Optimal sampling of tensor networks targeting wave function's fast decaying tails

Marco Ballarin[1,2], Pietro Silvi[1,2], Simone Montangero[1,2,3], and Daniel Jaschke[3,1,2]

[1]Dipartimento di Fisica e Astronomia "G. Galilei" & Padua Quantum Technologies Research Center, Università degli Studi di Padova, Italy I-35131, Padova, Italy

[2]INFN, Sezione di Padova, via Marzolo 8, I-35131, Padova, Italy

[3]Institute for Complex Quantum Systems, Ulm University, Albert-Einstein-Allee 11, 89069 Ulm, Germany

arXiv:2401.10330v2 [quant-ph] 14 Mar 2025

We introduce an optimal strategy to sample quantum outcomes of local measurement strings for isometric tensor network states. Our method generates samples based on an exact cumulative bounding function, without prior knowledge, in the minimal amount of tensor network contractions. The algorithm avoids sample repetition and, thus, is efficient at sampling distribution with exponentially decaying tails. We illustrate the computational advantage provided by our optimal sampling method through various numerical examples, involving condensed matter, optimization problems, and quantum circuit scenarios. Theory predicts up to an exponential speedup reducing the scaling for sampling the space up to an accumulated unknown probability ϵ from $\mathcal{O}(\epsilon^{-1})$ to $\mathcal{O}(\log(\epsilon^{-1}))$ for a decaying probability distribution. We confirm this in practice with over one order of magnitude speedup or multiple orders improvement in the error depending on the application. Our sampling strategy extends beyond local observables, e.g., to quantum magic.

In the era of the early development of quantum processors and simulators [1, 2, 3, 4, 5, 6, 7, 8, 9], a high demand arises for numerical emulators of programmable quantum devices: classical algorithms capable of replicating the input, real-time processing, and ouput of a quantum machine at small and intermediate scales [10, 11, 12, 13]. Such demand is motivated by benchmarking and certification of the programmable quantum device [14] as well as exploring the extent of quantum advantage: classical simulation of quantum machines has shown remarkable progress in recent experiments [15, 16] and new numerical algorithms have been developed to overcome existing obstacles [17, 18, 19, 20, 21, 22].

Tensor networks are often employed to numerically characterize intermediate-scale quantum systems and devices. As they store spatial correlations as compressed information, they are an invaluable tool for describing many-body quantum systems with limited entanglement content. Various classes of tensor networks have been developed so far, each adapted to a specific correlation geometry: from matrix product states (MPS) [23, 24, 25, 26, 27, 28, 29] via tree tensor networks (TTN) [30, 31, 32, 33] to projected entangled pair states (PEPS) [34, 35], and more [36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52]. Also in the context of sampling, tensor network methods have emerged as a powerful tool: Indeed, tensor network allow efficient output data acquisition from quantum states and processes [53, 54, 55]. Sampling of tensor network states has also been used to determine quantities that are otherwise difficult to measure, such quantum state *magic* [56], entanglement characterization [57], or to sample from distributions via quantum states [58]. Despite considerable recent advancements [59, 60], we identified a sampling strategy meant to outpower known methods.

In this work, we present an efficient sampling procedure, which we label Optimal tEnsor network Sampling (OPES), applicable to any isometric tensor network state [61]. The algorithm itself falls into the category of inverse transform sampling algorithms [62, 63, 64]; as additional challenge in comparison to the inverse transform sampling, we have no initial knowledge of the cumulative probability distribution or its inverse,

which even by the end is known only partially for the many-body quantum state. Our approach exploits the capability of isometric tensor networks to efficiently acquire exact conditional probabilities for partial outcome strings, and carefully avoids re-sampling of previously explored outcomes. Optimal tensor network sampling method is thus particularly suited to characterize peaked, non-flat distributions, where we observe a speedup beyond one order of magnitude over the standard sampling. The speedup to be expected depends foremost on the probability distribution, then on the number of samples or coverage to be achieved. To demonstrate the usefulness of the approach, we consider two popular classes of isometric tensor networks, namely MPS and TTN. We identify a substantial speedup both for sampling a fixed number of states as well as for covering a target portion of the probability space; while the exact probability of non-sampled space is known at all times. The OPES algorithm benefits from storing in memory all the partial tensor network contractions, which characterize the conditional probabilities, evaluated until that instant, to avoid repeating calculations. We analyze the advantage offered by the OPES method in five practical examples from different quantum applications, such as image representation, condensed matter or measurement of quantum magic. The examples include pure and mixed states. The developed code is open source and included in the *Quantum TEA* [65] library.

This manuscript is structured as follows. First, we focus on the idea underlying OPES in Sec. 1, then on the numerical implementations in Secs. 2 and 3. In Sec. 5, we introduce classes of different quantum states, each targeting different applications; we sample with the two methods from these states and analyze the speedup that one can expect in each application. Finally, we summarize our results in Sec. 6. The appendices contain the full algorithm in App. A, as well as additional implementation strategies for tensor networks, see App. B.

1 Theory and standard sampling from tensor network states

Although we compare exclusively the performance of the standard algorithm and the OPES algorithm for tensor networks in the following, the motivation originates in experimental realizations and how sampling works there. Figure 1a illustrates how experiments sample from the underlying probability distribution by running multiple shots for three qubits. The ratio of the number of shots for a state α over the total number of shots converges to the exact distribution in the limit of large number of shots. Due to the central limit theorem [66], the histogram approximates the actual probability distribution. The basis states α forming the projection in z-direction are an intuitive choice for the experiment, which for example can be sampled in an atom-qubit experiment with a fluorescence snapshot [1, 3, 9]. Similarly to the experimental results, both tensor network sampling methods can generate the number of shots per state α.

For simulations and theory, we encounter more possiblities to work with the actual underlying probability distribution of the quantum state, i.e., depicted in Fig. 1b. While sampling generates the outcome of simultaneous measured local observables of qudits, we focus on qubits as a building block for the many-body wave function and a projection onto the Pauli operator σ^z without loss of generality. The possible measurement outcomes on a single qubit are its logical states $\alpha \in \{|0\rangle, |1\rangle\}$. To build up the notation for the many-body case with a tensor network state, we consider first probabilities arising from the wave function $|\psi\rangle$ for n qubits, i.e.,

$$\begin{aligned} |\psi\rangle &= \sum_{q_0,\ldots,q_{n-1}=0}^{1} c_{q_0,\ldots,q_{n-1}} |q_0 \ldots q_{n-1}\rangle \\ &= \sum_{\alpha=0}^{2^n-1} c_\alpha |\alpha\rangle \ . \end{aligned} \tag{1}$$

The sampling algorithms aim to reproduce the probabilities $p_\alpha = |c_\alpha|^2$. We label the basis states α either with the binary number $q_0 \ldots q_{n-1}$ or its integer representation. While we focus on a pure state in Eq. (1), the concept evidently holds as well for mixed states. The order of the basis states α on the x-axis in Fig. 1b is used throughout the manuscript.

Based on this given order, the cumulative probability distribution is generated and shown in Fig. 1c. We obtain the ordered probability intervals, where we define the probability $p_\alpha = |c_\alpha|^2$

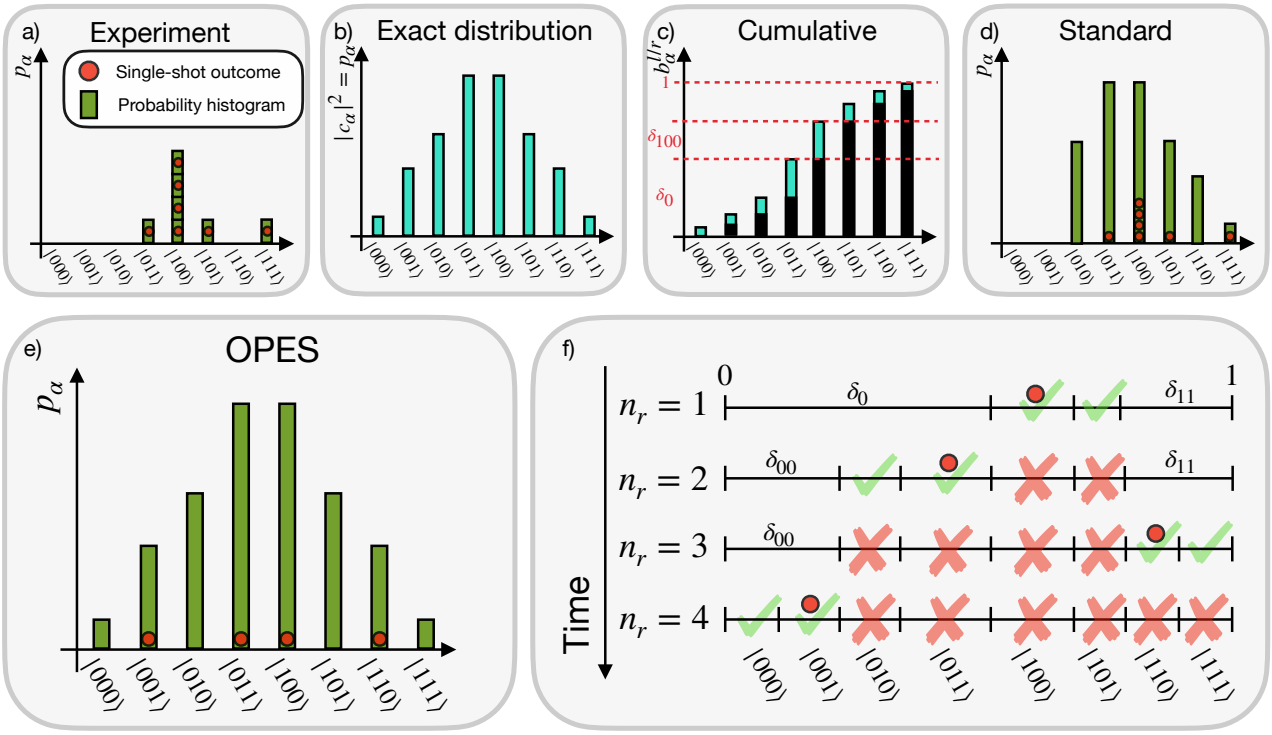


Figure 1: *Different ways of sampling a given probability distribution function encoded in an $n = 3$ qubit state.* a) The motivation to sample by shots are experiments, where the probabilities are proportional to the normalized number of shots for a given state and converge in the limit of large numbers. Thus, the probability bar in green is proportional to the shots shown as red circles. b) The actual probability distribution function is the reference distribution for any sampling method. c) One can rewrite the exact probabilities into a cumulative probability function which highlights that each state α can be associated with an interval δ_α, see also Eq. (2), where the intervals for the subsystem-state $|0\rangle_0$ and state $|100\rangle$ are indicated as δ_0 and δ_{100}. The intervals are defined via the left and right boundaries of the intervals, i.e., $\delta_\alpha = (b_\alpha^l, b_\alpha^r)$. The probability stemming from the previous states in the order is shown in black, the probability of the state itself is shown in cyan. d) The standard sampling follows a shot-by-shot approach to sample states α, but has access to the exact probabilities p_α. Additionally, probabilities of states differing only in the last qubit of the bitstring are known at no additional cost, see non-zero bars without shot (red circle). The probability of the green bars do not sum to one, which accounts for the unknown probability space. e) Using OPES, we avoid resampling and thus all shots are for different states; in detail, red circles fall into different bars. Moreover, pairs of probabilities are discovered together analog to *d)* and even the substates of the the first $(n-1)$ qubits of the shots in red are unique for each shot. Four steps are sufficient to sample a space of eight states, which generalizes to an upper bound of $\mathcal{O}(2^{n-1})$ samples to scan a full Hilbert space. The distribution of the shots sampled in this way does not converge to the true distribution anymore as there is either zero or one shot per bin; one always uses the exact probabilities. f) The OPES scaling is achieved by removing intervals from the space where the algorithm is allowed to sample; only previously not sampled states remain as potential next samples. While running the iterations top-down in this illustration, we gain partial knowledge of the cumulative probability distribution of *b)*, i.e., of the intervals of the full state when reaching the green checkmark as well as subsystem intervals as indicated. For example in step $n_\mathrm{r} = 1$, we identify the intervals δ_0 and δ_{11} for subsystems. The intervals with green checkmarks are then removed for all the following iterations, see red crosses.

as the width of an interval δ_α as

$$\delta_\alpha = \left(b^l_\alpha, b^l_\alpha + |c_\alpha|^2\right), \tag{2}$$

$$b^l_\alpha = \sum_{\beta<\alpha} |c_\beta|^2 . \tag{3}$$

The boundaries of each δ_α are well defined within the complete probability interval $(0, 1)$. Thus, the left boundary b^l_α of the probability interval associated with the state α can be defined as reported in Eq. (3). We stress that, as shown in Eq. (2), the right boundary is just $b^r_\alpha = b^l_\alpha + |c_\alpha|^2$. Inverse transform sampling algorithms can be used if these intervals are all known [62, 63, 64]. Then, one can sample a uniform random number in the interval $(0, 1)$, which serves as value on the y-axis of Fig. 1c, and finds the corresponding state α on the x-axis via the inverse cumulative probability distribution. While the inverse cumulative probability distribution is easily accessible once all p_α are known, they are initially unknown in the tensor network scenario and the number of states α grows exponentially in system size for many-body systems. But we can already anticipate based on Eq. (3) that neither all $\{|c_\beta|^2\}_{\beta\neq\alpha}$ nor even $\{|c_\beta|^2\}_{\beta<\alpha}$ need to be known. Partial knowledge can be sufficient to attribute a random number to the corresponding interval and its state α, e.g., knowing the Eq. (3) for an intermediate state with less than n qubits, i.e., the intervals $\delta_{\alpha'}$ with α' being a subsystem of α containing the first $m < n$ qubits; we explain the details in Sec. 3.

The standard approach to sample outcomes $(|c_\alpha|^2, \alpha)$ is the generation of bitstrings shot by shot. For each shot, the standard method consecutively projects the qubits into their basis state [67]. The method can also be used directly on the level of observables [68]. As we calculate probabilities shot-by-shot, we obtain the shot count for each bitstring α at the end of the standard sampling as shown in Fig. 1d. Instead of calculating the probability as ratio of shot count over number of shots, tensor networks have direct access to a pair of probabilities p_α with each shot. The pair always differs only in the measurement outcome of the last qubit. Measuring the system qubit by qubit, we accumulate a product of conditional probabilities

$$|c_\alpha|^2 = p(q_{n-1}|q_{n-2}\dots q_1 q_0)\dots p(q_1|q_0)p(q_0), \tag{4}$$

i.e., the easily accessible information about the couple $(|c_\alpha|^2, \alpha)$. The probability for only the last qubit being in the other state depends on the same probabilities of the state $|q_{n-2}\dots q_1 q_0\rangle$ and is obtained for free. Therefore, Fig. 1d shows the distribution of shots, the exact probability for all states measured via shots from Eq. (4), as well as the exact probabilities of some states without shots. The latter appear due to differing only in the last qubit from a state known due to a shot, e.g., $|010\rangle$ being obtained together with $|011\rangle$.

We observe that the standard sampling in Fig. 1d repeats the sampling of the same bitstring multiple times, while the exact probabilities already known, see state $|100\rangle$. We summarize the problems in the standard sampling approach that the OPES method overcomes as follows:

1. As the trial bitstrings $|q_0\dots q_{n-1}\rangle$ via the tensor tensor network measurement yield the probability from the exact distribution upon the first encounter, each of them might be sampled repeatedly without gaining any additional information;

2. Similarly, it is statistically unlikely to sample marginal outcomes, thus the exponentially small tails of peaked distribution either remain largely unexplored or need an exponential time to be sampled.

The fundamental idea of OPES is highlighted in Fig. 1e and f; details of OPES follow in Sec. 3. Figure 1e shows the complete knowledge of all probabilities after four shots. Because we can gain the probability of two states with each measurement of a tensor network, the optimal number of necessary measurements scales linearly with the size of the Hilbert space, i.e., as $\mathcal{O}(2^{n-1})$. To reach this scaling, we permanently remove the sampled outcome strings from the probability measure space in a way that they cannot be sampled again, as shown in Fig. 1f. After each step, we thus remove two intervals. Effectively, we renormalize the sampling probabilities of the remaining, missing outcomes. Thus, the sampling probability of a not yet sampled string increases at each shot. Let us suppose we just sampled the couple $(|c_{\alpha^*}|^2, \alpha^*)$ with probability p_{α^*} as the first sample. Then, the probabilities of sampling any $\alpha \neq \alpha^*$ at the following iteration are

$$p_{\alpha\neq\alpha^*} \to \frac{1}{1-p_{\alpha^*}} p_{\alpha\neq\alpha^*} > p_{\alpha\neq\alpha^*} , \tag{5}$$

and so forth. Thus, OPES increases the probability to draw a bitstring at every iteration which results in a faster converging strategy.

We observe an exponential speedup reducing the scaling for sampling from $\mathcal{O}(\epsilon^{-1})$ to $\mathcal{O}(\log(\epsilon^{-1}))$ for a decaying probability distribution, with ϵ^{-1} being the cumulative unknown probability. For details, see Sec. 4.

2 Standard tensor network sampling

We briefly summarize the algorithm for the standard tensor network sampling, which generates a single shot for two qubits as shown in Fig. 2a. At the end of the algorithm, we have obtained the conditional probabilities of Eq. (4). First, we draw a vector of random number$\vec{u} = (u_0, u_1, \ldots, u_{n-1}) \sim U([0,1])$, $\vec{u} \in \mathbb{R}^n$ uniformly distributed in the interval $[0,1]$. Then, we compute the probability p_0 of measuring $|0\rangle$ on the qubit 0, e.g., via the reduced density matrix ρ_0. Next, we project q_0 of the state $|\psi\rangle$ in the measured subspace based on the random number u_0,

$$q_0 = \begin{cases} |0\rangle & \text{if } u_0 < p_0\,, \\ |1\rangle & \text{if } u_0 \geq p_0\,. \end{cases} \tag{6}$$

Afterward, we repeat the procedure for the second qubit q_1 with the random number u_1 computing the conditional probability $p(q_1 = |0\rangle\,|q_0 = |0\rangle) = p_{0|0}$ of measuring q_1 in the state $|0\rangle$ conditioned on the measurement on q_0. In the two-qubit example of Fig. 2a, we have all the information to calculate the probability of the sampled state $p_{|01\rangle}$; moreover, we know the probability $p_{|11\rangle} = p_{1|1}p_0$ which depends on the same sequence of projections on the previous qubit, or qubits in the general case. We repeat this procedure until the end of the qubit chain, obtaining the probability of the state α as already pointed out in Eq. (4). For a detailed description of how these quantities are efficiently computed via reduced density matrices in an isometric tensor network, refer to App. B.

3 Optimal tensor network sampling

Optimal tensor network sampling optimizes the sampling procedure measuring each state only once, regardless of its probability. We divide the probability interval $[0,1]$ in unknown but ordered intervals $\{\delta_\alpha\}_{\alpha=0,\ldots,2^n-1}$ following the idea of the cumulative probability distribution shown in Fig. 1c. Considering one measurement as presented in Fig. 2b, we highlight the two intervals δ_{10} and δ_{11} gained for the full state as well as the interval of the partial state δ_0. We recall that the index α is the integer representation of the bitstring number. After generating the sample (δ_α, α) with OPES, we remove δ_α from the measurable intervals, and in the next iteration, we can only measure a new state $\beta \neq \alpha$. This exclusion of previously sampled states enables us to explore exponentially decaying tails efficiently.

The pedagogical and simplified algorithm for measuring simultaneous observables for two-level systems, depicted in Fig. 2b, is as follows:

1. Set the state α^* to the empty bitstring, i.e., meaning no measurement has taken place so far. Therefore, the interval to draw uniform random numbers from is $\Delta = [0,1]$; set the initial left boundary of the probability interval $b^l_{\alpha^*}$ and the state probability p_{α^*} to

$$b^l_{\alpha^*} = 0, \quad p_{\alpha^*} = 1\,. \tag{7}$$

   Sample $u_0 \sim U(\Delta)$, where $U(\Delta)$ is a uniform distribution over a given interval.

2. Compute the probability $p_0 = p(q_0 = |0\rangle)$ of measuring $|0\rangle$ on site 0 and project the wavefunction $|\psi\rangle$ on a given state according to its probability and the random number u_0:

$$q_0 = \begin{cases} |0\rangle & \text{if } u_0 < p_0, \\ |1\rangle & \text{if } u_0 \geq p_0, \end{cases} \tag{8}$$

$$|\alpha\rangle = |\alpha^*, q_0\rangle\,. \tag{9}$$

   Update the left boundary and the state probability according to the measure:

$$b^l_\alpha = \begin{cases} b^l_{\alpha^*} & \text{if } u_0 < p_0, \\ b^l_{\alpha^*} + p_0 & \text{if } u_0 \geq p_0 \end{cases} \tag{10}$$

$$p_\alpha = p_{\alpha^*} \cdot p(q_0). \tag{11}$$

3. Moving to the next qubit i, set $\alpha^* = \alpha$; the bitstring α^* now contains the qubits up to the $(i-1)^{\text{th}}$ qubit. Compute the conditional probability $p_{0|\alpha^*} = p(q_i = |0\rangle\,|\alpha^*)$. Select the measured state according to u as in the previous point and set b^l_α, p_α accord-

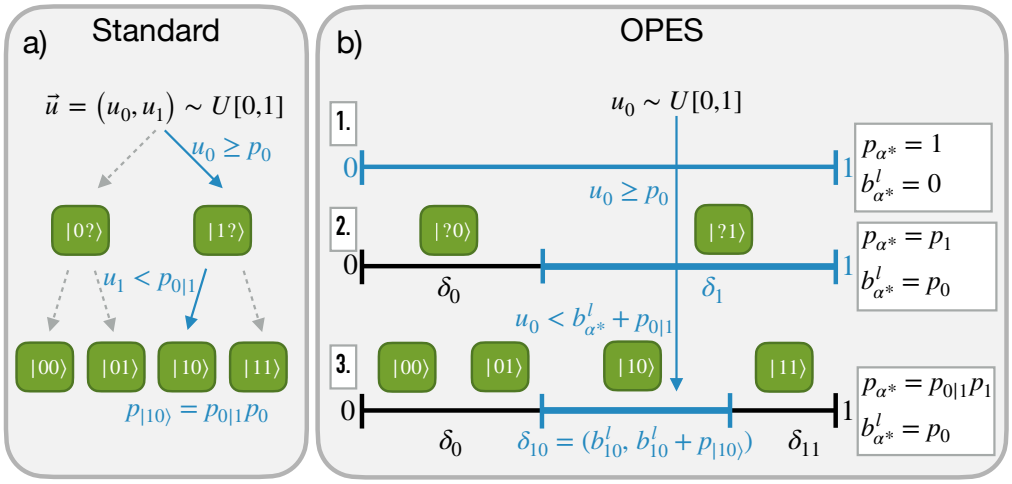


Figure 2: *Projective measurements per qubit versus OPES sampling for one sample and two qubits.* We represent with $|?\rangle$ the unknown state a qubit, with p_i the probability of measuring $|i\rangle$ on qubit 0 and with $p_{0|1}$ the probability of measuring $|0\rangle$ on qubit 1 conditioned by the measurement on qubit 0. a) We show the standard sampling, where we determine the measurement results by sampling a random number for each qubit. The resulting information is just the probability of the final state. We use the random numbers top-down, first to decide on the state of the first qubit following the blue arrow to the left; then, the random number for the second qubit samples state 0 and we follow the blue arrow to the right to the complete state $|10\rangle$. The reduced density matrix for the second qubits is calculated after using the projector $|1\rangle\langle 1|_1$ on the first qubit in this example. b) We report OPES, where a single random number is sampled for all the qubits. We collect knowledge of the probability intervals starting from an empty bitstring and the interval $(0, 1)$ containing the random number u_0, see step *1)*. In step *2)*, we obtain the intervals δ_0 and δ_1 based on the reduced density matrix of the first qubit. We apply the projector $|1\rangle\langle 1|_1$ because $u_0 \in \delta_1$ as highlighted in blue. We continue with the second qubit. By the end, we know the probability intervals δ_{10}, δ_1 with $u_0 \in \delta_{10} \in \delta_1$ while generating the bitstring, highlighted in blue; further, we know of δ_0 and δ_{11}. We enumerate the steps as in the main text, showing the evolution of the state probability p_{α^*} and its left boundary $b^l_{\alpha^*}$.

ingly, i.e.:

$$q_i = \begin{cases} |0\rangle & \text{if } u_0 < b^l_{\alpha^*} + p_{0|\alpha^*}, \\ |1\rangle & \text{if } u_0 \geq b^l_{\alpha^*} + p_{0|\alpha^*}, \end{cases} \tag{12}$$

$$|\alpha\rangle = |\alpha^*, q_i\rangle \ , \tag{13}$$

$$b^l_\alpha = \begin{cases} b^l_{\alpha^*} & \text{if } u_0 < b^l_{\alpha^*} + p_{0|\alpha^*} \ , \\ b^l_{\alpha^*} + p_{0|\alpha^*} p_{\alpha^*} & \text{if } u_0 \geq b^l_{\alpha^*} + p_{0|\alpha^*} \ , \end{cases} \tag{14}$$

$$p_\alpha = p_{\alpha^*} \cdot p(q_i|\alpha^*). \tag{15}$$

4. Repeat step *3.* until all the sites of the system are measured.
   Further details are shown in Figure 2b: we have access to the information of many nested intervals after measuring all the sites. For example, if the final result is the interval δ_{10} we have information about the following intervals:

$$\delta_{01} \in \delta_1 \in [0, 1], \ \delta_0, \delta_{11}. \tag{16}$$

We denote δ_i the interval where only the first qubit has been measured to simplify the notation. We stress that knowing δ_0 is enough to compute $b^l_{\alpha^*}$, and we do not need to know δ_{00}, δ_{01}. In general, the obtained results are:

$$b^l_\alpha = \sum_i p\left(\overline{q_i}|\{q_j\}_{j=0}^{i-1}\right) \theta(q_i - \frac{1}{2}), \tag{17}$$

$$p_\alpha = p(q_{n-1}|q_{n-2} \dots q_0) \dots p(q_1|q_0) p(q_0), \tag{18}$$

where $\theta(x)$ is the Heavyside step function and $\overline{q_i} = |1 - q_i|$.

Furthermore, we highlight that at each iteration we actually compute two probability intervals, i.e., $\delta_{q_0 q_1 \dots q_{n-2} q_{n-1}}$ and $\delta_{q_0 q_1 \dots q_{n-2} \overline{q_{n-1}}}$.

5. Remove the measured probability interval from our sampling interval, i.e., $\Delta = \Delta \setminus \delta_\alpha$. Each new sampled number $u_j \sim U(\Delta)$ explores a different probability interval $\delta_{\beta \neq \alpha}$.

This presentation of the algorithm is pedagogical for understanding the procedure and generates one sample. Steps 1) to 5) represent in the complete algorithm only a single superiteration with a number of random numbers sampled $n_r = 1$; we report the complete algorithm in Appendix A. The main differences are that for efficient computation of the intervals δ_α, multiple random numbers $\{u_j\}_{j=1,\dots,n_r}$ are drawn for each superiteration, repeating then n_S superiterations. While we do not obtain anymore one new state α for each random number sampled from $U(\Delta)$, we are able to cache intermediate results to optimize the procedure; nonetheless, we are still avoiding additional computation for the same interval. For more information about the caching strategy using the information of the intervals, see Appendix B.

As a rule of thumb, if you expect your state to have a peaked distribution with small tails you should run the first superiteration with $n_r > 1$, and then many iterations with a bigger $n_r \gg 1$ to explore the space. Instead, it is much better to always use $n_r \gg 1$ if you expect a flat distribution, since the overhead due to the cached intermediate results might reduce the performances.

4 Theoretical scalings

We proceed now to analyze the optimality of OPES and its theoretical scaling in treatable scenarios, i.e., for a flat and a decaying distribution.

4.1 Uniform distribution

In a uniform distribution, the probability of each state is $p = 1/D$ where D is the dimension of the Hilbert space. Say that we sample anyway and search for some state $|\psi\rangle$, we know that the probability to have sampled $|\psi\rangle$ after n iterations is

$$p(n) = 1 - \left(1 - \frac{1}{D}\right)^n = \frac{D^n - (D-1)^n}{D^n}. \quad (19)$$

Instead with OPES, we have

$$p_{OPES}(n) = 1 - \prod_{i=0}^{n-1}\left(1 - \frac{1}{D-i}\right) = \frac{n}{D}, \quad (20)$$

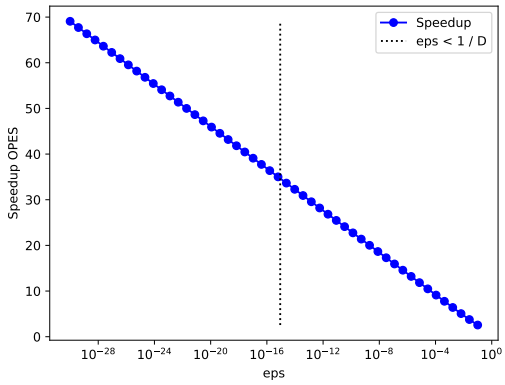


Figure 3: *Establishing scaling for uniform distribution.* Speedup for OPES sampling all space up to ϵ for a uniform distribution and 50 qubits. We observe a log speedup with ϵ.

which reflects that after $n = D$ we reach the state in any scenario. We rewrite it as ratio

$$\frac{p}{p_{OPES}} = \frac{1}{n}\sum_{i=0}^{n-1}\frac{(-1)^i\binom{n}{n-1-i}}{D^i}. \quad (21)$$

The ratio does not tell us much and is in the interval $[p(D), 1]$. We might have to change the question as ask how many samples are required to find the state $|\psi\rangle$ with a probability q. Then, we have

$$n(p(n) \geq q) = \frac{\log(1-q)}{\log\left(1 - \frac{1}{D}\right)}, \quad (22)$$

$$n_{OPES}(p_{OPES}(n) \geq q) = q \cdot D. \quad (23)$$

The expression for standard sampling diverges for $q \to 1$. The question now is how fast the function diverges. The divergence logarithmically with q.

To get an equal approach of the unsampled space ϵ, we can analyze the Poisson distribution $P_{\lambda=n/D}(k=0) = \lambda^k/k! \cdot e^{-\lambda}$ to get the number of states which have not been sampled, i.e., $D \cdot e^{-n/D}$:

$$n(\epsilon) = -D \cdot \log(\epsilon), \quad (24)$$

$$n_{OPES}(\epsilon) = D(1-\epsilon). \quad (25)$$

Assuming the overhead for bookkeeping is minimal, we have an OPES speedup of

$$S(\epsilon) = \frac{-\log(\epsilon)}{(1-\epsilon)}. \quad (26)$$

The log-speedup of the uniform distribution can be seen as lower bound for the speedop with OPES as it is the most difficult one to sample up to an ϵ.

4.2 Exponentially decaying distribution

We define the exponentially decaying distribution as follows:

$$p_i = 2^{-i}, i = 1, \dots 2^n - 1\,, \qquad p_{i=2^n} = 2^{-(2^n-1)}\,, \tag{27}$$

where the last probability $p_{i=2^n}$ ensures normalization. The number of qubits is n.

For the standard sampling, we assume that we have to sample probability intervals of the order of ϵ to reach the corresponding coverage. This statement is exact for the above example in Eq. (27). Then, the number of samples j needed to find the outcome with probability ϵ scales as

$$\mathcal{O}_{\text{Standard}} = \frac{1}{2}\mathcal{O}\left(\frac{1}{\epsilon}\right)\,, \tag{28}$$

where the factor of one-half stems from the fact of obtaining two probabilities from each sample.

Turning to the OPES algorithm, we want to avoid dealing with the renormalized probabilities after removing intervals. Therefore, we assume the samples are drawn according to their probability, highest probability first. Then, we find the number of probabilities to reach the given coverage, which is in case of Eq. (27)

$$\mathcal{O}_{\text{OPES}} = \frac{1}{2\log(2)}\mathcal{O}\left(\log\left(\frac{1}{\epsilon}\right)\right)\,. \tag{29}$$

In this specific case of Eq. (27), we can also calculate the new probabilities after j iterations, still assuming we draw them in order

$$p^j_{i>2j} = 2^{2j}\cdot p_i\,, \tag{30}$$

where the superscript indicates the iteration in the sampling. Thus, the favorable scaling of OPES originates in the log-scaling with ϵ^{-1} in comparison to the first-order polynomial scaling with ϵ^{-1} for the standard sampling. We emphasize that this arguments are independent of the size of the underlying Hilbert space.

To move forward and set up a numerical example, Eq. (27) is decaying too fast to be captured by double precision. We generalize the decaying probabilities to

$$p_i = \frac{d^{-i}}{\mathcal{N}}, i = 1, \dots 2^n\,, \tag{31}$$

where normalization is guaranteed by the scalar $\mathcal{N}$ and the decay is parameterized by d. In Fig. 4, we report an experiment with $n = 10$ qubits and $d = 1.01$, where we show that the numerical results agree with the expected scalings. The scalings from Eqs. (28) and (29) are shown as additional lines for confirmation with a suitable prefactor. If the threshold ϵ is too close to the minimal probability $\min(p_i)$ or below, the scaling breaks down as expected.

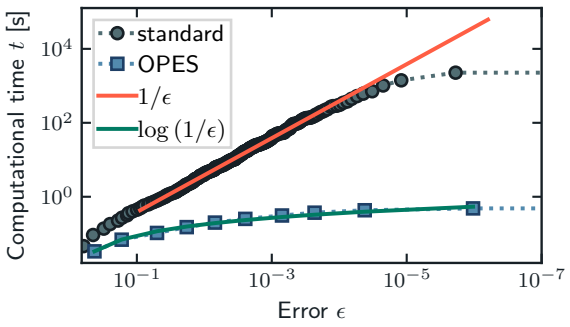


Figure 4: *Computational scaling.* We show the computational time t needed to obtain a given error ϵ for both methods. The probabilities are proportional to $p_i \propto d^{-i}$ with $d = 1.01$ for a system of $n = 10$ qubits. The exponential speedup shows in the polynomial versus log scaling with ϵ.

The generalization to other discrete probability distributions is non-trivial. For example, the discrete Gaussian distribution from Eq. (32) can be considered for large system as continuous. Even then, we obtain a scaling depending on the Gauss error function erf. Moreover, an estimate if the distribution has a smaller or larger benefit in comparison to Eq. (27) is not conclusive: in case of the discrete Gaussian, both methods should have a better scaling due to the smaller tails in the Gaussian versus Eq. (27). Therefore, a case-by-case analysis is necessary, either empirically as presented in Fig. 5, by an analytic prediction of the scaling, or numerically constructing the scaling.

5 Benchmarking sampling on applications

In this section, we consider a variety of quantum states, e.g., quantum images, quantum Ising ground states, Rydberg atoms, and quantum magic; we compare the computational time and accuracy of the sampling methods. Our results showcase the strengths of OPES over the standard sampling for states with decaying tails in the probability distribution. All computational times for the benchmarking of the sampling are referring to Cineca's *Galileo100* cluster, using a single compute node and 32 cores; the node consists of Intel CascadeLake 8260 with 2xCPU x86 Intel Xeon Platinum 8276-8276L (2.4Ghz).

5.1 Quantum state with Gaussian probability distribution with matrix product states

First, we evaluate the performance of OPES sampling on a Gaussian quantum state. The probability density of the state $|\psi\rangle$ has a Gaussian profile

$$|\psi\rangle = \frac{1}{\mathcal{N}} \sum_{i=0}^{2^n-1} \sqrt{e^{-\frac{(x_i)^2}{2\sigma^2}}} \, |i\rangle \, , \; x_i = \frac{i-\bar{i}}{2^n-1}, \quad (32)$$

where the state is a superposition of basis states $|i\rangle$ that one samples. We generate the state exactly and then convert it to an MPS representation without any truncation. The parameter n sets the dimension of the Hilbert space in terms of qubits; the standard deviation σ defines how close the state is to a delta-peak and the value $\bar{i}$ shifts the maximum of the Gaussian between states. The scalar $\mathcal{N}$ ensures that the state $|\psi\rangle$ is normalized. Then, we compare the computational time t required by two different methods to achieve a certain level of accuracy in characterizing the state distribution. Specifically, we aim to know the state's distribution up to a given threshold ϵ. Therefore, we define the coverage $\mathcal{C}$ and threshold ϵ of a set of states with $\{\delta_1, \ldots, \delta_m\}$ as

$$\mathcal{C} = \sum_{\alpha=1}^{m} |c_\alpha|^2 \, , \qquad \epsilon = 1 - \mathcal{C} \, . \quad (33)$$

We show the results for ten qubits $n = 10$, $\bar{i} = 2^{n-1} - 1$, and different values of the variance σ^2 in Fig. 5. The computational time for characterizing the probability interval up to the threshold ϵ increases for both methods as the variance increases. However, we find that OPES sampling outperforms traditional sampling significantly due to its efficient use of cached information and the tail's sampling. Let us define the speedup $\mathcal{S}$

$$\mathcal{S} = \frac{t_{\text{standard}}}{t_{\text{OPES}}}, \quad (34)$$

with t_{OPES} and t_{standard} the computational time for the OPES and standard sampling, respectively. In Fig. 5b, we show that the speedup achieved scales with the threshold ϵ, increasing from one order of magnitude for $\epsilon = 10^{-2}$ to three orders of magnitude for $\epsilon = 10^{-4}$. An in-depth argument on the scaling and speedup can be found in Sec. 4, but we have no expected theoretical scaling for a comparison in case of Eq. (32). Finally, the computational time t for OPES sampling is almost independent of the chosen coverage threshold. These results demonstrate the superiority of OPES sampling over traditional sampling when one aims to cover most of the probability distribution.

5.2 Quantum image with matrix product states

We proceed and analyze a state obtained after the application of a quantum circuit. In particular, we are not interested in exploring final states of quantum algorithms, since these algorithms are built to be easy to sample to minimize the number of measurements; examples of such algorithms are the Grover or Shor algorithms [69, 70]. We instead investigate a state obtained "at runtime". To perform operations on images using quantum hardware, we first have to load the image. There are several possible ways of doing so: amplitude encoding, angle encoding, or flexible representation of quantum images (FRQI) [71]. We focus on the latter case, encoding images using the algorithm in Ref. [72]. The question at hand is if we can reconstruct the image with high accuracy after loading it, using tensor network sampling. This procedure helps in understanding if the image is correctly reproduced even for sizes that are not accessible with the exact state vector simulation. In particular, we choose to encode the quantum tea logo, shown in Fig. 6a. The logo is a 600×600 grey-scale image with a total of $M = 360000$ pixels encoded in a 20-qubit state.

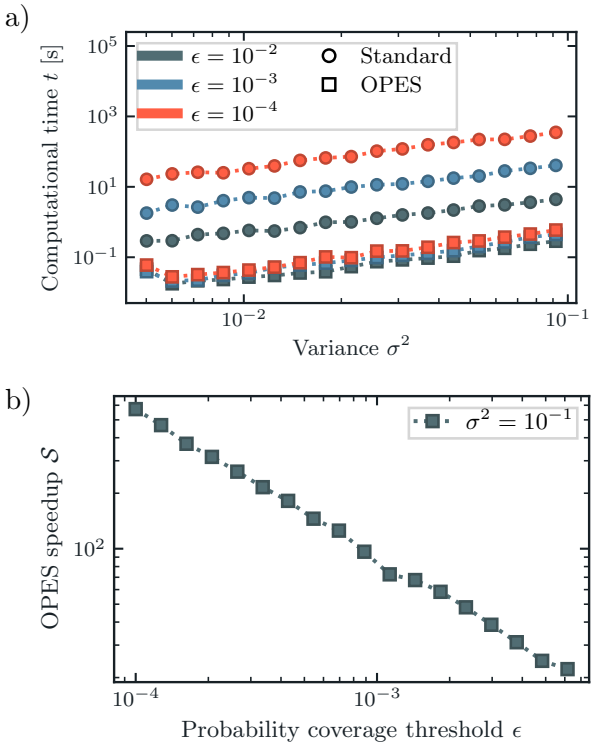


Figure 5: *Quantum state with a Gaussian probability distribution.* a) The computational time t of the standard sampling versus OPES for different probability coverage thresholds ϵ with $n = 10$ qubits. States with a higher standard deviation σ are more difficult to sample due to the tails of the distribution. b) Computational speedup $\mathcal{S}$ of the OPES sampling over the standard one with respect to ϵ at fixed variance σ^2.

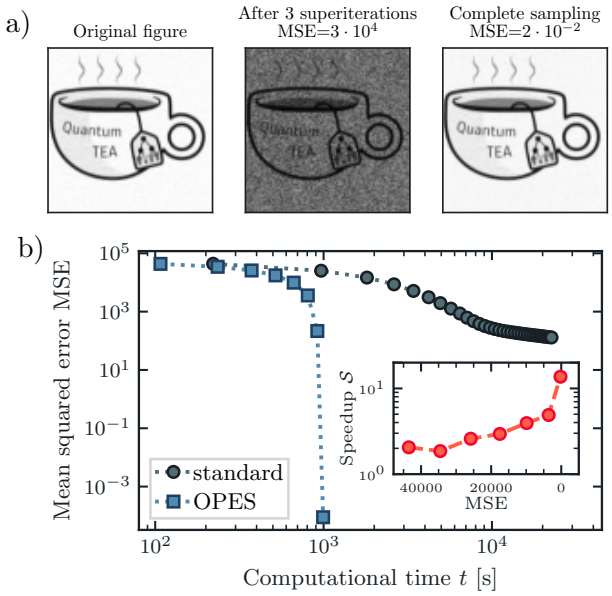


Figure 6: *Flexible representation of quantum images (FRQI) algorithm.* a) Quantum tea logos used in the flexible representation of quantum images algorithm. From left to right, we show the original image, the reconstructed image after three superiterations of the OPES sampling, the image at the end of the OPES sampling. b) Computational time t of the standard sampling versus OPES for reaching a given mean square error MSE in the image reconstruction. In the inset, the speedup of OPES over the standard method is highlighted.

We now proceed to a quantitative study of the sampling. In Fig. 6b, we show the computational time t needed to reach a given mean square error MSE with the original image, defined as:

$$\mathrm{MSE}(t) = \frac{1}{M}\sum_{i=1}^{M}\left(\mathrm{P}^i_{sv} - \mathrm{P}^i_{sm}(t)\right)^2, \qquad (35)$$

where P_{sv} is the picture obtained from the complete state vector of the state and $\mathrm{P}_{sm}(t)$ is the sampled picture at time t. The index i runs over the pixels of the image. Thus, the MSE is improved by over two orders of magnitude in the figure of merit while saving one order of magnitude in computational time. We also show the logos retrieved from the quantum state at different superiterations of the OPES sampling in Fig. 6a. The FRQI algorithm benefits in our opinion the most from the OPES sampling out of our examples. As we see in the inline of Fig. 6b, the speedup $\mathcal{S}$ of OPES over the standard method is exponential in the MSE, which is agreement with Sec. 4.

5.3 Quantum Ising model with tree tensor networks

As the next example, we consider the quantum Ising model [73] as an example from condensed matter. We use the definition of the quantum Ising model as

$$H = J\left(-\sum_{i=1}^{n-1}\sigma_i^z\sigma_{i+1}^z - g\sum_{i=1}^{n}\sigma_i^x\right), \qquad (36)$$

where the parameter J set the energy scale and g tunes the external field, respectively. The Pauli matrices acting on site i are σ_i^α. From the two limits, we expect three regions with respect to the sampling: (i) close to the ferromagnetic limit, the states almost align with our measurement basis; (ii) close to the paramagnetic limit, the local product states are orthogonal to the local measurement basis; and (iii) the region around the quantum critical point with high entanglement. The latter two are thus potentially difficult to

sample, especially in the case of the paramagnetic limit where the probability of each state scales inversely with the dimension of the Hilbert space.

The sampling of a fixed number of samples n_r within one superiteration scales better either for the standard or OPES approach according to Fig. 7a, depending on the quantum phase represented by one point in the ferromagnetic and paramagnetic phase, see $g = 0.5$ and $g = 1.5$ respectively, and an additional point close to the quantum critical point with $g = 1$. We consider a system size of $n = 64$. The probability distribution for $g = 1$ and $g = 1.5$ are too similar to a uniform distribution and OPES does not show a benefit here, but is even slower due to a potential overhead necessary for caching intermediate results. If we aim to obtain a certain coverage $\mathcal{C}$ of the probability space instead of a fixed number of iterations, the task of reaching the coverage becomes computationally more expensive, especially for flat-distributions. For example, we have to sample in an exponentially large space in the paramagnetic limit because the local eigenstates are orthogonal to our measurement basis. Keeping the measurement basis fixed throughout helps to understand the scaling for other models where a change into the "good" measurement basis might not be obvious. But we reduce the number of qubits to $N = 16$ and aim for coverages $\mathcal{C} = 30\%$ and $\mathcal{C} = 50\%$ in Fig. 7b; without recucing the number of qubits N, we need to sample 2^{N-1} states in the paramagnetic limit for $N = 64$ to reach $\mathcal{C} = 50\%$. The better scaling of the OPES method for all fields g originates from the fact that a new iteration does not resample states from previous iterations. In contrast, the standard method has a probability of almost 50% to throw a state away because the generated state is already known when reaching 50% coverage. Thus, the OPES still pays off in flat distributions for high coverages $\mathcal{C}$, though this scenario is restricted to moderate number of qubits due to the exponential growth of the Hilbert space.

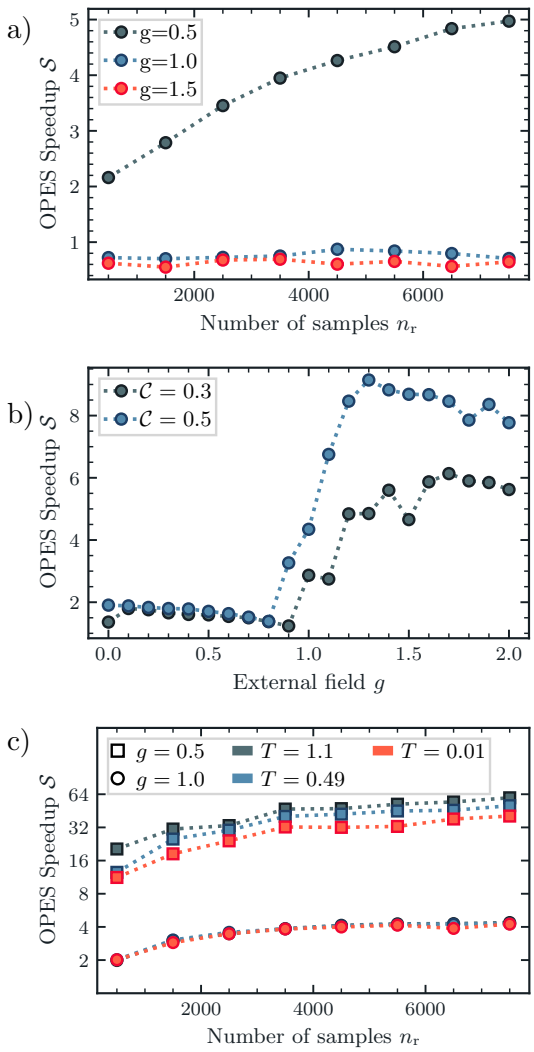


Figure 7: *Comparisons for the quantum Ising model and tree tensor networks.* We calculate the speedup $\mathcal{S}$ of OPES as the ratio of the computational time of the standard method over OPES. a) We sample n_r states within one superiteration from the ground state at the critical point $g = g_c = 1.0$ for 64 qubits. b) The speedup $\mathcal{S}$ to reach a coverage $\mathcal{C}$ for 16 qubits across the phase diagram. The paramagnetic phase is difficult as the eigenstate does not align with the measurement basis. c) The speedup $\mathcal{S}$ to produce n_r samples within one superiteration for the density matrix at finite temperatures $T \in [1.1, 0.49, 0.01]$ and two external fields g.

The third example moves from pure states, which we consider throughout all the other examples, to density matrices. We generate finite temperature density matrices for the external fields $g = 0.5$ and $g = 1.0$ via an imaginary time-dependent variational principle (TDVP) time evolution [74] with a locally purified tensor network (LPTN) [45], which is then converted into a tree tensor operator (TTO) [49]. The behavior for the external field resembles the ground state, i.e., OPES shows a greater benefit inside the ferromagnetic phase for the data points with $g = 0.5$. Surprisingly, the data indicates a bigger speedup at higher temperatures although the high-temperature limit itself is the

completely-mixed state with a flat distribution. In summary, any problem where a lower limit on the unsampled space is required can profit significantly from OPES even in scenarios similar to the condensed matter case across all phases treated here.

5.4 Crosstalk errors in Rydberg atoms with tree tensor networks

An error analysis is useful for the benchmarking of quantum processing units, e.g., for the state preparation. In such a scenario, we expect to be almost perfectly in the target state, where the remaining probability represents the error. Depending on the fidelity of the preparation, the measurement of a state with error has a small probability. The preparation of the Greenberger-Horne-Zeilinger state (GHZ) with errors including the projective measurement is shown in Ref. [14] and we consider here the aspect of the computational scaling of analyzing such a state. Figure 8a shows the computational time t for an imperfect 64-qubit GHZ state, where errors have been introduced during the protocol due to crosstalk; while the computation time for projective measurements scales with the number of samples for the standard approach, the OPES approach has an almost constant computational time t. This almost constant time t is related to the two large probabilities for the states $|00\ldots00\rangle$ and $|11\ldots11\rangle$. This scenario is one of the most favorable for OPES.

We further extract the coverage $\mathcal{C}$ as defined in Eq. (33) from the quantum state for each sampling method, i.e., how much of the probability space has not been sampled. This data could be used to generate data according to a quantum state's probability distribution up to a certain accuracy even without a tensor network representation. Figure 8b shows how the coverage improves over time with the superiterations of the algorithm, while we sample $n_\mathrm{r} = 250$ states per superiteration for the default approach in comparison to $n_\mathrm{r} = 25$ states for OPES. As OPES can target exclusively previously unexplored probability intervals, we observe a significant advantage.

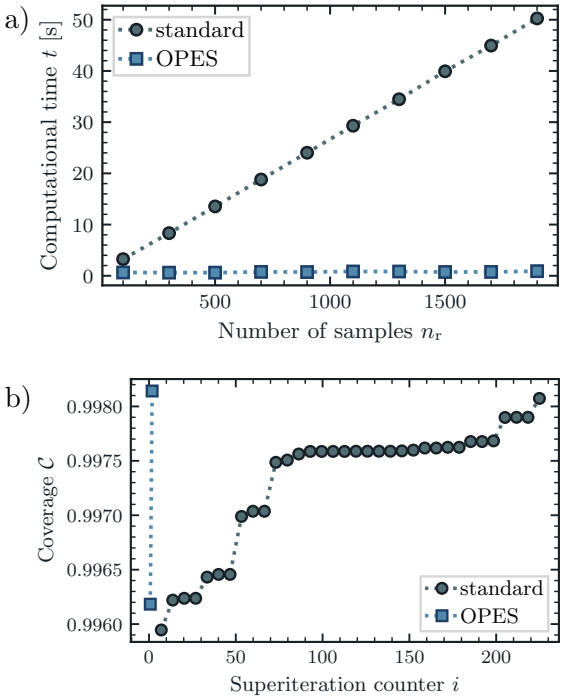


Figure 8: *Comparison for sampling an imperfect GHZ prepared under crosstalk.* The state is obtained from a simulation of Rydberg atoms encountering crosstalk during the preparation, which leads to deviations from the GHZ state. a) We sample a fixed number of samples n_r within one superiteration, where the major probability in the states $|00\ldots00\rangle$ and $|11\ldots11\rangle$ leads to the better scaling of OPES. b) We aim for a coverage $\mathcal{C} \geq 99.8\%$ and show the coverage $\mathcal{C}$ during the superiterations of the algorithms.

5.5 Magic with matrix product states

To stress the generality of the method mentioned in Sec. 1, we now generalize the sampling procedure to measure the quantum magic of a quantum state, also known as non-stabilizerness. Quantum magic is an important quantity since it limits the classical simulation of quantum systems for stabilizer simulators [75, 76], similarly to how the entanglement limits the simulations with tensor networks. Quantum magic can be measured through Stabilizer Rényi Entropies (SREs) [77, 78]. Reference [56] introduces a technique to measure the SREs through standard sampling of an MPS state. Thus, applying OPES can speed up the process of measuring quantum magic. The magic of a n-qubits state using the m-order SREs is defined as:

$$\mathcal{M}_m = \frac{1}{1-m} \log\left(\frac{1}{\mathcal{N}} \sum_{\mu=0}^{\mathcal{N}} \Pi(\vec{\sigma})^{m-1}\right) - n \log 2, \tag{37}$$

where $\Pi(\vec{\sigma})$ is the probability of measuring a given Pauli string, $\sigma_i \in \{\mathbb{I}, X, Y, Z\}$. To apply OPES to measure $\mathcal{M}$, we target the $\Pi(\vec{\sigma})$, keeping into account that each measure of probability on the single site $\pi(\sigma_i)$ branches into four possibilities ($\sigma_i \in \{\mathbb{I}, X, Y, Z\}$) instead of the two possibilities ($b_i = \{|0\rangle, |1\rangle\}$) treated up to now. For an extensive explanation of how to measure $\pi(\sigma_i)$ efficiently with an MPS, please refer to Ref. [56]. As an example, we compute the order-1 SREs of the state

$$|\psi(\phi)\rangle = \left(\frac{|0\rangle + e^{i\phi}|1\rangle}{\sqrt{2}}\right)^{\otimes n}. \tag{38}$$

Notice that $|\psi(0)\rangle$ for $\phi = 0$ is a stabilizer state, and thus has zero magic. The magic of the state is instead maximum for $\phi = \frac{\pi}{4}$. We call the exact 1-SREs of this state $\widetilde{\mathcal{M}}$:

$$\begin{aligned}\widetilde{\mathcal{M}} = &- \cos^2\phi \log\left(|\cos\phi|\right) \\ &- \sin^2\phi \log\left(|\sin\phi|\right).\end{aligned} \tag{39}$$

In Fig. 9, we address the computation of $\mathcal{M}_1$ for $n = 10$ qubits using for OPES $n_S = 10$ superiterations with $n_r = 100$ samples each. The results are mediated over 10 realization of the computations. We use for the standard method the same number of samples. We notice that, due to the flatness of the distribution, OPES gives no relevant improvement when $\phi = \frac{\pi}{64} \sim 0$. This result is to be expected, as already discussed for the paramagnetic phase of the quantum Ising model in Sec. 5.3. However, we can observe both an improvement in the estimation of the magic and a speedup in the other two cases, i.e., $\phi = \frac{\pi}{8}$ and $\phi = \frac{\pi}{4}$. Optimal tensor network sampling is twice as fast as the standard method, and the estimation of the quantum magic converges much faster with respect to the standard method.

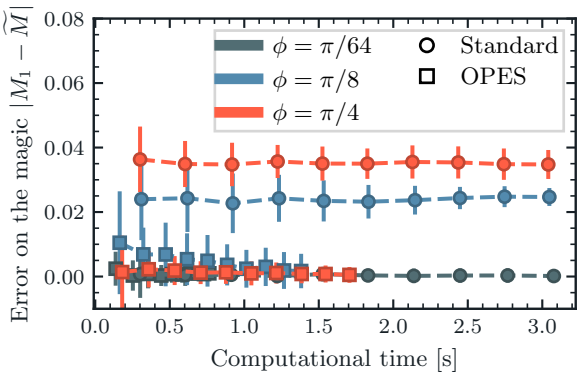


Figure 9: *Magic estimation through sampling.* We compare the standard and OPES methods in computing the magic of the state known by its theory formula, see $\widetilde{\mathcal{M}}$ in Eq. (39), for $n = 10$ qubits and different parameters ϕ with the same number of samples. The points are obtained as an average over ten realizations. Optimal tensor network sampling is always at least twice as fast as the standard method and converges faster to the true value $\mathcal{M}_1$ for $\phi = \frac{\pi}{8}$ and $\phi = \frac{\pi}{4}$.

6 Conclusions

Overall, we have introduced the optimal tensor network sampling (OPES), a sampling method for tensor network states that avoids sampling known intervals and caches information for optimal performance. From the computational statistics point of view, the algorithm represents an extension to the inverse transform sampling, which is used to sample pseudo-random numbers from an arbitrary probability distribution function by knowing its inverse probability distribution function. The extension consists in the fact that we never fully know the cumulative distribution function or its inverse of the quantum state represented as a tensor network. The method is applicable to all isometric tensor network states as well as exact diagonalization. We have investigated several quantum states and compared the performance of the standard and OPES methods in characterizing them; these results are obtained for matrix product states (MPS), tree tensor networks (TTN), and tree tensor operators (TTO). We compare to the standard sampling method. An enhanced standard sampling, as available in Ref. [79], can share some of the features of OPES as the speedup at the cost of sampling as well as storing more random numbers and loosing convenience in the numerical implementation.

In detail, we have shown that OPES offers a significant speedup compared to the standard method, especially when covering most of the probability distribution is demanded. On the one hand, this speedup is due to the efficient caching of previous information employing the MPS or TTN; on the other hand, we work with the underlying probability intervals of the full Hilbert space which allow us to exclude previously explored intervals from future superiterations. For example, OPES leads to a speedup of more than an order of magnitude while achieving lower er-

rors versus the standard sampling for applications as the flexible representation of quantum images (FRQI) algorithm. We have shown how the approach is generalizable to the measure of magic with matrix product state, and its speedup over standard methods. The speedup in computation time also leads to a reduced integrated energy consumption of the CPU as our additional data in Ref. [80] shows for one example.

Future directions of this approach for sampling can go into the direction of comparing parallelized versions, optimizing the caching strategy, or exploring the potential benefit in other algorithms and applications. While the standard sampling trivially parallelizes via data parallelism, OPES can, for example, parallelize over the groups of the sorted random numbers. For the application side, there exists a plethora of candidates with potential speedup where the examples given within this manuscript give a first intuition which applications benefit most from OPES, i.e., quantum states with a decaying probability distribution. Furthermore, since OPES only needs the computation of expectation values of local operators and marginal probabilities it could be applied to any isometric tensor network such as isoPEPS [61]. Similarly, the strings of projective measurements can be naturally replaced with Positive Operator-Valued Measures (POVM).

Code and data availability — The code implementing the algorithm for both MPS and TTNs is distributed through the quantum tea leaves [65] python package of *Quantum TEA*, while the code and data to generate the plots of this work is available via zenodo [81, 82]. All figures are available together with metadata under a CC-BY license at [80].

Acknowledgments — We thank Nora Reinić for providing finite-temperature states in the tree tensor operator representation. We thank Giovanni Cataldi, Marcello Dalmonte, Martina Frau, Marco Rigobello, Ilaria Siloi, Poetri Tarabunga, Marco Tesoro, and Emanuele Tirrito for discussions and feedback. We acknowledge financial support from the Italian Ministry of University and Research (MUR) via PRIN2022 project TANQU, and the Departments of Excellence grant 2023-2027 Quantum Frontiers; from the European Union via H2020 projects EuRyQa, and TEXTAROSSA, the QuantERA projects QuantHEP and T-NISQ, and the Quantum Flagship project Pasquans2, INFN project Quantum, from the German Federal Ministry of Education and Research (BMBF) via the funding program quantum technologies − from basic research to market − project QRydDemo, and from the World Class Research Infrastructure − Quantum Computing and Simulation Center (QCSC) of Padova University. This work was performed in part at the Aspen Center for Physics, which is supported by National Science Foundation grant PHY-2210452; the participation of D. J. at the Aspen Center for Physics was supported by the Simons Foundation. We also acknowledge computation time supported support by the state of Baden-Württemberg through bwHPC and the German Research Foundation (DFG) through grant no INST 40/575-1 FUGG (JUSTUS 2 cluster), the Atos Dibona cluster via TEXTAROSSA, as well as computation time on Cineca's *Galileo100* machine.

References

[1] C. Monroe. "Quantum information processing with atoms and photons". Nature **416**, 238–246 (2002).

[2] C. Monroe and J. Kim. "Scaling the ion trap quantum processor". Science **339**, 1164–1169 (2013).

[3] Mark Saffman. "Quantum computing with neutral atoms". National Science Review **6**, 24–25 (2018).

[4] G. Wendin. "Quantum information processing with superconducting circuits: a review". Reports on Progress in Physics **80**, 106001 (2017).

[5] Colin D. Bruzewicz, John Chiaverini, Robert McConnell, and Jeremy M. Sage. "Trapped-ion quantum computing: Progress and challenges". Applied Physics Reviews **6**, 021314 (2019).

[6] C. S. Adams, J. D. Pritchard, and J. P. Shaffer. "Rydberg atom quantum technologies". Journal of Physics B: Atomic, Molecular and Optical Physics **53**, 012002 (2019).

[7] V. Kaushal, B. Lekitsch, A. Stahl, J. Hilder, D. Pijn, C. Schmiegelow, A. Bermudez, M. Müller, F. Schmidt-Kaler, and U. Poschinger. "Shuttling-based trapped-ion

quantum information processing". AVS Quantum Science **2**, 014101 (2020).

[8] Morten Kjaergaard, Mollie E. Schwartz, Jochen Braumüller, Philip Krantz, Joel I.-J. Wang, Simon Gustavsson, and William D. Oliver. "Superconducting qubits: Current state of play". Annual Review of Condensed Matter Physics **11**, 369–395 (2020).

[9] M. Morgado and S. Whitlock. "Quantum simulation and computing with Rydberg-interacting qubits". AVS Quantum Science **3**, 023501 (2021).

[10] Alexandra Nagy and Vincenzo Savona. "Variational quantum Monte Carlo method with a neural-network ansatz for open quantum systems". Physical review letters **122**, 250501 (2019).

[11] Qiskit contributors. "Qiskit: An open-source framework for quantum computing" (2023).

[12] The cuQuantum development team. "NVIDIA cuQuantum SDK". Zenodo (2023).

[13] Stavros Efthymiou, Marco Lazzarin, Andrea Pasquale, and Stefano Carrazza. "Quantum simulation with just-in-time compilation". Quantum **6**, 814 (2022).

[14] Daniel Jaschke, Alice Pagano, Sebastian Weber, and Simone Montangero. "Ab-initio tree-tensor-network digital twin for quantum computer benchmarking in 2d". Quantum Science and Technology **9**, 035055 (2024).

[15] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C. Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando G. S. L. Brandao, David A. Buell, Brian Burkett, Yu Chen, Zijun Chen, Ben Chiaro, Roberto Collins, William Courtney, Andrew Dunsworth, Edward Farhi, Brooks Foxen, Austin Fowler, Craig Gidney, Marissa Giustina, Rob Graff, Keith Guerin, Steve Habegger, Matthew P. Harrigan, Michael J. Hartmann, Alan Ho, Markus Hoffmann, Trent Huang, Travis S. Humble, Sergei V. Isakov, Evan Jeffrey, Zhang Jiang, Dvir Kafri, Kostyantyn Kechedzhi, Julian Kelly, Paul V. Klimov, Sergey Knysh, Alexander Korotkov, Fedor Kostritsa, David Landhuis, Mike Lindmark, Erik Lucero, Dmitry Lyakh, Salvatore Mandrà, Jarrod R. McClean, Matthew McEwen, Anthony Megrant, Xiao Mi, Kristel Michielsen, Masoud Mohseni, Josh Mutus, Ofer Naaman, Matthew Neeley, Charles Neill, Murphy Yuezhen Niu, Eric Ostby, Andre Petukhov, John C. Platt, Chris Quintana, Eleanor G. Rieffel, Pedram Roushan, Nicholas C. Rubin, Daniel Sank, Kevin J. Satzinger, Vadim Smelyanskiy, Kevin J. Sung, Matthew D. Trevithick, Amit Vainsencher, Benjamin Villalonga, Theodore White, Z. Jamie Yao, Ping Yeh, Adam Zalcman, Hartmut Neven, and John M. Martinis. "Quantum supremacy using a programmable superconducting processor". Nature **574**, 505–510 (2019).

[16] Han-Sen Zhong, Hui Wang, Yu-Hao Deng, Ming-Cheng Chen, Li-Chao Peng, Yi-Han Luo, Jian Qin, Dian Wu, Xing Ding, Yi Hu, Peng Hu, Xiao-Yan Yang, Wei-Jun Zhang, Hao Li, Yuxuan Li, Xiao Jiang, Lin Gan, Guangwen Yang, Lixing You, Zhen Wang, Li Li, Nai-Le Liu, Chao-Yang Lu, and Jian-Wei Pan. "Quantum computational advantage using photons". Science **370**, 1460–1463 (2020).

[17] Feng Pan and Pan Zhang. "Simulation of quantum circuits using the big-batch tensor network method". Phys. Rev. Lett. **128**, 030501 (2022).

[18] Feng Pan, Keyang Chen, and Pan Zhang. "Solving the sampling problem of the sycamore quantum circuits". Phys. Rev. Lett. **129**, 090502 (2022).

[19] Yong (Alexander) Liu, Xin (Lucy) Liu, Fang (Nancy) Li, Haohuan Fu, Yuling Yang, Jiawei Song, Pengpeng Zhao, Zhen Wang, Dajia Peng, Huarong Chen, Chu Guo, Heliang Huang, Wenzhao Wu, and Dexun Chen. "Closing the "quantum supremacy" gap: Achieving real-time simulation of a random quantum circuit using a new sunway supercomputer". In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. SC '21New York, NY, USA (2021). Association for Computing Machinery.

[20] Jacob F. F. Bulmer, Bryn A. Bell, Rachel S. Chadwick, Alex E. Jones, Diana Moise, Alessandro Rigazzi, Jan Thorbecke, Utz-Uwe Haus, Thomas Van Vaerenbergh, Raj B. Patel, Ian A. Walmsley, and Anthony

Laing. "The boundary for quantum advantage in Gaussian boson sampling". Science Advances **8**, eabl9236 (2022).

[21] Andrew J. Daley, Immanuel Bloch, Christian Kokail, Stuart Flannigan, Natalie Pearson, Matthias Troyer, and Peter Zoller. "Practical quantum advantage in quantum simulation". Nature **607**, 667–676 (2022).

[22] Changhun Oh, Minzhao Liu, Yuri Alexeev, Bill Fefferman, and Liang Jiang. "Tensor network algorithm for simulating experimental Gaussian boson sampling" (2023). url: `https://doi.org/10.48550/arXiv.2306.03709`.

[23] Steven R. White. "Density matrix formulation for quantum renormalization groups". Phys. Rev. Lett. **69**, 2863–2866 (1992).

[24] Stefan Rommer and Stellan Östlund. "Class of ansatz wave functions for one-dimensional spin systems and their relation to the density matrix renormalization group". Phys. Rev. B **55**, 2164–2181 (1997).

[25] Stefan Rommer and Stellan Östlund. "Thermodynamic limit and matrix-product states". In Density-Matrix Renormalization. Pages 67–89. Berlin, Heidelberg (1999). Springer Berlin Heidelberg.

[26] U. Schollwöck. "The density-matrix renormalization group". Rev. Mod. Phys. **77**, 259–315 (2005).

[27] D. Perez-Garcia, F. Verstraete, M. M. Wolf, and J. I. Cirac. "Matrix Product State Representations". Quantum Information and Computation **7**, 401–430 (2007).

[28] Ian P. McCulloch. "From density-matrix renormalization group to matrix product states". Journal of Statistical Mechanics: Theory and Experiment **2007**, P10014 (2007).

[29] Ulrich Schollwöck. "The density-matrix renormalization group in the age of matrix product states". Annals of Physics **326**, 96 – 192 (2011).

[30] Y.-Y. Shi, L.-M. Duan, and G. Vidal. "Classical simulation of quantum many-body systems with a tree tensor network". Phys. Rev. A **74**, 022320 (2006).

[31] L. Tagliacozzo, G. Evenbly, and G. Vidal. "Simulation of two-dimensional quantum systems using a tree tensor network that exploits the entropic area law". Phys. Rev. B **80**, 235127 (2009).

[32] V. Murg, F. Verstraete, Ö. Legeza, and R. M. Noack. "Simulating strongly correlated quantum systems with tree tensor networks". Phys. Rev. B **82**, 205105 (2010).

[33] M. Gerster, P. Silvi, M. Rizzi, R. Fazio, T. Calarco, and S. Montangero. "Unconstrained tree tensor network: An adaptive gauge picture for enhanced performance". Phys. Rev. B **90**, 125154 (2014).

[34] Verstraete F., Murg V., and Cirac J.I. "Matrix product states, projected entangled pair states, and variational renormalization group methods for quantum spin systems". Advances in Physics **57**, 143–224 (2008).

[35] Román Orús. "A practical introduction to tensor networks: Matrix product states and projected entangled pair states". Annals of Physics **349**, 117 – 158 (2014).

[36] Michael Zwolak and Guifré Vidal. "Mixed-state dynamics in one-dimensional quantum lattice systems: A time-dependent superoperator renormalization algorithm". Phys. Rev. Lett. **93**, 207205 (2004).

[37] F. Verstraete, J. J. García-Ripoll, and J. I. Cirac. "Matrix product density operators: Simulation of finite-temperature and dissipative systems". Phys. Rev. Lett. **93**, 207204 (2004).

[38] Matteo Rizzi, Simone Montangero, and Guifre Vidal. "Simulation of time evolution with multiscale entanglement renormalization ansatz". Phys. Rev. A **77**, 052328 (2008).

[39] Lukasz Cincio, Jacek Dziarmaga, and Marek M. Rams. "Multiscale entanglement renormalization ansatz in two dimensions: Quantum Ising model". Phys. Rev. Lett. **100**, 240603 (2008).

[40] I. P. McCulloch. "Infinite size density matrix renormalization group, revisited" (2008). arXiv:0804.2509.

[41] J. Jordan, R. Orús, G. Vidal, F. Verstraete, and J. I. Cirac. "Classical simulation of

infinite-size quantum lattice systems in two spatial dimensions". Phys. Rev. Lett. **101**, 250602 (2008).

[42] G. Evenbly and G. Vidal. "Algorithms for entanglement renormalization". Phys. Rev. B **79**, 144108 (2009).

[43] G. Evenbly and G. Vidal. "Class of highly entangled many-body states that can be efficiently simulated". Phys. Rev. Lett. **112**, 240502 (2014).

[44] Ho N. Phien, Johann A. Bengua, Hoang D. Tuan, Philippe Corboz, and Román Orús. "Infinite projected entangled pair states algorithm improved: Fast full update and gauge fixing". Phys. Rev. B **92**, 035142 (2015).

[45] A. H. Werner, D. Jaschke, P. Silvi, M. Kliesch, T. Calarco, J. Eisert, and S. Montangero. "Positive tensor network approach for simulating open quantum many-body systems". Phys. Rev. Lett. **116**, 237201 (2016).

[46] Pietro Silvi, Ferdinand Tschirsich, Matthias Gerster, Johannes Jünemann, Daniel Jaschke, Matteo Rizzi, and Simone Montangero. "The Tensor Networks Anthology: Simulation techniques for many-body quantum lattice systems". SciPost Phys. Lect. NotesPage 8 (2019).

[47] S. Montangero. "Introduction to tensor network methods". Springer Nature Switzerland AG. Cham, CH (2018).

[48] Daniel Jaschke, Simone Montangero, and Lincoln D. Carr. "One-dimensional many-body entangled open quantum systems with tensor network methods". Quantum Science and Technology **4**, 013001 (2018).

[49] Nora Reinić. "Tree tensor networks for quantum many-body systems at finite temperature". Master's thesis. University of Zagreb. (2022). url: `https://urn.nsk.hr/urn:nbn:hr:217:027919`.

[50] L. Arceci, P. Silvi, and S. Montangero. "Entanglement of formation of mixed many-body quantum states via tree tensor operators". Phys. Rev. Lett. **128**, 040501 (2022).

[51] Mari Carmen Bañuls. "Tensor network algorithms: A route map". Annual Review of Condensed Matter Physics **14**, 173–191 (2023).

[52] Kouichi Okunishi, Tomotoshi Nishino, and Hiroshi Ueda. "Developments in the tensor network — from statistical mechanics to quantum entanglement". Journal of the Physical Society of Japan **91**, 062001 (2022).

[53] Abigail McClain Gomez, Susanne F. Yelin, and Khadijeh Najafi. "Reconstructing quantum states using basis-enhanced Born machines" (2022). url: `http://arxiv.org/abs/2206.01273`.

[54] Frances Fengyi Yang, Michele Sasdelli, and Tat-Jun Chin. "Training multilayer perceptrons by sampling with quantum annealers" (2023). url: `http://arxiv.org/abs/2303.12352`.

[55] Anatoly Dymarsky and Kirill Pavlenko. "Tensor network to learn the wave function of data". Physical Review Research **4**, 043111 (2022).

[56] Guglielmo Lami and Mario Collura. "Quantum magic via perfect sampling of matrix product states" (2023). url: `https://doi.org/10.48550/arXiv.2303.05536`.

[57] Simone Notarnicola, Andreas Elben, Thierry Lahaye, Antoine Browaeys, Simone Montangero, and Benoît Vermersch. "A randomized measurement toolbox for an interacting Rydberg-atom quantum simulator". New Journal of Physics **25**, 103006 (2023).

[58] Dominik S. Wild, Dries Sels, Hannes Pichler, Cristian Zanoci, and Mikhail D. Lukin. "Quantum sampling algorithms, phase transitions, and computational complexity". Phys. Rev. A **104**, 032602 (2021).

[59] Zhao-Yu Han, Jun Wang, Heng Fan, Lei Wang, and Pan Zhang. "Unsupervised generative modeling using matrix product states". Phys. Rev. X **8**, 031012 (2018).

[60] Sergey Bravyi, David Gosset, and Yinchen Liu. "How to simulate quantum measurement without computing marginals". Phys. Rev. Lett. **128**, 220503 (2022).

[61] Michael P. Zaletel and Frank Pollmann. "Isometric tensor network states in two dimensions". Phys. Rev. Lett. **124**, 037201 (2020).

[62] Harald Niederreiter. "7. random numbers and pseudorandom numbers". Pages 161–176. SIAM. (1992).

[63] Luc Devroye. "Chapter 4 nonuniform random variate generation". In Shane G. Henderson and Barry L. Nelson, editors, Simulation. Volume 13 of Handbooks in Operations Research and Management Science, pages 83–121. Elsevier (2006).

[64] Geof H. Givens and Jennifer A. Hoeting. "Simulation and Monte Carlo Integration". Chapter 6, pages 151–199. John Wiley & Sons, Ltd. (2012).

[65] Marco Ballarin, Giovanni Cataldi, Aurora Costantini, Daniel Jaschke, Giuseppe Magnifico, Simone Montangero, Simone Notarnicola, Alice Pagano, Luka Pavesic, Marco Rigobello, Nora Reinić, Simone Scarlatella, and Pietro Silvi. "Quantum tea: qtealeaves". Zenodo (2024).

[66] Hans Fischer. "A history of the central limit theorem: from classical to modern probability theory". Volume 4. Springer. (2011).

[67] Michael A. Nielsen and Isaac L. Chuang. "Quantum Computation and Quantum Information". Cambridge Univ. Press. Cambridge, United Kingdom (2007). 9. print. edition.

[68] Andrew J. Ferris and Guifre Vidal. "Perfect sampling with unitary tensor networks". Phys. Rev. B **85**, 165146 (2012).

[69] Lov K. Grover. "A fast quantum mechanical algorithm for database search". In Proceedings of the twenty-eighth annual ACM symposium on Theory of computing. Pages 212–219. (1996).

[70] P.W. Shor. "Algorithms for quantum computation: discrete logarithms and factoring". In Proceedings 35th Annual Symposium on Foundations of Computer Science. Pages 124–134. (1994).

[71] Phuc Q. Le, Fangyan Dong, and Kaoru Hirota. "A flexible representation of quantum images for polynomial preparation, image compression, and processing operations". Quantum Information Processing **10**, 63–84 (2011).

[72] Mercy G. Amankwah, Daan Camps, E. Wes Bethel, Roel Van Beeumen, and Talita Perciano. "Quantum pixel representations and compression for n-dimensional images". Scientific reports **12**, 7712 (2022).

[73] Subir Sachdev. "Quantum Phase Transitions". Cambridge University Press. Cambridge, United Kingdom (2011). 2nd edition.

[74] Jutho Haegeman, Christian Lubich, Ivan Oseledets, Bart Vandereycken, and Frank Verstraete. "Unifying time evolution and optimization with matrix product states". Phys. Rev. B **94**, 165116 (2016).

[75] Mark Howard, Joel Wallman, Victor Veitch, and Joseph Emerson. "Contextuality supplies the 'magic' for quantum computation". Nature **510**, 351–355 (2014).

[76] James R. Seddon, Bartosz Regula, Hakop Pashayan, Yingkai Ouyang, and Earl T. Campbell. "Quantifying quantum speedups: Improved classical simulation from tighter magic monotones". PRX Quantum **2**, 010345 (2021).

[77] Lorenzo Leone, Salvatore F. E. Oliviero, and Alioscia Hamma. "Stabilizer rényi entropy". Phys. Rev. Lett. **128**, 050402 (2022).

[78] Poetri Sonya Tarabunga, Emanuele Tirrito, Titas Chanda, and Marcello Dalmonte. "Many-body magic via Pauli-Markov chains—from criticality to gauge theories". PRX Quantum **4**, 040317 (2023).

[79] Johnnie Gray. "quimb: a python library for quantum information and many-body calculations". Journal of Open Source Software **3**, 819 (2018).

[80] Marco Ballarin, Pietro Silvi, Simone Montangero, and Daniel Jaschke. "Figures and supplemental material for "Optimal sampling of tensor networks targeting wave function's fast decaying tails"" (2024). Supplemental materials contain the figures of the manuscript, additional figures and animations, data for the figures.

[81] Marco Ballarin, Pietro Silvi, Simone Montangero, and Daniel Jaschke. "Simulation scripts for "Optimal sampling of tensor networks targeting wave function's fast decaying tail"" (2024).

[82] Marco Ballarin, Pietro Silvi, Simone Montangero, and Daniel Jaschke. "Input states for "Optimal sampling of tensor networks targeting wave function's fast decaying tails"". Zenodo (2024).

[83] E. M. Stoudenmire and Steven R. White. "Minimally entangled typical thermal state algorithms". New Journal of Physics **12**, 055026 (2010).

[84] Rohit Chandra, Leo Dagum, David Kohr, Ramesh Menon, Dror Maydan, and Jeff McDonald. "Parallel programming in openmp". Morgan Kaufmann. (2001).

Supplemental material

The appendices focus on providing additional information which is necessary to implement the OPES algorithm efficiently. Section A extends the pedagogical, simplified algorithm to measuring multiple shots in superiterations, while Sec. B explains the tensor network aspects and how to cache information for the tensor network.

A Implementation of the algorithm

Section 3 contains already one version of the algorithm; now, we extend the description to cover optional features like multiple shots, superiterations, or cache access. The algorithm takes as input the number of random numbers to be sampled n_r, the tensor network, and the set $\Delta = [0,1] \setminus \overline{\Delta} = \{\delta_1, \ldots, \delta_m\}$ of m known intervals δ_α. A superiteration of the algorithm has the following steps:

1. Set the state α^* to the empty bitstring, i.e., meaning no measurement has taken place so far. Therefore, the interval is $\Delta = [0,1]$ and set the initial left boundary of your probability interval $b^l_{\alpha^*}$ and the state probability p_{α^*}:
$$b^l_{\alpha^*} = 0, \quad p_{\alpha^*} = 1\,. \tag{40}$$

2. Sample n_r uniform random variables $\{u_j\}_{j=1,\ldots,n_r}$ from the interval Δ. Sort the random numbers $\{u_j\}_{j=1,\ldots,n_s}$ in ascending order. This step ensures that the cached memory is optimally managed.

3. Pick the next random number u_j. If the cache is not empty, find the longest possible bitstring matching u_j and set α^* to the longest matching bitstring; at least for the first qubit, there must be a match amongst the known intervals. Then, check
   - if the cache is empty, continue with the next step.
   - if the α^* contains already all qubits, continue with this step and u_{j+1}, i.e., $u_j \in \overline{\Delta}$.
   - else: load the corresponding tensors from the cache and continue directly with step *5*.

4. Compute the probability $p_0 = p(q_0 = |0\rangle)$ of measuring $|0\rangle$ on site 0 and project it on a given state according to its probability and the random number u_j:
$$q_0 = \begin{cases} |0\rangle & \text{if } u_j < p_0, \\ |1\rangle & \text{if } u_j \geq p_0, \end{cases} \tag{41}$$
$$|\alpha\rangle = |\alpha^*, q_0\rangle\ . \tag{42}$$
Update the left boundary and the state probability according to the measure:
$$b^l_\alpha = \begin{cases} b^l_{\alpha^*} & \text{if } u_j < p_0, \\ b^l_{\alpha^*} + p_0 & \text{if } u_j \geq p_0 \end{cases} \tag{43}$$
$$p_\alpha = p_{\alpha^*} \cdot p(q_0). \tag{44}$$

5. Moving to the next qubit i, set $\alpha^* = \alpha$; α^* is now the bitstring for the qubits up to the $(i-1)^{\text{th}}$ qubit. Compute the conditional probability $p_{0|\alpha^*} = p(q_i = |0\rangle\,|\alpha^*)$. Select the measured state according to u_j as in the previous point and set b^l_α, p_α accordingly, i.e.:
$$q_i = \begin{cases} |0\rangle & \text{if } u_j < b^l_{\alpha^*} + p_{0|\alpha^*}, \\ |1\rangle & \text{if } u_j \geq b^l_{\alpha^*} + p_{0|\alpha^*}, \end{cases} \tag{45}$$
$$|\alpha\rangle = |\alpha^*, q_i\rangle\ , \tag{46}$$
$$b^l_\alpha = \begin{cases} b^l_{\alpha^*} & \text{if } u_j < b^l_{\alpha^*} + p_{0|\alpha^*}\ , \\ b^l_{\alpha^*} + p_{0|\alpha^*} p_{\alpha^*} & \text{if } u_j \geq b^l_{\alpha^*} + p_{0|\alpha^*}\ , \end{cases} \tag{47}$$
$$p_\alpha = p_{\alpha^*} \cdot p(q_i|\alpha^*). \tag{48}$$

6. Repeat step *5* until all the sites of the system are measured. Further details for this step are explained in Sec. 3 and remain valid here.

7. Remove the measured probability interval from our sampling interval, i.e., $\Delta = \Delta \setminus \delta_\alpha$. Each new sampled number $u_j \sim U(\Delta)$ explores a different probability interval $\delta_{\beta \neq \alpha}$. Update the cache with the corresponding information and tensors.

8. Continue with step *3* if there are random numbers left for this superiteration.

9. If the number of superiterations or the desired coverage has been reached, exit. If the next superiteration is executed with the current cache, continue with step *2*. For the next superiteration with an empty cache, restart with step *1*.

A.1 Uniformly sampling from Δ

In this section, we discuss how the n_r uniform random variables $\{u_j\}_{j=1,\dots,n_r}$ from the interval Δ are sampled. First, we define Δ_i the M continuous intervals componing Δ, i.e. $\Delta = \bigcup_{i=1}^{M} \Delta_i$. We define the probability related to the interval Δ_i as p_i. The technique to sample u_j is as follows:

1. Determine how many samples n_r^i are sampled from each interval Δ_i, with $\sum_{i=1}^{M} n_r^i = n_r$. This task can be easily done by sampling n_r numbers uniformly in $U(0,1)$, and identify the sample $u_k \sim U(0,1)$ as belonging to Δ_i if:

$$\widetilde{p}_{i-1} \leq u_k < \widetilde{p}_i, \tag{49}$$

with $\widetilde{p}_0 = 0$ and $\widetilde{p}_i = p_i / \sum_{i=1}^{M} p_i$.

2. Sample n_r^i random numbers from the interval Δ_i:

$$\{u_j\}_{j=1,\dots,n_r} = \bigcup_{i=1}^{M} \{u_{j'} \sim U(\Delta_i)\}_{j'=1,\dots,n_r^i} \tag{50}$$

$$= \bigcup_{j=1}^{n_r} \{u_j \sim U(\Delta)\} \tag{51}$$

The resulting $\{u_j\}_{j=1,\dots,n_r}$ are uniformly distributed over Δ. We stress that the computational complexity C_Δ of sampling n_r random numbers from a discontinuous interval Δ with M continuous intervals Δ_i scales as:

$$C_\Delta = O(n_r M), \tag{52}$$

since we just need to generate $2n_r$ random numbers, but we have to check up to M conditions on n_r of them.

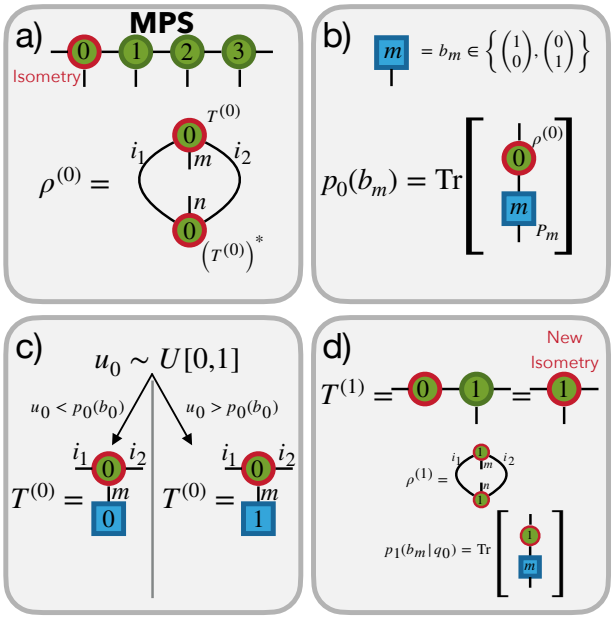


Figure 10: *Algorithm to sample from an n-sites MPS of local dimension $d = 2$ and bond dimension χ with complexity $O(n\chi^2 d^2)$.* Tensors are represented as circles and squares, connected legs are summed over. a) Computation of the reduced density matrix $\rho^{(0)}$ using the isometry center of the tensor network. b) Computation of the probability of measuring the basis states $b_0 = |0\rangle$ and $b_1 = |1\rangle$ using the projector $\{P_m = |b_m\rangle\langle b_m|\}_{m=0,1}$. c) Projection of $T^{(0)}$ on b_0 with probability $p_0(b_0)$ or on b_1 with probability $1 - p_0(b_0)$. d) Movement of the isometry center from $T^{(0)}$ to $T^{(1)}$, repeating the same computations of a) and b)

B Details of the tensor networks implementation

The fundamental requirement for the sampling is the ability to efficiently compute expectation values of local observables and the conditional probabilities of a given quantum state. Here, we review how these values are computed focusing on the standard sampling procedure, going afterward into the details of what we cache to optimize the process.

Suppose we want to sample from the n-body quantum state $|\psi\rangle$ represented as an isometric tensor network [61] with bond dimension χ. In Fig. 10, we report the procedure for an MPS with local dimension $d = 2$. In general, the steps (a) to (d) following Fig. 10 are the following:

(a) We set the isometry center to the left-most physical tensor $T^{(0)}$, in such a way that the contraction over the auxiliary legs leads to the single-site reduced density matrix ρ_0

with:

$$\rho^{(0)}_{mn} = \sum_{i_1,i_2\ldots i_n} T^{(0)}_{m,i_1,i_2\ldots i_n} \left(T^{(0)}_{n,i_1,i_2\ldots i_n}\right)^*, \tag{53}$$

where m, n represent the physical indexes of local dimension d, while the auxiliary indexes, i.e., indexes not representing the Hilbert space of site 0, are defined as $i_1, i_2 \ldots i_n$. The complex conjugate tensor is indicated by a superscript $*$.

(b) Defining the computational basis of a d-dimensional qudits as $\{b_m\}_{m=0,\ldots,d-1}$, we compute the probability of site 0 being in state b_m as:

$$p_0(b_m) = \langle\psi| P_i |\psi\rangle = \mathrm{Tr}\left[\rho^{(0)} P_m\right], \tag{54}$$

where $\{P_m = |b_m\rangle\langle b_m|\}_{m=0,\ldots,d-1}$ are the projectors over the computational basis.

(c) We randomly select a basis state b_m with probability $p_0(b_m)$ and project the physical tensor $T^{(0)}$ on that state:

$$T^{(0)}_{i_1,i_2\ldots i_n} = \sum_m b_m T^{(0)}_{m,i_1,i_2\ldots i_n}. \tag{55}$$

We call q_0 the measured basis state on site 0. The projection is done with a vector and not a projector for efficiency. Effectively, this step reduces the local Hilbert space to a one-dimensional Hilbert space and reduces the computational complexity.

(d) The isometry center is then moved to the next physical tensor T_1; by repeating the same procedure, we obtain the conditional probabilities of the second site:

$$p_1(b_m|q_0) = \mathrm{Tr}\left[\rho^{(1)} P_m\right], \tag{56}$$

and project into the basis state with these probabilities. Calling q_1 the measured basis state on site 1, we denote the probability as $p(q_1|q_0)$.

Iterating through the whole tensor network returns a single sample $\alpha = q_0, \ldots, q_{n-1}$ from the quantum state:

$$\begin{aligned} p_\alpha &= p_{q_0,\ldots,q_{n-1}} \\ &= p\left(q_{n-1}|q_{n-2}\ldots q_0\right)\ldots p(q_1|q_0)p(q_0). \end{aligned} \tag{57}$$

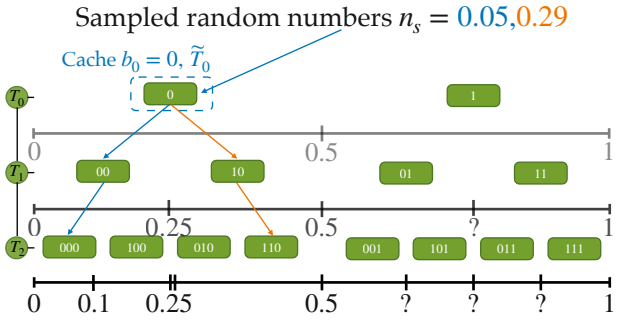


Figure 11: *Caching for matrix product states (MPS).* Example of OPES with $n = 3$ sites and local dimension $d = 2$. The intervals represent the conditional probabilities of the state represented above it, i.e., the probability $p_0(0) = 0.5$, $p_1(0|p_0 = 0) = 0.5$, and so on. First, we compute the probability associated with sample $s_0 = 0.05$ (blue line), obtaining the probability interval $\delta_0 = (0, 0.1)$ associated to the state $|000\rangle$. We cache the couples $(0, \widetilde{T}_0)$, $(00, \widetilde{T}_1)$, where with $\widetilde{T}_i$ we indicate the tensor after the projection with the measured basis state. When we continue with the sample $s_1 = 0.29$, we use the cached information and do not perform any operation on T_0. At the level of T_1, we go to the right instead of the left; thus, we can eliminate the cached information for T_1, T_2. We can do so because any further samples go to the right of the current interval, due to the ascending ordering.

Notice that following this procedure the state α is measured with probability p_α, thus this sampling method is unbiased, e.g., with respect to the order of qubits.

The entire procedure scales as $O(n\chi^2 d^2)$ for MPS [83], while as $O(n\chi^4)$ for TTNs. Since each sample is independent from the others the standard sampling can be trivially parallelized with OPENMP [84].

B.1 Caching the MPS intermediate measurements

One of the reasons for the speedup of OPES over the standard sampling is the caching of the intermediate states inside a superiteration. We suppose the left-to-right sampling ordering discussed in Sec. B, and that all $\{T_j\}_{0\le j<m}$, with $m > 0$, have been measured. To restart the computation from tensor T_m, it is sufficient to store the measured state $|b_0, b_1, \ldots, b_{m-1}\rangle$ and the tensor T_m contracted with tensor T_{m-1} after the projection on state $|b_{m-1}\rangle$. The caching in the superiter-

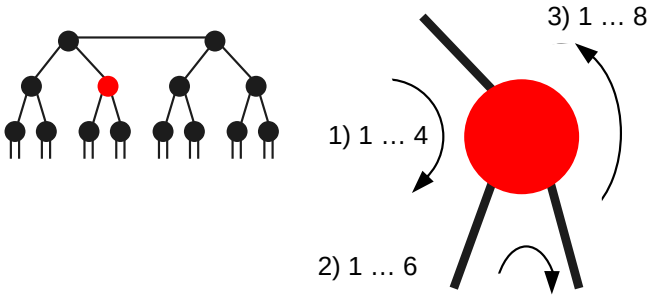


Figure 12: *Caching for tree tensor networks (TTN).* The caching for the TTN distinguishes which sites have been measured so far. Therein, an auxiliary tensor in the upper layers of the TTN can be traversed up to three times. The cache recognizes the direction based on the length of the state.

ation is optimal because, due to the ascending ordering of the samples n_s, we know that once the basis b_i of the tensor T_i pass from 0 to 1 we can delete all the cached information for $j < i$ since we completely explored that probability interval. The caching structure is shown in Fig. 11 for an MPS with $n = 3$ sites and $d = 2$ basis. Performance can be further optimized by keeping the cache between superiterations.

This procedure can be parallelized to using OPENMP [84], by assigning to different processors different "branches" of the tree, such as the blue and orange path from tensor T_1 in Fig. 11. Ensuring an equal workload for each process is more challenging for the OPES than for the standard sampling.

B.2 Caching the TTN intermediate measurements

The caching in the MPS is simplified due to the fact that the MPS is a chain and we respect the order of the MPS and treat one site after another; no tensor in the MPS is visited twice. The TTN does not fulfill this assumption and tensors can be visited up to three times during the projective measurements.

Figure 12 sketches the problem for a 16-site TTN and one path towards the solution. The tensors that are in the bulk, with respect to the layers and within a layer itself, are traversed three times with QR decompositions while running the projective measurements on the children. The tensor highlighted in red in Fig. 12 is one example. Therefore, the cache stores tensors for the same position of the TTN multiple times, distinguished by the length of the bitstring measured so far.