

QuTracer: Mitigating Quantum Gate and Measurement Errors by Tracing Subsets of Qubits

Peiyi Li*

North Carolina State University
Raleigh, USA
pli11@ncsu.edu

Ji Liu*

Argonne National Laboratory
Lemont, USA
ji.liu@anl.gov

Alvin Gonzales

Argonne National Laboratory
Lemont, USA
agonzales@anl.gov

Zain Hamid Saleem

Argonne National Laboratory
Lemont, USA
zsaleem@anl.gov

Huiyang Zhou

North Carolina State University
Raleigh, USA
hzhou@ncsu.edu

Paul Hovland

Argonne National Laboratory
Lemont, USA
hovland@mcs.anl.gov

Abstract—Quantum error mitigation plays a crucial role in the current noisy-intermediate-scale-quantum (NISQ) era. As we advance towards achieving a practical quantum advantage in the near term, error mitigation emerges as an indispensable component. One notable prior work, Jigsaw, demonstrates that measurement crosstalk errors can be effectively mitigated by measuring subsets of qubits. Jigsaw operates by running multiple copies of the original circuit, each time measuring only a subset of qubits. The localized distributions yielded from measurement subsetting suffer from less crosstalk and are then used to update the global distribution, thereby achieving improved output fidelity.

Inspired by the idea of measurement subsetting, we propose QuTracer, a framework designed to mitigate both gate and measurement errors in subsets of qubits by tracing the states of qubit subsets throughout the computational process. In order to achieve this goal, we introduce a technique, qubit subsetting Pauli checks (QSPC), which utilizes circuit cutting and Pauli Check Sandwiching (PCS) to trace the qubit subsets distribution to mitigate errors. The QuTracer framework can be applied to various algorithms including, but not limited to, VQE, QAOA, quantum arithmetic circuits, QPE, and Hamiltonian simulations. In our experiments, we perform both noisy simulations and real device experiments to demonstrate that QuTracer is scalable and significantly outperforms the state-of-the-art approaches.

I. INTRODUCTION

Quantum computing is rapidly emerging as a transformative technology, offering great potential for chemistry simulations [27], combinatorial optimization [16], machine learning [40], and other domains [8]. Ideal quantum computers with fully fault-tolerant error correction codes [1], [25] remain distant, and we currently find ourselves in the Noisy Intermediate Scale Quantum (NISQ) era [37], characterized by quantum computers comprising of tens to thousands of noisy qubits and limited connectivity.

In the NISQ era, quantum error mitigation has emerged as a promising strategy to deal with errors arising during quantum computation. Instead of fully correcting the errors, we may mitigate these errors to an acceptable level. Various approaches have been proposed to this end, including Zero Noise Extrapolation [18], [42], Clifford Data Regression [10],

Virtual Distillation [22], [26], Symmetry Verification [7], [9], [29], [30], [43], Pauli Check Sandwiching (PCS) [14], [19], [44], and measurement subsetting [11], [13].

Previous work in measurement subsetting [11], [13] made the observation that measuring a subset of qubits leads to lower measurement errors than measuring all the qubits by reducing the measurement crosstalk. The more accurate local distributions from subset measurements can then be used to improve the measurement results of all the qubits, i.e., the global distribution. Inspired by measurement subsetting, we propose the following hypothesis: if we can mitigate the noise of a subset of qubits throughout the entire circuit execution and achieve high-fidelity results for these subsets of qubits, can we significantly improve the overall fidelity? In this paper, we develop a novel qubit subsetting framework, QuTracer, to validate our hypothesis. QuTracer continuously tracks the state of qubit subsets throughout the computational process and mitigates errors along the way. The high-fidelity local distributions can be obtained and used to refine the noisy global distribution. Compared to the previous work on measurement subsetting, QuTracer effectively mitigates both gate and measurement errors. Mitigating noise in a subset of qubits is also much more effective and less costly than directly mitigating the noise on all the qubits.

QuTracer consists of multiple novel ideas. First, we repurpose the circuit cutting technique to track the state of qubit subsets, analogous to watchpoints during program execution. Second, we propose qubit subsetting Pauli check (QSPC) to mitigate the errors on qubit subsets. Our proposed implementation of QSPC has low circuit cost. For example, adding single-qubit subset Pauli checks only requires insertion of single-qubit gates. Third, we propose a multi-layer qubit subsetting approach designed to mitigate errors in circuits requiring multiple layers of Pauli checks. Fourth, we propose a series of novel optimizations specifically tailored for the process of qubit subsetting, offering improvements in efficiency and accuracy.

Our noisy simulator and real-device experimental results confirm the effectiveness of our proposed QuTracer framework.

* The first two authors contribute equally to this work.

Our contributions can be summarized as follows:

- We introduce QuTracer, a qubit subsetting framework that monitors the distribution of a subset of qubits to mitigate errors. It mitigates both gate errors and measurement crosstalk errors, thereby surpassing the capability of measurement subsetting.
- We repurpose circuit cutting to track the quantum states during the circuit execution. This is analogous to enabling watchpoints when debugging classical programs.
- We propose qubit subsetting Pauli Checks (QSPC) to mitigate the errors on qubit subsets. QSPC has a lower cost than the existing Pauli check error detection schemes.
- We propose and incorporate multiple optimizations for qubit subsetting. These optimizations include state preparation reduction, localized gate simulation, gate bypassing, state traceback, false dependency removal, and qubit remapping.
- We demonstrate the scalability and effectiveness of our approach through rigorous experimentation. Our experimental results on both noisy simulators and real quantum devices show that QuTracer significantly outperforms the state-of-the-art approaches.

This paper is organized as follows. In Section II, we review related works and discuss the motivation of qubit subsetting. In Section III, we use the inverse quantum Fourier transform circuit as an example to illustrate the QuTracer framework. In Section IV, we present the theory and the design of qubit subsetting Pauli checks. In Section V, we present the design of the QuTracer framework and discuss the relevant optimizations. In Section VI, we provide the experimental setup. In Section VII, we discuss both the noisy simulation and the real-device results. Finally, Section IX concludes the paper.

II. BACKGROUND AND MOTIVATION

A. Measurement Error Mitigation via Measurement Subsetting

Measurement subsetting, i.e., the JigSaw protocol, aims to reduce the effects of measurement errors [13]. The protocol splits the experiment shots into a set containing all the end measurements and a set containing only partial measurements. The circuits with partial measurements are equivalent to the complete circuit, except that some of the measurement operators at the end are removed. The circuit with all measurements generates a noisy global distribution. The partial measurement circuits generate local distributions that have high local fidelities due to the reduced measurement errors. Then, the local distributions refine the global distribution through a Bayesian recombination method that uses local information to update the global distribution. VarSaw is an improved version of JigSaw intended for variational quantum algorithms (VQA) [11]. It recognizes that various time and spatial redundancies exist in measurement subsetting for VQA circuits. By eliminating such redundancy, VarSaw can achieve better efficiency.

Takeaway: While Jigsaw and Varsaw target measurement errors, gate errors remain unmitigated. The effectiveness of

measurement subsetting is limited for circuits with a high circuit depth, leading to a high gate error rate. Therefore, a subsetting technique that can also effectively mitigate gate errors is desired.

B. Circuit Cutting

Quantum circuit cutting decomposes a payload circuit into smaller fragments such that they can run on smaller quantum devices [3], [4], [34], [35]. Circuit cutting works by measuring a complete basis set on the left side of the cut and preparing the corresponding states on the right side of the cut. To see how this can be done when cutting one qubit, we can fragment an arbitrary quantum state ρ as

$$\rho = \frac{1}{2} \sum_{M \in \mathcal{B}} M \otimes \text{tr}_j(M_j \rho), \quad (1)$$

where \mathcal{B} represents the basis set of 2×2 Pauli matrices, tr_j is the partial trace over qubit j , and M_j represents an operator that acts with Pauli operator M on qubit j and acts with identity I on other qubits. By expanding M in its spectral decomposition, we can decompose ρ into a sum of channels that contains measurements M on qubit j followed by the preparation of quantum states which are the eigenstates of M . When cutting multiple qubits, the number of Pauli basis scales exponentially $O(4^n)$ with the number of the cutting qubits n .

Measuring the probability distribution of qubits at the cut provides complete information about the quantum state at that position. In conventional circuit cutting, this state information is used to reconstruct the state after the circuit cutting. However, we make a key observation that the state information at the cut can also be leveraged to monitor the execution of the program and to facilitate error mitigation.

Takeaway: Circuit cutting, rather than being solely used for dividing a circuit into multiple subcircuits, can be repurposed to monitor the quantum states throughout the execution of a circuit.

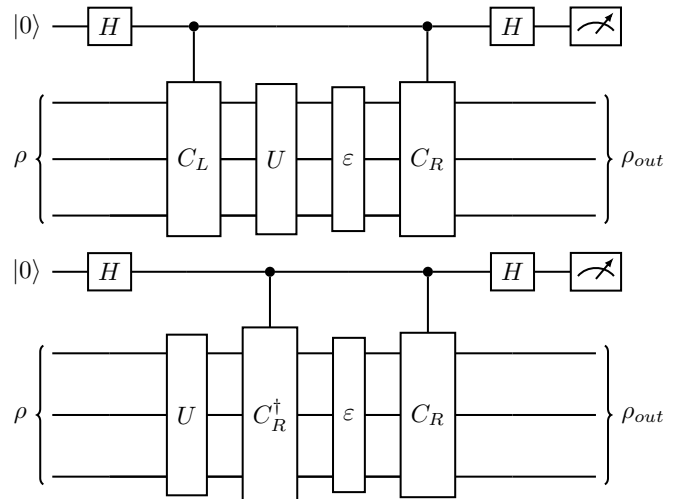


Fig. 1: General idea of Pauli Check Sandwicing (PCS). ϵ is the noise map due to U . The two circuits are equivalent as a result of Eq. (3), and the sandwiched ϵ can be seen as a transformed noise map. The gates that prepare the input state ρ are not shown.

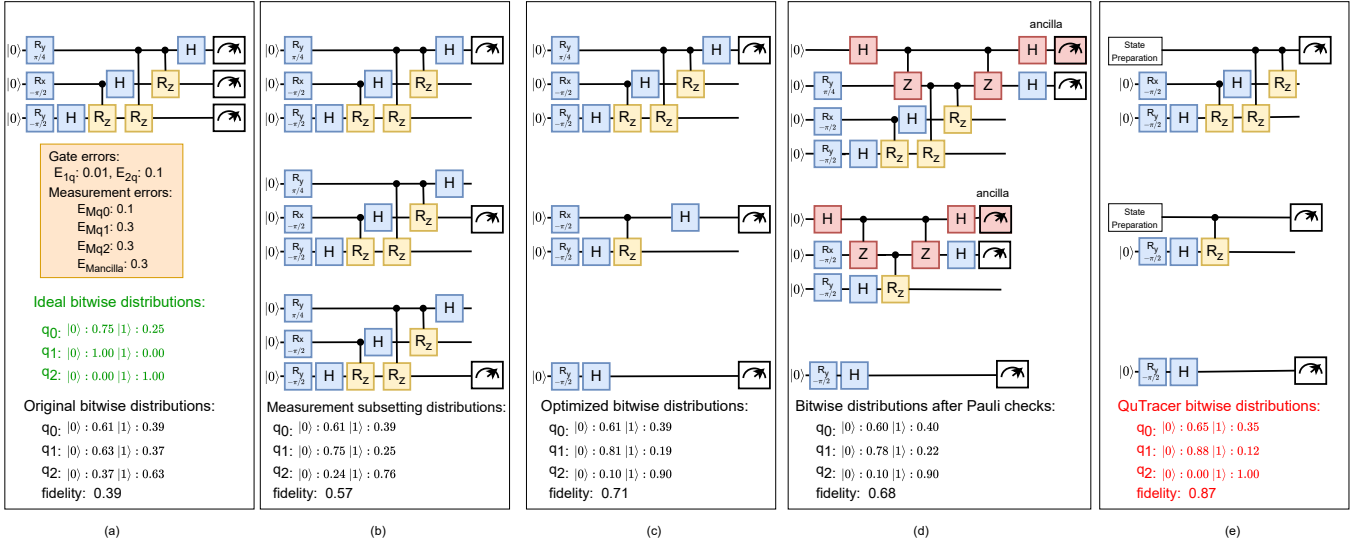


Fig. 2: QuTracer optimized circuits and the corresponding bitwise distributions (i.e., local distribution with subset size of 1) and output Hellinger fidelity (i.e., fidelity of global distribution). (a) The original iQFT circuit that generates the noisy global distribution; (b) Circuit copies with a measurement subsetting size of one; (c) Optimized circuit copies of (b) by removing the gates that the measurement subset has no dependence upon; (d) Circuit copies with PCS to mitigate gate errors; (e) The QuTracer optimized circuit copies of (d).

C. Pauli Check Sandwiching (PCS) and SQEM framework

PCS [19] is a technique designed to detect errors in quantum circuits. This method involves applying a set of checks that have known transformations with the payload circuit to mitigate the errors.

The n -qubit Pauli group P_n is defined as

$$P_n = \{I, X, Y, Z\}^{\otimes n} \times \{\pm 1, \pm i\}. \quad (2)$$

Let U denote the payload unitary circuit. Pauli checks are chosen such that the following constraint is satisfied

$$C_R U C_L = U \quad (3)$$

and typically, C_L and C_R are elements of the Pauli group. For a single pair of checks, the PCS scheme applies the circuit shown in Fig. 1 and post-selects on the zero outcome of a Z-basis measurement of the ancilla qubit. Note that the right check along with the post-selection can be recognized as the Hadamard test often used in quantum error correction [32]. The single layer scheme and its equivalent form are depicted in Fig. 1. As long as the error operator anti-commutes with C_R , it can be effectively eliminated as a result of post-selection.

In PCS and other error mitigation techniques, the error mitigation protocol introduces extra errors via additional gates (e.g., the Hadamard test gates) and ancilla qubits. This extra noise degrades the performance of the error mitigation schemes. Simulated Quantum Error Mitigation (SQEM) was developed to minimize the added noise [28] by leveraging circuit cutting. In SQEM, circuit cutting is performed so that the payload subcircuit becomes separated from the quantum error mitigation subcircuit. The payload circuit fragment runs on the quantum hardware and the quantum error mitigation

fragments are simulated on a classical computer. The results of the fragments are recombined via classical post-processing.

Although SQEM addresses the issue of additional noise from mitigation circuits, it inherits the scalability challenges of circuit cutting. When mitigating multi-layer VQE or QAOA circuits, SQEM requires cutting the error mitigation circuit across layers, leading to an overhead that scales exponentially with the number of layers.

Takeaway: PCS is a promising mitigation technique but the extra noise due to its additional gates and ancillas limits its effectiveness. Although the SQEM framework reduces this extra noise of PCS and can be used for shallow VQE circuits, there is a need for a more generalized framework capable of handling multi-layer, complex circuits. The crucial part is to address the scalability resulting from circuit cutting.

III. MOTIVATING EXAMPLE: INVERSE QFT

In this section, we use the inverse quantum Fourier transform (iQFT) circuit as an example to show the key idea behind the QuTracer framework. QFT and iQFT are basic building blocks in many quantum algorithms including Shor's algorithm [41], quantum adder [15], quantum multiplexer [39], quantum phase estimation [24], and HHL algorithm [21]. A three-qubit iQFT circuit is shown in Fig. 2(a). When running on a noisy simulator that incorporates both simulated gate and measurement errors, the Hellinger fidelity of the output state is 0.39 due to the noise in both circuit execution and qubit measurements. The measurement error vary from 0.1 to 0.3.

The key ideas behind our proposed qubit subsetting are that (a) it optimizes the circuit copies generated for qubit subsets, and (b) it enables error mitigation during the computation of these qubit subsets. The circuits in Fig. 2(b) employ measurement subsetting, i.e., Jigsaw, with a subset size of

one, resulting in three separate circuit copies for this three-qubit iQFT. Jigsaw maps the qubit subset to qubits with lower measurement errors, and the resulting output fidelity becomes 0.57 after combining the more accurate local distributions. In comparison, QuTracer further optimizes these circuits. First, the gates that the measurement operator has no dependency on can be removed. Second, as we only measure the probability distribution on the Z basis, the Rz gates, which only induce a phase change, can also be eliminated. The optimized circuits are shown in Fig. 2(c). Note that this optimization opportunity is unique and arises specifically when the focus is on the distribution of a subset of qubits. As the optimized circuits contain fewer gates, local distributions can be further improved, and the fidelity of the refined global state becomes 0.71.

After optimizing the circuit, we apply error mitigation schemes. Fig. 2(d) shows the circuits after applying PCS. Here, Pauli-Z checking is used as the Z gate on the control qubit commutes with the control-Z gate to be protected. By introducing an ancilla qubit and the Pauli-Z checking circuit, errors that anti-commute with the Pauli-Z operator (e.g., Pauli-X and Pauli-Y) can be detected. The ancilla qubit and the extra gates are marked in red in the figure. However, these extra operations themselves introduce noise, and there is no guarantee that the noise reduction would outweigh the induced errors due to the extra gates. In our experiment, this added mitigation actually showed worse local distributions than Fig. 2(c), and the output fidelity becomes 0.68. To overcome the noise due to the mitigation circuits, we propose to implement these mitigation operations “virtually.” As elaborated in the following section, we develop a qubit subsetting Pauli check (QSPC) technique, which transforms these operations into a collection of lower-cost circuits that introduce only minimal noise. Consequently, the circuits depicted in Fig. 2(d) are transformed into the more efficient ones shown in Fig. 2(e), which involve state preparation and measurements, but only introduce additional single-qubit gates, which have much lower error rates than two-qubit ones. With QSPC, the local distributions become more accurate as a result of mitigating both gate and measurement errors. The fidelity of the refined global distribution becomes 0.87, a 53% improvement over Jigsaw.

IV. QUBIT SUBSETTING PAULI CHECKS (QSPC)

In this section, we present Qubit Subsetting Pauli Checks (QSPC), a novel approach specifically designed to check a subset of qubits.

A. Intuition

QSPC incorporates two key elements. First, it employs PCS protocol to mitigate gate errors. Second, it virtualizes the PCS circuits by transforming them into an ensemble of state preparation and measurements. This is achieved by leveraging the circuit-cutting idea. As a side benefit, some quantum gates can be replaced with classical computation if they depend solely on the state at a cut. Moreover, as the post-selected output states can be prepared directly from the cut, there is

no need to measure the ancilla qubits, thus QSPC is immune to measurement errors on ancilla qubits. Virtualization of the PCS circuit also offers a unique advantage: it integrates the measurement errors from the original circuit into the error channel protected by the Pauli checks. This results in the virtual Pauli checks effectively mitigating both gate and measurement errors, a significant improvement over the original PCS that only addresses gate errors.

B. Theory

The circuit implementation of the PCS protocol shown in Fig. 1, uses a pair of n-qubit gates $\{C_L, C_R\}$ to check the n-qubit circuit U . The requirement is $C_R U C_L = U$. With the error channel being $\epsilon(\rho) = \sum_i E_i \rho E_i^\dagger$, the post-selected output state of the PCS protocol is [19]:

$$\rho_{out} = \frac{\sum_i [(C_R E_i C_R^\dagger + E_i) U \rho U^\dagger (C_R E_i^\dagger C_R^\dagger + E_i^\dagger)]}{\text{tr}(\sum_i [(C_R E_i C_R^\dagger + E_i) U \rho U^\dagger (C_R E_i^\dagger C_R^\dagger + E_i^\dagger)])} \quad (4)$$

As long as the error operator E_i anticommutes with C_R , $C_R E_i C_R^\dagger + E_i = 0$ and the error can be eliminated as a result of post-selection. When there is no error, i.e., $E_i = I$, $\rho_{out} = U \rho U^\dagger$, meaning that the execution of the gate U is noise free. In QSPC, we use Pauli-Z checks to protect a subset of qubits, the subset size of qubits is set to 1, i.e., $C_R = C_L = Z_j$, where Z_j represents an operator that acts with Pauli operator Z on qubit j and acts with identity I on other qubits, and qubit j is the qubit that we want to protect. Pauli-Z checks capture bitflip errors on the protected qubit as X anti-commutes with Z. The discussion of protecting more than one qubit simultaneously, i.e., setting the qubit subset size to be larger than 1 will be introduced in section V-D.

Generating the output state ρ_{out} shown in Eq. (4) requires ancilla qubit measurement and post-selection, which introduce extra noise. To address the issue, we make the observation that it is sufficient to obtain the expectation value of ρ_{out} with respect to an observable O , $\langle O \rangle = \text{tr}(\rho_{out} O)$, instead of the exact state ρ_{out} . There are two reasons for this. Firstly, in many algorithms, including variational ones, the expectation value of an observable is needed instead of the output distribution. Secondly, measuring the probability of a classical output is actually equivalent to measuring the corresponding observables. The benefit of using $\text{tr}(\rho_{out} O)$ is that it can be computed virtually by leveraging the circuit cutting technique. To compute $\text{tr}(\rho_{out} O)$, we first derive the following relationship: since $C_R U C_L = U$, we have $C_R^\dagger U = U C_L$ and $U^\dagger C_R = C_L^\dagger U^\dagger$. Then, we use Eq. (4) to compute the numerator and denominator of $\text{tr}(\rho_{out} O)$ separately. The numerator of $\text{tr}(\rho_{out} O)$ can be computed from the following four terms:

$$\text{tr}(\sum_i E_i U \rho U^\dagger E_i^\dagger O) \quad (5)$$

$$\text{tr}(\sum_i E_i U C_L \rho U^\dagger E_i^\dagger O C_R) \quad (6)$$

$$\text{tr}(\sum_i E_i U \rho C_L^\dagger U^\dagger E_i^\dagger C_R^\dagger O) \quad (7)$$

$$\text{tr}(\sum_i E_i U C_L \rho C_L^\dagger U^\dagger E_i^\dagger C_R^\dagger O C_R). \quad (8)$$

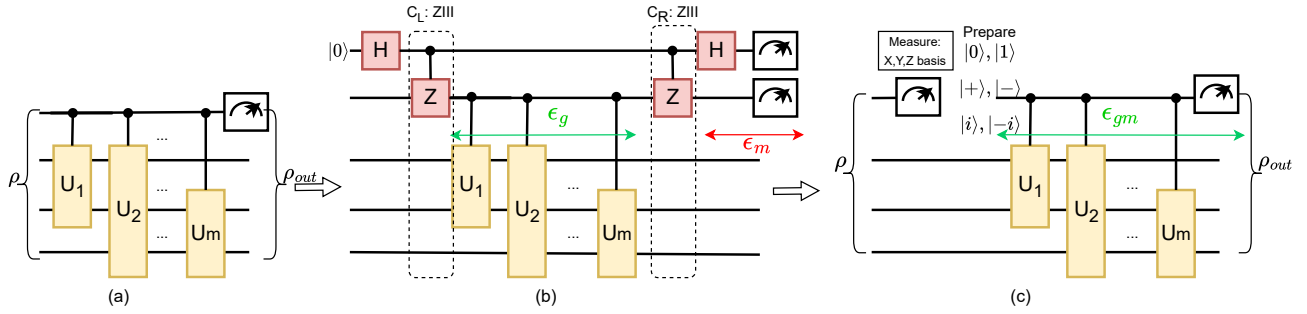


Fig. 3: Qubit subsetting Pauli check for a single qubit. (a) Circuit with one qubit to measure as a result of measurement subsetting; (b) Circuit with Pauli-Check Sandwiching; (c) Circuit with our proposed qubit subsetting Pauli checking. ϵ_g denotes the error channel that consists of gate errors. ϵ_m denotes the measurement error channel. ϵ_{gm} denotes the error channel that consists of both gate and measurement errors. The sequence of gates U_{1-m} sharing the same control qubit exists widely in algorithms such as QFT and QPE. We use this pattern to illustrate QuTracer’s ability to protect a series of operations commuting on a subset of qubits.

These four terms can be obtained by preparing $\rho, C_L\rho, \rho C_L^\dagger$, and $C_L\rho C_L^\dagger$, passing them through a quantum gate U followed by a noisy quantum channel $\epsilon(\rho)$, and measuring four observables $O, OC_R, C_R^\dagger O$, and $C_R^\dagger OC_R$, respectively. This process consists solely of state preparation and measurement and do not involve ancilla qubits and controlled unitary gates which are required in the PCS circuits. Therefore, the errors typically associated with the QSPC checking circuit can be significantly reduced. We will detail the procedure for calculating Term (6); other terms can be calculated by the same procedure.

For Term (6), we need to first prepare $C_L\rho$. But $C_L\rho$ is not a quantum state that can be directly prepared. However, based on Eq. (1), $C_L\rho$ can be decomposed as Eq. (9), which is a linear combination of channels that contains measurements followed by state preparation. This allows us to use a similar process as the wire cutting in quantum circuits to obtain $C_L\rho$.

$$C_L\rho = \frac{1}{2} \sum_{M \in \mathcal{B}} ZM \otimes \text{tr}_j(M_j\rho), \quad (9)$$

Therefore, the preparation of $C_L\rho$ transforms into setting up a series of circuits involving measurements followed by state preparation. In each circuit, the measurement operator M_j is applied to the state ρ , followed by preparing the eigenstates of ZM on the j th qubit. Then based on Term (6), all these circuits go through the quantum gate U and noisy quantum channel $\epsilon(\rho)$, the observable OC_R is measured to obtain the results. Fig. 3(c) gives an example of setting up the circuits to compute Term (6) when the checking qubit is the first qubit. The results from all the circuits are combined linearly to derive the outcome for Term (6).

We can calculate all the terms for the numerator of $\text{tr}(\rho_{out}O)$ by the above procedure. The calculation for the denominator of $\text{tr}(\rho_{out}O)$ is in the same manner with the observable O being the identity I for all the qubits. A maximum of 18 distinct circuits is sufficient to calculate $\text{tr}(\rho_{out}O)$. In some scenarios, we care about the information of the protected qubit of ρ_{out} in all three measurement basis X, Y , and Z . Then in the worst-case scenario, we need to prepare

30 different circuits to calculate $\text{tr}(\rho_{out}X_j), \text{tr}(\rho_{out}Y_j)$, and $\text{tr}(\rho_{out}Z_j)$ by reusing results from some circuits.

C. QSPC vs. SQEM

In our approach, the transformation of the PCS circuit into an ensemble of state preparations and measurements is akin to the process of cutting and simulating Pauli check circuits, as discussed in the SQEM framework [28]. However, the key distinction between our Qubit Subsetting Pauli Checks (QSPC) and the SQEM approach lies in the state reconstruction process. Circuit cutting necessitates preparing and measuring on all bases for complete state reconstruction. For a cut fragment with m measurement locations and n state preparation locations, the standard circuit cutting requires $3^m \times 4^n$ circuit copies, as detailed in [35]. Since there are two measurement locations and one state preparation location when cutting the circuit in Fig. 3(b), SQEM requires 36 circuit copies. In contrast, QSPC directly calculates the necessary state preparations and measurement bases, thus significantly reducing the number of required state preparations and measurements. As mentioned earlier, the calculation requires 18 circuits. Even in the worst case scenario, QSPC surpasses SQEM, requiring 30 circuits. In the experimental section (Section VII), we will present the overhead for various benchmarks and demonstrate the substantial reduction in circuit cost achieved through our method.

D. Measurement Error Mitigation

QSPC offers a unique advantage in that it enables the mitigation of both gate and measurement errors. In the original PCS scheme shown in Fig. 3(b), the error channel ϵ_g consists of gate errors that occur between the two checks. The measurement errors denoted by ϵ_m are not mitigated. However, with QSPC’s “virtual” implementation of checks, the final measurements on the qubits are also sandwiched between these checks. As we prepare states and measure observables in QSPC, the resultant error channel, denoted ϵ_{gm} , consists of both gate and measurement errors. The post-selected output state that we reconstructed is the state that mitigates both gate

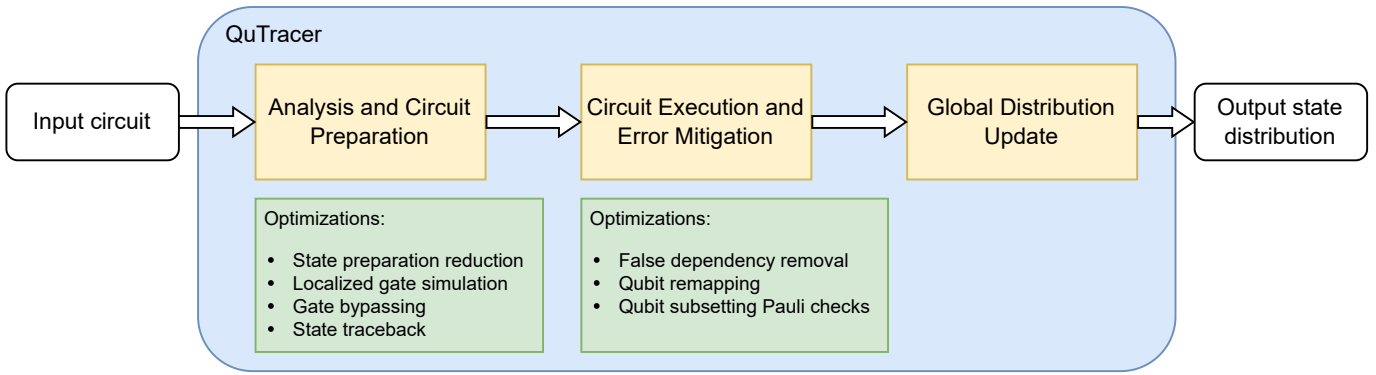


Fig. 4: QuTracer workflow and the optimizations

and measurement errors. This is particularly effective when measurement errors can be viewed as products of bit-flip Pauli X errors. Since these measurement errors anticommute with Pauli Z checks, they can be effectively mitigated using the QSPC method.

In summary, QSPC virtualizes the single-qubit PCS circuit in Fig. 3(b) and converts it to an ensemble of circuits in Fig. 3(c) that only contains additional single-qubit gates, thus eliminating the use of ancilla qubits and reducing the noise in these checking circuits. In addition, the ensemble of circuits mitigates the measurement errors. In the next section, we will discuss how to incorporate QSPC in our error mitigation framework to protect a general quantum circuit.

V. QUTRACER FRAMEWORK

A. Framework

Our QuTracer framework involves continual tracking of the state of qubit subsets throughout their computational process.

In order to track the state during circuit execution, we re-purpose the circuit-cutting technique: for a cut, we actually measure the distribution at the cutting point and prepare the necessary states accordingly. We can insert multiple cut points on a subset of qubits and measure at these points to track their state at each cut point. In other words, the purpose of these cuts is not to separate a circuit but to track the quantum states to ensure the correct execution of the circuit. This is analogous to creating watchpoints when debugging classical programs. Note that having watchpoints on many qubits incurs exceedingly high overhead. On the other hand, watchpoints are a good fit when used on a small subset of qubits as the overhead would be much more manageable. By measuring at the “quantum watchpoints” and error-mitigating the state of a subset of qubits, we can achieve high-fidelity qubit subset distributions.

The workflow of our approach is as follows. Given a quantum circuit, it is first executed to produce the global distribution. Then, QuTracer performs qubit subsetting to produce high-fidelity local distributions and refines the global one. At a high level, QuTracer takes three distinct steps shown in Fig. 4:

- **Analysis and Circuit Preparation:** For a circuit with a subset of qubits of interest, analyze the circuit to determine the cut locations where the measurement of the qubit subset states is needed. To measure the states of the qubit subset, multiple state preparation and measurement circuits will be generated following the circuit cutting protocol.
- **Circuit Execution and Error Mitigation:** Execute the circuits and subsequently update the qubit subset states. During this step, error mitigation techniques are strategically employed to ensure an accurate estimation of the subset state.
- **Global Distribution Update:** Refine the global distribution based on the qubit subset states. The global distribution is updated using the same Bayesian recombination algorithms utilized by SQEM [28].

Within each step, we propose further optimizations as listed in Fig. 4. To facilitate a clear and focused examination of the concepts and optimizations, we limit the size of the qubit subset to one in the following discussion.

B. Single-layer Qubit Subsetting

In this subsection, we use Quantum Phase Estimation (QPE) as an example to illustrate how its execution result can be refined by the QuTracer framework. Fig. 5(a) depicts a quantum phase estimation circuit featuring three ancilla qubits. As only these ancilla qubits are to be measured, qubit subsetting is specifically applied to these three qubits by obtaining the accurate distribution of one ancilla qubit at a time. Here, we focus on the third qubit. As shown in Fig. 5(a), the circuit analyzer strategically inserts three circuit cut points. The criteria for choosing cut points is to divide the gate operations into sets of commuting operations. The reason is that a sequence of commuting operations can be efficiently checked by our Qubit Subsetting Pauli Checks (QSPC) approach. Initially, each cut point requires measurement in three bases and preparation in six states. However, many state preparations and measurements can be optimized. The **Analysis and Circuit Preparation** step incorporates the following optimizations:

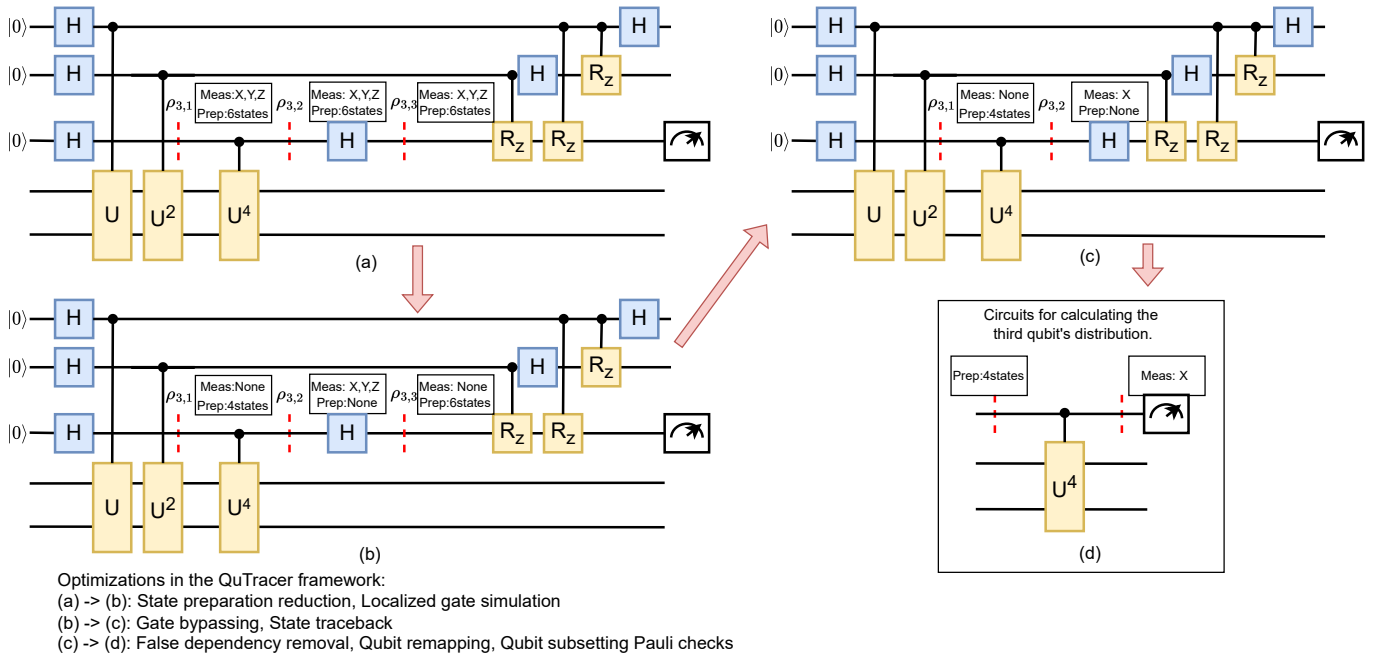


Fig. 5: QuTracer framework for Quantum Phase Estimation (QPE)

State preparation reduction: In Fig. 5(a), we prepare six states $|0\rangle$, $|1\rangle$, $|+\rangle$, $|-\rangle$, $|i\rangle$ and $| -i\rangle$ at the cut point $\rho_{3,1}$ in order to compute $\rho_{3,2}$. The expectation value of states $|-\rangle$ and $| -i\rangle$ can be derived with classical post-processing based on the expectation value of the other four states $|0\rangle$, $|1\rangle$, $|+\rangle$, and $|i\rangle$. Therefore, the number of state preparations in the cut point $\rho_{3,1}$ can be reduced to four.

Localized gate simulation: Since we restrict the size of the subsetting, the localized gates that operate only on the subset of qubits can be efficiently simulated using classical computers. In our example, we track the state on the third qubit. As such, we simulate all the single-qubit gates on the qubit from the start. Therefore, we can compute $\rho_{3,1}$ without the need for any measurement. Similarly, $\rho_{3,3}$ can be derived based on $\rho_{3,2}$ with a classically simulated Hadamard gate. As a result, we can eliminate the state preparation at $\rho_{3,2}$ and the measurement at $\rho_{3,3}$. This optimization also ensures the noiseless execution of the localized gates.

Gate bypassing: Leveraging gate properties, we can reduce the required number of measurements. For example, the controlled gates do not change the Z basis distribution on the control qubit. Since we compute the density matrix $\rho_{3,1}$, there is no need to measure $\rho_{3,2}$ in the Z basis because the sequence of controlled U gates leaves the distribution unchanged. This sequence of gates can be bypassed when tracking the Z basis distribution, thereby ensuring noiseless computation on the Z basis.

State traceback: We can reduce the number of measurements and preparations when we trace back from the final measurements on the computational basis. Since only the measurement in the Z basis is necessary at the circuit's

conclusion, we can omit the measurements on the X and Y bases. For instance, by following our discussion on gate bypassing, on the third qubit, the last two controlled-RZ gates do not alter the distribution on the Z basis. It suffices to acquire only the Z basis information of $\rho_{3,3}$. Since we simulate the single-qubit Hadamard gates, we only need to measure on the X basis for $\rho_{3,2}$ to acquire the Z basis distribution of $\rho_{3,3}$.

After the aforementioned optimizations, the circuit with the necessary preparations and measurements is shown in Fig. 5(c). Only four circuits are needed to calculate the output distribution on the third qubit. In the second step of the process, labeled **Circuit Execution and Error Mitigation**, we further implement the following optimizations:

False dependency removal: As we only need to measure a subset of qubits, we can remove the gates that the subset measurement is not dependent upon. This is similar to identifying causal cones in variational quantum ansatz studies [2], [5]. Nevertheless, there might be false dependencies in the circuit diagram. In the original circuit, the controlled-U gate and controlled- U^2 gate may affect our measurement of $\rho_{3,2}$. By employing gate commuting rules, we can shift these gates after the measurement of $\rho_{3,2}$. Then, it becomes evident that the controlled-U gate and controlled- U^2 gate can be removed. After this optimization, the gates that the $\rho_{3,2}$ measurement is dependent on are shown in Fig. 5(d).

Qubit remapping: Since the optimized circuit to be executed on the hardware is different from and smaller than the original circuit, we can remap it to physical qubits with low noise. The remapping ensures the low noise execution of the circuit. Furthermore, circuits for different subsets of qubits may re-utilize the same set of high-quality qubits. We employ

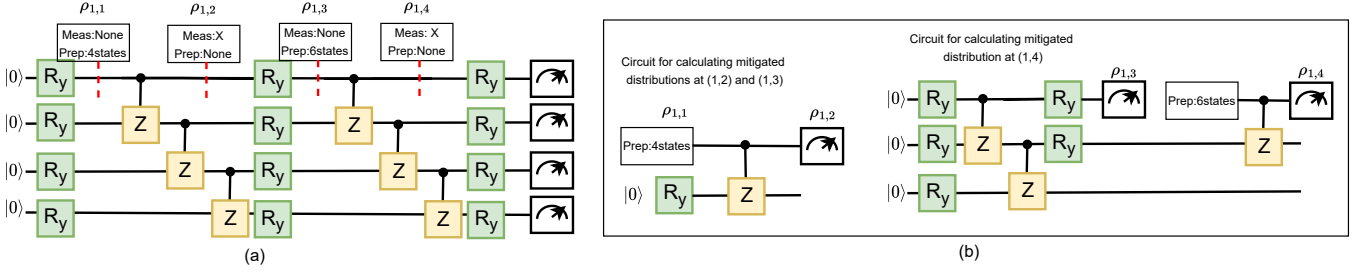


Fig. 6: QuTracer framework for two-layer VQE ansatz

the noise-aware mapping scheme [31] for our re-mapping.

Qubit subsetting Pauli checks: The final circuit shown in Fig. 5(d) can be protected by QSPC. As we prepare the input state in different states and acquire the measurement results from the real device, we can calculate the error mitigated distribution on the X basis following the discussion in Section IV. We start with the ground state $|0\rangle$ and calculate the noiseless density matrix $\rho_{3,1}$. Then, we run the circuit in Fig. 5(d) to acquire the noise-mitigated distribution on the X basis. Next, we simulate the Hadamard gate and bypass the Rz gates to get the final output distribution on the third qubit. In this process, the only step susceptible to hardware noise is the execution of the controlled gate U^4 . The reduction of the circuit size, coupled with QSPC for error mitigation, makes the local distribution of the third qubit substantially more accurate than measurement subsetting.

We can follow a similar procedure to obtain the accurate local distribution of the first and the second qubits. These qubits exhibit similar properties, and each only needs a single-qubit subsetting Pauli check. Notably, this requirement remains consistent regardless of the size of the Quantum Phase Estimation (QPE) algorithm, where the single-qubit distribution only demands a single-qubit subsetting Pauli check per individual qubit. In the next section, we will use the multilayer VQE algorithm as an example to explore the challenges associated with sequentially implementing multiple qubit subsetting Pauli checks.

C. Multi-layer Qubit Subsetting

In this subsection, we illustrate the application of the QuTracer framework on a multi-layer ansatz circuit for variational quantum algorithms. As depicted in Fig. 6(a), the ansatz circuit is composed of layers of single-qubit Y-rotation gates and linear entanglement with CZ gates. In our discussion, we concentrate on the top qubit. As illustrated in Fig. 6(a), four circuit cuts are placed as two circuit cuts are needed per layer to ensure that the check is performed only on the qubit subset. For example, we can use pairs of Pauli Z checks at cut points (1,1) and (1,2) to protect one layer of CZ gates. However, if we use only the cut points (1,1) and (1,4) to check two layers, we cannot find pairs of single-qubit gates C_L and C_R that would satisfy the requirements for Pauli checks. The reason is when a single-qubit Pauli check is specified on one side, the

corresponding check on the other side becomes a multi-qubit operation. Consequently, the operation would not be restricted to the subset, necessitating the tracking of additional qubits beyond the subset.

As we have identified the four cut points, we perform the aforementioned optimizations and construct the quantum circuits. As shown in Fig. 6(b), each layer generates a circuit that can be protected with a single-qubit subsetting Pauli check. For the first circuit, the calculations yield the mitigated distributions at (1,2) and (1,3). However, for the second circuit, we measure the state at (1,3) and obtain an unmitigated density matrix $\rho_{1,3}$, which is then used to calculate the mitigated density matrix $\rho_{1,4}$. The challenge arises when considering the sequential execution of these circuits. If they run separately and in sequence, the mitigation is restricted to one layer only, with no provision to transmit the mitigated data to the subsequent calculation. Attempting to merge these two circuits to simultaneously check the two layers (akin to simultaneously cutting at all four cut points) would lead to an exponential increase in the required state preparation and measurement as the number of layers increases. The primary challenge lies in effectively transferring the mitigated distribution from one layer to the next.

The problem of transferring the mitigated distribution from one layer to the next can be translated into the task of updating the global output distribution based on the local output distributions. Referencing Fig. 6(b), the circuit for calculating $\rho_{1,4}$ yields the output distribution $P = \{00 : a, 01 : b, 10 : c, 11 : d\}$ which is the measurement result for locations (1,3) and (1,4), and P can be viewed as the global output distribution when measuring locations (1,3) and (1,4) simultaneously. Since we already obtain the error-mitigated output distribution at (1,3) represented as $M = \{0 : \alpha, 1 : \beta\}$ based on the calculation in the first layer, we can use M to update the global output distribution P . The global output distribution is updated using the same Bayesian recombination algorithms utilized by SQEM [28]. The updated global output distribution P_{update} has a bitwise distribution for location (1,3) which is the same as the error-mitigated output distribution M . We can then employ the error-mitigated probability P_{update} in the second layer's qubit subsetting calculation, which will mitigate the noise in both layers.

D. Different Subset Sizes

While increasing the subset size captures more global correlation, the noise also increases. The previous measurement subsetting works [11], [13] suggest that a subset size of 2 strikes a balance. Increasing the subset size allows us to “virtualize” multiple Pauli checks concurrently. For instance, the VQE circuit shown in Fig. 6 consists of single-qubit Pauli Z checks. Extending the subset size to 2 allows simultaneous application of IZ and ZI checks, which detects more errors. However, as will be discussed in the following subsection, the classical and quantum overhead scales exponentially with the subset size. We restrict the subset size to one or two in our experiments. When running QuTracer with subset size of two, in the worst case scenario, it requires 30^2 circuits.

For circuits yielding symmetric output states, it’s advisable to use a subset size greater than one. This is because measuring a single qubit typically produces a uniformly distributed bitwise outcome, which does not effectively refine the global distribution from the original circuit. For example, the \mathbb{Z}_2 symmetry in the MaxCut problem and the QAOA ansatz results in output states that are bit-flip invariant, with single qubit distributions being uniformly distributed. Therefore, the subset size should be larger than one. We will show the results for QAOA by setting the subset size to be two in Section VII.

E. Scalability

The cost of our proposed QuTracer framework includes classical circuit analysis, quantum circuit execution, and classical post-processing. It scales linearly with the number of layers in the circuit. Consider the original circuit with n qubits, m layers, k shots, and a qubit subset size of s . First, we consider the overhead in quantum processes. The total number of shots represents the total execution time on a quantum device. For each qubit subsetting Pauli check, we require $O(C^s)$ number of preparation and measurement circuits, whereas C is the number of circuit copies for a single-qubit QSPC. Then, following the discussion in measurement subsetting [13], the number of subsets to be evaluated is $O(n)$. Since we have m layers, the total number of quantum circuits is $O(C^s nm)$. These circuits for the same check $O(C^s)$ need to be executed in sequential, while different checks $O(nm)$ can be parallelized. Since these quantum circuits only involve state preparation and measurement and do not rely on data from classical post-processing, they can be executed simultaneously with classical post-processing. These quantum circuits require fewer shots than the original circuit. As we measure the expectation values, the number of shots [36] scales polynomially with the number of qubits: $k \sim O(n^r/\epsilon^2)$, where r is a constant determined by the quantum circuit and is greater than one, and ϵ is the output error rate. Since we only measure a subset of qubits, the number of shots in each circuit copy ϵ is bounded by $O(\frac{s}{n}k)$ to maintain the same error rate. Therefore, the total number of shots in the QuTracer framework is $O(C^s mk)$, which increases linearly with the number of layers in the circuit. As discussed in Section IV, if we limit the subset size to 1, the number of

shots is upper-bounded by $O(30mk)$. As we will show in Section VII-E, the actual overheads are smaller due to our proposed optimizations. The constant C can be further reduced by utilizing Local Operations and Classical Communication (LOCC), specifically through mid-circuit measurements and classically controlled operations. Following the discussion in [20], we can prepare the states based on the measurement results and reduce the constant C from 30 to 15. However, due to the lower fidelity associated with mid-circuit measurements and classically controlled operations, we did not implement these techniques in our experiments.

Next, we analyze the classical computation overhead. The circuit analysis simply traverses the circuit and has a complexity of $O(nm)$. The primary bottleneck in classical computation arises in post-processing the measured data to calculate the post-selected output states. Since the number of basis states grows exponentially with the number of qubits, the classical computation overhead for each qubit subsetting check is $O(C'^s)$, where C' is the time for processing the measurement results of all the circuit copies for a single-qubit QSPC. Since we limit the subset size, both the classical memory and computation overhead is $O(C'^s nm)$, which scales linearly with the number of qubits and the number of layers. Due to the data dependency across layers, classical computations for the same check on the same qubit ($O(C'^s m)$) are sequential. The computations for different qubits $O(n)$ can be performed in parallel.

VI. EXPERIMENTAL METHODOLOGY

We evaluate our proposed QuTracer on noisy simulators and real quantum devices.

Benchmarks: The benchmarks in our experiments include Quantum Phase Estimation (QPE) [24], Bernstein-Vazirani algorithm, the QFT adder [15], the QFT multiplier [39], Variational Quantum Eigensolver (VQE) [23], and Quantum Approximate Optimization Algorithm (QAOA) [17].

Implementation: We implemented our QuTracer framework on top of the quantum computing framework Qiskit [38].

Devices: We perform our noise simulation experiments on the Qiskit noisy simulator with noise models that incorporate single- and two-qubit gate errors and measurement errors. The real device experiments are conducted on the 27-qubit quantum device `ibm_hanoi`, 127-qubit quantum device `ibm_kyoto`, and 127-qubit quantum device `ibm_cusco`.

Evaluation metrics and setup: We use Hellinger Fidelity [33] as the evaluation metrics. Hellinger Fidelity serves as a measure of similarity between two probability distributions. We use it to evaluate the closeness between the noisy distributions to the ideal (i.e., noise-free) probability distributions.

We map and optimize the circuits with the maximum optimization level in Qiskit. Since the circuit transpilation is a stochastic process, we transpiled a circuit fifty times and selected the design with the least number of CNOTs. The number of shots for the original circuits running on the real device is set to 100000. The experiments for the same

benchmark are executed in the same calibration cycle to avoid unexpected changes in hardware properties.

Comparison with JigSaw and SQEM: To evaluate the effectiveness of QuTracer, we compared it with the JigSaw and SQEM approach. JigSaw runs half of the shots in the global mode where all the qubits are measured, and the other half in the subset mode where only a subset of qubits are measured. Consistent with recommendations in prior measurement subsetting works [11], [13], the subset size in JigSaw is configured to 2.

VII. EVALUATION

A. Measurement Error Mitigation

In this experiment, we simulate a 15-qubit VQE circuit on Qiskit noisy simulator to study the measurement error mitigation effect of different error mitigation approaches. The 15-qubit VQE circuit has a similar circuit structure as shown in Fig. 6(a), but with one layer of linear entanglement CZ gates. The subsetting size of QuTracer and SQEM approach is set to 1. We use a depolarization noise model in which each 1- and 2-qubit gate has depolarizing noise with a certain probability. The single- and two-qubit gate errors are fixed at 0.001 and 0.01, respectively. The noise model also incorporates uniform single-qubit measurement errors. To study the effectiveness of the Pauli check circuits, we also simulate the circuits with the ideal Pauli checks (ideal PCS), i.e., the Pauli checks are implemented with no measurement errors on the ancilla qubits and no gate errors on the checking circuits. We use the Pauli Z checks shown in Fig. 3(b). Fig. 7 shows the output fidelities of different approaches as we vary the measurement error from 0.01 to 0.16. Since the noise model does not incorporate measurement crosstalk, the fidelity obtained from Jigsaw is similar to the fidelity of the original circuits. The circuits with ideal Pauli checks mitigate the gate errors and have shown improved fidelities over the original circuits. However, as we increase the measurement error, the measurement error becomes dominant and the fidelity improvement with ideal Pauli check circuits becomes less significant. Since SQEM and QuTracer both implement Pauli check circuits virtually, they are capable of mitigating the measurement noise. As the measurement error increases, both approaches mitigate a large proportion of the measurement error and lead to significant fidelity improvements. QuTracer achieves a higher fidelity than SQEM due to its optimizations and the reduced number of basis measurements. Based on the experiments, we evidently show that the “virtual” implementation of the Pauli checks enables the mitigation of measurement errors.

B. Gate Error Mitigation

In this experiment, we conduct simulations of an 8-qubit VQE circuit using Qiskit noisy simulator to study the gate error mitigation effect of different error mitigation approaches. The subsetting size of QuTracer and SQEM approach is set to 1. We use a depolarization noise model with a single-qubit gate error of 0.001, a two-qubit gate error of 0.01, and a single-qubit measurement error of 0.001 for all qubits. In order to

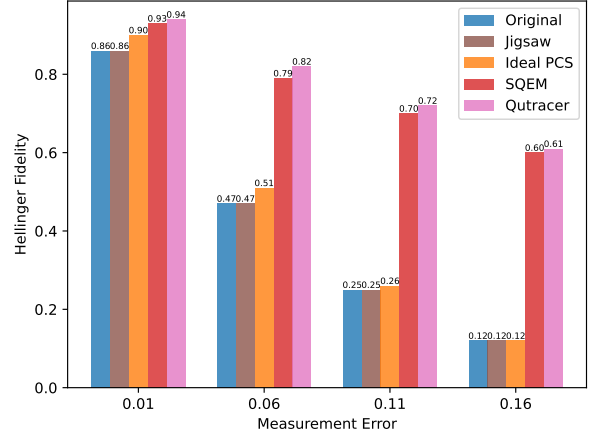


Fig. 7: Change in Hellinger fidelity with respect to the measurement error.

assess the influence of gate errors, we vary the CNOT depth in the circuit by changing the repetition times of the linear entanglement gate layer, ranging from 1 to 25. Fig. 8 shows the Hellinger fidelity under different error mitigation methods. JigSaw shows the same fidelity as the original circuit, as it does not mitigate gate errors. In comparison, both SQEM and QuTracer exhibit fidelity improvement. With increasing circuit depth, the fidelity gap between SQEM and QuTracer widens. This discrepancy arises because higher circuit depths introduce more gate errors. SQEM necessitates a state reconstruction process involving the preparation and measurement on all bases using the original circuit. As more gate noise is introduced, the accuracy of the state reconstruction process diminishes. On the other hand, QuTracer utilizes the optimization of False Dependency Removal to eliminate unnecessary gates, allowing it to outperform SQEM. A gate count comparison in Section VII-E confirms this distinction.

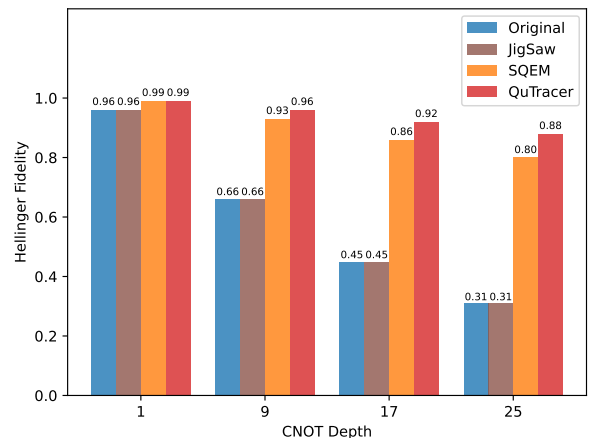


Fig. 8: Changes in Hellinger fidelity with respect to the CNOT depth.

C. Multi-layer Qubit Subsetting

In this experiment, we use a 10-qubit QAOA circuit on MaxCut problems with four layers to study the effectiveness

of multi-layer qubit subsetting in a realistic noise model. The subsetting size of QuTracer is set to 2. The noise model and coupling map are from the quantum device `ibmq_mumbai`. This noise model incorporates various factors, including gate errors, gate time, T_1 and T_2 relaxation times, and readout errors for each qubit. The parameter for this noise model is based on the calibration data collected on February 22, 2024. The median CNOT error is 7.611×10^{-3} , the median gate time is 426.667 ns, the median readout error is 1.810×10^{-2} , the median T_1 is 125.94 μ s, and the median T_2 is 188.75 μ s.

The results are shown in Fig. 9. We observe that as the number of checked layers increases, the fidelity improvement from QuTracer becomes more significant. Checking only the fourth layer improves the fidelity by 3.96%, checking both the third and the fourth layer improves the fidelity by 5.74%, checking the second, the third, and the fourth layer improves the fidelity by 7.68%, and checking all layers improves the fidelity by 9.42%, showing that sequentially checking multiple layers is effective in error mitigation.

We also compare with ideal Pauli checks (ideal PCS), i.e., the Pauli checks are implemented with no measurement errors on the ancilla qubits and no gate errors on the checking circuit. Interestingly, QuTracer surpasses the performance of the ideal PCS. This is because when QuTracer checks multiple layers, it can take advantage of utilizing the optimization of False Dependency Removal to remove unnecessary gates in the circuits for each of the layers, while PCS checks multiple layers simultaneously and can not take advantage of the optimizations for each of the layers, which results in a larger circuit with higher noise.

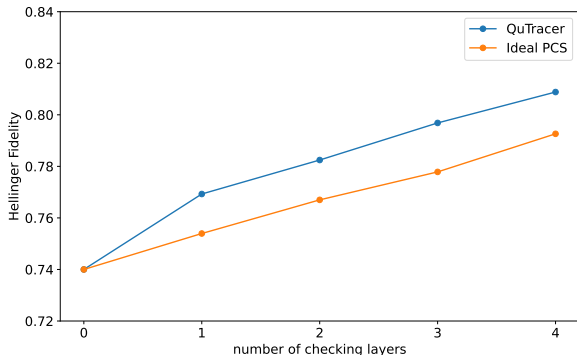


Fig. 9: Changes in Hellinger fidelity with respect to the number of checking layers.

D. Scaling of Circuit Depth

In this subsection, we use 10-qubit QAOA circuits on MaxCut problems with layers ranging from 1 to 5 to evaluate QuTracer’s efficiency in handling circuits of varying depths in a realistic noise model. The detailed parameters of the noise model are discussed in Section VII-C. The subsetting size of QuTracer is set to 2. The results are shown in Table I.

For the QAOA circuits, QuTracer interprets each circuit step as a distinct layer, applying Multi-layer Qubit Subsetting to check each of the layers. In examining individual layers, the

necessity to check multiple pairs of qubits can be reduced by the nature of the problem addressed by QAOA. Specifically, when the application of QAOA is to solve MaxCut problems on regular graphs, the QAOA circuits exhibit symmetric properties. As a result, checking one pair of qubits can yield results for multiple qubit pairs located in symmetric positions.

Results presented in Table I indicate that as circuit depth increases, the Hellinger Fidelity of the original circuits, prior to the application of any error mitigation strategies, diminishes due to the introduction of additional noisy layers. However, the efficacy of QuTracer in enhancing Hellinger Fidelity becomes more significant with deeper circuits, Table I includes the fidelity improvement of QuTracer over the unmitigated results, showing QuTracer’s capability to effectively mitigate errors in deeper circuits.

We also compared with JigSaw, which shows little improvement in Hellinger Fidelity. This is due to the Qiskit device noise model, which does not account for cross-talk noise, thereby impacting JigSaw’s performance.

E. Real Device Experiments

The experimental results on the real device are shown in Tables II and III. Applying SQEM to the benchmarks QFTMultiplier, QPE, QFTAdder, and QAOA is not practical as it incurs an exponential overhead. Therefore, we mark parts of the results in SQEM as N/A. The experiments on the same benchmark are executed in the same calibration cycle, while those for different benchmarks may be performed in different cycles with different hardware properties.

We initiate our discussion by focusing on the benchmarks that require only a single-layer QSPC in QuTracer. As shown in Table II, the benchmarks include QFTMultiplier, QPE, QFTAdder, VQE and QAOA with a single layer of CZ gates. QFTMultiplier, QPE, QFTAdder, and VQE are running on the 27 qubit machine `ibm_hanoi`, the subsetting size of QuTracer is set to 1; QAOA is running on 127 qubit machine `ibm_kyoto`, the subsetting size of QuTracer is set to 2. The results with QuTracer show the highest output fidelity with an average of $2.3\times$, $2.03\times$ and $2.15\times$ improvement over the unmitigated results, the results with JigSaw, and SQEM. In the VQE example, when increasing the number of qubits from 12 to 15, JigSaw’s results exhibit a noticeable decrease from 0.76 to 0.50. As shown in the table, the average CNOT counts for the QuTracer framework are significantly smaller than other approaches, which leads to improved output fidelity.

Table III shows the results of different schemes on the VQE algorithm and QAOA with multiple layers of CZ gates. VQE benchmark is running on 27 qubit machine `ibm_hanoi`, the subsetting size of QuTracer is set to 1; QAOA benchmark is running on a 127-qubit machine `ibm_cusco`, the subsetting size of QuTracer is set to 2. The SQEM framework is not included as its overhead scales exponentially with the number of layers. QuTracer improves the output fidelity by up to $9\times$ ($3.06\times$ on average) compared to the original circuit and by up to $9\times$ ($2.43\times$ on average) compared to JigSaw. As we increase the number of layers in the circuit, gate errors begin to play

TABLE I: Simulation results for QAOA circuits with different layers.

Workload	Normalized number of shots			Average 2-qubit basis gate count			Hellinger Fidelity			Fidelity Improvement
	Original	JigSaw	QuTracer	Original	JigSaw	QuTracer	Original	JigSaw	QuTracer	QuTracer
10-q QAOA with 1 layers	1	1	16	26	26	6	0.90	0.90	0.92	2.89%
10-q QAOA with 2 layers	1	1	106	52	52	21	0.80	0.80	0.83	3.58%
10-q QAOA with 3 layers	1	1	196	78	78	29	0.78	0.79	0.84	8.41%
10-q QAOA with 4 layers	1	1	286	104	104	37	0.74	0.74	0.81	9.42%
10-q QAOA with 5 layers	1	1	376	130	130	47	0.59	0.60	0.70	18.09%

TABLE II: Real device results for single-layer circuits.

Workload	Normalized number of shots				Average 2-qubit basis gate count				Hellinger Fidelity			
	Original	JigSaw	SQEM	QuTracer	Original	JigSaw	SQEM	QuTracer	Original	JigSaw	SQEM	QuTracer
4-q QFTMultiplier	1	1	N/A	11	28	28	N/A	18	0.49	0.49	N/A	0.65
5-q QPE	1	1	N/A	11	29	29	N/A	11	0.20	0.20	N/A	0.49
6-q QPE	1	1	N/A	11	44	44	N/A	17	0.19	0.19	N/A	0.29
7-q QFTAdder	1	1	N/A	15	75	75	N/A	37	0.22	0.22	N/A	0.35
9-q BV	1	1	13	11	21	21	21	2	0.07	0.09	0.13	0.89
12-q VQE with 1 layer	1	1	13	11	11	11	11	2	0.67	0.76	0.88	0.96
15-q VQE with 1 layer	1	1	13	11	14	14	14	2	0.36	0.50	0.65	0.87
10-q QAOA with 1 layer	1	1	N/A	16	26	26	N/A	6	0.57	0.57	N/A	0.86

TABLE III: Real device results for circuits with multiple layers.

Workload	Normalized number of shots			Average 2-qubit basis gate count			Hellinger Fidelity		
	Original	JigSaw	QuTracer	Original	JigSaw	QuTracer	Original	JigSaw	QuTracer
12-q VQE with 2 layers	1	1	29	22	22	7	0.37	0.52	0.65
12-q VQE with 3 layers	1	1	47	33	33	10	0.29	0.39	0.49
15-q VQE with 2 layers	1	1	29	28	28	7	0.21	0.28	0.69
15-q VQE with 3 layers	1	1	47	42	42	11	0.06	0.06	0.54
10-q QAOA with 2 layers	1	1	106	52	52	21	0.16	0.28	0.36
10-q QAOA with 3 layers	1	1	196	78	78	29	0.14	0.16	0.40

a more significant role. The superior fidelity improvement of QuTracer comes from its ability to effectively mitigate the increased gate errors.

VIII. DISCUSSION

A. Integration with Other Error Mitigation Techniques

Our proposed QuTracer framework is complementary with other error mitigation techniques. QuTracer generates multiple copies of the original circuit with fewer gates and fewer qubits. Existing error mitigation techniques such as dynamical decoupling [12], zero-noise extrapolation [18], Clifford data regression [10], and probabilistic error cancellation [6] can be applied to improve the fidelity of circuit copies on hardware. Moreover, the reduced size of the circuit copies enables a better scaling of mitigation approaches. For example, Clifford data regression requires fewer training data for smaller circuits. The study of integrating QuTracer with other error mitigation techniques is left for future work.

B. Application-tailored Extension of QuTracer

Prior work [11] presents a measurement subsetting framework, Varsaw, that is designed to mitigate measurement noise in variational quantum algorithms (VQA). Compared to Jigsaw, the Varsaw framework reduces the computational cost by identifying spatial redundancy across subsets from different VQA circuits and temporal redundancy across global distributions from different VQA iterations. The optimizations in Varsaw can be seamlessly integrated into the QuTracer framework to produce an application-specific extension that reduces computational overhead.

IX. CONCLUSION

In this paper, we propose a qubit subsetting framework, QuTracer, to mitigate both gate and measurement errors. The key idea is to track a subset of qubits to mitigate the error along the computation process so as to achieve high-fidelity local distributions, which are then used to refine the global distribution. Our experimental results on noisy simulators and real quantum devices show that the proposed framework significantly outperforms current state-of-the-art approaches.

ACKNOWLEDGEMENTS

We thank the anonymous reviewers for their valuable comments. The work is funded in part by NSF grants 1818914, 2325080 (with a subcontract to NC State University from Duke University), NSF grant 2120757 (with a subcontract to NC State University from the University of Maryland), and the U.S. Department of Energy, Office of Science, National Quantum Information Science Research Centers.

REFERENCES

- [1] D. Aharonov and M. Ben-Or, "Fault-tolerant quantum computation with constant error rate," *SIAM Journal on Computing*, vol. 38, no. 4, pp. 1207–1282, 2008. [Online]. Available: <https://doi.org/10.1137/S0097539799359385>
- [2] D. Amaro, C. Modica, M. Rosenkranz, M. Fiorentini, M. Benedetti, and M. Lubasch, "Filtering variational quantum algorithms for combinatorial optimization," *Quantum Science and Technology*, vol. 7, no. 1, p. 015021, 2022.
- [3] T. Ayril, F.-M. Le Régent, Z. Saleem, Y. Alexeev, and M. Suchara, "Quantum divide and compute: Hardware demonstrations and noisy simulations," in *2020 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2020, pp. 138–140.

- [4] T. Ayril, F.-M. L. Régent, Z. Saleem, Y. Alexeev, and M. Suchara, “Quantum divide and compute: Exploring the effect of different noise sources,” *SN Computer Science*, vol. 2, no. 3, p. 132, Mar 2021. [Online]. Available: <https://doi.org/10.1007/s42979-021-00508-9>
- [5] M. Benedetti, M. Fiorentini, and M. Lubasch, “Hardware-efficient variational quantum algorithms for time evolution,” *Physical Review Research*, vol. 3, no. 3, p. 033083, 2021.
- [6] E. v. d. Berg, Z. K. Mineev, A. Kandala, and K. Temme, “Probabilistic error cancellation with sparse pauli-lindblad models on noisy quantum processors,” *arXiv preprint arXiv:2201.09866*, 2022.
- [7] X. Bonet-Monroig, R. Sagastizabal, M. Singh, and T. E. O’Brien, “Low-cost error mitigation by symmetry verification,” *Phys. Rev. A*, vol. 98, p. 062339, Dec 2018. [Online]. Available: <https://link.aps.org/doi/10.1103/PhysRevA.98.062339>
- [8] H. Buhman, N. Chandran, S. Fehr, R. Gelles, V. Goyal, R. Ostrovsky, and C. Schaffner, “Position-based quantum cryptography: Impossibility and constructions,” *SIAM Journal on Computing*, vol. 43, no. 1, pp. 150–178, 2014. [Online]. Available: <https://doi.org/10.1137/130913687>
- [9] Z. Cai, “Quantum Error Mitigation using Symmetry Expansion,” *Quantum*, vol. 5, p. 548, sep 2021. [Online]. Available: <https://doi.org/10.22331/q-2021-09-21-548>
- [10] P. Czarnik, A. Arrasmith, P. J. Coles, and L. Cincio, “Error mitigation with Clifford quantum-circuit data,” *Quantum*, vol. 5, p. 592, Nov. 2021. [Online]. Available: <https://doi.org/10.22331/q-2021-11-26-592>
- [11] S. Dangwal, G. S. Ravi, P. Das, K. N. Smith, J. M. Baker, and F. T. Chong, “Varsaw: Application-tailored measurement error mitigation for variational quantum algorithms,” 2023.
- [12] P. Das, S. Tannu, S. Dangwal, and M. Qureshi, “Adapt: Mitigating idling errors in qubits via adaptive dynamical decoupling,” in *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, 2021, pp. 950–962.
- [13] P. Das, S. Tannu, and M. Qureshi, “Jigsaw: Boosting fidelity of nisq programs via measurement subsetting,” in *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO ’21. New York, NY, USA: Association for Computing Machinery, 2021, p. 937–949. [Online]. Available: <https://doi.org/10.1145/3466752.3480044>
- [14] D. M. Debrov and K. R. Brown, “Extended flag gadgets for low-overhead circuit verification,” *Phys. Rev. A*, vol. 102, p. 052409, Nov 2020. [Online]. Available: <https://link.aps.org/doi/10.1103/PhysRevA.102.052409>
- [15] T. G. Draper, “Addition on a quantum computer,” *arXiv preprint quant-ph/0008033*, 2000.
- [16] E. Farhi, J. Goldstone, and S. Gutmann, “A quantum approximate optimization algorithm,” *arXiv preprint arXiv:1411.4028*, 2014.
- [17] E. Farhi, J. Goldstone, and S. Gutmann, “A quantum approximate optimization algorithm,” *arXiv preprint arXiv:1411.4028*, 2014.
- [18] T. Giurgica-Tiron, Y. Hindy, R. LaRose, A. Mari, and W. J. Zeng, “Digital zero noise extrapolation for quantum error mitigation,” in *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)*, 2020, pp. 306–316.
- [19] A. Gonzales, R. Shaydulin, Z. H. Saleem, and M. Suchara, “Quantum error mitigation by pauli check sandwiching,” *Scientific Reports*, vol. 13, no. 1, p. 2122, 2023.
- [20] H. Harada, K. Wada, and N. Yamamoto, “Doubly optimal parallel wire cutting without ancilla qubits,” *arXiv preprint arXiv:2303.07340*, 2023.
- [21] A. W. Harrow, A. Hassidim, and S. Lloyd, “Quantum algorithm for linear systems of equations,” *Physical review letters*, vol. 103, no. 15, p. 150502, 2009.
- [22] 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, “Virtual distillation for quantum error mitigation,” *Physical Review X*, vol. 11, no. 4, p. 041036, 2021.
- [23] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, J. M. Gambetta *et al.*, “Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets,” *Nature*, vol. 549, no. 7671, pp. 242–246, 2017.
- [24] A. Y. Kitaev, “Quantum measurements and the abelian stabilizer problem,” *arXiv preprint quant-ph/9511026*, 1995.
- [25] A. Y. Kitaev, “Quantum computations: algorithms and error correction,” *Russian Mathematical Surveys*, vol. 52, no. 6, p. 1191, dec 1997. [Online]. Available: <https://dx.doi.org/10.1070/RM1997v052n06ABEH002155>
- [26] B. Koczor, “Exponential error suppression for near-term quantum devices,” *Phys. Rev. X*, vol. 11, p. 031057, Sep 2021. [Online]. Available: <https://link.aps.org/doi/10.1103/PhysRevX.11.031057>
- [27] B. P. Lanyon, J. D. Whitfield, G. G. Gillett, M. E. Goggin, M. P. Almeida, I. Kassal, J. D. Biamonte, M. Mohseni, B. J. Powell, M. Barbieri *et al.*, “Towards quantum chemistry on a quantum computer,” *Nature chemistry*, vol. 2, no. 2, pp. 106–111, 2010.
- [28] J. Liu, A. Gonzales, and Z. H. Saleem, “Classical simulators as quantum error mitigators via circuit cutting,” 2022.
- [29] S. McArdle, X. Yuan, and S. Benjamin, “Error-mitigated digital quantum simulation,” *Phys. Rev. Lett.*, vol. 122, p. 180501, May 2019. [Online]. Available: <https://link.aps.org/doi/10.1103/PhysRevLett.122.180501>
- [30] J. R. McClean, Z. Jiang, N. C. Rubin, R. Babbush, and H. Neven, “Decoding quantum errors with subspace expansions,” *Nature Communications*, vol. 11, no. 1, p. 636, Jan 2020. [Online]. Available: <https://doi.org/10.1038/s41467-020-14341-w>
- [31] P. D. Nation and M. Treinish, “Suppressing quantum circuit errors due to system variability,” *PRX Quantum*, vol. 4, no. 1, p. 010327, 2023.
- [32] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge University Press, 2011. [Online]. Available: <https://doi.org/10.1017/CBO9780511976667>
- [33] M. S. Nikulin *et al.*, “Hellinger distance,” *Encyclopedia of mathematics*, vol. 78, 2001.
- [34] T. Peng, A. W. Harrow, M. Ozols, and X. Wu, “Simulating large quantum circuits on a small quantum computer,” *Phys. Rev. Lett.*, vol. 125, p. 150504, Oct 2020. [Online]. Available: <https://link.aps.org/doi/10.1103/PhysRevLett.125.150504>
- [35] M. A. Perlin, Z. H. Saleem, M. Suchara, and J. C. Osborn, “Quantum circuit cutting with maximum-likelihood tomography,” *npj Quantum Information*, vol. 7, no. 1, p. 64, Apr 2021. [Online]. Available: <https://doi.org/10.1038/s41534-021-00390-6>
- [36] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O’Brien, “A variational eigenvalue solver on a photonic quantum processor,” *Nature communications*, vol. 5, no. 1, p. 4213, 2014.
- [37] J. Preskill, “Quantum computing in the nisq era and beyond,” *Quantum*, vol. 2, p. 79, 2018.
- [38] Qiskit contributors, “Qiskit: An open-source framework for quantum computing,” 2023.
- [39] L. Ruiz-Perez and J. C. Garcia-Escartin, “Quantum arithmetic with the quantum fourier transform,” *Quantum Information Processing*, vol. 16, pp. 1–14, 2017.
- [40] M. Schuld, I. Sinayskiy, and F. Petruccione, “An introduction to quantum machine learning,” *Contemporary Physics*, vol. 56, no. 2, pp. 172–185, 2015.
- [41] P. W. Shor, “Algorithms for quantum computation: discrete logarithms and factoring,” in *Proceedings 35th annual symposium on foundations of computer science*. Ieee, 1994, pp. 124–134.
- [42] K. Temme, S. Bravyi, and J. M. Gambetta, “Error mitigation for short-depth quantum circuits,” *Physical review letters*, vol. 119, no. 18, p. 180509, 2017.
- [43] K. Tsubouchi, Y. Suzuki, Y. Tokunaga, N. Yoshioka, and S. Endo, “Virtual quantum error detection,” 2023.
- [44] E. van den Berg, S. Bravyi, J. M. Gambetta, P. Jurcevic, D. Maslov, and K. Temme, “Single-shot error mitigation by coherent pauli checks,” *Physical Review Research*, vol. 5, no. 3, p. 033193, 2023.

A. Abstract

Our artifact provides the source code for QuTracer, and the Python scripts to run benchmarks on simulators and real quantum machines. These scripts also generate results for the evaluation part. Users can reproduce the results in Table I, II and III.

B. Artifact check-list (meta-information)

- **Compilation:** Qiskit transpiler
- **Data set:** Benchmarks listed in Section VI
- **Run-time environment:** Ubuntu 20.04.6 LTS
- **Hardware:** The experiments on real quantum machines require access to IBM quantum machines
- **Execution:** Run the bash scripts and python scripts
- **Metrics:** Hellinger fidelity, 2-qubit basis gate count, normalized number of shots
- **Output:** JSON files and CSV files
- **Experiments:** Apply different Error Mitigation methods to quantum circuits and compare the normalized number of shots, 2-qubit basis gate count, and Hellinger fidelity.
- **How much disk space required (approximately)?:** 2GB
- **How much time is needed to prepare workflow (approximately)?:** 1 hour
- **How much time is needed to complete experiments (approximately)?:** 12+ hours
- **Publicly available?:** Yes
- **Code licenses (if publicly available)?:** Apache-2.0 License
- **Archived (provide DOI)?:** 10.5281/zenodo.11075556

C. Description

1) *How to access:* The source code for QuTracer and the scripts to run the benchmarks are available in github: https://github.com/peiyi1/QuTracer_project

2) *Hardware dependencies:* The access to IBM quantum machines is needed to reproduce the results in Table II and III.

3) *Software dependencies:* Python 3.10 is used in our experiments.

4) *Data sets:* Benchmarks are listed in Section VI

D. Installation

1) *Create conda environment:* Download Anaconda at <https://www.anaconda.com/> and install it.

Create an environment named qutracer:
`$ conda create -y -n qutracer python=3.10`

Activate the environment:
`$ conda activate qutracer`

2) *Install Qiskit and other necessary packages:* The version of Qiskit used in our experiment is 0.45.1.

```
$ pip install qiskit==0.45.1
$ pip install qiskit-ibm-provider==0.7.3
$ pip install qiskit-aer==0.13.2
$ pip install networkx==0.13.2
$ pip install networkx==3.2.1
$ pip install matplotlib
```

```
$ pip install pylatexenc
$ pip install pyyaml
```

3) *QuTracer package installation:* Clone the repository of QuTracer:

```
$ git clone https://github.com/peiyi1/QuTracer_project.git
```

Go to the path `/QuTracer_project/QuTracer` and install QuTracer:

```
$ cd QuTracer_project/QuTracer
$ pip install .
```

E. Experiment workflow

1) *experiments on Qiskit AerSimulator:* In the path `/QuTracer_project/test_on_simulator`, there are three directories: `script`, `yaml_file`, and `saved_data`. The directory `script` contains all the scripts that can generate circuits, execute circuits, and process the results for the circuits in different error mitigation methods. The directory `yaml_file` contains all the configuration files for different benchmarks, users can modify the configuration file to set up different configurations for running the benchmark. The directory `saved_data` contains all the saved circuits and results in JSON files. To reproduce the results in the directory `saved_data`, run the following command.

Go to the path `/QuTracer_project/test_on_simulator`:
`$ cd QuTracer_project/test_on_simulator/`

Generate all the circuits needed in different error mitigation methods:

```
$ ./generate_circuits.sh
```

Execute all the circuits generated in the previous step:

```
$ ./execute_circuits.sh
```

Run the following command to process the results from all the circuits and calculate the Hellinger fidelity for different error mitigation methods. All the results are saved in the directory `saved_data`. For example, the results for 10-qubit QAOA with 1 layer are saved in the path `/QuTracer_project/test_on_simulator/saved_data/qaoa_reps1_n10/results`

```
$ ./results_postprocessing.sh
```

2) *experiments on IBM quantum machine:* In the path `/QuTracer_project/test_on_real_machine`, there are three directories: `script`, `yaml_file`, and `saved_data`. For all the YAML files in the directory `yaml_file`, the configuration option `'enable_running_on_real_machine'` is set to `'True'` to run all the circuits on real IBM quantum machines. To reproduce the results in the directory `saved_data`, run the following command.

Go to the path `/QuTracer_project/test_on_real_machine`:
`$ cd QuTracer_project/test_on_real_machine/`

Generate all the circuits needed in different error mitigation methods:

```
$ ./generate_circuits.sh
```

Execute all the circuits generated in the previous step:

```
$ ./execute_circuits.sh
```

After running the above command, all the circuits have been sent to the job queue of IBM quantum machines, the job status can be checked on the IBM Quantum Platform website: <https://quantum.ibm.com/jobs>. Wait until all the job statuses become completed, then run the following command to load and save all the job results:

```
$ ./save_results_real_machine.sh
```

Process the results from all the circuits and calculate the Hellinger fidelity for different error mitigation methods:

```
$ ./results_postprocessing.sh
```

F. Evaluation and expected results

After going through all the steps in the Experiment Workflow section, go back to the path /QuTracer_project and run following commands to generate CSV files, which correspond to the results in Table I, II and III.

Generate the file simulation_results_for_qaoa.csv, which contains results in Table I:

```
$ python generate_table_simulation_results_for_qaoa.py
```

Generate the file real_machine_results_for_single_layer_circuits.csv, which contains results in Table II:

```
$ python generate_table_real_machine_results_for_single_layer_circuits.py
```

Generate the file real_machine_results_for_circuits_with_multiple_layers.csv, which contains results in Table III:

```
$ python generate_table_real_machine_results_for_circuits_with_multiple_layers.py
```

G. Experiment customization

The configuration in the YAML files can be modified to run benchmarks with different configurations.

H. Notes

The Experiment Workflow subsection E2 requires access to IBM quantum machines, and the waiting time for running circuits on IBM quantum machines may vary based on the size of the waiting queue.

I. Methodology

Submission, reviewing and badging methodology:

- <https://www.acm.org/publications/policies/artifact-review-and-badging-current>
- <http://cTuning.org/ae/submission-20201122.html>
- <http://cTuning.org/ae/reviewing-20201122.html>