Scalable in situ qubit calibration during repetitive error detection

J. Kelly, R. Barends, A. G. Fowler, A. Megrant, E. Jeffrey, T. C. White, D. Sank, J. Y. Mutus, B. Campbell, Yu Chen, Z. Chen, B. Chiaro, A. Dunsworth, E. Lucero, M. Neeley, C. Neill, P. J. J. O'Malley, C. Quintana, P. Roushan, A. Vainsencher, J. Wenner, and John M. Martinis

Phys. Rev. A **94**, 032321 — Published 26 September 2016

DOI: 10.1103/PhysRevA.94.032321

Scalable in-situ qubit calibration during repetitive error detection

J. Kelly,[1] R. Barends,[1] A. G. Fowler,[1] A. Megrant,[1] E. Jeffrey,[1] T. C. White,[1] D. Sank,[1] J. Y. Mutus,[1]
B. Campbell,[2] Yu Chen,[1] Z. Chen,[2] B. Chiaro,[2] A. Dunsworth,[2] E. Lucero,[1] M. Neeley,[1] C. Neill,[2]
P. J. J. O'Malley,[2] C. Quintana,[2] P. Roushan,[1] A. Vainsencher,[2] J. Wenner,[2] and John M. Martinis[1,2]

[1] *Google Inc., Santa Barbara, California 93117, USA*
[2] *Department of Physics, University of California, Santa Barbara, CA 93106, USA*

We present a method to optimize qubit control parameters during error detection which is compatible with large-scale qubit arrays. We demonstrate our method to optimize single or two-qubit gates in parallel on a nine-qubit system. Additionally, we show how parameter drift can be compensated for during computation by inserting a frequency drift and using our method to remove it. We remove both drift on a single qubit and independent drifts on all qubits simultaneously. We believe this method will be useful in keeping error rates low on all physical qubits throughout the course of a computation. Our method is $O(1)$ scalable to systems of arbitrary size, providing a path towards controlling the large numbers of qubits needed for a fault-tolerant quantum computer.

A fault-tolerant quantum computer protects a quantum state from the environment through the careful manipulations of millions of physical qubits [1]. In such a computer, each qubit must reliably perform a series of quantum logic gates [2–5] to detect and negate errors [6–8]. However, operating such quantities of qubits at the necessary level of precision is an open challenge, as optimal control parameters can vary between qubits [9] and drift in time [10]. Here we present a method to optimize control parameters and counteract system drift that scales to arbitrary numbers of qubits, that can be performed during computation with no additional overhead in time. The presented approach is in principle applicable to any code that repetitively detects errors using small groups of qubits. We implement our method on a superconducting nine-qubit device performing repetitive error detection demonstrating how parameters for single and two-qubit gates can be scalably optimized in parallel. Additionally, we show how independent parameter drifts on each qubit can be tracked and removed during computation. These results provide a path forward to controlling the large-scale qubit arrays needed for fault-tolerant quantum computation.

Finding and maintaining optimal control parameters of a continuously running quantum computer is of great interest as useful algorithms on future computers will likely require large arrays of qubits operating without fail for days at a time. An ideal optimal quantum control method would run in parallel throughout the computation, to track and compensate for the unavoidable drifts in the system [11]. Optimal quantum control [12] has a rich history in state transfer [13, 14], creating macroscopic quantum states [15, 16], optimizing quantum gates [17–20], and controlling many-qubit systems [21, 22]. Conventional methods such as tomography [23] and randomized benchmarking [24] use the final state of the system as a metric to evaluate the performance of a control sequence [25], and would require interruption of the necessary error detection algorithm. Thus, these methods do not extend to a continuously running quantum computer, as control is done on-the-fly, the qubits which store the quantum data ("data qubits") may not be measured, and errors introduced while exploring parameter space may lead to logical failure. Perhaps most importantly, real-time device performance can only be assessed in detection events, the outputs of error detection operators.

We present a method that uses detection events – the rate that errors occur – as a metric. Using codes where error detection and propagation is bounded, we can partition a system into qubit groupings that can be tuned independently. We feedback the error rate from a grouping to improve control parameters for gates contained within that grouping. By choosing finite patterns of groupings, we can independently optimize every control parameter of every qubit and retain $O(1)$ scaling with system size. Since we use error detection to inform our control parameters, we are guaranteed optimal performance, and there is no need to interrupt error detection to perform calibration. Additionally, the qubits do not need to be operating below threshold to use this technique. We call our method Active Detection Event Parameter Tuning (ADEPT).

We demonstrate ADEPT on a nine-qubit superconducting quantum processor. It consists of a thin superconducting film of Aluminum on sapphire, which is lithographically defined into a linear chain of Xmon transmon qubits [26, 27] with individual control and readout (Fig 1a). Single qubit rotations are performed with microwave pulses at the qubit frequency (4-6 GHz), and a current bias can be applied to bring neighboring qubits into resonance, enabling a controlled-Z (CZ) gate [4]. Measurement is achieved using dispersive readout [28–31].

For our error detection algorithm, we choose to work with the repetition code. The nearest neighbor coupling makes it a natural choice for our architecture, it has been experimentally demonstrated to operate below the threshold for error correction, and is the one-dimensional primitive of the two-dimensional surface code [6]. The repetition code detects bit-flip errors: here, entangling gates are used to copy bit errors on data qubits, which store the quantum state, onto neighboring ancilla measurement qubits (Fig 1b **i**, **ii**), where they can be detected. Bit errors on measurement qubits will change their state, but will not propagate back to data qubits (Fig 1b, **iii**). Errors are detected by repeatedly performing the error detection circuit, a $\hat{Z}\hat{Z}$ stabilizer, and analyzing the measured states of the measurement qubits (Fig 1c). In a future quantum computer, these operations would be running continuously without interruption. While this is technically challenging today,

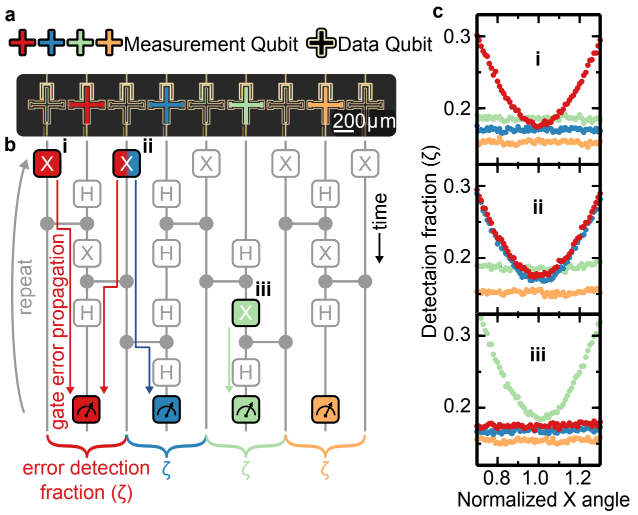

FIG. 1: (Color) **Error detection circuit and gate error detection.** (a) Linear chain of nine superconducting qubits with nearest neighbor coupling. (b) Error detection circuit for the bit-flip repetition code, where data qubits (black and gold) hold the quantum state, and measurement qubits (red (dark), blue (dark grey), green (light grey) and orange (grey)) are used to detect errors. Here we perform three experiments **i**-**iii**, where rotation angles are tuned from their optimal values and errors are recorded. Gate-induced bit errors on data qubits (**i** and **ii**) are copied to measurement qubits through CZ gates where they are detected as detection events, see text. Gate errors on measurement qubits (**iii**) are localized to those qubits. Hadamard gates are physically implemented with variable phase $\frac{\pi}{2}$ rotations. (c) Detection event fraction ζ vs variable angle X gates on data (**i** and **ii**) and measurement (**iii**) qubits. Each data point is the average of 6,000 instances of eight rounds of detection. Change in gate parameters from their ideal increases ζ. Change in ζ is localized to measurement qubits near the gate being varied, as ζ for unaffected qubits is constant.

we emulate the performance of N rounds of error detection by initializing the system into the logical $|0\rangle$ state (which has similar performance to the logical $|1\rangle$, see ref. 6), run eight rounds of error detection, and end the code. Then, we repeat this process to gather statistics and accumulate N total rounds of detection [32].

We now discuss how we process the measured qubit states into error detection events. In the presence of no error, measurement qubits will report a string of repeated or alternating states, depending on the states of the neighboring data qubits. In the presence of an error on the measurement qubit or a neighboring data qubit, the pattern of states will flip between the repeated or alternating pattern. This is known as a detection event, and indicates the presence of a nearby error (see ref 6 for more detail). The error rate of the system is thus directly related to the detection event fraction ζ, the fraction of measurements that are detection events. The ζ presented are consistent with the below threshold behavior demonstrated in ref 6.

As an experimental demonstration of the relation between gate errors and detection events, we have inserted error by tuning rotation angles away from the optimum for specific qubits. This error adds to the baseline ζ of the qubits that exists due to imperfect gates and coherence. Additionally, we see variation between ζ from inconsistencies in device coherence and operating conditions, consistent with Ref. 6. The results are shown in Fig 1c. The errors from a miscalibrated gate on data qubits (Fig 1c **i** and **ii**) are copied onto their neighboring measurement qubits (red (dark), and red/blue (dark/dark grey) respectively). Errors from miscalibrated gates on a measurement qubit are localized to that qubit (Fig 1c **iii**).

Note that there is a clear connection between the parameters of a gate and ζ of nearby measurement qubits. We see that away from the optimal rotation angle for an X gate (rotation around X-axis in the Bloch sphere representation), ζ increases, making it a natural metric to improve the gate parameters. Second, we see that the change in ζ is local: if the gate parameters are adjusted on a data qubit (i and ii), the dominant effect is on the neighboring measure qubit ζ. If the gate parameters on a measurement qubit are adjusted (iii), only that measurement qubit has a change in ζ. In both cases, unaffected qubits see no change.

Crucially, the direct, correspondence between detection events and gate errors on nearby qubits implies that we can tune gate parameters in parallel whenever the measurement qubits that pick up these gate errors do not overlap. This parallelization can be damaged due to crosstalk, as is visible In **ii** where we find some small crosstalk to the green (light grey) measure qubit. However, this is manageable and $O(1)$ scaling can be preserved as long as the effect is small and decays with distance[32].

In Figure 2, we demonstrate how ζ can be fed back to improve gate parameters in parallel. As a first experiment we optimize single qubit gates on all measurement qubits simultaneously (Figure 2a). Initial gate parameters were chosen using conventional methods, where each qubit was tuned in isolation. We use the ζ of each measurement qubit – calculated over $N = 36,000$ rounds of detection and constituting one "emulation step" – as an error metric for the Nelder-Mead optimization algorithm, and allow the algorithm to adjust the gate parameters such as amplitude and drive frequency independently on each measurement qubit (Figure 2b). We run this for all four measurement qubits simultaneously. After 50 emulation steps, we find that the ζ of each measurement qubit has been reduced, from an average of 0.202 to an average of 0.179.

In Figure 2c, we use the same technique to optimize CZ gate parameters performed between measurement and data qubits. Initial gate parameters were chosen by perturbing them away from the optimum. In this case, adjusting gate parameters will change the error rate on both data and measurement qubits involved in the CZ. The additional errors on the data qubit will be copied to both of its neighboring measurement qubits through the error detection circuit (Fig. 1 **ii**). Thus, adjusting CZ gate parameters will alter ζ of two measurement qubits. To ensure that our chosen error metrics do not overlap, we choose to optimize CZ gates that only involve every other measurement qubit, and take the average ζ of pairs of measurements qubits as the error metric for Nelder-Mead. After 80 emulations steps, we find a decrease in ζ from 0.350

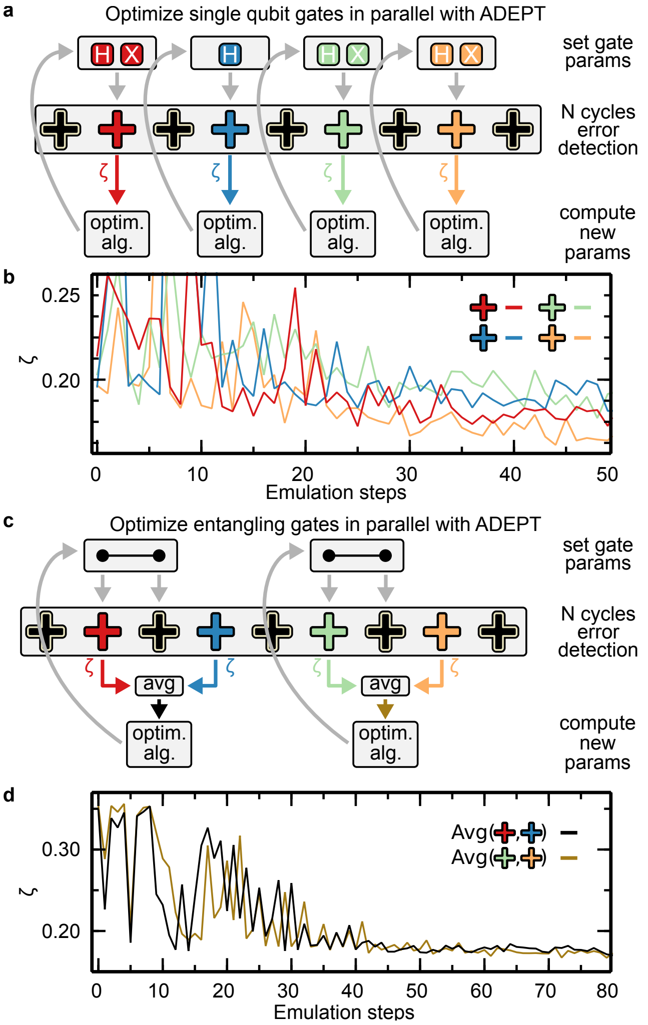

FIG. 2: (Color) **Parallel gate optimization using independent hardware groupings.** (a) Schematic of ADEPT for optimizing gates on measurement qubits. Gate parameters are chosen and error detection is run to determine ζ that is used as an error metric for the Nelder-Mead optimization algorithm, which chooses new gate parameters. (b) Parallel optimization of measurement qubit gate parameters. Each measurement qubit is optimized independently, and each iteration is 4,500 instances of eight rounds of detection. Average ζ is improved from 0.202 to 0.179. (c) Using ADEPT to optimize CZ gates. As CZ gates include data qubits, both neighboring measurement qubits will change ζ with changing gate parameters. To avoid error overlap, we optimize CZ that are well separated (see text). (d) Parallel Nelder-Mead optimization of two CZ gates. The metric is the average ζ of the measurement qubits that neighbor the data qubit in the CZ. Each iteration is 4,500 instances of eight rounds of detection. Average ζ is improved from 0.350 to 0.172.

to 0.172 (Fig. 2).

These data show that we can optimize gates in parallel without ever interrupting error detection. By extending these techniques we can experimentally determine (or theoretically derive for an ideal system [32]) a finite set of experiments that can be run to optimize every gate on every qubit while error detection is running [32]. As there are a finite number of experiments, ADEPT is fully parallelizable and has $O(1)$ scaling. We would like to point out that optimal qubit parameters will likely vary between physical qubits due to manufacturing variation, and that qubit parameters therefore need to be individually tuned. Our method will be able to perform this task, and scale to the arbitrary numbers of qubits in future processors running error detection.

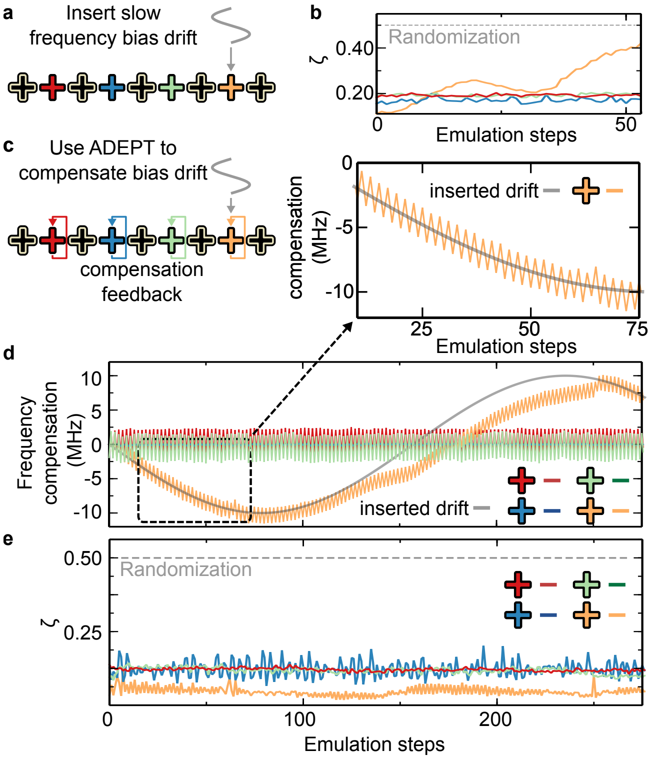

FIG. 3: (Color) **Tracking and negating frequency drift with ADEPT.** (a) A slow sinusoidally varying frequency drift of up to 10 MHz is inserted onto the orange (grey) measure qubit. (b) Uncorrected, this will cause ζ of the orange (grey) qubit to double within 20 iterations (with each iteration 12000 instances of eight rounds of detection), and eventually saturate near the randomization limit of 0.5 (c) The frequency drift can be compensated for by feeding ζ into a following algorithm. The algorithm samples points above and below the ideal bias value, and fits these points to a parabolic error model [32]. (d) The drift following algorithm tracks and compensates for the inserted frequency drift through one oscillation. The other algorithms produce no compensation, as no drift is inserted. (e) The measured ζ remain flat throughout the experiment, in contrast to (b).

We have shown ADEPT is adept at finding optimal control parameters, however in real physical systems these ideal parameters can change over time. Given that a future quantum computer will likely perform computations over hours or days [1], it is important that parameters remain optimal on this timescale.

In Figure 3, we show that ADEPT can be used to compensate for parameter drift. To emulate uncontrolled parameter drift, we insert a slowly varying voltage to the frequency bias of the orange (grey) qubit (Figure 3a), which will induce a large shift on the qubit frequency of up to ± 10 MHz and updates once each emulation step. We find that after just a 4 MHz shift over 20 rounds of emulation (equivalent to 84.2 ms of non-emulated operation given a 878 ns cycle time [32]), ζ for the orange (grey) qubit has more than doubled from 0.11 to 0.26. ζ eventually exceeds at 0.4 for a 8 MHz frequency error, indicating a near randomization of the measurement qubit output and failure to reliably detect errors. In Fig 3c, we now feedback ζ of each measurement qubit to a tracking algorithm that can adjust the frequency bias of that qubit by fitting ζ to a parabolic error model [32]. We find that the algorithm is able to use ζ as a metric to zero out the inserted frequency drift (Figure 3d). The added offset bias to the orange (grey) qubit follows the inserted bias, while the compensation for qubits without an inserted bias stays near zero or a constant value.

Importantly, we see that the ζ are stabilized in Figure 3e; the orange (grey) qubit stays at an average ζ of 0.12, and stays well below the randomization limit of 0.5. Thus, we show that ADEPT can be used to keep parameters for a single qubit near their ideal values. We note that the average ζ is slightly higher than the initial ζ of 0.11. This highlights a system tradeoff: in order to track the optimal value of a parameter we must sample away from the optimum of that parameter. We argue that paying this small price in error given that a future fault-tolerant computer should operate safely below threshold, and will provide a benefit in stability for long computations.

In Figure 4, we show that ADEPT can be used to compensate for individual parameter drift on every qubit simultaneously. To emulate what may happen in a real system, we insert frequency drifts for each qubit of ±10 MHz over 48 million emulated rounds of detection (equivalent to 42 seconds of non-emulated operation). We emphasize that this is a very large and fast drift compared to what is typically seen in hardware [10], making it an excellent stress test of our method. To compensate for independent drifts on each qubit, we cycle through the three different patterns, where in each pattern we optimize a subset of the qubits (Figure 4b). In pattern **i**, we use ζ of a measurement qubit as a metric its own frequency. In pattern **ii** and **iii** we optimize data qubits by using the average ζ of neighboring measurement qubits as a metric. By only adjusting the parameters relevant for each pattern at a time (Fig. 4c), we can compensate for an independent bias for each qubit (Fig. 4d). Using this strategy, we stabilize ζ of all measurement qubits – indicating that *all* qubits are adequately compensated for drift – and keep them well below the randomization limit (Fig. 4e). This demonstrates how ADEPT can be used to keep parameters near their optimum while running long algorithms.

We have introduced ADEPT in a one-dimensional chain of qubits running the repetition code, but this technique is generalizable to most error correction schemes. Any scheme that detects errors using groups of qubits of fixed maximum size, and the number of groups that any qubit belongs to does not scale with system size can use ADEPT. For example, this is compatible with all topological codes [33–41] including subsystem codes [42], and all concatenated codes [43–47] by focusing on the lowest level of concatenation. This includes surface [37] and color codes [38], and the Steane [45] and Shor [43] codes. ADEPT may not be compatible with finite rate block codes [48] if one wants to preserve $O(1)$ scaling with system size.

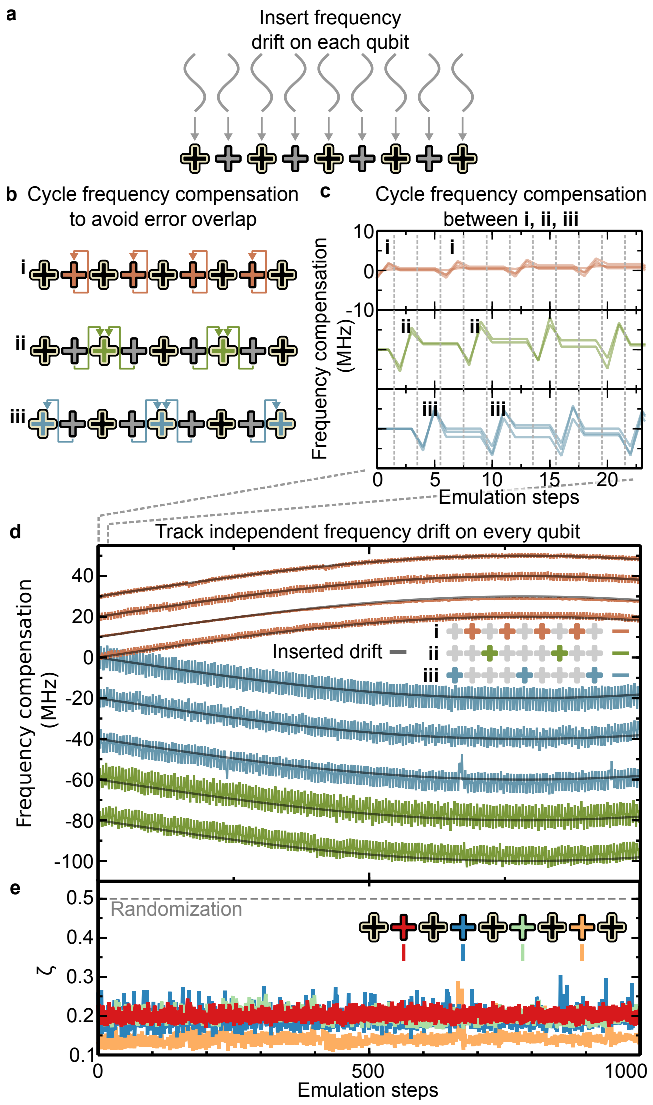

FIG. 4: (Color) **Tracking and negating independent frequency drift on all qubits.** (a) A different amplitude frequency drift is inserted onto each of nine qubits in the device. (b) ADEPT is used in conjunction with a frequency following algorithm on all qubits by interleaving the active hardware pattern. In **i**, ζ is used to inform the frequency following algorithm for the measurement qubits. In **ii** and **iii** ζ is used to inform the frequency following algorithm of neighboring data qubits. (c) Compensation algorithms associated with each hardware pattern **i-iii** are active one pattern a time in a sequential repeating fashion. (d) Independent frequency drift on each qubit is tracked independently. The traces have been spaced along the y-axis for viewing clarity. Each data point is 6000 instances of eight rounds of detection. (e) ζ is stabilized throughout the course of the experiment, indicated that all qubits have their frequency drift compensated for.

In a future quantum computer, ADEPT may be employed in the following manner. First, run error detection with no algorithm and use ADEPT to fine tune initial gate parameters. Second, use ADEPT to counteract parameters known to drift while an algorithm is running.

While ADEPT can be useful, it also has some limitations which are important to consider. It is challenging to relate ζ and the performance of error correction directly, and this may come with unexpected consequences. In principle, optimizing for ζ could trade off errors that are challenging to correct for which fewer detection events, for easily correctable errors with more detection events. ADEPT can find tradeoffs between gate fidelity and ζ, which may be an important consideration. Lastly, ADEPT is best suited for finding local minima, as the lowest achievable ζ is for measure qubits to never detect errors.

We have demonstrated the ADEPT control technique, which uses the error detection outcomes of operator measurements for system optimization. We have shown that this can be used to optimize gate parameters in a way that is $O(1)$ scalable to arbitrary numbers of qubits. Additionally, we have demonstrated that ADEPT can be used to compensate for system drift. By interleaving multiple hardware patterns, we can track parameter drift on every qubit, and potentially every parameter of every gate on every qubit, all without interruption of the error detection. With ADEPT, we are optimistic about controlling the many physical qubits that constitute a future fault-tolerant quantum computer.

Acknowledgements Devices were made at the UC Santa Barbara Nanofabrication Facility, a part of the US NSF-funded National Nanotechnology Infrastructure Network, and at the NanoStructures Cleanroom Facility.

Author Information Correspondence and requests for materials should be addressed to J.K. (juliankelly@google.com) or J.M.M. (jmartinis@google.com).

Appendix A: Determining patterns of independent parameter groups

1. Theoretical ideal case: Repetition code

We consider the repetition code algorithm in the case where gates are non-ideal and may generate errors, but only consider the case where gate errors occur on the same qubit as the gate is performed. We model an imperfect Clifford gate as an ideal gate followed by some probability of an error X, Y or Z gate. Using this treatment, we determine where gate errors from each gate in the algorithm will be detected in Fig. 7a. Single qubit gate errors on measurement qubits are detected on the same qubit. Single qubit gate errors on data qubits are detected on neighboring measurement qubits. Given a CZ between a measurement qubit and a data qubit, errors from that CZ will be detected on both measurement qubits neighboring the data qubit. With these propagations in mind, we can partition the system into different patterns, with each pattern containing multiple groupings. Each grouping contains gates and qubits: errors from gates within the grouping will not propagate to qubits outside of that grouping. Each grouping always contains measurement qubits; their detection fraction ζ is used as a metric for gate performance. Then, each grouping can have its constituent gates optimized independently, and all groupings in a pattern can be optimized in parallel.

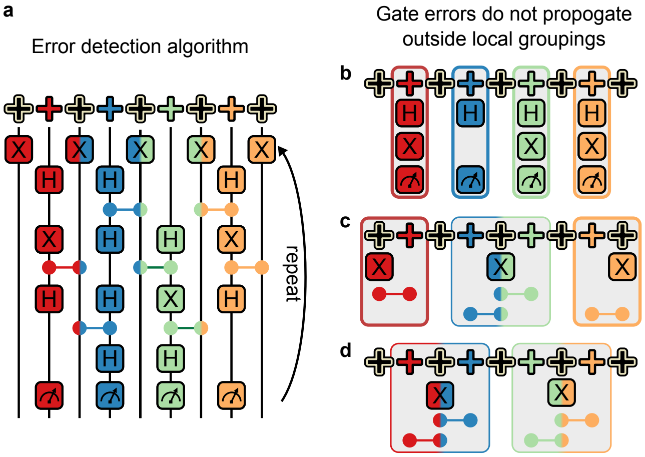

FIG. 5: **Hardware patterns and local groupings.** (a) Error detection circuit, where gate color corresponds to which measurement qubit will detect errors from that gate. (b) First hardware pattern, where each of four groupings contains a measurement qubit and the single-qubit operations for that qubit. Errors from these gates will not propagate back to neighboring qubits. The relative ζ from the measurement qubit can be used to inform changes in gate parameters. (c, d) Second and third hardware pattern, where each grouping contains one data qubit and up to two measurement qubits. Groupings contain single-qubit gates on measurement qubits and CZ gates. The ζ of measurement qubits within each grouping can be used to inform gate parameters for gates within the grouping.

The first hardware pattern is shown in Fig 7b, where each grouping contains a measurement qubit and the single qubit operations for that qubit. The second hardware pattern, shown in Fig 7c, contains every other data qubit, and the neighboring measurement qubits. These groupings contain single qubit gates on the data qubits, and the CZ gates that involve that data qubit. The third hardware pattern shown in Fig 7d is the same, with the compliment set of data qubits. Between the three pattern and their groupings, we can access all gates on all qubits shown in Fig. 7a.

To optimize all gates on all qubits, we would first (i) pick a hardware pattern, (ii) pick a gate to optimize within that hardware pattern and (iii) optimize that gate using ADEPT. We would then repeat i-iii with a different gate until we have covered all gates in all hardware patterns,

2. Experimental case

Any realistic system will deviate from the theoretically ideal case to some degree. For example, crosstalk may make gate parameters that should be local to one qubit effect the error rates of nearby qubits. However, this does not mean that ADEPT is not viable in such a system as long as crosstalk has a finite extent and is small enough. One can simply choose groups that contain qubits separated by the effective crosstalk distance. This will incur an additional overhead in the number of patterns used, but preserves $O(1)$ scaling with system size.

All non-idealities can be determined experimentally and algorithmically with a straightforward prescription. Assuming that a future large-scale system is composed of a cell of qubits with particular parameters (e.g. frequency, coupling) that is repeated throughout the computer, we can simply sweep each parameter of each gate of each qubit one at a time and record the corresponding change in error rate for the system. After going through all cases, and assuming that no parameter effects the entire system simultaneously, we will have a characteristic set of responses which can be used to determine independent patterns.

Appendix B: Emulating continuous error detection

Our current classical and quantum hardware is not yet suited for continuously running quantum error correction. In particular, our control software is not yet designed to update on-the-fly, and the qubits will likely require leakage reset to be able to operate with high fidelity for long numbers of rounds. However, we can still verify the underlying the principles behind ADEPT by emulating a continuously running system.

To simulate N rounds of continuously running error detection, we use an accumulation of experiments that each consist of eight rounds of error detection, see Fig. 6. We repeat many experiments, each with eight rounds of detections, until we have accumulated N error detection cycles in total.

It is of interest to estimate how fast a future continuously running quantum computer would be able to update parameters using ADEPT. We do this by quantifying "emulation" time; the timescale that a future computer will run at by subtracting out the time it takes to initialize, end and reset the code. For example, 36 "emulated" rounds of detection would take $\tau_{\text{emulated}} = 878\text{ns} \cdot 36 = 31.6\mu\text{s}$ on a future device, where our experiment would take $\tau_{\text{experiment}} = 878\text{ns} \cdot 36 + 3 \cdot (25\text{ns} + 1\mu\text{s} + 250\mu\text{s}) = 785\mu\text{s}$.

Appendix C: Bias-tracking algorithm

1. Quadratic error model

In principle, a variety of algorithms could be used to track the bias drift of the qubit, but we use a simple algorithm based on a parabolic error model, as it is easy to understand analytically. Our model assumes

$$\zeta = a(x - x_0)^2 + \zeta_0 \tag{C1}$$

where ζ is the instantaneous detection rate, ζ_0 is the background detection rate, a relates the bias error to ζ (determined prior to the experiment), x is the chosen bias parameter, and x_0 is the ideal bias value. Ideally, $x = x_0$. However, the ideal value can evolve in time from $x_0 \to x_1$, where x_0 becomes our most recent guess of the new ideal value x_1. We wish to determine $\delta x = x_1 - x_0$, the offset of our latest value to the ideal so that we can update our bias compensation.

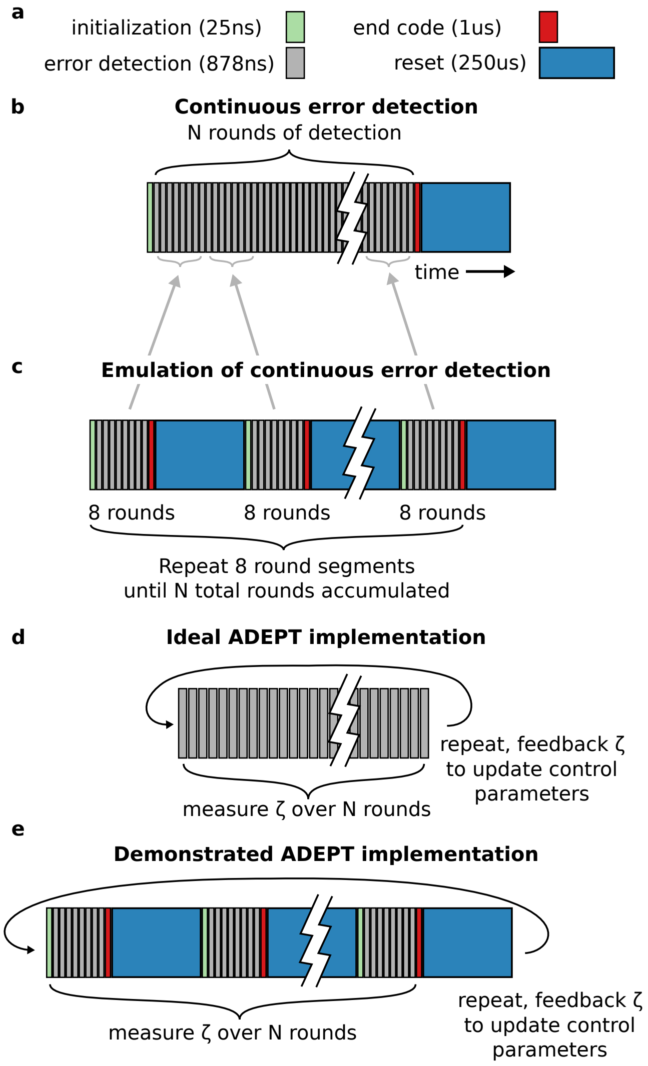

FIG. 6: **Emulation of continuosuly running error detection.** (a) Error detection experiments are composed of four steps: initialization, error detection, ending the code, and system reset. (b) N rounds of continuous error detection would involve initialization, N rounds of error detection, then ending the code and resetting the system. (c) We emulate continuous error detection by running an ensemble of experiments that each contain eight rounds of detection, such that we accumulate N total error detection rounds. (d) Ideally, ADEPT would run without ever needing to end the code of reset the system. Statistics over N rounds of detection would be used to calculate ζ, and then parameters would be updated on-the-fly. (e) We emulate this behavior by running ensembles of experiments to gather N rounds of error detection to compute ζ, and then updating parameters between ensembles. In main text Figures 3 and 4, the inserted bias is updated between ensembles of experiments.

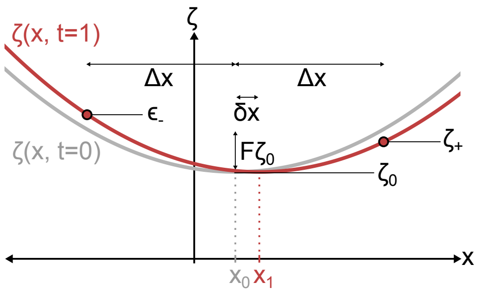

FIG. 7: **Parabolic error model.** Errors take the form of Eq. C1. After some time, the ideal parameter x_0 at $t = 0$ drifts to x_1 at $t = 1$. Detection fraction ζ is sampled at $x_0 \pm \Delta x$ defined in Eq. C3. ζ is increased by $F\zeta_0$ when using this sampling. After fitting to a parabolic error model, a new parameter x_1 is determined by adding δx as defined in Eq C6.

2. Where to sample

We wish to determine δ_x while also keeping the instantaneous ζ to the base detection rate ζ_0. In order to determine δx, we sample at points $x_0 \pm \Delta x$ and fit to a parabolic model. Importantly, sampling x away from the optimum will increase ζ, so there is a tradeoff in choosing Δx to produce a large enough signal, and keeping the instantaneous ζ of the qubits compared to the base error rate.

We aim to operate in the regime that $\Delta x \gg \delta x$, so let us set $\delta x = 0$ temporarily. suppose we choose to tolerate a fractional increase F in the base ζ_0. We wish to determine Δx, how far we should sample from our most recent optimum value x_0 while only incurring an additional detection of $F\zeta_0$,

$$\zeta = \zeta_0(1 + F) = a(\Delta x)^2 + \zeta_0 \tag{C2}$$

$$\Delta x = \sqrt{\zeta_0 F/a}. \tag{C3}$$

3. Updating the bias

Once we have sampled at $x = x_0 \pm \Delta$ giving us $\zeta_\pm$, we can determine δx.

$$\zeta_+ = a(\Delta x + x_0 - x_1)^2 + \zeta_0 \tag{C4}$$

$$\zeta_- = a(-\Delta x + x_0 - x_1)^2 + \zeta_0 \tag{C5}$$

$$\delta x = \frac{\zeta_- - \zeta_+}{4a\Delta x} \tag{C6}$$

4. Sampling statistics

As we sample for a finite time, we make imperfect measurements of $\zeta_\pm$. These will translate into noise in our determined δx parameter that we use to update the bias. We want to operate in the regime where for δx its standard error $SE_{\delta x}$ is much less than Δx, as we expect $\delta x \ll \Delta x$ for slow drift. Our goal is to accumulate enough statistics from sampling N times to achieve the condition for small relative noise $P \ll 1$.

$$P = \frac{SE_{\delta x}}{\Delta x} \tag{C7}$$

Given that $\zeta_\pm$ will be sampling the binomial distribution of detection fraction ζ, the standard error of the mean for $\zeta_\pm$ is

$$SE_{\zeta_\pm} = \frac{\sqrt{\zeta_\pm(1 - \zeta_\pm)}}{\sqrt{N}} \tag{C8}$$

where N is the number of experiments. Adding the standard deviations in quadrature for $\zeta_\pm$ and using Eq. C6 we find

$$SE_{\delta x} = \frac{\sqrt{2\zeta_\pm(1 - \zeta_\pm)}}{\sqrt{N}4a\Delta x} \tag{C9}$$

Solving Eq. C7 and Eq. C9 using Eq. C3 and $\zeta_\pm = \zeta_0(1 + F)$ one finds the condition for N

$$N \approx \frac{1}{8P^2F^2\zeta_0} \tag{C10}$$

Plugging in the relevant parameters $\zeta_0 = 0.15, F = 0.1$ and taking $P = 25$ we get $N \approx 50,000$ where as we used $N = 48,000$ for the experiment in main text Figure 4.

5. Sampling speed

Assuming the 1.1 MHz error detection rate, $N = 48,000$ measurements for each ζ measurement, two measurements per update, and three qubit patterns to cycle between as in main text Figure 4, frequency drift as fast as 0.3 Hz could be compensated for every qubit in a continuously running repetition code experiment.

[1] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, Phys. Rev. A **86**, 032324 (2012).

[2] J. M. Chow, J. M. Gambetta, A. Córcoles, S. T. Merkel, J. A. Smolin, C. Rigetti, S. Poletto, G. A. Keefe, M. B. Rothwell, J. Rozen, et al., Physical review letters **109**, 060501 (2012).

[3] T. Harty, D. Allcock, C. Ballance, L. Guidoni, H. Janacek, N. Linke, D. Stacey, and D. Lucas, Physical review letters **113**, 220501 (2014).

[4] R. Barends, J. Kelly, A. Megrant, A. Veitia, D. Sank, E. Jeffrey, T. White, J. Mutus, A. Fowler, B. Campbell, et al., Nature **508**, 500 (2014).

[5] C. Ballance, T. Harty, N. Linke, M. Sepiol, and D. Lucas, arXiv preprint arXiv:1512.04600 (2015).

[6] J. Kelly, R. Barends, A. Fowler, A. Megrant, E. Jeffrey, T. White, D. Sank, J. Mutus, B. Campbell, Y. Chen, et al., Nature **519**, 66 (2015).

[7] A. Córcoles, E. Magesan, S. J. Srinivasan, A. W. Cross, M. Steffen, J. M. Gambetta, and J. M. Chow, Nature communications **6** (2015).

[8] J. M. Chow, J. M. Gambetta, E. Magesan, D. W. Abraham, A. W. Cross, B. Johnson, N. A. Masluk, C. A. Ryan, J. A. Smolin, S. J. Srinivasan, et al., Nature communications **5** (2014).

[9] J. M. Gambetta, J. M. Chow, and M. Steffen, arXiv preprint arXiv:1510.04375 (2015).

[10] R. C. Bialczak, R. McDermott, M. Ansmann, M. Hofheinz, N. Katz, E. Lucero, M. Neeley, A. OConnell, H. Wang, A. Cleland, et al., Physical review letters **99**, 187006 (2007).

[11] C. Miquel, J. P. Paz, and W. H. Zurek, Physical review letters **78**, 3971 (1997).

[12] M. Shapiro and P. Brumer, Principles of the Quantum Control of Molecular Processes, by Moshe Shapiro, Paul Brumer, pp. 250. ISBN 0-471-24184-9. Wiley-VCH, February 2003. **1** (2003).

[13] N. Khaneja, T. Reiss, C. Kehlet, T. Schulte-Herbrüggen, and S. J. Glaser, Journal of Magnetic Resonance **172**, 296 (2005).

[14] C. J. Bardeen, V. V. Yakovlev, K. R. Wilson, S. D. Carpenter, P. M. Weber, and W. S. Warren, Chemical Physics Letters **280**, 151 (1997).

[15] C. P. Koch, J. P. Palao, R. Kosloff, and F. Masnou-Seeuws, Physical Review A **70**, 013402 (2004).

[16] U. Hohenester, P. K. Rekdal, A. Borzì, and J. Schmiedmayer, Physical Review A **75**, 023602 (2007).

[17] A. Spörl, T. Schulte-Herbrüggen, S. Glaser, V. Bergholm, M. Storcz, J. Ferber, and F. Wilhelm, Physical Review A **75**, 012302 (2007).

[18] J. Kelly, R. Barends, B. Campbell, Y. Chen, Z. Chen, B. Chiaro, A. Dunsworth, A. G. Fowler, I.-C. Hoi, E. Jeffrey, et al., Phys. Rev. Lett. **112**, 240504 (2014).

[19] D. Egger and F. Wilhelm, Physical review letters **112**, 240503 (2014).

[20] D. J. Egger and F. K. Wilhelm, Superconductor Science and Technology **27**, 014001 (2014).

[21] Y. Zhang, C. A. Ryan, R. Laflamme, and J. Baugh, Physical review letters **107**, 170503 (2011).

[22] J. Zhang, D. Gangloff, O. Moussa, and R. Laflamme, Physical Review A **84**, 034303 (2011).

[23] J. L. O'Brien, G. Pryde, A. Gilchrist, D. James, N. Langford, T. Ralph, and A. White, Physical review letters **93**, 080502 (2004).

[24] E. Magesan, J. M. Gambetta, and J. Emerson, Physical review letters **106**, 180504 (2011).

[25] T. Calarco, U. Dorner, P. Julienne, C. Williams, and P. Zoller, Physical Review A **70**, 012306 (2004).

[26] J. Koch, M. Y. Terri, J. Gambetta, A. A. Houck, D. Schuster, J. Majer, A. Blais, M. H. Devoret, S. M. Girvin, and R. J. Schoelkopf, Physical Review A **76**, 042319 (2007).

[27] R. Barends, J. Kelly, A. Megrant, D. Sank, E. Jeffrey, Y. Chen, Y. Yin, B. Chiaro, J. Mutus, C. Neill, et al., Phys. Rev. Lett. **111**, 080502 (2013).

[28] D. Schuster, A. Wallraff, A. Blais, L. Frunzio, R.-S. Huang, J. Majer, S. Girvin, and R. Schoelkopf, Physical review letters **94**, 123602 (2005).

[29] A. Wallraff, D. Schuster, A. Blais, L. Frunzio, J. Majer, M. Devoret, S. Girvin, and R. Schoelkopf, Physical Review Letters **95**, 060501 (2005).

[30] E. Jeffrey, D. Sank, J. Y. Mutus, T. C. White, J. Kelly, R. Barends, Y. Chen, Z. Chen, B. Chiaro, A. Dunsworth, et al., Phys. Rev. Lett. **112**, 190504 (2014).

[31] J. Y. Mutus, T. C. White, R. Barends, Y. Chen, Z. Chen, B. Chiaro, A. Dunsworth, E. Jeffrey, J. Kelly, A. Megrant, et al., Applied Physics Letters **104**, 263513 (2014).

[32] See supplementary information for discussion of ADEPT hardware patterns, emulating continuously running error detection, and details of the bias tracking algorithm.

[33] S. B. Bravyi and A. Y. Kitaev, quant-ph/9811052 (1998).

[34] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, J. Math. Phys. **43**, 4452 (2002), quant-ph/0110143.

[35] R. Raussendorf and J. Harrington, Phys. Rev. Lett. **98**, 190504 (2007), quant-ph/0610082.

[36] R. Raussendorf, J. Harrington, and K. Goyal, New J. Phys. **9**, 199 (2007), quant-ph/0703143.

[37] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, Phys. Rev. A **86**, 032324 (2012), arXiv:1208.0928.

[38] H. Bombin and M. A. Martin-Delgado, Phys. Rev. Lett. **97**, 180501 (2006), quant-ph/0605138.

[39] M. Ohzeki, Phys. Rev. E **80**, 011141 (2009), arXiv:0903.2102.

[40] H. G. Katzgraber, H. Bombin, R. S. Andrist, and M. A. Martin-Delgado, Phys. Rev. A **81**, 012319 (2010), arXiv:0910.0573.

[41] H. Bombin, Phys. Rev. X **5**, 031043 (2015), arXiv:1404.5504.

[42] R. S. Andrist, H. Bombin, H. G. Katzgraber, and M. A. Martin-Delgado, Phys. Rev. A **85**, 050302R (2012), arXiv:1204.1838.

[43] P. W. Shor, Phys. Rev. A **52**, R2493 (1995).

[44] A. R. Calderbank and P. W. Shor, Phys. Rev. A **54**, 1098 (1996), quant-ph/9512032.

[45] A. M. Steane, Proc. R. Soc. Lond. A **452**, 2551 (1996), quant-ph/9601029.

[46] E. Knill, Nature **434**, 39 (2005), quant-ph/0410199.

[47] D. Bacon, Phys. Rev. A **73**, 012340 (2006), quant-ph/0506023.

[48] D. Gottesman, arXiv:1310.2984 (2013).