Near-Term Quantum Computing Techniques: Variational Quantum Algorithms, Error Mitigation, Circuit Compilation, Benchmarking and Classical Simulation

He-Liang Huang[#,1,*] Xiao-Yue Xu[#,1] Chu Guo[#,1] Guojing Tian[#,2,3]

Shi-Jie Wei,[4] Xiaoming Sun,[2,3,†] Wan-Su Bao,[1,‡] and Gui-Lu Long[5,4,§]

[1]*Henan Key Laboratory of Quantum Information and Cryptography, Zhengzhou, Henan 450000, China*

[2]*State Key Lab of Processors, Institute of Computing Technology, Chinese Academy of Sciences, 100190 Beijing, China*

[3]*School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing 100049, China*

[4]*Beijing Academy of Quantum Information Sciences, Beijing 100193, China*

[5]*Department of Physics, Tsinghua University, Beijing 100084, China*

Quantum computing is a game-changing technology for global academia, research centers and industries including computational science, mathematics, finance, pharmaceutical, materials science, chemistry and cryptography. Although it has seen a major boost in the last decade, we are still a long way from reaching the maturity of a full-fledged quantum computer. That said, we will be in the *Noisy-Intermediate Scale Quantum* (NISQ) era for a long time, working on dozens or even thousands of qubits quantum computing systems. An outstanding challenge, then, is to come up with an application that can reliably carry out a nontrivial task of interest on the near-term quantum devices with non-negligible quantum noise. To address this challenge, several near-term quantum computing techniques, including variational quantum algorithms, error mitigation, quantum circuit compilation and benchmarking protocols, have been proposed to characterize and mitigate errors, and to implement algorithms with a certain resistance to noise, so as to enhance the capabilities of near-term quantum devices and explore the boundaries of their ability to realize useful applications. Besides, the development of near-term quantum devices is inseparable from the efficient classical simulation, which plays a vital role in quantum algorithm design and verification, error-tolerant verification and other applications. This review will provide a thorough introduction of these near-term quantum computing techniques, report on their progress, and finally discuss the future prospect of these techniques, which we hope will motivate researchers to undertake additional studies in this field.

I. INTRODUCTION

Quantum computing is a rapidly emerging new-generation computing paradigm that harnesses the laws of quantum mechanics, offering the potential of exponential speedup over classical computation for certain problems [1–4]. Over the last decade, quantum computing technologies have advanced by leaps and bounds into the NISQ era [5, 6], an important sign of which is the significant achievement of quantum computational advantage on quantum sampling problems has been realized in practice [7–12]. The near-term quantum processors, including superconducting qubits [5, 13, 14], trapped ions [15, 16], and photons [17–19], *etc.*, produced during this period contain only a few dozen or even a few thousand qubits, falling well short of the specifications for fault-tolerant quantum computing [20–24] (see TABLE. I for system parameters of some prototypes). They do, however, serve as fantastic testing grounds for investigating a variety of applications, including machine learning [25], secure cloud quantum computing [26–30], computational science, and complicated quantum chemistry [31] and many-body quantum systems [32] that are not feasible to be simulated with the state-of-the-art supercomputers.

To fully exploit the potential of near-term quantum devices,

algorithms/protocols have to be tailored to the constraint of present quantum hardware, especially a modest of qubits with non-negligible errors and limited qubit connectivity. To mitigate quantum errors and achieve valid computational results, several prominent classes of algorithms and tools have been developed specifically for near-term quantum computers, including:

- *Variational Quantum Algorithms (VQAs)* [33, 34]. VQAs have emerged as one of the leading candidates to achieve application-oriented quantum computational advantage on NISQ devices, owing to their hybrid quantum-classical approach which has potential noise resilience.

- *Quantum Error Mitigation (QEM)* [35]. QEM refers to a series of techniques that allow us to reduce the computational errors and then evaluate accurate results from noisy quantum circuits, although we still lack practical quantum error-correction technique.

- *Quantum Circuit Compilation (QCC)* [36–38]. QCC is a key technique to transform the nonconforming quantum circuit to an executable circuit on the target quantum platform according to its constraints, including the native gateset, connectivity and so on.

- *Benchmarking Protocols* [39, 40]. Quantum benchmarking helps to evaluate the basic performance of a quantum computer and even the capacity to solve real-world problems.

- *Classical Simulation* [41–45]. Classical simulation of quantum circuits is one of the core tools for designing quantum algorithms and validating quantum devices.

These techniques work together to accomplish computational tasks rather than independently of each other (see a rough relationship provided in Fig. 1). For instance, variational quantum algorithms, quantum error mitigation, quantum circuit compilation, and quantum benchmarking may all require the help of classical simulation for verification or algorithm design. These near-term quantum computing techniques offer the opportunity to attain practical quantum advantages that can be applied to significant applications such as chemistry [46, 47] and machine learning [48–61], and they also provide a continuous path for the advancement of quantum computing from today's near-term quantum hardware to tomorrow's fault-tolerant quantum computers.

This review aims to present an overview of these crucial techniques, part of which may serve as a supplement and update to related published review works [33–35, 39], and some algorithms and protocols are step-by-step taught to help the reader rapidly understand the specifics. We also discuss the challenges and opportunities, and offer some insights on potential future advances.
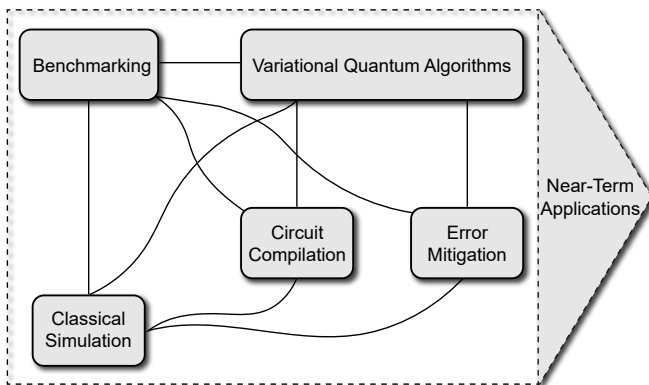

FIG. 1. **Relationship between near-term quantum techniques.** These techniques collaborate to execute various computing tasks.

TABLE I. System parameters of some quantum computing prototypes that have achieved quantum computational advantage.

Superconducting system	Sycamore [7]	*Zuchongzhi* 2.1 [8, 9]
Number of qubits	53	66
Single-qubit gate error	0.16%	0.16%
Two-qubit gate error	0.62%	0.60%
Readout error	3.8%	2.26%
T_1	16.04 μs	26.5 μs
Photonic system	*Jiuzhang* 2.0 [11]	Borealis [12]
Detected photons	113	219
Interferometer	144-mode	216-mode

II. VARIATIONAL QUANTUM ALGORITHMS

Although some quantum algorithms, such as Shor's algorithm [3] and Grover's search algorithm [4], promise to be much faster than classical algorithms, these algorithms are unattainable in the NISQ era due to the lack of error correction capability. Variational quantum algorithms (VQAs) have been shown to be naturally resistant [62] to, and even benefit from, noise, making them particularly suitable for near-term quantum devices, and are therefore considered the most promising route to quantum advantage on practical problems in the NISQ era [33]. In this section, we will first introduce the basic concepts, architectures and applications of VQAs, and then discuss the opportunities and challenges for future industrial applications or scientific research.

A. Basic concepts

Similar to the classical neural networks, the VQAs is implemented by training a parametrized quantum circuit (PQC) to minimize a problem-specific cost function. Thus, the basic components of VQAs include cost function, PQC and optimization algorithms, as shown in Fig. 2.

1. Cost function

The cost function (sometimes referred to as the loss or objective function) is an integral part of VQAs for encoding problems, and serves essentially the same role as the cost function in classical machine learning. During the training process, the cost function can be considered as a measure of the performance of a VQA with respect to the given training samples and the expected output, which helps to find the global minima. Without loss of generality, the cost function can be expressed as

$$C(\boldsymbol{\theta}) = f(O(U(\boldsymbol{\theta}), \rho_{\text{in}})) \tag{1}$$

where O are a set of observables on the output state obtained by the input state ρ_{in} under the action of the PQC $U(\boldsymbol{\theta})$ with tunable parameters $\boldsymbol{\theta}$, and f is some function, which is designed according to the specific problem. For example, the cost function of variational quantum eigensolver is often set as

$$\langle H \rangle_{O(U(\boldsymbol{\theta}), \rho_{\text{in}})}, \tag{2}$$

which is expectation value of qubit Hamiltonian H. In addition, for quantum machine learning tasks, one can usually follow the cost functions commonly used in classical community, such as using mean squared error (MSE) or cross entropy as the cost function for the classification task.

The choice of cost function could affect the trainability of VQAs. Cerezo *et al.* proved that barren plateaus are cost-function-dependent in shallow PQCs [63]. Specifically, the cost function with global observables leads to an exponentially vanishing gradient (*i.e.*, barren plateau), while the gradient vanishes polynomially when the cost function is defined
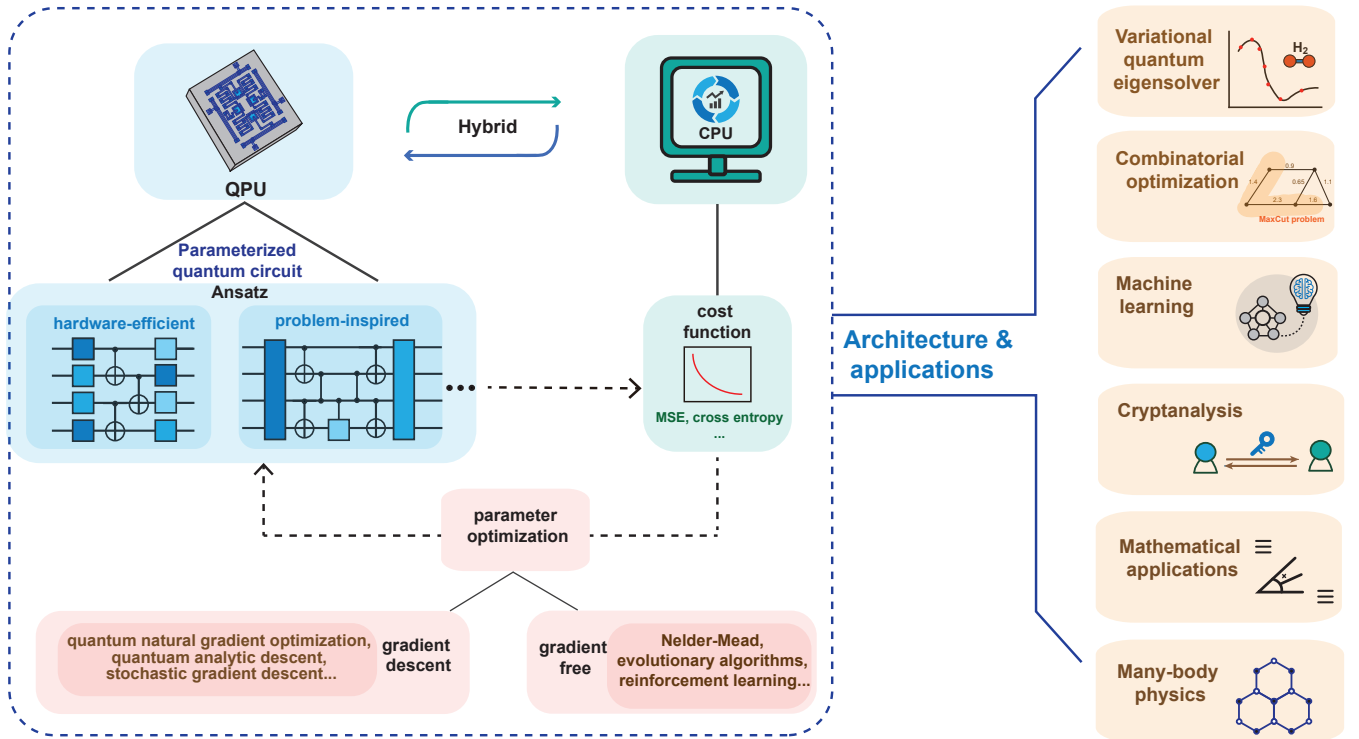
FIG. 2. **A high-level overview of variational quantum algorithms (VQAs).** VQAs are a hybrid quantum-classical optimization algorithm. During the training procedure, the quantum processing unit (QPU) estimates the cost function via the parameterized quantum circuit (PQC). Then a classical processing unit (CPU) is employed to implement the parameter optimization. VQAs can be analogous to the quantum counterpart for deep learning, and thus have the potential to be applied to a range of applications, such as find the ground state, combinatorial optimization, machine learning, *etc*.

in terms of local observables. Recently, Liu *et al.* found a similar phenomenon in tensor-network based machine learning [64].

2. *Parameterized quantum circuits (PQCs)*

PQCs are the main part that differentiates VQAs from classical neural networks. A PQC consists of a number of fixed quantum gates and trainable quantum gates, and may even includes some measurement and feedback operations. The trainable quantum gate is usually a single-qubit rotation, $R_x(\theta)$, $R_y(\theta)$, and $R_z(\theta)$ rotations, and the trainable parameter is its rotation angle θ. These trainable parameters are trained to minimize the cost function.

The structure of the variational ansatz plays an essential role in the performance of VQA, including the convergence speed and the closeness of the output to the optimal solution of the problem. The design of an effective structure is a challenge for VQAs, which requires optimization in many aspects, such as strong expressibility, shallow circuit depth, and small number of parameters. In the following, we will present some common ansatzes.

Problem-inspired ansatzes. The problem-inspired ansatz is constructed by using knowledge about a specific problem.

A typical example is the quantum approximate optimization algorithm (QAOA) [65], whose ansatz is set as

$$U(\boldsymbol{\gamma}, \boldsymbol{\beta}) = \prod_{l=1}^{p} e^{-i\beta_l H_M} e^{-i\gamma_l H_C}, \tag{3}$$

where H_C is the problem Hamiltonian, $H_M = \sum_{i=1}^{n} \sigma_x^i$ is the mixing Hamiltonian, and σ_x^i denotes the Pauli X operator acting on qubit i. This ansatz is inspired by quantum annealing to map the initial state to the ground state of the problem Hamiltonian H_C, by optimizing the parameters $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, ..., \gamma_p)$ and $\boldsymbol{\beta} = (\beta_1, \beta_2, ..., \beta_p)$.

The problem-inspired ansatz is also widely used in quantum chemistry problems. The unitary coupled cluster (UCC) [66] ansatz is a commonly used structure for variational quantum eigensolver (VQE). And historically, the first VQE experiment by Peruzzo *et al.* [67] utilized the UCC with singles and doubles (UCCSD) ansatz. Subsequently, some improved ansatzes enable shallower depths and higher accuracy. Lee *et al.* proposed the unitary pair coupled cluster with generalized singles and doubles (k-UpCCGSD) method to reduce circuit depth [68]. The orbital optimized UCC (OO-UCC) ansatz [69] is another variant of UCC to reduce the number of parameters and circuit depth, while maintains a similar level of accuracy to that of UCCSD. Besides reducing circuit depth,

Metcalf *et al.* employed the double unitary coupled-cluster (DUCC) method to effectively reduce the required number of qubits [70]. Here we only make a list of these ansatzes, more detailed introduction can be found in the Ref. [71].

In general, the problem-inspired ansatz is designed according to the characteristics of the problem, so that the output has a high accuracy. However, limited by the connectivity of quantum processors, implementing problem-inspired ansatz on real quantum devices may necessitate a relatively deep PQC, which might be a challenge.

Hardware-efficient ansatzes. Unlike the problem-inspired ansatz, the hardware-efficient ansatz [46] is designed around properties of quantum hardware, such as the qubit connectivity and restricted gate sets, to ensure efficient implementation on near-term quantum devices.

The trainable unitary operator $U(\boldsymbol{\theta})$, represented by the PQC of this anstz, is composed of L layers and each layer merges trainable single-qubit gates with a entangling block. Mathematically, we have $U(\boldsymbol{\theta}) := \prod_{l=1}^{L}(U_E U_l(\boldsymbol{\theta}))$, where $U_l(\boldsymbol{\theta})$ is the l-th trainable layer and U_E is the entanglement layer. In particular, we have $U_l(\boldsymbol{\theta}) = \bigotimes_{i=1}^{N}(U_S(\boldsymbol{\theta}^{(i,l)}))$, where $\boldsymbol{\theta}^{(i,l)}$ represents the (i,l)-th entry of $\boldsymbol{\theta} \in \mathcal{R}^{N \times L}$, U_S is the trainable unitary with $U_S \in SU(2)$, *e.g.*, the rotation single qubit gates R_x, R_y, and R_z. The entangle layer U_E is constructed from the entanglement operations that the hardware can provide, such as CNOT, CZ gates, iSWAP gate, and even many-body quantum evolution. These entanglement operations should follow the connectivity of quantum hardware, such as one-dimensional chains, lattices, *etc.*

Variable-structure ansatzes. Unlike the problem-inspired ansatz and hardware-efficient ansatz, which take a fixed structure, the circuit structure in the variable-structure ansatz is no longer fixed and trained as a parameter with the aim of further reducing the circuit depth and the number of gates. A typical example is ADAPT-VQE [72] and qubit-ADAPT-VQE [73], which are proposed to optimize the circuit structure of the ansatz to reduce the circuit depth in quantum chemistry applications. Zhu *et al.* also developed an adaptive QAOA for solving combinatorial problems [74]. Interested readers can refer to the Refs. [75–81] for other variants or optimizations, and we will not introduce them one by one.

3. Parameter optimization

The procedure of parameter optimization is implemented after the cost function and ansatz are determined, to minimize the cost function. An appropriate optimizer should be determined by considering the requirements of the application. There are mainly two classes of optimization methods, gradient-based and gradient-free methods, which we will introduce below.

Gradient descent methods. Gradient descent is a general-purpose optimization algorithm for finding a local minimum of the given cost function. For this purpose, the gradient of a cost function $C(\boldsymbol{\theta})$ with respect to its parameters $\boldsymbol{\theta}$, *i.e.*, $\nabla_{\boldsymbol{\theta}} C(\boldsymbol{\theta})$, is calculated to obtain the direction of steepest descent. Then a small updates to the parameters is implemented

according to

$$\boldsymbol{\theta} \rightarrow \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} C(\boldsymbol{\theta}), \tag{4}$$

where η is the step size, to iteratively approach the minimum value of the cost function.

The most commonly used gradient descent method for VQAs is parameter-shift rule [82–85]. As shown in Eq. 1, the cost functions are usually phrased in terms of expectation values of some observables O, evaluated on a output quantum state ρ_{out} of the PQC $U(\boldsymbol{\theta})$. In the case that ρ_{out} depends on a parameter θ_i that parametrizes a Pauli-rotation gate $e^{-i\theta_i P/2}$, where P is a Pauli operator, we can compute the derivative with respect to θ_i as

$$\nabla_{\theta_i} \langle O \rangle (\theta_i) = \frac{1}{2} \left(\langle O \rangle (\theta_i + \frac{\pi}{2}) - \langle O \rangle (\theta_i - \frac{\pi}{2}) \right). \tag{5}$$

The original parameter-shift rule is design for single-parameter gates. Wierichs *et al.* then extended this method to be applicable to multi-parameter quantum gates [86].

The gradient descent methods can also be transformed into the quantum version [87–92]. The quantum versions of the gradient descent was firstly proposed by Rebentrost *et al.* for high-dimensional optimization problems [93]. The gradient algorithm in Ref. [93] involves phase estimation, which requires substantial circuit depth and is difficult to implement with current quantum hardwares. Instead of using phase estimation, Li *et al.* proposed an experimental friendly quantum gradient algorithm [87] using the linear combination of unitaries (LCU).

In addition to above methods, some other gradient descent methods have been proposed, such as quantum analytic descent [94] and stochastic gradient descent [85, 95, 96], *etc.*

Gradient-free methods. Similar to the classical machine learning, gradient-free optimization methods can also be used to optimize PQCs. The Nelder-Mead method is one of commonly used gradient-free optimization for finding a local minimum of a function of several variables, and thus is naturally suitable for VQAs as well. Besides, evolutionary algorithms and reinforcement learning can be also used to train VQAs [97–101]. These optimization methods are standard in the classical machine learning community and we will not go into detail in this review.

B. Architectures and applications

VQAs can be applied to a wide range of fields, including finding ground states of molecules, combinatorial optimization, machine learning, *etc.* Here, we will introduce some typical applications and their experimental progress.

1. Variational quantum eigensolver (VQE)

The VQE [62, 67] is a flagship algorithm for NISQ applications. This type of algorithms allows one to find the ground state of a given Hamiltonian H, which may be used

for simulating molecules and chemical reactions. According to the Rayliegh-Ritz variational principle [102–104], the ground state energy E_0 associated with this Hamiltonian H is bounded by

$$E_0 \leq \frac{\langle \psi | H | \psi \rangle}{\langle \psi | \psi \rangle}, \tag{6}$$

where $|\psi\rangle$ is a trial quantum state. The original VQE is to find a quantum state approximating the ground state by training a PQC, such that the expected value of the Hamiltonian is constantly approaching the minimum. In addition to its original purpose, VQE has been widely extended to other objectives, such as finding excited states or the spectrum of Hamiltonian [105–111], nonequilibrium steady state [112–114], and calculating energy derivatives [115–117]. Recently, Wei *et al.* propose a full quantum eigensolver (FQE) [118] which treats the gradient descent part in a full quantum mechanical manner.

VQE has been demonstrated on various quantum architectures such as superconducting qubits [46, 47, 119–125], photonic system [67, 126], and trapped ions [127–129]. These experiments are still boil down to proof-of-principle demonstrations on small-molecule systems, and thus further experimental efforts are required to scale up this approach to larger molecular systems of chemical interest. Meanwhile, the algorithmic approaches also needs to be further developed to relax the hardware limitation, such as reducing the number of qubits [69, 73, 105, 130–135] and circuit depth [73, 136–140], error-mitigation techniques [141–143], accelerated VQE [144], measurement-based VQE [145] and so on.

2. *Combinatorial optimization*

Combinatorial optimization is a promising path to demonstrating practical quantum advantage on near-term quantum devices, most notably the QAOA [65, 146] for solving the combinatorial optimization problem MaxCut. Max-Cut is the NP-complete problem, which is defined to partition the nodes of a graph into two distinct sets A and B that maximizes the number of edges connecting nodes in opposite sets. It has been shown that to achieve an approximation ratio of $r \geq 16/17 \approx 0.9412$ for Max-Cut on all graphs is NP-Hard [147].

Mathematically, consider a graph with m edges and n vertices, we can use bitstring $z = z_1 \ldots z_n$ to represent the assignment of vertices to the two sets, where $z_i = 0$ if i-th vertex is belongs to set A and $z_i = 1$ if it belongs to the other set B. The objective is to maximize the number of edges cut, denoted as $C(z)$. When implementing the QAOA to find approximate solutions to the MaxCut problem, we denote the bitstring using computational basis states $|z\rangle$, and define the objective function to maximize as

$$C(z) = \sum_{\alpha=1}^{m} C_\alpha(z) = \sum_{\text{edge}(j,k)}^{m} \frac{1}{2}(1 - \sigma_z^j \sigma_z^k)|z\rangle, \tag{7}$$

where (j, k) vertices connect the α edge. C_α has the value 1 only if the j-th and k-th qubits have different measurement results on the Z basis, representing separate partitions. The p-layer QAOA parameered circuit is usually governed by the problem Hamiltonian $H_C = \sum_{\text{edge}(j,k)}^{m} \frac{1}{2}(1 - \sigma_z^j \sigma_z^k)$ and a mixing Hamiltonian $H_M = \sum_{i=1}^{n} \sigma_x^i$ alternatively in each layer, as

$$U(\boldsymbol{\gamma}, \boldsymbol{\beta}) = \prod_{l=1}^{p} e^{-i\beta_l H_M} e^{-i\gamma_l H_C}, \tag{8}$$

where $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_p)$ and $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)$ are variational parameters. The input state of QAOA circuit is set as an n-qubit uniform superposition state $|+\rangle^n$, and the final state $\varphi(\boldsymbol{\gamma}, \boldsymbol{\beta}) = U(\boldsymbol{\gamma}, \boldsymbol{\beta})|+\rangle^n$ is measured to output the bitstring z. We train the QAOA parameered circuit to evolve from the initial state into the bitstring z with the maximum partition.

There has been a lot of theoretical works discussing the performance of QAOA and evaluating the resources needed to achieve quantum computational supremacy [57, 148–154]. In Ref. [149], Streif *et al.* presented a comparison between the QAOA with competing methods, quantum annealing and simulated annealing. Guerreschi *et al.* claimed that at least several hundreds of qubits are required for QAOA to achieve quantum speed-up [150]. Dalzell *et al.* concluded that 420 qubits QAOA would be sufficient for quantum computational supremacy [152]. Furthermore, it was proven that classical Goemans-Williamson algorithm outperforms the QAOA for certain instances of MaxCut at any constant level [155]. However, Bravyi *et al.* [153] and Egger *et al.* [154] suggested that higher-level Recursive-QAOA is competitive (and often better than) to best known generic classical algorithm based on rounding an SDP relaxation. Overall, whether QAOA has advantages over classical algorithms, and in which issues, is still a controversial issue that needs more research. Besides, to improve the performance of the QAOA, great efforts have been made, such as finding a good classical optimizer [151, 156–159], ansatz design [74, 160–162], reducing circuit depth [163, 164] and number of qubits [165, 166], and so on.

In terms of experimental progress, QAOA has been experimentally implemented in superconducting [56, 167], trapped-ion [168], and Rydberg atom [55] platform. In particular, in 2021, Harrigan *et al.* realized the QAOA with up to 23 qubits on the Google's Sycamore superconducting quantum processor [56]. Ebadi *et al.* experimentally investigated quantum algorithms for solving the maximum independent set problem using Rydberg atom arrays with up to 289 qubits in two spatial dimensions [55].

3. *Machine learning*

Nowadays, artificial intelligence and machine learning have become an integral part of modern life. As quantum computing promises to enhance our capacity to do some crucial computational tasks, it is natural to look for linkages between

these two cutting-edge techniques in the goal of gaining a variety of advantages. In fact, VQAs can be seen as quantum analogs of classical neural networks, and thus can easily be used for various machine learning tasks, such as classification and generative models.

QCNN. Convolutional neural networks (CNNs) stand out from other neural networks for their outstanding capabilities in computer vision [169–172], and natural language processing [173–178]. A CNN usually consists of three main types of layers: 1) convolutional layer, 2) pooling layer, and 3) fully-connected layer, where convolutional layers use filters that perform convolution operations on the input, and pooling layers are a type of downsampling operation.

Inspired by CNNs, Cong *et al.* proposed a quantum convolutional neural network (QCNN), and showed its ability to solve quantum many-body problems [48]. In the QCNN architecture, the convolutional and pooling layers, as well as the fully-connected layer, are designed as quantum circuits, although their actual functions do not correspond exactly to the classical ones. The analysis by Pesah *et al.* implied that QCNNs do not exhibit barren plateaus [179], which are important for trainability as the system scales. The QCNN has been realized on a 7-qubit superconducting quantum processor to identify symmetry-protected topological phases of a spin model [180]. QCNN is more suitable for quantum problems, in contrast, the hybrid quantum-classical convolutional neural network (QCCNN) [50] is specially proposed for real-world problems. The central idea of QCCNN is to implement the feature map in the convolutional layer with a PQC, while other layers remain the classical operation. Such a design not only introduces a quantum-enhanced feature map, but also preserves important features of classical CNN, such as non-linearity and scalability. Numerical experiments on the Tetris dataset show that QCCNN can accomplish classification tasks with learning accuracy surpassing that of classical CNN with the same structure. Besides, Wei *et al.* proposed a QCNN framework based on LCU [181, 182] that can transform CNN to QCNN directly [183].

Quantum GAN. Generative adversarial networks (GANs) are at the forefront of the generative learning and have been widely used for image processing, video processing, and molecule development [184]. GANs exploit a two-player minimax game between a generator and a discriminator, where generator aims to output the generated data to fool discriminator, and meanwhile, discriminator tries to distinguish the true example from generator.

Recently, theoretical works show that quantum generative models may exhibit an exponential advantage over classical counterparts [52, 185, 186], arousing extensive research interest in the theory and experiment of quantum GANs [54, 187–195]. In particular, Huang *et al.* developed a resource-efficient quantum GAN, and for the first time accomplished the generative task of real-world handwritten digit image on a superconducting quantum processor [53]. Being able to handle real-world data generation tasks is an exciting thing for an "infant" quantum computer. Subsequently, Rudolph *et al.* provided an experimental implementation of generating more high-resolution images of handwritten digits with quantum GAN on an ion-trap quantum computer [54]. Niu *et al.* showed that by learning a shallow quantum circuit to generate a superposition of classical data, their proposed entangling quantum GAN can be used to create an approximate quantum random access memory (QRAM) [193].

Quantum kernel estimation. Quantum Kernel estimation is a common method for supervised learning classification task on the NISQ device. In Ref. [196], Schuld *et al.* interpreted the process of encoding classical information into a quantum state as a nonlinear feature map which assigns data to the Hilbert space of the quantum system, and then proposed to use a variational quantum circuit to process the feature vectors. Havlíček *et al.* proposed and experimentally implemented binary classifiers that use the quantum state space as feature space on a superconducting processor [49], providing a possible path to quantum advantage.

Quantum auto-encoder. Quantum auto-encoder (QAE) is an efficient VQA for quantum data compression. QAE starts from a large-scale quantum state ρ_{AB}, utilizes a PQC $U(\boldsymbol{\theta})$ to encode ρ_{AB} into the A subsystem (latent space), and then recovers the initial state ρ_{AB} from this compressed state with high fidelity via a decoder constructed by another PQC $V(\boldsymbol{\theta'})$. Usually V is set as $V(\boldsymbol{\theta'}) = U(\boldsymbol{\theta})^\dagger$. QAE has recently attracted great attentions [197–202], and its experimental implementations have been carried out on linear optical systems [203, 204] and superconducting systems [205].

4. Cryptanalysis

Shor's integer factorization algorithm is considered to be one of the most influential algorithms that shape the research interest in quantum computing today. However, implementing Shor's algorithm requires a fault-tolerant quantum computer, which is far from attainable. Variational Quantum Factoring (VQF) algorithm is heuristic alternative to Shor's algorithm suitable for NISQ devices [206]. It works by reducing the factoring problem to the ground state of the Ising Hamiltonian, which can then be found using VQA. Karamlou *et al.* reported a experimental demonstrations of VQF using a superconducting quantum processor [207], the integers 1099551473989, 3127, and 6557 are factored with 3, 4, and 5 qubits, respectively. Besides, the performance and resource analysis of VQF are presented in Refs. [208, 209].

In addition to the factoring task, VQAs can also be used to attack advanced encryption standard (AES)-like symmetric cryptography [210]. In their simulation results, sometimes the variational quantum attack algorithm is even faster than Grover's algorithm. Moreover, Coyle *et al.* proposed cryptanalysis algorithm based on variational quantum cloning (VarQlone), which allows an adversary to obtain optimal approximate cloning strategies for unknown quantum states with shallow quantum circuits [211].

5. Mathematical applications

Numerical solvers such as solving linear systems, non-linear differential equations, *etc*, have a very wide range of engineering application scenarios. Several classical-quantum hybrid algorithms derived from VQAs make it possible for near-term quantum devices to solve large-scale numerical problems without the use of fault-tolerant quantum computing.

Variational quantum linear solver. Solving systems of linear equations is a fundamental computational problem, and the vast majority of numerical problems in science and engineering boil down to solving systems of linear equations. Unlike the Harrow-Hassidim-Lloyd (HHL) algorithm [212], the variational quantum linear solver (VQLS) proposed by Bravo-Prieto *et al.* can run on near-term quantum computers [213]. By employing a hybrid quantum-classical algorithm, the goal of VQLS is to variationally prepare $|x\rangle$ such that $A|x\rangle \propto |b\rangle$. The numerical simulations give the evidence that the run time of VQLS has linear scaling in κ, logarithmic scaling in $1/\varepsilon$, and polylogarithmic scaling in N, where κ is the condition number of a $N \times N$ matrix A, and ε is the precision of the solution. In the same period, Huang *et al.* [214] and Xu *et al.* [215] independently proposed similar quantum algorithms.

Solving nonlinear differential equations. Differential equations are ubiquitous in various fields of science and engineering. Lubasch *et al.* first extended the concept of VQAs to introduce nonlinearity for solving nonlinear problems [216], such as the nonlinear Schrödinger equation. Subsequently, several approaches to solve the nonlinear differential equation using the VQAs were proposed [217–220], and the application to computational fluid dynamics was studied [221].

6. Many-body physics

It is more natural to use quantum circuits to process quantum data than to process classical data, because it avoids the input and output problems of classical-quantum data conversion. VQAs enable us to realize the simulation of many-body dynamics [222, 223], as well as identifying many-body phase transition [58, 180, 224, 225]. In particular, Gong *et al.* proposed a quantum neuronal sensing based on digital-analog variational quantum circuit, and exprimentally realized this scheme to classify the ergodic and localized phases of matter on a 61 qubit superconducting quantum processor [58]. This experiment demonstrates the experimental feasibility of large-scale VQAs, and opens new avenues for exploring quantum many-body phenomena in larger-scale systems.

We noticed that the application fields of VQAs are far more than those listed above. It can also be applied to finding quantum error-correcting codes [226], entanglement purification [227], amplitude estimation [228], variational Inference [229], *etc.*, and we will not introduce them one by one.

C. Opportunities and Challenges

Due to their wide applications and noise resistance to NISQ devices, VQAs have a huge potential to achieve quantum advantages for practically interesting problems. Over the past few years, VQAs have received unprecedented attention, and papers related to VQAs appear on the arXiv website almost every day. The following points deserve our attention in advancing VQAs into future industrial applications or truly meaningful scientific research.

Where is the main advantage? Although VQAs have been hotly researched for several years, this core question has never been well answered. Perhaps using VQAs to handle quantum problems is a natural advantage [58], since either simulating quantum systems or measuring quantum systems using classical methods would be exponentially resource-intensive. However, for real-world problems, it becomes very tricky to answer why we need VQAs. Does it offer speed or accuracy advantages over classical neural networks? Maybe we can try to answer this question in two ways: 1) Find the provable advantages of VQAs theoretically. This should be difficult, since today's machine learning algorithms are still theoretically difficult to study; 2) Experimentally achieve milestones that beat current state-of-the-art classical machine learning in terms of speed or accuracy on large-scale practical datasets (rather than toy models). Either way, it will take a huge effort.

I/O problems for real-world data. To fully exploit the superposition properties of quantum systems, we actually want the classical data to be encoded into the amplitude of quantum system. However, this is extremely resource-intensive, whether using QRAM [230] or pure quantum circuits [231, 232]. Thus, for real-world applications, perhaps we need to find some specfic classical problems suitable for quantum computing. For example, the input to this classical problem has a certain sparsity or symmetry, which is conducive to being encoded as a quantum state. For the output, in general we need to avoid the exponential consumption caused by a large number of measurements, which requires us to extract and analyze only part of the information of the output quantum state, such as the expected value of observables, principal component analysis [233], *etc.*

Trainability and training efficiency. Avoiding barren plateaus is now an important area of research, as barren plateaus in training landscapes will appear if care is not taken enough. Some results suggest that the exponential parameter space, the noise and decoherence, and even entanglement can induce barren plateaus [234–237]. At present, some methods have discussed how to mitigate barren plateaus [63, 238–241], and more research is still needed.

One of the weaknesses of quantum computers compared to classical computers is quantum noise, which also greatly affects the training performance. Generally, the resillience of VQAs exists for a wide class of noise, as analyzed in Refs. [62, 242, 243]. Yet such noise resilience is highly limited as the noise level rising. As suggested in Refs. [235, 244–247], large noise may leads to problems such as performance degradation or barren plateaus. Some special noises, such as

leakage error, also have a bad effect on the performance of VQAs [248]. Thus, the analysis of iteration complexity and behaviour with noise is necessary [237], and efficient error mitigation schemes for VQAs need to be developed [249–252].

Besides, the training process of VQA is very time-consuming, due to the incompatibility with backpropagation and the cost of a large number of measurements, posing a great challenge to the large-scale development of VQAs. Some approaches try to use multiple QPUs for parallel training to alleviate this deficiency, such as data parallelism [253] or parameter parallelism [254]. However, more fundamental solutions are urgently needed.

Hardware development. For practical NISQ applications, high-quality quantum processors with more qubits, longer coherence times and lower error rates are prerequisites. For VQAs, iterative training requires frequent interaction between classical and quantum computers, which imposes some new demands on the quantum hardware. The cryogenic electronic control architecture is a hardware-level solution for accelerating classical-quantum interactions [255–259], and also an advanced technique required for the future development of quantum computing.

III. QUANTUM ERROR MITIGATION

In the context of current NISQ devices, the impact of noise remains the greatest challenge for the practical applications [6]. Although quantum error correction (QEC) promises to enable quantum computation with arbitrary levels of noise, it is out of reach for near-term quantum processors. Quantum error mitigation (QEM) provides us a feasible alternative to mitigate errors of near-term quantum processors, and is also the continuous path that will take us from today's quantum hardware to tomorrow's fault-tolerant quantum computers. Instead of active error-correction, QEM methods usually estimate the error-free expectation value by classical post-processing of the noisy measurement results. While it introduces the additional sampling overhead caused by the increase in the variance of the mitigated observable, QEM requires fewer qubits and gate resources and is therefore more suitable for practical NISQ devices. In recent years, a wide variety of QEM methods have been proposed [35, 141, 142, 260–263], and we will discuss some typical QEM techniques in the following subsections.

A. Probabilistic error cancellation

We first introduce probabilistic error cancellation (PEC) [141, 264]. The key idea of PEC is to apply the quasi-probability decomposition of the inverse noise process, leading to a linear combination of a set of noisy circuits. The implementation of PEC requires the full knowledge of the noise model in the target circuit and is usually combined with characterization of the noise process. The PEC method was first introduced by Temme *et al.* [264] and further detailed for a practical scheme by Endo *et al.* [141]. It has been shown

to mitigate Markovian noise [141] and has been further generalized to the non-Markovian noise [265]. And it was experimentally demonstrated on superconducting [266] and trapped ion [267] quantum computers.

1. Standard PEC with discrete gate-based circuits

Quasi-probability decomposition of inverse noise model. Suppose we have a noisy quantum gate $\mathcal{E} \circ \mathcal{U}$ contaminated by a noise channel $\mathcal{E}$, where $\mathcal{U}(\rho) = U\rho U^\dagger$ denotes the noiseless gate with the ideal operator U and the initial quantum state ρ. With approaches to accurately characterize the noise channel $\mathcal{E}$, we can have the mathematical form of the inverse noise channel $\mathcal{E}^{-1}$, and then apply $\mathcal{E}^{-1}$ after the noisy gate $\mathcal{E} \circ \mathcal{U}$ to recover the ideal gate $\mathcal{U} = \mathcal{E}^{-1} \circ \mathcal{E} \circ \mathcal{U}$. Since the inverse noise channel $\mathcal{E}^{-1}$ may be unphysical, we need a set of basis operations $\{\mathcal{B}_k\}_k$ to decompose it as

$$\mathcal{E}^{-1} = \sum_k q_k \mathcal{B}_k, \tag{9}$$

where q_k is the the combination coefficients, *i.e.*, quasi-probabilities. So we can estimate the noiseless expectation value of the observable O using a linear combination of results obtained from circuits applied by different basis operations

$$\begin{aligned}\langle O \rangle_{\text{PEC}} &= \sum_k q_k \text{Tr}[\mathcal{B}_k \circ \mathcal{E} \circ \mathcal{U}(\rho)] \\ &= Q_\mathcal{E} \sum_k \text{sgn}(q_k)\frac{|q_k|}{Q_\mathcal{E}}\text{Tr}[\mathcal{B}_k \circ \mathcal{E} \circ \mathcal{U}(\rho)],\end{aligned} \tag{10}$$

where $Q_\mathcal{E} = \sum_k |q_k|$. Given above quasi-probability decomposition, we can use the Monte Carlo sampling to randomly apply the basis operation set $\{\mathcal{B}_k\}_k$ with the quasi-probability distribution $\{\frac{|q_k|}{Q_\mathcal{E}}\}$. Then we multiply the measurement outcome by the corresponding $\text{sgn}(q_k)$ and weight the Monte Carlo average by $Q_\mathcal{E}$. For the general circuit composed of a N-gate sequences $\mathcal{U} = \prod_{n=1}^N \mathcal{U}_n$, we can just find the quasi-probability decomposition for noise process $\mathcal{E}_n$ associated with each gate

$$\mathcal{E}_n^{-1} = \sum_{k_n} q_{k_n}\mathcal{B}_{k_n} = Q_n \sum_{k_n} \text{sgn}(q_{k_n})\frac{|q_{k_n}|}{Q_n}\mathcal{B}_{k_n}, \tag{11}$$

in which $Q_n = \sum_k |q_{n_k}|$.

Then we can obtain the error mitigated value in a similar way

$$\begin{aligned}\langle O \rangle_{\text{PEC}} &= \text{Tr}\left[\prod_n^N \left(\sum_{k_n} q_{k_n}\mathcal{B}_{k_n} \circ \mathcal{E}_n \circ \mathcal{U}_n\right)(\rho)\right] \\ &= Q_\mathcal{E} \sum_{\vec{k}} \text{sgn}(q_{\vec{k}})\frac{|q_{\vec{k}}|}{Q_\mathcal{E}}\text{Tr}\left[\prod_{n=1}^N (\mathcal{B}_{k_n} \circ \mathcal{E}_n \circ \mathcal{U})(\rho)\right],\end{aligned} \tag{12}$$

with $\vec{k} = (k_1, k_2, ..., k_N)$ for $q_{\vec{k}} = \prod_{n=1}^N q_{k_n}$ and $Q_\mathcal{E} = \prod_{n=1}^N Q_n$.
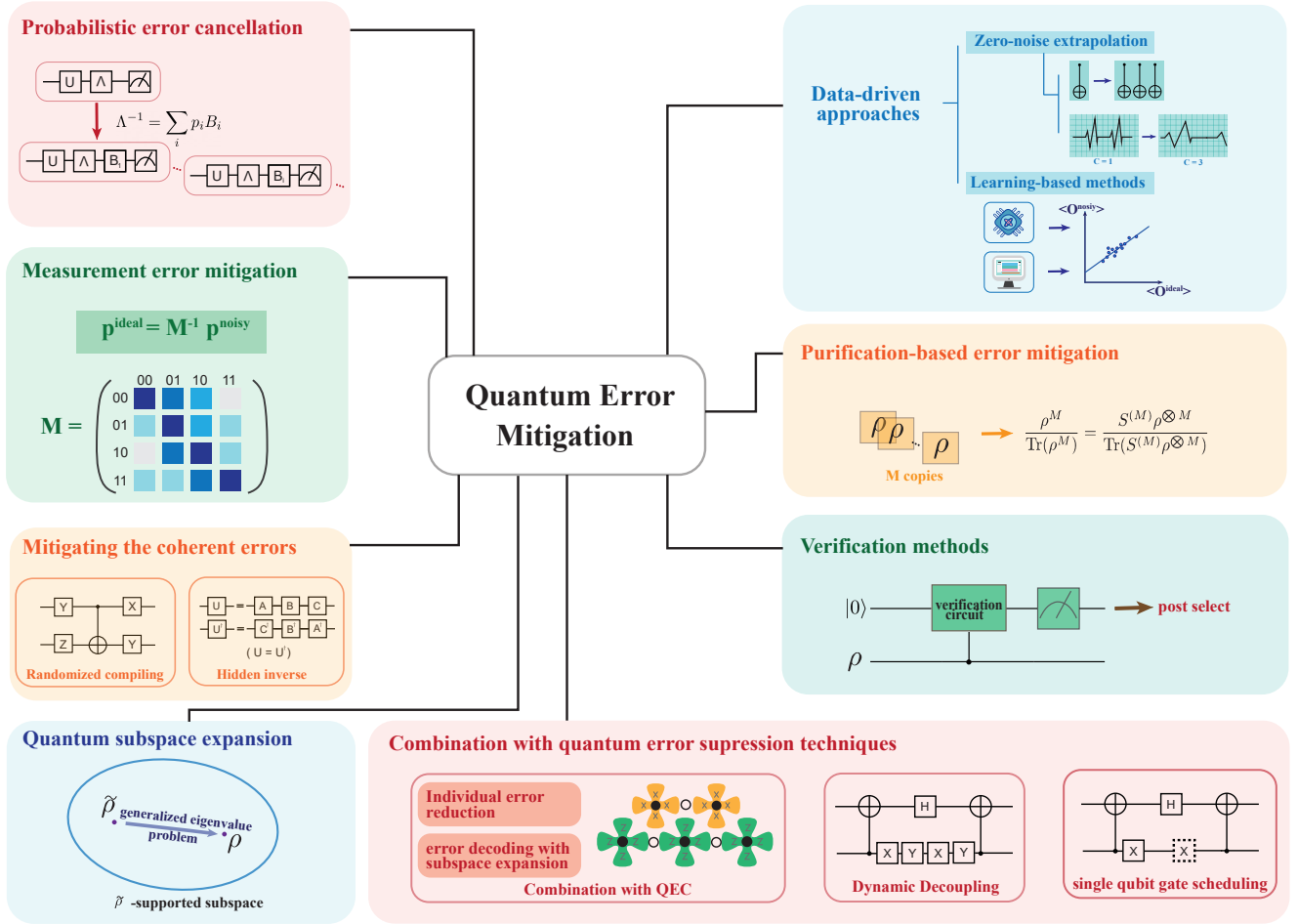
FIG. 3. **Summary of popular Quantum Error Mitigation(QEM) methods.** Probabilistic error cancellation estimates the noiseless expectation value using a linear combination of expectation values with different noise terms, which is based on the quasi-decomposition. The data-driven approaches including Zero-noise extrapolation and learning-based methods collect expectation values of circuits with different error rates, or circuits with similar structure. Purification-based error mitigation schemes use multiple (M) copies of the noisy state and estimate $\frac{\mathrm{Tr}(\rho^M O)}{\mathrm{Tr}(\rho^M)}$ using collective measurements. Verification methods are designed for systems with certain symmetries and combined with classical post-processing. Quantum subspace expansion also uses post-processing to mitigate errors for some VQE algorithms. Other mitigation methods for certain noise types, such as measurement error mitigation (for measurement errors) are also widely applied.

Sampling overhead. Note that the total resource overhead for PEC involves both the characterization cost of the noise channel, and the sampling overhead to estimate the linear combinations in Eq 10. Considering that the gate noise characterization is usually performed during the device calibration stage [268], we here focus on the sampling overhead for the implementation of PEC.

As for the factor $Q_{\mathcal{E}}$ multiplied with the averaged expectation value, the variance is increased by $Q_{\mathcal{E}}^2 = (\prod_{n=1}^{N} Q_n)^2$. Therefore, it needs $Q_{\mathcal{E}}^2$ times more samples to achieve the same measurement accuracy as the unmitigated case. The square of the multiplier $Q_{\mathcal{E}}^2$ can be regarded as the sampling overhead Υ_{PEC} for PEC, which grows exponentially with the overall gate error εN if we assume $Q_n = 1 + \mathcal{O}(\varepsilon)$ for stochastic noise process [141]. This sets limitations on the

efficiency of PEC schemes to $\varepsilon N = \mathcal{O}(1)$. Moreover, the theoretical analysis of the sample complexity for implementing PEC has been explored [269, 270] and it has been shown the optimality of PEC among all strategies in mitigating a certain type of noise [270].

Approaches to extract error parameters. To implement the PEC in practice, we need to accurately characterize the noise model $\mathcal{E}$ and obtain the analytical decomposition of the inverse noise map as Eq. 9 shown.

Gate set tomography. Endo *et al.* [141] has introduced gate set tomography (GST) [271, 272] to PEC, which is free from the SPAM errors. Although GST can reconstruct arbitrary noise process, the number of samples required for its reconstruction grows exponentially with the size of the system N_q and is quite costly. If only specific types of noise are considered, there are some more efficient characterization proto-

cols [273–275].

Cycle error reconstruction for Pauli noise. Cycle error reconstruction (CER) is an efficient approach to identify the Pauli noise in circuits [276, 277]. It utilizes Cycle Benchmarking [278] to characterize the Pauli noise channel and extract the Pauli error rates after post-processing. The implementation of PEC with CER has been shown on a 4-qubit superconducting processor [279]. Under the Pauli noise assumption, the quasi-probability decomposition can be further simplified using sparse Pauli-Lindblad models [280]. The sparse Pauli noise model considers Pauli operators with only one or two non-trivial terms, but is sufficient to capture the correlated errors. Moreover, the number of noise parameters in the sparse model scales polynomially with the system size, so it remains efficient in large-scale quantum systems. This approach has been experimentally demonstrated to learn the noise model on a superconducting quantum processor of up to 20 qubits [280].

2. PEC with continuous time evolutions

The standard PEC scheme we discussed above is based on the discrete gate-based quantum circuits with the gate-independent Markovian noise. To overcome this limitation, the stochastic QEM method [281] is proposed to extend the standard PEC to more general scenarios that may have strong gate-dependence and complicated nonlocal effects, and general computing models such as analog quantum simulators.

Specifically, the time evolution of a noisy quantum system can be described by the Lindblad master equation as

$$\frac{\partial \rho(t)}{\partial t} = -i[H^{\text{noisy}}(t), \rho(t)] + \lambda \mathcal{L}[\rho(t)],$$
$$\mathcal{L}(\rho) = \sum_i (2L_i \rho L_i^\dagger - L_i^\dagger L_i \rho - \rho L_i^\dagger L_i) \tag{13}$$

where $H^{\text{noisy}} = H + \delta H_0$ (H is the ideal evolution Hamiltonian, H_0 denotes the coherent errors), and $\mathcal{L}(\rho)$ is the noise Lindblad operator corresponding to the decoherent coupling with the environment with noise strength λ. We denote the ideal and noisy process by $\mathcal{E}_I$ and $\mathcal{E}_N$ respectively. Given a small time step δt with a time interval δ, both the ideal and noisy case of the evolution can be written as

$$\rho_i(t + \delta) = \mathcal{E}_i(t)\rho_i(t), \tag{14}$$

with $i = I, N$. Similar to the idea of PEC in Eq. 9, we can find a recovery process $\mathcal{E}_Q$ satisfying

$$\mathcal{E}_I = \mathcal{E}_Q \mathcal{E}_N + \mathcal{O}(\delta t^2) \approx \mathcal{E}_Q \mathcal{E}_N,$$
$$\mathcal{E}_Q = \sum_k q_k \mathcal{B}_k = Q \sum_k \text{sgn}(q_i) \frac{|q_i|}{Q}, \tag{15}$$

with $Q = \sum_k |q_k|$. Given the full evolution time T, we can apply the Monte Carlo sampling to realize the continuous recovery $\mathcal{E}_Q$ with a time interval δ at the mitigation cost of $\Upsilon_{\text{SEM}} = Q^{\frac{T}{\delta t}}$ [142, 281].

B. Data-driven approaches

We now introduce the data-driven approaches for error mitigation, including zero noise extrapolation (ZNE) and error mitigation with learning process such as Clifford data regression (CDR). These methods utilize different types of data from circuits with various error rates (*e.g.*, ZNE), or from near-Clifford circuits under similar circuit structure (*e.g.*, CDR), and they can be naturally combined with each other to better utilize data resources [282, 283].

1. Zero-Noise Extrapolation

ZNE is a typical error mitigation scheme using classical post-processing. It uses data collected at different error rates to fit the function of expectation values with respect to the error rates, and then extrapolate to the zero noise limit [141, 262–264, 282, 284, 285].

Fitting methods of ZNE. The chosen fitting method is essential to the performance of ZNE. A basic fitting method is the Richardson extrapolation, which utilizes the polynomial relationship between noisy and ideal expectation values yielded by Tailor expansion when the error rate is low. This estimation is slightly coarse. More refined models, such as exponential and poly-exponential fitting functions, can be introduced in the extrapolation method to improve the actual mitigation performance in specific cases. We discuss these fitting methods in detail.

Richardson extrapolation. Richardson extrapolation was the first introduced extrapolation method [263, 264] and later experimentally demonstrated using superconducting qubits [120].

Suppose the quantum circuit outputs a noisy N_q-qubit quantum state ρ_ε, where ε denotes the noise parameter which characterizes the error rate in the circuit. Then the expectation value of the target observable O under error rate ε can be regarded as a function towards ε: $\langle O \rangle(\varepsilon) = \text{Tr}(\rho_\varepsilon O)$. In order to evaluate the error-free expectation value $\langle O \rangle(0)$, we choose a set of error rates $\{\lambda_0 \varepsilon, ..., \lambda_n \varepsilon\}$ with $n + 1$ different coefficients $\{\lambda_i\}_{i=0,...,n}(\lambda_0 = 1)$, and then run circuits under these error rates to obtain the set of $n + 1$ expectation values $\{\langle O \rangle(\lambda_i \varepsilon)\}_i$. It is known to be difficult to reduce the error rate in the circuit, we thus usually boost it with amplified coefficients $1 = \lambda_0 < \lambda_1 < ... < \lambda_n$ using some noise scaling methods. Linear spacing of the amplified coefficients (*e.g.* $\{\lambda_i = i + 1\}_{i=0,...,n}$) is commonly used [264, 268] and some specific spacing methods have been explored for better performance [286].

The expectation value for $\lambda = 0$ (the error-free case) can be estimated using

$$\langle O \rangle_{\text{Rid}} = \sum_{i=0}^{n} \gamma_i \langle O \rangle(\lambda_i \varepsilon), \tag{16}$$

where the fitting coefficients $\{\gamma_i\}$ are chosen to satisfy both $\sum_{i=0}^{n} \gamma_i = 1$ and $\sum_{i=0}^{n} \gamma_i \lambda_i^j = 0$ for $j = 1, ..., n$, and the

solution gives

$$\gamma_i = \prod_{i \neq j} \frac{\lambda_j}{\lambda_j - \lambda_i}. \tag{17}$$

If the error rate ε is sufficiently low, we can expand the function $\langle O \rangle(\varepsilon)$ according to the Taylor expansion

$$\langle O \rangle(\varepsilon) = \langle O \rangle(0) + \sum_{j=1}^{n} a_j \varepsilon^j + \mathcal{O}(\varepsilon^{n+1}) \tag{18}$$

with some constant coefficients a_j. Then the Richardson mitigated value $\langle O \rangle_{\mathrm{Rid}}$ can be rewritten using Eq. 16 and 18

$$\langle O \rangle_{\mathrm{Rid}} = \sum_{i=0}^{n} \gamma_i \left(\langle O \rangle(0) + \sum_{j=1}^{n} a_j \lambda^j \varepsilon^j + \mathcal{O}(\varepsilon^{n+1}) \right)$$

$$= \langle O \rangle(0) + \sum_{j=1}^{n} \left(\sum_{i=0}^{n} \gamma_i \lambda_i^j \right) \varepsilon^j + \mathcal{O}(\varepsilon^{n+1})$$

$$= \langle O \rangle(0) + \mathcal{O}(\varepsilon^{n+1}),$$

Here we use $\sum_{i=0}^{n} \gamma_i = 1$ and $\sum_{i=0}^{n} \gamma_i \lambda_i^j = 0$ for $j = 1, ..., n$. We can see that the error rate is suppressed from the original ε to the order $\mathcal{O}(\varepsilon^{n+1})$, under the weak noise assumption. However, the assumption of a valid Taylor expansion for low error rate may be inaccurate in the large circuit limit, and the lack of specific information on the noise channel sets limitation to the efficacy of Richardson extrapolation.

As for the sampling overhead of Richardson extrapolation, we need to take into account the variance increase introduced in Eq. 16, which reads [142]

$$\mathrm{Var}(\langle O \rangle_{\mathrm{Rid}}) = \sum_{i=0}^{n} \gamma_i^2 \mathrm{Var}(\langle O \rangle_{\lambda_i \varepsilon}). \tag{19}$$

The variance of estimating $\langle O \rangle_{\mathrm{Rid}}$ is about $\sum_{i=0}^{n} \gamma_i^2$ larger than evaluating $\langle O \rangle$ without error mitigation, so it requires around $\Upsilon_{\mathrm{Rid}} = \sum_{i=0}^{n} \gamma_i^2$ more samples as the sampling overhead of Richardson extrapolation.

Besides, an optimized protocol [286] for the implementation of Richardson extrapolation has been proposed to further explore the relevant parameters of Richardson extrapolation.

Exponential Extrapolation. The error rate ε used in Richardson extrapolation quantifies the local noise strength, but in the context of NISQ error mitigation, it is more natural to consider the mean circuit error count [262] μ, where $\mu \approx N\varepsilon$ and N denotes the number of gates in the circuit. In the large circuit limit when $N \gg 1$, the number of errors happened in the circuit (denoted by k) follows the Poisson distribution with probability [262]

$$p_k = e^{-\mu} \frac{\mu^k}{k!}. \tag{20}$$

Denoting the expectation value with k errors occurred as $\langle O_k \rangle$, the expectation value with mean circuit error count μ is

$$\langle O_\mu \rangle = \sum_{k=0}^{\infty} p_k \langle O_k \rangle = e^{-\mu} \sum_{k=0}^{\infty} \frac{\mu^k}{k!} \langle O_k \rangle, \tag{21}$$

where the factor $e^{-\mu}$ implies an exponential decay with μ of the expectation value. By simply assuming an exponential function

$$\langle O \rangle(\mu) = e^{-f\mu} \langle O \rangle(0) \tag{22}$$

with a parameter f which denotes the observable decay rate, we can then obtain an two-points exponential extrapolation result with mean circuit error count μ and $\lambda\mu$ ($\lambda > 1$)

$$\langle O \rangle_{\exp} = \left(\frac{\langle O \rangle^\lambda(\mu)}{\langle O \rangle(\lambda\mu)} \right)^{\frac{1}{\lambda-1}}, \tag{23}$$

Endo *et al.* [141] first considered the exponential decay curve for error extrapolation, where the advantage of exponential extrapolation over Richardson extrapolation has been numerically demonstrated [141, 284]. And Cai [262] provided a general multi-exponential extrapolation framework

$$\langle O \rangle(\mu) = \sum_{i=1}^{\infty} A_i (e^{-f_i})^\mu, \text{ and } \sum_{i=1}^{\infty} A_i = \langle O \rangle(0), \tag{24}$$

which achieves a much lower estimation bias in numerical simulation.

Poly-exponential extrapolation. The poly-exponential extrapolation [284, 287] assumes a more general model that the exponential decay with the mean circuit error count μ has a polynomial expansion which can be fitted to the function

$$\langle O \rangle(\mu) = e^{\sum_{i=0} f_i \mu^i} \langle O \rangle(0). \tag{25}$$

Then the single-exponential extrapolation modeled by Eq. 22 is a particular case of the more general poly-exponential extrapolation.

Noise scaling methods. The implementation of ZNE, especially Richardson extrapolation, depends non-trivially on the ability to boost the error rate ε in the circuit by different amplified coefficients$\{\lambda_i\}_i$ [286]. Next we will introduce a variety of methods to amplify the error rate in a controlled manner.

Identity insertion [141, 288] is a hardware-agnostic approach which replaces a particular gate or layer $\mathcal{C}$ by $\mathcal{C}\{\mathcal{C}^\dagger\mathcal{C}\}^n$ for non-negative integer n. The circuit after insertion is logically equivalent to the original circuit but the error rate is amplified by a factor $2n + 1$ as the circuit depth increases. There are many variant identity insertion methods proposed [289, 290] for ZNE, which explore the trade-off between the inserted gate number and the required measurement to achieve the same (or higher) accuracy. And more general approaches using unitary folding can obtain an arbitrary real scaling factor [284]. The limitation of identify insertion method is that the amplified error rate may arbitrarily deviate from the desired one [291]. Given a self-adjoint noisy gate $\tilde{\mathcal{U}} = \mathcal{E} \circ \mathcal{U}$, after identity insertion of $\tilde{\mathcal{U}}^\dagger\tilde{\mathcal{U}} = \tilde{\mathcal{U}}\tilde{\mathcal{U}}$ we have $\tilde{\mathcal{U}}\tilde{\mathcal{U}}^\dagger\tilde{\mathcal{U}} = \mathcal{E}(\mathcal{U}\mathcal{E}\mathcal{U})\mathcal{E} \circ \mathcal{U} = \tilde{\mathcal{E}} \circ \mathcal{U}$. The actually amplified noise $\tilde{\mathcal{E}} = \mathcal{E}(\mathcal{U}\mathcal{E}\mathcal{U})\mathcal{E}$ is gate-dependent for general noise channel $\mathcal{E}$, leading to an unpredictable amplified error rate in the circuit.

Stretching the control pulses in time [120, 264, 291] is also used to implement circuits under different error rate. Given an

open quantum system described by the time-dependent multi-qubit Hamiltonian $H(t)$, the time evolution is

$$\frac{\partial}{\partial t}\rho_\varepsilon(t) = -i[H(t), \rho_\varepsilon(t)] + \varepsilon\mathcal{L}(\rho_\varepsilon(t)), \qquad (26)$$

where ε refers to the error rate in the system, and the noise term $\mathcal{L}$ is a Lindblad operator. Then we rescale the Hamiltonian $H(t)$ to $\frac{1}{\lambda}H(\frac{t}{\lambda})$ with a stretched factor $\lambda > 1$ and obtain a new state $\rho'_\varepsilon(t)$. Under the assumption that the generator $\mathcal{L}$ is invariant after time rescaling, and also independent from the way the Hamiltonian $H(t)$ is encoded, we can increase the evolution time by a factor λ. Then the error rate is boosted from ε to $\lambda\varepsilon$ according to $\rho'_\varepsilon(ct) = \rho_{\lambda\varepsilon}(t)$. This method has been experimentally demonstrated on superconducting processors [120, 291] using up to 26 qubits [291].

Note that two methods above do not require the exact form of the noise channel, and there are some approaches that utilize the specific structure of certain noise models. *Parameter noise scaling* [284] is designed for the pulse error caused by imperfect control or finite precision of the physical parameters in the parametric quantum gates. Considering a quantum gate $G(\vec{\theta}) = \exp(-i\sum_{j=1}^{k}\theta_j H_j)$ which is parameterized by k classical control parameters $\vec{\theta} = (\theta_1, \theta_2, ..., \theta_k)$ with Hamiltonian operators $H_1, H_2, ..., H_k$, the actually implemented gate suffered from pulse error is modeled by $G(\vec{\theta'})$ with

$$\theta'_j = \theta_j + \hat{\varepsilon}_j, \text{ for } j = 1, 2, ..., k, \qquad (27)$$

where $\hat{\varepsilon}$ is a random variable following Gaussian distribution with zero mean and variance σ_j^2 which denotes the effect of the stochastic calibration noise. Given the value of variance σ_j^2, we can rescale the noise by a factor $\lambda > 1$ using

$$\theta_j^{re} = \theta_j + \hat{\delta}_j, \qquad (28)$$

here the $\hat{\delta}_j$ is sampled from a zero-mean Gaussian distribution with variance $(\lambda - 1)\sigma_j^2$. Therefore, the effective parameter after rescaling is

$$\theta'_j = \theta_j + \sqrt{\lambda}\hat{\varepsilon}_j, \text{ for } j = 1, 2, ..., k, \qquad (29)$$

which achieves a noise rescaling by a factor λ without knowledge of the Hamiltonian operators H_j.

Pauli twirling techniques [292, 293] focus on the cases that error rates of single-qubit gates are much lower than those of two-qubit gates and measurement. By applying randomly chosen Pauli gates before and after the Clifford two-qubit gates, arbitrary noise channel can be converted into an effective stochastic Pauli channel [276, 277]. Then we can use additional Pauli gates to tune the practical error rate to any target value [263], but it requires the full knowledge of the exact Pauli noise channel. We can also combine efficient Pauli error reconstruction methods such as Cycle Error Reconstruction [277, 278] to amplify the error rate for ZNE [279].

Gate Trotterization [294] is a local noise scaling technique acting at the level of individual gates. We can replace each gate U of the circuit with the product of λ equal gates using the gate Trotterization technique

$$U \rightarrow (U^{\frac{1}{\lambda}})^\lambda, \ \lambda = 0, 1, 2... . \qquad (30)$$

However, the way the $U^{\frac{1}{\lambda}}$ is compiled by the hardware depends on λ, so the circuit depth may not increase by the expected factor.

Leveraging other error mitigation schemes as a noise scaling tool has also been explored for ZNE [262, 295]. Different from above tuning approaches which produces boosted error rate, we can also utilize other error mitigation methods such as PEC to reduce the error rate [262].

2. Multi-dimensional variant of ZNE

Least square fitting [296] may be regarded as a multi-dimensional version but not reliant on the Richardson extrapolation. It utilizes a hypersurface fit where one axis refers to the measurement results and the other axes describe the effect of difference noise parameters. The least square fitting method has been experimentally demonstrated on Rigetti's 8-qubit quantum processor [296].

Considering only one noise parameter ε with k different noise level $\varepsilon_1, \varepsilon_2, ..., \varepsilon_k$, the polynomial model up to n order in Eq. 18 can be generalized to a linear equation

$$\begin{pmatrix} 1 & \varepsilon_1 & \cdots & \varepsilon_1^n \\ 1 & \varepsilon_2 & \cdots & \varepsilon_2^n \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \varepsilon_k & \cdots & \varepsilon_k^n \end{pmatrix} \begin{pmatrix} \langle O\rangle(0) \\ \alpha_1 \\ \vdots \\ \alpha_n \end{pmatrix} = \begin{pmatrix} \langle O\rangle(\varepsilon_1) \\ \langle O\rangle(\varepsilon_2) \\ \vdots \\ \langle O\rangle(\varepsilon_k) \end{pmatrix}. \qquad (31)$$

We can solve this linear equation using standard least-squares methods and obtain the expansion parameters $\left(\langle O\rangle(0), \ \alpha_1, \ \cdots, \ \alpha_n \right)^T$, where $\langle O\rangle(0)$ is the error-free expectation value of the observable.

Then we can extend the consideration of one noise parameter to the multiple noise parameters. Take the spontaneous emission rate γ and the pure dephasing rate ζ for example, suppose we have k different noise level for both γ and ζ which are denoted by $\{\gamma_1, \gamma_2, ..., \gamma_k\}$ and $\{\zeta_1, \zeta_2, ..., \zeta_k\}$, respectively. The linear equation in Eq. 31 turns to

$$\begin{pmatrix} 1 & \gamma_1 & \zeta_1 & \gamma_1\zeta_1 & (\gamma_1)^2 & (\zeta_1)^2 \\ 1 & \gamma_2 & \zeta_2 & \gamma_2\zeta_2 & (\gamma_2)^2 & (\zeta_2)^2 \\ \vdots & \vdots & \ddots & \vdots & & \\ 1 & \gamma_k & \zeta_k & \gamma_k\zeta_k & (\gamma_k)^2 & (\zeta_k)^2 \end{pmatrix} \begin{pmatrix} \langle O\rangle(0) \\ \alpha_1 \\ \vdots \\ \alpha_5 \end{pmatrix} = \begin{pmatrix} \langle O\rangle(\gamma_1, \zeta_1) \\ \langle O\rangle(\gamma_2, \zeta_2) \\ \vdots \\ \langle O\rangle(\gamma_k, \zeta_k) \end{pmatrix}.$$
$$(32)$$

Here we truncate the series to the second order of γ and ζ.

3. Error mitigation with learning process

Instead of using the exact knowledge of the noise channel (such as PEC) or the expectation values at different error rates (like ZNE), the learning-based error mitigation optimizes the estimator of the observable's expectation value using automatical learning process [297–304]. Specifically, the learning-based error mitigation methods estimate the observable's expectation value of the targeted circuit using an ansatz that typically describes the relationship between the noisy and noiseless case of the expectation value. The training set fed into the

ansatz comes from classically simulable quantum circuits, or measurement results from relevant quantum circuits.

Clifford data regression [297] generates training data set $\{O_i^{\text{noisy}}, O_i^{\text{ideal}}\}_i$ composed of the noisy expectation value O_i^{noisy} from real quantum devices and the noiseless expectation value O_i^{ideal} simulated on the classical computers. To enable efficient classical simulation, the quantum circuits for generating training set are replaced with near-Clifford circuits generated by Markov Chain Monte Carlo sampling method. We can fit the training data set to a linear ansatz

$$O^{\text{ideal}} = f(O^{\text{noisy}}, \vec{\theta}) = \theta_1 O^{\text{noisy}} + \theta_2, \qquad (33)$$

where the parameters $\vec{\theta} = (\theta_1, \theta_2)$ are optimized by minimizing the cost function

$$C = \sum_i \left(O_i^{\text{ideal}} - (\theta_1 O_i^{\text{noisy}} + \theta_2) \right)^2. \qquad (34)$$

CDR has been experimentally demonstrated on a 16-qubit IBMQ quantum computer and can achieve an order-of-magnitude improvement for a ground state energy problem [297], and it also been shown to has the potential to outperform other state-of-the-art approaches [283]. CDR can be further improved by more efficient training set construction methods and by applying symmetries in certain quantum systems [305].

Learning-based probabilistic error cancellation [298] does not depend on the full tomography of the noise channel as the original PEC does. For a L-layer quantum circuit consisting of single-qubit unitary gates $\mathbf{R} = (R_1, R_2..., R_{N_q(L+1)})$ and multi-qubits Clifford gate layers $\mathbf{U} = (U_1, U_2, ..., U_L)$, the learning-based PEC applies the Pauli gates $\mathbf{P} = (P_1, P_2, ..., P_{2N_q(L+1)})$ as the error mitigating gates before and after each single-qubit unitary gates. It can then establish a circuit configuration where the multi-qubit Clifford gates are fixed, the other gates ($\mathbf{R}$ and $\mathbf{P}$) are served as variables, and $\mathbf{P} = \mathbf{I}$ indicates that all error mitigating gates are identity gates. Here we denote the noisy and error-free expectation value by $f(\mathbf{R}, \mathbf{P})$ and $f^{\text{ef}}(\mathbf{R}, \mathbf{P})$. According to the PEC, the error-mitigated expectation value $f^{\text{em}}(\mathbf{R}, \mathbf{I})$ is

$$f^{\text{em}}(\mathbf{R}, \mathbf{I}) = \sum_{\mathbf{R}} q(\mathbf{P}) f(\mathbf{R}, \mathbf{P}), \qquad (35)$$

where $q(\mathbf{P})$ is the the combination coefficients, *i.e.*, quasiprobabilities. By minimizing the cost function,

$$C(\mathbf{R}) = \frac{1}{|\mathbb{S}|} \sum_{\mathbf{R} \in \mathbb{S}} |f^{\text{em}}(\mathbf{R}, \mathbf{I}) - f^{\text{ef}}(\mathbf{R}, \mathbf{I})|^2, \qquad (36)$$

where a subset of single-qubit Clifford gates $\mathbb{S}$ is chosen as the training set so that the error-free expectation value $f^{\text{ef}}(\mathbf{R}, \mathbf{I})$ can be efficiently simulated, we can obtain a optimal distribution $q(\mathbf{P})$ for the training set $\mathbb{S}$ which can also be applied for the non-Clifford single-qubit gates. This scheme requires that all the single-qubit gates in the configuration are error-free and

do not rely on the exact error model of the multi-qubit Clifford gates. For practical reasons, since the spaces of $\mathbf{R}$ and $\mathbf{P}$ grow exponentially with the system size N_q, we need to truncate the spaces of the training set and error mitigating gates, or resort to variational optimization approaches [298].

Deep learning method [299, 300, 306] trains a deep neural network to model a noise channel utilizing the "black box" nature of the neural network. The training of the neural network requires the measurement results with both noise and ideal cases. However, since classical simulations may not be possible for large-scale non-Clifford quantum circuits, the training set can only consist of measurement results of specific quantum circuits whose ideal measurement outcomes are known [299, 306].

C. Measurement error mitigation

Measurement (or readout) error mitigation (MEM) schemes are designed to improve the accuracy of the measurement results obtained from noisy quantum devices.

1. MEM under classical noise assumption

Under the assumption of a classical noise model, MEM is usually applied via classical post-processing, where measurement error is modeled by a stochastic and invertible response matrix Λ [307–309].

Classical noise models. The ideal quantum measurement in the computational basis can be written in terms of positive operator valued measurement (POVM). For a n-qubit system ρ, suppose the POVM operator is $\Pi_{\boldsymbol{x}} = |\boldsymbol{x}\rangle\langle\boldsymbol{x}|$ satisfying $\Pi_{\boldsymbol{x}} \geq 0$ for $\forall \boldsymbol{x}$ and $\sum_{\boldsymbol{x}} \Pi_{\boldsymbol{x}} = \mathbb{I}$, where $\boldsymbol{x} \in \mathbb{Z}_2^{N_q}$ refers to the POVM outcome. The probability distribution of the measurement outcome is represented by a vector $\vec{p}$, where the $\boldsymbol{x}$ term of $\vec{p}$ is the probability of obtaining the outcome $\boldsymbol{x}$, given by $p(\boldsymbol{x}|\rho) = \text{Tr}(\rho\Pi_{\boldsymbol{x}})$ according to the Born's rule.

If the POVM elements have no non-trivial off-diagonal terms, we can treat the measurement noise channel as a classical noise channel, where the transformation between ideal and noisy measurement probability distributions can be written using a response matrix Λ:

$$\vec{p}_{ideal} = \Lambda^{-1} \vec{p}_{noisy}. \qquad (37)$$

Here we use $\vec{p}_{ideal}$ and $\vec{p}_{noisy}$ to represent the ideal and noisy probability distributions. And the element $\Lambda_{\boldsymbol{x},\boldsymbol{y}}$ of the response matrix Λ is defined as

$$\Lambda_{\boldsymbol{x},\boldsymbol{y}} = \langle\boldsymbol{x}|\rho|\boldsymbol{y}\rangle, \quad \boldsymbol{x}, \boldsymbol{y} \in \mathbb{Z}_2^{N_q}, \qquad (38)$$

which can be estimated directly by preparing the computational state and measuring it in the computational bases, or by using some tomographic means, such as quantum detector tomography (QDT) [307, 310, 311]. The main idea of QDT is to estimate an unknown set of fixed noisy POVM operators $\{\hat{\Pi}_{\boldsymbol{x}}\}_{\boldsymbol{x}}$ with a set of well-known states $\{\rho_i\}_i$. Once the

noisy POVM operators are reconstructed, we can extract the response matrix elements using

$$\hat{\Pi}_{\boldsymbol{x}} = \sum_{\boldsymbol{y}} \Lambda_{\boldsymbol{x},\boldsymbol{y}} \Pi_{\boldsymbol{y}}, \quad \forall \boldsymbol{x}, \tag{39}$$

where $\Pi_{\boldsymbol{y}}$ denotes the error-free POVM operator.

In practice, however, for non-classical noise, the estimated measurement probability distribution obtained from $\vec{p}_{est} = \Lambda^{-1}\vec{p}_{noisy}$ may be unphysical due to the negative terms of the vector $\vec{p}_{est}$. Moreover, the statistical uncertainties in the response matirx can be amplified through simply inversion, similar to the challenges faced in high-energy physics. Unfolding methods are thus introduced to readout error mitigation [312, 313], which show robustness to some failure cases of matrix inversion and least-squares.

Simplified classical models. Dealing with the full response matrix of size $2^{N_q} \times 2^{N_q}$ is not scalable beyond hundreds of qubits. Therefore, in order to avoid the exponential consumption of measurements, many scalable approaches [308, 314–316] have been proposed to simplify the response matrix model.

Tensor Product Noise (TPN) model. The simplest model assumes that the noise acts independently on each qubit and is called the Tensor Product Noise (TPN) model [308, 317, 318]. After such a simplification, the response matrix can be described in terms of the tensor product of N_q single-qubit response matrix as

$$\Lambda^{\text{TPN}} = \prod_{i=1}^{N_q} \begin{pmatrix} 1 - p_i & q_i \\ p_i & 1 - q_i \end{pmatrix}, \tag{40}$$

where p_i and q_i denote the single-qubit readout error probability of $1 \rightarrow 0$ and $0 \rightarrow 1$, respectively. We can see that the number of measurements required to construct the TPN response matrix is reduced from $\mathcal{O}(2^{N_q})$ to $\mathcal{O}(N_q)$. However, in the TPN model, the multi-qubit readout error correlations are neglected.

Continuous Time Markov Processes (CPTP) noise model. Bravyi *et al.* [308] have extended the TPN model to the CTMP noise model, which takes the correlated errors into account. It defines a matrix exponential $\Lambda = e^G$ to describe the response matrix, and the generator $G = \sum_{i=1}^{2N_q^2} r_i G_i$ models the readout error. Here $r_i \geq 0$ refers to error rate and G_i represents the single-qubit or two-qubit readout error. For example, the generator describing falsely measuring $|00\rangle$ to $|11\rangle$ on two qubits is $|11\rangle\langle00| - |00\rangle\langle00|$. The MEM with the CPTP model has been experimentally demonstrated on the superconducting device [308].

Subspace Reduction. When the measured probability distribution only contains a few principal bit strings with high probability, we can mitigate readout errors in a renormalized subspace defined by the observed probability distribution [314, 318]. The subspace reduction of the full 2^{N_q} dimensional response matrix efficiently circumvents the original exponential overhead and avoids matrix inversion by using the matrix-free iterative methods [314].

Readout symmetrizing. Due to the observation that mis-measuring $|1\rangle$ to $|0\rangle$ is more frequent than $|0\rangle$ to $|1\rangle$, Several studies symmetrizing the readout with targeted Pauli $\hat{X}$ gates are proposed [56, 319–321].

Readout balancing. We can exploit the readout asymmetry in the practical cases, which aims to rebalance the measurement outcomes by minimizing the expected number of qubits in the $|1\rangle$ state [319].

Bit-flip averaging. Bit-flip averaging [320] reduces the calibration overhead by an exponential factor without making any specific assumptions about the classical noise model for MEM. It applies Pauli $\hat{X}$ gates on the randomly chosen qubits before measurements and offsets the effect of $\hat{X}$ gates using classical post-processing. This leads to a symmetrized effective response matrix $\hat{\Lambda}$ which contains only $\mathcal{O}(2^{N_q})$ free parameters compared to $\mathcal{O}(2^{2N_q})$ in the full matrix. Another independent work [321] utilizes the same bit-flipping protocol. It transforms the bias between the ideal and noisy expectation value to a multiplicative factor, and eliminates it by deviding the noisy expectation by an estimated factor obtained in the calibration procedure.

2. Quantum noise model for MEM

Almost all of the readout mitigation approaches discussed above are applied under the classical noise model assumption. However, quantum measurements inevitably suffer from the quantum coherent noise [307, 310, 322]. With the presence of quantum noise, the noisy POVM operators have nontrivial off-diagonal values, so the classical assumption no longer holds. Fitting the difference between two specific measurement statistics to the Fourier series can effectively detect the presence of quantum noise [322]. Meanwhile, many approaches, such as IZ dephasing [307, 322], can be applied to eliminate the quantum noise, so that the effective measurement device has only classical noise. Then we can utilize those MEM methods mentioned above to deal with the classical noise.

3. MEM combined with other techniques

MEM using QEC. On devices whose readout errors dominate over the entangling gate errors, one can combine readout error mitigation with quantum error correction to actively reduce readout errors on a shot-by-shot basis, called active readout error mitigation [323].

Combination with readout compression. The compression readout compresses the large-scale quantum state into one qubit and recovers the state amplitude populations from the one-qubit measurement results [252]. Thus, after appling the compression readout technique, only single-qubit measurements are performed, so this method is free of correlated measurement errors and easy to combine with other MEM methods.

Neural network model for MEM. We can also utilize a trained artificial neural network to characterize an non-linear model [324] for measurement noise, which captures both classical and quantum noise. The neural network model for MEM

is experimentally performed on the IBM's 5-qubit quantum device [324].

D. Purification-based QEM scheme

Recently, a range of purification-based QEM schemes [261, 325–332] have been proposed to extract the ideal state component from the noisy state, without any requirements on the pri-ori knowledge of the quantum noise channel. These schemes assume that the state of interest is usually a pure state, while the noise tends to corrupt it into a mixed state.

The two typical methods are Virtual Distillation (VD) [261] and Error Suppression by Derangement (ESD) [325], which use multiple copies of the noisy state to reduce the state preparation error. Given an error-free state $|\psi_0\rangle\langle\psi_0|$, the noisy output state ρ can be expressed via the spectral decomposition

$$\rho = (1 - \lambda)|\psi\rangle\langle\psi| + \lambda\rho_{\text{err}}, \tag{41}$$

where $\lambda \geq 0$ denotes the error rate. The pure state $|\psi\rangle\langle\psi|$ denotes the dominant eigenvector which may not match the ideal state $|\psi_0\rangle\langle\psi_0|$ due to the coherent error [333]. And the mixed state ρ_{err} refers to the noisy component consisting of the weighted sum of other eigenvectors in the spectral decomposition. Using M copies of noisy state, the M-degreed mitigated expectation value is

$$\langle\mathcal{O}\rangle_{\text{em}} = \frac{\text{Tr}(\rho^M O)}{\text{Tr}(\rho^M)} = \frac{(1-\lambda)^M\langle\psi|O\psi\rangle + \lambda^M\text{Tr}(O\rho_{\text{err}}^M)}{(1-\lambda)^M + \lambda^M\text{Tr}(\rho_{\text{err}}^M)}$$
$$\approx \langle\psi|O\psi\rangle + \mathcal{O}(\lambda^M) \tag{42}$$

where $\mathcal{O}(\lambda^M)$ is the approximation error which decays exponentially with the number of copies.
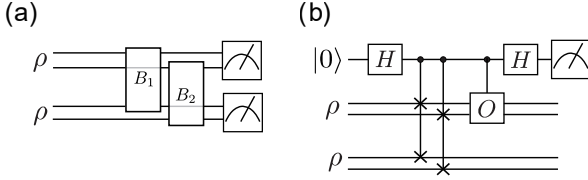

FIG. 4. **The illustration of two implementations of virtual distillation for 2 qubit states.** (a) We can apply a layer of diagonalization gates B_i to compute Eq. 4, with a joint measurement on M(here M=2) copies. (b) We can also add an ancilla qubit and use controlled-S^M gates to estimate the expected value $\text{Tr}(O\rho^M)$ using the probability of measuring the ancilla qubit in the $|0\rangle$ state.

We usually don't need to prepare the exact ρ^M state to compute the $\langle\mathcal{O}\rangle_{\text{em}}$. An illustration of the implementation of VD and ESD is given in Fig. 4. They both utilize the cyclic shift operator (also called derangement operator) $S^{(M)}$ on the system with size MN_q. Using the property of $S^{(M)}$

$$S^{(M)}|\psi_1\rangle|\psi_2\rangle\ldots|\psi_M\rangle = |\psi_M\rangle|\psi_1\rangle\ldots|\psi_{M-1}\rangle, \tag{43}$$

we have

$$\frac{\text{Tr}(O\rho^M)}{\text{Tr}(\rho^M)} = \frac{\text{Tr}(O^i S^{(M)}\rho^{\otimes M})}{\text{Tr}(S^{(M)}\rho^{\otimes M})}, \tag{44}$$

where O^i refers to the observable O acting on an arbitrary subsystem i. The VD shown in Fig. 4(a) applies a layer of diagonalization gates B_i to compute Eq. 4, with a joint measurement on M copies. While the ESD given in Fig. 4(b) adds an ancilla qubit and uses controlled-S^M gates to estimate the expected value $\text{Tr}(O\rho^M)$ (and for $\text{Tr}(\rho^M)$ setting $O = I$) by

$$\text{Tr}(O\rho^M) = 2P_0 - 1. \tag{45}$$

Here we denote the probability of measuring the ancilla qubit in the $|0\rangle$ state by P_0.

The VD (or ESD) scheme is subject to some limitations. First, the coherent dismatch between the target pure state $|\psi_0\rangle\langle\psi_0|$ and the dominant eigenvector $|\psi\rangle\langle\psi|$ of the noisy state contributes to a noise floor regardless of the increasing of M. Second, the increased overhead of qubits and controlled gates with large M is hard to afford for the NISQ devices. Taking these limitations into account, many alternative protocols have been explored.

1. Resource-efficient variants for purification-based schemes

Realization with classical shadows. Classical shadows [334–336] are protocols used to efficiently predict many different properties, in particular linear functions. Considering a trade-off between qubits (and controlled gates) and measurements overhead, many methods implement the VD(or ESD) scheme with classical shadows [329, 330].

Dual-state purification. This protocol [331] purifies states using $\frac{\rho\bar{\rho}+\bar{\rho}\rho}{2}$, and the $\bar{\rho}$ is the dual state of ρ. The ideal expectation value of observable O is estimated as

$$\langle\mathcal{O}\rangle_{\text{em}} = \frac{\text{Tr}(\frac{\rho\bar{\rho}+\bar{\rho}\rho}{2}O)}{\text{Tr}(\frac{\rho\bar{\rho}+\bar{\rho}\rho}{2})}. \tag{46}$$

It works when the noisy state ρ and its dual state $\bar{\rho}$ share the same dominant eigenvector in their spectral decompositions. Dual-state purification is efficient in terms of qubit overhead since its implementation requires at most one ancilla qubit.

Combination with active qubit resets. The active qubit reset technique is enabled by major quantum computing architectures including superconducting qubit and trapped-ion devices. With the use of active qubit resets, we can achieve a similar error suppression using $2N + 1$ qubits compared to $MN + 1$ qubits with the original VD [332], which shows a space-time trade-off in the computational resources.

2. Generalized purification methods

In order to deal with the coherent dismatch, we can consider a general polynomial function $f(\rho, M) = c_0 + c_1\rho + c_2\rho^2 + \ldots + c_M\rho^M$ [329, 337]. Recently Xiong *et al.* [328] provides

a more general framework called permutation filters for those schemes using permutations. It uses a filter

$$y_{\text{filter}} = \frac{\text{Tr}(O\mathcal{F}_{\vec{\alpha}}(\rho, M))}{\text{Tr}(\mathcal{F}_{\vec{\alpha}}(\rho, M))} \quad (47)$$

to substitute the polynomial $\frac{\text{Tr}(\rho^M O)}{\text{Tr}(\rho^M)}$, where $\mathcal{F}$ denotes some complex function and the parameter optimization for $\vec{\alpha}$ is proven to be an invex problem, which can always converge to the global optimum.

E. Quantum subspace expansion

Quantum subspace expansion (QSE) is first proposed to explore the excited states, and it can also contribute to error mitigation in VQE with additional classical resources [105, 121, 338, 339]. The QSE effectively mitigates coherent errors due to the imperfect variational optimization. Given a noisy ground state ρ prepared on the quantum device and the target Hamiltonian H, we consider a variational subspace spanned by ρ, as

$$\{\rho_{\text{sub}} = \sum_{i,j=1}^{M} c_i c_j^* P_i \rho P_j | c_i \in \mathcal{C}, P_i \in \{I, X, Y, Z\}^{\otimes N_q}$$

$$\text{and } \text{Tr}(\rho_{\text{sub}}) = 1\}, \quad (48)$$

where the parameters $\vec{c} = (c_1, ..., c_M)$ denote the expansion coefficients and M refers to the number of expansion terms. Then, to obtain the error-mitigated value, we need to solve the following minimization problem

$$\min_{\vec{c}} \{\text{Tr}(\rho_{\text{sub}} H)\}$$
$$\text{such that } \text{Tr}(\rho_{\text{sub}}) = 1. \quad (49)$$

The spectrum of the Hamiltonian within the classically expanded subspace can be calculated as the solution to a generalised eigenvalue problem

$$\hat{H}\vec{c} = E\hat{B}\vec{c}. \quad (50)$$

Here, E is a diagonal matrix with elements representing the eigenenergies. $\hat{H}_{ij} = \text{Tr}(P_i \rho P_j H)$ and $\hat{B}_{ij} = \text{Tr}(P_i \rho P_j)$ both can be efficiently estimated via quantum computers. With the optimized parameter obtained from minimization, we can compute the error-mitigated expectation value using

$$\langle O \rangle_{\text{em}} = \sum_{i,j=1}^{M} c_i c_j^* \text{Tr}(P_i \rho P_j H). \quad (51)$$

Note that the efficiency of the QSE method depends on the number of expansion terms M, thus it may not effective in suppressing stochastic errors due to the requirement on exponential expansion terms to project the noisy state to the error-free subspace [105, 142, 337]. While when combined with certain properties of the target state such as symmetry [340], the QSE can handle stochastic errors more efficiently because

the complexity of constructing the projection subspace is reduced. Beside, a generalised framework of QSE [337] has extended the expansion operators from Pauli operators to more general operators relevant to the noisy state.

F. Verification methods

Verification methods [338, 340, 341] exploit the knowledge of inherent symmetry within the quantum system, such as the spin symmetry in quantum many-body physics. This methods focus on the errors that place state outside the symmetry-preserving subspace.

The symmetry commonly used is Pauli symmetry $\hat{S}$, which can be effectively estimated. Suppose the Pauli symmetry of the system with size n is $\hat{S} = \prod_{i=1}^{M} \hat{S}_i, \hat{S}_i \in \mathbb{P}^{\otimes N_q}$, where M is the number of non-trivial terms in $\hat{S}$, and $\mathbb{P}^n$ refers to the n-qubit Pauli group. The verification of the symmetry can be achieved by dropping the circuit runs which fail the verification or by using the post-processing appraoches, as we will discuss below.

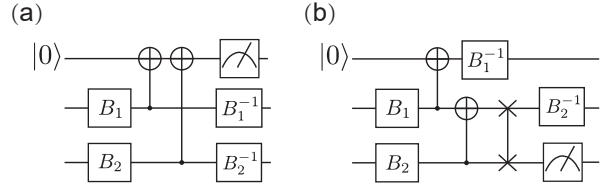#### 1. Symmetry verification

FIG. 5. **Symmetry Verification.** (a) Symmetry verification circuit. (b) The low-cost version of symmetry verification circuit.

Symmetry verification [338, 340] discards the circuit runs failed to pass the verification, which costs additional measurement overhead. It utilizes an ancilla qubit interacting with each qubit in the system register, which is of the form shown in Fig. 5(a). The gates $\{B_i\}_{i=1,2}$ are the basis transformation gates that map the eigenstate of symmetry $\hat{S}$ with eigenvalue s to $|0\rangle$ state. If the ancilla qubit reads 1, we discard this circuit run. Note that the circuit can only detect odd number of error(s). In the low-cost version [340] of symmetry verification shown in Fig. 5(b), it shuffles the ancilla qubit along the system register, which needs only local CNOT and SWAP two-qubits gates.

Both cases require $\mathcal{O}(M)$ circuit depth to ensure the entanglement between the ancilla qubit and each register qubit individually, which is general intractable in quantum circuit. Moreover, the verification circuit applied on the noisy state may introduce extra errors, which reduces the reliability of verification. An alternative way to perform symmetry verification will be detailed below.

2. Symmetry expansion

We can also implement verifications via post-processing approach [340]. Usually if the target quantum state is the eigenstate of $\hat{S}$ (or in the eigen-subspace) with eigenvalue s, we can construct projector valued measurement to project the output state ρ to the $\hat{S} = s$ subspace. Suppose the corresponding projector is $\hat{\Pi}_s$ and we have $\hat{\Pi}_s\hat{\Pi}_s = \hat{\Pi}_s$. The effective density matrix after projection is

$$\rho_s = \frac{\hat{\Pi}_s\rho\hat{\Pi}_s}{\text{Tr}(\hat{\Pi}_s\rho\hat{\Pi}_s)} = \frac{\hat{\Pi}_s\rho\hat{\Pi}_s}{\text{Tr}(\hat{\Pi}_s\rho)}. \tag{52}$$

If we will measure an observable O which commutes with the symmetry $\hat{S}$ and $[O, \hat{S}] = 0$ and $[O, \hat{\Pi}_s] = 0$. Then the expectation value under the projected state is

$$\langle O \rangle_{\text{em}} = \text{Tr}(O\rho_s) = \frac{\text{Tr}(O\hat{\Pi}_s\rho)}{\text{Tr}(\hat{\Pi}_s\rho)}. \tag{53}$$

Note that the probability of passing the verification is just the probability of projecting the noisy state to the $\hat{S} = s$ subspace, which is $\text{Tr}(\hat{\Pi}_s\rho)$. Hence we need $\Upsilon_{\text{SE}} = \frac{1}{1-\text{Tr}(\hat{\Pi}_s\rho)}$ more measurement shots as the sampling overhead for mitigation.

We can combine the post-processing approaches with QSE to further improve the efficiency of mitigation, which called s-QSE [340]. If we choose $\hat{\Pi}_s = \frac{I+s\hat{S}}{2}$, we will have

$$\langle O \rangle_{\text{em}} = \frac{\text{Tr}(O\rho) + s\text{Tr}(O\hat{\Pi}_s\rho)}{1 + s\text{Tr}(\hat{\Pi}_s\rho)}. \tag{54}$$

Then the original minimization problem (as we mentioned in the Section III E) is reformulated to a generalized eigenvalue problem, and the terms $\text{Tr}(O\rho)$, $\text{Tr}(O\hat{\Pi}_s\rho)$ and $\text{Tr}(\hat{\Pi}_s\rho)$ can be efficiently estimated via quantum devices. The s-QSE method has been experimentally demonstrated to mitigate errors in the VQE of H_2 with two transmon qubits [342].

Cai has extended the post-processing approaches to a general framework called symmetry expansion [341], which encompasses a broader range of symmetry-based error mitigation methods. For example, VD can be seen as a special case of symmetry expansion using permutation symmetry. Moreover, the particle number is also applied as symmetry in the post-processing approach[343].

3. Other verification methods

Verified phase estimation. It applies phase estimation to estimate expectation values while effectively post-selects for the system register to be in the starting state [344]. It can also be adapted to the case without the use of control qubits, which simplifies the control circuits.

Pauli check sandwiching. The Pauli check sandwiching scheme [345] applies multiple pairs of Pauli checks to detect the occurrence of errors, and obtains the mitigated results using post-selecting. Each pair of Pauli checks uses one ancilla qubit to detect a component of the error operator.

G. Mitigating the coherent errors

Coherent errors refer to the imperfect or unwanted unitary rotations acting in the circuits, which can be modeled as

$$U(\vec{\theta}) = e^{-\frac{i}{2}\vec{\theta}\cdot\vec{\sigma}}, \tag{55}$$

where $\vec{\theta} = (\theta_1, ..., \theta_{4^{N_q}})$ quantifies the strength of the coherent error on the 4^{N_q} Pauli bases. Coherent errors map the noiseless pure states to another pure states, since unitary operators maintain the quantum coherence of the states. While they are purity-preserving, it can pose a threat to the reliable multi-qubit quantum computation. Now we will introduce several typical methods for mitigating coherent errors [1, 346].

1. Randomized compiling

Randomized compiling (RC) [276, 277] is designed to tailor coherent errors into stochastic Pauli errors, and we can combine RC techniques with other QEM schemes [280, 291, 301].

When the two-qubit Clifford gate errors dominant over other types of error, we can mitigate the coherent error in the two-qubit gates using RC. After sandwiching each two-qubit gate between randomly sampled Pauli gates and compiling those twirling Pauli gates into the original single-qubit gates (which can be implemented on the classical computers in advance), the newly generated circuits are logically equivalent to the bare circuits with the same circuit depth. And the averaged results over many logically-equivalent circuits is exactly the desired result with tailored noise.

2. Hidden inverses

Hidden inverses (HI) [347–349] is first introduced to mitigate certain coherent errors (such as over-rotations and phase misalignment) in the trapped-ion quantum computer [347], and then it has been extended to implement on the superconducting hardwares [349]. The HI method relies on the self-adjoint unitary operators satisfying $U = U^\dagger$ or self-inverse unitary operators with $U = U^{-1}$, such as H and CX. Though these gates represent the same operation in the error-free case, they may suffer from noise with different strength in practice, due to the different compiling ways for U and $U^\dagger(U^{-1})$. Then we can mitigate coherent errors by a local optimization which determines to construct the same gate from the original elementary gate sequence or the sequence of the inverted (conjugate) gate.

H. Combination with quantum error suppression techniques

1. Combination with QEC

Individual error reduction. The individual error reduction method [350] make sufficient use of the limited QEC techniques which have been well demonstrated on several qubits. It reduces error on each qubit respectively and obtains the mitigated expectation value of the observable by post-processing. Consider the Lindblad master equation describing the noise process

$$\frac{\partial \rho}{\partial t} = \mathcal{L}(\rho) = \sum_i \mathcal{L}_i(\rho), \tag{56}$$

where the $\mathcal{L}_i$ denotes the Lindblad operator acting on the single qubit. We can give a solution for this equation with the duration τ of the noise process to the first order approximation

$$\begin{aligned} \rho(t+\tau) &= \mathcal{E}_\tau(\rho(t)), \\ \mathcal{E}_\tau(\rho) &= e^{[\tau\mathcal{L}(\rho)]} \approx 1 + \tau \sum i\mathcal{L}_i(\rho), \end{aligned} \tag{57}$$

where $\mathcal{E}_\tau$ is the Lindblad evolution operator. Suppose the error on the i-th qubit is reduced by a known factor g_i via QEC, then the corresponding Lindblad evolution operator $\mathcal{E}_\tau^i$ turns to

$$\mathcal{E}_\tau^i(\rho) \approx 1 + \tau \sum_{j \neq i} \mathcal{L}_j(\rho) + \tau(1-g_i)\mathcal{L}_i(\rho). \tag{58}$$

Hence, after QEC applied on the i-th qubit, the expectation value of observable O is $\langle O \rangle_i = \text{Tr}(\rho^i O)$ with the output density matrix ρ^i. We can estimate the ideal case of the expectation value of observable O by a linear combination of different $\langle O \rangle_i$:

$$\langle O \rangle_{\text{Ind}} = \text{Tr}(\rho^{\text{Ind}} O) = \langle O \rangle - \sum_i \frac{1}{g_i}(\langle O \rangle - \langle O \rangle_i). \tag{59}$$

Note that the individual error reduction has suppressed errors to the first order and can be combined with other error mitigation methods for higher-order error suppression. However, it relies on the accurate estimation of the factor g_i which results in extra cost for characterizations. And additional quantum resources (qubits or gates) are needed to implement QEC on a single qubit.

Code space projection. The code space projection method [329, 351] decodes errors on logical qubits via post-processing, without additional qubits and operations for syndrome measurements. To encode k logical qubits with N physical qubits, we use a stabilizer code $[[N, k]]$ defined by a stabilizer group $S = \langle S_1, S_2, ..., S_{N-k} \rangle$ with generators $\{S_i\}_i$. And the code subspace is determined by the projection operator

$$\Pi = \prod_{i=1}^{N-k} \frac{\mathbb{I} + S_i}{2} = \frac{1}{2^{N-k}} \sum_j M_j, \tag{60}$$

where $M_j \in S$ refers to the element in the group. Given a logical observable O which can be decomposed by Pauli operators $O = \sum_m \gamma_m P_m$, this method can reduce errors which take the physical state outside the code space. Then we estimate the ideal expectation value using

$$\langle O \rangle_{\text{CS}} = \frac{1}{c2^{N-k}} \sum_{j,m} \gamma_m \text{Tr}(\rho M_j P_m). \tag{61}$$

Stochastic and deterministic subspace expansion schemes have been proposed [351] for estimation of Eq. 61. We can also realize the code space projection using classical shadow techniques [329].

2. Dynamic Decoupling

Dynamical decoupling (DD) [352–357] methods are designed to suppress decoherence caused by system-environment interaction. The main idea of DD is to apply specific pulse sequence called DD sequence to the idle qubits, which keeps the overall logic of the circuit unchanged. Introducing the DD techniques into the QEM schemes can improve the circuit performance while it dosen't result in additional sampling overhead [291, 358].

3. Single-qubit gate scheduling

Rather than adding additional gates to the circuit as DD does, single-qubit gate scheduling [359] optimizes circuits by scheduling within idle windows which are periods of idle qubit waiting for the next operation. *As Late As Possible* (ALAP) scheduling is a typical scheduling technique used as a default approach in IBM Qiskit [360] to execute the single-qubit gates at the end of the idle windows. We can also tune the positions of single-qubit gates within idle windows by circuit slicing and inverting [359]. A variational QEM method designed to perform DD and single-gate scheduling within the VQA framework has been proposed recently [358], which avoids specific configuration selections for DD sequences and gate positions.

I. Framework for QEM

Recently, much effort has been devoted to exploring the unified framework for QEM schemes and the general bounds for sampling overhead [268–270, 361–363]. Takagi *et al.* [270] described the QEM process as the concatenation of a quantum-classical channel composed of quantum operations and independent POVM measurements, and a classical-classical channel for the classical post-processing. They also introduced the maximum bias to benchmark the worst-case performance of the QEM strategies for an arbitrary state and observable, and then derive a general bound on the sampling overhead based on the maximum bias. And their related work [363] has showed the explicit universal lower bounds

on the sampling overhead, which grow exponentially with the circuit depth, indicating the fundamental limitations for QEM schemes.

Other metric such as extraction rate was proposed to characterize the cost-effectiveness of QEM schemes within the *linear quantum error mitigation* [268] framework, where the process of error mitigation is regarded as extracting the effective state out of the unmitigated state. Statistics principles are also applied into a general QEM formalism [361] where an approximate quadratic reduction with gate number N has drawn in error increase.

IV. QUANTUM CIRCUIT COMPILATION

Compilation in the classical computing community is the process of converting a high-level programming language into a machine language that the computer can understand and execute fluently. Similarly, in the field of quantum computation, quantum circuit compilation (QCC) is used to transform a quantum algorithm always in a mathematical form into its corresponding quantum circuit which can be executed on a real quantum device.

Usually, QCC includes three stages as follows (see Fig. 6).

1. *Decomposion* (also referred to as the synthesis). Usually, a mathematically designed quantum algorithm can be represented by several n-qubit unitary operators, such as the Grover iteration in Grover's algorithm, the controlled exponentiation in Shor's algorithm, the Hamiltonian simulation in HHL algorithm and so on. However, real quantum devices, especially NISQ devices, usually provide only some elementary quantum gates, such as single-qubit gates and CNOT gates. Thus, it is one of the most fundamental problems that how to decompose a specific quantum algorithm or unitary operator into as few the provided elementary quantum gates as possible for implementation.

2. *Optimization*. Despite the rapid development of quantum hardware, the qubit lifetime of some quantum systems is still not satisfactory and the quantum noise is also inevitable [7, 364]. The optimization aims to further reduce the depth and size of quantum circuits with the help of auxiliary qubits after the decomposition process to alleviate the problem of short qubit lifetime. Thus, the optimization seems like a kind of space-time trade-off, *i.e.*, using ancillary qubits (space) to efficiently reduce circuit size or depth (time).

3. *Mapping* (also referred to as the qubit routing, qubit allocation, quantum circuit transformation). The final stage is to transform the logical quantum circuit to the physical quantum circuit which can be operated on a real quantum device. Actually after decomposition and optimization, there are only elementary quantum gates in the logical circuit, and the size and depth of which are sufficiently optimized. For the quantum hardware with fully-connected architecture, any logical circuit can be directly implemented. However, for the quantum hardware with limited connectivity architecture [7, 364], we should find the correspondence between logical qubits and physical qubits to complete the transformation.

Although QCC has been separated into the above three stages, sometimes there is no exact order or boundaries. Alternatively, one can perform all the three stages simultaneously to complete the QCC, such as the works in Refs. [365–367]. QCC is extremely important as a bridge between quantum algorithms and quantum hardware devices, especially for NISQ devices. There have been a series of results on QCC [364–409] and we will present them according to the different approaches of the three stages mentioned above.

A. Decomposition

The first result in the aspect of *Decomposition* was published in 1995, where all n-qubit unitary operators can be expressed as compositions of a finite number $O(n^3 4^n)$ of single-qubit gates and CNOT gates exactly [368]. Soon, this upper bound was lowered to $O(n 4^n)$ by Knill [369]. Nine years later, the upper bound for the exact decomposition of n-qubit unitary operator has been improved to $O(4^n)$ [370], which coincides with the theoretical lower bound [371]. In 2004, Möttönen *et al.* came up with Cosine-Sine Decomposition (CSD), *i.e.*,

$$F_n^l(R_a) = \begin{pmatrix} R_a(\alpha_1) & & \\ & \ddots & \\ & & R_a(\alpha_{2^{n-1}}) \end{pmatrix}. \qquad (62)$$

to express an arbitrary n-qubit unitary operator in terms of $n-1$-qubit unitary operators [372], which further decreased the upper bound to $4^n - 2^{n+1}$ CNOT gates. Next in 2005, Shende *et al.* applied the NQ matrix decomposition to implement an arbitrary n-qubit operator only using $\frac{1}{2} \times 4^n - 3 \times 2^{n-1} + 1$ CNOT gates [373], where the n-qubit operator can be implemented by a circuit containing three uniformly controlled rotations and four $(n-1)$-qubit operators. And then in 2006, Möttönen *et al.* applied CSD recursively to yield a synthesis algorithm, in which the number of CNOT gates is $\frac{23}{48} 4^n - \frac{3}{2} 2^n + \frac{1}{3}$, which firstly decreased the upper bound less than $\frac{1}{2} 4^n$ CNOT gates [374, 375].

The analysis of lower bound was first came up with by Barenco *et al.* in [368] as a conjecture, *i.e.*, $\frac{1}{9} 4^n - \frac{1}{3} n - \frac{1}{9}$, based on dimension counting. In 2004 Shende *et al.* raised the lower bound to $\frac{1}{4}(4^n - 3n - 1)$ by the technique of parameter counting [371], which also implies a depth lower bound of $\Omega(4^n/n)$.

We can see that there is a gap on the optimal depth within the range of $[\Omega(4^n/n), O(4^n)]$ for general n-qubit circuit optimization without ancillary qubits. Most recently, based on Gray code and unary encoding [376], Sun *et al.* have proved a fact that any n-qubit unitary operator can be implemented by a quantum circuit of size
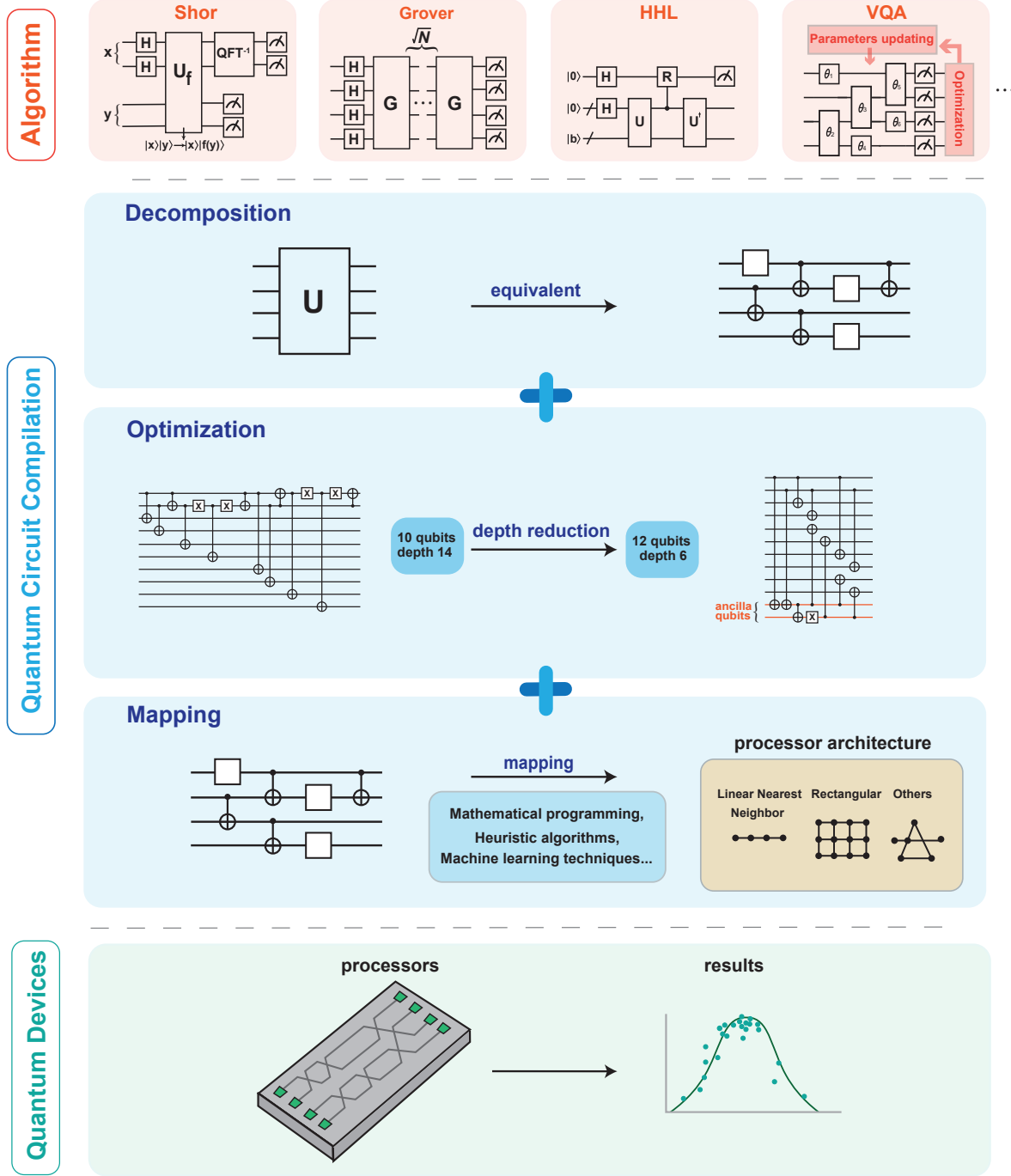
$$O(4^n) \qquad (63)$$

FIG. 6. **Quantum circuit compilation.** Quantum circuit compilation translates the abstract quantum algorithm into a compiled circuit optimized for a specific quantum hardware platform, enabling efficient and highly reliable execution through the use of decomposition, optimization, and mapping methods.

and depth

$$O\left(n2^n + \frac{4^n}{m+n}\right) \quad (64)$$

with $m \leq 2^n$ ancillary qubits [377]. They used CSD re-

peatedly and factored an arbitrary unitary operator into a sequence of uniformly controlled gates. Namely, they presented that any n-qubit unitary operator U can be decomposed as $U = V_n^n(0) \cdot \prod_{i=1}^{2^{n-1}-1} V_{n-\zeta(i)}^n(i) \cdot V_n^n(2^{n-1})$, where $\zeta(n)$

is the Ruler function defined as $\zeta(n) = \max\{k : 2^{k-1}|n\}$, V_k^n denotes an n-qubit uniformly controlled gate whose index of target qubit is k, and different i in $V_k^n(i)$ denote different forms of n-qubit uniformly controlled gates despite the same target qubit k. Combined with their implementation of any n-qubit uniformly controlled gate, i.e., any n-qubit $V_k^n(i)$ can be implemented by a circuit of size $O(2^n)$ and depth $O(n + \frac{2^n}{n+m})$, they derived the above result. What we should note is that their size and depth are simultaneously optimized to asymptotically optimal based on the lower bound in [371]. In specific without any ancillary qubits, the depth can be optimized to $O(4^n/n)$, which is exactly the lower bound.

Besides the most general decomposition on n-qubit unitary, there have already been some results about the special cases. When the elements of unitary belong to the ring $\mathbb{Z}[\frac{1}{\sqrt{2}}, i]$, the single-qubit unitary can be constructed using H and T gates only [378], in which the similar n-qubit case was conjectured to be implemented by the Clifford and T gates. This conjecture has been proved soon afterwards. An n-qubit unitary operator has an exact representation over the Clifford+T gate set assisted by only one ancilla if and only if its entries are in the ring $\mathbb{Z}[\frac{1}{\sqrt{2}}, i]$, and the total gate count is $O(3^{2n}nk)$, where k is the denominator exponent of the unitary [379]. Besides the above results on exact decomposition of unitary, there have already been some works on approximate decomposition. Based on Solovay-Kitaev theorem, Dawson and Nielsen designed an approximate algorithm, which runs in $O(\log^{2.71}\frac{1}{\epsilon})$ time and produces as output a sequence of $O(\log^{3.97}\frac{1}{\epsilon})$ quantum gates including only Hadamard, controlled-not, and $\pi/8$ gates [380].

B. Optimization

As seen in the rapid development of quantum hardware, the number of physical qubits may increase much faster than the qubit lifetime. Thus, when optimizing quantum circuits, we can consider how to reduce the circuit depth using auxiliary qubits.

In 2000, Cleve showed the quantum Fourier transform can be approximated to the depth of $O(\log n + \log\log(1/\epsilon))$ with sufficient ancillary qubits [381]. Next in 2001, Moore *et al.* proved that a circuit of any size on n qubits composed entirely of CNOT gates (CNOT circuit) can be parallelized to $O(\log n)$ depth with $O(n^2)$ ancilla qubits [382]. Similarly for the circuit of controlled-Pauli and H gate, they derived the same result. And for the Clifford+T circuit, Selinger in 2013 made use of four ancilla qubits to represent Toffoli gate with a T-depth of 1 and a total depth of 7 [383]. Amy *et al.* described an algorithm to reduce the T-depth for Clifford+T circuit [384], and it has worst case runtime of cubic in the number of T gates, qubits and Hadamard gates. Next let us turn our eyes back to the optimization of CNOT circuit because of its significance in quantum computation [368, 410]. In 2020, Jiang *et al.* established an asymptotically optimal space-depth tradeoff for the design of CNOT circuits, that is,

any n-qubit CNOT circuit can be parallelized to

$$O(\max\{\log n, \frac{n^2}{(n+m)\log(n+m)}\})$$

depth with m ancillary qubits, where m being any natural number [385]. Firstly, any n-qubit CNOT circuit can be regarded as an invertible matrix $\mathbf{M}$ over the finite field $\mathbb{F}_2^{n\times n}$. Take a three-qubit CNOT circuit shown in Fig. 7 as an example, the left circuit can be represented as the right matrix. Based on the above representation, the CNOT circuit opti-
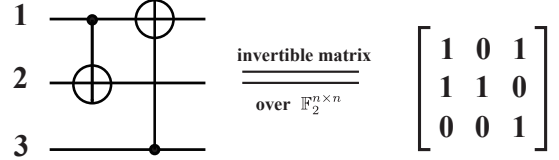

FIG. 7. **The matirx representation of a CNOT circuit.**

mization problem will be transformed into a corresponding parallel Gaussian elimination problem mathematically. Combined with a standard technique in reversible computation, they firstly constructed two CNOT circuits $\mathcal{C}_1$, $\mathcal{C}_2$ with $2n$ and $3sn$ qubits respectively, where $1 \leq s \leq O(n/\log^2 n)$. Specifically, for $\mathbf{x}, \mathbf{j} \in \mathbb{F}_2^n$,

$$\mathcal{C}_1|\mathbf{x}\rangle|\mathbf{j}\rangle|0\rangle^{\otimes 3sn} = |\mathbf{x}\rangle|\mathbf{j}\oplus\mathbf{Mx}\rangle|0\rangle^{\otimes 3sn},$$

$$\mathcal{C}_2|\mathbf{x}\rangle|\mathbf{j}\rangle|0\rangle^{\otimes 3sn} = |\mathbf{x}\rangle|\mathbf{j}\oplus\mathbf{M}^{-1}\mathbf{x}\rangle|0\rangle^{\otimes 3sn}.$$

Then they applied $\mathcal{C}_1, \mathcal{C}_2$ to transform the initial state $|\mathbf{x}\rangle|0\rangle^{\otimes n}|0\rangle^{\otimes 3sn}$ to the final state $|\mathbf{Mx}\rangle$ after permuting the first and second n qubits. This boun d is tight and this algorithm can be directly extended to stabilizer circuits with the reason that any stabilizer circuit has a canonical form "H-C-P-C-P-C-H-P-C-P-C", where H and P are one layer of Hadamard gates and Phase gates respectively [411].

Near-term quantum devices are limited not only by the qubit lifetime, but also by the connectivity between qubits. All the above optimization results focus on the optimal depth, without considering the constraints on connectivity. Thus, it is necessary to consider the quantum circuit optimization problem for the quantum processors with sparsely connected structure. We should note that the optimization here usually refers to reducing the circuit depth with the help of ancillary qubits, which is quite different from the mapping introduced in the next section, which focuses on reducing the number of SWAP gates added to realize the logical circuit physically and keeping the number of qubits constant.

Actually there have already existed some results on reducing the circuit size under the limited connectivity architecture [365–367, 386]. In 2019, based on Gaussian elimination, Kissinger and Griend presented an algorithm which can extract any n-qubit CNOT circuit to $2n^2$-size equivalent CNOT

circuit if the architecture contains a Hamiltonian path [365]. But if there is no Hamiltonian in the architecture, their optimized size is $O(n^3)$. Next in Ref. [366], Nash *et al.* studied the CNOT circuit optimization on any connected graph, and they proposed an algorithm which gives a $4n^2$-size equivalent CNOT circuit for any n-qubit CNOT circuit. This bound has been reduced to $2n^2$ [367]. Furthermore for size optimization, Wu *et al.* came up with an algorithm, inspired by the result in [412], which can reduce the size of any n-qubit CNOT circuit to

$$O(n^2/\log \delta)$$

on any connected graph with δ being the minimum degree, and this bound is optimal when the limited connectivity architecture is a regular graph [367]. In addition, they also optimized the depth of CNOT circuit with the assistance of ancillary qubits. Specifically, their algorithm can reduce the depth of any given n-qubit CNOT circuit to

$$O(n^2/\min\{m_1, m_2\}),$$

with $3n \leq m_1 m_2 \leq n^2$ ancillary qubits and the limited connectivity architecture is $m_1 \times m_2$ grid graph [367]. And this bound is asymptotically optimal when $m_1 \times m_2 = n$. Also this result can be similarly generalized to any constant dimensional grid graph. Most recently, Maslov and Yang considered the optimization of Hadamard-free Clifford circuit on Linear Nearest Neighbor (LNN) [387] qubit connectivity architectures, and derived this circuit can be implemented over LNN in depth $5n$.

C.   Mapping

The study of mapping problem firstly focused on its complexity. In 2009, it has been conjectured that mapping a non-LNN (Linear Nearest Neighbor) circuit into an LNN circuit is NP-complete [388], because the number of possible combinations of all permutations for all two-qubit gates in circuit is $((n-1)!)^k$, where n is the number of qubits and k is the number of two-qubit gates. Obviously, the number $((n-1)!)^k$ will become huge with n and k increasing. In 2014, Shafaei *et al.* modeled the logical quantum circuit as an interaction graph and the structure of quantum device as a connectivity graph. Thus the mapping problem is a standard graph embedding problem with connectivity and interaction graphs as the host and guest graphs, respectively. And the objective is then to minimize the total distance between adjacent nodes of the interaction graph. For a 2D grid connectivity graph, the mapping problem is NP-complete [389]. In 2018, Siraichi *et al.* proved the mapping problem, similar to the classical register allocation, is NP-complete in general [373], which is also derived in [390, 391]. Due to the NP-completeness of mapping problem, researchers have taken more attention on how to reduce the SWAP gates needed approximately. We will introduce and analyze the relevant results in three aspects based on different approaches to solve the mapping problem.

Mathematical programming. It is a most natural idea to reformulate the mapping problem on some specific physical

architectures into a corresponding optimisation problem and to solve it with state-of-the-art tools and solvers [389, 392–396]. In 2009, Hirata *et al.* considered a sorting of the initial qubit order according to a function they defined at first, then sorted this qubit order to the final one. Thus the objective is to minimize the sum of SWAP gates of the first and second step [388]. Next in 2013, Shafaei *et al.* regarded the mapping problem as improving locality of a given quantum circuit, *i.e.*, the minimum linear arrangement (MinLA) problem in graph theory, and they showed the effectiveness of this approach for quantum Fourier transformation and reversible benchmarks through experimental results [393]. Also they focused on 2D grid architecture [389] and refactored the mapping problem to Mixed Integer Programming problem. In 2015, Lye *et al.* formulated the mapping problem on multi-dimensional quantum architectures as Pseudo-Boolean Optimization (PBO) problem, and used a state-of-the-art PBO solver [397] to achieve a minimal number of SWAP gates [392]. In 2019, Wille *et al.* formulated the mapping problem as a symbolic optimization problem that is solved using reasoning engines like Boolean satisfiability solvers, and they provided a method that maps small-size logical circuits to IBM's QX architectures with a minimal number of SWAP and H gates [394].

Heuristic algorithms. To adapt real quantum devices with complex connectivity architectures, a series of heuristic algorithms [364, 398–404] on the mapping problem were developed. Actually for mapping problem, the most important thing is to determine how to insert the SWAP gate effectively and efficiently. Here "effectively" means mapping the logical quantum circuit to physical quantum circuit and running that circuit on a quantum hardware device successfully, while "efficently" means adding as few SWAP gates as possible during the mapping procedure. These two aspects correspond to the two steps in the heuristic algorithm design. The first step is to determine the candidate SWAP-gate sets at present, and the second step is to design the evaluation function that can evaluate the effectiveness of the present SWAP gate.

In 2011, Saeedi *et al.* proposed two heuristic methods to determine which qubits should be reordered, a template matching optimization and an exact synthesis approach [405], which, from experimental results, improved the quantum cost by more than 50% on average compared to the naive method. In 2016, Wille *et al.* considered the effect of inserting SWAP gate on the following two-qubit gates and came up with a look-ahead scheme, which reduced the number of SWAP gates of 56% in the best experimental evaluation [406]. This look-ahead scheme also leads to the improvement for 2D architectures. In 2019, Zulehner, the first-place winner of IBM Q Award, devised a much better algorithm in which quantum gates firstly would be partitioned into layers such that each layer contains only gates acted on distinct sets of qubit. Then they employed the A^* algorithm [413] to search and determine how to insert the SWAP gate for the current layer, which can help to decrease the circuit depth simultaneously. But this algorithm is not suitable to large-scale logical circuit because of the space and time cost of $A*$ algorithm. In order to improve the efficiency of the searching algorithm, in 2019, Li *et al.* proposed a SWAP-based BidiREctional heuris-

tic search algorithm [398]. Instead of searching for a mapping in the full space, they only searched for SWAPs associated with the qubits and designed a heuristic cost function to help find the SWAP that can reduce the sum of distances between each qubit pairs in the front layer. Based on this idea, in 2020, Zhou *et al.* calculated the costs of SWAP gates on all front layers heuristically and chose the one with the minimum cost [399], which would further reduce the complexity in time polynomial in all parameters. Most recently, they proposed a Monte Carlo Tree Search framework to tackle the mapping problem [400], which enabled the search process to go much deeper.

Machine learning techniques. Machine learning techniques have also been exploited to provide a more precise evaluation tool for mapping problem [402–404], as they can provide a more precise assessment of the effects of the present SWAP gate. In 2020, Pozzi *et al.* framed the mapping problem as a reinforcement learning problem, then applied the combinatorial optimization techniques to search for the action leading to the highest-quality under their defined quality function, and finally scheduled SWAP gates [401]. In Ref. [402], graph neural networks were used as an improved architecture to help guide the tree search used to find an optimal set of SWAP gates, then they used an array of mutex locks to represent the current parallelization state to help to reduce the depth of the output circuits. In Ref. [403], Zhou *et al.* took both short- and long-term rewards into consideration to design a scoring mechanism, and proposed a search algorithm which is polynomial in all relevant parameters.

V. BENCHMARKING PROTOCOLS

Benchmarking is a technique to assess the performance and capabilities of quantum devices, and is therefore critical to accelerating progress in quantum computing. Design a "good" benchmarking protocol for quantum devices is complicated for the following reasons: 1) There are different physical ways to implement quantum computing, such as ion traps, photons, superconducting, *etc*, and each has its own strengths and weaknesses. It is difficult for us to design a comprehensive protocol that can evaluate the performance of each physical system well. 2) Different types of applications may have different forms of assessment and core metrics. 3) When the benchmarking protocol is applied to large-scale systems, significant resource consumption may be required.

To date, a number of benchmarking protocols have been proposed and widely used in experiments. These benchmarking protocols can be broadly divided into two major categories, gate-level benchmarking and circuit-level benchmarking, which will be introduced in this section.

A. Gate-level benchmarking

1. Randomized benchmarking (RB)

Quantum process tomography (QPT) is a standard approach to fully characterizing quantum process, but it is sensitive to the state preparation and measurement (SPAM) errors and infeasible for large systems. RB is a technique for estimating the average fidelity of a set of quantum gates without relying on accurate SPAM [414, 415].

RB is originally proposed for random unitary gates [414, 416, 417], and now is developed as a widely used experimental technique for characterizing the average error of quantum operations. Typically, RB is executed with gates from the Clifford group, which is referred to Clifford randomized benchmarking (CRB) [40, 415, 418, 419]. A CRB protocol consists of the following steps:

Step 1: Generate RB sequences. First randomly sample a sequence of gates of a fixed length m from the Clifford group G on n-qubits, and then a global inversion gate is added to return the qubits to the initial state in the ideal case without noise. For each length m, we choose K_m RB sequences.

Step 2: Execute the RB sequences on the noisy quantum devices. Measure the output state to estimate its overlap with the initial state, which is named as survival probability α_m. Repeat K_m sequences to obtain the a single average survival probability $\overline{\alpha_m}$.

Step 3: Fit the results. Repeat Steps 2 for various sequence lengths m to produce a list of average survival probabilities $\{\overline{\alpha_m}\}_m$, and then fit $\{\overline{\alpha_m}\}_m$ to the a single exponential decay model:

$$\overline{\alpha_m} = A_0 p^m + B_0 \tag{65}$$

where A_0 and B_0 absorb state preparation and measurement errors as well as an *edge effect* from the error on the final gate. The quality parameter p determines the average error-rate r according to the relation

$$r = \frac{2^n - 1}{2^n}(1 - p). \tag{66}$$

The RB protocol has been extensively developed in recent years. It has been extended for other finite groups [278, 420–427], and it has been also extensively experimentally studied [40, 428–431]. More recently, a general framework for randomized benchmarking has been discussed [432].

2. Cross-entropy benchmarking (XEB)

Linear XEB is a sampling-based approach to characterize an arbitrary quantum gate [433–435], and is experimentally friendly for large-scale quantum devices with dozens of qubits. In particular, XEB provides a measure to approximate the fidelity of random quantum circuit (RQC) in the quantum computational advantage experiments with limited bitstring samples [7–9].
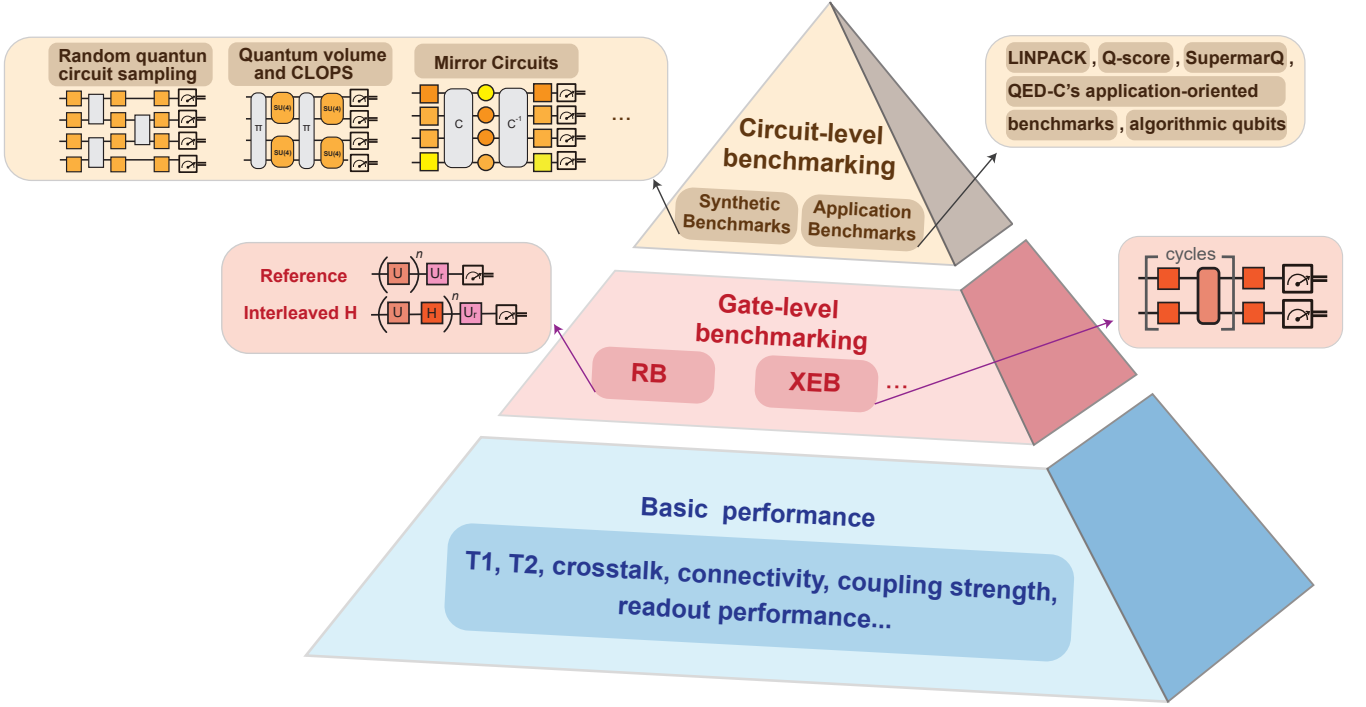
FIG. 8. **Benchmarking methods for different levels.** These methods can be divided into three levels, basic performance benckmarking, gate-level benckmarking and circuit-level benchmarking. At the lowest level, the basic performance of quantum devices, such as T_1, T_2, and chip connectivity, is calibrated, while in the gate-level benckmarking, each individual quantum operations, such as single-qubit gate and two-qubit gate, are optimized to enable the implementation of quantum circuits. In the circuit-level benchmarking, the performance of the quantum system is tested by executing complete quantum circuits or particular applications. Higher-level benchmarks reflect a integrated performance, while lower-level benchmarks are the basic steps performed prior to the utilization of the processor.

For single-qubit gates XEB, the steps are:

Step 1: Generate XEB sequences. First randomly sample a sequence of gates of a fixed length m from the single-qubit gateset formed by the eight $\pi/2$ rotations along the following axes in the Bloch sphere representation: $\pm X, \pm Y$, and $\pm(X \pm Y)$ on a single-qubit, and then end with a final random single-qubit gate and measurement. For each length m, we choose K_m XEB sequences.

Step 2: Execute the XEB sequences on the noisy quantum devices. By comparing the measured state probabilities with the ideal ones, we can calculate the relative cross-entropy differences between probability distributions [434],

$$\alpha = \frac{H_{\text{uniform,ideal}} - H_{\text{experiment,ideal}}}{H_{\text{uniform,ideal}} - H_{\text{ideal}}} \quad (67)$$

where $H = -\sum_i p_i \log(q_i)$ is the cross entropy between two probability distributions $\{p_i\}$ and $\{q_i\}$, and $H_{\text{uniform,ideal}}$ is the cross-entropy between the uniform and ideal distributions, $H_{\text{experiment,ideal}}$ is the cross-entropy between the experimentally measured and ideal distributions, and H_{ideal} is the self-entropy of the ideal distribution. Repeat K_m sequences to obtain the a single average survival probability $\overline{\alpha_m}$.

Step 3: Fit the results. Repeat Steps 2 for various sequence lengths m produces a list of average survival probabilities $\{\overline{\alpha_m}\}_m$, and then fit $\{\overline{\alpha_m}\}_m$ to the a single exponential decay model:

$$\overline{\alpha_m} = A_0 p^m + B_0 \quad (68)$$

where A_0 and B_0 absorb state preparation and measurement errors. The decay parameter p can be converted to the average error r and Pauli error r_P,

$$r = \frac{N-1}{N}(1-p) \quad (69)$$

$$r_P = \frac{N-1}{N}(r) \quad (70)$$

with $N = 2^n$ the dimensionality of the system having n qubits.

The procedure of characterizing two-qubit gates using XEB is similar to that of single-qubit gates, except that the circuit sequence generated in Step 1 is different. Specifically, each cycle of the XEB circuit consisting of a layer of random single-qubit gates and the two-qubit gate, with a final round of random single-qubit gates appended at the end, where single-qubit gate set is still $\pm X, \pm Y$, and $\pm(X \pm Y)$. Finally, divide the Pauli fidelity of the cycle by the Pauli fidelity of the two single-qubit gates to obtain the Pauli fidelity of the two two-qubit gates.

XEB is a powerful calibration tool for near-term quantum devices. Some works have attempted to analyze how to spoof linear XEB, an important topic related to the quantum computational advantage experiments [436, 437]. In addition, Chen *et al.* proposed the *Clifford* XEB [438], which replaces the random circuits with Clifford circuits, to enable the characterization of large-scale quantum systems, since Clifford circuits can be simulated in polynomial time.

B. Circuit-level benchmarking

1. Random Quantum Circuit (RQC) Sampling

RQC sampling is outstanding candidate to demonstrate quantum computational advantages, and also a tool to characterize the overall performance of the quantum processor. The steps of RQC sampling are:

Step 1: Generate RQC. Each RQC is composed of m cycles, and each cycle is composed of a single-qubit gate layer and a two-qubit gate layer. In the single-qubit gate layer, single-qubit gates are applied on all qubits and chosen randomly from the set of $\{\sqrt{X}, \sqrt{Y}, \sqrt{W}\}$, where $\sqrt{X}=R_X(\pi/2)$, $\sqrt{Y}=R_Y(\pi/2)$, and $\sqrt{W}=R_{(X+Y)}(\pi/2)$ are $\pi/2$-rotation around specific axis. Each single-qubit gate on a qubit in subsequent cycle is independently and randomly chosen from the subset of $\{\sqrt{X}, \sqrt{Y}, \sqrt{W}\}$, which does not include the single-qubit gate to this qubit in the preceding cycle. In the two-qubit gate layer, two-qubit gates are applied according to a specified pattern, labeled by A, B, C, and D, in sequence of ABCDCDAB. By executing the four-cycle sequence ABCD once, the two-qubit gate between all qubits can be executed exactly once. Finally, an additional single-qubit gate layer is applied after m cycles and before measurement.

Step 2: Execute the RQC on the noisy quantum devices and collect the measured bitstrings $\{x_i\}$.

Step 3: Compute the linear cross-entropy benchmarking fidelity,

$$F_{\mathrm{XEB}} = 2^n \langle p(x_i) \rangle_i - 1 \qquad (71)$$

where n is the number of qubits, $p(x_i)$ is the probability of bitstring x_i calculated for the ideal quantum circuit.

2. Quantum volume (QV) and circuit layer operations per second (CLOPS)

QV is a single-number metric for quantifying the computational power of a near-term quantum computers of modest size [439]. In general, to achieve higher QVs, quantum computing systems not only require more qubits, but also to maintain high-fidelity operations, high qubit connectivity, high gate parallelism, as well as high-performance circuit compilation techniques.

A QV benchmark procedure is as follows:

Step 1: Generate QV circuits. The circuits U consist of d sequential layers that act on m qubits. Each layer consists of a random permutation of the qubits, followed by Haar-random two-qubit gates (from SU(4)) performed on neighbouring pairs of qubits.

Step 2: Execute the QV protocol on the noisy quantum devices. If we cannot execute circuit U perfectly or efficiently with the provided gate set provided by the target system, then we need to resort to a circuit-to-circuit transpiler to find an approximate U' with infidelity $1 - F_{\mathrm{avg}}(U, U') \leq 1$, where

$$F_{\mathrm{avg}}(U, U') = \frac{|Tr(U^\dagger U')|^2/2^m + 1}{2^m + 1} \qquad (72)$$

is the average gate fidelity [440] between m-qubit unitaries U and U'. Arbitrary circuit transpiler techniques can be used to facilitate efficient execution on hardware. Finally, observe the distribution $q_U(x)$ for the implementation U'.

Step 3: Determine whether the result satisfies the heavy output. Calculate the ideal output distribution $p_U(x) = |\langle x|U|0\rangle|^2$ of the circuit U, and sort it in ascending order $p_0 \leq p_1 \leq \cdots \leq p_{2^m-1}$, where $x \in \{0,1\}^m$ is an observable bit-string. The heavy H_U outputs are

$$H_U = \{x \in \{0,1\}\}^m \text{ such that } p_U(x) > p_{\mathrm{med}}\}, \qquad (73)$$

where $p_{\mathrm{med}} = (p_{2^{(m-1)}} + p_{2^{(m-1)}-1})/2$ is the median of the set of probability. The probability that the experimental sampling satisfies the heavy output is:

$$h_U = \sum_{x \in H_U} q_U(x) \qquad (74)$$

If $h_U > 2/3$, the test is passed.

Step 4: Calculate the QV. For m-qubit quantum system, repeat steps 1-3 to get the largest achievable depth $d(m)$ such that

$$h_1, h_2, \ldots, h_{d(m)} > 2/3 \text{ and } h_{d(m)+1} \leq 2/3. \qquad (75)$$

By increasing the system size until $d(m) < m$, we then get the QV V_Q, which is defined as

$$\log_2 V_Q = \arg \max_{\mathrm{m}} \min(m, d(m)) \qquad (76)$$

QV has become a widely used benchmarking tool as it reflects the comprehensive performance of quantum devices. Recently, Quantinuum claimed a 8,192 QV is measured on their trapped-ion quantum computing [441]. IBM Quantum has has achieved a 256 QV on their superconducting quantum processor Falcon r10 [442].

CLOPS [443] is a measure correlated with how many QV circuits a QPU can execute per unit of time, and it is a speed benchmark that serves as a complement to QV. Thus, IBM Quantum team take the number of qubits, QV, and CLOPS as three key attributes to measure the performance of near-term quantum computers, in terms of scale, quality, and speed.

3. Mirror Circuits

Both QV and RQC sampling require predicting the ideal outcomes, whose classical complexity grows exponentially.

Mirror circuits – quantum circuits with a reflection structure – is an efficiently verifiable benchmarking method [444], which does not require huge classical computing resources. Specifically, a mirror circuit consists of: 1) A layer to prepare each qubit in a randomized Pauli eigenstate; 2) The layer of quantum circuit C; 3) A randomly chosen Pauli layer Q; 3) The 'quasi-inverse' circuit C^{-1} so that $C^{-1}QC$ is a Pauli operation. By construction, every mirror circuit has a unique and easy-to-calculate outcome bitstring in the ideal case without noise.

Run the mirror circuit on the noisy quantum devices, and observe the probability S of the bitstring that should be output theoretically. Calculate the polarization

$$p = (S - 1/2^w)/(1 - 1/2^w), \qquad (77)$$

which is a metric represents performance on C, where w is the number of qubits.

4. Application-Oriented Benchmarks

Application-oriented benchmarks aim to measure the usefulness of quantum system at handling actual problems, rather than simply measuring its theoretical performance. Recently, various application-oriented benchmarks have been introduced [445–449]. These benchmarks deal with different practical problems. Inspired by the classical LINPACK benchmark, the quantum LINPACK benchmark [445] is to measure the performance of quantum computers for scientific computing applications by considering the problem of solving quantum linear system problem. The Q-score metric [447] is defined as the size of the largest graph for which the quantum device can approximately solve the Max-Cut problem with a solution that significantly outperforms a random guessing algorithm. While, the SupermarQ [446], QED-C's application-oriented benchmarks [448], and algorithmic qubits [449] are more integrated by executing a series of common quantum algorithms and programs.

VI.   CLASSICAL SIMULATION

In the NISQ era, classical simulators for quantum algorithms are particularly important. On one hand, classical simulators are perfect benchmarking baselines for noisy quantum computers. As an example, classical simulators have been extensively used in RQC sampling experiments, ranging from calibrating the gate-level fidelities and the circuit-level fidelities, to estimating the fidelities of the hardest RQC sampling experiments [7–9]. Moreover, the claim of *quantum supremacy* for RQC sampling also requires an efficient classical simulator to set the performance baseline for a fair comparison. On the other hand, the classical simulators also provide an alternative platform for running quantum algorithms without resorting to an actual quantum computer, which could be very helpful, for example, for exploring near-term quantum algorithms such as VQE, or for testing quantum noise models or quantum error mitigation schemes.

In this section we will first review several important classical algorithms used to implement classical simulators for both noiseless and noisy quantum circuits, and then we will briefly review different ways to compute the gradients of parametric quantum circuits on classical computers.

A.   Simulating noiseless quantum circuits

Here we introduce some definitions and notations which will be used throughout this section. Mathematically, an n-qubit pure quantum state $|\phi\rangle$ can be written as

$$|\phi\rangle = \sum_{\sigma_1,\sigma_2,\ldots,\sigma_n} c_{\sigma_1,\sigma_2,\ldots,\sigma_n} |\sigma_1, \sigma_2, \ldots, \sigma_n\rangle, \qquad (78)$$

where $|\sigma_1, \sigma_2, \ldots, \sigma_n\rangle$ represents a specific computational basis and $c_{\sigma_1,\sigma_2,\ldots,\sigma_n}$ is the coefficient (amplitude) for it. All the coefficients constitute a rank-n tensor, which contains 2^n complex numbers since each σ_l can be 0 or 1. The coefficient tensor is usually equivalently viewed as a long *state vector* of length 2^n.

Generally, a quantum algorithm starts by initializing the n-qubit quantum register in a particular state $|0^{\otimes n}\rangle$ where all the qubits are set to state $|0\rangle$. Then one applies a series of quantum gate operations onto the initial state to obtain the final quantum state $|\psi\rangle$:

$$|\psi\rangle = \hat{\mathcal{C}}|0^{\otimes n}\rangle = \hat{Q}^m \cdots \hat{Q}^1 |0^{\otimes n}\rangle, \qquad (79)$$

where $\hat{Q}^j$ denotes the j-th quantum gate operation, m is the total number of gates and $\hat{\mathcal{C}}$ is an abbreviation for all the gate operations. We will also use Q to denote the tensor which corresponds to the quantum operator $\hat{Q}$. Finally, a quantum algorithm usually ends by measuring the final quantum state $|\psi\rangle$ to obtain the measurement outcomes in the form of bitstrings, which are either used as the final outputs or used to further compute the expectation values of some quantum observables depending on the specific quantum algorithms. We note that a general quantum algorithm could also allow quantum measurements to be performed in between the quantum gate operations, which however adds no difficulty for classical simulators as long as each elementary quantum operation could be simulated.

For classical simulators, one could usually perform more flexible "measurements" compared to quantum computers since the quantum state is often stored in a certain data structure on classical computers which can be easily copied and reused. We will primarily focus on the following three types of measurements that can be performed using classical simulators: 1) computing the amplitude of a given bitstring $\vec{b} = \{b_1, b_2, \ldots, b_n\}$, which is useful to verify the fidelity of the measured bitstrings by noisy quantum computers (the XEB test is one such example); 2) simulating the sampling process, namely generating a number of samples (bitstrings), this could be done as long as one can compute amplitudes for any given bitstrings, for example one could first generate a number of random bitstrings and then perform a rejective sampling to select a subset of them as samples according to their

Schrödinger Algorithm

Schrödinger simulator

Schrödinger-Feynman simulator

Tensor Network Algorithm
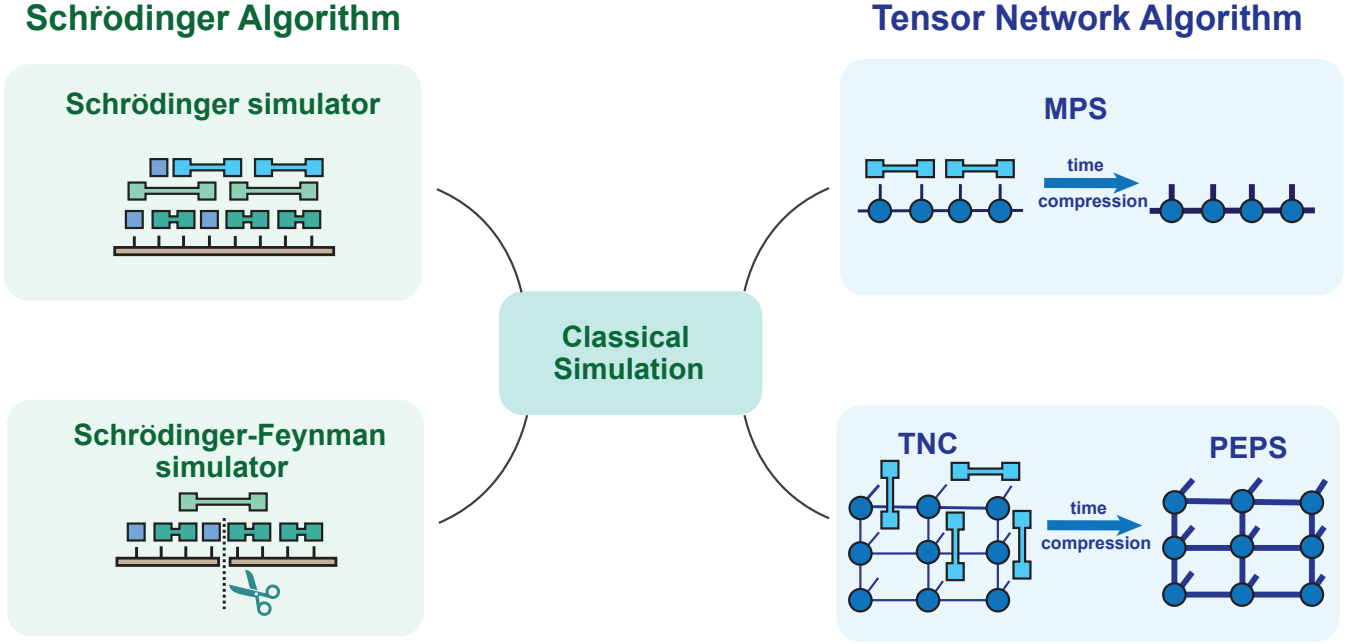
MPS

TNC

PEPS

Classical Simulation

FIG. 9. **Classical algorithms for simulating quantum circuits.** The Schrödinger simulator directly stores the quantum state as a state vector and its memory requirement grows exponentially with the system size. The Schrödinger-Feynman simulator divides the system into two subsystems and store the two smaller quantum states accordingly, the complexity of which grows exponentially with the number of "cross gates" acting on both subsystems simultaneously. In tensor network based algorithms the quantum state (and the quantum circuit) are indirectly stored as a tensor network. Concretely, the matrix product states based simulator represents the quantum state as an MPS, which is a special instance of the one-dimensional tensor network. The tensor network contraction algorithm represents the quantum state together with the quantum circuit as a whole tensor network. Pre-contracting the tensor network formed in the TNC algorithm in the time direction will result in a projected-entangled pair state (assuming that the underlying quantum circuit has a clear two-dimensional structure, which are physically motivated by the geometry of the current quantum processors).

amplitudes [450]; and 3) computing the expectation values of some quantum observables (Pauli strings for example). We note that although expectation values could be computed in a similar way to quantum computers as long as one can generate samples, for classical simulators there often exist shortcuts in which they can be directly computed without sampling errors, and these exact expectation values could be used as perfect references for noisy quantum computers.

1. Schrödinger simulator

The most straightforward approach to simulate quantum circuits is the so-called Schrödinger algorithm, which stores a quantum state $|\phi\rangle$ on a classical computer by directly storing its coefficient tensor $c_{\sigma_1,\sigma_2,...,\sigma_n}$ as a state vector [451–454]. As a result the amount of memory required to store an n-qubit quantum state grows exponentially with n. To given some explicit numbers of the memory complexity using the Schrödinger simulator (we assume that single precision complex numbers are used), it takes around 1 GB memory for $n = 27$, around 0.5 TB memory for $n = 36$, and around 2 PB memory for $n = 48$ which is already beyond the reach of most of the existing supercomputers in the world! In fact till

now the largest classical simulation based on the Schrödinger simulator has only reached 45 qubits which uses about 0.5 PB memory [453].

For the Schrödinger simulator, the initialization is straightforward since the full quantum state is stored in memory. One can simply set the amplitude corresponding to the computational basis $|0,\ldots,0\rangle$ to 1 and the amplitudes for the rest to be 0. The quantum gate operations on the state vector can be simulated by following their mathematical definitions. For example a single-qubit gate operation acting on the i-th qubit, denoted as $Q_{\sigma_i}^{\sigma_i'}$, can be simulated as

$$c_{\sigma_1,...,\sigma_i',...,\sigma_n} = \sum_{\sigma_i} Q_{\sigma_i}^{\sigma_i'} c_{\sigma_1,...,\sigma_i,...,\sigma_n}, \qquad (80)$$

and a two-qubit gate operation acting on the two qubits i and j ($1 \leq i < j \leq n$), denoted as $Q_{\sigma_i,\sigma_j}^{\sigma_{i'},\sigma_{j'}}$, can be simulated as

$$c_{\sigma_1,...,\sigma_i',...,\sigma_j',...,\sigma_n} = \sum_{\sigma_i,\sigma_j} Q_{\sigma_i,\sigma_j}^{\sigma_i',\sigma_j'} c_{\sigma_1,...,\sigma_i,...,\sigma_j,...,\sigma_n}. \quad (81)$$

Eqs.(80,81) are simply tensor contractions. However, directly implementing them as tensor contractions on classical computers would be extremely slow for large n since tensor contraction would generally involve copies of the tensors (a standard implementation for tensor contraction is to first permute

the tensor indices of each tensors and then do matrix multiplication, where the tensors will be copied during the tensor index permutation). Additionally, in actual implementation one usually prefers to implement the gate operations in an inplace fashion to reduce the memory usage, namely one directly uses the result to overwrite the input tensor for the quantum state instead of generating a new one.

The memory-efficient way to implement Eqs.(80,81) is to directly follow the definition of tensor contraction, that is, iterating over all the uncontracted indices and performing inplace operations for the contracted indices inside each iteration. Taking the two-qubit gate operation in Eq.(81) as a concrete example, one could first group all the uncontracted indices into three contiguous sets to reduce the number of for loops, which results in a rank-5 tensor denoted as $\tilde{c}$:

$$\tilde{c}_{(\sigma_1,...,\sigma_{i-1}),\sigma_i,(\sigma_{i+1},...,\sigma_{j-1}),\sigma_j,(\sigma_{j+1},...,\sigma_n)} = c_{\sigma_1,...,\sigma_n}, \tag{82}$$

where the set of indices insider the parentheses mean that they are treated together as a single index. We note that in real coding nothing needs to be done in Eq.(82), it is just an equivalent way to view the same state vector. The tensor $\tilde{c}$ have a shape $2^{i-1} \times 2 \times 2^{j-i-1} \times 2 \times 2^{n-j}$, and $Q_{\sigma_i,\sigma_j}^{\sigma_{i'},\sigma_{j'}}$ can be applied onto $\tilde{c}$ using three for loops as shown in Algorithm. 1.

Algorithm 1 Implementation of a two-qubit gate operation $Q_{\sigma_i,\sigma_j}^{\sigma_{i'},\sigma_{j'}}$ on an n-qubit quantum state stored as a state vector. Column-major storage and zero-based indexing are assumed for all the tensors.

1: **for** $s_3 = 0 : 2^{n-j} - 1$ **do**
2:     **for** $s_2 = 0 : 2^{j-i-1} - 1$ **do**
3:         **for** $s_1 = 0 : 2^{i-1} - 1$ **do**
4:             $\tilde{c}[s_1, \sigma_i', s_2, \sigma_j', s_3] = \sum_{\sigma_i,\sigma_j} Q_{\sigma_i,\sigma_j}^{\sigma_{i'},\sigma_{j'}} \times \tilde{c}[s_1, \sigma_i, s_2, \sigma_j, s_3]$
5:         **end for**
6:     **end for**
7: **end for**

We note that the operation inside the for loops of Algorithm. 1 is simply a 4×4 matrix times a vector of size 4. The overall floating number operations in Algorithm. 1 is thus $O(4 \times 2^n)$. At the same time one needs to fetch all the elements of the state vector from the main memory into the CPU cache and then send the results back after the computation for at least once, namely the amount of memory access is $O(2 \times 2^n)$. As a result the number of memory access compared to the number of floating point arithmetic is roughly $1/2$, while for modern classical computing hardware the memory bandwidth is usually much lower compared to the Flops efficiency. For example the V100 GPU has a memory bandwidth of 1.9 TB/s but has a single-precision performance of 19.5 TFlops (the former is only about 10% of the latter). Therefore the efficiency of the Schrödinger simulator is essentially bounded by the memory bandwidth. Nevertheless, one could slightly increase the compute density by unrolling the inner most for loop such that several vectors of size 4 are processed simultaneously, that is, several matrix-vector multiplications are grouped together into a single matrix-matrix mul-

tiplication to optimize the usage the CPU cache (one would also make sure that the sizes of the matrices exactly fit into the CPU cache size for the best performance). The loop unrolling will also have an another advantage that one could fetch a bunch of numbers which are contiguous in memory simultaneously, which is an operation that is vectorized by most modern computing hardware. Additionally, the two outer for loops in Algorithm. 1 can be parallelized straightforwardly on a shared-memory architecture. The sparsity of gate operations could also be explored to further speed up the calculation, for example, the control-control-Z gate would only have a single non-trivial operation for its inner matrix-vector multiplication, which means that applying such a three-qubit gate would have a complexity which is only $1/8$ of the complexity of a dense single-qubit gate operation.

After all the gate operations have been performed, one obtains the final state $|\psi\rangle$ in the form of a state vector. No additional computations need to be done to obtain the amplitudes since they have been directly stored. The expectation value of a local observable $\hat{O}^i$ on the i-th qubit, whose corresponding tensor is $O_{\sigma_i}^{\sigma_i'}$, can be computed as

$$\langle\psi|\hat{O}^i|\psi\rangle = \sum_{\sigma_1,...,\sigma_{i-1},\sigma_i,\sigma_i',\sigma_{i+1},...,\sigma_n} c_{\sigma_1,...,\sigma_i,...,\sigma_n} \times O_{\sigma_i}^{\sigma_i'} c_{\sigma_1,...,\sigma_i',...,\sigma_n}^*. \tag{83}$$

One way to implement Eq.(83) is to treat $\hat{O}^i$ as a single-qubit operator and apply it onto $|\psi\rangle$ to obtain $\hat{O}^i|\psi\rangle$, and then compute the dot product between $|\psi\rangle$ and $\hat{O}^i|\psi\rangle$. The advantage of this approach is that it could easily be generalized to more general observables such as a general Pauli string written as

$$\hat{P} = M^1 \otimes M^2 \otimes \cdots \otimes M^n, \tag{84}$$

where each $M^j \in \{X, Y, Z, I_2\}$ (X, Y, Z are Pauli matrices and I_2 is the 2×2 identity matrix) is a local matrix, for which one can simply apply M^j sequentially onto $|\psi\rangle$ to obtain $\hat{P}|\psi\rangle$ and then compute the dot product between $|\psi\rangle$ and $\hat{P}|\psi\rangle$. The disadvantage is that one has to use an additional copy of the quantum state to store the intermediate state $\hat{O}^i|\psi\rangle$ or $\hat{P}|\psi\rangle$. A better approach to take is to directly evaluate Eq.(83) using several for loops similar to Algorithm. 1, for which no copy of the quantum state is required, however for a general Pauli string this approach would require n for loops which is likely to be less efficient.

As long as one can compute the expectation of a local observable and can apply gate operations onto the quantum state for some classical simulator, one is able to faithfully simulate the quantum measurement process (the following procedures to simulate quantum measurement are valid for all classical simulators which support those operations). Assuming that one wants to simulate a local quantum measurement on the i-th qubit, one can first compute the expectation value of the local operator $|1_i\rangle\langle 1_i|$ which is the probability that the i-th qubit is in state 1, then one generates a random number p according to the uniform distribution and outputs 1 if $p < \langle\psi|1_i\rangle\langle 1_i|\psi\rangle$

and outputs 0 otherwise. The "collapse" the quantum state after measurement can be simulated by applying a local operator $|1_i\rangle\langle 1_i|$ onto $|\psi\rangle$ if 1 is obtained from the measurement, or applying $|0_i\rangle\langle 0_i|$ if 0 is obtained (the resulting quantum state should also be normalized by the corresponding probability).

An important advantage of the Schrödinger simulator is that its computational time scales strictly linearly against the number of gate operations (therefore it would also be beneficial to perform gate fusion at the beginning to absorb smaller gates into larger ones to reduce the total number of gate operations). There already exists several excellent libraries which are based on the Schrödinger simulator, such as ProjectQ [455], Yao [456], Qiskit [360], Quest [457], Qulacs [458].

2. *Schrödinger-Feynman simulator*

In the Schrödinger algorithm the whole quantum state is directly stored in memory. There is another extreme, the path-integral (Feynman) algorithm, which does not store any quantum state at all. In the Feynman algorithm, the amplitude of a specific bitstring $\vec{b}$ is computed as

$$
\begin{aligned}
\langle\vec{b}|\psi\rangle &= \langle\vec{b}|\hat{Q}^m \dots \hat{Q}^1|0^{\otimes n}\rangle \\
&= \sum_{\vec{x}_1,\dots,\vec{x}_{m-1}} \langle\vec{b}|\hat{Q}^m|\vec{x}_{m-1}\rangle\langle\vec{x}_{m-1}|\dots|\vec{x}_1\rangle\langle\vec{x}_1|\hat{Q}^1|0^{\otimes n}\rangle,
\end{aligned}
$$
(85)

where each $\vec{x}_j$ denotes a specific computational basis. To evaluate Eq.(85), all one needs are terms $\langle x_j|\hat{Q}^j|x_{j-1}\rangle$ which are simply the entries of the tensor Q^j. Therefore the quantum state is never explicitly stored in the Feynman algorithm. Each combination $\{\vec{x}_1,\dots,\vec{x}_{m-1}\}$ specifies a particular path and the total number of different paths (thus the time complexity) scales exponentially with the number of gate operations m, while in the meantime the memory complexity is negligible. The Feynman algorithm is of course impractical for any quantum circuit containing a few hundreds of gate operations due to its scaring time complexity.

The Schrödinger-Feynman algorithm aims to take advantages of both the Schrödinger algorithm and the Feynman algorithm [450]. Concretely, it divides the whole system into two subsystems and stores two quantum states corresponding to the two subsystems instead of storing the global quantum state. For the Sycamore quantum processor with 53 qubits, one only needs to store two quantum states of 26 and 27 qubits respectively (one could of source increase the number of subsystems if there is not enough memory on the classical computer to store the quantum states of the subsystems). Once the partition is fixed, the gate operations acting on qubits which belong to one subsystem can be simply applied onto the quantum state corresponding to this subsystem. For gate operations which act on qubits from both subsystems, denoted as $\hat{Q}^{g_l}$ ($1 \le l \le m_c$ with m_c the total number of such cross gates), one could first decompose it into the following tensor product form (the details of the decomposition will be shown later)

$$
\hat{Q}^{g_l} = \sum_{s_l} \hat{U}_{s_l}^{g_l} \otimes \hat{V}_{s_l}^{g_l},
$$
(86)

where the operators $\hat{U}_{s_l}^{g_l}$ and $\hat{V}_{s_l}^{g_l}$ act on the two subsystems respectively for each s_l, then the amplitude in Eq.(85) can be computed as

$$
\begin{aligned}
\langle\vec{b}|\psi\rangle = \sum_{s_1,\dots,s_{m_c}} \langle\vec{b}|\dots\hat{U}_{s_2}^{g_2}\hat{V}_{s_2}^{g_2}(\hat{Q}^{g_2-1}\dots\hat{Q}^{g_1+1})\times \\
\times \hat{U}_{s_1}^{g_1}\hat{V}_{s_1}^{g_1}(\hat{Q}^{g_1-1}\dots\hat{Q}^1)|0^{\otimes n}\rangle.
\end{aligned}
$$
(87)

The operators in the summand of Eq.(87) act either on one subsystem or the other subsystem, therefore one only needs to store the two quantum states of the two subsystems. Compared to Eq.(85), the total number of paths in Eq.(87) will only grow exponentially against the total number of cross gates, namely m_c, instead of m. One could also evaluate the expectation values of observables (local operators or Pauli strings) similar to Eq.(87), however the number of possible paths will generally be squared.

From Eq.(87), the different paths specified by $\{s_1,\dots,s_{m_c}\}$ can be computed in parallel with no data communication, therefore the Schrödinger-Feynman algorithm is extremely friendly for large-scale parallelization on distributed architectures. Moreover, one could easily use the Schrödinger-Feynman algorithm to simulate the simplified quantum circuits by removing a number of cross gates from the original quantum circuit. For these reasons it is used as the first classical benchmarking baseline to characterize *quantum supremacy* for the Sycamore quantum processor [7].

3. *Matrix product state based simulator*

Tensor network states based algorithms belong to an important class of algorithms which could often leverage the exponential memory requirement when simulating quantum algorithms. Generally, in tensor network states based simulators one represents the underlying quantum state as a tensor network which only consists of low-rank tensors, then the gate operations are simulated by updating the local tensors of the tensor network been acted on and quantum measurements could be simulated by contracting some tensor network.

Tensor network states are originally introduced to solve quantum many-body problems, which are mostly used to approximate the low-energy states of quantum many-body Hamiltonians [459]. Depending on the structure of the physical problem, various tensor network ansatz have been used to represent the underlying quantum state, such as matrix product states (MPS) [460–462], projected entangled pair states (PEPS) [463], tree tensor network (TTN) [464], multiscale entanglement renormalization ansatz (MERA) [465]. In particular, MPS and PEPS are designed for one-dimensional and two-dimensional quantum many-body systems respectively [459, 466], and have achieved great success therein for the past thirty years.

Out of all tensor network states, MPS is a very special and important one, mostly because that the errors in MPS based algorithms can be often well controlled and that it allows very efficient and stable ground state searching [467, 468] and time evolution algorithms [460–462]. Therefore although MPS is mostly designed to represent the ground states of one-dimensional quantum many-body systems, it could also be used as a general-purpose ansatz similar to the state vector for arbitrary quantum states (in the general case the cost of MPS could also grow exponentially) [469]. MPS based simulator has been used to simulate Shor's algorithm up to 60 qubits [470, 471], to simulate (noisy) optical quantum circuits [472], to simulate the Sycamore RQCs [473], and to simulate VQE algorithms for quantum chemistry problems with up to 100 qubits [474]. Some typical simulations of VQE using the Schrödinger simulator and the MPS based simulator are listed in TABLE. II.

TABLE II. Typical simulations of molecular systems with classical simulators. The number of atoms (N_a), number of qubits (N_q), the estimated number of CNOT gates (N_{CNOT}), and the algorithms (Alg.) used are listed for comparison. SA is a shorthand for the Schrödinger algorithm.

Work	System	N_a	N_q	N_{CNOT}	Alg.
Microsoft QDK [475]	H_2	2	4	64	
Cirq [476]	CH_2O	4	6	272	
Qulacs [477]	He crystal	1	8	1.6×10^3	
Qiskit [478]	N_2	2	16	2.7×10^4	
Yao.jl [479]	C_{18}	18	16	5.4×10^4	SA
VQEChem [480]	H chain	2	16	5.4×10^4	
QCQC [481]	Si crystal	2	16	1.1×10^5	
Tequila [482]	BH	2	22	1.6×10^5	
HiQ [483]	C_2H_4	6	28	8.6×10^5	
iQCC-VQE [484]	Ir(ppy)$_3$	61	56	~ 60	
Differentiable MPS [485]	CO_2	3	30	3.7×10^4	
MPS-VQE [474]	H_2	2	92	1.4×10^5	MPS
MPS-VQE [474]	C_2H_6	8	32	4.4×10^5	

To simulate quantum circuits based on the MPS representation of the quantum state, one could directly use an MPS based time evolution algorithm such as time evolving block decimation (TEBD) [461]. In the following we will introduce a variant of TEBD to enhance the stability of the algorithm. MPS rewrites the rank-n coefficient tensor $c_{\sigma_1,\dots,\sigma_n}$ in Eq.(78) as the product of a chain of rank-3 tensors as

$$c_{\sigma_1,\sigma_2,\dots,\sigma_n} = \sum_{a_0,\dots,a_n} B^{\sigma_1}_{a_0,a_1} B^{\sigma_2}_{a_1,a_2} \dots B^{\sigma_n}_{a_{n-1},a_n}, \quad (88)$$

where $\sigma_j \in \{0, 1\}$ is the "physical" index and a_j the "auxiliary" index. The indices a_0 and a_n at the boundaries are trivial indices added for notational convenience. The size of an MPS can be conveniently characterized by the largest size of the auxiliary indices

$$D = \max_{1 \le j \le n-1} (\dim(a_j)), \quad (89)$$

which is usually referred to as the *bond dimension* of MPS. In principle, Eq.(88) is able to represent any quantum states exactly if D increases exponentially. A quantum state is said to be efficiently representable as an MPS if D is nearly a constant as n grows. Generally, for MPS based algorithms the memory complexity scales as $O(nD^2)$ and the computational complexity scales as $O(nD^3)$.

Slightly different from the TEBD algorithm, we assume that the MPS is prepared in a right-canonical form, where each site tensor satisfies the right-canonical condition

$$\sum_{\sigma_j, a_j} (B^{\sigma_j}_{a'_{j-1}, a_j})^* B^{\sigma_j}_{a_{j-1}, a_j} = \delta_{a_{j-1}, a'_{j-1}}. \quad (90)$$

We also assume that the bipartition singular vectors, denoted as λ_{a_j}, which are the Schmidt numbers when splitting the system into two subsystems from qubits 1 to j and from qubits $j+1$ to n, are also stored. The initial state $|0^{\otimes n}\rangle$ can be easily written as an MPS with $D = 1$, for which each site tensor satisfies

$$B^{\sigma_j=0}_{a_{j-1}=0,a_j=0} = 1, B^{\sigma_j=1}_{a_{j-1}=0,a_j=0} = 0, \quad (91)$$

(In fact any separable state can be written as an MPS with $D = 1$ similarly). The corresponding singular vectors are simply initialized as $\lambda_{a_j=0} = 1$. It is straightforward to verify that the site tensors initialized in this way satisfy Eq.(90) and that the singular vectors are the correct Schmidt numbers. The gate operations are then applied sequentially onto the MPS. A single-qubit gate operation acting on the j-th qubit, denoted as $Q^{\sigma'_j}_{\sigma_j}$, changes the j-th site tensor of the MPS as

$$\tilde{B}^{\sigma'_j}_{a_{j-1},a_j} = \sum_{\sigma_j} Q^{\sigma'_j}_{\sigma_j} B^{\sigma_j}_{a_{j-1},a_j}. \quad (92)$$

Since $Q^{\sigma'_j}_{\sigma_j}$ is unitary, it is straightforward to verify that $\tilde{B}^{\sigma'_j}_{a_{j-1},a_j}$ will satisfy Eq.(90) as long as $B^{\sigma_j}_{a_{j-1},a_j}$ does. For a nearest-neighbour two-qubit gate $Q^{\sigma'_j,\sigma'_{j+1}}_{\sigma_j,\sigma_{j+1}}$ acting on two qubits j and $j+1$ (the j-th bond), it will change the both the j and $j+1$-th site tensors of the MPS. To simulate a two-qubit gate operation while preserving the right canonical property of the underlying MPS, we use the technique first introduced in Ref. [486], which is shown as follows. First we contract the two site tensors $B^{\sigma_j}_{a_{j-1},a_j}$ and $B^{\sigma_{j+1}}_{a_j,a_{j+1}}$ with $Q^{\sigma'_j,\sigma'_{j+1}}_{\sigma_j,\sigma_{j+1}}$ to get a two-site tensor

$$C^{\sigma'_j,\sigma'_{j+1}}_{a_{j-1},a_{j+1}} = \sum_{a_j,\sigma_j,\sigma_{j+1}} Q^{\sigma'_j,\sigma'_{j+1}}_{\sigma_j,\sigma_{j+1}} B^{\sigma_j}_{a_{j-1},a_j} B^{\sigma_{j+1}}_{a_j,a_{j+1}}. \quad (93)$$

Then we contract $C^{\sigma'_j,\sigma'_{j+1}}_{a_{j-1},a_{j+1}}$ with the singular vector $\lambda_{a_{j-1}}$ at the $j-1$-th bond to get a new two-site tensor as

$$\tilde{C}^{\sigma'_j,\sigma'_{j+1}}_{a_{j-1},a_{j+1}} = \lambda_{a_{j-1}} C^{\sigma'_j,\sigma'_{j+1}}_{a_{j-1},a_{j+1}}. \quad (94)$$

Now we perform singular value decomposition (SVD) on the tensor $\tilde{C}^{\sigma'_j,\sigma'_{j+1}}_{a_{j-1},a_{j+1}}$ and get

$$\mathrm{SVD}(\tilde{C}^{\sigma'_j,\sigma'_{j+1}}_{a_{j-1},a_{j+1}}) = \sum_{a_j} U^{\sigma'_j}_{a_{j-1},a_j} \tilde{\lambda}_{a_j} V^{\sigma'_{j+1}}_{a_j,a_{j+1}}, \quad (95)$$

during which we will also truncate the small singular values below a certain threshold ϵ or simply reserve the D largest singular values if the singular values larger than ϵ is more than D. Here we note that this truncation step is the only step that would introduce errors in the algorithm, and such errors could be made arbitrarily small in principle if one sets ϵ to be very small and D to be very large (the actual size of D required is of course problem-dependent). After the SVD, the updated site tensors $\tilde{B}^{\sigma'_j}_{a_{j-1},a_j}$ and $\tilde{B}^{\sigma'_{j+1}}_{a_j,a_{j+1}}$ can be obtained as

$$\tilde{B}^{\sigma'_j}_{a_{j-1},a_j} = \sum_{\sigma'_{j+1},a_{j+1}} C^{\sigma'_j,\sigma'_{j+1}}_{a_{j-1},a_{j+1}} \left(V^{\sigma'_{j+1}}_{a_j,a_{j+1}}\right)^*; \quad (96)$$

$$\tilde{B}^{\sigma'_{j+1}}_{a_j,a_{j+1}} = V^{\sigma'_{j+1}}_{a_j,a_{j+1}}. \quad (97)$$

To verify that $\tilde{B}^{\sigma'_j}_{a_{j-1},a_j}$ and $\tilde{B}^{\sigma'_{j+1}}_{a_j,a_{j+1}}$ are indeed the solutions, we can see that $\sum_{a_j} \tilde{B}^{\sigma'_j}_{a_{j-1},a_j} \tilde{B}^{\sigma'_{j+1}}_{a_j,a_{j+1}} = C^{\sigma'_j,\sigma'_{j+1}}_{a_{j-1},a_{j+1}}$ up to truncation errors. The reason that one does not directly do SVD on $C^{\sigma'_j,\sigma'_{j+1}}_{a_{j-1},a_{j+1}}$ to get the updated site tensors is to preserve the right canonical properties of the solutions, in fact it could be verified that $\tilde{B}^{\sigma'_j}_{a_{j-1},a_j}$ and $\tilde{B}^{\sigma'_{j+1}}_{a_j,a_{j+1}}$ both satisfy Eq.(90) ($\tilde{B}^{\sigma'_{j+1}}_{a_j,a_{j+1}}$ is right canonical by construction from Eq.(95)). The new singular vector $\tilde{\lambda}_{\alpha_j}$ contains the correct Schmidt numbers on bond j of the updated MPS and is used to replace the old λ_{α_j}. Similar to the Schrödinger simulator, one can also absorb the single-qubit gate operations into two-qubit gate operations before hand, but the speedup will not be as significant since the complexity of a single-qubit gate operations is negligible compared to a two-qubit gate for MPS based simulator. A three-qubit gate which acts on three contiguous qubits could also be simulated similarly, where the three site tensors as well as the two singular vectors in between will be updated. A non-nearest-neighbour gate operation could be simulated by decomposing it into a series of nearest-neighbour gate operations using the SWAP gate.

With a right-canonical MPS, there is a very efficient way to compute the expectation of a local operator or a Pauli string. For example, the expectation value of a single-qubit observable $\hat{O}^j$ on the j-th qubit can be simply computed as

$$\langle\psi|\hat{O}^j|\psi\rangle = \sum_{a_{j-1},a_j,\sigma_j,\sigma'_j} \lambda^2_{a_{j-1}} O^{\sigma'_j}_{\sigma_j} B^{\sigma'_j}_{a_{j-1},a_j}(B^{\sigma_j}_{a_{j-1},a_j})^*, \quad (98)$$

and the expectation value of a generic two-qubit observable $\hat{O}^{i,j}$ on the i and j-th qubits ($i < j$) can be computed as

$$\langle\psi|\hat{O}^{i,j}|\psi\rangle = \sum_{a_{j:i-1},\sigma_{j:i},\sigma'_{j:i}} \lambda^2_{a_{i-1}} O^{\sigma'_i,\sigma'_j}_{\sigma_i,\sigma_j} B^{\sigma'_i}_{a_{i-1},a_i}(B^{\sigma_i}_{a_{i-1},a_i})^* \times$$
$$\cdots \times B^{\sigma'_j}_{a_{j-1},a_j}(B^{\sigma_j}_{a_{j-1},a_j})^*, \quad (99)$$

where we have used $x_{j:i} = \{x_i, x_{i+1}, \ldots, x_j\}$ as an abbreviation for a list of indices. The expectation value of a general

n-qubit Pauli string could be computed similarly, with a complexity that scales as $O(nD^3)$.

After one obtains the quantum state $|\psi\rangle$ in MPS form, the amplitude of a given bitstring $\vec{b}$ can be computed as

$$\langle\vec{b}|\psi\rangle = \sum_{a_0,\ldots,a_n} B^{\sigma_1=b_1}_{a_0,a_1} B^{\sigma_2=b_2}_{a_1,a_2} \ldots B^{\sigma_n=b_n}_{a_{n-1},a_n}, \quad (100)$$

for which only $O(n)$ matrix-vector multiplications need to be performed. Given the ability to efficiently compute the amplitude, one could also simulate the sampling problem by further using a sampling algorithm. However, sampling an MPS can be done exactly and more efficiently [487]. Given an MPS in the right-canonical form as in Eq.(88), the reduced density matrix for qubits from 1 to j can be simply computed as

$$\rho^{\sigma'_1,\ldots,\sigma'_j}_{\sigma_1,\ldots,\sigma_j} = \sum_{a_{j-1:0},a'_{j-1:0},a_j} (B^{\sigma'_1}_{a_0,a'_1})^* B^{\sigma_1}_{a_0,a_1} \times \cdots$$
$$\times (B^{\sigma'_{j-1}}_{a'_{j-2},a'_{j-1}})^* B^{\sigma_{j-1}}_{a_{j-2},a_{j-1}} (B^{\sigma'_j}_{a'_{j-1},a_j})^* B^{\sigma_j}_{a_{j-1},a_j}. \quad (101)$$

To generate a single sample (bitstring), one could first compute the reduced density matrix of the first qubit, $\rho^{\sigma'_1}_{\sigma_1}$, using Eq.(101), and then one could perform a local sampling on $\rho^{\sigma'_1}_{\sigma_1}$ to get the sampling output for the first qubit, denoted as ν_1. After that, the reduced density matrix of the second qubit conditioned on the sampling outcome of the first qubit, denoted as $\rho^{\sigma'_2}_{\sigma_2}|_{\nu_1}$, can be computed as

$$\rho^{\sigma'_2}_{\sigma_2}|_{\nu_1} = \frac{1}{p(\nu_1)} \sum_{a_{1:0},a'_{1:0},a_2} (B^{\sigma'_1=\nu_1}_{a'_0,a'_1})^* B^{\sigma_1=\nu_1}_{a_0,a_1} (B^{\sigma'_2}_{a'_1,a_2})^* B^{\sigma_2}_{a_1,a_2}, \quad (102)$$

where $p(\nu_1) = \rho^{\sigma'_1=\nu_1}_{\sigma_1=\nu_1}$ is the probability to get ν_1 when measuring the first qubit. By a local sampling on $\hat{\rho}^{\sigma'_2}_{\sigma_2}|_{\nu_1}$ on could obtain the sampling outcome of the second qubit, denoted as ν_2. Repeating this process from left to right, one can obtain one sample $\vec{\nu} = \{\nu_1, \ldots, \nu_n\}$. The complexities of computing one amplitude and generating one sample both scale as $O(nD^2)$.

4. Projected entangled pair states based simulator

PEPS is a natural extension of MPS to two-dimensional systems [463]. Currently, the mainstream quantum processors all have a two-dimensional geometry, therefore PEPS could be a natural tensor network ansatz to represent quantum states generated on those quantum processors. A PEPS on a square lattice can be written as

$$|\psi\rangle = \sum_{\sigma_1,\sigma_2,\ldots,\sigma_n} \mathcal{F}(A^{\sigma_1}_1 A^{\sigma_2}_2 \ldots A^{\sigma_n}_n)|\sigma_1, \sigma_2, \ldots, \sigma_n\rangle, \quad (103)$$

where each $A^{\sigma_j}_j$ is a short hand for the rank-5 site tensor $A^{\sigma_j}_{l,r,u,d}$ with a physical index σ_j and 4 auxiliary indices

l, r, u, d representing the connecting of this tensor with the ones on the left, right, upper and lower respectively. $\mathcal{F}(\ldots)$ means to contract the pairs of connected auxiliary indices. Similar to MPS, the size of a PEPS can also be characterized by its *bond dimension* defined as the largest size of its auxiliary indices, which we will still denote as D. The memory complexity to store a PEPS scales as $O(nD^4)$.

The initial state $|0^{\otimes n}\rangle$ can be easily constructed as a PEPS with $D = 1$, with site tensors satisfying

$$A_{l=0,r=0,u=0,d=0}^{\sigma_j=0} = 1, \quad A_{l=0,r=0,u=0,d=0}^{\sigma_j=1} = 0. \tag{104}$$

To simulate the gate operations on a PEPS, one needs to properly update the site tensors of the PEPS. However for a PEPS the updating is much less straightforward than for an MPS. In the context of quantum many-body physics, many sophisticated methods have been proposed to approximately update the site tensors of a PEPS after a gate operation has been applied on the PEPS, such that the bond dimension of the updated PEPS is bounded by a given bond dimension D [488–491]. Here we will only present the method used in Ref. [41] to simulate RQCs, which allows D to grow after each gate operation and is essentially exact.

For single-qubit gate operation $Q_{\sigma_j}^{\sigma_j'}$ acting on the j-th qubit, it only updates the j-th site tensor of the PEPS as

$$\tilde{A}_{l,r,u,d}^{\sigma_j} = \sum_{\sigma_j'} Q_{\sigma_j}^{\sigma_j'} A_{l,r,u,d}^{\sigma_j'}. \tag{105}$$

For a two-qubit gate operation $Q_{\sigma_i,\sigma_j}^{\sigma_i',\sigma_j'}$, one first splits it into the product of two rank-3 tensors using SVD as in Eq.(86). Concretely, this can be done by first decomposing it as

$$\mathrm{SVD}(Q_{\sigma_i,\sigma_j}^{\sigma_i',\sigma_j'}) = \sum_s \tilde{U}_{\sigma_i,s}^{\sigma_i'} \lambda_s \tilde{V}_{s,\sigma_j}^{\sigma_j'}, \tag{106}$$

then Eq.(86) can be satisfied by choosing the tensors U and V as

$$U_{\sigma_i,s}^{\sigma_i'} = \sum_s \tilde{U}_{\sigma_i,s}^{\sigma_i'} \sqrt{\lambda_s}; \tag{107}$$

$$V_{s,\sigma_j}^{\sigma_j'} = \sum_s \sqrt{\lambda_s} \tilde{V}_{s,\sigma_j}^{\sigma_j'}. \tag{108}$$

Assuming that the qubits i and j are a pair of nearest-neighbour qubits in the horizontal direction, then the site tensors $A_i^{\sigma_i}$ and $A_j^{\sigma_j}$ can be updated as

$$\tilde{A}_{l,r',u,d}^{\sigma_i'} = \sum_{\sigma_i} U_{\sigma_i,s}^{\sigma_i'} A_{l,r,u,d}^{\sigma_i}; \tag{109}$$

$$\tilde{A}_{l',r,u,d}^{\sigma_j'} = \sum_{\sigma_j} V_{s,\sigma_j}^{\sigma_j'} A_{l,r,u,d}^{\sigma_j}, \tag{110}$$

where we have used $r' = (r, s)$ and $l' = (s, l)$. We note that the size of the right auxiliary index of the site tensor $A_i^{\sigma_i}$ (and the left auxiliary index of the site tensor $A_j^{\sigma_j}$) will be enlarged by χ times after updating where χ is the number of nonzero

Schmidt number in Eq.(106). A two-qubit gate operation on a vertical pair of qubits can be done similarly by updating the auxiliary indices in the vertical directions (d and u) instead.

After one obtains the final quantum state $|\psi\rangle$ as a PEPS, the amplitude of a specific bitstring $\vec{b}$ can be computed as

$$\langle \vec{b} | \psi \rangle = \mathcal{F}(A_1^{\sigma_1=b_1} A_2^{\sigma_1=b_2} \cdots A_n^{\sigma_n=b_n}), \tag{111}$$

which amounts to contracting a two-dimensional tensor network with no output indices. For an arbitrary PEPS, the contraction of Eq.(111) is a hard problem. Nevertheless in the context of quantum many-body physics there has been quite a few approximate methods to contract a two-dimensional tensor network similar to Eq.(111), which are fairly efficient and accurate [490, 492–497]. However the two dimensional tensor networks formed in quantum many-body problems usually has a much clear physical origin, such as resulting from computing a local observable on a PEPS which represents the ground state of a local Hamiltonian. When applied to computing amplitudes of RQCs, the physical context is not as clear, and the approximate methods to contract Eq.(111) could easily result in significant errors. Therefore in Ref. [41] Eq.(111) is contracted exactly. For a rectangular lattice with size $n_h \times n_v$, it is estimated that the time complexity $\mathcal{C}^t$ of directly contracting Eq.(111) is

$$\mathcal{C}^t = (n_h - 2)(n_v - 2)D^{\min(n_h,n_v)+3}. \tag{112}$$

One could also compute local observables similar to Eq.(111), however the resulting two-dimensional tensor network will have a bond dimension D^2 instead, for which one may have to resort to those approximate methods. The PEPS simulator could also be easily adapted for other two-dimensional geometries [42]. Additionally, the algorithms used in Ref. [498, 499] are essentially equivalent to the PEPS algorithm presented here, although they are not presented using the PEPS language.

Currently, PEPS based simulator has only been applied to simulate the RQC sampling problem. For RQC on a square lattice, the largest simulation scale till now is reached by using the PEPS based simulator, where one amplitude of a 10×10 RQC with depth 40 is computed within 60 hours [43].

Besides the RQC sampling problem, PEPS based simulator is also promising for simulating other quantum algorithms which are designed for two-dimensional architectures, such as VQE which uses a physically motivated ansatz. In such situations one may also explore those approximate PEPS updating and contraction algorithms to significantly speedup the calculations.

5. *Tensor network contraction based simulator*

Instead of representing the quantum state as a tensor network state and then applying the gate operations sequentially onto it, the quantum circuit itself, as in Eq.(79), can be naturally viewed as a large tensor network, in which the initial state $|0^{\otimes n}\rangle$ is simply the tensor product of n vectors

$|0\rangle = [1, 0]$, each single-qubit gate operation is a rank-2 tensor and each two-qubit gate operation is a rank-4 tensor [500]. There are n uncontracted (output) indices in the end of the tensor network which correspond to the final quantum state. Contracting this tensor network one would directly obtain the final quantum state $|\psi\rangle$ as a state vector, which is of source only feasible for very small problems. Depending on the specific task to do, the tensor network could be simplified. For example for computing the amplitude of a bitstring, one would project each output index into a particular qubit state of the bitstring, resulting in a new tensor network with no output indices. For computing a local observables, one needs to contract a new tensor network where the observable is sandwiched between the two tensor networks formed by $|\psi\rangle$ and its conjugate $\langle\psi|$. In either case, the solution of the problem could be obtained by contracting a large tensor network with no or very few output indices.

Once the problem has been equivalently transformed into a tensor network contraction (TNC) problem, the key for the performance is to identify a near optimal tensor network contraction order (TNCO). Here we note that there is an important difference between the TNC algorithm and the tensor network states based algorithms for simulating quantum circuits, although in both cases the target problem will eventually be converted into the problem of contracting a tensor network. Taking the task of computing one amplitude as an example, for a quantum circuit which only contains single-qubit and two-qubit gate operations, the total number of tensors of the tensor network formed in the TNC algorithm will be approximately equal to the number of two-qubit gates, while for MPS or PEPS algorithms the number of tensors in the final tensor network is strictly equal to the total number of qubits (but in the latter case each tensor could be very large). In fact, the PEPS algorithm can also be viewed as a specialized instance of the TNC algorithm by choosing a particular TNCO: one first contracts all the tensors in the time direction, which results in a two-dimensional tensor network in the spatial directions (assuming that the quantum processor has a two-dimensional geometry), and then contracts the resulting two-dimensional tensor network. For PEPS based simulator since the resulting two-dimensional tensor network has a much smaller number of tensors and also has a very regular structure, one could easily identify an optimal TNCO. Therefore when the tensor network is contracted exactly, the TNC based simulator has the maximum flexibility to choose an optimal TNCO which can result in lower computational complexity compared to the PEPS based simulator in principle.

The TNC algorithm is powered by the highly efficient heuristic methods to search for near optimal TNCOs developed in recent years [44, 501, 502]. When applied to RQCs on Sycamore(-like) quantum processors, the TNC algorithm gives much lower computational complexity than PEPS. The central idea of those heuristic methods is to use existing graph partitioning methods to first divide the large tensor network with hundreds of tensors (corresponding to a graph with hundreds of nodes) into many smaller tensor networks such that one could easily compute the TNCO for each smaller tensor network and then assembly then together into a large TNCO

for the original tensor network [501]. For very large tensor networks, slicing is also an indispensable technique which "cuts" a number of edges in the original tensor network such that the original tensor network is transformed into the sum of many smaller tensor networks with fewer edges. The slicing technique could significantly reduce the memory usage since only simpler tensor networks are being contracted, and if used properly, it could only result in moderate computational overhead compared to directly contracting the original tensor network [502]. By understanding the mechanism for the computational overhead induced by slicing, a lifetime based heuristic method is proposed to systematically find a near optimal slicing scheme [503].

For the task of computing one amplitude or a correlated batch of amplitudes for the hardest Sycamore RQC (with a circuit depth 20), various works have all given a computational complexity of the order $O(10^{18})$. While such a complexity scale can be accomplished on a exascale supercomputer within seconds in principle, the fastest record till now takes around 150 seconds to compute a batch of correlated amplitudes [503]. For computing a number of uncorrelated amplitudes (or generating uncorrelated samples), a recent work develops a multiple-amplitude TNC algorithm which caches intermediate computations to reduce the overall complexity, and based on this method millions of uncorrelated amplitudes for the Sycamore RQCs with depths up to 16 are successfully computed [504]. A sparse-state method is also proposed to compute a large number of amplitudes with a limited fidelity, which is used to generate one million uncorrelated samples for the hardest Sycamore RQC using 512 GPUs for 15 hours, with a higher fidelity than in the quantum supremacy experiment [45]. For RQC on a square lattice the largest reported simulation scale using the TNC algorithm reaches 9×9 qubits with a circuit depth 40 [505]. Typical large-scale classical simulations of RQCs have also been summarized in TABLE. III.

B. Simulating noisy quantum circuits

In certain situations it is also important to simulate noisy quantum circuits, for example, to understand the behaviors of quantum algorithms under certain level of noises, to test quantum noise models for quantum computers or to test quantum error mitigations schemes.

In a noisy quantum circuit, the underlying quantum state should be represented as a mixed state (density operator), and the quantum gate operations should be generalized to quantum channels which are super operators acting on the density operator. To simulate a noisy quantum circuit classically, two very different but in principle equivalent approaches can be used. In the first approach, one simply inserts random Pauli errors into the original noiseless quantum circuit with certain probabilities [1]. By averaging the results over a large number of error realizations, one could reproduce the result of a noisy quantum circuit. Since for each noise realization one only needs to deal with a quantum circuit which only contains unitary quantum gate operations, the classical simulators for

TABLE III. Typical large-scale classical simulations of random quantum circuits, where the results for two different circuit geometries, rectangular (Rect.) lattice and Sycamore are listed. For rectangular lattice the circuit size is shown by $height \times width$. The column #Amplitudes show that number of amplitudes been computed. QTIA is a shorthand for a quantum teleportation inspired algorithm used in Ref. [506].

Hardware	RQC (Two-qubit gate)	Circuit Size (depth)	Time	Fidelity	#Amplitudes	Algorithm
8, 192 nodes of Cori II [453]	Rect. (CZ)	5×9 (1+25+1)	10 minutes	100%	all	SA
4, 096 nodes of Blue Gene/Q [454]	Rect. (CZ)	7×7 (1+27+1)	2 days	100%	1	SA
32, 768 nodes of Sunway TaihuLight [507]	Rect. (CZ)	7×7 (1+39+1)	4.2 hours	100%	all	TNC
4, 600 Summit [499]	Rect. (CZ)	7×7 (1+40+1)	2.4 hours	0.5%	1.01×10^6	PEPS
4, 096 nodes of Tianhe-2A [41]	Rect. (CZ)	7×7 (1+40+1)	31 minutes	100%	1	PEPS
131, 072 nodes of Alibaba [505]	Rect. (CZ)	9×9 (40)	13 hours	100%	1	TNC
16, 384 nodes of Sunway TaihuLight [506]	Rect. (CZ)	8×125 (1+40+1)	131.6 minutes	100%	1	QTIA
60 GPUs [44]	Sycamore (FSIM)	53 (20)	5 days	100%	1	TNC
512 GPUs [45]	Sycamore (FSIM)	53 (20)	15 hours	0.37%	10^6	TNC
107, 520 nodes of New Sunway [43]	Sycamore (FSIM)	53 (20)	304 seconds	100%	1	TNC
107, 520 nodes of New Sunway [43]	Rect. (CZ)	10×10 (1+40+1)	60 hours	100%	1	PEPS

noiseless quantum circuits can be directly used.

In the second approach, one directly simulates the noisy quantum circuit by representing the quantum state as a density operator, then one applies the quantum channels onto the density operator and performs quantum measurements based on the density operator in the end. Compared to the first approach, the second approach is free of the errors resulting from a finite number of noise realizations. However, the memory cost of the second approach could be significantly larger since the size of a density operator is the square of a pure state in general. Although a noisy quantum circuit seems very different from a noiseless one mathematically, one can equivalently map it into a problem which is very similar to the noiseless case, after which one could directly make use of all the classical algorithms used for simulating noiseless quantum circuits. This mapping is demonstrated as follows, which is described mostly based on the state vector representation of the quantum state but is completely general for all the classical simulators we have mentioned.

First we map the density operator into an enlarged "pure state". Mathematically, a density operator $\hat{\rho}$ for an n-qubit quantum system can be denoted as

$$\hat{\rho} = \sum_{\sigma_{n:1}, \sigma'_{n:1}} \rho^{\sigma'_1, \ldots, \sigma'_n}_{\sigma_1, \ldots, \sigma_n} |\sigma_1, \ldots, \sigma_n\rangle\langle\sigma'_1, \ldots, \sigma'_n|, \quad (113)$$

where $\rho^{\sigma'_1, \ldots, \sigma'_n}_{\sigma_1, \ldots, \sigma_n}$ is the rank-$2n$ coefficient tensor (density matrix). One can equivalently view the density operator $\hat{\rho}$ as a pure state with $2n$ qubits as

$$|\hat{\rho}\rangle = \sum_{\sigma_{n:1}, \sigma'_{n:1}} \tilde{\rho}_{\sigma_1, \ldots, \sigma_n, \sigma'_1, \ldots, \sigma'_n} |\sigma_1, \ldots, \sigma_n, \sigma'_1, \ldots, \sigma'_n\rangle,$$

$$(114)$$

where $\tilde{\rho}_{\sigma_1, \ldots, \sigma_n, \sigma'_1, \ldots, \sigma'_n}$ is similar to $c_{\sigma_1, \ldots, \sigma_n}$ in the noiseless case. Again in the mapping of the coefficient tensor from ρ to $\tilde{\rho}$, nothing needs to be done actually, it is just a different way to view the same tensor. We will also refer to $\tilde{\rho}$ as the *squashed state vector*.

The quantum channel is defined as a super operator acting on the density operator $\hat{\rho}$. In the squashed state vector representation, the quantum channel will be a normal quantum

operator acting on $|\hat{\rho}\rangle$ instead. Taking the case of a single-qubit quantum channel for example, which is denoted as $\mathcal{K}$ and acts on the j-th qubit, we assume that it is written in the standard Sudarshan-Kraus-Choi form as [508–510]

$$\mathcal{K}(\rho) = \sum_{s, \sigma_j, \sigma'_j} K^s_{\tau'_j, \sigma'_j} \rho^{\sigma'_1, \ldots, \sigma'_n}_{\sigma_1, \ldots, \sigma_n} (K^s_{\tau_j, \sigma_j})^*. \quad (115)$$

In the squashed state vector representation, the quantum channel $\mathcal{K}$ will be mapped into a normal operator acting on $\tilde{\rho}$, denoted as $\tilde{\mathcal{K}}$, which can be writen as

$$\tilde{\mathcal{K}}(\tilde{\rho}) = \sum_{s, \sigma_j, \sigma'_j} K^s_{\tau'_j, \sigma'_j} (K^s_{\tau_j, \sigma_j})^* \tilde{\rho}_{\sigma_1, \ldots, \sigma_n, \sigma'_1, \ldots, \sigma'_n}$$

$$= M^{\tau_j, \tau'_j}_{\sigma_j, \sigma'_j} \tilde{\rho}_{\sigma_1, \ldots, \sigma_n, \sigma'_1, \ldots, \sigma'_n}, \quad (116)$$

where in the second line we have defined

$$M^{\tau_j, \tau'_j}_{\sigma_j, \sigma'_j} = \sum_s K^s_{\tau'_j, \sigma'_j} (K^s_{\tau_j, \sigma_j})^*. \quad (117)$$

We can see from Eq.(116) that the effect of a single-qubit quantum channel acting on $\hat{\rho}$ is exactly equivalent to a two-qubit "gate operation" acting on $|\hat{\rho}\rangle$. Similarly any the l-qubit quantum channel acting on $\hat{\rho}$ can be mapped to a $2l$-qubit gate operation on $|\hat{\rho}\rangle$. Thus all the quantum channels can be applied onto the squashed state vector using the same algorithm which applies unitary quantum gate operations onto a state vector.

For a density operator, the amplitude of a given bitstring $\vec{b}$ is not well defined, but one can compute the probability of $\vec{b}$ instead, which is

$$\langle\vec{b}|\hat{\rho}|\vec{b}\rangle = \rho^{\sigma'_1 = b_1, \ldots, \sigma'_n = b_n}_{\sigma_1 = b_1, \ldots, \sigma_n = b_n}$$

$$= \tilde{\rho}_{\sigma_1 = b_1, \ldots, \sigma_n = b_n, \sigma'_1 = b_1, \ldots, \sigma'_n = b_n}. \quad (118)$$

Therefore the probability $\langle\vec{b}|\hat{\rho}|\vec{b}\rangle$ can be computed by computing the "amplitude" of the squashed state vector $|\hat{\rho}\rangle$ on an enlarged bitstring $\{b_1, \ldots, b_n, b_1, \ldots, b_n\}$. A local observable

$\hat{O}^j$ acting on the j-th qubit is by definition

$$\text{tr}(\hat{O}^j\hat{\rho}) = \sum_{\sigma_{j-1:1},\sigma_{n:j+1},\sigma_j,\sigma'_j} O^{\sigma'_j}_{\sigma_j}\rho^{\sigma_1,\ldots,\sigma'_j,\ldots,\sigma_n}_{\sigma_1,\ldots,\sigma_j,\ldots,\sigma_n}, \quad (119)$$

which can be computed by applying a two-qubit gate operation on $|\hat{\rho}\rangle$, and then view $|\hat{\rho}\rangle$ as a density operator again and take the trace of it (instead of taking the dot product between two state vectors as in the noiseless case).

Therefore we can see that the second approach to simulate a noisy quantum circuit could be equivalently mapped into simulating a noiseless quantum circuit, with some minor differences that the effective number of qubits is doubled and that one may need to slightly adapt the algorithms used to simulate quantum measurements. There is another subtle difference that in case of noisy quantum circuits the "gate operations" on the squashed state vector are no longer unitary, which would affect the classical simulators which explicitly make use of this property. For the classical simulators that we have introduced, only the MPS based simulator has explicitly used the unitary property to preserve the right canonical form of MPS. In the noisy case, the gate operation algorithms based on MPS can still be used for $|\hat{\rho}\rangle$ in principle if the truncation error is set to be very small. However it could be less accurate and less stable compared to the unitary case since the singular values being truncated no longer correspond to the correct Schmidt numbers. The non-unitarity of noisy quantum circuits could also affect the stability of the PEPS based simulator if the gate operation and the tensor network contraction are not performed exactly. The density operator based approach for noisy quantum circuits has been implemented in Qiskit [360], Quest [457], Qulacs [458], TensorCircuit [511], which mostly uses the Schrödinger algorithm.

C. Computing gradients for parametric quantum circuits

In the VQE algorithm, the quantum circuit often contains a number of tunable parameters, and the solution is approached by iteratively updating those parameters. To accelerate convergence, a gradient-based optimization algorithm is usually preferred, especially when there is a large number of parameters. To simulate VQE on a classical computer, it would also be helpful to efficiently compute the gradients on classical computers. Compared to quantum computers, computing gradients on classical computers is much more flexible. In the following we will show several different approaches to compute the gradients of parametric quantum circuits, which could be useful in different situations.

We assume that the quantum circuit which is parameterized by a list of parameters $\vec{\theta} = \{\theta_1, \theta_2, \ldots, \theta_m\}$, denoted as $\hat{\mathcal{C}}(\vec{\theta})$. We also assume that $\hat{\mathcal{C}}(\vec{\theta})$ can be written as

$$\hat{\mathcal{C}}(\vec{\theta}) = \hat{Q}(\theta_m)\cdots\hat{Q}(\theta_2)\hat{Q}(\theta_1) \quad (120)$$

where each parametric gate $\hat{Q}(\theta_j)$ contains a single parameter θ_j (the case that there exists several parameters in one gate could be analyzed similarly). A quite generic task to do using

parametric quantum circuits is to minimize some loss function in the form

$$\mathcal{L}(\vec{\theta}) = \langle 0^{\otimes n}|\hat{\mathcal{C}}^\dagger(\vec{\theta})\hat{H}\hat{\mathcal{C}}(\vec{\theta})|0^{\otimes n}\rangle, \quad (121)$$

where $\hat{H}$ is a Hermitian quantum operator (which does not have to be local).

The most straightforward approach to compute the gradient of the loss function in Eq.(121) is the finite difference method, in which the gradient with respect to one parameter θ_j is evaluated as

$$\frac{\partial\mathcal{L}(\vec{\theta})}{\partial\theta_j} \approx \frac{\mathcal{L}(\ldots,\theta_j+\delta,\ldots)-\mathcal{L}(\vec{\theta})}{\delta}, \quad (122)$$

with δ a small positive number. Denoting the complexity of evaluating Eq.(121) as $\mathcal{S}$, we can see that the complexity of evaluating the gradients with respect to all the parameters using Eq.(122) is $O(m\mathcal{S})$ with m the total number of parameters. Eq.(122) evaluates the first-order derivative against δ, which can be slightly modified to be second-order:

$$\frac{\partial\mathcal{L}(\vec{\theta})}{\partial\theta_j} \approx \frac{\mathcal{L}(\ldots,\theta_j+\delta,\ldots)-\mathcal{L}(\ldots,\theta_j-\delta,\ldots)}{2\delta}. \quad (123)$$

The complexity of evaluating Eq.(123) is $O(2m\mathcal{S})$. One could also use a higher-order finite-difference method for computing the gradients, which would in general be more accurate by also incur higher computational cost.

For gradient-based algorithms one often wants to obtain gradients which are *numerically exact*. For those parametric quantum circuits whose parameters are all encoded in the single-qubit rotational gates R_x, R_y, R_z defined as

$$R_x(\theta) = \begin{bmatrix} \cos(\frac{\theta}{2}) & -\text{i}\sin(\frac{\theta}{2}) \\ -\text{i}\sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{bmatrix}; \quad (124)$$

$$R_y(\theta) = \begin{bmatrix} \cos(\frac{\theta}{2}) & -\sin(\frac{\theta}{2}) \\ \sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{bmatrix}; \quad (125)$$

$$R_z(\theta) = \begin{bmatrix} e^{-\text{i}\frac{\theta}{2}} & 0 \\ 0 & e^{\text{i}\frac{\theta}{2}} \end{bmatrix}, \quad (126)$$

the exact gradients can be computed using the parameter shift rule as [83]

$$\frac{\partial\mathcal{L}(\vec{\theta})}{\partial\theta_j} = \mathcal{L}(\ldots,\theta_j+\frac{\pi}{2},\ldots)-\mathcal{L}(\ldots,\theta_j-\frac{\pi}{2},\ldots). \quad (127)$$

We can see that the complexity of evaluating Eq.(127) is the same to Eq.(123). For more general parametric quantum gate operations Eq.(127) may not hold, nevertheless more sophisticated methods have been proposed evaluate the gradients exactly [84]. Importantly, Eqs.(122,123,127) can all be evaluated on quantum computers, since they only require the ability for *forward evaluation* of the loss function in Eq.(121).

Similar to computing the expectation values, for classical simulators there often exists shortcuts to evaluate the gradients. In fact the key for the success of classical deep learning is the *automatic differentiation* algorithm which allows to

compute the gradients very efficiently as evaluating the loss function itself [512]. Since for classical simulators the loss function only runs on a classical computer, in principle one can also benefit from the automatic differentiation algorithm by generalizing it to complex numbers [513]. However, if one directly applies the automatic differentiation algorithm in the classical simulator, then the intermediate outputs after applying each $\hat{Q}(\theta_j)$ onto the quantum state have to be cached so as to enable efficient back propagation, namely one has to store at least m copies of the quantum state. This would be impossible (or highly inefficient) even for moderate values of m. Nevertheless, by exploring the reversibility of the quantum circuit, one could use a memory efficient back propagation algorithm for parametric quantum circuits where only two copies of the quantum state have to be used at least. The algorithm is shown in the following based on the Schrödinger simulator.

Taking the partial derivative of Eq.(121) with respect to one of the parameters, θ_j, we get

$$\frac{\partial \mathcal{L}(\vec{\theta})}{\partial \theta_j} = \langle \psi(\vec{\theta})|\hat{H}\hat{\mathcal{C}}_{m:j+1}\frac{d\hat{Q}(\theta_j)}{d\theta_j}\hat{\mathcal{C}}_{j-1:1}|0^{\otimes n}\rangle + \text{H.c.}, \tag{128}$$

where we have used $|\psi(\vec{\theta})\rangle = \hat{\mathcal{C}}(\vec{\theta})|0^{\otimes n}\rangle$ and $\hat{\mathcal{C}}_{b:a} = \hat{Q}(\theta_b)\hat{Q}(\theta_{b-1})\cdots\hat{Q}(\theta_a)$. Now we define

$$|\Phi_j\rangle = \hat{\mathcal{C}}_{j:1}|0^{\otimes n}\rangle; \tag{129}$$

$$|\Psi_j\rangle = \hat{\mathcal{C}}^{\dagger}_{m:j+1}\hat{H}|\psi(\vec{\theta})\rangle, \tag{130}$$

then Eq.(128) can be rewritten as

$$\frac{\partial \mathcal{L}(\vec{\theta})}{\partial \theta_j} = \langle \Psi_j|\frac{d\hat{Q}(\theta_j)}{d\theta_j}|\Phi_{j-1}\rangle + \text{H.c.}. \tag{131}$$

Now the memory efficient back propagation algorithm to compute all the partial derivatives runs as follows. First we run a forward evaluation of Eq.(121), during which the two intermediate states $|\Phi_m\rangle = |\psi(\vec{\theta})\rangle$ and $|\Psi_m\rangle = \hat{H}|\psi(\vec{\theta})\rangle$ are both saved. Then we do a backward evolution of the two states $|\Phi_m\rangle$ and $|\Psi_m\rangle$ by applying each of the gate operations in $\hat{\mathcal{C}}^{\dagger}(\vec{\theta})$ onto them. The details of the backward evolution algorithm is shown in Algorithm. 2.

Algorithm 2 Memory-efficient back propagation algorithm for computing the gradient of a parametric quantum circuit, where $|\Phi_m\rangle$ and $|\Psi_m\rangle$ are produced during the forward evaluation.

1: $grads = zeros(m)$
2: **for** $j = m - 1 : -1 : 0$ **do**
3:    $|\Phi_j\rangle = \hat{Q}(\theta_{j+1})^{\dagger}|\Phi_{j+1}\rangle$
4:    $grads[j] = 2\text{Re}(\langle \Psi_{j+1}|\frac{d\hat{Q}(\theta_j)}{d\theta_j}|\Phi_j\rangle)$
5:    $|\Psi_j\rangle = \hat{Q}(\theta_{j+1})^{\dagger}|\Psi_{j+1}\rangle$
6: **end for**
7: Return $grads$

We can see that in Algorithm. 2 no additional memory needs to be allocated in principle, since the gate operations can be done in an inplace fashion, and the "expectation value" on two different states as in Eq.(131) can also be implemented without allocating new memory. Therefore only two copies of the quantum states are required. We can also see that the time complexity of Algorithm. 2 is approximately two times the complexity of the forward evaluation, namely $O(2\mathcal{S})$, since one needs to (backward) evolve the two states $|\Phi_m\rangle$ and $|\Psi_m\rangle$.

To this end we stress that the $O(2\mathcal{S})$ scaling for Algorithm. 2 is only an ideal estimation, in which we have implicitly assumed that the complexity of the state $|\Psi_j\rangle$ is equal to $|\Phi_m\rangle$ ($|\Psi_j\rangle$ is not a proper quantum state since $\hat{H}$ may not be unitary in general), which is true for the Schrödinger simulator but may not be true for other classical simulators. Taking the MPS based simulator for example, the state $|\Psi_m\rangle$ requires to apply $\hat{H}$ onto $|\psi(\vec{\theta})\rangle$, however, since $\hat{H}$ could be a complex summation of a large number of Pauli strings, the $|\Psi_m\rangle$ would have a much larger bond dimension than $|\psi(\vec{\theta})\rangle$ if this operation is performed accurately and the gate operations on $|\Psi_m\rangle$ has to be simulated with higher complexity. The integration of the MPS based simulator into the automatic differentiation framework has been proposed and implemented in Ref. [485], referred to as the *differentiable MPS*. It is also possible to generalize Algorithm. 2 to compute the gradients of noisy parametric quantum circuits (but the details of the algorithm have to be significantly modified), as long as the quantum channels are reversible.

VII. CONCLUSION AND OUTLOOK

This review comprehensively summarizes near-term quantum computing techniques, including variational quantum algorithms, quantum error mitigation, quantum circuit compilation, benchmarking protocols, and classical simulation, from basic concepts to current progress. To develop a practical near-term quantum computing system, a high level of interaction and collaboration between quantum hardware and these near-term quantum computing techniques is required. Over the last years, crucial theoretical and experimental advancements have been made. To realize the leap from the quantum computational advantage for the sampling task without too much practical use to the application-oriented quantum computational advantage, however, greater efforts are required.

First, more profound research are required to fully grasp the potential of NISQ devices and what is the right goal for the NISQ era. Fortunately, we have seen a shift in the focus of research from an initial blind pursuit to serious consideration of these issues [514]. We need to find the "killer apps" for NISQ, evaluate the resources they consume, and determine whether they can provide us with speed, accuracy or other advantages (especially whether it will be de quantized [515–517]), to fully unleash the power of the near-term quantum computing systems.

Second, to continually enhance the capabilities quantum computing hardware, and make it grow massively, a large number of cutting-edge experimental techniques should be conquered. For example, advanced processes, materials and

designs are required for the fabrication of quantum computing processors; Dilution refrigerators with larger space and higher cooling power; Cryogenic electronic control techniques; and so on. We believe that the influx of more companies will be of great help to the benign development of the quantum computing industry. Besides, the deep integration of classical and quantum computing is also a preferable way to further enhance the computational power, such as the circuit-cutting method [518, 519].

The future quantum computers may shape our daily lives and science and technology in ways that we cannot currently foresee. Near-term quantum computing techniques are crucial to the entire development stage of quantum computing, serving as an enabler from proof-of-principle demonstrations to engineering scaling. In addition, the available NISQ era cloud-based platforms provide great convenience for exploring and developing practical quantum computing algorithms and methods [28, 520]. We expect more young, bright, energetic people entering this field to accelerate the pace of realizing the full promise of quantum computing.

[1] M. A. Nielsen and I. Chuang, "Quantum computation and quantum information," (2002).

[2] T. D. Ladd, F. Jelezko, R. Laflamme, Y. Nakamura, C. Monroe, and J. L. O'Brien, Nature **464**, 45 (2010).

[3] P. W. Shor, SIAM review **41**, 303 (1999).

[4] L. K. Grover, in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing* (1996) pp. 212–219.

[5] H.-L. Huang, D. Wu, D. Fan, and X. Zhu, SCIENCE CHINA Information Sciences **63**, 180501 (2020).

[6] J. Preskill, Quantum **2**, 79 (2018).

[7] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. Brandao, D. A. Buell, *et al.*, Nature **574**, 505 (2019).

[8] Y. Wu, W.-S. Bao, S. Cao, F. Chen, M.-C. Chen, X. Chen, T.-H. Chung, H. Deng, Y. Du, D. Fan, *et al.*, Physical Review Letters **127**, 180501 (2021).

[9] Q. Zhu, S. Cao, F. Chen, M.-C. Chen, X. Chen, T.-H. Chung, H. Deng, Y. Du, D. Fan, M. Gong, *et al.*, Science bulletin **67**, 240 (2022).

[10] H.-S. Zhong, H. Wang, Y.-H. Deng, M.-C. Chen, L.-C. Peng, Y.-H. Luo, J. Qin, D. Wu, X. Ding, Y. Hu, *et al.*, Science **370**, 1460 (2020).

[11] H.-S. Zhong, Y.-H. Deng, J. Qin, H. Wang, M.-C. Chen, L.-C. Peng, Y.-H. Luo, D. Wu, S.-Q. Gong, H. Su, *et al.*, Physical Review Letters **127**, 180502 (2021).

[12] L. S. Madsen, F. Laudenbach, M. F. Askarani, F. Rortais, T. Vincent, J. F. Bulmer, F. M. Miatto, L. Neuhaus, L. G. Helt, M. J. Collins, *et al.*, Nature **606**, 75 (2022).

[13] P. Krantz, M. Kjaergaard, F. Yan, T. P. Orlando, S. Gustavsson, and W. D. Oliver, Applied Physics Reviews **6**, 021318 (2019).

[14] M. Kjaergaard, M. E. Schwartz, J. Braumüller, P. Krantz, J. I.-J. Wang, S. Gustavsson, and W. D. Oliver, Annual Review of Condensed Matter Physics **11**, 369 (2020).

[15] H. Häffner, C. F. Roos, and R. Blatt, Physics reports **469**, 155 (2008).

[16] C. D. Bruzewicz, J. Chiaverini, R. McConnell, and J. M. Sage, Applied Physics Reviews **6**, 021314 (2019).

[17] J. Wang, F. Sciarrino, A. Laing, and M. G. Thompson, Nature Photonics **14**, 273 (2020).

[18] S. Slussarenko and G. J. Pryde, Applied Physics Reviews **6**, 041303 (2019).

[19] F. Flamini, N. Spagnolo, and F. Sciarrino, Reports on Progress in Physics **82**, 016001 (2018).

[20] P. W. Shor, in *Proceedings of 37th conference on foundations of computer science* (IEEE, 1996) pp. 56–65.

[21] Y. Zhao, Y. Ye, H.-L. Huang, Y. Zhang, D. Wu, H. Guan, Q. Zhu, Z. Wei, T. He, S. Cao, *et al.*, Physical Review Letters **129**, 030501 (2022).

[22] S. Krinner, N. Lacroix, A. Remm, A. Di Paolo, E. Genois, C. Leroux, C. Hellings, S. Lazar, F. Swiadek, J. Herrmann, *et al.*, Nature **605**, 669 (2022).

[23] H.-L. Huang, M. Narożniak, F. Liang, Y. Zhao, A. D. Castellano, M. Gong, Y. Wu, S. Wang, J. Lin, Y. Xu, *et al.*, Physical Review Letters **126**, 090502 (2021).

[24] C. Liu, H.-L. Huang, C. Chen, B.-Y. Wang, X.-L. Wang, T. Yang, L. Li, N.-L. Liu, J. P. Dowling, T. Byrnes, *et al.*, Optica **6**, 264 (2019).

[25] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Nature **549**, 195 (2017).

[26] A. Broadbent, J. Fitzsimons, and E. Kashefi, in *2009 50th Annual IEEE Symposium on Foundations of Computer Science* (IEEE, 2009) pp. 517–526.

[27] H.-L. Huang, Q. Zhao, X. Ma, C. Liu, Z.-E. Su, X.-L. Wang, L. Li, N.-L. Liu, B. C. Sanders, C.-Y. Lu, *et al.*, Physical Review Letters **119**, 050503 (2017).

[28] H.-L. Huang, Y.-W. Zhao, T. Li, F.-G. Li, Y.-T. Du, X.-Q. Fu, S. Zhang, X. Wang, and W.-S. Bao, Frontiers of Physics **12**, 120305 (2017).

[29] H.-L. Huang, W.-S. Bao, T. Li, F.-G. Li, X.-Q. Fu, S. Zhang, H.-L. Zhang, and X. Wang, Quantum Information Processing

16, 199 (2017).

[30] S. Barz, E. Kashefi, A. Broadbent, J. F. Fitzsimons, A. Zeilinger, and P. Walther, science **335**, 303 (2012).

[31] Y. Cao, J. Romero, J. P. Olson, M. Degroote, P. D. Johnson, M. Kieferová, I. D. Kivlichan, T. Menke, B. Peropadre, N. P. Sawaya, *et al.*, Chemical reviews **119**, 10856 (2019).

[32] P. Coleman, *Introduction to many-body physics* (Cambridge University Press, 2015).

[33] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio, *et al.*, Nature Reviews Physics **3**, 625 (2021).

[34] K. Bharti, A. Cervera-Lierta, T. H. Kyaw, T. Haug, S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke, *et al.*, Reviews of Modern Physics **94**, 015004 (2022).

[35] Z. Cai, R. Babbush, S. C. Benjamin, S. Endo, W. J. Huggins, Y. Li, J. R. McClean, and T. E. O'Brien, arXiv:2210.00921 (2022).

[36] A. Botea, A. Kishimoto, and R. Marinescu, in *Eleventh annual symposium on combinatorial search* (2018).

[37] D. Venturelli, M. Do, B. O'Gorman, J. Frank, E. Rieffel, K. E. Booth, T. Nguyen, P. Narayan, and S. Nanda, SPARK 2019 , 95 (2019).

[38] J. Kusyk, S. M. Saeed, and M. U. Uyar, IEEE Transactions on Quantum Engineering **2**, 2501616 (2021).

[39] J. Eisert, D. Hangleiter, N. Walk, I. Roth, D. Markham, R. Parekh, U. Chabaud, and E. Kashefi, Nature Reviews Physics **2**, 382 (2020).

[40] E. Knill, D. Leibfried, R. Reichle, J. Britton, R. B. Blakestad, J. D. Jost, C. Langer, R. Ozeri, S. Seidelin, and D. J. Wineland, Physical Review A **77**, 012307 (2008).

[41] C. Guo, Y. Liu, M. Xiong, S. Xue, X. Fu, A. Huang, X. Qiang, P. Xu, J. Liu, S. Zheng, *et al.*, Physical Review Letters **123**, 190501 (2019).

[42] C. Guo, Y. Zhao, and H.-L. Huang, Physical Review Letters **126**, 070502 (2021).

[43] Y. Liu, X. Liu, F. Li, H. Fu, Y. Yang, J. Song, P. Zhao, Z. Wang, D. Peng, H. Chen, *et al.*, in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (2021) pp. 1–12.

[44] F. Pan and P. Zhang, Physical Review Letters **128**, 030501 (2022).

[45] F. Pan, K. Chen, and P. Zhang, Physical Review Letters **129**, 090502 (2022).

[46] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Nature **549**, 242 (2017).

[47] G. A. Quantum, Collaborators*†, F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, S. Boixo, M. Broughton, B. B. Buckley, *et al.*, Science **369**, 1084 (2020).

[48] I. Cong, S. Choi, and M. D. Lukin, Nature Physics **15**, 1273 (2019).

[49] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, Nature **567**, 209 (2019).

[50] J. Liu, K. H. Lim, K. L. Wood, W. Huang, C. Guo, and H.-L. Huang, Science China Physics, Mechanics & Astronomy **64**, 290311 (2021).

[51] H.-L. Huang, X.-L. Wang, P. P. Rohde, Y.-H. Luo, Y.-W. Zhao, C. Liu, L. Li, N.-L. Liu, C.-Y. Lu, and J.-W. Pan, Optica **5**, 193 (2018).

[52] S. Lloyd and C. Weedbrook, Physical Review Letters **121**, 040502 (2018).

[53] H.-L. Huang, Y. Du, M. Gong, Y. Zhao, Y. Wu, C. Wang, S. Li, F. Liang, J. Lin, Y. Xu, *et al.*, Physical Review Applied **16**, 024051 (2021).

[54] M. S. Rudolph, N. B. Toussaint, A. Katabarwa, S. Johri, B. Peropadre, and A. Perdomo-Ortiz, Physical Review X **12**, 031010 (2022).

[55] S. Ebadi, A. Keesling, M. Cain, T. T. Wang, H. Levine, D. Bluvstein, G. Semeghini, A. Omran, J.-G. Liu, R. Samajdar, *et al.*, Science , eabo6587 (2022).

[56] M. P. Harrigan, K. J. Sung, M. Neeley, K. J. Satzinger, F. Arute, K. Arya, J. Atalaya, J. C. Bardin, R. Barends, S. Boixo, *et al.*, Nature Physics **17**, 332 (2021).

[57] L. Zhou, S.-T. Wang, S. Choi, H. Pichler, and M. D. Lukin, Physical Review X **10**, 021067 (2020).

[58] M. Gong, H.-L. Huang, S. Wang, C. Guo, S. Li, Y. Wu, Q. Zhu, Y. Zhao, S. Guo, H. Qian, *et al.*, arXiv:2201.05957 (2022).

[59] C. Ding, X.-Y. Xu, Y.-F. Niu, S. Zhang, W.-S. Bao, and H.-L. Huang, arXiv:2208.02104 (2022).

[60] Y. Liu, D. Wang, S. Xue, A. Huang, X. Fu, X. Qiang, P. Xu, H.-L. Huang, M. Deng, C. Guo, X. Yang, and J. Wu, Physical Review A **101**, 052316 (2020).

[61] S. Xue, Y. Liu, Y. Wang, P. Zhu, C. Guo, and J. Wu, Physical Review A **105**, 032427 (2022).

[62] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, New Journal of Physics **18**, 023023 (2016).

[63] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles, Nature Communications **12**, 1791 (2021).

[64] Z. Liu, L.-W. Yu, L.-M. Duan, and D.-L. Deng, arXiv:2108.08312 (2021).

[65] E. Farhi, J. Goldstone, and S. Gutmann, arXiv:1411.4028 (2014).

[66] A. G. Taube and R. J. Bartlett, International journal of quantum chemistry **106**, 3393 (2006).

[67] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'brien, Nature Communications **5**, 4213 (2014).

[68] J. Lee, W. J. Huggins, M. Head-Gordon, and K. B. Whaley, Journal of chemical theory and computation **15**, 311 (2018).

[69] W. Mizukami, K. Mitarai, Y. O. Nakagawa, T. Yamamoto, T. Yan, and Y.-y. Ohnishi, Physical Review Research **2**, 033421 (2020).

[70] M. Metcalf, N. P. Bauman, K. Kowalski, and W. A. De Jong, Journal of chemical theory and computation **16**, 6165 (2020).

[71] D. A. Fedorov, B. Peng, N. Govind, and Y. Alexeev, Materials Theory **6**, 2 (2022).

[72] H. R. Grimsley, S. E. Economou, E. Barnes, and N. J. Mayhall, Nature Communications **10**, 3007 (2019).

[73] H. L. Tang, V. Shkolnikov, G. S. Barron, H. R. Grimsley, N. J. Mayhall, E. Barnes, and S. E. Economou, PRX Quantum **2**, 020310 (2021).

[74] L. Zhu, H. L. Tang, G. S. Barron, F. Calderon-Vargas, N. J. Mayhall, E. Barnes, and S. E. Economou, Physical Review Research **4**, 033029 (2022).

[75] Y.-X. Yao, N. Gomes, F. Zhang, C.-Z. Wang, K.-M. Ho, T. Iadecola, and P. P. Orth, PRX Quantum **2**, 030307 (2021).

[76] Z.-J. Zhang, T. H. Kyaw, J. Kottmann, M. Degroote, and A. Aspuru-Guzik, Quantum Science and Technology **6**, 035001 (2021).

[77] D. Claudino, J. Wright, A. J. McCaskey, and T. S. Humble, Frontiers in Chemistry **8**, 1152 (2020).

[78] J. Liu, Z. Li, and J. Yang, The Journal of chemical physics **154**, 244112 (2021).

[79] S.-X. Zhang, C.-Y. Hsieh, S. Zhang, and H. Yao, Quantum Science and Technology **7**, 045023 (2022).

[80] S.-X. Zhang, C.-Y. Hsieh, S. Zhang, and H. Yao, Machine

Learning: Science and Technology **2**, 045027 (2021).

[81] L. Funcke, T. Hartung, K. Jansen, S. Kühn, M. Schneider, and P. Stornati, arXiv:2111.11489 (2021).

[82] J. Li, X. Yang, X. Peng, and C.-P. Sun, Physical Review Letters **118**, 150503 (2017).

[83] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, Physical Review A **98**, 032309 (2018).

[84] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran, Physical Review A **99**, 032331 (2019).

[85] L. Banchi and G. E. Crooks, Quantum **5**, 386 (2021).

[86] D. Wierichs, J. Izaac, C. Wang, and C. Y.-Y. Lin, Quantum **6**, 677 (2022).

[87] K. Li, S. Wei, P. Gao, F. Zhang, Z. Zhou, T. Xin, X. Wang, P. Rebentrost, and G. Long, npj Quantum Information **7**, 16 (2021).

[88] J. Stokes, J. Izaac, N. Killoran, and G. Carleo, Quantum **4**, 269 (2020).

[89] N. Yamamoto, arXiv:1909.05074 (2019).

[90] D. Wierichs, C. Gogolin, and M. Kastoryano, Physical Review Research **2**, 043246 (2020).

[91] B. Koczor and S. C. Benjamin, arXiv:1912.08660 (2019).

[92] T. Jones, arXiv:2011.02991 (2020).

[93] P. Rebentrost, M. Schuld, L. Wossnig, F. Petruccione, and S. Lloyd, New Journal of Physics **21**, 073023 (2019).

[94] B. Koczor and S. C. Benjamin, Physical Review Research **4**, 023017 (2022).

[95] A. W. Harrow and J. C. Napp, Physical Review Letters **126**, 140502 (2021).

[96] R. Sweke, F. Wilde, J. Meyer, M. Schuld, P. K. Fährmann, B. Meynard-Piganeau, and J. Eisert, Quantum **4**, 314 (2020).

[97] A. Anand, M. Degroote, and A. Aspuru-Guzik, Machine Learning: Science and Technology **2**, 045012 (2021).

[98] T. Zhao, G. Carleo, J. Stokes, and S. Veerapaneni, Machine Learning: Science and Technology **2**, 02LT01 (2020).

[99] S. Khairy, R. Shaydulin, L. Cincio, Y. Alexeev, and P. Balaprakash, arXiv:1911.04574 (2019).

[100] M. M. Wauters, E. Panizon, G. B. Mbeng, and G. E. Santoro, Physical Review Research **2**, 033446 (2020).

[101] J. Yao, M. Bukov, and L. Lin, in *Mathematical and Scientific Machine Learning* (PMLR, 2020) pp. 605–634.

[102] J. Rayleigh, Phil. Trans **161**, 16 (1870).

[103] W. Ritz, **1909**, 1 (1909).

[104] S. H. Gould, *Variational methods for eigenvalue problems: an introduction to the methods of Rayleigh, Ritz, Weinstein, and Aronszajn* (Courier Corporation, 1966).

[105] J. R. McClean, M. E. Kimchi-Schwartz, J. Carter, and W. A. de Jong, Physical Review A **95**, 042308 (2017).

[106] K. M. Nakanishi, K. Mitarai, and K. Fujii, Physical Review Research **1**, 033062 (2019).

[107] O. Higgott, D. Wang, and S. Brierley, Quantum **3**, 156 (2019).

[108] R. M. Parrish, E. G. Hohenstein, P. L. McMahon, and T. J. Martínez, Physical Review Letters **122**, 230401 (2019).

[109] T. Jones, S. Endo, S. McArdle, X. Yuan, and S. C. Benjamin, Physical Review A **99**, 062304 (2019).

[110] F. Zhang, N. Gomes, Y. Yao, P. P. Orth, and T. Iadecola, Physical Review B **104**, 075159 (2021).

[111] J. Tilly, G. Jones, H. Chen, L. Wossnig, and E. Grant, Physical Review A **102**, 062425 (2020).

[112] N. Yoshioka, Y. O. Nakagawa, K. Mitarai, and K. Fujii, Physical Review Research **2**, 043289 (2020).

[113] S. Endo, I. Kurata, and Y. O. Nakagawa, Physical Review Research **2**, 033281 (2020).

[114] J. Kreula, S. R. Clark, and D. Jaksch, Scientific reports **6**, 32940 (2016).

[115] K. Mitarai, Y. O. Nakagawa, and W. Mizukami, Physical Review Research **2**, 013129 (2020).

[116] R. M. Parrish, E. G. Hohenstein, P. L. McMahon, and T. J. Martinez, arXiv:1906.08728 (2019).

[117] T. E. O'Brien, B. Senjean, R. Sagastizabal, X. Bonet-Monroig, A. Dutkiewicz, F. Buda, L. DiCarlo, and L. Visscher, npj Quantum Information **5**, 113 (2019).

[118] S. Wei, H. Li, and G. Long, Research **2020** (2020).

[119] J. Zhang, R. Ferguson, S. Kühn, J. F. Haase, C. Wilson, K. Jansen, and C. A. Muschik, arXiv:2108.08248 (2021).

[120] A. Kandala, K. Temme, A. D. Córcoles, A. Mezzacapo, J. M. Chow, and J. M. Gambetta, Nature **567**, 491 (2019).

[121] J. I. Colless, V. V. Ramasesh, D. Dahlen, M. S. Blok, M. E. Kimchi-Schwartz, J. R. McClean, J. Carter, W. A. de Jong, and I. Siddiqi, Physical Review X **8**, 011021 (2018).

[122] A. Montanaro and S. Stanisic, arXiv:2006.01179 (2020).

[123] M.-C. Chen, M. Gong, X. Xu, X. Yuan, J.-W. Wang, C. Wang, C. Ying, J. Lin, Y. Xu, Y. Wu, *et al.*, Physical Review Letters **125**, 180501 (2020).

[124] R. N. Tazhigulov, S.-N. Sun, R. Haghshenas, H. Zhai, A. T. Tan, N. C. Rubin, R. Babbush, A. J. Minnich, and G. K. Chan, arXiv:2203.15291 (2022).

[125] W. J. Huggins, B. A. O'Gorman, N. C. Rubin, D. R. Reichman, R. Babbush, and J. Lee, Nature **603**, 416 (2022).

[126] D. Lee, J. Lee, S. Hong, H.-T. Lim, Y.-W. Cho, S.-W. Han, H. Shin, J. ur Rehman, and Y.-S. Kim, Optica **9**, 88 (2022).

[127] C. Hempel, C. Maier, J. Romero, J. McClean, T. Monz, H. Shen, P. Jurcevic, B. P. Lanyon, P. Love, R. Babbush, *et al.*, Physical Review X **8**, 031022 (2018).

[128] Y. Nam, J.-S. Chen, N. C. Pisenti, K. Wright, C. Delaney, D. Maslov, K. R. Brown, S. Allen, J. M. Amini, J. Apisdorf, *et al.*, npj Quantum Information **6**, 33 (2020).

[129] J.-N. Zhang, I. Arrazola, J. Casanova, L. Lamata, K. Kim, and E. Solano, Physical Review A **101**, 052333 (2020).

[130] S. Bravyi, J. M. Gambetta, A. Mezzacapo, and K. Temme, arXiv:1701.08213 (2017).

[131] K. Setia, R. Chen, J. E. Rice, A. Mezzacapo, M. Pistoia, and J. D. Whitfield, Journal of Chemical Theory and Computation **16**, 6091 (2020).

[132] J. Romero, R. Babbush, J. R. McClean, C. Hempel, P. J. Love, and A. Aspuru-Guzik, Quantum Science and Technology **4**, 014008 (2018).

[133] T. Takeshita, N. C. Rubin, Z. Jiang, E. Lee, R. Babbush, and J. R. McClean, Physical Review X **10**, 011004 (2020).

[134] J.-G. Liu, Y.-H. Zhang, Y. Wan, and L. Wang, Physical Review Research **1**, 023025 (2019).

[135] K. Fujii, K. Mizuta, H. Ueda, K. Mitarai, W. Mizukami, and Y. O. Nakagawa, PRX Quantum **3**, 010346 (2022).

[136] Y. Zhang, L. Cincio, C. F. Negre, P. Czarnik, P. J. Coles, P. M. Anisimov, S. M. Mniszewski, S. Tretiak, and P. A. Dub, npj Quantum Information **8**, 96 (2022).

[137] N. V. Tkachenko, J. Sud, Y. Zhang, S. Tretiak, P. M. Anisimov, A. T. Arrasmith, P. J. Coles, L. Cincio, and P. A. Dub, PRX Quantum **2**, 020337 (2021).

[138] K. Mitarai, T. Yan, and K. Fujii, Physical Review Applied **11**, 044087 (2019).

[139] Y. Fan, C. Cao, X. Xu, Z. Li, D. Lv, and M.-H. Yung, arXiv:2106.15210 (2021).

[140] S.-X. Zhang, Z.-Q. Wan, C.-K. Lee, C.-Y. Hsieh, S. Zhang, and H. Yao, Physical Review Letters **128**, 120502 (2022).

[141] S. Endo, S. C. Benjamin, and Y. Li, Physical Review X **8**, 031027 (2018).

[142] S. Endo, Z. Cai, S. C. Benjamin, and X. Yuan, Journal of the Physical Society of Japan **90**, 032001 (2021).

[143] E. Rosenberg, P. Ginsparg, and P. L. McMahon, Quantum Science and Technology **7**, 015024 (2022).

[144] D. Wang, O. Higgott, and S. Brierley, Physical Review Letters **122**, 140504 (2019).

[145] R. R. Ferguson, L. Dellantonio, A. Al Balushi, K. Jansen, W. Dür, and C. A. Muschik, Physical Review Letters **126**, 220501 (2021).

[146] S. Hadfield, Z. Wang, B. O'gorman, E. G. Rieffel, D. Venturelli, and R. Biswas, Algorithms **12**, 34 (2019).

[147] J. Håstad, Journal of the ACM (JACM) **48**, 798 (2001).

[148] C. Moussa, H. Calandra, and V. Dunjko, Quantum Science and Technology **5**, 044009 (2020).

[149] M. Streif and M. Leib, arXiv:1901.01903 (2019).

[150] G. G. Guerreschi and A. Y. Matsuura, Scientific reports **9**, 6903 (2019).

[151] G. E. Crooks, arXiv:1811.08419 (2018).

[152] A. M. Dalzell, A. W. Harrow, D. E. Koh, and R. L. La Placa, Quantum **4**, 264 (2020).

[153] S. Bravyi, A. Kliesch, R. Koenig, and E. Tang, Quantum **6**, 678 (2022).

[154] D. J. Egger, J. Mareček, and S. Woerner, Quantum **5**, 479 (2021).

[155] S. Bravyi, A. Kliesch, R. Koenig, and E. Tang, arXiv:1910.08980 (1910).

[156] Z.-C. Yang, A. Rahmani, A. Shabani, H. Neven, and C. Chamon, Physical Review X **7**, 021027 (2017).

[157] D. Wecker, M. B. Hastings, and M. Troyer, Physical Review A **94**, 022309 (2016).

[158] S. Khairy, R. Shaydulin, L. Cincio, Y. Alexeev, and P. Balaprakash, in *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34 (2020) pp. 2367–2375.

[159] M. Cain, E. Farhi, S. Gutmann, D. Ranard, and E. Tang, arXiv:2207.05089 (2022).

[160] B. Tan and J. Cong, in *2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD)* (IEEE, 2020) pp. 1–9.

[161] Y. J. Patel, S. Jerbi, T. Bäck, and V. Dunjko, arXiv:2207.06294 (2022).

[162] R. Majumdar, D. Madan, D. Bhoumik, D. Vinayagamurthy, S. Raghunathan, and S. Sur-Kolay, arXiv:2106.02812 (2021).

[163] R. Herrman, L. Treffert, J. Ostrowski, P. C. Lotshaw, T. S. Humble, and G. Siopsis, Algorithms **14**, 294 (2021).

[164] R. Majumdar, D. Bhoumik, D. Madan, D. Vinayagamurthy, S. Raghunathan, and S. Sur-Kolay, arXiv:2110.04637 (2021).

[165] Z. Zhou, Y. Du, X. Tian, and D. Tao, arXiv:2205.11762 (2022).

[166] D. Amaro, C. Modica, M. Rosenkranz, M. Fiorentini, M. Benedetti, and M. Lubasch, Quantum Science and Technology **7**, 015021 (2022).

[167] A. Bengtsson, P. Vikstål, C. Warren, M. Svensson, X. Gu, A. F. Kockum, P. Krantz, C. Krizan, D. Shiri, I.-M. Svensson, *et al.*, arXiv:1912.10495 (2019).

[168] Y. Zhu, Z. Zhang, B. Sundar, A. M. Green, C. H. Alderete, N. H. Nguyen, K. Hazzard, and N. M. Linke, Quantum Science and Technology **8**, 015007 (2022).

[169] A. Krizhevsky, I. Sutskever, G. E. Hinton, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, "Advances in neural information processing systems," (2012).

[170] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, International journal of computer vision **115**, 211 (2015).

[171] K. Simonyan and A. Zisserman, arXiv:1409.1556 (2014).

[172] K. He, X. Zhang, S. Ren, and J. Sun, in *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016) pp. 770–778.

[173] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, in *Proceedings of the 34th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 70, edited by D. Precup and Y. W. Teh (PMLR, 2017) pp. 933–941.

[174] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, in *International conference on machine learning* (PMLR, 2017) pp. 1243–1252.

[175] Y. Zhang and B. Wallace, arXiv:1510.03820 (2015).

[176] A. Kirillov, D. Schlesinger, S. Zheng, B. Savchynskyy, P. H. Torr, and C. Rother, in *Asian Conference on Computer Vision* (Springer, 2016) pp. 221–236.

[177] S. Song, H. Huang, and T. Ruan, Multimedia Tools and Applications **78**, 857 (2019).

[178] A. W. Yu, D. Dohan, M.-T. Luong, R. Zhao, K. Chen, M. Norouzi, and Q. V. Le, arXiv:1804.09541 (2018).

[179] A. Pesah, M. Cerezo, S. Wang, T. Volkoff, A. T. Sornborger, and P. J. Coles, Physical Review X **11**, 041011 (2021).

[180] J. Herrmann, S. M. Llima, A. Remm, P. Zapletal, N. A. McMahon, C. Scarato, F. Swiadek, C. K. Andersen, C. Hellings, S. Krinner, *et al.*, Nature Communications **13**, 4144 (2022).

[181] G.-L. Long, Communications in Theoretical Physics **45**, 825 (2006).

[182] S.-J. Wei and G.-L. Long, Quantum Information Processing **15**, 1189 (2016).

[183] S. Wei, Y. Chen, Z. Zhou, and G. Long, AAPPS Bulletin **32**, 2 (2022).

[184] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, Communications of the ACM **63**, 139 (2020).

[185] X. Gao, Z.-Y. Zhang, and L.-M. Duan, Science advances **4**, eaat9004 (2018).

[186] J. Romero and A. Aspuru-Guzik, Advanced Quantum Technologies **4**, 2000003 (2021).

[187] H. Situ, Z. He, Y. Wang, L. Li, and S. Zheng, Information Sciences **538**, 193 (2020).

[188] P.-L. Dallaire-Demers and N. Killoran, Physical Review A **98**, 012324 (2018).

[189] B. T. Kiani, G. De Palma, M. Marvian, Z.-W. Liu, and S. Lloyd, arXiv:2101.03037 (2021).

[190] S. Chakrabarti, H. Yiming, T. Li, S. Feizi, and X. Wu, Advances in Neural Information Processing Systems **32** (2019).

[191] K. Bartkiewicz, P. Tulewicz, J. Roik, and K. Lemr, arXiv:2112.13255 (2021).

[192] E. Y. Zhu, S. Johri, D. Bacon, M. Esencan, J. Kim, M. Muir, N. Murgai, J. Nguyen, N. Pisenti, A. Schouela, *et al.*, arXiv:2109.06315 (2021).

[193] M. Y. Niu, A. Zlokapa, M. Broughton, S. Boixo, M. Mohseni, V. Smelyanskyi, and H. Neven, Physical Review Letters **128**, 220505 (2022).

[194] P. Braccia, F. Caruso, and L. Banchi, New Journal of Physics **23**, 053024 (2021).

[195] B. Coyle, M. Henderson, J. C. J. Le, N. Kumar, M. Paini, and E. Kashefi, Quantum Science and Technology **6**, 024013 (2021).

[196] M. Schuld and N. Killoran, Physical Review Letters **122**, 040504 (2019).

[197] D. Bondarenko and P. Feldmann, Physical Review Letters **124**, 130502 (2020).

[198] C. Bravo-Prieto, Machine Learning: Science and Technology **2**, 035028 (2021).

[199] C. Cao and X. Wang, Physical Review Applied **15**, 054012 (2021).

[200] D. F. Locher, L. Cardarelli, and M. Müller, arXiv:2202.00555 (2022).

[201] M. Srikumar, C. D. Hill, and L. C. Hollenberg, Quantum Science and Technology **7**, 015020 (2021).

[202] Y. Du and D. Tao, arXiv:2106.15432 (2021).

[203] A. Pepper, N. Tischler, and G. J. Pryde, Physical Review Letters **122**, 060501 (2019).

[204] C.-J. Huang, H. Ma, Q. Yin, J.-F. Tang, D. Dong, C. Chen, G.-Y. Xiang, C.-F. Li, and G.-C. Guo, Physical Review A **102**, 032412 (2020).

[205] Y. Ding, L. Lamata, M. Sanz, X. Chen, and E. Solano, Advanced Quantum Technologies **2**, 1800065 (2019).

[206] E. R. Anschuetz, J. P. Olson, A. Aspuru-Guzik, and Y. Cao, arXiv:1808.08927 (2018).

[207] A. H. Karamlou, W. A. Simon, A. Katabarwa, T. L. Scholten, B. Peropadre, and Y. Cao, npj Quantum Information **7**, 156 (2021).

[208] V. Phan, A. Pönni, M. Raasakka, and I. Tittonen, arXiv:2208.07085 (2022).

[209] L. Qiu, M. Alam, A. Ash-Saki, and S. Ghosh, in *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design* (2020) pp. 229–234.

[210] Z. Wang, S. Wei, G.-L. Long, and L. Hanzo, Science China Information Sciences **65**, 200503 (2022).

[211] B. Coyle, M. Doosti, E. Kashefi, and N. Kumar, Physical Review A **105**, 042604 (2022).

[212] A. W. Harrow, A. Hassidim, and S. Lloyd, Physical Review Letters **103**, 150502 (2009).

[213] C. Bravo-Prieto, R. LaRose, M. Cerezo, Y. Subasi, L. Cincio, and P. J. Coles, arXiv:1909.05820 (2019).

[214] H.-Y. Huang, K. Bharti, and P. Rebentrost, arXiv:1909.07344 (2019).

[215] X. Xu, J. Sun, S. Endo, Y. Li, S. C. Benjamin, and X. Yuan, Science Bulletin **66**, 2181 (2021).

[216] M. Lubasch, J. Joo, P. Moinier, M. Kiffner, and D. Jaksch, Physical Review A **101**, 010301 (2020).

[217] O. Kyriienko, A. E. Paine, and V. E. Elfving, Physical Review A **103**, 052416 (2021).

[218] J. Joo and H. Moon, arXiv:2109.09216 (2021).

[219] K. Kubo, Y. O. Nakagawa, S. Endo, and S. Nagayama, Physical Review A **103**, 052425 (2021).

[220] H.-L. Liu, Y.-S. Wu, L.-C. Wan, S.-J. Pan, S.-J. Qin, F. Gao, and Q.-Y. Wen, Physical Review A **104**, 022418 (2021).

[221] D. Jaksch, P. Givi, A. J. Daley, and T. Rung, arXiv:2209.04915 (2022).

[222] C. K. Lee, P. Patil, S. Zhang, and C. Y. Hsieh, Physical Review Research **3**, 023095 (2021).

[223] M. Benedetti, M. Fiorentini, and M. Lubasch, Phys. Rev. Research **3**, 033083 (2021).

[224] A. Uvarov, A. Kardashin, and J. D. Biamonte, Physical Review A **102**, 012415 (2020).

[225] S. Liu, S.-X. Zhang, C.-Y. Hsieh, S. Zhang, and H. Yao, arXiv:2111.13719 (2021).

[226] C. Cao, C. Zhang, Z. Wu, M. Grassl, and B. Zeng, arXiv:2204.03560 (2022).

[227] H. Zhang, X. Xu, C. Zhang, M.-H. Yung, T. Huang, and Y. Liu, arXiv:2209.08306 (2022).

[228] K. Plekhanov, M. Rosenkranz, M. Fiorentini, and M. Lubasch, Quantum **6**, 670 (2022).

[229] M. Benedetti, B. Coyle, M. Fiorentini, M. Lubasch, and M. Rosenkranz, Phys. Rev. Applied **16**, 044057 (2021).

[230] V. Giovannetti, S. Lloyd, and L. Maccone, Physical Review Letters **100**, 160501 (2008).

[231] X. Sun, G. Tian, S. Yang, P. Yuan, and S. Zhang, arXiv:2108.06150 (2021).

[232] X.-M. Zhang, T. Li, and X. Yuan, arXiv:2201.11495 (2022).

[233] S. Lloyd, M. Mohseni, and P. Rebentrost, Nature Physics **10**, 631 (2014).

[234] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, Nature Communications **9**, 4812 (2018).

[235] S. Wang, E. Fontana, M. Cerezo, K. Sharma, A. Sone, L. Cincio, and P. J. Coles, Nature Communications **12**, 6961 (2021).

[236] C. O. Marrero, M. Kieferová, and N. Wiebe, PRX Quantum **2**, 040316 (2021).

[237] E. C. Martín, K. Plekhanov, and M. Lubasch, arXiv:2209.00292 (2022).

[238] S. H. Sack, R. A. Medina, A. A. Michailidis, R. Kueng, and M. Serbyn, PRX Quantum **3**, 020365 (2022).

[239] T. Haug and M. Kim, arXiv:2104.14543 (2021).

[240] L. Broers and L. Mathey, arXiv:2111.08085 (2021).

[241] E. R. Anschuetz and B. T. Kiani, arXiv:2205.05786 (2022).

[242] L. Gentini, A. Cuccoli, S. Pirandola, P. Verrucchi, and L. Banchi, Physical Review A **102**, 052414 (2020).

[243] K. Sharma, S. Khatri, M. Cerezo, and P. J. Coles, New Journal of Physics **22**, 043006 (2020).

[244] W. Saib, P. Wallden, and I. Akhalwaya, arXiv:2108.12388 (2021).

[245] J. Wright, M. Gowrishankar, D. Claudino, P. C. Lotshaw, T. Nguyen, A. J. McCaskey, and T. S. Humble, arXiv:2112.15540 (2021).

[246] M. Gowrishankar, J. Wright, D. Claudino, T. Nguyen, A. McCaskey, and T. S. Humble, in *2021 IEEE International Conference on Quantum Computing and Engineering (QCE)* (2021) pp. 155–159.

[247] K. Ito, W. Mizukami, and K. Fujii, arXiv:2106.03390 (2021).

[248] C. Ding, X.-Y. Xu, S. Zhang, H.-L. Huang, and W.-S. Bao, Physical Review A **106**, 042421 (2022).

[249] S. Wang, P. Czarnik, A. Arrasmith, M. Cerezo, L. Cincio, and P. J. Coles, arXiv:2109.01051 (2021).

[250] G. S. Barron and C. J. Wood, arXiv:2010.08520 (2020).

[251] L. Botelho, A. Glos, A. Kundu, J. A. Miszczak, Ö. Salehi, and Z. Zimborás, Physical Review A **105**, 022441 (2022).

[252] C. Ding, Y.-F. Niu, W.-S. Bao, and H.-L. Huang, arXiv:2109.06805 (2021).

[253] Y. Du, Y. Qian, X. Wu, and D. Tao, IEEE Transactions on Quantum Engineering **3**, 1 (2022).

[254] Y.-F. Niu, S. Zhang, C. Ding, W.-S. Bao, and H.-L. Huang, arXiv:2208.00450 (2022).

[255] J. Hornibrook, J. Colless, I. C. Lamb, S. Pauka, H. Lu, A. Gossard, J. Watson, G. Gardner, S. Fallahi, M. Manfra, *et al.*, Physical Review Applied **3**, 024010 (2015).

[256] F. Sebastiano, H. Homulle, B. Patra, R. Incandela, J. van Dijk, L. Song, M. Babaie, A. Vladimirescu, and E. Charbon, in *Proceedings of the 54th Annual Design Automation Conference 2017* (2017) pp. 1–6.

[257] M. Mehrpoo, B. Patra, J. Gong, J. van Dijk, H. Homulle, G. Kiene, A. Vladimirescu, F. Sebastiano, E. Charbon, and M. Babaie, in *2019 IEEE International Symposium on Circuits and Systems (ISCAS)* (IEEE, 2019) pp. 1–5.

[258] R. N. Das, J. Yoder, D. Rosenberg, D. Kim, D. Yost, J. Mallek, D. Hover, V. Bolkhovsky, A. Kerman, and W. Oliver, in *2018 IEEE 68th Electronic Components and Technology Conference (ECTC)* (IEEE, 2018) pp. 504–514.

[259] F. Sebastiano, H. A. Homulle, J. P. van Dijk, R. M. Incandela, B. Patra, M. Mehrpoo, M. Babaie, A. Vladimirescu, and E. Charbon, in *2017 7th IEEE International Workshop on Advances in Sensors and Interfaces (IWASI)* (IEEE, 2017) pp. 59–62.

[260] D. Qin, X. Xu, and Y. Li, Chinese Physics B **31**, 090306 (2022).

[261] W. J. Huggins, S. McArdle, T. E. O'Brien, J. Lee, N. C. Rubin, S. Boixo, K. B. Whaley, R. Babbush, and J. R. McClean, Physical Review X **11**, 041036 (2021).

[262] Z. Cai, npj Quantum Information **7**, 80 (2021).

[263] Y. Li and S. C. Benjamin, Physical Review X **7**, 021050 (2017).

[264] K. Temme, S. Bravyi, and J. M. Gambetta, Physical Review Letters **119**, 180509 (2017).

[265] M. Huo and Y. Li, Communications in Theoretical Physics **73**, 075101 (2021).

[266] C. Song, J. Cui, H. Wang, J. Hao, H. Feng, and Y. Li, Science Advances **5**, eaaw5686 (2019).

[267] S. Zhang, Y. Lu, K. Zhang, W. Chen, Y. Li, J.-N. Zhang, and K. Kim, Nature Communications **11**, 587 (2020).

[268] Z. Cai, arXiv:2110.05389 (2021).

[269] R. Takagi, Physical Review Research **3**, 033178 (2021).

[270] R. Takagi, S. Endo, S. Minagawa, and M. Gu, npj Quantum Information **8**, 114 (2022).

[271] S. T. Merkel, J. M. Gambetta, J. A. Smolin, S. Poletto, A. D. Córcoles, B. R. Johnson, C. A. Ryan, and M. Steffen, Physical Review A **87**, 062119 (2013).

[272] D. Greenbaum, arXiv:1509.02921 (2015).

[273] R. Harper, S. T. Flammia, and J. J. Wallman, Nature Physics **16**, 1184 (2020).

[274] S. T. Flammia, arXiv:2108.05803 (2021).

[275] S. T. Flammia and J. J. Wallman, ACM Transactions on Quantum Computing **1**, 1 (2020).

[276] J. J. Wallman and J. Emerson, Physical Review A **94**, 052325 (2016).

[277] A. Hashim, R. K. Naik, A. Morvan, J.-L. Ville, B. Mitchell, J. M. Kreikebaum, M. Davis, E. Smith, C. Iancu, K. P. O'Brien, I. Hincks, J. J. Wallman, J. Emerson, and I. Siddiqi, Physical Review X **11**, 041039 (2021).

[278] A. Erhard, J. J. Wallman, L. Postler, M. Meth, R. Stricker, E. A. Martinez, P. Schindler, T. Monz, J. Emerson, and R. Blatt, Nature Communications **10**, 5347 (2019).

[279] S. Ferracin, A. Hashim, J.-L. Ville, R. Naik, A. Carignan-Dugas, H. Qassim, A. Morvan, D. I. Santiago, I. Siddiqi, and J. J. Wallman, arXiv:2201.10672 (2022).

[280] E. van den Berg, Z. K. Minev, A. Kandala, and K. Temme, arXiv:2201.09866 (2022).

[281] J. Sun, X. Yuan, T. Tsunoda, V. Vedral, S. C. Benjamin, and S. Endo, Physical Review Applied **15**, 034026 (2021).

[282] A. Lowe, M. H. Gordon, P. Czarnik, A. Arrasmith, P. J. Coles, and L. Cincio, Physical Review Research **3**, 033098 (2021).

[283] D. Bultrini, M. H. Gordon, P. Czarnik, A. Arrasmith, P. J. Coles, and L. Cincio, arXiv:2107.13470 (2021).

[284] T. Giurgica-Tiron, Y. Hindy, R. LaRose, A. Mari, and W. J. Zeng, in *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)* (2020) pp. 306–316.

[285] C. Cao, Y. Yu, Z. Wu, N. Shannon, B. Zeng, and R. Joynt, Quantum Science and Technology **8**, 015004 (2022).

[286] M. Krebsbach, B. Trauzettel, and A. Calzona, arXiv:2201.08080 (2022).

[287] C. Cirstoiu, S. Dilkes, D. Mills, S. Sivarajah, and R. Duncan, arXiv:2204.09725 (2022).

[288] E. F. Dumitrescu, A. J. McCaskey, G. Hagen, G. R. Jansen, T. D. Morris, T. Papenbrock, R. C. Pooser, D. J. Dean, and P. Lougovski, Physical Review Letters **120**, 210501 (2018).

[289] A. He, B. Nachman, W. A. de Jong, and C. W. Bauer, Physical Review A **102**, 012426 (2020).

[290] V. R. Pascuzzi, A. He, C. W. Bauer, W. A. de Jong, and B. Nachman, Physical Review A **105**, 042406 (2022).

[291] Y. Kim, C. J. Wood, T. J. Yoder, S. T. Merkel, J. M. Gambetta, K. Temme, and A. Kandala, arXiv:2108.09197 (2021).

[292] M. R. Geller and Z. Zhou, Physical Review A **88**, 012314 (2013).

[293] Z. Cai and S. C. Benjamin, Scientific Reports **9**, 11281 (2019).

[294] K. Schultz, R. LaRose, A. Mari, G. Quiroz, N. Shammah, B. D. Clader, and W. J. Zeng, arXiv:2201.11792 (2022).

[295] A. Mari, N. Shammah, and W. J. Zeng, Physical Review A **104**, 052607 (2021).

[296] M. Otten and S. K. Gray, Physical Review A **99**, 012338 (2019).

[297] P. Czarnik, A. Arrasmith, P. J. Coles, and L. Cincio, Quantum **5**, 592 (2021).

[298] A. Strikis, D. Qin, Y. Chen, S. C. Benjamin, and Y. Li, PRX Quantum **2**, 040330 (2021).

[299] A. Zlokapa and A. Gheorghiu, arXiv:2005.10811 (2020).

[300] A. Zhukov and W. Pogosov, Quantum Information Processing **21**, 93 (2022).

[301] M. Urbanek, B. Nachman, V. R. Pascuzzi, A. He, C. W. Bauer, and W. A. de Jong, Physical Review Letters **127**, 270502 (2021).

[302] J. Vovrosh, K. E. Khosla, S. Greenaway, C. Self, M. S. Kim, and J. Knolle, Physical Review E **104**, 035309 (2021).

[303] A. Montanaro and S. Stanisic, arXiv:2102.02120 (2021).

[304] S.-X. Zhang, Z.-Q. Wan, C.-Y. Hsieh, H. Yao, and S. Zhang, "Variational quantum-neural hybrid error mitigation," (2021), arXiv:2112.10380.

[305] P. Czarnik, M. McKerns, A. T. Sornborger, and L. Cincio, arXiv:2204.07109 (2022).

[306] C. Kim, K. D. Park, and J.-K. Rhee, IEEE Access **8**, 188853 (2020).

[307] Y. Chen, M. Farahzad, S. Yoo, and T.-C. Wei, Physical Review A **100**, 052315 (2019).

[308] S. Bravyi, S. Sheldon, A. Kandala, D. C. Mckay, and J. M. Gambetta, Physical Review A **103**, 042605 (2021).

[309] F. B. Maciejewski, F. Baccari, Z. Zimborás, and M. Oszmaniec, Quantum **5**, 464 (2021).

[310] F. B. Maciejewski, Z. Zimborás, and M. Oszmaniec, Quantum **4**, 257 (2020).

[311] J. S. Lundeen, A. Feito, H. Coldenstrodt-Ronge, K. L. Pregnell, C. Silberhorn, T. C. Ralph, J. Eisert, M. B. Plenio, and I. A. Walmsley, Nature Physics **5**, 27 (2009).

[312] B. Nachman, M. Urbanek, W. A. de Jong, and C. W. Bauer, npj Quantum Information **6**, 84 (2020).

[313] I. Ouadah and H. Benaissa, *Dealing with Quantum Computer Readout Noise Through High Energy Physics Unfolding Methods*, Ph.D. thesis (2021).

[314] P. D. Nation, H. Kang, N. Sundaresan, and J. M. Gambetta, PRX Quantum **2**, 040326 (2021).

[315] B. Yang, R. Raymond, and S. Uno, arXiv:2201.11046 (2022).

[316] L. Funcke, T. Hartung, K. Jansen, S. Kühn, P. Stornati, and X. Wang, Physical Review A **105**, 062404 (2022).

[317] M. R. Geller, Quantum Science and Technology **5**, 03LT01 (2020).

[318] M. R. Geller and M. Sun, arXiv:2001.09980 (2020).

[319] R. Hicks, C. W. Bauer, and B. Nachman, Physical Review A **103**, 022407 (2021).

[320] A. W. R. Smith, K. E. Khosla, C. N. Self, and M. S. Kim, Science Advances **7**, eabi8009 (2021).

[321] E. van den Berg, Z. K. Minev, and K. Temme, Physical Review A **105**, 032620 (2022).

[322] S. Tang, C. Zheng, and K. Wang, arXiv:2206.13743 (2022).

[323] R. Hicks, B. Kobrin, C. W. Bauer, and B. Nachman, Physical

[324] J. Kim, B. Oh, Y. Chong, E. Hwang, and D. K. Park, New Journal of Physics **24**, 073009 (2022).

[325] B. Koczor, Physical Review X **11**, 031057 (2021).

[326] J. Cotler, S. Choi, A. Lukin, H. Gharibyan, T. Grover, M. E. Tai, M. Rispoli, R. Schittko, P. M. Preiss, A. M. Kaufman, M. Greiner, H. Pichler, and P. Hayden, Physical Review X **9**, 031013 (2019).

[327] Z. Cai, arXiv:2107.07279 (2021).

[328] Y. Xiong, S. X. Ng, and L. Hanzo, IEEE Transactions on Communications **70**, 1927 (2022).

[329] H.-Y. Hu, R. LaRose, Y.-Z. You, E. Rieffel, and Z. Wang, arXiv:2203.07263 (2022).

[330] A. Seif, Z.-P. Cian, S. Zhou, S. Chen, and L. Jiang, arXiv:2203.07309 (2022).

[331] M. Huo and Y. Li, Physical Review A **105**, 022427 (2022).

[332] P. Czarnik, A. Arrasmith, L. Cincio, and P. J. Coles, arXiv:2102.06056 (2021).

[333] B. Koczor, New Journal of Physics **23**, 123047 (2021).

[334] R. Huang, Hsin-Yuanand Kueng and J. Preskill, Nature Physics **16**, 1050 (2020).

[335] H.-Y. Huang, R. Kueng, and J. Preskill, Physical Review Letters **127**, 030503 (2021).

[336] A. Elben, R. Kueng, H.-Y. R. Huang, R. van Bijnen, C. Kokail, M. Dalmonte, P. Calabrese, B. Kraus, J. Preskill, P. Zoller, and B. Vermersch, Physical Review Letters **125**, 200501 (2020).

[337] N. Yoshioka, H. Hakoshima, Y. Matsuzaki, Y. Tokunaga, Y. Suzuki, and S. Endo, Physical Review Letters **129**, 020502 (2022).

[338] S. McArdle, X. Yuan, and S. Benjamin, Physical Review Letters **122**, 180501 (2019).

[339] M. Urbanek, D. Camps, R. Van Beeumen, and W. A. de Jong, Journal of Chemical Theory and Computation **16**, 5425 (2020).

[340] X. Bonet-Monroig, R. Sagastizabal, M. Singh, and T. E. O'Brien, Physical Review A **98**, 062339 (2018).

[341] Z. Cai, Quantum **5**, 548 (2021).

[342] R. Sagastizabal, X. Bonet-Monroig, M. Singh, M. A. Rol, C. C. Bultink, X. Fu, C. H. Price, V. P. Ostroukh, N. Muthusubramanian, A. Bruno, M. Beekman, N. Haider, T. E. O'Brien, and L. DiCarlo, Physical Review A **100**, 010302 (2019).

[343] W. J. Huggins, J. R. McClean, N. C. Rubin, Z. Jiang, N. Wiebe, K. B. Whaley, and R. Babbush, npj Quantum Information **7**, 23 (2021).

[344] T. E. O'Brien, S. Polla, N. C. Rubin, W. J. Huggins, S. McArdle, S. Boixo, J. R. McClean, and R. Babbush, PRX Quantum **2**, 020317 (2021).

[345] A. Gonzales, R. Shaydulin, Z. Saleem, and M. Suchara, arXiv:2206.00215 (2022).

[346] A. Hashim, *Noise Tailoring for Enhancing the Capabilities of Quantum Computers*, Ph.D. thesis, UC Berkeley (2022).

[347] B. Zhang, S. Majumder, P. H. Leung, S. Crain, Y. Wang, C. Fang, D. M. Debroy, J. Kim, and K. R. Brown, Physical Review Applied **17**, 034074 (2022).

[348] S. Majumder, C. G. Yale, T. D. Morris, D. S. Lobser, A. D. Burch, M. N. H. Chow, M. C. Revelle, S. M. Clark, and R. C. Pooser, arXiv:2205.14225 (2022).

[349] V. Leyton-Ortega, S. Majumder, and R. C. Pooser, arXiv:2204.12407 (2022).

[350] M. Otten and S. K. Gray, npj Quantum Information **5**, 11 (2019).

[351] J. R. McClean, Z. Jiang, N. C. Rubin, R. Babbush, and H. Neven, Nature Communications **11**, 636 (2020).

[352] E. L. Hahn, Physical Review Journals Archive **80**, 580 (1950).

[353] L. Viola, E. Knill, and S. Lloyd, Physical Review Letters **82**, 2417 (1999).

[354] M. J. Biercuk, H. Uys, A. P. VanDevender, N. Shiga, W. M. Itano, and J. J. Bollinger, Nature **458**, 996 (2009).

[355] G. Quiroz and D. A. Lidar, Physical Review A **88**, 052306 (2013).

[356] H. Qi, J. P. Dowling, and L. Viola, Quantum Information Processing **16**, 272 (2017).

[357] J. Zeng, X.-H. Deng, A. Russo, and E. Barnes, New Journal of Physics **20**, 033011 (2018).

[358] G. S. Ravi, K. N. Smith, P. Gokhale, A. Mari, N. Earnest, A. Javadi-Abhari, and F. T. Chong, in *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)* (2022) pp. 288–303.

[359] K. N. Smith, G. S. Ravi, P. Murali, J. M. Baker, N. Earnest, A. Javadi-Abhari, and F. T. Chong, arXiv:2105.01760 (2021).

[360] G. Aleksandrowicz, T. Alexander, P. Barkoutsos, L. Bello, Y. Ben-Haim, D. Bucher, F. J. Cabrera-Hernández, J. Carballo-Franquis, A. Chen, C.-F. Chen, *et al.*, "Qiskit: An Open-source Framework for Quantum Computing," (2019).

[361] D. Qin, Y. Chen, and Y. Li, arXiv:2112.06255 (2021).

[362] Y. Quek, D. S. França, S. Khatri, J. J. Meyer, and J. Eisert, arXiv:2210.11505 (2022).

[363] R. Takagi, H. Tajima, and M. Gu, "Universal sampling lower bounds for quantum error mitigation," (2022), arXiv:2208.09178.

[364] A. Zulehner, A. Paler, and R. Wille, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **38**, 1226 (2019).

[365] A. Kissinger and A. M.-v. de Griend, arXiv:1904.00633 (2019).

[366] B. Nash, V. Gheorghiu, and M. Mosca, Quantum Science and Technology **5**, 025010 (2020).

[367] B. Wu, X. He, S. Yang, L. Shou, G. Tian, J. Zhang, and X. Sun, arXiv:1910.14478 (2019).

[368] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, Physical Review A **52**, 3457 (1995).

[369] E. Knill, quant-ph/9508006 (1995), 10.48550/ARXIV.QUANT-PH/9508006.

[370] J. J. Vartiainen, M. Möttönen, and M. M. Salomaa, Physical Review Letters **92**, 177902 (2004).

[371] V. V. Shende, I. L. Markov, and S. S. Bullock, Physical Review A **69**, 062321 (2004).

[372] M. Möttönen, J. J. Vartiainen, V. Bergholm, and M. M. Salomaa, Physical Review Letters **93**, 130502 (2004).

[373] V. V. Shende, S. S. Bullock, and I. L. Markov, in *Proceedings of the 2005 Asia and South Pacific Design Automation Conference*, ASP-DAC '05 (Association for Computing Machinery, New York, NY, USA, 2005) p. 272–275.

[374] S. Shannon, *Trends in Quantum Computing Research* (Nova Science Publishers, Inc., USA, 2006).

[375] M. Mottonen and J. J. Vartiainen, quant-ph/0504100 (2005).

[376] S. Johri, S. Debnath, A. Mocherla, A. Singk, A. Prakash, J. Kim, and I. Kerenidis, npj Quantum Information **7**, 122 (2021).

[377] X. Sun, G. Tian, S. Yang, P. Yuan, and S. Zhang, arXiv:2108.06150 (2021).

[378] V. Kliuchnikov, D. Maslov, and M. Mosca, arXiv:1206.5236 (2012).

[379] B. Giles and P. Selinger, Physical Review A **87**, 032332 (2013).

[380] C. M. Dawson and M. A. Nielsen, arXiv preprint quant-

[381] ph/0505030 (2005).

[381] R. Cleve and J. Watrous, in Proceedings 41st Annual Symposium on Foundations of Computer Science , 526 (2000).

[382] C. Moore and M. Nilsson, SIAM Journal on Computing 31, 799 (2001).

[383] P. Selinger, Physical Review A 87, 042302 (2013).

[384] M. Amy, D. Maslov, and M. Mosca, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 33, 1476 (2014).

[385] J. Jiang, X. Sun, S.-H. Teng, B. Wu, K. Wu, and J. Zhang, "Optimal space-depth trade-off of cnot circuits in quantum logic synthesis," in Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 213–229.

[386] C. Aravanis, G. Korpas, and J. Marecek, arXiv e-prints , arXiv:2209.14356 (2022), arXiv:2209.14356 [quant-ph].

[387] D. Maslov and W. Yang, arXiv e-prints , arXiv:2210.16195 (2022), arXiv:2210.16195 [quant-ph].

[388] Y. Hirata, M. Nakanishi, S. Yamashita, and Y. Nakashima, in 2009 Third International Conference on Quantum, Nano and Micro Technologies, pp. 26–33.

[389] A. Shafaei, M. Saeedi, and M. Pedram, in 2014 19th Asia and South Pacific Design Automation Conference (ASP-DAC), pp. 495–500.

[390] A. M. Childs, E. Schoute, and C. M. Unsal, in 14th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2019), Leibniz International Proceedings in Informatics (LIPIcs), Vol. 135, edited by W. van Dam and L. Mancinska (Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2019) pp. 3:1–3:24.

[391] D. Maslov, S. M. Falconer, and M. Mosca, arXiv e-prints , quant-ph/0703256 (2007), arXiv:quant-ph/0703256 [quant-ph].

[392] A. Lye, R. Wille, and R. Drechsler, in The 20th Asia and South Pacific Design Automation Conference (2015) pp. 178–183.

[393] A. Shafaei, M. Saeedi, and M. Pedram, in 2013 50th ACM/EDAC/IEEE Design Automation Conference (DAC), pp. 1–6.

[394] R. Wille, L. Burgholzer, and A. Zulehner, in 2019 56th ACM/IEEE Design Automation Conference (DAC) (2019) pp. 1–6.

[395] P. Zhu, X. Cheng, and Z. Guan, Quantum Information Processing 19, 391 (2020).

[396] C. Zhang, A. B. Hayes, L. Qiu, Y. Jin, Y. Chen, and E. Z. Zhang, in Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS '21 (Association for Computing Machinery, New York, NY, USA, 2021) p. 360–374.

[397] F. A. Aloul, B. A. Rawi, and M. Aboelaze, in 2006 3rd International Conference on Electrical and Electronics Engineering (2006) pp. 1–4.

[398] G. Li, Y. Ding, and Y. Xie, in Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS '19 (Association for Computing Machinery, New York, NY, USA, 2019) p. 1001–1014.

[399] X. Zhou, S. Li, and Y. Feng, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 39, 4683 (2020).

[400] X. Zhou, Y. Feng, and S. Li, ACM Trans. Des. Autom. Electron. Syst. 27, Article 59 (2022).

[401] M. G. Pozzi, S. J. Herbert, A. Sengupta, and R. D. Mullins, arXiv:2007.15957 (2020).

[402] A. Sinha, U. Azad, and H. Singh, Proceedings of the AAAI Conference on Artificial Intelligence 36, 9935 (2022).

[403] X. Zhou, Y. Feng, and S. Li, in Proceedings of the 39th International Conference on Computer-Aided Design, ICCAD '20 (Association for Computing Machinery, New York, NY, USA, 2020).

[404] A. Paler, L. M. Sasu, A.-C. Florea, and R. Andonie, ACM Transactions on Quantum Computing (2022), 10.1145/3565271.

[405] M. Saeedi, R. Wille, and R. Drechsler, Quantum Information Processing 10, 355 (2011).

[406] R. Wille, O. Keszocze, M. Walter, P. Rohrs, A. Chattopadhyay, and R. Drechsler, in 2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC), pp. 292–297.

[407] D. Cuomo, M. Caleffi, and A. S. Cacciapuoti, IET Quantum Communication 1, 3 (2020), https://ietresearch.onlinelibrary.wiley.com/doi/pdf/10.1049/iet-qtc.2020.0002.

[408] D. Ferrari, A. S. Cacciapuoti, M. Amoretti, and M. Caleffi, IEEE Transactions on Quantum Engineering 2, 1 (2021).

[409] D. Cuomo, M. Caleffi, K. Krsulich, F. Tramonto, G. Agliardi, E. Prati, and A. S. Cacciapuoti, arXiv e-prints , arXiv:2112.14139 (2021), arXiv:2112.14139 [quant-ph].

[410] S. Aaronson and D. Gottesman, Physical Review A 70, 052328 (2004).

[411] S. Aaronson and D. Gottesman, Physical Review A 70, 052328 (2004).

[412] K. Markov, I. Patel, and J. Hayes, Quantum Information and Computation 8, 0282 (2008).

[413] P. E. Hart, N. J. Nilsson, and B. Raphael, IEEE Transactions on Systems Science and Cybernetics 4, 100 (1968).

[414] J. Emerson, R. Alicki, and K. Życzkowski, Journal of Optics B: Quantum and Semiclassical Optics 7, S347 (2005).

[415] E. Magesan, J. M. Gambetta, and J. Emerson, Physical Review Letters 106, 180504 (2011).

[416] C. Dankert, R. Cleve, J. Emerson, and E. Livine, Physical Review A 80, 012304 (2009).

[417] B. Lévi, C. C. López, J. Emerson, and D. G. Cory, Physical Review A 75, 022314 (2007).

[418] E. Magesan, J. M. Gambetta, B. R. Johnson, C. A. Ryan, J. M. Chow, S. T. Merkel, M. P. Da Silva, G. A. Keefe, M. B. Rothwell, T. A. Ohki, et al., Physical Review Letters 109, 080505 (2012).

[419] J. Emerson, M. Silva, O. Moussa, C. Ryan, M. Laforest, J. Baugh, D. G. Cory, and R. Laflamme, Science 317, 1893 (2007).

[420] R. Barends, J. Kelly, A. Veitia, A. Megrant, A. Fowler, B. Campbell, Y. Chen, Z. Chen, B. Chiaro, A. Dunsworth, et al., Physical Review A 90, 030303 (2014).

[421] E. Onorati, A. Werner, and J. Eisert, Physical Review Letters 123, 060501 (2019).

[422] A. Carignan-Dugas, J. J. Wallman, and J. Emerson, Physical Review A 92, 060302 (2015).

[423] A. W. Cross, E. Magesan, L. S. Bishop, J. A. Smolin, and J. M. Gambetta, npj Quantum Information 2, 16012 (2016).

[424] J. Helsen, X. Xue, L. M. Vandersypen, and S. Wehner, npj Quantum Information 5, 71 (2019).

[425] D. S. França and A. Hashagen, Journal of Physics A: Mathematical and Theoretical 51, 395302 (2018).

[426] T. J. Proctor, A. Carignan-Dugas, K. Rudinger, E. Nielsen, R. Blume-Kohout, and K. Young, Physical Review Letters 123, 030503 (2019).

[427] Y. Zhang, W. Yu, P. Zeng, G. Liu, and X. Ma, arXiv:2203.10320 (2022).

[428] J. P. Gaebler, A. M. Meier, T. R. Tan, R. Bowler, Y. Lin, D. Hanneke, J. D. Jost, J. Home, E. Knill, D. Leibfried, et al.,

Physical Review Letters **108**, 260503 (2012).

[429] J. M. Gambetta, A. D. Córcoles, S. T. Merkel, B. R. Johnson, J. A. Smolin, J. M. Chow, C. A. Ryan, C. Rigetti, S. Poletto, T. A. Ohki, *et al.*, Physical Review Letters **109**, 240504 (2012).

[430] D. C. McKay, S. Sheldon, J. A. Smolin, J. M. Chow, and J. M. Gambetta, Physical Review Letters **122**, 200502 (2019).

[431] S. Garion, N. Kanazawa, H. Landa, D. C. McKay, S. Sheldon, A. W. Cross, and C. J. Wood, Physical Review Research **3**, 013204 (2021).

[432] J. Helsen, I. Roth, E. Onorati, A. H. Werner, and J. Eisert, PRX Quantum **3**, 020357 (2022).

[433] S. Boixo, S. V. Isakov, V. N. Smelyanskiy, R. Babbush, N. Ding, Z. Jiang, M. J. Bremner, J. M. Martinis, and H. Neven, Nature Physics **14**, 595 (2018).

[434] C. Neill, P. Roushan, K. Kechedzhi, S. Boixo, S. V. Isakov, V. Smelyanskiy, A. Megrant, B. Chiaro, A. Dunsworth, K. Arya, *et al.*, Science **360**, 195 (2018).

[435] R. Barends, C. Quintana, A. Petukhov, Y. Chen, D. Kafri, K. Kechedzhi, R. Collins, O. Naaman, S. Boixo, F. Arute, *et al.*, Physical Review Letters **123**, 210501 (2019).

[436] S. Aaronson and S. Gunn, arXiv:1910.12085 (2019).

[437] B. Barak, C.-N. Chou, and X. Gao, arXiv:2005.02421 (2020).

[438] J. Chen, D. Ding, C. Huang, and L. Kong, arXiv:2206.08293 (2022).

[439] A. W. Cross, L. S. Bishop, S. Sheldon, P. D. Nation, and J. M. Gambetta, Physical Review A **100**, 032328 (2019).

[440] M. Horodecki, P. Horodecki, and R. Horodecki, Physical Review A **60**, 1888 (1999).

[441] "Quantinuum sets new record with highest ever quantum volume," `https://www.quantinuum.com/pressrelease/quantinuum-sets-new-record-with-highest-ever-quantum-volume`.

[442] "Pushing quantum performance forward with our highest quantum volume yet," `https://research.ibm.com/blog/quantum-volume-256`.

[443] A. Wack, H. Paik, A. Javadi-Abhari, P. Jurcevic, I. Faro, J. Gambetta, and B. Johnson, arXiv:2110.14108 (2021).

[444] T. Proctor, K. Rudinger, K. Young, E. Nielsen, and R. Blume-Kohout, Nature Physics **18**, 75 (2022).

[445] Y. Dong and L. Lin, Physical Review A **103**, 062412 (2021).

[446] T. Tomesh, P. Gokhale, V. Omole, G. S. Ravi, K. N. Smith, J. Viszlai, X.-C. Wu, N. Hardavellas, M. R. Martonosi, and F. T. Chong, in *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)* (IEEE, 2022) pp. 587–603.

[447] W. van der Schoot, D. Leermakers, R. Wezeman, N. Neumann, and F. Phillipson, in *2022 IEEE International Conference on Quantum Software (QSW)* (IEEE, 2022) pp. 9–16.

[448] T. Lubinski, S. Johri, P. Varosy, J. Coleman, L. Zhao, J. Necaise, C. H. Baldwin, K. Mayer, and T. Proctor, arXiv:2110.03137 (2021).

[449] "Algorithmic qubits: A better single-number metric," `https://ionq.com/posts/february-23-2022-algorithmic-qubits`.

[450] I. L. Markov, A. Fatima, S. V. Isakov, and S. Boixo, arXiv:1807.10749 (2018).

[451] K. De Raedt, K. Michielsen, H. De Raedt, B. Trieu, G. Arnold, M. Richter, T. Lippert, H. Watanabe, and N. Ito, Computer Physics Communications **176**, 121 (2007).

[452] M. Smelyanskiy, N. P. Sawaya, and A. Aspuru-Guzik, arXiv:1601.07195 (2016).

[453] T. Häner and D. S. Steiger, in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '17 (Association for Computing Machinery, New York, NY, USA, 2017).

[454] E. Pednault, J. A. Gunnels, G. Nannicini, L. Horesh, T. Magerlein, E. Solomonik, E. W. Draeger, E. T. Holland, and R. Wisnieff, arXiv:1710.05867 (2017).

[455] D. S. Steiger, T. Häner, and M. Troyer, Quantum **2**, 49 (2018).

[456] X.-Z. Luo, J.-G. Liu, P. Zhang, and L. Wang, Quantum **4**, 341 (2020).

[457] T. Jones, A. Brown, I. Bush, and S. C. Benjamin, Scientific Reports **9**, 10736 (2019).

[458] Y. Suzuki, Y. Kawase, Y. Masumura, Y. Hiraga, M. Nakadai, J. Chen, K. M. Nakanishi, K. Mitarai, R. Imai, S. Tamiya, T. Yamamoto, T. Yan, T. Kawakubo, Y. O. Nakagawa, Y. Ibe, Y. Zhang, H. Yamashita, H. Yoshimura, A. Hayashi, and K. Fujii, Quantum **5**, 559 (2021).

[459] R. Orús, Annals of physics **349**, 117 (2014).

[460] G. Vidal, Physical Review Letters **91**, 147902 (2003).

[461] G. Vidal, Physical Review Letters **93**, 040502 (2004).

[462] A. J. Daley, C. Kollath, U. Schollwöck, and G. Vidal, Journal of Statistical Mechanics: Theory and Experiment **2004**, P04005 (2004).

[463] F. Verstraete and J. I. Cirac, cond-mat/0407066 (2004).

[464] Y.-Y. Shi, L.-M. Duan, and G. Vidal, Physical Review A **74**, 022320 (2006).

[465] G. Vidal, Physical Review Letters **99**, 220405 (2007).

[466] U. Schollwöck, Annals of Physics **326**, 96 (2011).

[467] S. R. White, Physical Review Letters **69**, 2863 (1992).

[468] S. R. White, Physical Review B **48**, 10345 (1993).

[469] A. McCaskey, E. Dumitrescu, M. Chen, D. Lyakh, and T. Humble, PLOS ONE **13**, 1 (2018).

[470] D. S. Wang, C. D. Hill, and L. C. Hollenberg, Quantum Information Processing **16**, 176 (2017).

[471] A. Dang, C. D. Hill, and L. C. Hollenberg, Quantum **3**, 116 (2019).

[472] H.-L. Huang, W.-S. Bao, and C. Guo, Physical Review A **100**, 032305 (2019).

[473] Y. Zhou, E. M. Stoudenmire, and X. Waintal, Physical Review X **10**, 041038 (2020).

[474] H. Shang, L. Shen, Y. Fan, Z. Xu, C. Guo, J. Liu, W. Zhou, H. Ma, R. Lin, Y. Yang, *et al.*, arXiv:2207.03711 (2022).

[475] E. J. Bylaska, D. Song, N. P. Bauman, K. Kowalski, D. Claudino, and T. S. Humble, Frontiers In Chemistry **9**, 26 (2021).

[476] S. Yalouz, B. Senjean, J. Günther, F. Buda, T. E. O'Brien, and L. Visscher, Quantum Science and Technology **6**, 024004 (2021).

[477] D. Z. Manrique, I. T. Khan, K. Yamamoto, V. Wichitwechkarn, and D. M. Ramo, arXiv:2008.08694 (2021).

[478] R. Xia and S. Kais, Quantum Science and Technology **6**, 015001 (2020).

[479] W. Li, Z. Huang, C. Cao, Y. Huang, Z. Shuai, *et al.*, Chemical Science **13**, 8953 (2021).

[480] J. Liu, L. Wan, Z. Li, and J. Yang, Journal of Chemical Theory and Computation **16**, 6904 (2020).

[481] Y. Fan, J. Liu, Z. Li, and J. Yang, The Journal of Physical Chemistry Letters **12**, 8833 (2021).

[482] J. S. Kottmann, P. Schleich, T. Tamayo-Mendoza, and A. Aspuru-Guzik, The Journal of Physical Chemistry Letters **12**, 663 (2021).

[483] C. Cao, J. Hu, W. Zhang, X. Xu, D. Chen, F. Yu, J. Li, H.-S. Hu, D. Lv, and M.-H. Yung, Physical Review A **105**, 062452 (2022).

[484] I. G. Ryabinkin, A. F. Izmaylov, and S. N. Genin, Quantum Science and Technology **6**, 024012 (2021).

[485] Z. Xu, Y. Fan, H. Shang, and C. Guo, arXiv:2211.07983 (2022).

[486] M. B. Hastings, Journal of mathematical physics **50**, 095207 (2009).

[487] Z.-Y. Han, J. Wang, H. Fan, L. Wang, and P. Zhang, Physical Review X **8**, 031012 (2018).

[488] H. C. Jiang, Z. Y. Weng, and T. Xiang, Physical Review Letters **101**, 090603 (2008).

[489] P. Corboz, R. Orús, B. Bauer, and G. Vidal, Physical Review B **81**, 165104 (2010).

[490] J. Jordan, R. Orús, G. Vidal, F. Verstraete, and J. I. Cirac, Physical Review Letters **101**, 250602 (2008).

[491] M. Lubasch, J. I. Cirac, and M.-C. Bañuls, Physical Review B **90**, 064425 (2014).

[492] M. Levin and C. P. Nave, Physical Review Letters **99**, 120601 (2007).

[493] F. Verstraete, V. Murg, and J. I. Cirac, Advances in physics **57**, 143 (2008).

[494] R. Orús and G. Vidal, Physical Review B **80**, 094403 (2009).

[495] Z. Y. Xie, H. C. Jiang, Q. N. Chen, Z. Y. Weng, and T. Xiang, Physical Review Letters **103**, 160601 (2009).

[496] H. H. Zhao, Z. Y. Xie, Q. N. Chen, Z. C. Wei, J. W. Cai, and T. Xiang, Physical Review B **81**, 174411 (2010).

[497] Z. Y. Xie, J. Chen, M. P. Qin, J. W. Zhu, L. P. Yang, and T. Xiang, Physical Review B **86**, 045139 (2012).

[498] B. Villalonga, S. Boixo, B. Nelson, C. Henze, E. Rieffel, R. Biswas, and S. Mandrà, npj Quantum Information **5**, 86 (2019).

[499] B. Villalonga, D. Lyakh, S. Boixo, H. Neven, T. S. Humble, R. Biswas, E. G. Rieffel, A. Ho, and S. Mandrà, Quantum Science and Technology **5**, 034003 (2020).

[500] I. L. Markov and Y. Shi, SIAM Journal on Computing **38**, 963 (2008).

[501] J. Gray and S. Kourtis, Quantum **5**, 410 (2021).

[502] C. Huang, F. Zhang, M. Newman, X. Ni, D. Ding, J. Cai, X. Gao, T. Wang, F. Wu, G. Zhang, *et al.*, Nature Computational Science , 1 (2021).

[503] Y. Chen, Y. Liu, X. Shi, J. Song, X. Liu, L. Gan, C. Guo, H. Fu, D. Chen, and G. Yang, arXiv:2205.00393 (2022).

[504] G. Kalachev, P. Panteleev, and M.-H. Yung, arXiv:2108.05665 (2021).

[505] J. Chen, F. Zhang, C. Huang, M. Newman, and Y. Shi, arXiv:1805.01450 (2018).

[506] M.-C. Chen, R. Li, L. Gan, X. Zhu, G. Yang, C.-Y. Lu, and J.-W. Pan, Physical Review Letters **124**, 080502 (2020).

[507] R. Li, B. Wu, M. Ying, X. Sun, and G. Yang, IEEE Transactions on Parallel and Distributed Systems **31**, 805 (2019).

[508] E. C. G. Sudarshan, P. M. Mathews, and J. Rau, Physical Review Journals Archive **121**, 920 (1961).

[509] T. F. Jordan and E. C. G. Sudarshan, Journal of Mathematical Physics **2**, 772 (1961).

[510] M.-D. Choi, Linear Algebra and its Applications **10**, 285 (1975).

[511] S.-X. Zhang, J. Allcock, Z.-Q. Wan, S. Liu, J. Sun, H. Yu, X.-H. Yang, J. Qiu, Z. Ye, Y.-Q. Chen, *et al.*, arXiv:2205.10091 (2022).

[512] A. Griewank, Optimization Methods and Software **1**, 35 (1992).

[513] C. Guo and D. Poletti, Physical Review E **103**, 013309 (2021).

[514] M. Schuld and N. Killoran, arXiv:2203.01340 (2022).

[515] E. Tang, in *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing* (2019) pp. 217–228.

[516] C. Ding, T.-Y. Bao, and H.-L. Huang, IEEE Transactions on Neural Networks and Learning Systems (2021), 10.1109/TNNLS.2021.3084467.

[517] J. M. Arrazola, A. Delgado, B. R. Bardhan, and S. Lloyd, arXiv:1905.10415 (2019).

[518] T. Peng, A. W. Harrow, M. Ozols, and X. Wu, Physical Review Letters **125**, 150504 (2020).

[519] C. Ying, B. Cheng, Y. Zhao, H.-L. Huang, Y.-N. Zhang, M. Gong, Y. Wu, S. Wang, F. Liang, J. Lin, *et al.*, arXiv:2207.14142 (2022).

[520] H.-L. Huang, A. K. Goswami, W.-S. Bao, and P. K. Panigrahi, SCIENCE CHINA Physics, Mechanics & Astronomy **61**, 060311 (2018).