^n$ and $\langle \cdot, \cdot \rangle$ indicates the scalar product in \mathbb{R}^n .

By gathering the terms in the summation (4) in g groups of k elements (we suppose for simplicity of notation that $n = gk$, but the procedure can be generalized when k does not divide n , by changing the cardinality of one or more of such groups) one sees that

$$\text{trace}(f(\Omega)) = \sum_{m=1}^g \mathcal{Q}(\mathbf{E}_m; f), \quad (6)$$

where \mathbf{E}_m are the matrices composed of k columns of the identity matrix:

$$\mathbf{E}_m = [\mathbf{e}_{(m-1)k+1}, \dots, \mathbf{e}_{mk}], \quad m = 1, \dots, g, \tag{7}$$

and accordingly the quadratic form acts on “vectors” \mathbf{U} in $\mathbb{R}^{n \times k}$ as

$$\mathcal{Q}(\mathbf{U}; f) = \text{trace}(\mathbf{U}^T f(\Omega)\mathbf{U}). \tag{8}$$

Note that we use the same symbol \mathcal{Q} in (4), and (6) and (8), the difference being revealed by the argument of the quadratic form, in lower (4) or upper case (6, 8). Clearly, the quadratic forms depend on Ω . Not to overburden the notation, this dependence will be left implicit—as well as that of related measures. Similarly, we use the same symbol $\langle \cdot, \cdot \rangle$ for the scalar product in \mathbb{R}^n , (5), and the scalar product in the space of $n \times k$ real matrices, whose associated norm is the Frobenius norm:

$$\langle \mathbf{U}, \mathbf{V} \rangle = \text{trace}(\mathbf{U}^T \mathbf{V}). \tag{9}$$

This usage cannot lead to confusion. In fact, it is easily seen that (5) is a special case of (9) for $k = 1$, where \mathbf{U} reduces to \mathbf{u} and $\mathbf{V} = f(\Omega)\mathbf{u}$. Moreover, letting $k = n$ implies that \mathbf{E}_1 is the identity matrix, so that $\mathcal{Q}(\mathbf{I}; f) = \text{trace}(f(\Omega))$ and (2) can also be fitted into this framework. Throughout the paper, we shall move to and fro the two limiting cases.

Let us therefore focus on the quadratic functional (8), starting from the case $k = 1$.

2.1 Gaussian quadrature

Gauss-type quadrature rules [12–15, 26] can be employed to estimate the $k = 1$ case in the previous subsection, that is, the quadratic form (5). An overview is presented in [15]. This technique also provides upper and lower bounds to $\mathcal{Q}(\mathbf{u}; f)$. Although it is now quite standard, we briefly review it, since is at the core of successive developments.

The quadratic form in (5) takes a significant form when one employs the spectral factorization

$$\Omega = \mathbf{Q} \Lambda \mathbf{Q}^T, \tag{10}$$

in which the columns of the orthogonal matrix $\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_n] \in \mathbb{R}^{n \times n}$ are the orthonormalized eigenvectors of Ω : $\Omega \mathbf{q}_j = \lambda_j \mathbf{q}_j$, with $j = 1, \dots, n$, and Λ is the diagonal matrix $\Lambda = \text{diag} [\lambda_1, \lambda_2, \dots, \lambda_n] \in \mathbb{R}^{n \times n}$. Eigenvalues and eigenvectors are conveniently ordered according to $\lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n$. In so doing, one obtains

$$\mathcal{Q}(\mathbf{u}; f) = \mathbf{u}^T \mathbf{Q} f(\Lambda) \mathbf{Q}^T \mathbf{u} = \sum_{j=1}^n \langle \mathbf{u}, \mathbf{q}_j \rangle^2 f(\lambda_j), \tag{11}$$

which can be understood as a Stieltjes integral:

$$\mathcal{Q}(\mathbf{u}; f) = \int_0^\infty f(x) d\mu_{\mathbf{u}}(x). \tag{12}$$

In the above, $\mu_{\mathbf{u}}$ is a piece-wise constant distribution function with jumps of size $\langle \mathbf{u}, \mathbf{q}_j \rangle^2$ at the eigenvalues λ_j of Ω . Equivalently, $\mu_{\mathbf{u}}$ is a positive discrete measure

composed of n atoms:

$$\mu_{\mathbf{u}} = \sum_{j=1}^n \langle \mathbf{u}, \mathbf{q}_j \rangle^2 \delta_{\lambda_j}. \quad (13)$$

We can take \mathbf{u} as a unit norm vector so that $\mu_{\mathbf{u}}$ is a probability measure. Therefore, the integral (12) can be approximated by the ℓ -point Gaussian quadrature rule

$$\mathcal{G}^{(\ell)}(\mu_{\mathbf{u}}, f) = \sum_{i=1}^{\ell} w_i^{(\ell)} f(z_i^{(\ell)}). \quad (14)$$

Equation (14) is itself an integral of f with respect to a discrete measure. As it is well known, this measure can be derived from $J^{(\ell)}(\mu_{\mathbf{u}})$, the truncation of rank ℓ of the Jacobi matrix of $\mu_{\mathbf{u}}$, which can be formally written as

$$J^{(\ell)}(\mu_{\mathbf{u}}) = \begin{bmatrix} a_0 & b_1 & & & \\ b_1 & a_1 & b_2 & & \\ & b_2 & \ddots & \ddots & \\ & & \ddots & \ddots & b_{\ell-1} \\ & & & b_{\ell-1} & a_{\ell-1} \end{bmatrix} \in \mathbb{R}^{\ell \times \ell}.$$

In fact, the nodes $z_i^{(\ell)}$ are the eigenvalues of $J^{(\ell)}(\mu_{\mathbf{u}})$ and the weights $w_i^{(\ell)}$ are the squares of the first components of the associated normalized eigenvectors [27], quantities that can be profitably computed by the Golub-Welsch method [28].

Note that quantity (14) can also be computed starting from the Jacobi matrix $J^{(\ell)}(\mu_{\mathbf{u}})$, that is:

$$\mathcal{G}^{(\ell)}(\mu_{\mathbf{u}}, f) = \mathbf{e}_1^T f(J^{(\ell)}(\mu_{\mathbf{u}})) \mathbf{e}_1. \quad (15)$$

3 The Lanczos method

The required Jacobi matrix can be obtained by ℓ steps of the Lanczos method reproduced here in Algorithm 1, applied to the matrix Ω with initial vector \mathbf{u} .

Suppose that the function $f(x)$ is completely monotonic, i.e., when z belongs to the convex hull of the spectrum of Ω , the ℓ -th derivative of f is such that $(-1)^\ell f^{(\ell)}(z) \geq 0$ (at least, in a range of values of ℓ). Then, analysis of the remainder formula for Gaussian integration reveals that $\mathcal{G}^{(\ell)}(\mu_{\mathbf{u}}, f)$ provides a lower bound to the integral (12). It has been proved, in exact arithmetic, that in this case increasing ℓ the bound gets tighter; see [15].

Under the same conditions, one also proves that the $(\ell + 1)$ -point Gauss-Radau quadrature rule $\mathcal{R}^{(\ell+1)}(\hat{\mu}_{\mathbf{u}}, f)$ yields tightening upper bounds. This quadrature is constructed with a fixed node at z_0 (z_0 can be any real value to the left of the spectrum of Ω). The rule can be computed in the same way as in (15), from the truncated Jacobi matrix of the positive measure $d\hat{\mu}_{\mathbf{u}}(s) = (s - z_0) d\mu_{\mathbf{u}}(s)$, which can be easily obtained as follows:

$$J^{(\ell+1)}(\hat{\mu}_{\mathbf{u}}) = \begin{bmatrix} J^{(\ell)}(\mu_{\mathbf{u}}) & b_\ell \mathbf{e}_\ell \\ b_\ell \mathbf{e}_\ell^T & \hat{a}_\ell \end{bmatrix} \in \mathbb{R}^{(\ell+1) \times (\ell+1)},$$

In this equation $\hat{a}_\ell = z - b_\ell^2 \pi_{\ell-1}(z_0)/\pi_\ell(z_0)$, and π_ℓ is the monic orthogonal polynomial of degree ℓ of $\mu_{\mathbf{u}}$. Similarly to the computation of $\mathcal{G}^{(\ell)}(\mu_{\mathbf{u}}, f)$ in (14), the quantity $\mathcal{R}^{(\ell+1)}(\hat{\mu}_{\mathbf{u}}, f)$ can also be computed starting from the Jacobi matrix $J^{(\ell+1)}(\hat{\mu}_{\mathbf{u}})$, that is:

$$\mathcal{R}^{(\ell+1)}(\hat{\mu}_{\mathbf{u}}, f) = \mathbf{e}_1^T f(J^{(\ell+1)}(\hat{\mu}_{\mathbf{u}}))\mathbf{e}_1. \tag{16}$$

Combining the two bounds, when f is completely monotonic, the following sequence of inequalities holds for any ℓ :

$$\mathcal{G}^{(\ell-1)}(\mu_{\mathbf{u}}, f) < \mathcal{G}^{(\ell)}(\mu_{\mathbf{u}}, f) < \mathcal{Q}(\mathbf{u}; f) < \mathcal{R}^{(\ell+1)}(\hat{\mu}_{\mathbf{u}}, f) < \mathcal{R}^{(\ell)}(\hat{\mu}_{\mathbf{u}}, f); \tag{17}$$

see, e.g., [15, 27, 29] for details. In practice, it is sufficient that the j -th derivative of f is completely monotonic for a certain j , possibly small for the computational purpose.

We assume that the Lanczos algorithm does not breakdown, that is, $b_{\ell-1}$ in the matrix $J^\ell(\mu_{\mathbf{u}})$ is larger than zero. When b_ℓ is null, the spectrum of the Jacobi matrix is a subset of the spectrum of Ω and $\mathcal{G}^{(\ell)}(\mu_{\mathbf{u}}, f) = \mathcal{R}^{(\ell+1)}(\hat{\mu}_{\mathbf{u}}, f) = \mathbf{u}^T Qf(\Lambda)Q^T \mathbf{u}$. If b_ℓ turns out to be negative, this means that numerical instabilities (typically, loss of normalization) have affected the computation.

Require: Matrix $\Omega \in \mathbb{R}^{n \times n}$, vector $\mathbf{u} \in \mathbb{R}^n$, Jacobi matrix size ℓ , completely monotonic function f .

Ensure: lower bound $L^{(\ell)}(\mathbf{u}; f)$, upper bound $U^{(\ell+1)}(\mathbf{u}; f)$, and approximation $\mathcal{Q}(\mathbf{u}; f) \simeq \frac{1}{2}(L^{(\ell)}(\mathbf{u}; f) + U^{(\ell+1)}(\mathbf{u}; f))$

- 1: $\mathbf{u}_0 = \mathbf{u}, \mathbf{u}_{-1} = \mathbf{0}, b_0 = 1$
 - 2: **for** $j = 1, \dots, \ell$ **do**
 - 3: $a_{j-1} = \langle \mathbf{u}_{j-1}, \Omega \mathbf{u}_{j-1} \rangle$
 - 4: $\mathbf{r}_j = \Omega \mathbf{u}_{j-1} - a_{j-1} \mathbf{u}_{j-1} - b_{j-1} \mathbf{u}_{j-2}$
 - 5: $b_j = \sqrt{\langle \mathbf{r}_j, \mathbf{r}_j \rangle}$
 - 6: $\mathbf{u}_j = \mathbf{r}_j / b_j$
 - 7: **end for**
 - 8: Compute the Gauss rule $L^{(\ell)}(\mathbf{u}; f) := \mathcal{G}^{(\ell)}(\mu_{\mathbf{u}}, f)$ via the Golub-Welsch algorithm [28] applied to $J^{(\ell)}(\mu_{\mathbf{u}})$.
 - 9: Enlarge $J^{(\ell)}(\mu_{\mathbf{u}})$ to yield $J^{(\ell+1)}(\hat{\mu}_{\mathbf{u}})$
 - 10: Compute the Gauss-Radau rule $U^{(\ell+1)}(\mathbf{u}; f) := \mathcal{R}^{(\ell+1)}(\hat{\mu}_{\mathbf{u}}, f)$ via the Golub-Welsch algorithm [28] applied to $J^{(\ell+1)}(\hat{\mu}_{\mathbf{u}})$.
-

Algorithm 1 The Scalar Lanczos algorithm

3.1 The global Lanczos method

The method of the previous section can be generalized to the case of the quadratic form (8), replacing vectors \mathbf{u} by matrices \mathbf{U} in steps 1 to 8 (see Algorithm 2). The generalization is particularly instructive when $k = n$ and $\mathbf{U} = \mathbf{I}$ is the identity matrix. In this case, the Lanczos Algorithm 2 yields the sequence of matrix orthogonal polynomials $p_j(\Omega)$ that satisfy the three-terms relation

$$\Omega p_j(\Omega) = b_{j+1} p_{j+1}(\Omega) + a_j p_j(\Omega) + b_j p_{j-1}(\Omega), \tag{18}$$

Require: Matrix $\Omega \in \mathbb{R}^{n \times n}$, vector $\mathbf{U} \in \mathbb{R}^{n \times k}$, Jacobi matrix size ℓ , completely monotonic function f

Ensure: lower bound $L^{(\ell)}(\mathbf{U}; f)$, upper bound $U^{(\ell+1)}(\mathbf{U}; f)$, and approximation $\mathcal{Q}(\mathbf{U}; f) \simeq \frac{1}{2}(L^{(\ell)}(\mathbf{U}; f) + U^{(\ell+1)}(\mathbf{U}; f))$

1: Replace \mathbf{u} by \mathbf{U} , $\mathbf{u}_{-1} = \mathbf{0} \in \mathbb{R}^{n \times k}$ and employ the scalar product (9) in Algorithm 1.

Algorithm 2 The Global Lanczos algorithm

initialized by $p_0(\Omega) = \mathbf{I}$ and $p_{-1}(\Omega) = \mathbf{0}$, $\mathbf{0} \in \mathbb{R}^{n \times k}$ being the null operator. It is easy to show (see, e.g., [18], Sect. 10 Lemma 10.1) that the Jacobi matrix so produced corresponds to the measure $\mu_{\mathbf{I}}$ that, in this case, is precise ν_{Ω} defined in (3), the counting measure of the eigenvalues. In the mathematical physics literature, this measure (when normalized to unity) is also known as the *density of states* of the operator Ω , to distinguish it from the *local density of states*, typically represented by (13).

Because of this formal equivalence, the considerations presented in the previous subsection extend to this case: when $f(x)$ is a completely monotonic function, Gauss and Gauss-Radau quadratures provide upper and lower bounds to the desired quantity (2).

The case $1 < k < n$, discussed in [19], is intermediate between the two extremes and can be described by the same theory. In fact, in this case, one can identify the measure $\mu_{\mathbf{U}}$ as follows. Let $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_k] \in \mathbb{R}^{n \times k}$ be a matrix with normalized columns $\mathbf{u}_s \in \mathbb{R}^n$, for $s = 1$ to k . Taking $f(x) = x$, we observe that

$$\mathcal{Q}(\mathbf{U}; f) = \langle \mathbf{U}, \Omega \mathbf{U} \rangle = \sum_{j=1}^n \lambda_j \sum_{s=1}^k \langle \mathbf{u}_s, \mathbf{q}_j \rangle^2, \quad (19)$$

which shows that the measure $\mu_{\mathbf{U}}$ is

$$\mu_{\mathbf{U}} = \sum_{j=1}^n \mu_{\mathbf{U}_j} \delta_{\lambda_j}, \quad (20)$$

with the positive weights $\mu_{\mathbf{U}_j}$ implicitly defined by (19). Quite in the same way as before, computing the Jacobi matrix $J(\mu_{\mathbf{U}})$ via the Lanczos algorithm and proceeding with Golub and Welsch diagonalization and finally quadratures yield upper and lower bounds to $\mathcal{Q}(\mathbf{U}; f)$ for a completely monotonic $f(x)$. The synthetic scheme of this procedure is presented in Algorithm 2.

Finally, choosing in sequence $\mathbf{U} = \mathbf{E}_m$, $m = 1, \dots, g$, as in (7), one obtains upper and lower bounds to the desired quantity $f(\Omega)$, via (6).

4 Monte Carlo approach

Often, the size of the matrix Ω is exceedingly large so computer time limitations hinder the computation of the full sums (4) or (6). A stochastic approach is then advisable, to provide a probabilistic estimate of the desired quantity $\mathcal{T}(\Omega; f)$. A first

approach employs the following result from [30, 31] that the reader can easily prove by direct computation (also see below, in Section 5, (24)).

Proposition 1 ([30, 31]) *Let H be an $n \times n$ symmetric matrix. Let $\mathbf{u} \in \mathbb{R}^n$ be a random vector whose entries are independent, equally distributed random variables of zero mean and unit variance. Then, the random variable $\langle \mathbf{u}, H\mathbf{u} \rangle$ is an unbiased estimator of $\text{trace}(H)$, while the variance of the estimator is $2 \sum_{i \neq j} \langle \mathbf{e}_i, H\mathbf{e}_j \rangle^2$.*

To apply this proposition to the problem at hand, where $H = f(\Omega)$, the matrix function must be evaluated first. To solve this problem, the authors of [1] combined stochastic sampling with the Stieltjes technique. This produces Algorithm 3 reproduced herein, where we use the symbol k to denote the cardinality of the random sample. This choice is dictated by formal equivalence with the material in Section 3.1, as it will be apparent in a moment.

Require: Matrix $\Omega \in \mathbb{R}^{n \times n}$, sample vectors $\mathbf{u}_s \in \mathbb{R}^n$, $s = 1, \dots, k$, Jacobi matrix size ℓ , completely monotonic function f
Ensure: stochastic approximation of $\text{trace}(f(\Omega)) \simeq \frac{1}{2k} \sum_{s=1}^k (L^{(\ell)}(\mathbf{u}_s; f) + U^{(\ell+1)}(\mathbf{u}_s; f))$
 1: **for** $s = 1, \dots, k$ **do**
 2: compute lower bound $L^{(\ell)}(\mathbf{u}_s; f)$ and upper bound $U^{(\ell+1)}(\mathbf{u}_s; f)$ for $\mathcal{Q}(\mathbf{u}_s; f) = \langle \mathbf{u}_s, f(\Omega)\mathbf{u}_s \rangle$ by Algorithm 1
 3: accumulate the values $L^{(\ell)}(\mathbf{u}_s; f)$ and $U^{(\ell+1)}(\mathbf{u}_s; f)$.
 4: **end for**

Algorithm 3 The Monte Carlo approach introduced in [1]

In words, the algorithm evaluates upper and lower bounds for the k functionals $\mathcal{Q}(\mathbf{u}_s; f)$ and averages these values to get an estimate of $\text{trace}(f(\Omega))$. In fact, observe that the bounds $L(\mathbf{u}_s; f)$ and $U(\mathbf{u}_s; f)$ are random function themselves, via the random vectors \mathbf{u}_s . Consider the inequality (10) in reference [1] (adapted to our notation)

$$\frac{1}{k} \sum_{s=1}^k L^{(\ell)}(\mathbf{u}_s; f) \leq \frac{1}{k} \sum_{s=1}^k \langle \mathbf{u}_s, f(\Omega)\mathbf{u}_s \rangle \leq \frac{1}{k} \sum_{s=1}^k U^{(\ell+1)}(\mathbf{u}_s; f) \tag{21}$$

and the related sentence from the same paper: *It is natural to expect that with suitable sample size, the mean of the computed bounds yields a good estimation of the quantity $\text{trace}(f(\Omega))$.* Yet, Proposition 1 only guarantees that the expectation of the *central* term in the above inequalities is equal to the desired quantity so that the expectation of the first and third term (the ones computed by the algorithm) are respectively smaller and larger than $\text{trace}(f(\Omega))$. Clearly, the expectation of the random variable $\frac{1}{2}[L^{(\ell)}(\mathbf{u}_s; f) + U^{(\ell+1)}(\mathbf{u}_s; f)]$ is between these bounds, but it cannot be assumed to be *equal* to $\text{trace}(f(\Omega))$. This leaves a theoretical gap that was implicitly acknowledged in the quoted sentence.

In practice, though, the statistical variation in the computation of the expectation of the indicator $\frac{1}{2}[L^{(\ell)}(\mathbf{u}_s; f) + U^{(\ell+1)}(\mathbf{u}_s; f)]$ is much wider than its possible bias, so that numerical experiments in [1, 18] together with the use of statistical inequalities seemed to provide reliable estimates.

We are now in the position to show theoretically that the possible bias in the above indicator decreases in the limit of increasing Jacobi matrix size, actually in a wider context than originally considered in [1].

Proposition 2 *Let Ω be an $n \times n$ symmetric matrix. Let $\mathbf{u} \in \mathbb{R}^n$ be a random vector whose entries are independent, equally distributed random variables of zero mean and unit variance, and let f be a completely monotonic real function. Then, the expectation of the random variables $\mathcal{G}^{(\ell)}(\mu_{\mathbf{u}}, f)$ and $\mathcal{R}^{(\ell+1)}(\hat{\mu}_{\mathbf{u}}, f)$ converge to $\text{trace}(f(\Omega))$ when ℓ tends to infinity.*

Proof On the one hand, each quadrature in the proposition is a function of the random vector \mathbf{u} in the probability space \mathbb{R}^n , which is drawn according to a generic Borel probability measure $\rho(\mathbf{u})$. On the other hand, when the moment problem is determined, which is always the case for the spectral measure $\mu_{\mathbf{u}}$ of a finite matrix Ω (the result extends to a much larger family of operators) for any fixed vector \mathbf{u} the sequence $\mathcal{G}^{(\ell)}(\mu_{\mathbf{u}}, f)$ converges *monotonically* (for a completely monotonic f) to $\int f(x) d\mu_{\mathbf{u}}(x)$, which is equal to $\langle \mathbf{u}, f(\Omega)\mathbf{u} \rangle$. Therefore, we can apply Beppo Levi’s theorem to prove that

$$\begin{aligned} \lim_{\ell \rightarrow \infty} \int d\rho(\mathbf{u}) \mathcal{G}^{(\ell)}(\mu_{\mathbf{u}}, f) &= \int d\rho(\mathbf{u}) \lim_{\ell \rightarrow \infty} \mathcal{G}^{(\ell)}(\mu_{\mathbf{u}}, f) = \\ &= \int d\rho(\mathbf{u}) \int f(x) d\mu_{\mathbf{u}}(x) = \int d\rho(\mathbf{u}) \langle \mathbf{u}, f(\Omega)\mathbf{u} \rangle = \text{trace}(f(\Omega)). \end{aligned} \tag{22}$$

The first term in the above equation is the limit of the expectation of the random variable $\mathcal{G}^{(\ell)}(\mu_{\mathbf{u}}, f)$ that therefore converges to the expectation of the random variable $\langle \mathbf{u}, f(\Omega)\mathbf{u} \rangle$, which, in force of Proposition 1, is equal to the trace of $f(\Omega)$, see also (23) and (24) below. Similar reasoning applies to the upper bounds provided by Gauss-Radau quadrature. □

It is evident that, in exact arithmetic, when Ω is a finite matrix the limit in the above proposition is certainly achieved at $\ell = n$. Since ℓ is typically smaller than n in applications, the monotonic decrease of the bias predicted theoretically is of practical relevance. As mentioned before, the argument can be extended to more general operators in Hilbert spaces of infinite dimension, like the multiplication operator by x in L^2_{μ} , when μ is a singular continuous measure [32], where the limit for ℓ tending to infinity is essential.

Summarizing, the quantities $\frac{1}{k} \sum_{s=1}^k L^{(\ell)}(\mathbf{u}_s; f)$ and $\frac{1}{k} \sum_{s=1}^k U^{(\ell+1)}(\mathbf{u}_s; f)$ are stochastically distributed, with standard deviation inversely proportional to \sqrt{k} , around a mean value which tends to the trace of $f(\Omega)$ when ℓ grows. In the practical

implementation the individual components of \mathbf{u}_s are conveniently chosen as random variables taking the two values one and minus one with equal probability—the so-called \mathbb{Z}_2 noise—divided by \sqrt{n} to achieve normalization.

When the size n of Ω is exceedingly large, affordable values of k imply that the above standard deviations are much larger than the difference between the two sample averages: it is then unnecessary to consider (and compute) both. In force of the central limit theorem, the distribution of these samples approaches a normal one for k large, yielding stricter probabilistic bounds than those obtained by Hoeffding’s inequality [18].

5 Stochastic evaluation of the Jacobi matrix of the eigenvalue counting measure

As described at the beginning of Section 3.1, Algorithm 2 with the identity matrix as starting “vector” yields the sequence of matrix orthogonal polynomials $p_j(\Omega)$. It requires the computation of the scalar products $\langle p_j(\Omega), \Omega^r p_j(\Omega) \rangle$, where r can take the values zero and one. Since the scalar product (9) is defined as the trace of an operator, it is computed by a summation involving all basis vectors \mathbf{e}_i , $i = 1$ to n , as in (4), or quite equivalently all matrices \mathbf{E}_m , $m = 1, \dots, g$, as in (6). Clearly, the computational complexity of this procedure grows as n times the complexity of the matrix-vector product, which is the most computationally demanding task and is linear in n in the case of sparse Ω as in complex network applications.

A stochastic implementation of the Lanczos algorithm for the Jacobi matrix of the counting measure that vastly reduces computational complexity was described in [18] Sect. 10, Algorithm 5 remark just following. The core of the algorithm coincides with steps 1 to 8 in the Global Lanczos algorithm proposed in [19]. In fact, the stochastic method starts from constructing k normalized random vectors \mathbf{u}_s as in the previous section, but it assembles them in an $n \times k$ random matrix $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_k]$. The Global Lanczos algorithm 2 is then applied to \mathbf{U} . The difference between the scalar version (Algorithm 1) and the global version (Algorithm 2) can be appreciated by spelling out the steps in the latter. In fact, taking into account the structure of the recursion relation and of the scalar product, the sequence of operations can be described as in Algorithm 4.

A few important remarks are in order. Firstly, observe the difference between Algorithms 3 and 4. While they take the same input and attempt to compute the same quantity, the procedure is different. In fact, Algorithm 3 computes the Jacobi matrices $J^{(\ell)}(\mu_{\mathbf{u}_s})$ of the local density of states $\mu_{\mathbf{u}_s}$ in (13) and successively employs Proposition 2. At difference, Algorithm 4 aims at a stochastic computation of the Jacobi matrix of the global density of states, via the measure $\mu_{\mathbf{U}}$ in (20). To put it even more clearly, while in the former algorithm each stochastic vector \mathbf{u}_s yields a Jacobi matrix $J^{(\ell)}(\mu_{\mathbf{u}_s})$, in the latter the vectors \mathbf{u}_s conspire to form a single Jacobi matrix $J^{(\ell)}(\mu_{\mathbf{U}})$, as can be noted in lines 6 and 10 of Algorithm 4 that are outside the loop 7–9. The two procedures yield different results that rapidly converge to the same limit when ℓ grows. Numerical experience (see Section 6) suggests that this convergence can be quite rapid, also profiting from the phenomenon of *self-averaging* [33]

Require: Matrix $\Omega \in \mathbb{R}^{n \times n}$, sample random matrix $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_k]$ with normalized columns $\mathbf{u}_s \in \mathbb{R}^n$, $s = 1, \dots, k$, Jacobi matrix size ℓ , completely monotonic function f

Ensure: stochastic approximation of $\text{trace}(f(\Omega)) \simeq \frac{1}{2}(L^{(\ell)}(\mathbf{U}; f) + U^{(\ell+1)}(\mathbf{U}; f))$

- 1: $b_0 = 1$
- 2: **for** $s = 1, \dots, k$ **do**
- 3: $\mathbf{u}_{0,s} = \mathbf{u}_s$, $\mathbf{u}_{-1,s} = \mathbf{0}$
- 4: **end for**
- 5: **for** $j = 1, \dots, \ell$ **do**
- 6: $a_{j-1} = \frac{1}{k} \sum_{s=1}^k \langle \mathbf{u}_{j-1,s}, \Omega \mathbf{u}_{j-1,s} \rangle$
- 7: **for** $s = 1, \dots, k$ **do**
- 8: $\mathbf{r}_{j,s} = \Omega \mathbf{u}_{j-1,s} - a_{j-1} \mathbf{u}_{j-1,s} - b_{j-1} \mathbf{u}_{j-2,s}$
- 9: **end for**
- 10: $b_j^2 = \frac{1}{k} \sum_{s=1}^k \langle \mathbf{r}_{j,s}, \mathbf{r}_{j,s} \rangle$
- 11: $b_j = \sqrt{b_j^2}$
- 12: **for** $s = 1, \dots, k$ **do**
- 13: $\mathbf{u}_{j,s} = \mathbf{r}_{j,s} / b_j$
- 14: **end for**
- 15: **end for**
- 16: Compute the Gauss rule $L^{(\ell)}(\mathbf{U}; f) := \mathcal{G}^{(\ell)}(\mu_{\mathbf{U}}, f)$ via the Golub-Welsch algorithm [28] applied to $J^{(\ell)}(\mu_{\mathbf{U}})$
- 17: Enlarge $J^{(\ell)}(\mu_{\mathbf{U}})$ to yield $J^{(\ell+1)}(\hat{\mu}_{\mathbf{U}})$
- 18: Compute the Gauss-Radau rule $U^{(\ell+1)}(\mathbf{U}; f) := \mathcal{R}^{(\ell+1)}(\hat{\mu}_{\mathbf{U}}, f)$ via the Golub-Welsch algorithm [28]

Algorithm 4 Stochastic computation of $\text{trace}(f(\Omega))$ via the Jacobi matrix of the eigenvalue counting measure.

of large matrices. The arguments in Proposition 2 can be repeated verbatim in this case, leading to the following result.

Proposition 3 *Let Ω be an $n \times n$ symmetric matrix. Let \mathbf{U} be a random $n \times k$ matrix whose entries are independent, equally distributed random variables of zero mean and unit variance, and let f be a completely monotonic real function. Then, the expectation of the random variables $\mathcal{G}^{(\ell)}(\mu_{\mathbf{U}}, f)$ and $\mathcal{R}^{(\ell+1)}(\hat{\mu}_{\mathbf{U}}, f)$ converge to $\text{trace}(f(\Omega))$ when ℓ tends to infinity.*

Proof Since \mathbf{U} is a random $n \times k$ matrix, (22) becomes

$$\lim_{\ell \rightarrow \infty} \int d\rho(\mathbf{U}) \mathcal{G}^{(\ell)}(\mu_{\mathbf{U}}, f) = \int d\rho(\mathbf{U}) \lim_{\ell \rightarrow \infty} \mathcal{G}^{(\ell)}(\mu_{\mathbf{U}}, f) = \int d\rho(\mathbf{U}) \langle \mathbf{U}, f(\Omega) \mathbf{U} \rangle, \quad (23)$$

where $\rho(\mathbf{U})$ is a probability measure over $\mathbb{R}^{n \times k}$, which is the product of k probability measures $\rho(\mathbf{u})$ over \mathbb{R}^n employed in Proposition 2, and it is finally the product of $n \times k$ real probability measures $\rho(x)$ of null mean and unit variance. The right-hand

side of (23) is precisely equal to $\text{trace}(f(\Omega))$ because

$$\begin{aligned} \int d\rho(\mathbf{U}) \langle \mathbf{U}, \Omega \mathbf{U} \rangle &= \sum_{j=1}^n \lambda_j \sum_{s=1}^k \int d\rho(\mathbf{u}_s) \langle \mathbf{u}_s, \mathbf{q}_j \rangle^2 = \\ &= \sum_{j=1}^n \lambda_j \int d\rho(x) x^2 \sum_{i=1}^n \langle \mathbf{e}_i, \mathbf{q}_j \rangle^2 = \text{trace}(\Omega). \end{aligned} \tag{24}$$

Since $\int d\rho(x)x^2 = 1$ is the variance of $\rho(x)$ and since $\sum_{i=1}^n \langle \mathbf{e}_i, \mathbf{q}_j \rangle^2 = 1$ by normalization of the eigenvectors. A similar reasoning applies to the upper bounds provided by $\mathcal{R}^{(\ell+1)}(\hat{\mu}_{\mathbf{U}}, f)$. □

6 Estimating the von Neumann entropy of graphs

As described in Section 1, the *von Neumann entropy* is obtained by choosing

$$f(x) = -x \log(x), \tag{25}$$

where we define $0 \log(0) = 0$ by convention [3]. Simple calculations show that $(-1)^\ell f^{(\ell)}(x) \leq 0$ for $\ell \geq 2$, which implies that the previous theory can be applied for Jacobi matrices of rank larger or equal to two, switching the role of Gauss and Gauss-Radau quadratures because of the reversed inequality.

This section presents the results of numerical experiments on the performance of the methods described in the previous sections. The operator Ω under consideration is the (normalized) Laplacian operator L of several undirected networks. This matrix is defined from A_{ij} , the adjacency matrix, which takes the value one when nodes i and j are connected and zero otherwise, and D , the diagonal matrix of the nodes' degrees, that is, the number of links connecting a single node:

$$L = D - A. \tag{26}$$

Normalization is effected by dividing by $\text{trace}(L)$, thereby obtaining a *density matrix*

$$\Omega = L/\text{trace}(L),$$

with non-negative spectrum and unit trace. The entropy function f defined in (25) is then most appropriately applied to this matrix Ω . Such von Neumann entropy has therefore been previously computed in several works, see [6] and references therein. We have applied the previous techniques to this problem.

Codes have been programmed in Matlab R2021a. Computations have been performed on an Intel Xeon Gold 6136 computer (16 cores, 32 threads) equipped with 128 GB RAM, running the Linux operating system.

We test algorithms 1–4 on the following undirected networks:

Yeast (2114 nodes, 4480 edges) describes the protein interaction network for yeast. Each edge represents an interaction between two proteins [34, 35]. The data set was originally included in the Notre Dame Networks Database, and it is now available at [36].

Internet (22963 nodes, 96872 edges) is a symmetrized snapshot of the structure of the Internet at the level of autonomous systems, reconstructed from BGP (Border Gateway Protocol) tables posted by the University of Oregon Route Views Project. This snapshot was created by Mark Newman from data for July 22, 2006 [37].

Collaboration (40421 nodes, 351304 edges) is the collaboration network of scientists who submitted preprints to the condensed matter archive at www.arxiv.org [38] between January 1, 1995, and March 31, 2005. The original network is weighted, here we consider an unweighted version [37].

Facebook (63731 nodes, 1545686 edges) is the largest example we consider. It describes all the user-to-user links (*friendships*) of the Facebook New Orleans network. It was studied in [39], and the data set is available at [40].

The methods employed are labeled in what follows:

Full diagonalization: This method computes $\text{trace}(f(\Omega))$ as $-\sum_{j=1}^n \lambda_j \log(\lambda_j)$, where the eigenvalues λ_j are obtained by full numerical diagonalization, through the `eig` function of Matlab, since the computation of λ_j is well-conditioned being Ω positive semidefinite. We consider this value to be the reference value of the trace. In particular, the error reported both in the tables and in the figures below is the relative error of the other methods with respect to this value.

Scalar Lanczos: This method computes $\text{trace}(f(\Omega))$ as in (4), where each quadratic form $Q(\mathbf{e}_i; f)$, $i = 1, \dots, n$, is approximated as in Algorithm 1.

Global Lanczos: This method computes the trace as in (6), where each quadratic form $Q(\mathbf{E}_m; f)$, $m = 1, \dots, g$, is approximated as explained in Section 3.1, Algorithm 2.

Monte Carlo: This method employs the procedure described in Section 4, Algorithm 3.

Block Monte Carlo: This method employs the procedure described in Section 5, Algorithm 4.

The cardinality of the set of random vectors for (Block) Monte Carlo is k like the number of columns of the matrices \mathbf{E}_m in Global Lanczos, which we recall reduces to scalar Lanczos for $k = 1$.

In the first comparison, the number of Lanczos steps ℓ is not fixed *a priori*, but is chosen *on the go* to ensure that

$$\frac{\mathcal{G}^{(\ell)}(\mu_{\mathbf{v}}, f) - \mathcal{R}^{(\ell+1)}(\hat{\mu}_{\mathbf{v}}, f)}{\mathcal{G}^{(\ell)}(\mu_{\mathbf{v}}, f)} \leq 10^{-3}, \quad \forall \mathbf{v} \in \mathcal{S}, \quad (27)$$

where the set \mathcal{S} is defined in Table 1 depending on the algorithm. We test the previous algorithms for three values of the parameter $k = 10, 20, 30$. Tables 2, 3, 4 and 5 shows the relative errors, with respect to the true $\text{trace}(f(\Omega))$ obtained by full diagonalization, and the related CPU time. For all networks, the proposed Block Monte Carlo method is robust with respect to the stopping criteria (27) providing a relative error comparable with Global Lanczos but with a much lower CPU time. On the other hand, the CPU time is comparable with the Monte Carlo method, in particular for the large networks in Tables 4 and 5, but with a lower relative error. The low accuracy of the Monte Carlo method is due to the fact that the stopping criterion is applied to

Table 1 Set \mathcal{S} of the “vectors” that have to satisfy the stopping criteria (27)

Method	\mathcal{S}
Scalar Lanczos	$\{\mathbf{e}_i \in \mathbb{R}^n : i = 1, \dots, n\}$
Global Lanczos	$\{\mathbf{E}_m \in \mathbb{R}^{n \times k} : m = 1, \dots, g\}$
Monte Carlo	$\{\mathbf{u}_s \in \mathbb{R}^n : s = 1, \dots, k\}$
Block Monte Carlo	$\{\mathbf{U} \in \mathbb{R}^{n \times k}\}$

each vector \mathbf{u}_s separately, stopping earlier than in the Block Monte Carlo. Nevertheless, for the large networks “Collaboration” and “Facebook,” the CPU time of Block Monte Carlo is lower than that of Monte Carlo.

To provide a fair comparison between Block Monte Carlo and Monte Carlo methods, we run the two methods for the same value of $k = 5, 10, 15, \dots, 200$ and the same number $\ell = 2, 3, \dots, 10$ of Lanczos steps comparing the relative errors and the CPU time. Figures 1 and 2 show the relative error and the CPU time, respectively, varying k and ℓ in the ranges above. Each pixel is color-coded according to the color bar on the right of each graph. The colors of different graphs cannot be directly compared but must be interpreted according to their color bar. In both figures, the first row contains the results for the Yeast network, while in the second row there are the results for the Internet network. From the first to the third column, we find in sequence the global Lanczos method, the Monte Carlo method, and the Block Monte Carlo method.

The first comparison of the CPU times confirms that the two stochastic methods require a CPU time much lower than the global Lanczos method. Moreover, for the two stochastic methods, the CPU time increases both with k and ℓ . On the other hand, for global Lanczos, according to the results in [19], for a fixed ℓ , the CPU time can decrease when increasing k . Comparing the two stochastic methods, we observe that for the Yeast network the block Monte Carlo method reduces the CPU time with respect to the Monte Carlo method by a factor two, while for the larger network Internet the reduction factor is about 4. This leads us to expect even greater reductions in computation time for larger problems.

Table 2 Performance of the different methods for the Yeast network ($n = 2114$)

k	Method	Error	Time (s)
1	Scalar Lanczos	2.760638e-02	2.446410e-01
	Global Lanczos	8.455259e-04	1.161382e+00
10	Monte Carlo	1.090971e-01	1.447850e-03
	Block Monte Carlo	3.760693e-03	1.142855e-02
	Global Lanczos	9.816665e-04	7.319500e-01
20	Monte Carlo	1.092738e-01	2.599600e-03
	Block Monte Carlo	3.690083e-03	1.756205e-02
	Global Lanczos	1.026877e-03	8.089930e-01
30	Monte Carlo	1.094411e-01	3.702750e-03
	Block Monte Carlo	3.879727e-03	2.527865e-02

Table 3 Performance of the different methods for the Internet network ($n = 22963$)

k	Method	Error	Time (s)
1	Scalar Lanczos	4.614127e-03	6.047011e+01
	Global Lanczos	5.102064e-04	1.383564e+02
10	Monte Carlo	2.011468e-01	3.198965e-02
	Block Monte Carlo	8.272466e-04	1.036716e-01
	Global Lanczos	5.147810e-04	1.393176e+02
20	Monte Carlo	2.012633e-01	6.562980e-02
	Block Monte Carlo	7.080867e-04	1.754521e-01
	Global Lanczos	5.159189e-04	1.529930e+02
30	Monte Carlo	2.013468e-01	9.623835e-02
	Block Monte Carlo	5.407044e-04	2.399765e-01

Table 4 Performance of the different methods for the Collaboration network ($n = 40421$)

k	Method	Error	Time (s)
1	Scalar Lanczos	5.633735e-03	3.222226e+02
	Global Lanczos	2.120762e-04	2.886263e+02
10	Monte Carlo	6.380080e-02	1.000027e-01
	Block Monte Carlo	5.859218e-04	1.092159e-01
	Global Lanczos	1.753134e-04	2.812086e+02
20	Monte Carlo	6.376821e-02	2.111761e-01
	Block Monte Carlo	4.526990e-04	1.928351e-01
	Global Lanczos	1.612020e-04	2.511428e+02
30	Monte Carlo	6.371428e-02	3.109078e-01
	Block Monte Carlo	2.868224e-04	2.585439e-01

Table 5 Performance of the different methods for the Facebook matrix ($n = 63731$)

k	Method	Error	Time (s)
1	Scalar Lanczos	1.479209e-03	3.872303e+03
	Global Lanczos	4.069790e-04	2.370421e+03
10	Monte Carlo	7.523729e-02	7.350576e-01
	Block Monte Carlo	2.918577e-04	4.855025e-01
	Global Lanczos	3.937851e-04	2.372613e+03
20	Monte Carlo	7.517143e-02	1.580860e+00
	Block Monte Carlo	2.681684e-04	8.418812e-01
	Global Lanczos	3.911914e-04	2.526310e+03
30	Monte Carlo	7.516319e-02	2.360758e+00
	Block Monte Carlo	2.956494e-04	1.236741e+00

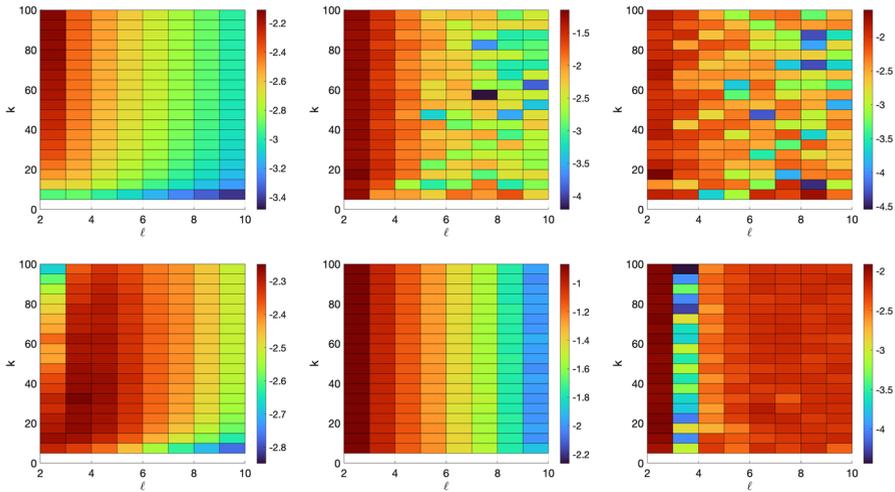


Fig. 1 Relative error varying $k = 5, 10, 15, \dots, 200$ and $\ell = 2, 3, \dots, 10$. Compare global Lanczos method (first column), Monte Carlo method (second column), block Monte Carlo method (third column) for Yeast network (first row) and Internet network (second row)

Concerning the relative errors, Fig. 1 shows that all methods produce comparable results. Clearly, the relative error reduces increasing ℓ , while it is not very sensitive varying $k > 10$. In particular, for the two stochastic methods, already a few random vectors, e.g., $k = 20$, are enough to achieve an accurate enough approximation. This behavior is in agreement with the results in [41] for the scalar Algorithm 3. The two stochastic methods provide about the same relative error although this is usually

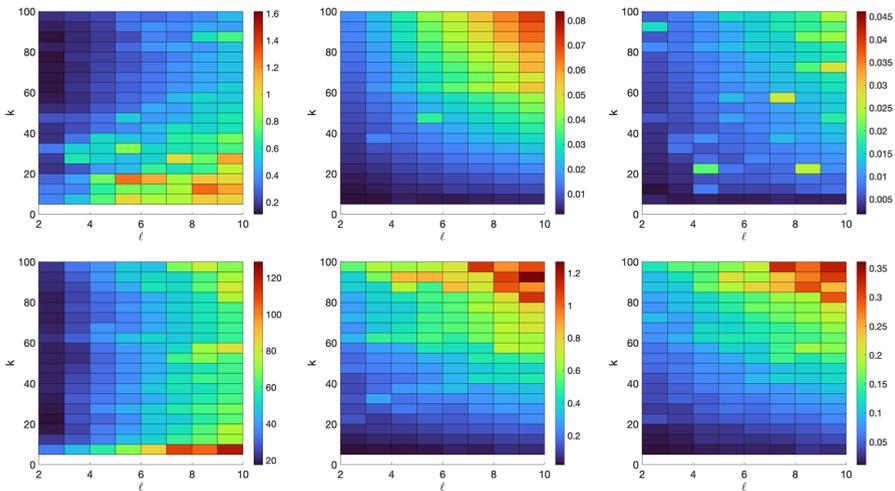


Fig. 2 CPU time varying $k = 5, 10, 15, \dots, 200$ and $\ell = 2, 3, \dots, 10$. Compare global Lanczos method (first column), Monte Carlo method (second column), block Monte Carlo method (third column) for Yeast network (first row) and Internet network (second row)

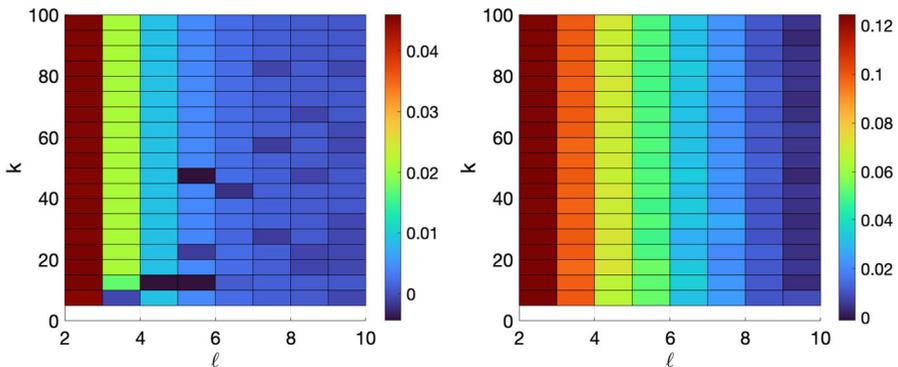


Fig. 3 Difference of the relative error between Monte Carlo method and the block Monte Carlo Method varying $k = 5, 10, 15, \dots, 200$ and $\ell = 2, 3, \dots, 10$, for Yeast network (left) and Internet network (right)

slightly lower for the block Monte Carlo method, see Fig. 3 where the depicted values, which are obtained by subtracting the relative errors for the block Monte Carlo method from the relative errors for the Monte Carlo method, are usually positive.

7 Conclusions

In this paper, we have reviewed various Lanczos techniques for computing the trace of a matrix function. In particular, we have focused on the stochastic approach that was originally presented in [1] for matrix elements—whose associated measure is the local density of states—and later generalized to the global density of states in [18]. Interpolating among these extremes we have defined a block version of such algorithms in a similar fashion to the partial global Lanczos method in [19].

The block Monte Carlo method exhibits interesting properties with respect to accuracy and computational time. Indeed, the numerical results performed to evaluate the Von Neumann entropy of complex networks, show that it is faster than the algorithm in [1], usually providing a better approximation of the solution. Such promising results lead us to believe that the technique might be useful for the general problem (1), when dealing with large matrices, like in the theory of complex networks (see, e.g., the Estrada index studied in [19]) and also in more challenging quantum mechanical applications.

Author contribution All authors have equally contributed to the development of this manuscript.

Funding Marco Donatelli is partially supported by the INdAM-GNCS research project “Tecniche numeriche per l’analisi delle reti complesse e lo studio dei problemi inversi”.

Caterina Fenu is partially supported by Regione Autonoma della Sardegna research project “Algorithms and Models for Imaging Science [AMIS]” (RASSR57257, intervento finanziato con risorse FSC 2014-2020 - Patto per lo Sviluppo della Regione Sardegna) and by the INdAM-GNCS research project “Tecniche numeriche per l’analisi delle reti complesse e lo studio dei problemi inversi”. Caterina Fenu also gratefully acknowledges Regione Autonoma della Sardegna for the financial support provided under the Operational Programme P.O.R. Sardegna F.S.E. (European Social Fund 2014-2020 - Axis III Education and Formation, Objective 10.5, Line of Activity 10.5.12).

Giorgio Mantica is partially supported by PRIN Research Project No. 2017S35EHN “Regular and stochastic behavior in dynamical systems” of the Italian Ministry of Education, University and Research (MIUR).

Availability of data and materials The datasets used and analyzed during the current study are available from the corresponding author upon request.

Code availability The computational code is only prototypal, but it is available from the authors upon request.

Declarations

Conflict of interest The authors declare no competing interests.

References

1. Bai, Z., Fahey, M., Golub, G.: Some large-scale matrix computation problems. *J. Comput. Appl. Math.* **74**, 71–89 (1996)
2. Katok, A., Hasselblatt, B.: *Introduction to the Modern Theory of Dynamical Systems*, vol. 54. Cambridge University Press, Cambridge (1997)
3. Braustein, S.L., Gosh, S., Severini, S.: The Laplacian of a graph as a density matrix: A basic combinatorial approach to separability of mixed states. *Ann. Comb.* **10**, 291–317 (2006)
4. Anand, K., Bianconi, G.: Entropy measures for networks: Toward an information theory of complex topologies. *Phys. Rev. E* **80**–045102 (2009)
5. Anand, K., Bianconi, G., Severini, S.: Shannon and von Neumann entropy of random networks with heterogeneous expected degree. *Phys. Rev. E* **83**–036109 (2020)
6. Choi, H., He, J., Hu, H., Shi, Y.: Fast computation of von Neumann entropy for large-scale graphs via quadratic approximations. *Linear Algebra Appl.* **585**, 127–146 (2020)
7. Arrigo, F., Benzi, M., Fenu, C.: Computation of generalized matrix functions. *SIAM J. Matrix Anal. Appl.* **37**(3), 836–860 (2016)
8. Benzi, M., Boito, P.: Quadrature rule-based bounds for functions of adjacency matrices. *Linear Algebra Appl.* **433**(3), 637–652 (2010)
9. Benzi, M., Boito, P.: Matrix functions in network analysis. *GAMM-Mitteilungen* **43**(3), 202000012 (2020)
10. Brezinski, C., Fika, P., Mitrouli, M.: Estimations of the trace of powers of self-adjoint operators by extrapolation of the moments. *Electron. Trans. Numer. Anal.* **39**, 144–159 (2012)
11. Brezinski, C., Fika, P., Mitrouli, M.: Moments of a linear operator on a Hilbert space, with applications to the trace of the inverse of matrices and the solution of equations. *Numer. Linear Algebra Appl.* **19**, 937–953 (2012)
12. Gordon, R.G.: Error bounds in equilibrium Statistical Mechanics. *J. Math. Phys.* **9**(5), 655–663 (1968)
13. Haydock, R., Heine, V., Kelly, M.J.: Electronic structure based on the local atomic environment for tight-binding bands. *J. Phys. C: Solid State Phys.* **5**(20), 2845–2858 (1972)
14. Nex, C.M.M.: Estimation of integrals with respect to a density of states. *J. Phys. A: Math. Gen.* **11**(4), 653–663 (1978)
15. Golub, G.H., Meurant, G.: *Matrices, Moments and Quadrature with Applications*. Princeton University Press, Princeton (2010)
16. Kuijlaars, A.B.J., McLaughlin, K.T.-R.: Generic behavior of the density of states in random matrix theory and equilibrium problems in the presence of real analytic external fields. *Comm. Pure Appl. Math.* **53**, 736–785 (2000)
17. Stahl, H., Totik, V.: *General Orthogonal Polynomials*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, Cambridge (2010)
18. Mantica, G.: Quantum dynamical entropy and an algorithm by Gene Golub. *Electron. Trans. Numer. Anal.* **28**, 190–205 (2008)
19. Bellalij, M., Reichel, L., Rodriguez, G., Sadok, H.: Bounding matrix functionals via partial global block Lanczos decomposition. *Appl. Numer. Math.* **94**, 127–139 (2015)

20. Fenu, C., Reichel, L., Rodriguez, G.: GCV For Tikhonov regularization via global Golub–Kahan decomposition. *Numer. Linear Algebra Appl.* **23**(3), 467–484 (2016)
21. Fenu, C., Reichel, L., Rodriguez, G., Sadok, H.: GCV For Tikhonov regularization by partial SVD. *BIT* **57**, 1019–1039 (2017)
22. Brezinski, C., Sadok, H.: Lanczos type methods for solving systems of linear equations. *Appl. Numer. Math.* **11**, 443–473 (1993)
23. Brezinski, C.: The block Lanczos and Vorobyev methods. *C. R. Acad. Sci. Paris Ser. I*(331), 137–142 (2000)
24. Bujanovic, Z., Kressner, D.: Norm and trace estimation with random rank-one vectors. *SIAM J. Matrix Anal. Appl.* **42**(1), 202–223 (2021)
25. Cortinovis, A., Kressner, D.: On randomized trace estimates for indefinite matrices with an application to determinants. *Found. Comput. Math.* **22**(3), 875–903 (2022)
26. Golub, G.H., Strakoš, Z.: Estimates in quadratic formulas. *Numer. Algo.* **8**, 241–268 (1994)
27. Gautschi, W. In: Nevai, P. (ed.): *Computational Aspects of Orthogonal Polynomials*, pp. 181–216. Springer, Berlin (1990)
28. Golub, G.H., Welsch, J.H.: Calculation of Gauss quadrature rules. *Math. Comp* **23**, 221–230 (1969)
29. Lagomasino, G.L., Reichel, L., Wunderlich, L.: Matrices, moments, and rational quadrature. *Linear Algebra Appl.* **429**, 2540–2554 (2008)
30. Dong, S.-J., Liu, K.-F.: Stochastic estimation with Z_2 noise. *Phys. Lett. B* **328**(1–2), 130–136 (1994)
31. Hutchinson, M.F.: A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Commun. Statist. Simula.* **18**, 1059–1076 (1989)
32. Mantica, G.: A Stieltjes technique for computing Jacobi matrices associated with singular measures. *Constr. Appr.* **12**, 509–530 (1996)
33. Iitaka, T., Nomura, S., Hirayama, H., Zhao, X., Aoyagi, Y., Sugano, T.: Calculating the linear response functions of noninteracting electrons with a time-dependent schrödinger equation. *Phys. Rev. E* **56**(1), 1222–1229 (1997)
34. Jeong, H., Mason, S., Barabási, A.-L., Oltvai, Z.N.: Lethality and centrality of protein networks. *Nature* **411**, 41–42 (2001)
35. Sun, S., Ling, L., Zhang, N., Li, G., Chen, R.: Topological structure analysis of the protein-protein interaction network in budding yeast. *Nucleic Acids Res.* **31**, 2443–2450 (2003)
36. Batagelj, V., Mrvar, A.: Pajek data sets. <http://vlado.fmf.uni-lj.si/pub/networks/data> (2006). Accessed 30 Sept 2022
37. Mark Newman’s web page. <http://www-personal.umich.edu/mejn/netdata>
38. Newman, M.E.J.: The structure of scientific collaboration networks. *Proc. Natl. Acad. Sci. USA* **98**, 404–409 (2001)
39. Viswanath, B., Mislove, A., Cha, M., Gummadi, K.P.: On the evolution of user interaction in Facebook. In: 2nd ACM SIGCOMM Workshop on Social Networks (WOSN’09), pp. 37–42. Barcelona, Spain (2009)
40. The Max Planck Institute for Software Systems web site. <http://socialnetworks.mpi-sws.org/data-wosn2009.html>
41. Avron, H., Toledo, S.: Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix. *J. ACM (JACM)* **58**(2), 1–34 (2011)

Publisher’s note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.