ARTICLE OPEN

Check for updates

Low-rank density-matrix evolution for noisy quantum circuits

Yi-Ting Chen[1,2✉], Collin Farquhar[1] and Robert M. Parrish[1]

In this work, we present an efficient rank-compression approach for the classical simulation of Kraus decoherence channels in noisy quantum circuits. The approximation is achieved through iterative compression of the density matrix based on its leading eigenbasis during each simulation step without the need to store, manipulate, or diagonalize the full matrix. We implement this algorithm using an in-house simulator and show that the low-rank algorithm speeds up simulations by more than two orders of magnitude over existing implementations of full-rank simulators, and with negligible error in the noise effect and final observables. Finally, we demonstrate the utility of the low-rank method as applied to representative problems of interest by using the algorithm to speed up noisy simulations of Grover's search algorithm and quantum chemistry solvers.

npj Quantum Information (2021)7:61 ; https://doi.org/10.1038/s41534-021-00392-4

INTRODUCTION

The scaling of quantum computers is limited by quantum decoherence[1–4]. State of the art quantum computers that consist of fewer than 100 qubits[5,6] are of interest for noisy intermediate-scale quantum (NISQ) applications, where qubits are used without error correction[7]. Therefore, classically simulating imperfect and noisy circuits are vital for the development and design of NISQ-era algorithms as well as characterizing errors in quantum hardware[8]. Existing classical simulators have typically focused on emulating noiseless circuits. In this case, simulations of quantum circuits with more than 50 qubits have been demonstrated[9–11]. There are several techniques for speeding up simulations of certain types of circuits[12–16] or algorithms[17–20]. High performance computations have been developed for simulations of noisy circuits[21–24], as well as lightweight open-source tools such as density-matrix simulators in Qiskit[25] and Cirq[26]. These simulators are based on evolving full density matrices (FDMs), which are prohibitively expensive for large numbers of qubits.

In an open quantum system, the decoherence can be modeled as interactions with a large environment[27]. One can determine the properties of an open quantum system with the Monte Carlo wave-function method where a system is decomposed into an ensemble of pure states that evolve individually and then are averaged[28–32]. On the other hand, the dynamics can also be characterized by the Lindblad equation[33,34], which well describes decoherence in various quantum hardware architectures[35–37]. To reduce the complexity of the Lindblad equation, one can project the quantum states onto a lower dimensional basis using filtering theory and simulate the states more efficiently with reasonable accuracy[38,39].

In this work, we combine the ideas of the pure state decomposition[28,29] and the low dimension basis projection[39] to efficiently simulate noisy quantum circuits. Compressed representations are commonly applied to classical simulation of quantum systems with high symmetry or low entanglement. Applications range from compressed sensing of quantum state tomography[40,41], limiting bond dimensions of tensor networks[14,42,43], and low-rank factorization of Hamiltonians[44] to efficient representations of states[15,45–47]. In our case, it is found that the von Neumann entropy of a density matrix is often small when the noise level is low, implying that it is possible to model the density matrix using a matrix of lower rank with minimal information loss. We achieve this by iteratively projecting onto a subspace of the eigenbasis, and evolving only a small ensemble of pure states.

In the following, we present a complete and explicit algorithm, which decomposes a mixed density matrix into a low-rank matrix akin to an ensemble of pure states, applies gate and Kraus operators to this low-rank matrix, and computes the output density matrix and probability distribution. The procedure involves iterative compression of the density matrix to maintain a numerically compact form with minimal error. As an example, Fig. 1a shows a six-qubit density matrix after a quantum circuit that solves a Grover's search problem for finding computational basis states with Hamming weight ≤2. The same circuit is simulated with depolarizing noise with noise strength $p \simeq 0.33\%$ by an exact method and by our low-rank method. In this example, we use a low-rank representation that has only 20% of the full rank. Figure 1b, c shows that the low-rank method simulates noise with high accuracy. More extensive benchmarking and detailed descriptions on the performance and accuracy of the method are in "Implementation and benchmarking."

In fact, we show that it is possible to evolve and compute quantities of interest (e.g., expectation values or qubit probabilities) of a $(2^N \times 2^N)$ density matrix without ever forming a matrix of size $(2^N \times 2^N)$, where N is the number of qubits. This reduction in the required memory may allow for the studying of systems which would otherwise be prohibitively large.

The algorithm is then assessed by a sequence of random benchmarking under various types and strengths of noise channels to test its practical speedup and error. We show that the algorithm performs consistently in random circuits and in structured circuits for quantum algorithms such as Grover's search, with a speedup of more than two orders of magnitude and with a small error (around 0.01%) in the probability distribution associated with the final output density matrix. Furthermore, as N becomes larger, and approaches the range for which classical simulations become difficult, the advantage of this algorithm continues to increase over the standard method of FDM evolution.

RESULTS

We present an algorithm that simulates noisy circuits using a low-rank representation of density matrices. The algorithm consists of two parts, low-rank evolution and eigenvalue truncation, which are covered in "Low-rank evolution" and "Eigenvalue

[1]QC Ware Corp., Palo Alto, CA, USA. [2]Department of Applied Physics, Stanford University, Stanford, CA, USA. ✉email: yitchen@stanford.edu

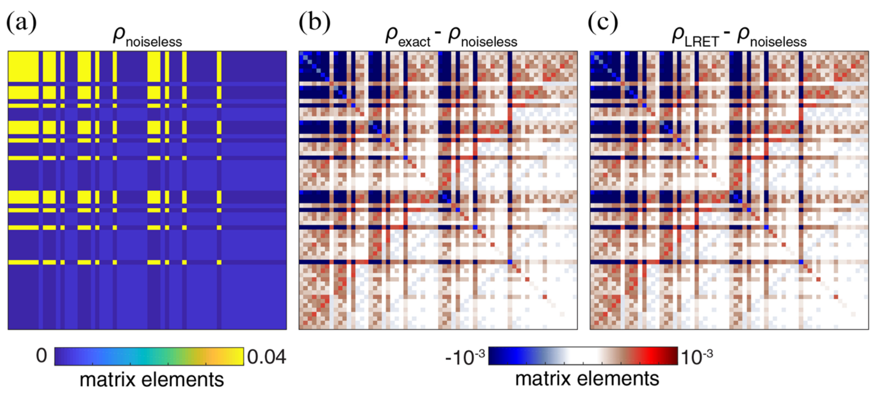

Fig. 1 An example of the low-rank method simulating noise with negligible error. a Density matrix from noiseless simulation, $\rho_{\text{noiseless}}$. **b** Difference of density matrices from an exact noise simulation and from a noiseless simulation, $\rho_{\text{exact}} - \rho_{\text{noiseless}}$. **c** Difference of density matrices from a low-rank noise simulation and from a noiseless simulation, $\rho_{\text{LRET}} - \rho_{\text{noiseless}}$ where LRET, the focus of this article, is a low-rank method. The rank of ρ_{LRET} is 20% of that of ρ_{exact}, and corresponds to 2.2% of distortion (defined in "Implementation and benchmarking"). The noise strength is $p \simeq 0.33\%$ and the truncation threshold is $\epsilon = 3 \times 10^{-4}$.

truncation" below. In "Kraus operator decomposition", an iterative procedure consisting of these two parts is introduced. Then, in "The LRET algorithm" we explain how to sample the associated probability distribution without explicitly forming the FDM.

Low-rank evolution

A coherent quantum system can be represented by either a state vector $|\psi\rangle$ or a density matrix $\rho = |\psi\rangle\langle\psi|$. $|\psi\rangle$ has dimensions $(2^N \times 1)$, while ρ has dimensions $(2^N \times 2^N)$. Because of the substantial difference in the sizes of $|\psi\rangle$ and ρ, most classical simulations of coherent quantum systems work directly with the state vector representation. Unfortunately, the state vector representation does not directly allow for the presence of decoherence. Instead, the evolution of a quantum system in a decoherent noise channel can only be described by a density matrix ρ, which is generally more computationally demanding to simulate. The evolution of a quantum state in a noisy quantum circuit is described by refs. [48–50]

$$\rho^{(d+1)} = \sum_{a} p_a K_a \rho^{(d)} K_a^{\dagger} \tag{1}$$

where K_a are Kraus matrices, p_a is the corresponding probability, and $\rho^{(d)}$ and $\rho^{(d+1)}$ are density matrices before and after a noise channel. This Kraus-based noise model is capable of encapsulating many different types of decoherence channels such as bit flip, depolarizing, etc., and therefore is an extremely useful tool in modeling the operation of NISQ-era quantum algorithms in the presence of decoherence noise on real devices. However, most current implementations of this Kraus-based decoherence model explicitly work with a $(2^N \times 2^N)$ density matrix. Our approach exploits the fact that for realistically small noise levels ($p \simeq 0.01$), the von Neumann entropy of the density matrix usually remains low, raising the possibility of working with an approximate, but accurate, low-rank representation of the density matrix. This observation is supported by an entropic analysis in the supplementary information.

While the possibility of a low-entropy density-matrix evolution is tantalizing, there remains to be resolved many pragmatic details about how to efficiently identify and exploit this rank structure while avoiding formation and manipulation of any density-matrix-sized quantities in the rank identification process. Here and in "Eigenvalue truncation" we describe an algorithm that can accomplish this for density matrices that exhibit the desired low-rank structure. A density matrix, ρ, can always be decomposed as an outer product of the $L \in \mathbb{C}^{2^N \times V}$ matrix,

$$\rho \equiv LL^{\dagger}, \tag{2}$$

for some $V \in \mathbb{N}$. While the choice of L is not unique, in general, it is possible to find L with V equal to the rank of the density matrix using decomposition methods such as singular value decomposition. For density matrices with rank smaller than 2^N, this form most compactly represents the state with the minimal column dimension. Using the decomposition in Eq. (2), we can evolve the density matrix by updating L without evaluating $\rho^{(d)}$ explicitly as in Eq. (1). For a gate operation

$$\begin{aligned}\rho^{(d+1)} &= \mathcal{G}^{(d)}\rho^{(d)} = G^{(d)}\rho^{(d)}G^{(d)\dagger}\\ &= G^{(d)}L^{(d)}L^{(d)\dagger}G^{(d)\dagger}\\ &= L^{(d+1)}L^{(d+1)\dagger}\end{aligned} \tag{3}$$

where $L^{(d+1)} \equiv G^{(d)}L^{(d)}$, $\mathcal{G}^{(d)}$ is a gate operation, and $G^{(d)}$ is its corresponding gate matrix. Likewise, for a Kraus operator,

$$\begin{aligned}\rho^{(d+1)} &= \mathcal{K}^{(d)}\rho^{(d)} = \sum_{a=1}^{A} p_a K_a^{(d)}\rho^{(d)}K_a^{(d)\dagger}\\ &= \sum_{a=1}^{A} p_a K_a^{(d)}L^{(d)}L^{(d)\dagger}K_a^{(d)\dagger}\\ &= \sum_{a=1}^{A} J_a^{(d+1)}J_a^{(d+1)\dagger} = L^{(d+1)}L^{(d+1)\dagger}\end{aligned} \tag{4}$$

where $J_a^{(d+1)} \equiv \sqrt{p_a}K_a^{(d)}L^{(d)}$, $\mathcal{K}^{(d)}$ is a Kraus operator, and $K_a^{(d)}$ and p_a are its corresponding Kraus matrices and probability factor, and A is the number of Kraus matrices in the operation. $L^{(d+1)}$ is formed by concatenating $J_a^{(d+1)}$ as columns, i.e., $L^{(d+1)} \equiv [J_1^{(d+1)}, J_2^{(d+1)}, \ldots, J_A^{(d+1)}]$. Note that, due to the concatenation, the number of columns of L changes after each noise operation; each column in $L^{(d)}$ will evolve to A columns in $L^{(d+1)}$. For example, if the dimension of $L^{(d)}$ is $2^N \times 3$ and $A = 2$, then $L^{(d+1)}$ has dimension $2^N \times 6$. The computational complexity of updating L is $\mathcal{O}(V2^N)$.

Eigenvalue truncation

The number of J_a vectors in layer d grows to A times the number of J_a vectors in layer $d-1$. For a system starting from a pure state, this number is A^d at the dth layer. This scaling makes tracking all J_a vectors computationally intractable over time if left unchecked. Furthermore, in practice, when the noise level is small, the number of columns corresponding to significant eigenvalues of $L^{(d)}$ is often found to grow only polynomially with the system size. An eigenvalue truncation procedure is used to project the density matrix into a lower-rank representation that retains only the highest contributing columns, akin to quantum filtering in simulating open quantum systems[38]. We truncate the eigenvectors whose

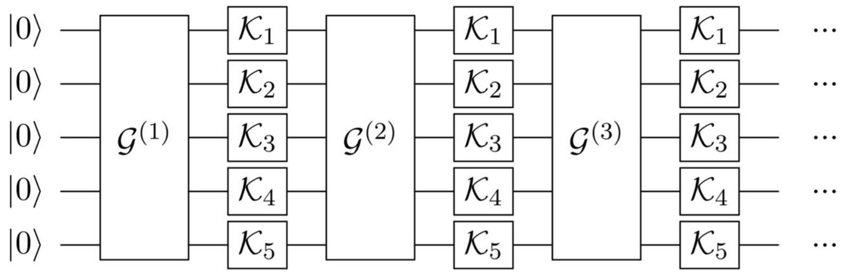

Fig. 2 Schematic of a noisy quantum circuit. $\mathcal{G}^{(i)}$ represents gate operations at the ith layer and $\mathcal{K}$ represents a one-qubit Kraus operator that models the noise.

eigenvalues are negligible by

$$\rho^{(d)} = U^{(d)}\Lambda^{(d)}U^{(d)\dagger} \simeq \tilde{U}^{(d)}\tilde{\Lambda}^{(d)}\tilde{U}^{(d)\dagger} \tag{5}$$

where $U^{(d)}$, $\Lambda^{(d)}$ are eigenvectors and eigenvalues of $\rho^{(d)}$, and from which we define $\tilde{U}^{(d)}$, $\tilde{\Lambda}^{(d)}$ as approximations for which the unimportant eigenvalues and eigenvectors are truncated. The truncation is based on a threshold ϵ. The descending-ordered eigenvalues are picked up one by one until they sum to $1-\epsilon$. The remaining eigenvalues sum to ϵ are thrown away along with their associated eigenvectors. Although more sophisticated ways of truncation exist[51,52], we use this simple cutoff criteria to better control the error introduced by the procedure. Furthermore, this truncation method is the optimal scheme to preserve the trace and the 2-norm of a matrix, known as the Eckart–Young theorem[53].

The representation above is a useful method to retain only the most important eigenvectors in Kraus noise models. However, the approach appears to have the computational problem that it involves the eigendecomposition of a $(2^N \times 2^N)$ matrix. Solving an eigenvalue problem of a $(2^N \times 2^N)$ matrix has a complexity of $\mathcal{O}((2^N)^3)$, which is very expensive and would overwhelm the benefit of low-rank simulation. However, using the result from theorem 1 in the Supplementary information, we can efficiently compute the eigenvectors and eigenvalues without explicitly constructing the density matrix. The complexity of the eigenvalue problem is instead $\mathcal{O}((AV)^2 2^N)$ where $V < 2^N$ is the number of columns of L in Eq. (2), and A is the number of Kraus matrices comprising the Kraus operator. Although the the complexity still contains 2^N, it is an improvement by an exponential factor over $\mathcal{O}((2^N)^3)$.

Kraus operator decomposition

We model noisy quantum channels with single-qubit Kraus operators. To model noise induced by gate operations, one may apply Kraus operators following each gate. If the circuit is sparse in terms of gates, correspondingly, there are only a few Kraus operators per layer. In this case, AV stays small and eigenvalue truncation can be done relatively efficiently. However, consider a dense noisy circuit with a depolarizing Kraus operator acting on every qubit at each time-step (Fig. 2). The depolarizing Kraus operator is comprised of four matrices[50], and therefore $A = 4^N$, making eigenvalue truncation intractable with complexity $\mathcal{O}((4^N V)^2 2^N)$. This can be resolved by decomposing the Kraus operator in several groups

$$\mathcal{K}^{(d)} = \prod_{\beta=1}^{B} \mathcal{K}_\beta^{(d)}. \tag{6}$$

This decomposition is possible because noise channels in a quantum computer can be well-described by a combination of one and two-qubit Kraus operators[5]. In the example in Fig. 2, instead of an eigenvalue truncation after each whole Kraus layer, a truncation is applied after each $\mathcal{K}_\beta^{(d)}$ in order to prevent A from getting too large. In other words, the evolution $\rho^{(d+1)} \to \mathcal{K}^{(d)}\rho^{(d)}$ is approximated by

$$\rho^{(d+1)} = \mathcal{E}\mathcal{K}_B^{(d)}\mathcal{E}\mathcal{K}_{B-1}^{(d)}\cdots\mathcal{E}\mathcal{K}_2^{(d)}\mathcal{E}\mathcal{K}_1^{(d)}\rho^{(d)} \tag{7}$$

where $\mathcal{E}$ is eigenvalue truncation operation. Because a Kraus operator is decomposed into B operations, each decomposed operation has only $\bar{A} = 4^{\frac{N}{B}}$ Kraus matrices, denoted as $K_{\beta,\alpha}^{(d)}$ with subscript α running from 1 to $\bar{A}$. Each truncation has complexity $\mathcal{O}((4^{\frac{N}{B}}V)^2 2^N)$ and there are B truncation steps, so the total complexity is $\mathcal{O}(B(4^{\frac{N}{B}}V)^2 2^N)$. We choose B such that $M = \frac{N}{B}$ is constant and the complexity becomes $\mathcal{O}(N\frac{(4^M V)^2}{M} 2^N)$. We define the "intermediate rank" V_I as

$$V_I^2 \equiv N\frac{(4^M V)^2}{M}. \tag{8}$$

This quantity will be important to estimate the conditions for which low-rank simulation is faster than FDM simulation in "Implementation and benchmarking".

We focus on the simulation of NISQ-era circuits, which are shallow in terms of circuit depth. However, note that for very deep circuits, V_I can grow larger than 2^N. In this case, there is no benefit of doing low-rank evolution, and we switch back to FDM evolution. The algorithm for low-rank noise simulation is summarized in Algorithm 1.

The LRET algorithm

"Low-rank evolution", "Eigenvalue truncation", and "Kraus operator decomposition" together describe the algorithm for getting the final low-rank representation, $L^{(D)}$. We refer to this algorithm as low-rank simulation with eigenvalue truncation (LRET). The time complexity of LRET is $\mathcal{O}(V^2 2^N)$, while the complexity of a FDM simulation is $\mathcal{O}(2^{2N})$. The LRET algorithm provides an improvement by an exponential factor over the FDM method. The space complexity of LRET is $\mathcal{O}(V2^N)$, while the FDM method has $\mathcal{O}(2^{2N})$. The reduced memory requirement may allow for simulations of larger circuits than would be possible otherwise. The full procedure of LRET is summarized in Algorithm 1. The concatenation and the eigenvalue truncation in the algorithm are described in "Low-rank evolution" and "Eigenvalue truncation", respectively. Note that although we use the $|000\ldots\rangle$ state as the initial state (as shown in Algorithm 1) throughout this article, the algorithm works with any fiducial state. Also note that the density matrix, probability distribution, and expectation value in Algorithm 1 are optional and are included as examples of user-specified outputs.

Once we have the final low-rank representation, $L^{(D)}$, we can construct the density matrix using Eq. (2). However, this FDM quantity is rarely needed in standard practice. For example, one may want to simulate the behavior of quantum hardware where the only information we obtain is from measurements in a fixed computational basis, which sample the probability mass function, Prob(x), that is defined by the underlying density matrix. In this case, low-rank simulation gains an additional speedup as the probability distribution is simply

$$\mathrm{Prob}\,(x) = \sum_{v=1}^{V} L_{x,v}^{(D)} L_{v,x}^{(D)\dagger}. \tag{9}$$

where subscript x and v run over the computational basis dimension and column dimension of the matrix, respectively. The measurement count for each state is then sampled from this distribution. Note that, if the goal of a circuit simulation is to observe and count the measurement outputs, a density matrix is not formed at any point of the simulation as long as the intermediate rank is smaller than 2^N. Similarly, we can evaluate observables in low-rank form using $O = \mathrm{Tr}(\rho\mathcal{O}) = \mathrm{Tr}(\mathrm{LL}^\dagger\mathcal{O}) = \mathrm{Tr}(\mathrm{L}^\dagger\mathcal{O}\mathrm{L})$ where O is the expectation value of an observable $\mathcal{O}$.

```
Algorithm 1 Low-rank simulation with eigenvalue truncation
  $L^{(0)} = [1, 0, 0,...,0]$
  for $d \leftarrow 1$ to $D$ do
  $L^{(d)} \leftarrow G^{(d)}L^{(d-1)}$
  for $\beta \leftarrow 1$ to $B$ do
  $L^{(d)} \leftarrow \mathrm{Concatenate}_\alpha(\sqrt{p_\alpha^{(d)}}K_{\beta,\alpha}^{(d)}L^{(d)})$
  $L^{(d)} \leftarrow \mathrm{Eigenvalue\ Truncation}_\epsilon(L^{(d)})$
  end for
  end for
  Compute quantities of user's choice:
  Density Matrix $\rho^{(D)} \leftarrow L^{(D)}L^{(D)\dagger}$
  Prob (x) $\leftarrow \sum_v \mathrm{L}_{x,v}^{(D)}\mathrm{L}_{v,x}^{(D)\dagger}$
  Expectation $\leftarrow \mathrm{Tr}(\mathrm{L}^{(D)\dagger}\mathcal{O}\mathrm{L}^{(D)})$
```

Implementation and benchmarking

In the following sections, we discuss the performance of the algorithm for low-rank noise simulation from our implementations using an in-house quantum circuit simulator built in Python. In our simulator, one can specify one of two options for a noisy simulation: FDM simulation, and LRET as described in Algorithm 1. We benchmark the two simulation methods in three scenarios: randomized benchmarking, state preparation for quantum chemistry, and Grover's search algorithm. For time benchmarking, we use Cirq 0.5.0, a widely used open-source FDM simulator, to show that our implementation of the FDM method is reasonably optimized and serves as a good baseline for comparison. All of the benchmarking was executed on an AWS c5.12xlarge instance.

The general result is that the LRET method is two orders of magnitude faster than the FDM method with a trade-off of ~0.01% error. The error is measured by the variational distance[54] between the output density matrices from the LRET method (ρ_{LRET}) and from an exact method (ρ_{exact}), such as FDM. Because this quantity depends on the noise level, we define a more appropriate measure for error benchmarking

$$\mathrm{Distortion} \equiv \frac{\mathrm{T}(\rho_{\mathrm{LRET}}, \rho_{\mathrm{exact}})}{\mathrm{T}(\rho_{\mathrm{exact}}, \rho_{\mathrm{noiseless}})} \quad (10)$$

where $\rho_{\mathrm{noiseless}}$ is the density matrix from the simulation of the

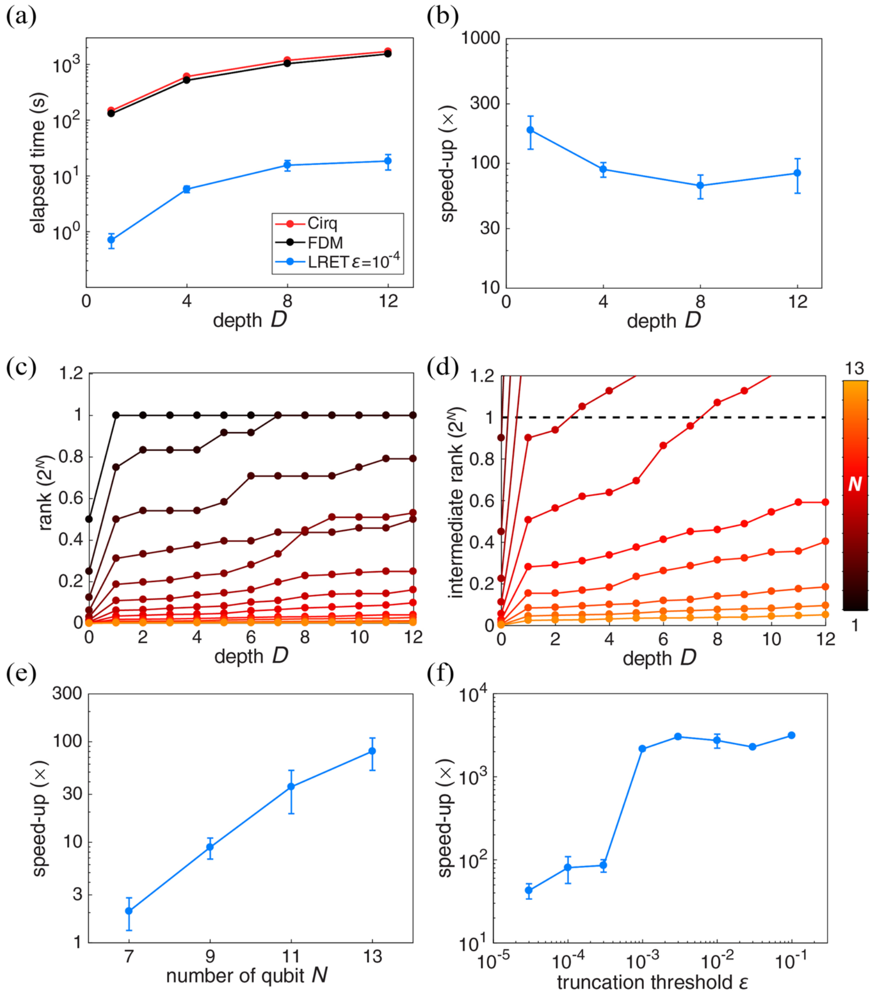

Fig. 3 Time benchmarking under depolarizing noise. a Averaged elapsed time for quantum circuits with $N = 13$ qubits using different simulators. Error bar is the standard deviation from several random circuits. **b** Speedup, defined by the ratio of the elapsed time between FDM and LRET, for different circuit depths. **c** Rank evolution for different size of quantum circuit. **d** Intermediate rank for different size of quantum circuit. **e** Speedup for different number of qubit ($D = 13$). **f** Speedup for different truncation threshold ($N = 13$, $D = 13$). Parameters are $p = 0.1\%$ and $\epsilon = 10^{-4}$. The error bars are standard-deviation-calculated across a set of simulations.

same circuit without noise, and T is the variational distance between the probability distributions defined by the two density matrices in the computational basis, i.e., $T(\rho^A, \rho^B) = \sum_i |\rho^A_{ii} - \rho^B_{ii}|$. To aid in a qualitative understating of the distortion measure, it is useful to note that the distance between ρ_{LRET} and ρ_{exact} captures the error or information loss incurred by the eigenvalue truncation procedure. The denominator scales this value relative to the change induced by the noise channel to $\rho_{\mathrm{noiseless}}$. For example, when the output error is 0.01% and the change induced by noise is 0.1%, the distortion is 10%.

Randomized benchmarking: depolarizing noise

Randomized benchmarking is a standard tool used to evaluate the performance of quantum hardware[55–57]. We use the idea to benchmark time and error metrics for the two simulation methods on an ensemble of randomly generated circuits. The circuits are generated from random choices of common gates, including X, Y, Z, S, T, RX, RY, RZ, SWAP, CZ, and CNOT. The "Randomized benchmarking: circuit sparsity and connectivity" below discusses the different types of circuits we use for benchmarking. If not explicitly stated otherwise, the random circuits are dense circuits where one- and two-qubit gates appear with equal probability, and the two-qubit gates connect to adjacent qubits. Dense circuits are those in which a gate acts on each qubit at each time-step. Inspired by the fact that noise is well-described by a set of Kraus operators whose dimension is independent of the circuit size[5], all Kraus operators act on single qubits in the benchmarking, as illustrated in Fig. 2.

To understand the conditions for which the LRET method gains a speedup against the FDM method, we inspect the rank evolution of density matrices evolved using LRET. The rank and the intermediate rank (as defined in Eq. (8)) directly influence the computational complexity and the speed of the LRET algorithm. In the following sections, we first benchmark the time, error, and rank for simulations under different noise channels. Then, we assess performance on a variety of differently characterized random circuits.

In this section, we benchmark dense circuits with the depolarizing noise channel, which is defined as $\rho \to (1-p)\rho + \frac{p}{3}X\rho X + \frac{p}{3}Y\rho Y + \frac{p}{3}Z\rho Z$. It has been shown that noise in quantum hardware, in average, behaves like depolarizing noise[58,59], making it a good description of realistic noise channels.

Figure 3a shows that, while our FDM method and Cirq take a similar amount of time to run for 13-qubit circuits with $p = 0.1\%$ under depolarizing noise, the LRET method is much faster than both. In shallow circuits, LRET is 200× faster than FDM (Fig. 3b). Even for higher depth circuits, LRET remains roughly 100× faster. This can be understood by considering the size of the numerical representation these methods are keeping track of. While the FDM method evolves a $2^N \times 2^N$ density matrix, LRET only keeps track of a $2^N \times V$ representation of a density matrix. Effective use of the LRET algorithm amounts to choosing the truncation threshold as to best manage the trade-off between the speed of the simulation and the error in the simulation results. This trade-off is characterized in Fig. 4.

Although V is always smaller than 2^N at low depth for $N > 2$ (Fig. 3c), the conditions for a speedup is determined by the intermediate rank V_I, defined in Eq. (8); the LRET method is faster if $V_I < 2^N$. As shown in Fig. 3d, e, there is a significant speedup only when $N > 7$ for the circuit depth consider herein. Since V_I increases approximately polynomially and 2^N increases exponentially in N, the range of depths for which LRET has an advantage will increase even more as the number of qubits increases. Critically, LRET has an advantage precisely in the range where classical simulations of FDMs begin to become burdensome. Furthermore, the space of circuit sizes in which LRET provides a significant advantage also characterizes the circuits of the early NISQ area with few tens of qubits, shallow-depth circuits, and with noise strengths $p < 0.01$[7].

The LRET method gains a speedup by truncating the negligible components of a density matrix. Although the truncation in each step is small, over time the discrepancy from the exact methods, like FDM, can build up. Here, we benchmark the error induced by the eigenvalue truncation in the LRET method.

Figure 4a shows the distortion as a function of the number of qubits (N), depth of the circuit (D), and eigenvalue truncation

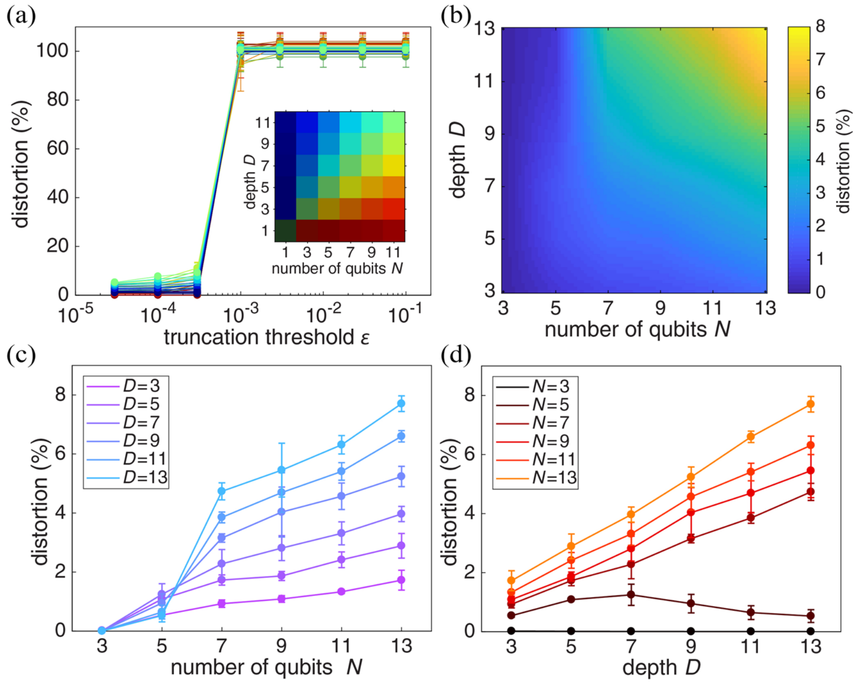

Fig. 4 Error benchmarking under depolarizing noise. a Distortion as a function of number of qubits N, circuit depth D, and ϵ. (Inset) The colormap for the curves, indicating their circuit size. **b** Distortion as a function of N and D with $\epsilon = 10^{-4}$. **c** Horizontal line cut of **b**. **d** Vertical line cut of **b**. Depolarizing noise has $p = 0.1\%$. The error bars are standard-deviation-calculated across a set of simulations.

threshold (ϵ). While the distortion depends on N and D, ϵ is the most impactful. There is a general trend that the error starts to increase rapidly for values larger than $\epsilon \simeq 10^{-4}$, so we take $\epsilon = 10^{-4}$ as a reasonable choice. The speedup for different ϵ is plotted in Fig. 3f, and shows the trade-off between accuracy and speedup; the speedup decreases when we increase accuracy (decrease distortion). From Fig. 4b, we can see that the error is <8% for all the N and D considered herein. By slicing out the number of qubits and circuit depths axes in Fig. 4b, we can see that the distortion grows roughly linearly with N and D (Fig. 4c, d).

We now see how the noise strength affects the performance of the LRET method. All benchmarking in this section uses dense

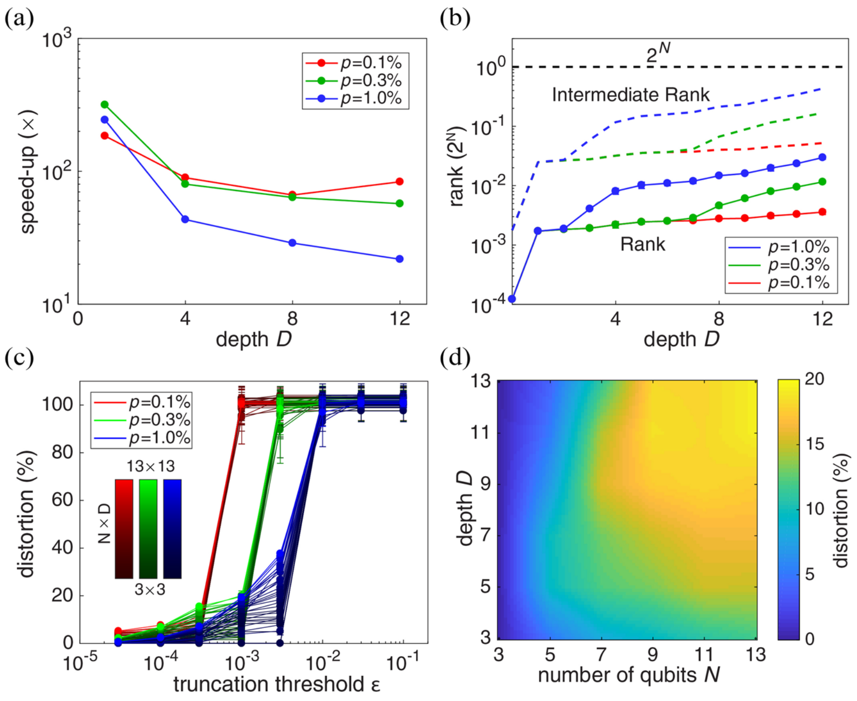

Fig. 5 Noise strength benchmarking under depolarizing noise. a Comparison of elapsed time of the LRET method for noise strength $p = 0.1\%$, $p = 0.3\%$, and $p = 1.0\%$ ($N = 13$). **b** Comparison of V and V_l for the LRET method for the three noise strengths ($N = 13$). **c** Distortion as a function of circuit size and eigenvalue truncation threshold ϵ for the three noise strengths. **d** Distortion for $p = 1\%$ and $\epsilon = 10^{-3}$ as a function of N and D. In **a**, **b**, and **d**, the ratio ϵ/p stays constant. The error bars are standard-deviation-calculated across a set of simulations.

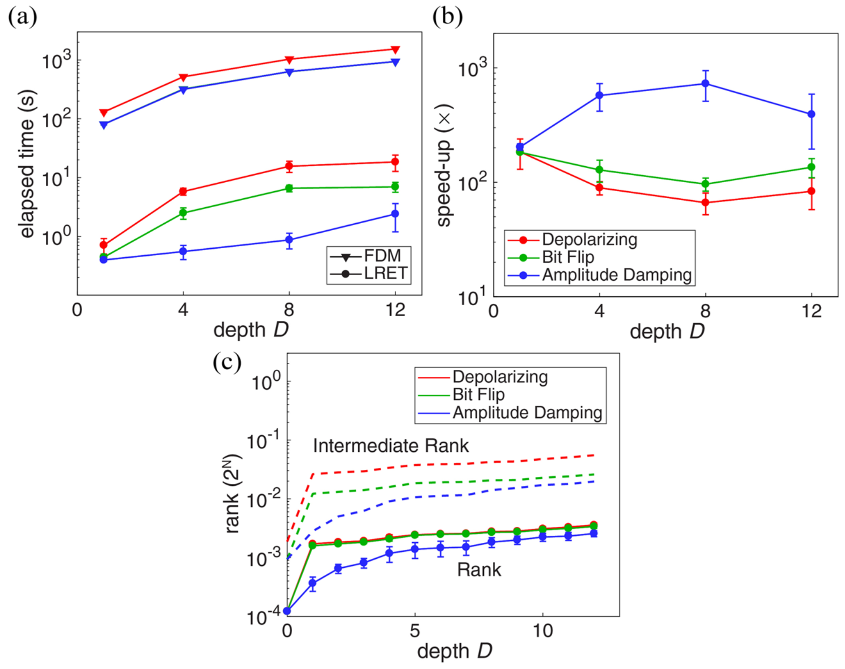

Fig. 6 Noise type benchmarking: time. a Averaged elapsed time using FDM and LRET. The color code is the same as **b**. **b** Speedup, defined by the ratio of elapsed time between FDM and LRET. **c** The evolution of rank (solid line) and intermediate rank (dashed line) for different D. Parameters are $p = 0.1\%$, $N = 13$, and $\epsilon = 10^{-4}$. The error bars are standard-deviation-calculated across a set of simulations.

circuits with depolarizing noise channels of various strengths. In Fig. 5a, the speedup of the LRET method against the FDM method degrades as the noise strength grows. From $p = 0.1\%$ to $p = 1\%$, the speedup drops from the order of 100× to 10× at $D = 12$. This degradation is due to the higher-order terms in noise which scale super-linearly in p. While the truncation threshold ϵ is adapted linearly by fixing the ratio $\epsilon/p = 0.1$ in this benchmarking, more higher-order terms need to be included to meet the truncation threshold. This results in a larger V and V_l (Fig. 5b), and thus a longer computational time.

Figure 5c shows that the distortion as a function of ϵ has a universal shape regardless of the circuit size and/or noise strength p. The magnitude of the distortion is relatively insensitive to circuit size. The noise strength p proportionally shifts the curves in the ϵ axis (i.e., when p and ϵ are scaled by a same factor, the error stays in a similar range).

Randomized benchmarking: other noise channels

We now consider noise simulations under bit flip and amplitude damping channels for dense circuits with $p = 0.1\%$ and $\epsilon = 10^{-4}$. Bit flip can be represented by $\rho \to (1-p)\rho + pX\rho X$ in the operator-sum formalism. In other words, the quantum state is rotated about the x-axis with probability p. The bit flip channel is

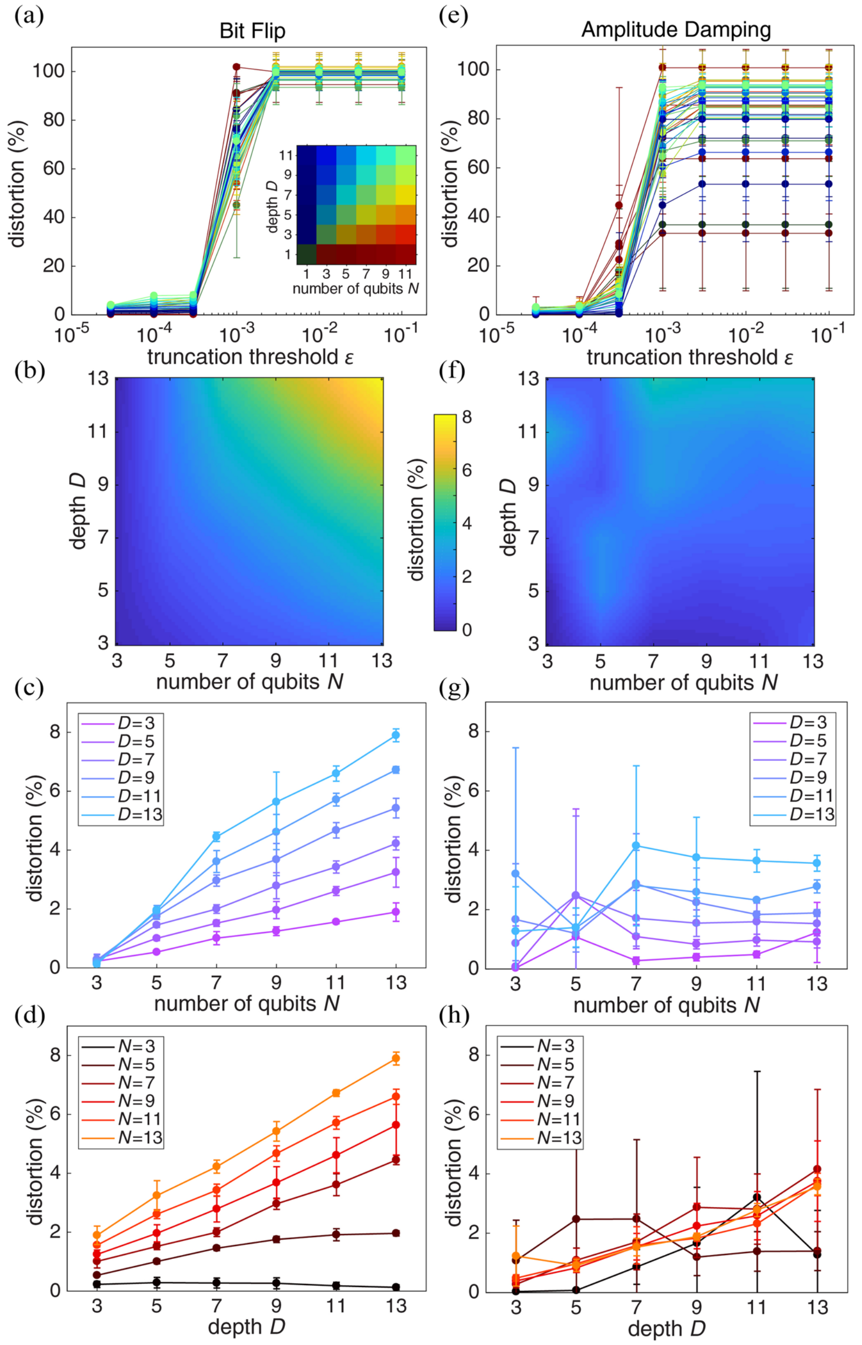

Fig. 7 Noise type benchmarking: error. Benchmarking of error under bit flip (left column) and amplitude damping (right column) with $p = 0.1\%$. **a**, **e** Distortion as a function of the number of qubits, circuit depth, and ϵ. (Inset) The colormap for the curves, indicating their their sizes of quantum circuit. **b**, **f** Distortion as a function of qubit number, circuit depth with $\epsilon = \times 10^{-4}$. **c**, **g** Horizontal line cut of **b** and **f**. **d**, **h** Vertical line cut of **b** and **f**. The error bars are standard-deviation-calculated across a set of simulations.

a special case of anisotropic noise. The results for the bit flip channel generalizes to other types of anisotropic noise in randomized benchmarking, such as phase flip and all other channels described by $\rho \to (1-p)\rho + pU\rho U^{\dagger}$ where U is any (2×2) unitary matrix. The amplitude damping channel dissipates the energy of a qubit toward its lower energy basis, usually denoted as the $|0\rangle$. We use the operator-sum formalism $\rho \to E_0\rho E_0 + E_1\rho E_1$, where E_0 and E_1 are Kraus operators for amplitude damping, defined as $E_0 = \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{1-p} \end{bmatrix}$ and $E_1 = \begin{bmatrix} 0 & \sqrt{p} \\ 0 & 0 \end{bmatrix}$.

Figure 6a shows that LRET is significantly faster than FDM for all noise types. Especially for the amplitude damping channel, where LRET completes in <3 s while FDM takes more than 25 min at depth = 12. The speedup of the depolarizing and bit flip channels are about 100× faster while the speedup for amplitude damping is almost 1000× faster (Fig. 6b). This is related to the slower increase of intermediate rank (Fig. 6c) due to the fact that amplitude damping has a preferred state, the $|0\rangle$ state, regardless of the details of the qubit state.

The error benchmarking for the bit flip channel, Fig. 7a–d, is very similar to that of the depolarizing channel, except that bit flip is more tolerant to ϵ when the circuit size is small. At $\epsilon = 10^{-3}$, the distortion is ~50% in the bit flip channel while it is ~100% in the depolarizing channel. When $\epsilon = 10^{-4}$, distortion is reasonably small for all N and D considered herein, so we take 10^{-4} as a recommended choice of ϵ.

In contrast to depolarizing and bit flip channels, under amplitude damping the distortion saturates at lower values when ϵ is large (Fig. 7e). This is possibly because amplitude damping prefers the ground state, favoring the LRET method, which keeps only few important components of a quantum state. When $\epsilon = 10^{-4}$, the distortion is smaller than 4% for all circuits considered in Fig. 7f. We see that the distortion grows slowly but linearly with circuit depth (Fig. 7g, h).

Randomized benchmarking: circuit sparsity and connectivity

A quantum circuit can be characterized by its sparsity and connectivity of gates. In all of the benchmarking above, the random circuits are dense, i.e., for all time steps a gate acts on each qubit (e.g., the circuit in Fig. 2a), and the connections are local, which means that two-qubit gates only connect adjacent qubits. Below, we consider other types of circuits. The first is dense and global, and the second is sparse and local. In sparse circuits, each qubit does not always interact with a gate at every time-step and Kraus operators are only inserted after gates (i.e., the noise is as sparse as the gates). In a globally connected circuit, two-qubit gates can connect any pair of qubits in the circuit. In this section, we use the bit flip channel for benchmarking the LRET method on all the aforementioned circuit-types with $p = 0.1\%$ and $\epsilon = 10^{-4}$.

In terms of time-cost, simulating different circuit types goes from harder to easier as circuits go from dense to sparse and from global to local. In Fig. 8a, one can see that the LRET method retains it's speedup for all circuit types. The time difference between the sparse and the dense is because there are about twice as many gates in dense circuits than in sparse circuits. The time difference between the global and the local is because the set of all fixed-depth globally connected circuits spans a larger Hilbert space. Therefore, the rank of dense-global circuits grows slightly faster (Fig. 8c) and the simulations are slightly slower (Fig. 8a). Interestingly, in Fig. 8a, one can observe that in FDM simulation, due to the number of gates and the memory allocation, the time-cost grows at a similar rate as that of LRET with increasing circuit depth. Thus, speedups are consistently ~100× faster with LRET than with FDM (Fig. 8b).

State preparation for quantum chemistry

Quantum simulation is one of the most promising application areas for NISQ devices[7]. Algorithms have been developed to solve physical and optimization problems through quantum simulation

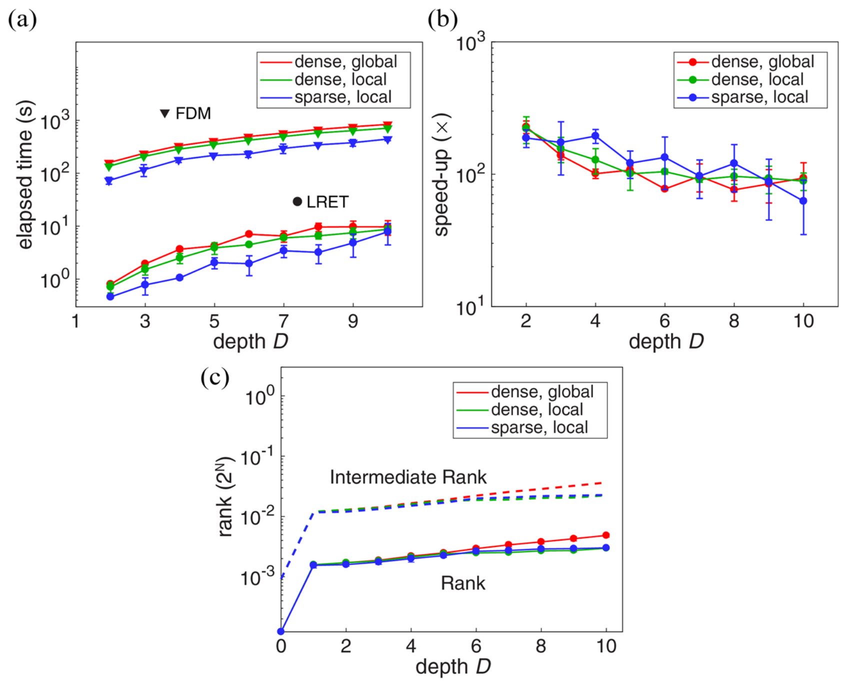

Fig. 8 Benchmarking for different circuit types under bit flip noise. a Averaged elapsed time for quantum circuits with different gate sparsity and connectivity. For the sparse circuit, the sparsity is 50%. **b** Speedup of LRET for different circuit types. **c** The evolution of rank (circle) and intermediate rank (dashed line) of density matrix for different circuit types. The distortion values for each of the circuits simulated using the LRET method are similar and are all <9.2%. The parameters are $p = 0.1\%$, $N = 13$, and $\epsilon = 10^{-4}$. The error bars are standard-deviation-calculated across a set of simulations.

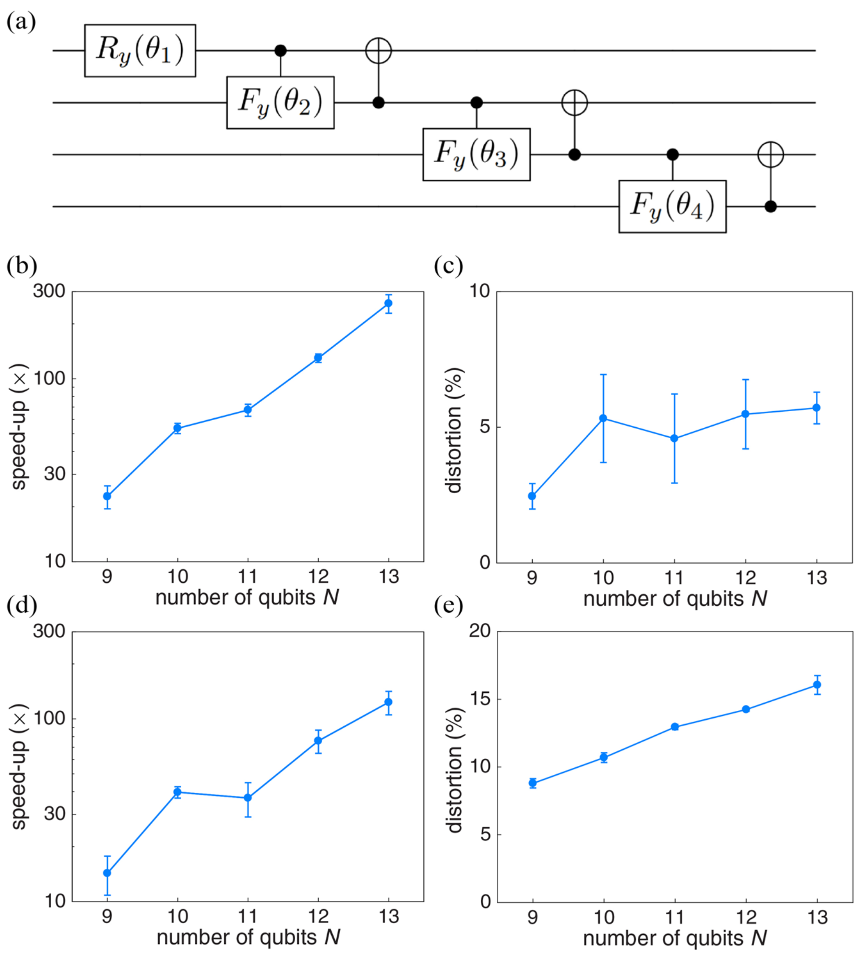

Fig. 9 Results of state preparation under depolarizing noise. a Circuit for state preparation with four qubits as an example. **b**, **c** Speedup and distortion of the LRET method under sparse noise channels. **d**, **e** Speedup and distortion of LRET under dense noise channels. Parameters are $p = 0.1\%$ and $\epsilon = 10^{-4}$. The error bars are standard-deviation-calculated across a set of simulations.

approaches[60–62]. Here, we use our low-rank noise simulator to run a circuit that generates generalized-amplitude W states[63] and Dicke states[64]. Although states of this kind are not hard to simulate classically, they are commonly used as subroutines for quantum information processing[65–69] and thus are of high interest for simulations. Ordinarily the parameters of the circuit are initialized according to the solution of a configuration interaction singles chemistry problem, but for benchmarking purposes, we set the parameters randomly.

Unlike most of the circuits used in the randomized benchmarking above, the circuit in Fig. 9a is sparse. Two ways to model noise channels are (1) placing Kraus operators only after each gate, and (2) after each qubit at every time-step. We call the former sparse noise, and the later dense noise. We use a depolarizing noise channel with $p = 0.1\%$ and $\epsilon = 10^{-4}$. Figure 9b, d shows that the LRET method has at least a 10× speedup, and more than 100× when N is larger. The distortion caused by LRET is about 5% in sparse noise and 15% in dense noise (Fig. 9c, e). For the 13-qubit circuits under sparse or dense noise, the rank of the final density matrix in LRET is just 0.4% or 1% of the full rank, respectively. The disparity is due to the rank of a density matrix in a circuit with dense noise increasing faster. The quantum states produced by this state preparation are highly entangled. The fact that the low-rank method gains an order of magnitude speedup demonstrates its utility when applied to practical algorithms.

Grover's Search algorithm and amplitude amplification

Amplitude amplification is a generalization of Grover's quantum search algorithm[70–72]. The algorithm aims to find a solution, x, such that $f(x) = 1$ if x is a solution and $f(x) = 0$ otherwise, implying that x is not a solution. If x is a solution we say that x is *good*. Let p_{good} be the probability of randomly sampling a search space, $\mathcal{X}$, and finding a good input. For a classical search algorithm, it is expected that one would have to sample from the input space on the order of $\frac{1}{p_{\text{good}}}$ times to find a solution; however, using amplitude amplification, one can expect to find a solution using in only $\mathcal{O}(\sqrt{\frac{1}{p_{\text{good}}}})$ samples—a quadratic speedup over the classical case[70,73].

In the algorithm, qubits are initialized to a uniform superposition over the entire search space, where each basis state corresponds to an element, $x \in \mathcal{X}$. Next, a number of unitaries, known as Grover Iterates, act on the initialized state and boost the amplitudes of states that correspond to good solutions. The number of Grover iterates to apply is given by $\lfloor\frac{\pi}{4}\sqrt{\frac{1}{p_{\text{good}}}}\rfloor$. A full measurement of the resulting circuit yields states corresponding to good solutions with high probability.

In our implementation, we define the function f such that $f(x) = 1$ (i.e., a good input), when the binary string, x, has a Hamming weight, HW(x), ≤2.

$$f = \begin{cases} 1 & \text{if HW(x)} \leq 2 \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

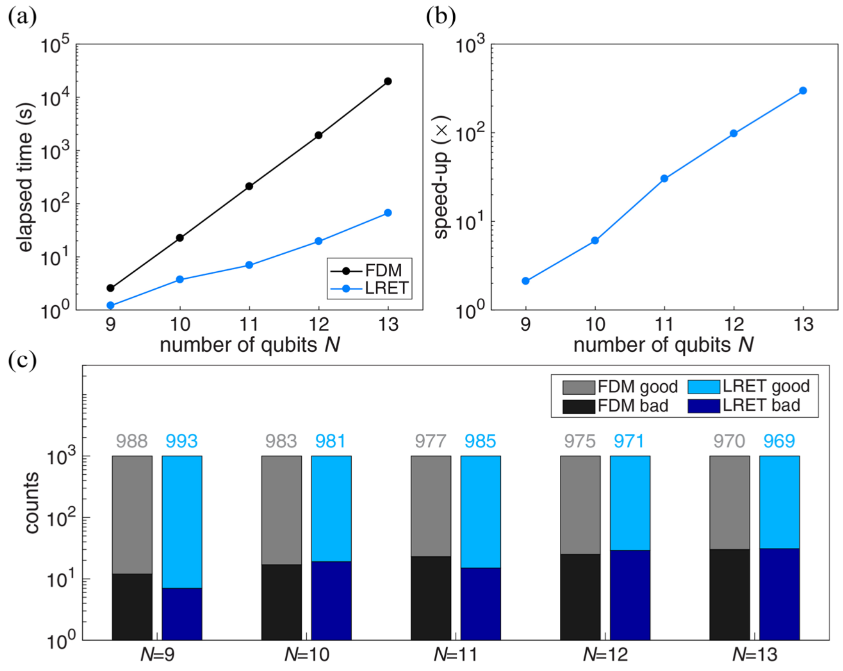

Fig. 10 Results of Grover's search algorithm under depolarizing noise. a Elapsed time on a quantum circuit using FDM (black) and LRET (light blue) in log scale. **b** Speedup, defined by the ratio of elapsed time between FDM and LRET, in log scale. **c** Counts of good and bad solutions. The sampling number is 1000. The number on top of each bar marks the counts of good solution found by Grover's Search algorithm for the corresponding method and number of qubits N. The distortion of LRET density matrices are <3.8% in these benchmarking. Parameters are $p = 0.1\%$ and $\epsilon = 10^{-4}$.

We run amplitude amplification on circuits ranging from 9 to 13 qubits under depolarizing noise with $p = 0.1\%$ and $\epsilon = 10^{-4}$ and compare LRET and FDM methods (Fig. 10). In the 13-qubit circuit, the rank of the final density matrix from LRET is 0.5% of the full rank with a trade-off of 3.7% distortion. The similarity of the measurement results from both methods demonstrates the accuracy of LRET; in other words, that any information loss from eigenvalue truncation is insignificant when sampling from the resulting density matrix. Time benchmarking of the two methods illustrates the speedup provided by LRET, which continues to improve as the number of qubits increases.

We note that it is not the intent of this study to most accurately predict the results of running this experiment on a particular hardware specification. When running on quantum hardware, gates must be decomposed into the set of gates native to the particular hardware, whereas, here, we model each of the Grover iterates as a single unitary. Rather, the aim of this study is to show that LRET retains its accuracy and computational advantage not only for the random circuits used for benchmarking, but also for circuits that may have legitimate applications and which exhibit more structure than the randomized circuits.

DISCUSSION

In this work we have demonstrated a method to simulate the evolution of mixed quantum states in noise channels that is efficient in practice. Iterative compression of the density matrices enable us to take advantage of low-rank evolution throughout the simulation of a noisy circuit with minimal error. Provided that the noise level of the channel is sufficiently small (<0.1–1% depending on the other simulation parameters), then the density matrices are found to be well-approximated by low-rank matrices. We provide an entropy argument in the supplementary information to support this finding. Under the low noise assumption, our results show that the algorithm provides orders of magnitude of speedup while only causing a small error, on the order of 10^{-4}, in the probability distributions associated with the output density matrix. The performance in speed and in distortion is robust in different circuit structures and for varying levels of entanglement since we make no assumption on the symmetry or the entanglement of the circuits.

Our approach is based in linear algebraic primitives, namely, matrix multiplication and eigendecomposition. Both are common routines in GPU libraries, such as GEMM in cuBLAS and eigensolvers in MAGMA. Since the eigendecomposition is performed on a comparatively small matrix, most of the run-time is devoted to matrix multiplications. It is fairly straightforward to use GPUs to accelerate these matrix multiplications, without the need for parallel reduction strategies. Therefore, one may utilize GPUs to speedup our algorithm even further, and potentially simulate in a regime that would ordinarily be intractable.

While our attention rests on the column space of density matrices, further speedups can be achieved by optimizing the representation with respect to the computational basis[14,74,75].

DATA AVAILABILITY

Source data for figures are available at https://github.com/tim-yitchen/Low-Rank-Simulator-Data.

CODE AVAILABILITY

Details of code that was used to analyze the data are available from the corresponding author upon reasonable request.

Received: 14 September 2020; Accepted: 18 February 2021;
Published online: 20 April 2021

REFERENCES

1. Pellizzari, T., Gardiner, S. A., Cirac, J. I. & Zoller, P. Decoherence, continuous observation, and quantum computing: a cavity QED Model. *Phys. Rev. Lett.* **75**, 3788 (1995).

2. Chuang, I. L., Laflamme, R., Shor, P. W. & Zurek, W. H. Quantum computers, factoring, and decoherence. *Science* **270**, 1633 (1995).
3. Copsey, D. et al. A design overview for a simulation infrastructure for exploring quantum architecture. In *Proc. 30th Annual International Symposium on Computer Architecture* (2003).
4. Ashhab, S., Johansson, J. R. & Nori, F. Decoherence in a scalable adiabatic quantum computer. *Phys. Rev. A* **74**, 52330 (2006).
5. Arute, F. et al. Quantum supremacy using a programmable superconducting processor. *Nature* **574**, 505–510 (2019).
6. Gomes, L. Quantum computing: both here and not here. *IEEE Spectr.* **55**, 42–47 (2018).
7. Preskill, J. Quantum computing in the NISQ era and beyond. *Quantum* **2**, 79 (2018).
8. Harper, R., Flammia, S. T. & Wallman, J. J. Efficient learning of quantum noise. *Nat. Phys.* **16**, 1184–1188 (2020).
9. Pednault, E. et al. Breaking the 49-Qubit barrier in the simulation of quantum circuits. http://arxiv.org/abs/1710.05867v3 (2017).
10. Chen, Z.-Y. et al. 64-qubit quantum circuit simulation. *Sci. Bull.* **63**, 964–971 (2018).
11. Dang, A., Hill, C. D. & Hollenberg, L. C. L. Optimising matrix product state simulations of shor's algorithm. *Quantum* **3**, 116 (2019).
12. Bravyi, S. & Gosset, D. Improved classical simulation of quantum circuits dominated by Clifford gates. *Phys. Rev. Lett.* **116**, 250501 (2016).
13. Jozsa, R. & Van Den Nest, M. Classical simulation complexity of extended Clifford circuits. *Quantum Info Comput.* **14**, 633–648 (2014).
14. Vidal, G. Efficient classical simulation of slightly entangled quantum computations. *Phys. Rev. Lett.* **91**, 147902 (2003).
15. Plesch, M. & Bužek, V. Efficient compression of quantum information. *Phys. Rev. A* **81**, 32317 (2010).
16. Bartlett, S. D., Sanders, B. C., Braunstein, S. L. & Nemoto, K. Efficient classical simulation of continuous variable quantum information processes. *Phys. Rev. Lett.* **88**, 97904 (2002).
17. Yoran, N. & Short, A. J. Efficient classical simulation of the approximate quantum Fourier transform. *Phys. Rev. A* **76**, 42321 (2007).
18. Browne, D. E. Efficient classical simulation of the quantum Fourier transform. *N. J. Phys.* **9**, 146 (2007).
19. Shi, Y.-Y., Duan, L.-M. & Vidal, G. Classical simulation of quantum many-body systems with a tree tensor network. *Phys. Rev. A* **74**, 22320 (2006).
20. Kassal, I., Jordan, S. P., Love, P. J., Mohseni, M. & Aspuru-Guzik, A. Polynomial-time quantum algorithm for the simulation of chemical dynamics. *Proc. Natl Acad. Sci.* **105**, 18681–18686 (2008).
21. Khammassi, N., Ashraf, I., Fu, X., Almudever, C. G. & Bertels, K. QX: a high-performance quantum computer simulation platform. In *Proc. Design Automation & Test in Europe Conference & Exhibition* 464–469 (2017).
22. Wei, S.-J., Xin, T. & Long, G.-L. Efficient universal quantum channel simulation in IBM's cloud quantum computer. *Sci. China Phys., Mech. Astron.* **61**, 70311 (2018).
23. Chaudhary, H. et al. A software simulator for noisy quantum circuits. http://arxiv.org/abs/1908.05154 (2019).
24. Jones, T., Brown, A., Bush, I. & Benjamin, S. C. QuEST and high performance simulation of quantum computers. *Sci. Rep.* **9**, 10736 (2019).
25. Aleksandrowicz, G. et al. Qiskit: an open-source framework for quantum computing. http://qiskit.org/ (2019).
26. Google. Cirq: a python framework for creating, editing, and invoking Noisy Intermediate Scale Quantum (NISQ) circuits. http://github.com/quantumlib/Cirq (2019).
27. Breuer, H.-P. & Petruccione, F. *The Theory of Open Quantum Systems* (Oxford University Press, 2007).
28. Dalibard, J., Castin, Y. & Mølmer, K. Wave-function approach to dissipative processes in quantum optics. *Phys. Rev. Lett.* **68**, 580–583 (1992).
29. Mølmer, K., Castin, Y. & Dalibard, J. Monte Carlo wave-function method in quantum optics. *J. Opt. Soc. Am. B* **10**, 524–538 (1993).
30. Bassi, A. & Deckert, D.-A. Noise gates for decoherent quantum circuits. *Phys. Rev. A* **77**, 32323 (2008).
31. Guerreschi, G. G., Hogaboam, J., Baruffa, F. & Sawaya, N. P. D. Intel Quantum Simulator: a cloud-ready high-performance simulator of quantum circuits. *Quantum Sci. Technol.* **5**, 34007 (2020).
32. Abid Moueddene, A., Khammassi, N., Bertels, K. & Almudever, C. G. Realistic simulation of quantum computation using unitary and measurement channels. *Phys. Rev. A* **102**, 052608 (2020).
33. Gorini, V., Kossakowski, A. & Sudarshan, E. C. G. Completely positive dynamical semigroups of N level systems. *J. Math. Phys.* **17**, 821–825 (1976).
34. Lindblad, G. On the generators of quantum dynamical semigroups. *Comm. Math. Phys.* **48**, 119–130 (1976).
35. Jelezko, F., Gaebel, T., Popa, I., Gruber, A. & Wrachtrup, J. Observation of coherent oscillations in a single electron spin. *Phys. Rev. Lett.* **92**, 76401 (2004).
36. Raitzsch, U. et al. Investigation of dephasing rates in an interacting Rydberg gas. *N. J. Phys.* **11**, 55014 (2009).
37. Fitzpatrick, M., Sundaresan, N. M., Li, A. C. Y., Koch, J. & Houck, A. A. Observation of a dissipative phase transition in a one-dimensional circuit QED lattice. *Phys. Rev. X* **7**, 11016 (2017).
38. van Handel, R. & Mabuchi, H. Quantum projection filter for a highly nonlinear model in cavity QED. *J. Opt. B Quantum Semiclassical Opt.* **7**, S226 (2005).
39. Le Bris, C. & Rouchon, P. Low-rank numerical approximations for high-dimensional Lindblad equations. *Phys. Rev. A* **87**, 22125 (2013).
40. Gross, D., Liu, Y.-K., Flammia, S. T., Becker, S. & Eisert, J. Quantum state tomography via compressed sensing. *Phys. Rev. Lett.* **105**, 150401 (2010).
41. Kyrillidis, A. et al. Provable compressed sensing quantum state tomography via non-convex methods. *npj Quantum Inf.* **4**, 36 (2018).
42. Vidal, G. Efficient simulation of one-dimensional quantum many-body systems. *Phys. Rev. Lett.* **93**, 40502 (2004).
43. White, C. D., Zaletel, M., Mong, R. S. K. & Refael, G. Quantum dynamics of thermalizing systems. *Phys. Rev. B* **97**, 35127 (2018).
44. Motta, M. et al. Low rank representations for quantum simulation of electronic structure. http://arxiv.org/abs/1808.02625 (2018).
45. Cao, K., Zhou, Z.-W., Guo, G.-C. & He, L. Efficient numerical method to calculate the three-tangle of mixed states. *Phys. Rev. A* **81**, 34302 (2010).
46. Wu, X.-C. et al. Memory-efficient quantum circuit simulation by using lossy data compression. http://arxiv.org/abs/1811.05630 (2018).
47. Wu, X.-C. et al. Amplitude-aware lossy compression for quantum circuit simulation. http://arxiv.org/abs/1811.05140 (2018).
48. Kraus, K., Böhm, A., Dollard, J. & Wootters, W. *States, Effects, and Operations Fundamental Notions of Quantum Theory*, Vol. 190 (Springer, 1983).
49. Bacon, D. et al. Universal simulation of Markovian quantum dynamics. *Phys. Rev. A* **64**, 62302 (2001).
50. Nielsen, M. A. & Chuang, I. L. *Quantum Computation and Quantum Information: 10th Anniversary Edition* 10th edn. (Cambridge University Press, 2011).
51. Alquier, P., Butucea, C., Hebiri, M., Meziani, K. & Morimae, T. Rank-penalized estimation of a quantum system. *Phys. Rev. A* **88**, 32113 (2013).
52. Butucea, C., Guţă, M. & Kypraios, T. Spectral thresholding quantum tomography for low rank states. *N. J. Phys.* **17**, 113050 (2015).
53. Trefethen, L. & Bau, D. *Numerical Linear Algebra* (SIAM, 1997).
54. Crooks, G. *On Measures of Entropy and Information*. (2015).
55. Emerson, J., Alicki, R. & Życzkowski, K. Scalable noise estimation with random unitary operators. *J. Opt. B Quantum Semiclassical Opt.* **7**, S347 (2005).
56. Knill, E. et al. Randomized benchmarking of quantum gates. *Phys. Rev. A* **77**, 12307 (2008).
57. Onorati, E., Werner, A. H. & Eisert, J. Randomized benchmarking for individual quantum gates. *Phys. Rev. Lett.* **123**, 60501 (2019).
58. Emerson, J., Weinstein, Y. S., Lloyd, S. & Cory, D. G. Fidelity decay as an efficient indicator of quantum chaos. *Phys. Rev. Lett.* **89**, 284102 (2002).
59. Weinstein, Y. S. et al. Quantum process tomography of the quantum Fourier transform. *J. Chem. Phys.* **121**, 6117–6133 (2004).
60. Abrams, D. S. & Lloyd, S. Simulation of many-body fermi systems on a universal quantum computer. *Phys. Rev. Lett.* **79**, 2586–2589 (1997).
61. Peruzzo, A. et al. A variational eigenvalue solver on a photonic quantum processor. *Nat. Commun.* **5**, 4213 (2014).
62. Farhi, E., Goldstone, J. & Gutmann, S. A quantum approximate optimization algorithm. http://arxiv.org/abs/1411.4028 (2014).
63. Diker, F. Deterministic construction of arbitrary W states with quadratically increasing number of two-qubit gates. http://arxiv.org/abs/1606.09290 (2016).
64. Bärtschi, A. & Eidenbenz, S. Deterministic preparation of Dicke states. In *Proc. 22nd International Symposium Fundamentals of Computation Theory* 126–139 (Cham, 2019).
65. Murao, M., Jonathan, D., Plenio, M. B. & Vedral, V. Quantum telecloning and multiparticle entanglement. *Phys. Rev. A* **59**, 156–161 (1999).
66. Prevedel, R. et al. Experimental realization of Dicke states of up to six qubits for multiparty quantum networking. *Phys. Rev. Lett.* **103**, 20503 (2009).
67. Pezzè, L., Smerzi, A., Oberthaler, M. K., Schmied, R. & Treutlein, P. Quantum metrology with nonclassical states of atomic ensembles. *Rev. Mod. Phys.* **90**, 35005 (2018).
68. Parrish, R. M., Hohenstein, E. G., McMahon, P. L. & Martinez, T. J. Quantum computation of electronic transitions using a variational quantum eigensolver. *Phys. Rev. Lett.* **122**, 230401 (2019).
69. Hadfield, S. et al. From the quantum approximate optimization algorithm to a quantum alternating operator Ansatz. *Algorithms* **12**, 34 (2019).
70. Grover, L. K. A fast quantum mechanical algorithm for database search. In *Proc. Twenty-eighth Annual ACM Symposium on Theory of Computing* 212–219 (ACM, 1996).
71. Grover, L. K. A framework for fast quantum mechanical algorithms. In *Proc. Thirtieth Annual ACM Symposium on Theory of Computing* 53–62 (ACM, 1998).

72. Tulsi, A. Quantum computers can search rapidly by using almost any selective transformation. *Phys. Rev. A* **78**, 22332 (2008).
73. Brassard, G., HØyer, P. & Tapp, A. Quantum counting BT—automata, languages and programming. In *Proc. International Colloquium on Automata, Languages and Programming* 820–831 (Springer, 1998).
74. Zhou, Y., Miles Stoudenmire, E. & Waintal, X. What limits the simulation of quantum computers? *Phys. Rev. X* **10**, 041038 (2020).
75. Noh, K., Jiang, L. & Fefferman, B. Efficient classical simulation of noisy random quantum circuits in one dimension. *Quantum* **4**, 318 (2020).

ACKNOWLEDGEMENTS

This work was supported by the U.S. Department of Energy, Office of Science, Basic Energy Sciences, Chemical Sciences, Geosciences and Biosciences Division. The authors thank Dr Sean Weinberg, Juan I. Adame, Dr Fabio Sanches, and Dr Adam Bouland for discussions on the theoretical ground of this work and providing useful references.

AUTHOR CONTRIBUTIONS

R.M.P. conceived the algorithm, and Y.-T.C., C.F., and R.M.P. conceived the numerical experiments. Y.-T.C. and C.F. carried out the implementation and the benchmarking. Y.-T.C. wrote the initial draft and all authors contributed to the revisions and editing of the manuscript.

COMPETING INTERESTS

R.M.P. owns stock/options in QC Ware Corporation.

ADDITIONAL INFORMATION

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41534-021-00392-4.

Correspondence and requests for materials should be addressed to Y.-T.C.