RANDOMIZED METHOD FOR ESTIMATING THE VON NEUMANN ENTROPY OF LARGE-SCALE DENSITY MATRICES

Hayoung Choi, Xuming Song, and Yuanming Shi

School of Information Science and Technology, ShanghaiTech University, Shanghai 201210, China
Email: {hchoi, songxm, shiym}@shanghaitech.edu.cn

ABSTRACT

In this article the randomized algorithm for estimating the von Neumann entropy of large-scale density matrices is considered. By capturing the dominant eigenspace via a k-rank approximation of the density matrix we estimate the entropy. We analyze the error bound with the eigenvalues of density matrix. Numerical experiments show that the proposed method is extensively efficient for large-scale density matrices.

1. INTRODUCTION

Entropy is a fundamental quantity in many areas of science and engineering. The *von Neumann entropy*, which was introduced by John von Neumann, is the extension of classical entropy concepts to the field of quantum mechanics [1]. He introduced the notion of the density matrix, which facilitated the extension of the tools of classical statistical mechanics to the quantum domain in order to develop a theory of quantum measurements. The density matrix $\boldsymbol{\rho}$ is a symmetric positive semidefinite in $\mathbb{R}^{n\times n}$ with unit trace. Then the *von Neumann entropy*, denoted by $\mathcal{S}(\boldsymbol{\rho})$, is defined as

$$\mathcal{S}(\boldsymbol{\rho}) = -\operatorname{tr}(\boldsymbol{\rho}\log\boldsymbol{\rho}).$$

The above definition is a proper extension of both the Gibbs entropy and the Shannon entropy to the quantum case. It is obvious that the entropy can be computed by the eigendecomposition of $\boldsymbol{\rho}$ which has time complexity $O(n^3)$. Clearly, for large-scale density matrices, the algorithm is impractical. Recently, how to solve the large-scale problem efficiently has been taken more and more attentions [2, 3]. For this problem, it is also required to improve numerical algorithms that approximate the von Neumann entropy of large-scale density matrices faster than the trivial $O(n^3)$ approach [4].

1.1. Background

The Frobenius norm is denoted by $\|\boldsymbol{A}\|_2 := [\operatorname{tr}(\boldsymbol{A}^*\boldsymbol{A})]^{1/2}$.

This work was partially supported by National Nature Science Foundation of China (NSFC) under Grant No. 61601290, and Shanghai Sailing Program under Grant 16YF1407700.

In quantum theory, the density matrix $\boldsymbol{\rho}$ represents the statistical mixture of pure states and has the form

$$\boldsymbol{\rho} = \sum_{i=1}^{k} p_i \boldsymbol{\psi}_i \boldsymbol{\psi}_i^\top, \tag{1}$$

where the vectors $\boldsymbol{\psi}_i \in \mathbb{R}^n$ for all $i = 1, \ldots, k$ represent the $k \le n$ pure states and can be assumed that $\langle \boldsymbol{\psi}_i, \boldsymbol{\psi}_j \rangle = \delta_{ij}$ for i, j while the p_i's correspond to the probability of each state and satisfy $p_i > 0$ and $\sum_{i=1}^{k} p_i = 1$. Equivalently, (1) can expressed as

$$\boldsymbol{\rho} = \boldsymbol{\Psi}\boldsymbol{\Lambda}\boldsymbol{\Psi}^\top \in \mathbb{R}^{n\times n}, \tag{2}$$

where $\boldsymbol{\Psi} \in \mathbb{R}^{n\times n}$ is the orthogonal matrix (i.e., $\boldsymbol{\Psi}^\top\boldsymbol{\Psi} = \boldsymbol{I}_n$) whose first k columns are the vectors $\boldsymbol{\psi}_i$ and $\boldsymbol{\Lambda} \in \mathbb{R}^{n\times n}$ is a diagonal matrix whose top diagonal entries are p_i's and bottom $n-k$ diagonal entries are zeros. Note that (2) is the (thin) singular value decomposition of $\boldsymbol{\rho}$.

1.2. Prior work

We present and analyze randomized algorithms to approximate the von Neumann entropy of density matrices. Recently Taylor series expansion and Chebyschev polynomials are used to approximate of the matrix function $-\boldsymbol{A}\log\boldsymbol{A}$ for a large-scale matrix $\boldsymbol{A}$ [5]. Our work is motivated by [4, 6]. We show that most eigenvalues of the large-scale density matrix are close to zero, and then k-rank approximation of the density matrix can be used to approximate of the matrix function $-\boldsymbol{A}\log\boldsymbol{A}$. In [6] error bounds of k-rank approximations for trace and log determinant are considered. We generalize it for matrix functions of large-scale matrices, especially von Neumann entropy of large-scale density matrices, which was not considered in these earlier studies.

2. RANDOMIZED ESTIMATORS

2.1. Preliminary and algorithm

Let $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ be a Hermitian positive semi-definite matrix with eigenvalue decomposition

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^*, \quad \boldsymbol{\Lambda} = \operatorname{diag}(\lambda_1 \cdots \lambda_n) \in \mathbb{R}^{n\times n},$$

where the eigenvector matrix $\boldsymbol{U} \in \mathbb{C}^{n\times n}$ is unitary, and the eigenvalues are denoted as $0 \le \lambda_n \le \lambda_{n-1} \le \cdots \le \lambda_1$. We

assume that $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ has k dominant eigenvalues separated by a gap from the remaining $n-k$ sub-dominant eigenvalues, $\lambda_n \le \cdots \le \lambda_{k+1} \ll \lambda_k \le \cdots \le \lambda_1$.

To distinguish the dominant eigenvalues from the sub-dominant ones, we consider the following submatrices.

$$\boldsymbol{\Lambda} = \begin{pmatrix} \boldsymbol{\Lambda}_1 & \\ & \boldsymbol{\Lambda}_2 \end{pmatrix}, \quad \boldsymbol{U} = \begin{pmatrix} \boldsymbol{U}_1 & \boldsymbol{U}_2 \end{pmatrix},$$

where $\boldsymbol{\Lambda}_1 = \mathrm{diag}(\lambda_1 \cdots \lambda_k) \in \mathbb{R}^{k\times k}$, and $\boldsymbol{U}_1 \in \mathbb{C}^{n\times k}$. We denote the gap as

$$\gamma \equiv \frac{\lambda_{k+1}}{\lambda_k} = \|\boldsymbol{\Lambda}_2\|_2 \|\boldsymbol{\Lambda}_1^{-1}\|_2 \ll 1.$$

Given a number of power iterations $q \ge 1$, and a starting guess $\boldsymbol{\Omega} \in \mathbb{C}^{n\times \ell}$ with $k \le \ell \le n$ columns, we assume that the product has full column rank,

$$\mathrm{rank}(\boldsymbol{A}^q\boldsymbol{\Omega}) = \ell.$$

Extract an orthonormal basis for $\mathrm{range}(\boldsymbol{A}^q\boldsymbol{\Omega})$ with a thin QR decomposition $\boldsymbol{A}^q\boldsymbol{\Omega} = \boldsymbol{Q}\boldsymbol{R}$, where $\boldsymbol{Q} \in \mathbb{C}^{n\times\ell}$ with $\boldsymbol{Q}^*\boldsymbol{Q} = \boldsymbol{I}_\ell$, and the matrix $\boldsymbol{R} \in \mathbb{C}^{\ell\times\ell}$ is nonsingular. To distinguish of the effect of the dominant subspace on the starting guess from that of the sub-dominant space, partition

$$\boldsymbol{U}^*\boldsymbol{\Omega} = \begin{pmatrix} \boldsymbol{U}_1^*\boldsymbol{\Omega} \\ \boldsymbol{U}_2^*\boldsymbol{\Omega} \end{pmatrix} = \begin{pmatrix} \boldsymbol{\Omega}_1 \\ \boldsymbol{\Omega}_2 \end{pmatrix},$$

where $\boldsymbol{\Omega}_1 = \boldsymbol{U}_1^*\boldsymbol{\Omega} \in \mathbb{C}^{k\times\ell}$ and $\boldsymbol{\Omega}_2 = \boldsymbol{U}_2^*\boldsymbol{\Omega} \in \mathbb{C}^{(n-k)\times\ell}$. We assume that $\boldsymbol{\Omega}$ has a sufficient contribution in the dominant subspace of $\boldsymbol{A}$,

$$\mathrm{rank}(\boldsymbol{\Omega}_1) = k.$$

Algorithm 1 is the procedure for computing a rank-k approximation of $\boldsymbol{A}$. The algorithm includes the following steps.

Step 1. The algorithm first generates a random matrix $\boldsymbol{\Omega} \in \mathbb{C}^{n\times\ell}$.

Step 2. The random matrix $\boldsymbol{\Omega}$ is used to map the matrix A to a low-dimensional subspace by $\boldsymbol{Y} = \boldsymbol{A}^q\boldsymbol{\Omega}$. The q-th power of the matrix $\boldsymbol{A}$ is applied here to improve the accuracy for the slow decay eigenvalues.

Step 3. Compute Thin QR factorization $\boldsymbol{Y} = \boldsymbol{Q}\boldsymbol{R}$. Here columns of $\boldsymbol{Q}$ are an orthonormal basis of $\boldsymbol{Y}$.

Step 4. Compute $\boldsymbol{Q}^*\boldsymbol{A}\boldsymbol{Q}$, denoted by $\boldsymbol{A}_{app}$.

Algorithm 1 Randomized method with a single sketch

Require: $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ (Hertimian positive semi-definite matrix), k (target rank), $q \ge 1$ (number of subspace iterations), p (oversampling parameter), $\ell = k+p$ (dimension of the sketched column space).

Ensure: Find rank-k approximation $\boldsymbol{A}_{app} \in \mathbb{C}^{\ell\times\ell}$ of $\boldsymbol{A}$.

1: Generate an $n \times \ell$ random matrix $\boldsymbol{Omega} \in \mathbb{C}^{n\times\ell}$.
2: Compute $\boldsymbol{Y} = \boldsymbol{A}^q\boldsymbol{\Omega}$.
3: Find $\boldsymbol{Q}$ with Thin QR factorization $\boldsymbol{Y} = \boldsymbol{Q}\boldsymbol{R}$.
4: Compute $\boldsymbol{A}_{app} = \boldsymbol{Q}^*\boldsymbol{A}\boldsymbol{Q}$.

The idea is to capture the dominant eigenspace associated with $\lambda_1, \ldots, \lambda_k$ via a k-rank approximation $\boldsymbol{A}_{app}$ of $\boldsymbol{A}$. The matrix $\boldsymbol{Q}$ approximates the dominant eigenspace of $\boldsymbol{A}$, and is computed from q iterations of subspace iteration applied to a starting guess $\boldsymbol{\Omega}$, followed by the thin QR factorization of $\boldsymbol{A}^q\boldsymbol{\Omega}$. Mathematical and statistical analysis for $\mathrm{tr}(\boldsymbol{A}_{app}) \approx \mathrm{tr}(\boldsymbol{A})$ are studied [6]. It shows that the following absolute error bounds for Hermitian positive semi-definite matrices.

Theorem 1. (Deterministic Bound) *With the assumptions in Sec. 2.1, let $\boldsymbol{A}_{app} = \boldsymbol{Q}^*\boldsymbol{A}\boldsymbol{Q}$ be computed by Algorithm 1. Then*

$$0 \le \mathrm{tr}(\boldsymbol{A}) - \mathrm{tr}(\boldsymbol{A}_{app}) \le (1+\theta_1)\,\mathrm{tr}(\boldsymbol{\Lambda}_2),$$

where $\theta_1 \equiv \min\{\gamma^{q-1}\|\boldsymbol{\Omega}_2\boldsymbol{\Omega}_1^\dagger\|_2, \gamma^{2q-1}\|\boldsymbol{\Omega}_2\boldsymbol{\Omega}_1^\dagger\|_2^2\}$.

Here, the superscript † denotes the Moore-Penrose inverse.

Theorem 2. (Expectation) *With the assumptions in Sect. 2.1, let $\boldsymbol{A}_{app} = \boldsymbol{Q}^*\boldsymbol{A}\boldsymbol{Q}$ be computed by Algorithm 1 with a Gaussian starting guess $\boldsymbol{\Omega}^{n\times(k+p)}$ and let $p \ge 2$. Then*

$$0 \le \mathbb{E}[\mathrm{tr}(\boldsymbol{A}) - \mathrm{tr}(\boldsymbol{A}_{app})] \le (1+\gamma^{2q-1}C_{ge})\,\mathrm{tr}(\boldsymbol{\Lambda}_2),$$

where

$$C_{ge} \equiv \frac{e^2(k+p)}{(p+1)^2}\left(\frac{1}{2\pi(p+1)}\right)^{\frac{2}{p+1}}(\mu+\sqrt{2})^2\left(\frac{p+1}{p-1}\right).$$

Here, $\mu \equiv \sqrt{n-k} + \sqrt{k+p}$.

2.2. Main results

Our goal is to efficiently compute

$$\mathrm{tr}(f(\boldsymbol{A})) = \sum_{i=1}^{n} f(\lambda_i), \tag{3}$$

where $\lambda_1, \ldots, \lambda_n$ are eigenvalues of $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ and $f : \mathbb{R} \to \mathbb{R}$ is a given function. For example, if $f(x) = \log(1+x)$, then we have $\mathrm{tr}(f(\boldsymbol{A})) = \mathrm{tr}(\log(\boldsymbol{I}_n + \boldsymbol{A})) = \log\det(\boldsymbol{A})$. For more information, see [7–9]. If $f(x) = -x\log x$, then it follows that

$$\mathrm{tr}(f(\boldsymbol{A})) = -\,\mathrm{tr}(\boldsymbol{A}\log(\boldsymbol{A})). \tag{4}$$

For a given matrix $\boldsymbol{A} \in \mathbb{C}^{n\times n}$, if $f(\boldsymbol{A})$ is computed, then we can find low rank approximation of $f(\boldsymbol{A})$ with Algorithm 1. By Theorem 1, it holds that $\mathrm{tr}((f(\boldsymbol{A}))_{app}) \approx \mathrm{tr}(f(\boldsymbol{A}))$. However, if the size of $\boldsymbol{A}$ is very large, to compute $f(\boldsymbol{A})$ is practically impossible, implying that that $f(\boldsymbol{A})_{app}$ cannot be computed with Algorithm 1. One possible approach is to find p_m, a polynomial approximation with degree m, such that $p_m \approx f$ using Taylor series or Chebyschev polynomials [10]. Then $\mathrm{tr}(p_m(\boldsymbol{A}))$ can be approximated for $\mathrm{tr}(f(\boldsymbol{A}))$ [8]. In this article, we focus on a rank-k approximation $\boldsymbol{A}_{app} \approx \boldsymbol{A}$ to

approximate $\mathrm{tr}(f(\boldsymbol{A}))$. Since $\boldsymbol{A}_{app} = \boldsymbol{Q}^*\boldsymbol{A}\boldsymbol{Q}$ and $\boldsymbol{Q}\boldsymbol{Q}^* = \boldsymbol{I}_\ell$, it is easy to check that

$$f(\boldsymbol{A}_{app}) = \boldsymbol{Q}^* f(\boldsymbol{A})\boldsymbol{Q} = (f(\boldsymbol{A}))_{app}.$$

So,

$$\mathrm{tr}(f(\boldsymbol{A})) \approx \mathrm{tr}((f(\boldsymbol{A}))_{app}) = \mathrm{tr}(\boldsymbol{Q}^* f(\boldsymbol{A})\boldsymbol{Q}) = \mathrm{tr}(f(\boldsymbol{A}_{app})).$$

After finding approximation $\boldsymbol{A}_{app}$ with Algorithm 1, we need to perform two additional steps as follows:
Step 5. Since the size of $\boldsymbol{A}_{app}$ is sufficiently small, it is possible to find eigenvalues of $\boldsymbol{A}_{app}$ with any existing algorithm.
Step 6. We calculate $\mathrm{tr}(f(\boldsymbol{A}_{app}))$.

Additional Algorithm
Require: $\boldsymbol{A}_{app} \in \mathbb{C}^{\ell\times\ell}$ (the approximation matrix of $\boldsymbol{A}$).
Ensure: Approximate $\mathrm{tr}(f(\boldsymbol{A}_{app}))$.
5: Compute the eigenvalues of $\boldsymbol{A}_{app}$, denoted by $\tilde{\lambda}_\ell \le \ldots \le \tilde{\lambda}_1$.
6: Compute $\mathrm{tr}(f(\boldsymbol{A})_{app}) = \sum_{j=1}^{\ell} f(\tilde{\lambda}_j)$.

We have the similar error bound as Theorem 1 for

$$\mathrm{tr}(f(\boldsymbol{A})) \approx \mathrm{tr}(f(\boldsymbol{A})_{app}) = \mathrm{tr}(f(\boldsymbol{A}_{app})).$$

Theorem 1 shows that the error bound depends on the eigenvalue of $\boldsymbol{A}$ and $\boldsymbol{\Omega}$. Note that

$$f(\boldsymbol{A}) = \boldsymbol{U} f(\boldsymbol{\Lambda})\boldsymbol{U}^*,$$

where $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^*$ is Hermitian positive semi-definite matrix with eigenvalue decomposition. Thus, the error bound for $f(\boldsymbol{A})$ depends on the eigenvalue of $f(\boldsymbol{A})$, which is $f(\lambda_i)$ where λ_i are the eigenvalues of $\boldsymbol{A}$. In other words, the error bound for $f(\boldsymbol{A})$ depends on the eigenvalues of $\boldsymbol{A}$ and the function f. Specifically, if $f(\boldsymbol{A})_{app} = \boldsymbol{Q}^* f(\boldsymbol{A})\boldsymbol{Q}$ is computed by Algorithm 1, then by Theorem 1, it follows that

$$0 \le \mathrm{tr}(f(\boldsymbol{A})) - \mathrm{tr}(f(\boldsymbol{A}_{app})) \le (1+\theta_1)\,\mathrm{tr}(f(\boldsymbol{\Lambda}_2)),$$

where $\theta_1 \equiv \min\{\hat{\gamma}^{q-1}\|\boldsymbol{\Omega}_2\boldsymbol{\Omega}_1^\dagger\|_2, \hat{\gamma}^{2q-1}\|\boldsymbol{\Omega}_2\boldsymbol{\Omega}_1^\dagger\|_2^2\}$. Here $\hat{\gamma} := \frac{f(\lambda_{k+1})}{f(\lambda_k)}$, since the eigenvalues of f are $f(\lambda_i)$, where λ_i are the eigenvalues of $\boldsymbol{A}$.

2.3. Von Neumann entropy

Now we consider the following function for von Neumann entropy. Define a function $f : [0,1] \longrightarrow \mathbb{R}$ by

$$f(x) = \begin{cases} -x\log x & \text{if } x \neq 0 \\ 0 & \text{if } x = 0. \end{cases} \tag{5}$$

Then the Von Neumann entropy of $\boldsymbol{\rho}$ can be written as

$$\mathcal{S}(\boldsymbol{\rho}) = -\,\mathrm{tr}\,(f(\boldsymbol{\rho})),$$

where $f(\boldsymbol{\rho}) = \boldsymbol{\Psi} f(\boldsymbol{\Lambda})\boldsymbol{\Psi}^\top$.

Denote the number of all elements of a set $\boldsymbol{A}$ as $\#(\boldsymbol{A})$.

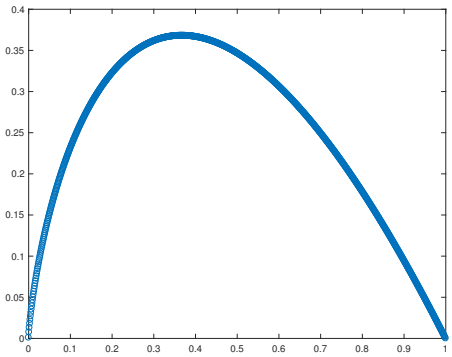

Fig. 1. $f(x) = -x\log(x)$ on $[0,1]$

Lemma 1. *Let $\boldsymbol{\rho} \in \mathbb{R}^{n\times n}$ be a density matrix with the eigenvalues $0 \le \lambda_n \le \lambda_{n-1} \le \cdots \le \lambda_1 \le 1$. Then it holds that for each $k = 1, 2, \ldots, n$*

$$\#\left\{j : \frac{1}{k} < \lambda_j \le 1\right\} < k. \tag{6}$$

Furthermore, for each $k = 1, 2, \ldots, n$ it holds that $\lambda_k \le \frac{1}{k}$.

Proof. Note that $\lambda_1 + \cdots + \lambda_n = 1$, and $0 \le \lambda_i \le 1$ for all $1 \le i \le n$. If $\lambda_k > \frac{1}{k}$, then $\sum_{j=1}^{k} \lambda_j \ge k\lambda_k > 1$, which is contradiction to $\mathrm{tr}(\boldsymbol{\rho}) = 1$. □

Equivalently, it holds that for $\varepsilon > 0$

$$\#\left\{j : 0 \le \lambda_j < \varepsilon\right\} \ge n - \frac{1}{\varepsilon}.$$

Thus, for a sufficiently large $n \times n$ density matrix, if there are distinct eigenvalues, then many of them are close to 0.

Lemma 2. *With the assumptions in Sec. 2.1, let $\boldsymbol{A}_{app} = \boldsymbol{Q}^*\boldsymbol{A}\boldsymbol{Q}$ be computed by Algorithm 1. If f is a function defined as* (5)*, then the eigenvalues of $f(\boldsymbol{A})$ are*

$$0 \le f(\lambda_n) \le \cdots \le f(\lambda_{k+1}) \le f(\lambda_k) \le \cdots \le f(\lambda_3).$$

And it holds that

$$\gamma < \hat{\gamma} \le \gamma + \frac{1}{e\log k}.$$

Proof. By Lemma 1, $0 \le \lambda_j \le \frac{1}{e}$ for $3 \le j \le n$. Since f is increasing on $[0, \frac{1}{e}]$, we have $f(\lambda_{j+1}) \le f(\lambda_j)$ for all $3 \le j \le n-1$. Note that $f(\lambda_1)$ and $f(\lambda_2)$ can possibly be small numbers, depending on f. By the definition of f and γ, it follows that

$$\hat{\gamma} = \frac{\lambda_{k+1}\log\lambda_{k+1}}{\lambda_k\log\lambda_k} = \frac{\gamma\log\lambda_{k+1}}{\log\lambda_k} = \frac{\gamma\log\gamma}{\log\lambda_k} + \gamma$$

Since $0 < f(x) \leq e^{-1}$ for all $0 \leq x < 1$,

$$0 \leq \frac{\gamma \log \gamma}{\log \lambda_k} < \frac{-1}{e \log \lambda_k}.$$

By Lemma 1, it holds that

$$\gamma \leq \hat{\gamma} < \gamma - \frac{1}{e \log \lambda_k} \leq \gamma + \frac{1}{e \log k}.$$

□

Theorem 3. *Let $\boldsymbol{\rho} \in \mathbb{R}^{n \times n}$ be a density matrix. With the assumptions in Sec. 2.1, let $\boldsymbol{\rho}_{app} = \boldsymbol{Q}^* \boldsymbol{\rho} \boldsymbol{Q} \in \mathbb{R}^{\ell \times \ell}$ be the approximation of ρ, which is computed by Algorithm 1.*

$$0 \leq \mathcal{S}(\boldsymbol{\rho}) - \mathcal{S}(\boldsymbol{\rho}_{app}) \leq (1+\theta)\,\mathrm{tr}\left(\boldsymbol{\Lambda}_2 \log(\boldsymbol{\Lambda}_2)\right),$$

where $\theta \equiv \min\left\{\eta^{q-1}\|\boldsymbol{\Omega}_2\boldsymbol{\Omega}_1^{\dagger}\|_2, \eta^{2q-1}\|\boldsymbol{\Omega}_2\boldsymbol{\Omega}_1^{\dagger}\|_2^2\right\}$ and $\eta = \gamma - \frac{1}{e \log \lambda_k}$.

Remark that if $\mathrm{rank}(\boldsymbol{\rho}) = k$, then $\lambda_j = 0$ for $k+1 \leq j \leq n$, implying $\mathcal{S}(\boldsymbol{\rho}) = \mathcal{S}(\boldsymbol{\rho}_{app})$.

3. NUMERICAL RESULTS

In this section we show numerical results in order to prove the practical efficiency of our proposed method. The proposed method were implemented in Matlab. We generated random density matrices using the QETLAB Matlab toolbox [11] to derive (real-valued) density matrices of size 4096×4096 with different rank. We used the function *RandomDensityMatrix* of QETLAB and the Haar measure.

We first compared the accuracy of our method with Monte Carlo (MC) method [5]. The accuracy was evaluated by measuring the relative error $\frac{|\mathcal{S}(\boldsymbol{\rho}) - \hat{\mathcal{S}}(\boldsymbol{\rho})|}{\mathcal{S}(\boldsymbol{\rho})}$. Consider density matrices with different rank (i.e., 4096, 2048, and 1024), $q = 1$ in our proposed sketch method and let the number of terms retained in the Taylor series approximation $m = 10$ or $m = 20$. The relative error with increasing sample size ℓ (i.e., increasing the oversampling parameter p) is shown in Fig. 2 and 3, respectively. From these two figures, initially the Taylor MC method seems to outperform our method for small sample sizes, however the error in our proposed method decays sharply. Also the overhead brings by increasing the sample size to achieve better performance than the Taylor MC method is negligible. Besides, if the density matrices have low rank, we can get superior performance with small sample size. Increasing m in the Taylor MC method can slightly improve the accuracy, but it will suffer from expensive computational time cost. We also compared the time cost of with the Taylor series approach Monte Carlo method for the full rank case in Table 1. Our proposed method is extremely time effective.

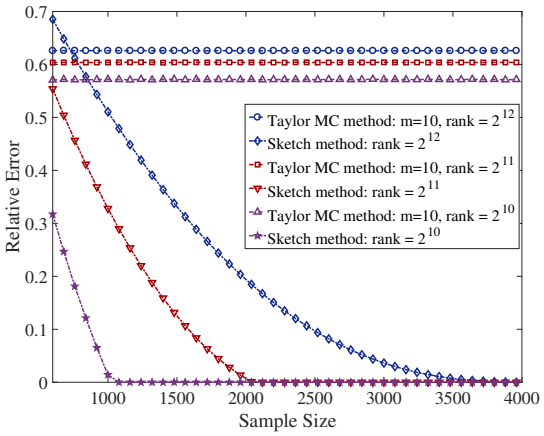

Fig. 2. Relative error with different sample size and fixed Taylor item $m = 10$.

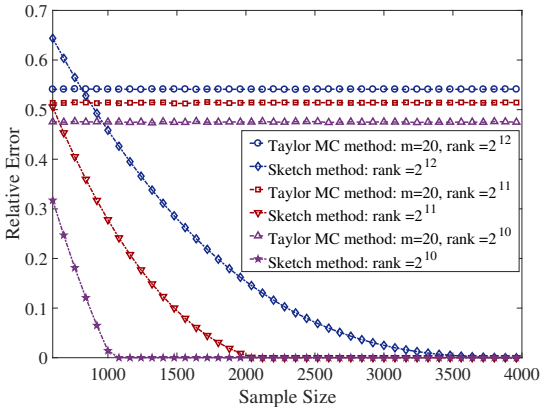

Fig. 3. Relative error with different sample size and fixed Taylor item $m = 20$.

ℓ	T-MC $m = 10$	T-MC $m = 20$	Sketch
600	1.941×10^3	6.202×10^3	1.342×10
1600	5.287×10^3	1.673×10^4	3.028×10
2600	8.442×10^3	2.657×10^4	4.059×10
3600	9.449×10^3	2.900×10^4	4.594×10

Table 1. Time cost (seconds) for different case.

4. CONCLUSION

We used the randomized low-rank approximation algorithm for estimating the von Neumann entropy of large-scale density matrices. Numerical simulations demonstrated that the proposed method outperforms state-of-art method for large-scale density matrices and its computational cost is extremely cheap.

5. REFERENCES

[1] B. C. Hall, *Quantum theory for Mathematicians*, vol. 267, Springer, 2013.

[2] Y. Shi, J. Zhang, K. B. Letaief, B. Bai, and W. Chen, "Large-scale convex optimization for ultra-dense cloud-ran," *IEEE Wireless Commun.*, vol. 22, no. 3, pp. 84–91, Jun., 2015.

[3] Y. Shi, J. Zhang, B. O'Donoghue, and K. B. Letaief, "Large-scale convex optimization for dense wireless cooperative networks," *IEEE Trans. Signal Process.*, vol. 63, no. 18, pp. 4729–4743, Sep., 2015.

[4] N. Halko, P. Martinsson, and J. Tropp, "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions," *SIAM Rev.*, vol. 53, no. 2, pp. 217 – 288, 2011.

[5] E.-M. Kontopoulou, A. Grama, W. Szpankowski, and P. Drineas, "Randomized linear algebra approaches to estimate the von neumann entropy of density matrices," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, to appear, 2018.

[6] A. K. Saibaba, A. Alexanderian, and I. C. F. Ipsen, "Randomized matrix-free trace and log-determinant estimators," *Numer. Math.*, vol. 137, no. 2, pp. 353–395, 2017.

[7] C. Boutsidis, P. Drineas, P. Kambadur, E.-M. Kontopoulou, and A. Zouzias, "A randomized algorithm for approximating the log determinant of a symmetric positive definite matrix," *Linear Algebra Appl.*, vol. 533, pp. 95 – 117, 2017.

[8] I. Han, D. Malioutov, H. Avron, and J. Shin, "Approximating spectral sums of large-scale matrices using stochastic chebyshev approximations," *SIAM J. Scientific Comput.*, vol. 39, no. 4, pp. A1558 – A1585, 2017.

[9] I. Han, D. Malioutov, and J. Shin, "Large-scale log-determinant computation through stochastic chebyshev expansions," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - vol. 37*. 2015, ICML'15, pp. 908–917, JMLR.org.

[10] R. K. Pace and J. P. LeSage, "Chebyshev approximation of log-determinants of spatial weight matrices," *Computational Statistics & Data Analysis*, vol. 45, no. 2, pp. 179 – 196, 2004.

[11] N. Johnston, "QETLAB: A MATLAB toolbox for quantum entanglement, version 0.9," http://qetlab.com, 2016.