Testing platform-independent quantum error mitigation on noisy quantum computers

Vincent Russo,[1,*] Andrea Mari,[1] Nathan Shammah,[1] Ryan LaRose,[1,2] William J. Zeng[1,3]

[1] *Unitary Fund*

[2] *Institute of Physics, École Polytechnique Fédérale de Lausanne (EPFL), CH-1015 Lausanne, Switzerland*

[3] *Goldman Sachs & Co., New York, NY*

We apply quantum error mitigation techniques to a variety of benchmark problems and quantum computers to evaluate the performance of quantum error mitigation in practice. To do so, we define an empirically motivated, resource-normalized metric of the improvement of error mitigation which we call the improvement factor, and calculate this metric for each experiment we perform. The experiments we perform consist of zero-noise extrapolation and probabilistic error cancellation applied to two benchmark problems run on IBM, IonQ, and Rigetti quantum computers, as well as noisy quantum computer simulators. Our results show that error mitigation is on average more beneficial than no error mitigation — even when normalized by the additional resources used — but also emphasize that the performance of quantum error mitigation depends on the underlying computer.

I. INTRODUCTION

Quantum computers have steadily improved over the past two decades as can be seen in component metrics like T1 and T2 times [HWFZ20, SR20] as well as full system metrics like quantum volume [CBS+19]. While we expect these hardware improvements to continue, it is generally accepted that error rates cannot be made low enough purely by hardware improvements. Rather, to achieve error rates low enough for useful applications, hardware improvements should be coupled with algorithmic or software methods.

The most commonly pursued algorithmic method is quantum error correction [Sho95, CS96, Ste96], which generally provides a tradeoff in qubit quantity for qubit quality — i.e., using more qubits to achieve a lower logical error rate. Today, the state-of-the-art experiments in quantum error correction [AI21, AI22] confirm an exponential suppression of errors as the code distance increases but do so for relatively small code distances. For example, the largest surface code implementation to our knowledge [AI22] uses 49 physical qubits to encode one logical qubit in a distance five surface code, while rough estimates for current error rates require around 1000 physical qubits per logical qubit for fault tolerance.

Because the experimental requirements of quantum error correction are very demanding, and because of widespread interest in applications of noisy quantum computers [Pre18], a new set of algorithmic methods to deal with errors has emerged in recent years. These new methods are referred to as quantum error mitigation [ECBY21, CBB+22] and are designed to be less experimentally demanding than full quantum error correction. However, this comes at the cost of being less general and more heuristic than quantum error correction.

While a relatively large number of error mitigation techniques have been proposed [SCW+19, Vui17, KTC+19, GTHL+20, UNdJ20, QCA+20, ZLZ+20], there have been relatively few experiments using error mitigation, despite the fact that error mitigation is specifically designed for current quantum computers. A summary from the literature of quantum error mitigation experiments performed on quantum computers is shown in Table I. Note that this table is not exhaustive for all benchmarks outside of the context of quantum error mitigation. For instance, this review [SL22] provides a thorough collection of T1 and T2 times for the dynamical decoupling benchmark.

In this work, we evaluate quantum error mitigation in practice using a suite of experiments on various benchmarks and quantum computers. We consider two error mitigation techniques, two benchmark problems, and four quantum computers. To quantify the performance of quantum error mitigation (relative to no error mitigation), we define a natural metric that we call the *improvement factor*.

Our results show that quantum error mitigation improves the performance of noisy quantum computations in nearly all experiments we consider, even when normalized by the additional resources (namely, samples) used in the error mitigation techniques. Depending on the number of qubits, circuit depth, and particular computer in the experiment, our results show between a 1x and 7x improvement from quantum error mitigation. Further, the error mitigation we use is "out-of-the-box" in that it is not tailored to the benchmark problems or computers we consider. Because of this, we expect quantum error mitigation to be an essential component of NISQ and even error-corrected computations and offer perspective on these points.

The rest of the paper is organized as follows. Section II describes our methods for assessing the performance of quantum error mitigation in practice. This includes our definition of the improvement factor (Sec. II A), the error mitigation techniques (Sec. II B), the benchmark problems (Sec. II C), and the quantum computers (Sec. II D) used in our experiments. We present the results of our experiments in Sec. III, and we discuss them in the larger

* Corresponding author: vincent@unitary.fund.

QEM	Benchmark	Qubits n	Computer(s)	Ref.
ZNE	RB	1, 2	5-qubit superconducting device	[KTC+19]
	RB	2	IBMQ London & Rigetti Aspen-8	[LMK+20]
	RB, PG	2 - 5	IBMQ Lagos & IBMQ Casablanca	[CDM+22]
	RB, MC	3, 5, 12	IBMQ Lima, IBMQ Kolkata, Rigetti Aspen-M2, IonQ Harmony	(This work)
	VQE	4	5-qubit superconducting device	[KTC+19]
	QV	5	IBMQ Belem, IBMQ Lima, & IBMQ Quito	[LMR+22]
	RB, TE	26	27-qubit superconducting device	[KWY+21]
PEC	RB	2	2-qubit trapped ion ($^{171}Yb^{+}$) device	[ZLZ+20]
	RB, MC	3	Rigetti Aspen-M2, IBMQ Lima, IonQ Harmony	(This work)
	TE, CYC	4, 10	27-qubit superconducting device	[BMKT22]
	CYC	4	4-qubit superconducting device	[FHV+22]
DD	IDLE	1, 2	IBMQX4, IBMQX5, & Rigetti Acorn	[PAFL18]
	Adder, GHZ, QAOA, QFT, VQE	4, 5, 6	IBMQ Guadalupe, & IBMQ Jakarta	[SRM+22]
	QPE	5	IBMQ Paris, IBMQ Guadalupe, & IBMQ Toronto	[DTDQ21]
	QV	6	IBMQ Montreal	[JJAB+21]
	QFT	6, 7	IBMQ Paris, IBMQ Guadalupe, & IBMQ Toronto	[DTDQ21]
	BV	7, 8	IBMQ Paris, IBMQ Guadalupe, & IBMQ Toronto	[DTDQ21]
	QAOA	8, 10	IBMQ Paris, IBMQ Guadalupe, & IBMQ Toronto	[DTDQ21]
CDR	RB, PG	2 - 5	IBMQ Lagos & IBMQ Casablanca	[CDM+22]
	VQE	5	IBMQ Rome	[CACC21]
	VQE	6	IBMQ Toronto	[CMSC22]
	VQE	16	IBMQ Almaden	[ZCN+21]
SSE	VQE	2	2-qubit superconducting device	[CRD+18]
	VQE	2	3-qubit superconducting device	[SBMS+19]
VD	GHZ	5	5-qubit trapped ion, UMD, ($^{171}Yb^{+}$) device	[SCZ+22]

TABLE I: A history of quantum error mitigation experiments on quantum computers in literature. Quantum error mitigation (QEM) Technique acronyms: ZNE = zero-noise extrapolation, PEC = probabilistic error cancellation, also referred to as quasi-probabilistic decomposition (QPD) by some authors, DD = dynamical decoupling, SSE = subspace expansion, VD = virtual distillation, CDR = Clifford data regression. Benchmark acronyms: RB = randomized benchmarking, VQE = variational quantum eigensolver, TE = time evolution, IDLE = allowing a state to idle (identity operation), CYC = alternate cycles of single-qubit layers and Clifford layers, GHZ = Greenberger–Horne–Zeilinger, MC = mirror circuits, Adder = Ripple Carry Adder.

context of quantum error mitigation and quantum computation in Sec. IV.

II. METHODS

To assess the experimental performance of quantum error mitigation, we define a natural measure comparing the accuracy of an experiment with quantum error mitigation to the accuracy without quantum error mitigation. This measure, which we call the *improvement factor*, is motivated and defined in Sec. II A. We experimentally calculate this measure using two quantum error mitigation techniques (Sec. II B) with two benchmark problems (Sec. II C) on four quantum computers and three noisy quantum computer simulators (Sec. II D). All the error mitigation techniques for this study were implemented using the Mitiq error mitigating compiler [LMK+20].

A. Improvement factor

The goal of most quantum error mitigation techniques is to improve the estimation of expectation values. Following an empirical approach, we quantify the improvement of error mitigation by comparing the estimation errors obtained with and without error mitigation.

Let ρ be an ideal n-qubit quantum state prepared by a noiseless quantum computer after the execution of some given quantum circuit C, i.e., $\rho = C|0^{\otimes n}\rangle\langle 0^{\otimes n}|C^{\dagger}$. For an observable $\hat{A} = \hat{A}^{\dagger}$, the ideal (noiseless) expectation value is

$$A = \mathrm{tr}[\rho\hat{A}] = \mathrm{tr}[C|0^{\otimes n}\rangle\langle 0^{\otimes n}|C^{\dagger}\hat{A}]. \quad (1)$$

When using a noisy quantum computer, we instead prepare a noisy state ρ' and collect N shots (samples) to obtain an empirical estimate A' of the expectation value.

The goal of quantum error mitigation (QEM) is to compute some quantity A_{QEM} which is a more accurate estimate of the ideal expectation value A compared to the unmitigated estimate A'[1]. Generally, computing A_{QEM} is done by executing a set of circuits $\{C_1, ..., C_{k_{\mathrm{QEM}}}\}$ related to C — usually with a different number of qubits,

[1] Note that here and throughout, we use the acronym QEM to refer to a generic quantum error mitigation technique and a specific acronym for a specific quantum error mitigation technique. So,

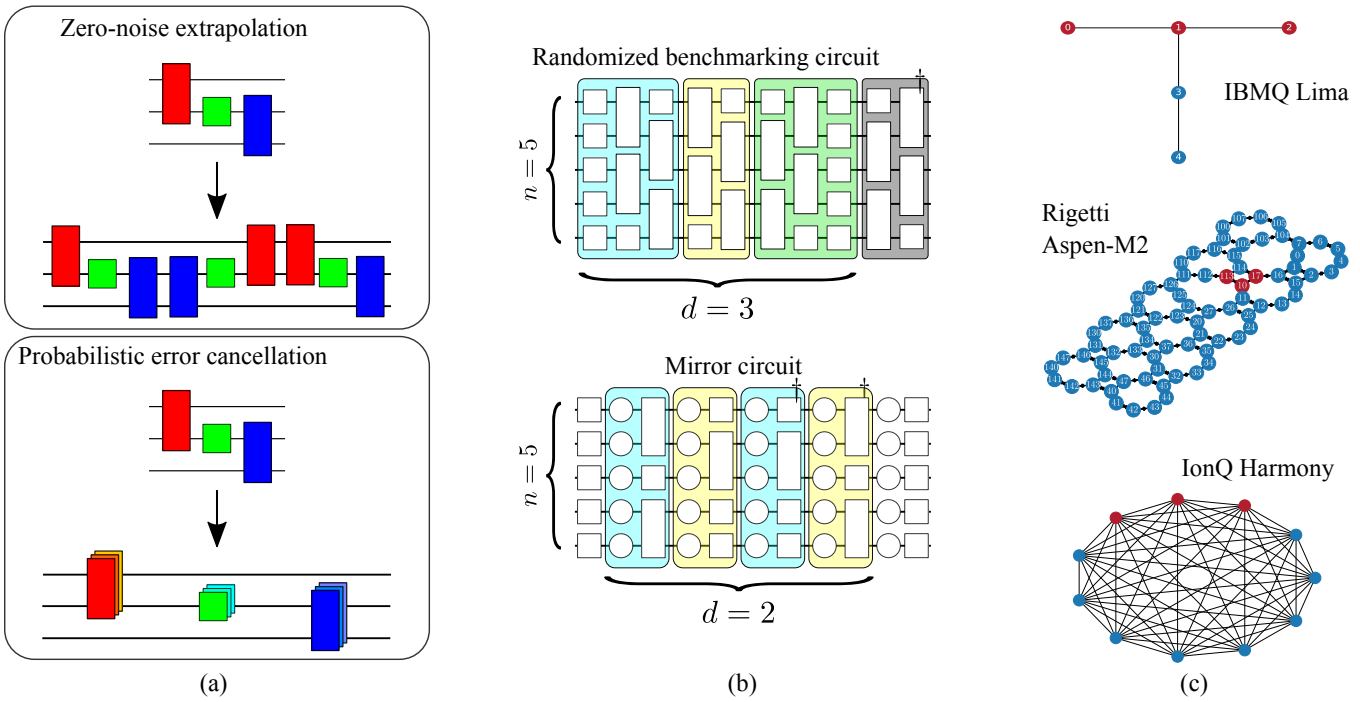

FIG. 1: An overview of our method to assess the performance of quantum error mitigation in practice. An experiment consists of **(a)** a QEM technique, **(b)** a benchmark problem, and **(c)** a quantum computer. The result of an experiment is the improvement factor (Sec. II A) μ_{QEM} defined in Eq. (5), potentially at various numbers of qubits n and/or circuit depths d. **(a)** Cartoon graphics of the two quantum error mitigation techniques we consider. In zero-noise extrapolation (Sec. II B 1), a circuit C is mapped to a set of noise-scaled circuits by adding gates that compile (without noise) to the identity. The expectation value is computed for each noise-scaled circuit, and the results are extrapolated to zero-noise with either linear or Richardson extrapolation. In probabilistic error cancellation (Sec. II B 2), each ideal (unitary) gate of a circuit is expressed in the noisy basis of the computer. A number of new circuits are sampled from this expansion and executed results are combined to produce the error-mitigated result. **(b)** The two benchmark problems we use in our experiments (Sec. II C). An n-qubit, depth d mirror circuit C is defined by a single layer of Clifford gates (white squares), d Clifford layers followed by their inverses (rectangles in colored boxes, daggers denote inverses) with intermediate random Pauli gates (circles in colored boxes), and a final layer of Pauli (white circles) and Clifford gates. This sequence produces a single bitstring $|z_C\rangle$, and we take $\hat{A} = |z_C\rangle\langle z_C|$ as the observable. An n-qubit, depth d randomized benchmarking circuit is defined by a random sequence of d elements of the n-qubit Clifford group and a final inverse such that the final state is $|0\rangle$, and we take $\hat{A} = |0\rangle\langle 0|$ as the observable. **(c)** Qubit coupling maps for the four quantum computers we perform experiments on (see Sec. II D for device characteristics and error rates). Red nodes show qubits used for $n = 3$ qubit experiments. Qubit selection was not available on IonQ Harmony and so nodes are unlabeled. We also perform experiments on noisy quantum computer simulators (Sec. II D 5).

gates, and/or total shots N_{QEM} — then post-processing the noisy results to obtain the error-mitigated estimate A_{QEM}.

We refer to an evaluation of an unmitigated expectation value A' as a trial. After performing t trials $A'_{[1]}, ..., A'_{[t]}$ we can quantify the estimation error through the root-mean-square error (RMSE)

$$\sqrt{\frac{1}{t}\sum_{i=1}^{t}\left(A'_{[i]} - A\right)^2}. \tag{2}$$

We use the RMSE since it reduces to the absolute error when all $A'_{[i]}$ are approximately equal (e.g. in the limit of large N) and, at the same time, it also takes into account the estimation error due to the statistical fluctuations of the results $A'_{[i]}$ over different trials.

Similarly, we refer to an evaluation of A_{QEM} as a QEM trial. After performing t QEM trials $A^{[1]}_{\text{QEM}}, ..., A^{[t]}_{\text{QEM}}$ we

for example, the zero-noise extrapolated expectation value of $\hat{A}$ is denoted A_{ZNE}, and similarly for other quantities. A summary of our notation is included in Appendix A.

evaluate the RMSE

$$\sqrt{\frac{1}{t}\sum_{i=1}^{t}\left(A_{\mathrm{QEM}}^{[i]}-A\right)^2}. \tag{3}$$

As noted, evaluating each $A_{\mathrm{QEM}}^{[i]}$ potentially uses additional resources in the form of circuits, qubits, gates, and/or shots. To account for these additional resources, we define the *problem-specific improvement factor*

$$\mu_{\mathrm{QEM}}(C,\hat{A}) := \frac{\sqrt{N\sum_{i=1}^{t}(A'_{[i]}-A)^2}}{\sqrt{N_{\mathrm{QEM}}\sum_{i=1}^{t}(A_{\mathrm{QEM}}^{[i]}-A)^2}}, \tag{4}$$

i.e., the shot-normalized ratio of root-mean-square errors. (See Sec. IV B for a discussion on normalizing by other resources, e.g. qubits and gates, in addition to shots.) This value is a natural, empirically defined measure of the performance of quantum error mitigation for a specific expectation value problem defined by a circuit C and observable $\hat{A}$. To generalize over different problems in addition to averaging over multiple trials, we also average over a set of circuits $\mathcal{C}$ and a set of observables $\hat{\mathcal{A}}$ to define the *improvement factor*

$$\mu_{\mathrm{QEM}} := \frac{\sqrt{N\sum_{C\in\mathcal{C},\hat{A}\in\hat{\mathcal{A}}}\sum_{i=1}^{t}(A'_{[i]}-A)^2}}{\sqrt{N_{\mathrm{QEM}}\sum_{C\in\mathcal{C},\hat{A}\in\hat{\mathcal{A}}}\sum_{i=1}^{t}(A_{\mathrm{QEM}}^{[i]}-A)^2}}. \tag{5}$$

Here, as in Eq. (4), the circuit C is implicit in the expectation values A, $A'_{[i]}$, and $A_{\mathrm{QEM}}^{[i]}$, e.g. $A = \mathrm{tr}[C|0\rangle\langle 0|C^\dagger\hat{A}]$. While this definition is general with respect to the circuits C, experimentally we consider two classes of benchmark circuits (Sec. II C) and quote the results from these classes of circuits separately. Indeed, for most experiments on quantum computers, we generate $|\mathcal{C}| = 4$ randomized instances of benchmark circuits from the two classes. We choose benchmark circuits such that there is one natural observable for each circuit, i.e. $\left|\hat{\mathcal{A}}\right| = 1$, where $|\cdot|$ denotes the cardinality of the set. In all cases, due to limited device availability, we perform $t = 1$ trial for each $C, \hat{A} \in \mathcal{C}\times\hat{\mathcal{A}}$.

We note that authors of [CDM+22] also define a measure of the improvement from error mitigation, in particular a problem-specific measure. This quantity, which they call the relative mitigation error and denote by ϵ, is given by (in the notation of this paper)

$$\epsilon_{\mathrm{QEM}}(C,\hat{A}) := \frac{|A_{\mathrm{QEM}}-A|}{|A'-A|}. \tag{6}$$

For $t = 1$, $\mu_{\mathrm{QEM}}(C,\hat{A}) = \sqrt{\frac{N}{N_{\mathrm{QEM}}}}\epsilon_{\mathrm{QEM}}^{-1}(C,\hat{A})$.

B. Quantum error mitigation techniques

1. Zero-noise extrapolation

We apply zero-noise extrapolation (ZNE) [TBG17, LB17, KTC+19] with both linear and Richardson extrapolation — which we respectively denote ZNE(L) and ZNE(R) — to our benchmark problems. For both cases, we evaluate $k_{\mathrm{ZNE}} = 3$ noisy expectation values $A'(\lambda_i)$ at different noise scale factors $\lambda_i \in \{1,2,3\}$ and the zero-noise limit is obtained as a linear combination of the results

$$A_{\mathrm{ZNE}} = \sum_{i=1}^{k_{\mathrm{ZNE}}}\eta_i A'(\lambda_i). \tag{7}$$

For Richardson extrapolation, the best fit coefficients η_i in Eq. (7) are given by [GTHL+20]

$$\eta_i := \prod_{j\neq i}\frac{\lambda_j}{\lambda_j-\lambda_i}. \tag{8}$$

For linear extrapolation, the coefficients η_i are obtained from a linear best fit and also only depend on the noise scale factors, but the analytical expression is more involved (see Eq. (26) of [GTHL+20]).

For all ZNE experiments, we use global unitary folding [GTHL+20] to scale noise. For odd integer scale factors λ_i, this amounts to replacing the circuit C by $C(C^\dagger C)^{(1-\lambda_i)/2}$. If λ_i is not an odd integer, a fraction of the full circuit is folded and appended to the circuit as described in [GTHL+20]. Each noise-scaled circuit is executed with $\lfloor N/k_{\mathrm{ZNE}}\rfloor = \lfloor 10^4/3\rfloor$ shots so that $N_{\mathrm{ZNE}} \simeq N = 10^4$ (i.e., so that we use the same total number of shots in ZNE as in the unmitigated experiment). For more details on our ZNE implementation, see Appendix B 4 a.

2. Probabilistic error cancellation

We also apply probabilistic error cancellation (PEC) [TBG17, EBL18, ZLZ+20] to each benchmark problem. Here, the first step is to characterize the set of noisy, implementable operations $\{\mathcal{O}_\alpha\}$ of a computer so that we can represent the ideal (noiseless) operations $\{\mathcal{G}_i\}$ of a circuit in this basis, namely

$$\mathcal{G}_i = \sum_\alpha \eta_{i,\alpha}\mathcal{O}_\alpha. \tag{9}$$

Note that the calligraphic symbols $\mathcal{G}_i$ and $\mathcal{O}_\alpha$ stand for super-operators acting on the quantum state of the qubits as linear quantum channels, and $\eta_{i,\alpha}\in\mathbb{R}$. In principle, this requires full tomographic knowledge of the noisy operations $\{\mathcal{O}_\alpha\}$, but we make two simplifying assumptions in our experiments:

1. We neglect errors of single-qubit gates.

2. We assume that all two-qubit gates $\mathcal{G}_{2Q}$ (CNOT or CZ in our experiments) are followed by local depolarizing noise, i.e.

$$\mathcal{G}_{2Q}^{(\text{noisy})} = (\mathcal{D}_p \otimes \mathcal{D}_p) \circ \mathcal{G}_{2Q}, \tag{10}$$

where $\mathcal{D}_p(\rho) = (1-p)\rho + \frac{p}{3}(X\rho X + Y\rho Y + Z\rho Z)$ is the single-qubit depolarizing channel and p is the local error probability.

Under these assumptions, the quasi-probability representation of the ideal $\mathcal{G}_{2Q}$ gate can be derived for any value of p [TBG17, Tak21]. For each $\mathcal{G}_{2Q}$ operation, we obtain p from the error rate in the calibration data reported by the hardware vendor for the associated two-qubit gate (IBM, Rigetti), or from the average two-qubit error rate (IonQ). More precisely, in Eq. (10) the overall two-qubit error probability is $p_{2Q} = 1-(1-p)^2$. So, given the parameter p_{2Q} reported by the calibration data of the computer, we estimate $p = 1-\sqrt{1-p_{2Q}}$ and use this in Eq. (10) to obtain the basis $\mathcal{O}_\alpha$.

After obtaining the basis $\mathcal{O}_\alpha$, we represent all two-qubit gates G_i in the circuit in this basis as in Eq. (9) and stochastically sample $k_{\text{PEC}} = 100$ new circuits to execute. Each circuit is executed with $N/k_{\text{PEC}} = 10^4/100$ shots so that $N_{\text{PEC}} = N = 10^4$ (i.e., so that we use the same total number of shots in PEC as in the unmitigated experiment). For more details on our implementation of PEC, see Appendix B 4 b.

C. Benchmark problems

A benchmark problem is defined by an n-qubit, depth d quantum circuit C, and an observable $\hat{A}$ as in Eq. (1). In this work, we consider two benchmark problems in which the circuit C produces (without noise) a single bitstring $z_C \in \{0,1\}^n$, and we always take $\hat{A} = |z_C\rangle\langle z_C|$ as the corresponding observable. Both circuits have a number of qubits n and a depth d which can be varied independently, and we use $|\mathcal{C}| = 4$ (random) instances of each circuit for a given n, d. In experiments on quantum computers, we choose $n \in \{3,5\}$ and $d \subseteq \{1,3,5,7,9\}$. Additionally, for a specific device (IBMQ Kolkata), we also perform a larger experiment with $n = 12$ qubits and $d \in \{1,5,9\}$. The number of one- and two-qubit gates for a given n, d depend on the circuit type and is discussed for each circuit type below. As discussed in Sec. II D 5 we repeat each of these experiments on noisy quantum computer simulators for comparison.

1. *Randomized benchmarking*

We use randomized benchmarking (RB) circuits [EME10, MGE12, CGC+13, GCM+12, MSS+19] as one benchmark problem in our experiments. An n-qubit, depth d RB circuit is a sequence of d random Clifford group elements $U_d U_{d-1} \cdots U_1$, followed by a (classically computed) inverse element $U_{\text{inv}} = (U_d U_{d-1} \cdots U_1)^{-1}$, such that the full circuit

$$C = U_{\text{inv}} U_d U_{d-1} \cdots U_1 \tag{11}$$

is the identity operation (without noise). As such, the only bitstring that should be measured is $z_C = 0^n$ and we take the observable to be $\hat{A} = |z_C\rangle\langle z_C| = |0\rangle\langle 0|^{\otimes n}$.

In all experiments, we use a line of qubits and apply 2-qubit RB sequences to each neighboring pair of qubits on the line. If the total number of qubits is odd we also apply a 1-qubit RB sequence to the last qubit. The rationale for this choice is that a linear topology can be easily embedded in the connectivity graph of all quantum computers we consider in this work, and therefore this choice makes our benchmarks more consistent across different computers.

Note that the parameter d is the number of (parallel) random Clifford elements, not the actual physical depth of the circuit. Each Clifford element must be decomposed into two-qubit gates and single-qubit gates. The total number of two-qubit gates depends on both d and the number of qubits n. In Table II, we report the average number of two-qubit and single-qubit gates used in our experiments.

d	$n=3$	$n=5$	$n=12$
1	3 (22)	7 (39)	19 (99)
3	6 (44)	11 (73)	36 (204)
5	9 (64)	18 (118)	53 (307)
7	12 (80)	24 (150)	73 (403)
9	15 (108)	31 (194)	89 (506)
12	18 (135)	37 (242)	115 (651)

TABLE II: Average number of two-qubit (single-qubit) gates for an n-qubit, depth d RB circuit. The average is taken over ten random instances. Note that the number of single-qubit gates may differ on different hardware due to the final compilation into native gates, but the number of two-qubit gates is hardware-independent.

2. *Mirror circuits*

We also use mirror circuits [PSR+21, PRY+22] as a benchmark problem. Mirror circuits are similar to RB circuits in that they have a structure of random layers, but their final state $|z_C\rangle$ is randomized. This configuration allows a more uniform sampling of measurement errors.

An n-qubit, depth d mirror circuit C is a randomized sequence of d Clifford layers and Pauli layers, where Clifford layers are organized in such a way to conjugate a Pauli layer into a rotated Pauli layer (see Fig. 1 of [PSR+21]). The full mirror circuit C is equivalent to a random Pauli operator $\mathcal{P}$, thus the final ideal (noiseless) state is a random computational basis state

$|z_C\rangle = \mathcal{P}|00\ldots0\rangle$, and we take the observable to be $\hat{A} = |z_C\rangle\langle z_C|$.

It is important to notice that the Clifford depth d reported in our results for mirror circuits is the number of random Clifford layers. Since each Clifford layer is compiled into elementary gates, and since additional random Pauli layers are present in the circuit, the Clifford depth d is different from the number of physical gates applied in the circuit. The average number of two-qubit and single-qubit gates used in our experiments are reported in Table III for different values of d and n.

d	$n = 2$	$n = 5$	$n = 12$
1	2 (26)	4 (41)	10 (95)
3	6 (46)	12 (68)	30 (158)
5	10 (68)	20 (98)	51 (223)
7	14 (87)	28 (125)	72 (284)
9	18 (108)	36 (154)	92 (350)
12	24 (138)	48 (197)	121 (448)

TABLE III: Average number of two-qubit (single-qubit) gates for an n-qubit, depth d mirror circuit. The average is taken over ten random instances. Note that the number of single-qubit gates may differ on different hardware due to the final compilation into native gates, but the number of two-qubit gates is hardware-independent.

D. Quantum computers

We test error mitigation techniques with each benchmark circuit on four quantum computers — IBMQ Kolkata, IBMQ Lima, Rigetti Aspen-M2, and IonQ Harmony — shown in Fig. 1 and described in the following sections. We also perform experiments on noisy quantum computer simulators for comparison to hardware and for additional experiments. The noise models we use are described in Sec. II D 5.

1. *IBMQ Lima*

The IBMQ Lima computer consists of five superconducting transmon qubits arranged in a "T-shape" topology shown in Fig. 1(c). The error rates for the computer are listed in Table V in Appendix B 3. In our $n = 3$ qubit experiments, we use the qubits with the lowest two-qubit error rates, namely the qubits labeled $(0, 1, 2)$. For $n = 5$ qubit experiments we use all qubits on the device.

2. *Rigetti Aspen-M2*

The Rigetti Aspen-M2 computer consists of 80 superconducting qubits arranged in a hexagonal lattice shown in Fig. 1(c). The error rates for the computer are listed in Table VII in Appendix B 3. We perform $n = 3$ qubit experiments on Rigetti Aspen-M2 using a line of qubits with relatively low two-qubit error rates, namely the qubits labeled $(10, 17, 113)$ highlighted in Fig. 1(c). Due to limited device availability, we were only able to perform $n = 3$ qubit experiments on this computer.

3. *IonQ Harmony*

The IonQ Harmony computer consists of 11 trapped ion qubits with all-to-all connectivity, shown in Fig. 1(c). Unlike the IBMQ Lima and Rigetti Aspen-M2 computers, at the time of performing experiments on IonQ Harmony, it was not possible to select which qubits to use when submitting jobs or to check which qubits were used after jobs were completed. The average one-qubit and two-qubit gate errors are respectively $\epsilon_{1Q} = 0.0029$ and $\epsilon_{2Q} = 0.0073$.

It is also worthwhile to note that, at the time of performing experiments, it was not possible (from the AWS platform) to disable compilation on IonQ Harmony, unlike on IBMQ Lima and on Rigetti Aspen-M2. Disabling compilation is important in error mitigation because techniques often insert gates that are logically trivial (e.g., $GG^\dagger$ in zero-noise extrapolation) or perform other modifications to produce circuits that are meant to be run exactly as specified. To avoid these problems when running ZNE on IonQ Harmony, we add barriers of single-qubit infinitesimal rotations as described in Appendix B 6. Due to limited device availability, we were only able to perform $n = 3$ qubit experiments on this computer.

4. *IBMQ Kolkata*

To assess the performance of error mitigation on larger benchmark problem sizes (namely, $n = 12$ qubit experiments), we use the IBMQ Kolkata device based on the 27-qubit superconducting chip — see Appendix B 3 for the coupling map (Fig. 6) and error rates (Table VI).

5. *Noisy quantum computer simulators*

In addition to quantum computers, we also perform experiments on noisy quantum computer simulators (hereafter "noisy simulators"). There are two primary reasons for this. First, this allows us to compare how error mitigation performs with simple noise models relative to actual quantum computers. Second, using noisy simulators allows us to circumvent practical limitations like device availability to perform additional experiments.

We consider two classes of noisy simulators: (i) one which implements a simple noise model of single-qubit depolarizing noise after each gate, and (ii) one which is based on the error rates of a particular computer and

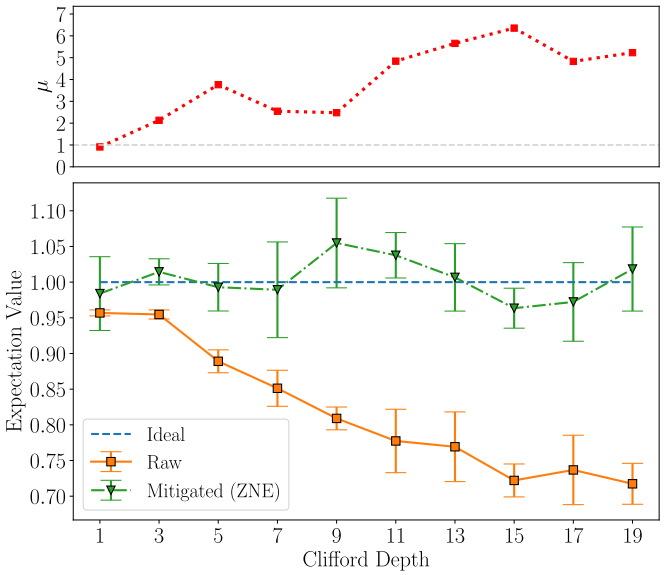

FIG. 2: (Bottom panel) Unmitigated expectation values (orange squares) and the corresponding mitigated expectation values using ZNE(R) (green triangles) for $n = 3$ qubit RB circuits executed on IBMQ Lima. For all depths, the ideal expectation value is equal to 1 (dotted line). (Top panel) The improvement factor Eq. (5) at each depth for the results in the bottom panel.

so is meant to closely emulate that computer. The simple noise model we use is 1% depolarizing noise after each two-qubit gate — i.e., each two-qubit gate (CNOT) is replaced by Eq. (10) with $p = 0.01$. In addition to this simple noise model, we also use a noise model based on the error rates of the IBMQ Lima computer (referred to as FakeLima). This noisy simulator has the same topology as IBMQ Lima shown in Fig. 1(c) and includes inhomogeneous single-qubit gate errors, two-qubit gate errors, and measurement errors based on the device characteristics in Table V. Similarly, we used IBMQ FakeKolkataV2 to classically simulate the hardware experiments performed on the real IBMQ Kolkata device. Its coupling map and error rates are reported in Fig. 6 and Table VI, respectively.

III. RESULTS

As described in Sec. II, we applied ZNE(L), ZNE(R), and PEC to RB circuit and mirror circuit benchmarks on IBM, IonQ, and Rigetti quantum computers, as well as noisy quantum computer simulators. For each of these experiments, we compute the improvement factor Eq. (5) to quantify the performance of each error mitigation technique.

An example of the results from one particular experiment is shown in Fig. 2. Here, we show the result from applying ZNE(R) to $n = 3$ qubit RB circuits of various depths on IBMQ Lima. At each depth, we generate $|\mathcal{C}| = 4$ RB circuits and evaluate the expectation value with and without error mitigation using $t = 1$ trial, and use this to compute the improvement factor Eq. (5). As shown in Fig. 2, the "raw" (unmitigated) results diverge from the ideal (noiseless) expectation value as the depth d increases, while the ZNE(R) results are closer to the ideal expectation value but generally have higher variance. This is quantified in the improvement factor which here ranges from $\mu_{\mathrm{ZNE(R)}} \simeq 1$ to $\mu_{\mathrm{ZNE(R)}} \simeq 6$. In particular, all depths $d > 1$ show an improvement factor $\mu_{\mathrm{ZNE(R)}} > 1$, indicating ZNE(R) was always beneficial to use in this example.

We show the results of all $n = 3$ experiments in Fig. 3. Here, results are arranged in a grid displaying error mitigation techniques and benchmark problems, and different colored markers in each subplot show results on different quantum computers, including noisy simulators. As a baseline for comparison, we consider a very simple noise model of 1% depolarizing noise (see Sec. II D 5), and we find as expected that this simple noise model generally produces the largest improvement factors in experiments. (The interesting exception is ZNE(R) for which IBMQ Lima shows the largest improvement factors.) On real quantum computers, there are additional sources of error including state preparation and measurement (SPAM) error as well as more complicated (in)coherent gate and crosstalk errors, so it is expected — as we see in the results — that these improvement factors are lower. However the improvement factors on the IBMQ Lima computer — as well as the improvement factors on the IBMQ Lima simulator which follow the real computer fairly closely — are still above $\mu = 1$, generally ranging between $\mu \simeq 1$ and $\mu \simeq 4$, indicating that error mitigation is beneficial on this device.

The improvement factors for PEC are lower, between $\mu \simeq 1$ and $\mu \simeq 2$, so PEC was generally less beneficial to run than ZNE, but still more beneficial than no error mitigation for most backends (IBM, IonQ, and all simulators.). Moreover, we should take into account that PEC was applied assuming a very simplified noise model (depolarizing) and so we expect better performances when PEC is based on a more faithful noise characterization. On IonQ Harmony, there are several cases where $\mu < 1$, especially in ZNE(R) experiments, so ZNE(R) was generally worse to use than no error mitigation on this computer, while ZNE(L) was more beneficial than no error mitigation. Interestingly, most improvement factors on Rigetti Aspen-M2 are close to $\mu = 1$, so error-mitigated results were generally the same as unmitigated results on this computer. Recall that our improvement factor Eq. (5) normalizes by additional shots.

We repeat the same type of experiments using $n = 5$ qubits and show these results in a similar format in Fig. 4. Here we were unable to perform experiments on Rigetti Aspen-M2 or IonQ Harmony due to limited device availability. We see again in these results that the improvement factors for the simple 1% depolarizing noise model

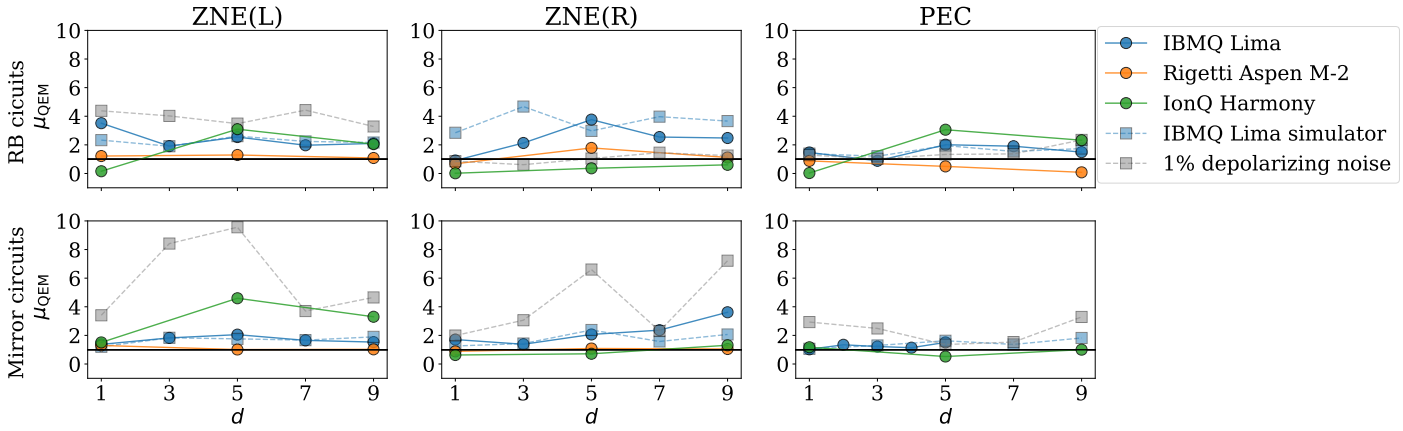

FIG. 3: Improvement factor Eq. (5) results for $n = 3$ qubit experiments. From left to right, the quantum error mitigation techniques are ZNE with linear extrapolation, ZNE with Richardson extrapolation, and probabilistic error cancellation. The top panel shows improvement factors for over $|\mathcal{C}| = 4$ randomized benchmarking circuits, and the bottom panel shows results for $|\mathcal{C}| = 4$ mirror circuits. Circle markers show quantum computer results and square markers show noisy quantum computer simulator results. In most cases, improvement factors are highest for the 1% depolarizing noise model (grey squares), which is expected as this is the simplest noise model. Improvement factors on IBMQ Lima (blue circles) are almost always above $\mu = 1$, and the IBMQ Lima simulator results (blue squares) follow the computer results fairly closely. Improvement factors on Rigetti Aspen-M2 (orange circles) and IonQ Harmony (green circles) are frequently below $\mu = 1$ — notably for ZNE(R) RB circuits — indicating that error mitigation did not help in these experiments. Improvement factors from PEC are notably smaller than the ZNE experiments but are mostly above $\mu = 1$.

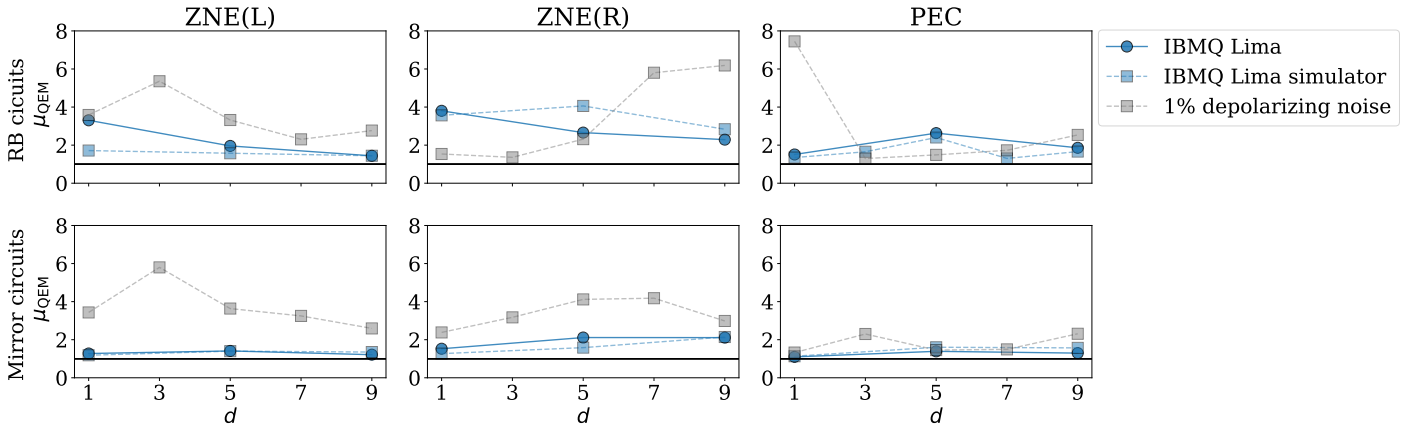

FIG. 4: Improvement factor Eq. (5) results for $n = 5$ qubit experiments, in the same format as Fig. 3. In this case, we still see that improvement factors are usually highest on the 1% depolarizing noise simulator, as expected. The average improvement factors for ZNE are comparable but slightly lower than those of the $n = 3$ qubit experiments, whereas the average improvement factors for PEC are noticeably larger than those of the $n = 3$ qubit experiments.

are generally the largest, as expected. The improvement factors on IBMQ Lima are comparable in magnitude to the $n = 3$ experiments, and in all cases, $\mu \geq 1$ so error mitigation was always beneficial. We also see that the IBMQ Lima simulator results follow the results of the real computer fairly closely, as in the $n = 3$ qubit experiment.

To further test the performance of quantum error mitigation as the problem size increases, we perform $n = 12$ qubit experiments on both a hardware device and a noisy quantum computer simulator and show these results in Fig. 5. The hardware device is the 27-qubit IBMQ Kolkata computer and the noisy simulator is based on this Kolkata device (see Fig. 6 for the coupling map and Table VI for the error rates). Here we see that improvement factors range from $\mu \simeq 1$ to $\mu \simeq 3$ indicating that error mitigation is still effective on larger problem sizes. The improvement factors for (parallel) randomized benchmarking circuits are higher than for mirror circuits in all cases, likely due to the fact that mirror circuits con-

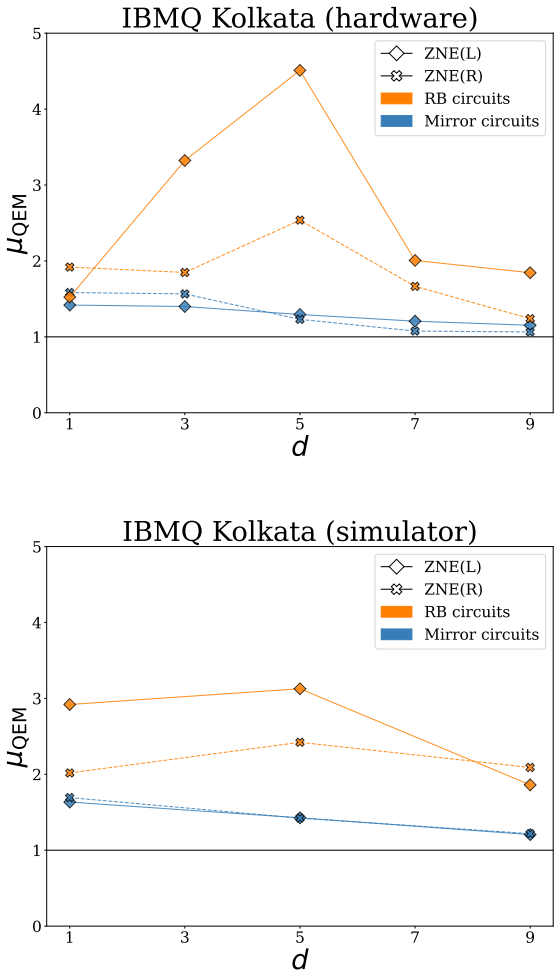

FIG. 5: Improvement factor Eq. (5) results for $n = 12$ qubit experiments on both a quantum device and a noisy simulator based on the IBMQ Kolkata computer (see Fig. 6 for the coupling map and Table VI for error rates).

tain two-qubit gates across all edges. This is somewhat visible in $n = 3, 5$ qubit experiments but more accentuated here due to the larger problem size, and suggests that additional error mitigation on top of ZNE may be necessary to mitigate errors and crosstalk effects in larger applications. However, ZNE still performed better than the unmitigated experiments in all cases.

IV. DISCUSSION

A. Positive features of our work

A positive aspect of our experiments is that we apply error mitigation techniques "out-of-the-box", i.e., we do not tailor any techniques to the benchmark problems or computers we consider. Indeed, all experiments were performed with high-level API calls to quantum error mitigation software [LMK+20] (see Appendix B for more on the implementation details). While tailoring techniques to specific experiments are likely to provide better results and is advisable in most applications, our approach gives a picture of what can be expected from quantum error mitigation in general applications.

Another positive feature of our work is the comparison of results across several computers. This experimentally verifies that error mitigation can indeed be viewed as an algorithmic or software method independent of hardware, but our results also emphasize that the performance of error mitigation depends on the underlying computer. A clear example that illustrates this is zero-noise extrapolation with very deep circuits such that the final state is approximately the maximally mixed state. In this scenario scaling noise further does not produce any signal from which one can extrapolate, so ZNE has no advantage relative to no error mitigation. On a more accurate quantum computer, however, the final state may not be maximally mixed and noise may be able to be scaled with small-scale factors. On the opposite limit, if a quantum computer is already very accurate, the improvement factor due to error mitigation is necessarily small. In fact, in the limit of very weak noise, the bias of expectation values is negligible compared to the statistical variance which is typically not reduced by error mitigation (actually it is often increased by it).

Beyond experimental results, our work introduces a quantitative, problem-independent, and resource-normalized measure of the improvement of quantum error mitigation, the improvement factor (namely, Eq. (5)). This is a natural and empirically-motivated measure that introduces a standardized metric for measuring and comparing error-mitigated quantum computer performance. For this reason, we expect the improvement factor metric to be used in other future experiments, beyond this work. We have shown the relation of our metric to the metric in [CDM+22] so that results can be compared, and we encourage the use of quantitative metrics in future error mitigation work to continue this effort. Finally, we incorporated the notion of normalizing by additional resources (namely, shots) in our definition of the improvement factor, and experimentally showed error mitigation can still be beneficial even when adjusting by these extra resource requirements.

B. Limitations of our work

While the improvement factor defined in Eq. (5) normalizes by additional shots used in quantum error mitigation, it does not normalize by additional qubits, gates, or circuits. While the error mitigation techniques we used in this work do not increase the number of qubits in executed circuits, other techniques do require more qubits in executed circuits, and accounting for this resource is important to understand the value of quantum error mit-

igation.

In more detail, the techniques we use can increase both the number of gates and different circuits executed in an experiment. In particular, in ZNE the number of gates in each circuit is primarily increased but the number of circuits is only slightly increased, while in PEC the number of gates in each circuit is essentially the same as the original circuit but the number of circuits to execute is significantly higher. The optimal way to account for these additional resources is not immediately obvious, for example, whether to count additional circuits separately or to only count the total number of gates in all circuits. Similarly, even if additional circuits use the same number of qubits as the original circuit, they could be executed simultaneously by using additional qubits, so there is some subtlety concerning the most appropriate way to normalize for these space-time tradeoffs. Ultimately these questions are likely to depend on the particular computer being used, but the variety and flexibility of available error-mitigating techniques should be encouraging to builders and users of the many different quantum computer architectures being proposed.

Another potential resource in quantum error mitigation is pre-processing / noise characterization. For example, in PEC one needs to determine the noisy basis of the computer, and in measurement error mitigation one needs to characterize the confusion matrix. However, these examples and others usually do not grow with the size of the problem. Thus they are likely less important to account for in the improvement factor. (Recall that we assumed a particular noise model for PEC and did not perform gate characterization, so pre-processing cost is not present in our improvement factor results.) Applying "out-of-the-box" error mitigation techniques at the gate level may be paired with tailored and more complex noise models going beyond error calibration information produced by hardware providers [SLM+22, BMKT22].

Although we experimentally evaluate quantum error mitigation more generally than current literature (Table I), we still considered just two error mitigation techniques out of (roughly) dozens proposed in the literature. Additionally, both benchmark problems we used are based on random circuits — while we expect our results to extend to structured circuits (say for time evolution), this needs to be experimentally verified. Due to limited device availability (i.e., which computers and how much computer time we had access to), we were only able to perform experiments on up to $n = 12$ qubits. This is fairly typical for error mitigation experiments (Table I) and we used noisy simulators to test the performance of error mitigation on larger problem sizes, but experiments on larger computers are still desirable. Last, we only considered experiments with a single error mitigation technique. Experiments composing two or more error mitigation techniques, for example, zero-noise extrapolation with dynamical decoupling and measurement error mitigation, will likely yield the largest improvement in applications. We leave quantifying this improvement (again normalized by additional resources used) and other points mentioned here for future work.

C. Relationship to literature

Our work adds a significant number of quantum error mitigation experiments relative the current literature (Table I). Notably, our work introduces a quantitative, problem-independent, and resource-normalized measure of the improvement of quantum error mitigation, and we compute this quantity on multiple quantum computers. With the notable exception [CDM+22], our work goes significantly beyond most experimental studies of quantum error mitigation which typically consider one technique for a specific experiment and do not explicitly evaluate a quantitative improvement of quantum error mitigation.

D. Future outlook of QEM

Based on our results and other experiments in the literature using error mitigation, we expect quantum error mitigation to be an essential component of virtually all experiments on "NISQ" computers [Pre18], where we can take "NISQ" to roughly mean computers with up to $n \sim 10^2$ qubits capable of implementing $d \sim 10^3$ two-qubit gates. Indeed, depending on how much overhead is required for error correction, error mitigation techniques may continue to be important at even larger scales.

Abstractly, quantum error mitigation can be viewed as combining NISQ computers with classical processors, and we have shown that this combination is still beneficial even when normalized by additional resources used. This combination is likely to be most beneficial when applied to problems that are classically hard [AAB+19]. In this setting, a quantum computer is used to get a rough solution to a hard problem, and a classical computer is used to improve the accuracy of the solution. We expect that combining the strengths of both devices in this manner will be necessary for solving problems too hard for either device to solve individually.

Furthermore, it is likely that quantum error mitigation will play an important role beyond NISQ computations, namely in error-corrected computations. Although QEM and QEC are sometimes thought of as separate techniques due to their different resource requirements and generality, they are similar in that both are algorithmic or software techniques to deal with errors in quantum computers. For example, dynamical decoupling — largely considered to be a quantum error mitigation technique in many settings — has been an important technique in recent quantum error correction experiments [AI22] to improve the fidelity of data qubits while syndrome measurements are performed. Further, Ref. [SEFT22] provides a more theoretical discussion about the application of quantum error mitigation in fault-tolerant quantum

computing. We anticipate additional work connecting error mitigation to error correction at both the theoretical and experimental levels. Our results and experimental framework for obtaining these results [LMK+20] provide some foundation for progress in this direction.

V. CONCLUSION

In this work we experimentally tested the performance of quantum error mitigation. Using an empirically-motivated metric that normalizes by the amount of additional resources used in a quantum error mitigation technique, we quantified the improvement of error mitigation using a variety of benchmark problems and quantum computers. In particular, we tested zero-noise extrapolation and probabilistic error mitigation on two benchmark problems and three quantum computers. The largest of such error mitigation benchmarks involved quantum circuits acting on 12 qubits with more than 100 two-qubit gates and more than 600 single qubit gates. Our results show that error mitigation is on average more useful than no error mitigation, even when normalizing by the additional resources used and applying "out-of-the-box" error mitigation — i.e., not tailoring the technique to the specific benchmark problem or the specific quantum computer. While these latter points are likely to provide further improvements and are encouraged in applications, our results provide a general picture of what can be expected of quantum error mitigation in practice.

Our definition of the improvement factor is, to our knowledge, the first quantitative metric to normalize by additional resources used in error mitigation, and we encourage the adoption of this or similar metrics in future work. It is also of interest to expand this metric for resources we do not account for here — for example additional qubits and gates — to more fully understand and quantify the value of quantum error mitigation in real experiments. This also can be considered with error mitigation techniques we did not use here — for example dynamical decoupling) [VKL99, VL98, ZSBS14, PAFL18, DTDQ21], Clifford data regression [LGC+21, CACC21], and noise-extended probabilistic error cancellation [MSZ21]. Additional experimental and theoretical results of this nature will help to further the progress made in this work and better understand the value of error mitigation in the larger context of quantum computing and quantum error correction.

Note added: While preparing our manuscript, we noticed a recent review of quantum error mitigation [CBB+22] which is similar in scope but discusses quantum error mitigation from a more theoretical rather than experimental perspective as in this paper.

CODE AND DATA AVAILABILITY

Code and data are available upon reasonable HTTPS request to https://github.com/unitaryfund/research/.

ACKNOWLEDGEMENTS

We would like to thank Ethan Hickman for proposing, in a public discussion on the Mitiq Discord channel, the idea of using small rotations to block backend compilation. We would also like to thank Derek Wang for running the IBM Kolkata hardware experiments. This work was supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Accelerated Research in Quantum Computing under Award Numbers DE-SC0020266 and DE-SC0020316 as well as by IBM under Sponsored Research Agreement No. W1975810. We thank IBM, Rigetti, and IonQ for providing access to their quantum computers. We thank the AWS Braket team for facilitating access to Rigetti and IonQ computers through the AWS Braket platform. The views expressed in this paper are those of the authors and do not reflect those of AWS, IBM, IonQ, or Rigetti.

[AAB+19] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C. Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando G. S. L. Brandao, David A. Buell, Brian Burkett, Yu Chen, Zijun Chen, Ben Chiaro, Roberto Collins, William Courtney, Andrew Dunsworth, Edward Farhi, Brooks Foxen, Austin Fowler, Craig Gidney, Marissa Giustina, Rob Graff, Keith Guerin, Steve Habegger, Matthew P. Harrigan, Michael J. Hartmann, Alan Ho, Markus Hoffmann, Trent Huang, Travis S. Humble, Sergei V. Isakov, Evan Jeffrey, Zhang Jiang, Dvir Kafri, Kostyantyn Kechedzhi, Julian Kelly, Paul V. Klimov, Sergey Knysh, Alexander Korotkov, Fedor Kostritsa, David Landhuis, Mike Lindmark, Erik Lucero, Dmitry Lyakh, Salvatore Mandrà, Jarrod R. McClean, Matthew McEwen, Anthony Megrant, Xiao Mi, Kristel Michielsen, Masoud Mohseni, Josh Mutus, Ofer Naaman, Matthew Neeley, Charles Neill, Murphy Yuezhen Niu, Eric Ostby, Andre Petukhov, John C. Platt, Chris Quintana, Eleanor G. Rieffel, Pedram Roushan, Nicholas C. Rubin, Daniel Sank, Kevin J. Satzinger, Vadim Smelyanskiy, Kevin J. Sung, Matthew D. Trevithick, Amit Vainsencher, Benjamin Villalonga, Theodore White, Z. Jamie Yao, Ping Yeh, Adam Zalcman, Hartmut Neven, and John M. Martinis. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, Oct 2019.

[AI21] Google Quantum AI. Exponential suppression of bit or phase errors with cyclic error correction. *Nature*, 595:383–387, 2021.

[AI22] Google Quantum AI. Suppressing quantum errors by scaling a surface code logical qubit. *arXiv preprint arXiv:2207.06431*, 2022.

[Ama20] Amazon Web Services. Amazon Braket, 2020.

[BMKT22] Ewout van den Berg, Zlatko K Minev, Abhinav Kandala, and Kristan Temme. Probabilistic error cancellation with sparse Pauli-Lindblad models on noisy quantum processors. *arXiv preprint arXiv:2201.09866*, 2022.

[CACC21] Piotr Czarnik, Andrew Arrasmith, Patrick J Coles, and Lukasz Cincio. Error mitigation with Clifford quantum-circuit data. *Quantum*, 5:592, 2021.

[CBB+22] Zhenyu Cai, Ryan Babbush, Simon C. Benjamin, Suguru Endo, William J. Huggins, Ying Li, Jarrod R. McClean, and Thomas E. O'Brien. Quantum error mitigation. *arXiv preprint arXiv:2210.00921*, 2022.

[CBS+19] Andrew W Cross, Lev S Bishop, Sarah Sheldon, Paul D Nation, and Jay M Gambetta. Validating quantum computers using randomized model circuits. *Physical Review A*, 100(3):032328, 2019.

[CDM+22] Cristina Cirstoiu, Silas Dilkes, Daniel Mills, Seyon Sivarajah, and Ross Duncan. Volumetric benchmarking of error mitigation with Qermit. *arXiv preprint arXiv:2204.09725*, 2022.

[CGC+13] Antonio D Córcoles, Jay M Gambetta, Jerry M Chow, John A Smolin, Matthew Ware, Joel Strand, Britton LT Plourde, and Matthias Steffen. Process verification of two-qubit quantum gates by randomized benchmarking. *Physical Review A*, 87(3):030301, 2013.

[CMSC22] Piotr Czarnik, Michael McKerns, Andrew T. Sornborger, and Lukasz Cincio. Improving the efficiency of learning-based error mitigation. (arXiv:2204.07109), Apr 2022. arXiv:2204.07109 [quant-ph].

[CRD+18] James I Colless, Vinay V Ramasesh, Dar Dahlen, Machiel S Blok, Mollie E Kimchi-Schwartz, Jarrod R McClean, Jonathan Carter, Wibe A de Jong, and Irfan Siddiqi. Computation of molecular spectra on a quantum processor with an error-resilient algorithm. *Physical Review X*, 8(1):011021, 2018.

[CS96] A Robert Calderbank and Peter W Shor. Good quantum error-correcting codes exist. *Physical Review A*, 54(2):1098, 1996.

[DTDQ21] Poulami Das, Swamit Tannu, Siddharth Dangwal, and Moinuddin Qureshi. ADAPT: Mitigating idling errors in qubits via adaptive dynamical decoupling. In *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 950–962, 2021.

[EBL18] Suguru Endo, Simon C Benjamin, and Ying Li. Practical quantum error mitigation for near-future applications. *Physical Review X*, 8(3):031027, 2018.

[ECBY21] Suguru Endo, Zhenyu Cai, Simon C Benjamin, and Xiao Yuan. Hybrid quantum-classical algorithms and quantum error mitigation. *Journal of the Physical Society of Japan*, 90(3):032001, 2021.

[EME10] Jay M. Easwar Magesan, Gambetta and Joseph Emerson. Robust randomized benchmarking of quantum processes. *arXiv preprint arXiv:1009.3639*, 2010.

[FHV+22] Samuele Ferracin, Akel Hashim, Jean-Loup Ville, Ravi Naik, Arnaud Carignan-Dugas, Hammam Qassim, Alexis Morvan, David I Santiago, Irfan Siddiqi, and Joel J Wallman. Efficiently improving the performance of noisy quantum computers. *arXiv preprint arXiv:2201.10672*, 2022.

[GCM+12] Jay M Gambetta, Antonio D Córcoles, Seth T Merkel, Blake R Johnson, John A Smolin, Jerry M Chow, Colm A Ryan, Chad Rigetti, Stefano Poletto, Thomas A Ohki, Mark B Ketchen, and M Steffen. Characterization of addressability by simultaneous randomized benchmarking. *Physical Review Letters*, 109(24):240504, 2012.

[GTHL+20] Tudor Giurgica-Tiron, Yousef Hindy, Ryan LaRose, Andrea Mari, and William J Zeng. Digital zero noise extrapolation for quantum error mitigation. In *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)*, pages 306–316. IEEE, 2020.

[HWFZ20] He-Liang Huang, Dachao Wu, Daojin Fan, and Xiaobo Zhu. Superconducting quantum computing: a review. *Science China Information Sciences*, 63(8):1–32, 2020.

[IBM] IBM. IBM Quantum. https://quantum-computing.ibm.com/.

[JJAB+21] Petar Jurcevic, Ali Javadi-Abhari, Lev S Bishop, Isaac Lauer, Daniela F Bogorin, Markus Brink, Lauren Capelluto, Oktay Günlük, Toshinari Itoko, Naoki Kanazawa, Abhinav Kandala, George A Keefe, Kevin Krsulich, William Landers, Eric P Lewandowski, Douglas T McClure, Giacomo Nannicini, Adinath Narasgond, Hasan M Nayfeh, Emily Pritchett, Mary Beth Rothwell, Srikanth Srinivasan, Neereja Sundaresan, Ken X Wang, Cindy Wei, Christopher J Wood, Jeng-Bang Yau, Eric J Zhang, Oliver E Dial, Jerry M Chow, and Jay M Gambetta. Demonstration of quantum volume 64 on a superconducting quantum computing system. *Quantum Science and Technology*, 6(2):025020, 2021.

[KTC+19] Abhinav Kandala, Kristan Temme, Antonio D Córcoles, Antonio Mezzacapo, Jerry M Chow, and Jay M Gambetta. Error mitigation extends the computational reach of a noisy quantum processor. *Nature*, 567(7749):491–495, 2019.

[KWY+21] Youngseok Kim, Christopher J Wood, Theodore J Yoder, Seth T Merkel, Jay M Gambetta, Kristan Temme, and Abhinav Kandala. Scalable error mitigation for noisy quantum circuits produces competitive expectation values. *arXiv preprint arXiv:2108.09197*, 2021.

[LB17] Ying Li and Simon C Benjamin. Efficient variational quantum simulator incorporating active error minimization. *Physical Review X*, 7(2):021050, 2017.

[LGC+21] Angus Lowe, Max Hunter Gordon, Piotr Czarnik, Andrew Arrasmith, Patrick J Coles, and Lukasz Cincio. Unified approach to data-driven quantum error mitigation. *Physical Review Research*, 3(3):033098, 2021.

[LMK+20] Ryan LaRose, Andrea Mari, Sarah Kaiser, Peter J Karalekas, Andre A Alves, Piotr Czarnik, Mohamed El Mandouh, Max H Gordon, Yousef Hindy, Aaron Robertson, Purva Thakre, Misty Wahl, Danny Samuel, Rahul Mistri, Maxime Tremblay, Nick Gardner, Nathaniel T Stemen, Nathan Shammah, and William J Zeng. Mitiq: A software package for error mitigation on noisy quantum computers. *arXiv preprint arXiv:2009.04417*, 2020.

[LMR+22] Ryan LaRose, Andrea Mari, Vincent Russo, Dan Strano, and William J Zeng. Error mitigation increases the effective quantum volume of quantum computers. *arXiv preprint arXiv:2203.05489*, 2022.

[MGE12] Easwar Magesan, Jay M Gambetta, and Joseph Emerson. Characterizing quantum gates via randomized benchmarking. *Physical Review A*, 85(4):042311, 2012.

[MSS+19] David C McKay, Sarah Sheldon, John A Smolin, Jerry M Chow, and Jay M Gambetta. Three-qubit randomized benchmarking. *Physical Review Letters*, 122(20):200502, 2019.

[MSZ21] Andrea Mari, Nathan Shammah, and William J Zeng. Extending quantum probabilistic error cancellation by noise scaling. *Physical Review A*, 104(5):052607, 2021.

[PAFL18] Bibek Pokharel, Namit Anand, Benjamin Fortman, and Daniel A Lidar. Demonstration of fidelity improvement using dynamical decoupling with superconducting qubits. *Physical Review Letters*, 121(22):220502, 2018.

[Pre18] John Preskill. Quantum computing in the NISQ era and beyond. *Quantum*, 2:79, Aug 2018. arXiv: 1801.00862.

[PRY+22] Timothy Proctor, Kenneth Rudinger, Kevin Young, Erik Nielsen, and Robin Blume-Kohout. Measuring the capabilities of quantum computers. *Nature Physics*, 18(1):75–79, 2022.

[PSR+21] Timothy Proctor, Stefan Seritan, Kenneth Rudinger, Erik Nielsen, Robin Blume-Kohout, and Kevin Young. Scalable randomized benchmarking of quantum computers using mirror circuits. *arXiv preprint arXiv:2112.09853*, 2021.

[QCA+20] Google AI Quantum, Collaborators*†, Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C Bardin, Rami Barends, Sergio Boixo, Michael Broughton, Bob B Buckley, et al. Hartree-Fock on a superconducting qubit quantum computer. *Science*, 369(6507):1084–1089, 2020.

[SBMS+19] Ramiro Sagastizabal, Xavier Bonet-Monroig, Malay Singh, M Adriaan Rol, CC Bultink, Xiang Fu, CH Price, VP Ostroukh, N Muthusubramanian, A Bruno, M Beekman, N Haider, TE O'Brien, and L DiCarlo. Experimental error mitigation via symmetry verification in a variational quantum eigensolver. *Physical Review A*, 100(1):010302, 2019.

[SCW+19] Chao Song, Jing Cui, H Wang, J Hao, H Feng, and Ying Li. Quantum computation with universal error mitigation on a superconducting quantum processor. *Science advances*, 5(9):eaaw5686, 2019.

[SCZ+22] Alireza Seif, Ze-Pei Cian, Sisi Zhou, Senrui Chen, and Liang Jiang. Shadow distillation: Quantum error mitigation with classical shadows for near-term quantum processors. *arXiv preprint arXiv:2203.07309*, 2022.

[SEFT22] Yasunari Suzuki, Suguru Endo, Keisuke Fujii, and Yuuki Tokunaga. Quantum error mitigation as a universal error reduction technique: Applications from the nisq to the fault-tolerant quantum computing eras. *PRX Quantum*, 3(1):010345, 2022.

[Sho95] Peter W Shor. Scheme for reducing decoherence in quantum computer memory. *Physical review A*, 52(4):R2493, 1995.

[SL22] Peter Stano and Daniel Loss. Review of performance metrics of spin qubits in gated semiconducting nanostructures. *Nature Reviews Physics*, 4(10):672–688, 2022.

[SLM+22] Kevin Schultz, Ryan LaRose, Andrea Mari, Gregory Quiroz, Nathan Shammah, B David Clader, and William J Zeng. Reducing the impact of time-correlated noise on zero-noise extrapolation. *arXiv preprint arXiv:2201.11792*, 2022.

[SR20] Jaime Sevilla and C Jess Riedel. Forecasting timelines of quantum computing. *arXiv preprint arXiv:2009.05045*, 2020.

[SRM+22] Kaitlin N Smith, Gokul Subramanian Ravi, Prakash Murali, Jonathan M Baker, Nathan Earnest, Ali Javadi-Cabhari, and Frederic T Chong. Timestitch: Exploiting slack to mitigate decoherence in quantum circuits. *ACM Transactions on Quantum Computing*, 4(1):1–27, 2022.

[Ste96] Andrew M Steane. Error correcting codes in quantum theory. *Physical Review Letters*, 77(5):793, 1996.

[Tak21] Ryuji Takagi. Optimal resource cost for error mitigation. *Physical Review Research*, 3(3):033178, 2021.

[TBG17] Kristan Temme, Sergey Bravyi, and Jay M Gambetta. Error mitigation for short-depth quantum circuits. *Physical Review Letters*, 119(18):180509, 2017.

[UNdJ20] Miroslav Urbanek, Benjamin Nachman, and Wibe A de Jong. Error detection on quantum computers improving the accuracy of chemical calculations. *Physical Review A*, 102(2):022427, 2020.

[Uni22] UnitaryFund. UnitaryFund Research. https://github.com/unitaryfund/research/, July 2022.

[VKL99] Lorenza Viola, Emanuel Knill, and Seth Lloyd. Dynamical decoupling of open quantum systems. *Physical Review Letters*, 82(12):2417, 1999.

[VL98] Lorenza Viola and Seth Lloyd. Dynamical suppression of decoherence in two-state quantum systems. *Physical Review A*, 58(4):2733, 1998.

[Vui17] Christophe Vuillot. Is error detection helpful on IBM 5Q chips? *arXiv preprint arXiv:1705.08957*, 2017.

[ZCN+21] Yu Zhang, Lukasz Cincio, Christian F. A. Negre, Piotr Czarnik, Patrick Coles, Petr M. Anisimov, Susan M. Mniszewski, Sergei Tretiak, and Pavel A. Dub. Variational quantum eigensolver with reduced circuit complexity. (arXiv:2106.07619), Jun 2021. arXiv:2106.07619 [quant-ph].

[ZLZ+20] Shuaining Zhang, Yao Lu, Kuan Zhang, Wentao Chen, Ying Li, Jing-Ning Zhang, and Kihwan Kim. Error-mitigated quantum gates exceeding physical fidelities in a trapped-ion system. *Nature communications*, 11(1):1–8, 2020.

[ZSBS14] Jingfu Zhang, Alexandre M Souza, Frederico Dias Brandao, and Dieter Suter. Protected quantum computing: Interleaving gate operations with dynamical decoupling sequences. *Physical Review Letters*, 112(5):050502, 2014.

Appendix A: Notation

A summary of notation is shown in Table IV.

Appendix B: Implementation details

In this appendix, we discuss the implementation of the experiment and provide more details pertaining to how these experiments were programmatically carried out.

1. Experiment setup

For a given experimental run, the user specifies whether to run on a quantum hardware or simulator device, the platform to target (IBM, IonQ, or Rigetti) covered in B 3, the error mitigation method to apply (PEC or ZNE) covered in B 4, and the circuit type to consider (RB or mirror) covered in B 5.

C	Quantum circuit (unitary)
ρ	Final state of circuit C, $\rho = C\|0\rangle\langle 0\|C^\dagger$
$\hat{A}$	Hermitian observable
n	Number of qubits
d	Depth of circuit
N	Number of shots (samples)
A	Ideal expectation value $\mathrm{tr}[\rho\hat{A}]$
$\mathcal{E}$	Channel of a noisy computer (simulator)
A'	Noisy expectation value $\mathrm{tr}[\mathcal{E}(\rho)\hat{A}]$
QEM	A quantum error mitigation technique
A_{QEM}	Error-mitigated expectation value
N_{QEM}	Total number of shots used in the QEM technique
$\mathcal{C}$	A set of (benchmark) circuits
$\hat{\mathcal{A}}$	A set of (benchmark) observables
μ_{QEM}	Improvement factor Eq. (5) of QEM technique
ZNE	Zero-noise extrapolation
ZNE(L)	ZNE with linear extrapolation
ZNE(R)	ZNE with Richardson extrapolation
PEC	Probabilistic error cancellation

TABLE IV: Summary of notation. Note: The number of single-qubit and two-qubit gates in a circuit of depth d depends on the circuit type — see Sec. II C.

Once we obtain either our RB or mirror circuit C, we define an executor function that takes the input circuit and returns an expectation value $\langle A\rangle$ where $A = |z\rangle\langle z|$ for each circuit and where z denotes the correct bitstring.

The circuits may contain gates that are not in the supported gatesets for the hardware device we are targeting to run on. In this case, we compile the gates in the circuit to gates in the supported gateset (more on this process in B 5.)

We then loop over each Clifford depth and within this loop, iterate over the number of trials we want to perform at each depth. For each iteration within our trial, we calculate the result of applying our error mitigation method. The resulting data is saved. Further information on the saved data can be found in B 2.

2. Software and experiment data

Our experiments were carried out using Python 3.9. The error mitigation methods of PEC and ZNE were applied via version `0.18.0` of the Mitiq Python software package. The libraries Cirq (version `1.0.0`), `amazon-braket-sdk` (version `1.25.2`), and Qiskit (version `0.38.0`) were used to specify the circuits for our experiments. Further information on the Mitiq package can be found on the official GitHub repository https://github.com/unitaryfund/mitiq as well as in [LMK+20].

The data obtained from our experiments and used to generate the plots in this work can be found in the `data` directory

```
data/TYPE/QEM/CIRCUIT/PLATFORM/
```

where `TYPE` $\in$ {`hardware`, `software`} describes whether the experiment was run on either an actual quantum device or a simulator, `QEM` $\in$ {`pec`, `zne`} describes the error mitigation method that was applied, `CIRCUIT` $\in$ {`mirror`, `rb`} describes the circuit type considered, and where `PLATFORM` $\in$ {`ibmq`, `ionq`, `rigetti`, `depolarizing`} describes on which platform the experimental data was obtained from.

Contained in each such directory is a subfolder with the following form

```
PLATFORM_QEM_CIRCUIT_QUBITS_MIN_MAX_SHOTS_TRIALS
```

where `QUBITS` is the number of qubits used in the experiment, `MIN` is the minimum Clifford depth, `MAX` is the maximum Clifford depth, `SHOTS` is the total number of shots used in the experiment (this is 10,000 for all of our experiments) and `TRIALS` is the total number of trials carried out per experiment (this is 4 for all of our experiments).

In each such subfolder is a listing of files with the following prefixes:

- `cnot_counts`: The number of CNOT gates in the circuit.
- `noise_scaled_expectation_values`: Noise-scaled expectation values (for ZNE only).
- `noisy_values`: The non-scaled noisy expectation values (prior to applying error mitigation).
- `oneq_counts`: The number of circuit instructions (modulo the number of CNOT operations).
- `true_values`: The ideal values (these are always equal to 1).
- `mitigated_values`: The error-mitigated values.

Each row represents the value obtained at the depth corresponding to the index and each column represents the data obtained for a given trial.

Running the software in [Uni22] that is responsible for capturing quantum device hardware experiment data requires possessing an AWS Braket account (for IonQ and Rigetti) an an IBM Quantum account (for IBM). Running the software on exclusively quantum simulators can be done without any such account access.

To run the software on a simulator device the variable `use_noisy_simulator` should be set to `True` (and alternatively, `False` if the desire is to run on quantum device hardware). Setting the `mitigation_type` variable to either `pec` and `zne` runs PEC or ZNE error mitigation, respectively. The type of circuit to use can be set via the variable `circuit_type` to either `rb` for randomized benchmarking circuits or `mirror` for mirror circuits. Specifying the target platform can be done by setting the `hardware_type` variable to either `ibmq`, `ionq`, or `rigetti` for IBM, IonQ, or Rigetti, respectively.

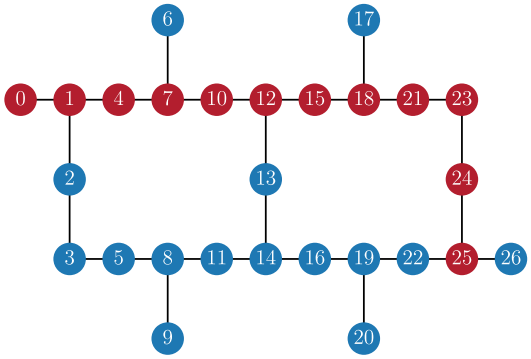

FIG. 6: Coupling map for the 27-qubit IBMQ Kolkata computer. We use this device (see Table VI for its properties) to perform $n = 12$ qubit error mitigation experiments on the red qubits. Results are shown in Fig. 5.

3. Quantum computing device platforms

Access to the Rigetti Aspen-M2 and IonQ Harmony hardware devices were provided via the Amazon Braket service on AWS. Hardware information pertaining to qubit topology, error rates, etc. were obtained via the Amazon Braket API. Access to the IBM hardware Lima device and IBM FakeLima and FakeKolkataV2 simulator devices were provided by the IBM Quantum Compute Resources page [IBM].

Qubit	ϵ_{1Q}	ϵ_{CX}	ϵ_M
0	9.028×10^{-4}	1.026×10^{-2}	2.740×10^{-2}
1	6.452×10^{-4}	1.083×10^{-2}	1.510×10^{-2}
2	6.016×10^{-4}	7.375×10^{-3}	1.700×10^{-2}
3	2.523×10^{-4}	1.570×10^{-2}	2.360×10^{-2}
4	6.167×10^{-4}	1.655×10^{-2}	4.410×10^{-2}

TABLE V: IBMQ Lima error rates for each qubit in the coupling map from Fig. 1(c). Here, ϵ_{1Q}, ϵ_{CX}, and ϵ_M represent the single-qubit $\sqrt{X}$-gate error, the average two-qubit CNOT error, and the readout assignment error, respectively [IBM].

4. Error mitigation

The PEC and ZNE error mitigation methods were applied using the Mitiq software package.

a. *ZNE*

The Mitiq package employs local folding [GTHL+20] and global folding [SLM+22] as gate-level noise scaling methods for ZNE. In this work, we used global folding.

Qubit	ϵ_{1Q}	ϵ_{CX}	ϵ_M
0	1.640×10^{-4}	3.997×10^{-3}	1.820×10^{-2}
1	1.339×10^{-4}	$5.737 \times 10^{-}$	1.850×10^{-2}
4	1.660×10^{-4}	6.636×10^{-3}	2.750×10^{-2}
7	2.027×10^{-4}	1.259×10^{-2}	2.270×10^{-2}
10	4.948×10^{-4}	1.397×10^{-2}	1.320×10^{-2}
12	2.380×10^{-4}	9.326×10^{-3}	8.500×10^{-3}
15	2.626×10^{-4}	4.086×10^{-2}	5.000×10^{-3}
18	2.248×10^{-4}	1.052×10^{-2}	6.200×10^{-3}
21	1.772×10^{-4}	6.663×10^{-3}	1.350×10^{-2}
23	2.221×10^{-4}	5.340×10^{-3}	8.100×10^{-3}
24	2.858×10^{-4}	5.027×10^{-1}	1.360×10^{-2}
25	5.048×10^{-4}	3.368×10^{-1}	6.600×10^{-3}

TABLE VI: IBMQ Kolkata error rates for each qubit used in our $n = 12$ qubit experiments (see Fig. 6). We use a noisy simulator based on these error rates to perform experiments. Here, ϵ_{1Q}, ϵ_{CX}, and ϵ_M represent the single-qubit $\sqrt{X}$-gate error, the average two-qubit CNOT error, and the readout assignment error, respectively [IBM].

Qubit specs		**Edge specs**	
Qubit	Readout fidelity	Edge	ϵ_{CX}
10	99.3%	10-17	3.79×10^{-3}
17	98.2%	10-113	4.58×10^{-3}
113	94.7%		

TABLE VII: The Rigetti Aspen-M2 measures of the average two-qubit CNOT error (ϵ_{CX}) for the qubit edge and relative readout fidelity rates for the qubits in the device that we use in our experiments. Accessed from [Ama20].

```python
import qiskit
from mitiq.zne.scaling import fold_global

# Define a quantum circuit.
qubits = qiskit.QuantumRegister(2)
circuit = qiskit.QuantumCircuit(qubits)
circuit.h(0)
circuit.cnot(0, 1)

# Apply global folding.
scaled_circuit = fold_global(circuit,
    scale_factor=3)
```

Listing 1: Apply global folding to a circuit in Mitiq.

Global folding increases the effective length of the quantum circuit by compiling the input circuit with a larger number of gates. Each set of layers in the circuit is replaced by $GG^\dagger G$. Since $GG^\dagger = I$, in the case where we are running our circuit on an ideal simulator this has no effect on the circuit. However, in the case where one uses a noisy device, this increases the noise and effective gate errors of the computation. An arbitrary example that depicts the global folding technique is shown in Fig. 7.

An example of how to make use of global folding in Mitiq is provided in Listing 1 that makes use of $\lambda = 3$

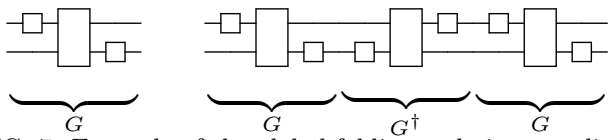

FIG. 7: Example of the global folding technique applied to an arbitrary circuit. For a given collection of gates denoted by G, we increase the effective length of the overall circuit by creating $GG^{\dagger}G$.

being a scale factor. Note that the smallest scale factor one can select is $\lambda = 1$ that corresponds to not performing any folding where selecting $\lambda = 3$ folds all of the gates in the circuit. For any scale factors $\lambda > 3$ in Mitiq, this folds some or all gates in the circuit.

For ZNE, we applied linear and Richardson extrapolation methods provided as factory objects in Mitiq as `LinearFactory` and `RichardsonFactory`, respectively.

```
from mitiq import zne

zne_value = zne.execute_with_zne(
    circuit,
    execute,
    scale_noise=zne.scaling.fold_global,
    factory=zne.inference.RichardsonFactory
        (scale_factors=[1, 2, 3])
)
```

Listing 2: Applying ZNE in Mitiq using Richardson extrapolation with noise scale factors $\lambda_i \in \{1, 2, 3\}$.

b. *PEC*

To apply probabilistic error cancellation [TBG17, EBL18, ZLZ+20] we use the associated Mitiq module. We extract the two-qubit error probability p_{2Q} from the backend properties as reported by the hardware vendor. We use this information together with the utilities in `mitiq.representations` to generate the quasi-probability representations for all the two-qubit operations acting on neighboring qubits of the quantum processor. We store the result as a list (`representations`) of `mitiq.pec.OperationRepresentation` objects. After this preliminary step, we can obtain all the error-mitigated expectation values as described in the next code block.

```
from mitiq import pec

pec_value = pec.execute_with_pec(
    circuit,
    execute,
    representations,
    num_samples=num_samples,
    random_state=local_seed,
)
```

Listing 3: Apply PEC using Mitiq.

5. Circuits

In order to construct our RB circuits, we define an RB pattern by splitting the qubits into 2-qubit pairs. We then generate a generic RB sequence via the Qiskit library. If the circuit is to be run on either Rigetti or IonQ, we perform a conversion of the Qiskit circuit to a Braket circuit.

```
import qiskit.ignis.verification.
    randomized_benchmarking as rb

def get_circuit(depth, seed):
    circuit = rb.
        randomized_benchmarking_seq(
        length_vector=[depth],
        rb_pattern=rb_pattern,
        group_gates="0",
        rand_seed=seed,
    )[0][0][0]
```

Listing 4: Defining RB circuits in Mitiq.

We generate mirror circuits via the `generate_mirror_circuit` function from Mitiq. Mirror circuits parameterize the number of random Clifford layers to be generated.

```
from mitiq.benchmarks import
    generate_mirror_circuit

def get_circuit(depth, seed):
    circuit, correct_bitstring =
        generate_mirror_circuit(
        nlayers=depth,
        two_qubit_gate_prob=1.0,
        connectivity_graph=computer,
        two_qubit_gate_name="CNOT",
        seed=seed,
        return_type=return_type,
    )
```

Listing 5: Defining mirror circuits in Mitiq.

a. *Circuit compilation*

At the time of this writing, both Rigetti and IonQ hardware support the option of *verbatim compilation*; a method that directs the compiler to run the specified circuit exactly as defined without adding any modifications. We attempt to disable automatic compilation by the platform service or QPU providers in order to have as much control as possible on the compiled circuit and hence error mitigation scaling.

The usage of verbatim compilation requires that every gate in the circuit is a gate that is natively supported by the hardware it is running on. For Aspen-M2, the native gate set is $\{\mathrm{RX}, \mathrm{RZ}, \mathrm{CPHASE}, \mathrm{CZ}, \mathrm{XY}\}$. In our `compile_to_rigetti_gateset` function, we iterate through every instruction in our circuit and compile all of the gates into equivalent ones that are in the supported

native gateset. As IonQ only recently added support for verbatim compilation, we were not able to take advantage of this option for the IonQ Harmony experiments.

b. *Task batching*

The AWS braket platform allows for task batching; the ability to launch jobs in parallel. For the majority of our experiments, serial execution within the allotted device time windows were sufficient to carry out a complete run of our experiment. One exception to this was the PEC experiments on Rigetti and IonQ hardware. In order to ensure these experiments ran within the allotted device time window, we needed to make use of the batching functionality.

6. Rotation barriers

Many hardware backends internally optimize circuits before actually running physical gates on a quantum processor. This can be a problem for some error mitigation techniques. For example, in ZNE we want to run circuits whose depth is intentionally increased by unitary folding $G \to GG^\dagger G$ (see Fig. 7). However, if the internal optimizer of a backend detects that $G^\dagger G = I$, it will simplify the unitary folding structure such that any noise scaling effect is lost. This is a relevant practical issue for gate-level ZNE.

The best way to avoid this effect is to optimize circuits before applying ZNE and to switch off any further circuit optimizations on the backend side. When this is not possible, a simple workaround is the addition of barriers of infinitesimal gates that generate a negligible unitary effect on the quantum state but that can block the action of circuit optimizers.

In practice, in our experiments we apply ZNE by using a slightly modified version of the unitary folding rule:

$$G \to GR_1G^\dagger R_2 G, \tag{B1}$$

where G is the circuit acting on n qubits and $R_j = [R_x(\epsilon_{x,j})R_y(\epsilon_{y,j})R_z(\epsilon_{z,j})]^{\otimes n}$ is the tensor-product of n infinitesimal rotations. For each circuit block of the unitary folding structure, we apply a rotation barrier. The way in which the small rotation angles are chosen is quite arbitrary as long as they are sufficiently small but nonzero. In our experiments, we randomly sample between two fixed small angles ($\pm 10^{-4}$) as reported in the next code block.

```python
import cirq
import numpy as np

def make_rotation_barrier(
    circuit,
    delta=0.0001,
):
    """Returns a barrier of infinitesimal
       rotations."""
    qubits = list(circuit.all_qubits())
    delta_x = np.random.choice([1.0, -1.0])
        * delta
    delta_y = np.random.choice([1.0, -1.0])
        * delta
    delta_z = np.random.choice([1.0, -1.0])
        * delta
    barrier = cirq.Circuit()
    for q in qubits:
        barrier.append(cirq.rx(delta_x)(q))
        barrier.append(cirq.ry(delta_y)(q))
        barrier.append(cirq.rz(delta_z)(q))
    return barrier
```

Listing 6: Function returning a layer of infinitesimal rotations. Each layer is applied as a barrier between circuit blocks as shown in Eq. (B1).

Note that all our ZNE experiments on IonQ hardware have been done via the AWS cloud platform before *verbatim compilation* became available. Today *verbatim compilation* of quantum circuits into native gates is supported for both IonQ and Rigetti devices. Therefore, the workaround of applying rotation barriers is probably not necessary anymore to reproduce similar experiments.

Platform	Computer	QEM	Extrapolation	Circuit	Qubits	Simulator
IBM	Lima	ZNE	Richardson	RB	3	No
IBM	Lima	ZNE	linear	RB	3	No
IBM	Lima	PEC	-	RB	3	No
IBM	Lima	ZNE	Richardson	RB	5	No
IBM	Lima	ZNE	linear	RB	5	No
IBM	Lima	PEC	-	RB	5	No
IBM	Lima	ZNE	Richardson	Mirror	3	No
IBM	Lima	ZNE	linear	Mirror	3	No
IBM	Lima	PEC	-	Mirror	3	No
IBM	Lima	ZNE	Richardson	Mirror	5	No
IBM	Lima	ZNE	linear	Mirror	5	No
IBM	Lima	PEC	-	Mirror	5	No
IBM	Kolkata	ZNE	Richardson	RB	3	No
IBM	Kolkata	ZNE	linear	RB	3	No
IBM	Kolkata	ZNE	Richardson	Mirror	3	No
IBM	Kolkata	ZNE	linear	Mirror	3	No
IonQ	Harmony	ZNE	Richardson	RB	3	No
IonQ	Harmony	ZNE	linear	RB	3	No
IonQ	Harmony	PEC	-	RB	3	No
IonQ	Harmony	ZNE	Richardson	Mirror	3	No
IonQ	Harmony	ZNE	linear	Mirror	3	No
IonQ	Harmony	PEC	-	Mirror	3	No
Rigetti	Aspen-M2	ZNE	Richardson	RB	3	No
Rigetti	Aspen-M2	ZNE	linear	RB	3	No
Rigetti	Aspen-M2	PEC	-	RB	3	No
Rigetti	Aspen-M2	ZNE	Richardson	Mirror	3	No
Rigetti	Aspen-M2	ZNE	linear	Mirror	3	No
IBM	FakeLima	ZNE	Richardson	RB	3	Yes
IBM	FakeLima	ZNE	linear	RB	3	Yes
IBM	FakeLima	PEC	-	RB	3	Yes
IBM	FakeLima	ZNE	Richardson	RB	5	Yes
IBM	FakeLima	ZNE	linear	RB	5	Yes
IBM	FakeLima	PEC	-	RB	5	Yes
IBM	FakeLima	ZNE	Richardson	Mirror	3	Yes
IBM	FakeLima	ZNE	linear	Mirror	3	Yes
IBM	FakeLima	PEC	-	Mirror	3	Yes
IBM	FakeLima	ZNE	Richardson	Mirror	5	Yes
IBM	FakeLima	ZNE	linear	Mirror	5	Yes
IBM	FakeLima	PEC	-	Mirror	5	Yes
IBM	FakeKolkataV2	ZNE	Richardson	RB	12	Yes
IBM	FakeKolkataV2	ZNE	linear	RB	12	Yes
IBM	FakeKolkataV2	ZNE	Richardson	Mirror	12	Yes
IBM	FakeKolkataV2	ZNE	linear	Mirror	12	Yes
AWS	1% depolarizing noise	ZNE	Richardson	RB	3	Yes
AWS	1% depolarizing noise	ZNE	linear	RB	3	Yes
AWS	1% depolarizing noise	PEC	-	RB	3	Yes
AWS	1% depolarizing noise	ZNE	Richardson	RB	5	Yes
AWS	1% depolarizing noise	ZNE	linear	RB	5	Yes
AWS	1% depolarizing noise	PEC	-	RB	5	Yes
AWS	1% depolarizing noise	ZNE	Richardson	Mirror	3	Yes
AWS	1% depolarizing noise	ZNE	linear	Mirror	3	Yes
AWS	1% depolarizing noise	PEC	-	Mirror	3	Yes

TABLE VIII: Summary of experiments performed.