Dynamical decoupling for superconducting qubits: A performance survey

Nic Ezzell,[1,2,*] Bibek Pokharel,[1,2,†] Lina Tewala,[3,4,‡] Gregory Quiroz,[3,5,§] and Daniel A. Lidar[1,2,6,7,¶]

[1] *Department of Physics and Astronomy, University of Southern California, Los Angeles, California 90089, USA*
[2] *Center for Quantum Information Science & Technology, University of Southern California, Los Angeles, California 90089, USA*
[3] *Johns Hopkins University Applied Physics Laboratory, Laurel, Maryland 20723, USA*
[4] *Thomas C. Jenkins Department of Biophysics, Johns Hopkins University Baltimore, Maryland 21218, USA*
[5] *William H. Miller III Department of Physics & Astronomy, Johns Hopkins University, Baltimore, Maryland 21218, USA*
[6] *Department of Electrical & Computer Engineering, University of Southern California, Los Angeles, California 90089, USA*
[7] *Department of Chemistry, University of Southern California, Los Angeles, California 90089, USA*

(Received 14 July 2022; revised 3 May 2023; accepted 16 November 2023; published 14 December 2023)

Dynamical decoupling (DD) is perhaps the simplest and least resource-intensive error-suppression strategy for improving quantum computer performance. Here we report on a large-scale survey of the performance of 60 different DD sequences from ten families, including basic as well as advanced sequences with high-order error cancelation properties and built-in robustness. The survey is performed using three different superconducting-qubit IBMQ devices, with the goal of assessing the relative performance of the different sequences in the setting of arbitrary quantum state preservation. We find that the high-order universally robust (UR) and quadratic DD (QDD) sequences generally outperform all other sequences across devices and pulse-interval settings. Surprisingly, we find that DD performance for basic sequences such as the Carr-Purcell-Meiboom-Gill and XY4 sequences can be made to nearly match that of UR and QDD sequences by optimizing the pulse interval, with the optimal interval being substantially larger than the minimum interval possible on each device.

DOI: 10.1103/PhysRevApplied.20.064027

I. INTRODUCTION

In the pre-fault-tolerance era, quantum computing research has two main near-term goals: to examine the promise of quantum computers via demonstrations of quantum algorithms [1–3] and to understand how quantum error correction and other noise mitigation methods can pave a path towards fault-tolerant quantum computers [4,5]. The last decade has seen the rise of multiple cloud-based quantum computing platforms that allow a community of researchers to test error-suppression and error-correction techniques [6–16]. Error suppression using dynamical decoupling (DD) [17–21] is among the earliest methods to have been experimentally demonstrated, using experimental platforms such as trapped ions [22,23], photonic qubits [24], electron paramagnetic resonance [25], nuclear magnetic resonance (NMR) [26–28], trapped atoms [29], and nitrogen vacancies in diamond [30]. It is known that DD can be used to improve the fidelity of quantum computation both without [31–38] and with quantum error correction [39,40]. Several recent cloud-based demonstrations have shown that DD can unequivocally improve the performance of superconducting-qubit-based devices [41–46], even leading to algorithmic quantum advantage [47].

In this work, we systematically compare a suite of known and increasingly elaborate DD sequences developed over the past two decades (see Table I for a complete list and Ref. [62] for a detailed review). These DD sequences reflect a growing understanding of how to build features that suppress noise to increasingly higher order and with greater robustness to pulse imperfections. Our goal is to study the efficacy of the older and the more recent advanced sequences on currently available quantum computers. To this end, we implement these sequences on three different IBM Quantum Experience (IBMQE)

*nezzell@usc.edu
†pokharel@usc.edu
‡lina.tewala@emory.edu
§gregory.quiroz@jhuapl.edu
¶lidar@usc.edu

 © 2023 American Physical Society

TABLE I. Summary of the DD sequences surveyed in this work, along with the original references. A sequence has a uniform (pulse) interval provided $\tau_i = \tau_j$ for all i,j [see Eq. (7)] and is nonuniform otherwise. A sequence is universal (in theory) if it cancels an arbitrary H_{err} [Eq. (2)] to first order in the pulse interval, and practically this means that it protects all states equally well. Otherwise, it only protects a subset of states (e.g., CPMG, which only protects $|\pm\rangle$). In practice, this distinction is more subtle due to rotating frame effects, as discussed in Sec. II D. For those listed as both Y & N, such as RGA_n, we mean that not all sequences in the family are universal. For example, RGA_2 is not universal but RGA_n for $n \geq 4$ is universal. The last column lists whether our eventual implementation requires OpenPulse [48] or can be implemented faithfully with just the traditional circuit API [49,50] (see Appendix B).

Sequence	Uniform interval	Universal	Needs OpenPulse
Hahn echo [51]	Y	N	N
PX or CPMG [52,53]	Y	N	N
XY4 [54]	Y	Y	Y
CDD_n [55]	Y	Y	Y
EDD [56]	Y	Y & N	Y
RGA_n [57]	Y	Y & N	Y
KDD [58]	Y	Y	Y
UR_n [59]	Y	Y	Y
UDDx_n [60]	N	N	N
$\text{QDD}_{n,m}$ [61]	N	Y	Y

transmon qubit-based platforms: ibmq_armonk (Armonk), ibmq_bogota (Bogota), and ibmq_jakarta (Jakarta). We rely on the open-pulse functionality [63] of IBMQE, which enables us to precisely control the pulses and their timing. The circuit-level implementation of the various sequences can be suboptimal, as we detail in the Appendix B.

We assess these DD sequences for their ability to preserve an arbitrary single-qubit state. Previous work, focused on the XY4 sequence, has studied the use of DD to improve two-qubit entanglement [41] and the fidelity of two-qubit gates [44], and we leave a systematic survey of the multiqubit problem for a future publication, given that the single-qubit case is already a rich and intricate topic, as we discuss below. By and large, we find that all DD sequences outperform the "unprotected" evolution (without DD). The higher-order DD sequences, like concatenated DD (CDD [55]), Uhrig DD (UDD [60]), quadratic DD (QDD [61]), nested UDD (NUDD [64]), and universally robust (UR [59]), perform consistently well across devices and pulse placement settings. While these more elaborate sequences are statistically better than the traditional sequences such as Hahn echo [51], Carr-Purcell-Meiboom-Gill (CPMG), and XY4 [54] for short pulse intervals, their advantage diminishes with sparser pulse placement. As both systematic and random errors, e.g., due to finite pulse width and limited control, are reduced, advanced sequences will likely provide further performance improvements. Overall, our study indicates that the robust DD sequences can be viewed as the preferred choice over their traditional counterparts.

The structure of this paper is as follows. In Sec. II we review the pertinent DD background and describe the various pulse sequences we tested. In Sec. III, we detail the cloud-based demonstration setup, the nuances of DD sequence implementation, and the chosen success metrics. We describe the results and what we learned about the sequences and devices in Sec. IV. A summary of results and possible future research directions are provided in Sec. V. Additional details are provided in the appendices.

II. DYNAMICAL DECOUPLING BACKGROUND

For completeness, we first provide a brief review of DD. In this section we focus on a small subset of all sequences studied in this work, primarily to introduce key concepts and notation. The details of all the other sequences are provided in Appendix A. The reader who is already an expert in the theory may wish to skim this section to become familiar with our notation.

A. DD with perfect pulses

Consider a time-independent Hamiltonian

$$H = H_S + H_B + H_{SB}, \tag{1}$$

where H_S and H_B contain terms that respectively act only on the system or the bath, and H_{SB} contains the system-bath interactions. We write $H_S = H_S^0 + H_S^1$, where H_S^1 represents an undesired, always-on term (e.g., due to crosstalk), so that

$$H_{\text{err}} = H_S^1 + H_{SB} \tag{2}$$

represents the "error Hamiltonian" we wish to remove using DD. Hamiltonian H_S^0 contains all the terms we wish to keep. The corresponding *free* unitary evolution for duration τ is given by

$$f_\tau \equiv U(\tau) = \exp(-i\tau H). \tag{3}$$

DD is generated by an additional, time-dependent control Hamiltonian $H_c(t)$ acting purely on the system, so that the total Hamiltonian is

$$H(t) = H_S^0 + H_{\text{err}} + H_B + H_c(t). \tag{4}$$

An "ideal" or "perfect" pulse sequence is generated by a control Hamiltonian that is a sum of error-free, instantaneous Hamiltonians $\{\Omega_0 H_{P_k}\}_{k=1}^n$ that generate the pulses at

corresponding intervals $\{\tau_k\}_{k=1}^n$:

$$\hat{H}_c(t) = \Omega_0 \sum_{k=1}^{n} \delta(t - t_k) H_{P_k}, \qquad t_k = \sum_{j=1}^{k} \tau_j. \quad (5)$$

Here we use the hat notation to denote ideal conditions and let Ω_0 have units of energy. Choosing Ω_0 such that $\Omega_0 \Delta = \pi/2$, where Δ is the "width" of the Dirac-delta function (this is made rigorous when we account for pulse width in Sec. II B 1 below), this gives rise to instantaneous unitaries or pulses

$$\hat{P}_k = e^{-i\pi H_{P_k}/2}, \quad (6)$$

so that the total evolution is

$$\bar{U}(T) = f_{\tau_n} \hat{P}_n \cdots f_{\tau_2} \hat{P}_2 f_{\tau_1} \hat{P}_1, \quad (7)$$

where $T \equiv t_n = \sum_{j=1}^{n} \tau_j$ is the total sequence time. The unitary $\bar{U}(T) = U_0(T)B(T)$ can be decomposed into the desired error-free evolution $U_0(T) = \exp\left(-iTH_S^0\right) \otimes I_B$ and the unitary error $B(T)$. Ideally, $B(T) = I_S \otimes e^{-i\bar{T}\bar{B}}$, where $\bar{B}$ is an arbitrary Hermitian bath operator. Hence, by applying N repetitions of an ideal DD sequence of duration T, the system stroboscopically decouples from the bath at uniform intervals $T_j = jT$ for $j = 1, \ldots, N$. In reality, we only achieve approximate decoupling, so that $B(T) = I_S \otimes e^{-i\bar{T}\bar{B}} + \text{err}$, and the history of DD design is motivated by making the error term as small as possible under different and increasingly more realistic physical scenarios.

1. First-order protection

Historically, the first observation of stroboscopic decoupling came from NMR spin echoes observed by Erwin Hahn in 1950 [51] with a single X pulse [65]. Several years later, Carr and Purcell [52] and Meiboom and Gill [66] independently proposed the improved CPMG sequence with two X pulses. In theory, both sequences are only capable of partial decoupling in the ideal pulse limit. In particular, $B(T) \approx I_S \otimes e^{-i\bar{T}\bar{B}}$ only for states near $|\pm\rangle = (|0\rangle \pm |1\rangle)/\sqrt{2}$ (where $|0\rangle$ and $|1\rangle$ are the $+1$ and -1 eigenstates of σ^z, respectively), as we explain below. Nearly four decades after Hahn's work, Maudsley proposed the XY4 sequence [54], which is *universal* since $B(T) \approx I_S \otimes e^{-i\bar{T}\bar{B}}$ on the full Hilbert space, which means that all states are equally protected. Equivalently, universality means that arbitrary single-qubit interactions with the bath are decoupled to first order in τ.

To make this discussion more precise, we first write $H_B + H_{SB}$ in a generic way for a single qubit:

$$\overline{H} \equiv H_B + H_{SB} = \sum_{\alpha=0}^{3} \gamma_\alpha \sigma^\alpha \otimes B^\alpha \quad (8)$$

with the B^α bath terms and $\sigma^{(0)} = I$.

Since distinct Pauli operators anticommute, i.e., $\{\sigma_i, \sigma_j\} = 2I\delta_{ij}$, then, for $k \neq 0$,

$$\sigma_k \overline{H} \sigma_k = -\sum_{\alpha \neq k} \gamma_\alpha \sigma^\alpha \otimes B^\alpha + \gamma_k \sigma_k \otimes B_k. \quad (9)$$

The minus sign is an effective time reversal of the terms that anticommute with σ_k. In the ideal pulse limit, this is enough to show that "pure-X," defined as

$$\text{PX} \equiv X - f_\tau - X - f_\tau, \quad (10)$$

induces an effective error Hamiltonian

$$H_{\text{PX}}^{\text{eff}} = \gamma_x \sigma^x \otimes B^x + I_S \otimes \bar{B} + \mathcal{O}(\tau^2) \quad (11)$$

every 2τ. Note that the CPMG sequence is defined similarly, i.e.,

$$\text{CPMG} \equiv f_{\tau/2} - X - f_\tau - X - f_{\tau/2}, \quad (12)$$

which is just a symmetrized placement of the pulse intervals; see Sec. II C below. The PX and CPMG sequences have the same properties in the ideal pulse limit, but we choose to begin with the PX sequence for simplicity of presentation. Intuitively, the middle $X - f_\tau - X$ is a time-reversed evolution of the $\sigma^{(y,z)}$ terms, followed by a forward evolution, which cancel to first order in τ using the Zassenhaus formula [67], $\exp\{\tau(A+B)\} = e^{\tau A} e^{\tau B} + \mathcal{O}(\tau^2)$, an expansion that is closely related to the familiar Baker-Campbell-Hausdorff formula. The undesired noise term $\gamma_x \sigma^x \otimes B^x$ does not decohere $|\pm\rangle$, but all other states are subject to bit-flip noise in the absence of suppression. By adding a second rotation around y, the XY4 sequence,

$$\text{XY4} \equiv Y - f_\tau - X - f_\tau - Y - f_\tau - X - f_\tau, \quad (13)$$

cancels the remaining σ^x term and achieves universal (first-order) decoupling at time 4τ:

$$H_{\text{XY4}}^{\text{eff}} = I_S \otimes \bar{B} + \mathcal{O}(\tau^2). \quad (14)$$

Practically, this means that all single-qubit states are equally protected to first order. These results can be generalized by viewing DD as a symmetrization procedure [68], with an intuitive geometrical interpretation wherein the pulses replace the original error Hamiltonian by a sequence of Hamiltonians that are arranged symmetrically so that their average cancels out [69].

2. *Higher-order protection*

While the XY4 sequence is universal for qubits, it only provides first-order protection. A great deal of effort has been invested in developing DD sequences that provide higher-order protection. We start with concatenated dynamical decoupling, or CDD_n [55]. CDD_n is an nth-order recursion of the XY4 sequence [70]. For example, $\mathrm{CDD}_1 \equiv \mathrm{XY4}$ is the base case, and

$$\begin{aligned}\mathrm{CDD}_n &\equiv \mathrm{XY4}([\mathrm{CDD}_{n-1}]) \\ &= Y - [\mathrm{CDD}_{n-1}] - X - [\mathrm{CDD}_{n-1}] - Y \\ &\quad - [\mathrm{CDD}_{n-1}] - X - [\mathrm{CDD}_{n-1}],\end{aligned} \tag{15}$$

which is just the definition of the XY4 sequence in Eq. (13) with every f_τ replaced by CDD_{n-1}. This recursive structure leads to an improved error term $\mathcal{O}(\tau^{n+1})$ provided τ is "small enough." To make this point precise, we must define a measure of error under DD. Following Ref. [39], one useful way to do this is to separate the "good" and "bad" parts of the joint system-bath evolution, i.e., to split $\tilde{U}(T)$ [Eq. (7)] as

$$\tilde{U}(T) = \mathcal{G} + \mathcal{B}, \tag{16}$$

where $\mathcal{G} = U_0(T) \otimes B'(T)$, and where—as above—$U_0(T)$ is the ideal operation that would be applied to the system in the absence of noise, and $B'(T)$ is a unitary transformation acting purely on the bath. Operator $\mathcal{B}$ is the "bad" part, i.e., the deviation of $\tilde{U}(T)$ from the ideal operation. The error measure is then [71]

$$\eta_{\mathrm{DD}} = \|\mathcal{B}\|. \tag{17}$$

Put simply, η_{DD} measures how far the DD-protected evolution $\tilde{U}(T)$ is from the ideal evolution $\mathcal{G}$. With this error measure established, we can bound the performance of various DD sequences in terms of the relevant energy scales:

$$\beta \equiv \|H_B\|, \qquad J \equiv \|H_{SB}\|, \qquad \epsilon \equiv \beta + J. \tag{18}$$

Using these definitions, we can replace the coarse $\mathcal{O}$ estimates with rigorous upper bounds on η_{DD}. In particular, as shown in Ref. [39],

$$\eta_{\mathrm{XY4}} = (4J\tau)\left[\tfrac{1}{2}(4\epsilon\tau) + \tfrac{2}{9}(4\epsilon\tau)^2\right] + \mathcal{O}(\tau^3), \tag{19a}$$

$$\eta_{\mathrm{CDD}_n} = 4^{n(n+3)/2}(c\epsilon\tau)^n(J\tau) + \mathcal{O}(\tau^{n+2}), \tag{19b}$$

where c is a constant of order 1. This more careful analysis implies that (1) $\epsilon\tau \lesssim 1$ is sufficient for the XY4 sequence to provide error suppression and that (2) CDD_n has an optimal concatenation level induced by the competition between taking longer (the bad approximate 4^{n^2} scaling) and more error suppression [the good $(c\epsilon\tau)^n$ scaling]. The corresponding optimal concatenation level is

$$n_{\mathrm{opt}} = \lfloor \log_4(1/\bar{c}\epsilon\tau) - 1 \rfloor, \tag{20}$$

where $\lfloor\cdot\rfloor$ is the floor function and $\bar{c}$ is another constant of order 1 [defined in Eq. (165) of Ref. [39]]. That such a saturation in performance should occur is fairly intuitive. By adding more layers of recursion, we suppress noise that was unsuppressed before. However, at the same time, we introduce more periods of free evolution f_τ that cumulatively add up to more noise. At some point, the noise wins since there is no active noise removal in an open loop procedure such as DD.

Though CDD_n derived from recursive symmetrization allows for $\mathcal{O}(\tau^{n+1})$ suppression, it employs approximately 4^n pulses. One may ask whether a shorter sequence could achieve the same goal. The answer is provided by the UDD sequence [60]. The idea is to find which DD sequence acts as an optimal filter function on the noise-spectral density of the bath while relaxing the constraint of uniform pulse intervals [22,60,72,73]. For a brief overview, we first assume that a qubit state decoheres as $e^{-\chi(t)}$. For a given noise spectral density $S(\omega)$,

$$\chi(t) = \frac{2}{\pi}\int_0^\infty \frac{S(\omega)}{\omega^2}F(\omega t)d\omega, \tag{21}$$

where the frequency response of the system to DD is captured by the filter function $F(\omega t)$. For example, for n ideal π pulses executed at times $\{t_j\}$ [74],

$$F_n(\omega\tau) = \left|1 + (-1)^{n+1}e^{i\omega\tau} + 2\sum_{j=1}^{n}(-1)^j e^{i\omega t_j}\right|^2, \tag{22}$$

which can be substituted into Eq. (21) and optimized for $\{t_j\}$; the result is UDD [60]. For a desired total evolution T, the solution (and definition of UDD) is simply to place π pulses with *nonuniform* pulse intervals,

$$t_j = T\sin\left(\frac{j\pi}{2(n+1)}\right)^2. \tag{23}$$

When we use n X-type pulses in particular, we obtain UDDx_n. It turns out that UDDx_n achieves $\mathcal{O}(\tau^n)$ suppression for states near $|\pm\rangle$ using only n pulses, and this is the minimum number of π pulses needed [60,75]. It is in this sense that UDD is provably optimal. However, it is important to note that this assumes that the total sequence time is fixed; only in this case can the optimal sequence be used to make the distance between the protected and unperturbed qubit states arbitrarily small in the number of applied pulses. On the other hand, if the minimum pulse interval is fixed and the total sequence time is allowed to

scale with the number of pulses, then—as in CDD—longer sequences need not always be advantageous [76].

UDD can be improved from a single-axis sequence to the universal QDD sequence [61,75,77] using recursive design principles similar to those that lead to the XY4 sequence and eventually CDD_n from the PX sequence. Namely, to achieve universal decoupling, we use a recursive embedding of a UDDy_m sequence into a UDDx_n sequence. Each X pulse in UDDx_n is separated by a free evolution period $f_{t_{j+1}-t_j}$ that can be filled with a UDDy_m sequence. Hence, we can achieve $\min\{\tau^n, \tau^m\}$ universal decoupling, and when $m = n$, we obtain universal order τ^n decoupling using only n^2 pulses instead of the approximately 4^n in CDD_n. This is nearly optimal [61], and an exponential improvement over CDD_n. When $m \neq n$, the exact decoupling properties are more complicated [75]. Similar comments as for UDDx_n regarding the difference between a fixed total sequence time T versus a fixed minimum pulse interval apply for QDD as well [78].

While QDD is universal and near optimal for single-qubit decoherence, the ultimate recursive generalization of UDD is NUDD [64], which applies for general multi-qubit decoherence, and whose universality and suppression properties have been proven and analyzed in a number of works [78–80]. In the simplest setting, suppression to Nth order of general decoherence afflicting an m-qubit system requires $(N+1)^{2m}$ pulses under NUDD.

B. DD with imperfect pulses

So far, we have reviewed DD theory with ideal pulses. An ideal pulse is instantaneous and error-free, but in reality, finite bandwidth constraints and control errors matter. Much of the work since CPMG has been concerned with (1) accounting for finite pulse width, (2) mitigating errors induced by finite width, and (3) mitigating systematic errors such as over- or under-rotations. We address these concerns in order.

1. Accounting for finite pulse width

During a finite width pulse, P_j, the effect of $H_{\mathrm{err}} + H_B$ cannot be ignored, so the analysis of Sec. II A needs to be modified correspondingly. Nevertheless, both the symmetrization and filter function approaches can be augmented to account for finite pulse width.

We may write a realistic DD sequence with Δ-width pulses just as in Eq. (7), but with the ideal control Hamiltonian replaced by

$$\hat{H}_c(t) = \sum_k \Omega(t-\tau_k) H_{P_k}, \quad \int_{-\Delta/2}^{\Delta/2} \Omega(t)dt = \Omega_0 \Delta = \frac{\pi}{2}, \tag{24}$$

where $\Omega(t)$ is sharply (but not infinitely) peaked at $t = 0$ and vanishes for $|t| > \Delta/2$. The corresponding DD pulses are of the form

$$P_k \equiv \exp\left\{-i \int_{\tau_k-\Delta/2}^{\tau_k+\Delta/2} dt[\Omega(t-\tau_k)H_{P_k} + H_{\mathrm{err}} + H_B]\right\}. \tag{25}$$

Note that the pulse intervals remain τ_k as before, now denoting the peak-to-peak interval; the total sequence time therefore remains $T = \sum_{j=k}^{n} \tau_k$. The ideal pulse limit of Eq. (7) is obtained by taking the pulse width to zero, so that $H_{\mathrm{err}} + H_B$ can be ignored:

$$\hat{P}_k = \lim_{\Delta\to 0} P_k = e^{-i\pi H_{P_k}/2}, \qquad \lim_{\Delta\to 0} \Omega(t) = \Omega_0 \delta(t). \tag{26}$$

We can then recover a result similar to Eq. (9) by entering the toggling frame with respect to the control Hamiltonian $H_c(t)$ (see Appendix D), and computing η_{DD} with a Magnus expansion or Dyson series [39]. Though the analysis is involved, the final result is straightforward: η_{DD} picks up an additional dependence on the pulse width Δ. For example, Eq. (19a) is modified to

$$\eta_{\mathrm{XY4}}^{(\Delta)} = 4J\Delta + \eta_{\mathrm{XY4}}, \tag{27}$$

which now has a linear dependence on Δ. This new dependence is fairly generic, i.e., the previously discussed PX, XY4, CDD_n, UDD_n, and $\mathrm{QDD}_{n,m}$ sequences all have an error η with an additive $\mathcal{O}(\Delta)$ dependence. Nevertheless, (1) DD is still effective provided $J\Delta \ll 1$ and (2) concatenation to order n is still effective provided the $J\Delta$ dependence does not dominate the $\epsilon\tau^n$ dependence. For CDD_n, this amounts to an effective noise strength floor [39],

$$\eta_{\mathrm{CDD}_n} \geq 16\Delta J, \tag{28}$$

which modifies the optimal concatenation level n_{opt}.

2. Mitigating errors induced by finite width

A natural question is to what extent we can suppress this first-order $\mathcal{O}(\Delta)$ dependence. One solution is Eulerian symmetrization [81], which exhibits robustness to pulse-width errors [56,82,83]. For example, the palindromic sequence

$$\mathrm{EDD} \equiv Xf_\tau Yf_\tau Xf_\tau Yf_\tau Yf_\tau Xf_\tau Yf_\tau Xf_\tau, \tag{29}$$

which is an example of Eulerian DD (EDD), has error term [39]

$$\eta_{\mathrm{EDD}} = (8J\tau)\left[\tfrac{1}{2}(8\epsilon\tau) + \tfrac{2}{9}(8\epsilon\tau)^2 + \mathcal{O}(\tau^3)\right], \tag{30}$$

which contains no first-order $\mathcal{O}(\Delta)$ term. Nevertheless, the constant factors are twice as large compared to the XY4 sequence, and it turns out that the EDD sequence outperforms the XY4 sequence when $\Delta/\tau \gtrsim 8\epsilon\tau$ (see Fig. 9 of

Ref. [39]) [84]. The same Eulerian approach can be used to derive the pulse-width robust version of the Hahn-echo and CPMG sequences, which we refer to with a "super" prefix (derived from the "Eulerian supercycle" terminology of Ref. [83]):

$$\text{super-Hahn} \equiv X f_\tau \overline{X} f_\tau, \tag{31a}$$

$$\text{super-CPMG} \equiv X f_\tau X f_\tau \overline{X} f_\tau \overline{X} f_\tau, \tag{31b}$$

with

$$\overline{P}_k \equiv \exp\left\{-i\int_{\tau_k-\Delta/2}^{\tau_k+\Delta/2} dt[-\Omega(t-\tau_k)H_{P_k} + H_{\text{err}} + H_B]\right\} \tag{32}$$

[compare to Eq. (25)]. Intuitively, if X is a finite pulse that generates a rotation about the x axis of the Bloch sphere then $\overline{X}$ is (approximately) a rotation about the $-x$ axis, i.e., with opposite orientation.

These robust sequences, coupled with concatenation, suggest that we can eliminate the effect of pulse width to arbitrary order $\mathcal{O}(\Delta^n)$; up to certain caveats, this indeed holds with concatenated dynamically corrected gates (CDCGs) [85] (see also Ref. [86]) [87]. However, this approach deviates significantly from the sequences consisting of only π rotations we have considered so far. To our knowledge, no strategy better than EDD exists for sequences consisting of only π pulses [82].

3. Mitigating systematic errors

In addition to finite width errors, real pulses are also subject to systematic errors. For example, $\Omega(t)$ might be slightly miscalibrated, leading to a systematic over- or under-rotation, and any aforementioned gain might be lost due to the accumulation of these errors. A useful model of pulses subject to systematic errors is

$$P_j^r = \exp\left\{\pm i\frac{\pi}{2}(1+\epsilon_r)\sigma^\alpha\right\} \tag{33}$$

for $\alpha \in \{x,y,z\}$. This represents instantaneous X, Y, Z and $\overline{X}, \overline{Y}, \overline{Z}$ pulses subject to systematic over- or under-rotation by ϵ_r, also known as a *flip-angle error*. Another type of systematic control error is *axis misspecification*, where instead of the intended σ^α in Eq. (33) a linear combination of the form $\sigma^\alpha + \epsilon_\beta\sigma^\beta + \epsilon_\gamma\sigma^\gamma$ is implemented, with $\epsilon_\beta, \epsilon_\gamma \ll 1$ and $\alpha \neq \beta \neq \gamma$ denoting orthogonal axes [30].

Fortunately, even simple π pulses can mitigate systematic errors if rotation axes other than $+x$ and $+y$ are used. We consider three types of sequence: robust genetic algorithm (RGA) DD [57], Knill DD (KDD) [58,88], and UR DD [59].

The RGA-DD sequence. The basic idea of RGA is as follows. A universal DD sequence should satisfy $\prod_{k=1}^n P_k = I$ up to a global phase, but there is a great deal of freedom in what combination of pulses are used that satisfy this constraint. In Ref. [57], this freedom was exploited to find, by numerical optimization with genetic algorithms, a class of sequences robust to over- or under-rotations.

Subject to a generic single-qubit error Hamiltonian as in Eq. (8), optimal DD sequences were then found for a given number of pulses under different parameter regimes (i.e., the relative magnitudes of J, β, etc.). This numerical optimization "rediscovered" the CPMG, XY4, and Eulerian DD sequences as base sequences with $\mathcal{O}(\tau^2)$ errors. Higher-order sequences were then found to be concatenations of the latter. For example,

$$\text{RGA}_{8c} \equiv \text{EDD}, \tag{34a}$$

$$\text{RGA}_{64c} \equiv \text{RGA}_{8c}[\text{RGA}_{8c}]. \tag{34b}$$

A total of 12 RGA sequences were found in total; more details are given in Appendix A.

The KDD sequence. This sequence is similar in its goal to RGA because it mitigates systematic over- or under-rotations. In design, however, it uses the principle of composite pulses [89–91]. The idea is to take a universal sequence such as XY4 and replace each pulse P_j with a composite series of pulses $(CP)_j$ that have the same effect as P_j but remove pulse imperfections by self-averaging; for details, see Appendix A.

The KDD sequence is robust to flip-angle errors [Eq. (33)]. For example, suppose that $\epsilon_r = \pi/20$ and that we apply idealized KDD ten times [which we denote by $(\text{KDD})^{10}$]. Then, by direct calculation, $\|(\text{KDD})^{10} - I\| \approx 7\times10^{-7}$, whereas $\|(\text{XY4})^{10} - I\| \approx 3\times10^{-2}$, and in fact, KDD is robust up to $\mathcal{O}(\epsilon_r^5)$ [59,62]. This robustness to over-rotations comes at the cost of 20 free evolution periods instead of 4, so we only expect KDD to work well in an ϵ_r-dominated parameter regime. As a preview of the results we report in Sec. IV, KDD is not among the top-performing sequences. Hence, it appears reasonable to conclude that our demonstrations are conducted in a regime that is *not* ϵ_r dominated.

The UR-DD sequence An alternative approach to devise robust DD sequences is the UR_n-DD family [59], developed for a semiclassical noise model. In particular, the system Hamiltonian is modified by a random term instead of including an explicit quantum bath and system-bath interaction as for the other DD sequences we consider in this work. The model is expressed using an arbitrary unitary pulse that includes a fixed systematic error ϵ_r as in Eq. (33), and reduces to a π pulse in the ideal case. As detailed in Appendix A, this leads to a family of sequences that give rise to an error scaling as $\epsilon_n \sim \epsilon_r^{n/2}$ using n pulses.

These sequences recover some known results at low order: the UR_2 sequence is the CPMG sequence and the UR_4 sequence is the XY4 sequence. In other words, the CPMG and XY4 sequences are also robust to some pulse

imperfections as they cancel flip-angle errors to a certain order. Moreover, by the same recursive arguments, CDD_n can achieve arbitrary flip-angle error suppression while achieving arbitrary $\mathcal{O}(\tau^n)$ (up to saturation) protection. Still, CDD_n requires exponentially more pulses than UR_n, since UR_n is by design a semiclassical $\mathcal{O}(\tau^2)$ sequence. Whether UR_n is also an $\mathcal{O}(\tau^2)$ sequence for a fully quantum bath is an interesting open problem.

C. Optimizing the pulse interval

Nonuniform pulse-interval sequences such as UDD and QDD already demonstrate that introducing pulse intervals longer than the minimum possible ($\tau_{\min}$) can be advantageous. In particular, such alterations can reduce spectral overlap between the filter function and bath spectral density. A longer pulse interval also results in pulses closer to the ideal limit of a small Δ/τ ratio when Δ is fixed. Empirical studies have also found evidence for better DD performance under longer pulse intervals [58,62,73].

We may distinguish two ways to optimize the pulse interval: an asymmetric or symmetric placement of additional delays. For example, the asymmetric and symmetric forms of the XY4 sequence we optimize take the forms

$$\mathrm{XY4}_a(d) \equiv Yf_dXf_dYf_dXf_d, \tag{35a}$$

$$\mathrm{XY4}_s(d) \equiv f_{d/2}Yf_dXf_dYf_dXf_{d/2}, \tag{35b}$$

where d sets the duration of the pulse interval. The asymmetric form here is consistent with how we defined the XY4 sequence in Eq. (13), and except the CPMG sequence we have tacitly defined every sequence so far in its asymmetric form for simplicity. The symmetric form is a simple alteration that makes the placement of pulse intervals symmetric about the midpoint of the sequence. For a generic sequence whose definition requires nonuniform pulse intervals like UDD, we can write the two forms as

$$\mathrm{DD}_a(d) \equiv P_1f_{\tau_1+d}P_2f_{\tau_2+d}\cdots P_nf_{\tau_n+d}, \tag{36a}$$

$$\mathrm{DD}_s(d) \equiv f_{d/2}P_1f_{\tau_1+d}P_2f_{\tau_2+d}\cdots P_nf_{\tau_n+d/2}. \tag{36b}$$

Here, the symmetric nature of DD_s is harder to interpret. The key is to define $\tilde{P}_j = P_1f_{\tau_1}$ as the "effective pulse." In this case, the delay-pulse motif is $f_{d/2}\tilde{P}_jf_{d/2}$ for every pulse in the sequence exhibiting a reflection symmetry of the pulse interval about the center of the pulse. In the asymmetric version, it is $\tilde{P}_jf_d$ instead. Note that $\mathrm{PX}_s(\tau) =$ CPMG, i.e., the symmetric form of the PX sequence is the CPMG sequence. As the CPMG sequence is a well-known sequence, hereafter we refer to the PX sequence as the CPMG sequence throughout our data and analysis regardless of symmetry.

D. Superconducting hardware physics relevant to DD

So far, our account has been abstract and hardware agnostic. Since our demonstrations involve IBMQ superconducting hardware, we now provide a brief background relevant to the operation of DD in such devices. We closely follow Tripathi *et al.* [44], who derived the effective system Hamiltonian of transmon superconducting qubits from first principles and identified the importance of modeling DD performance within a frame rotating at the qubit drive frequency. It was found that DD is still effective with this added complication both in theory and cloud-based demonstrations and, in practice, DD performance can be modeled reasonably well by ideal pulses with a minimal pulse spacing of $\tau_{\min} = \Delta$. We now address these points in more detail.

The effective system Hamiltonian for two qubits (the generalization to $n > 2$ is straightforward) is

$$H_S = -\frac{\omega_{q1}}{2}Z_1 - \frac{\omega_{q2}}{2}Z_2 + JZZ, \tag{37}$$

where the ω_{q_i} are qubit frequencies and $J \neq 0$ is an undesired, always-on ZZ crosstalk. The appropriate frame for describing the superconducting qubit dynamics is that corotating with the number operator $\hat{N} = \sum_{k,l\in\{0,1\}}(k+l)\,|kl\rangle\langle kl| = I - \frac{1}{2}(Z_1+Z_2)$. The unitary transformation into this frame is $U(t) = e^{-it\omega_d\hat{N}t}$, where ω_d is the drive frequency used to apply gates [92]. In this frame, the effective dynamics is given by the Hamiltonian

$$\tilde{H}(t) = \sum_{i=1}^{2}\left(\frac{\omega_d - \omega_{q_i}}{2}\right)Z_i + JZZ + \tilde{H}_{SB}(t) + H_B, \tag{38}$$

where $\tilde{H}_{SB} = U^\dagger(t)H_{SB}U(t)$. To eliminate unwanted interactions, DD must symmetrize JZZ and $\tilde{H}_{SB}$. The JZZ term is removed by applying an X-type DD to the first qubit ("the DD qubit"): $X_1Z_1Z_2X_1 = -Z_1Z_2$, so symmetrization still works as intended. However, the $\tilde{H}_{SB}$ term is time dependent in the rotating frame $U(t)$, which changes the analysis. First, the sign flipping captured by Eq. (9) no longer holds due to the time dependence of $\tilde{H}_{SB}$. Second, some terms in $\tilde{H}_{\mathrm{err}}$ self-average and nearly cancel even without applying DD [93]:

$$\{\sigma_1^x, \sigma_2^x, \sigma_1^y, \sigma_2^y, \sigma^x\sigma^z, \sigma^z\sigma^x, \sigma^y\sigma^z, \sigma^z\sigma^y\}. \tag{39}$$

The remaining terms,

$$\{\sigma_2^z, \sigma^x\sigma^x, \sigma^x\sigma^y, \sigma^y\sigma^x, \sigma^y\sigma^y\}, \tag{40}$$

are not canceled at all. This differs from expectation in that the two terms containing σ_1^y in Eq. (40) are not canceled, whereas in the $\omega_d \to 0$ limit, all terms containing σ_1^y are fully canceled.

Somewhat surprisingly, a nominally universal sequence such as the XY4 sequence, is no longer universal in the rotating frame, again due to the time dependence acquired by $\tilde{H}_{SB}$. In particular, the only terms that perfectly cancel to $\mathcal{O}(\tau)$ are the same Z_1 and ZZ as with the CPMG sequence. However, the list of terms that approximately cancels grows to include

$$\{\sigma^x\sigma^x, \sigma^x\sigma^y, \sigma^y\sigma^x, \sigma^y\sigma^y\}, \tag{41}$$

and when τ is fine-tuned to an integer multiple of $2\pi/\omega_d$, then the XY4 sequence cancels all terms except, of course, terms involving I_1, which commute with the DD sequence.

Consequently, without fine-tuning τ, we should expect the CPMG and XY4 sequences to behave similarly when the terms in Eq. (41) are not significant. Practically, this occurs when $T_1 \gg T_2$ for the qubits coupled to the DD qubit [44]. However, when instead $T_1 \lesssim T_2$ for coupled qubits, the XY4 sequence should prevail. In addition, the analysis in Ref. [44] was carried out under the assumption of ideal π pulses with $\tau_{\min} = \Delta$, and yet, the specific qualitative and quantitative predictions bore out in the cloud-based demonstrations. Hence, it is reasonable to model DD sequences on superconducting transmon qubits as

$$\mathrm{DD}_{\mathrm{sc}} \equiv \hat{P}_1 f_{\tau_1+\Delta_1} \cdots \hat{P}_n f_{\tau_n+\Delta_n}, \tag{42}$$

where $\hat{P}_j$ is once again an ideal pulse with zero width, and the free evolution periods have been incremented by Δ_j—the width of the actual pulse P_j.

E. What this theory means in practice

We conclude our discussion of the background theory by discussing its practical implications for actual DD memory cloud-based demonstrations. This section motivates many of our design choices and our interpretation of the results.

At a high level, our primary demonstration—whose full details are fleshed out in Sec. III below—is to prepare an initial state and apply DD for a duration T. We estimate the fidelity overlap of the initial state and the final prepared state at time T by an appropriate measurement at the end. By adjusting T, we map out a *fidelity decay curve* [see Fig. 1(a) for examples], which serves as the basis of our performance assessment of the DD sequence.

The first important point from the theory is the presence of ZZ crosstalk, a spurious interaction that is always present in fixed-frequency transmon processors, even when the intentional coupling between qubits is turned off. Without DD, the always-on ZZ term induces coherent oscillations in the fidelity decay curve, such as the "Free" curve in Fig. 1(a), depending on the transmon drive frequency, the dressed transmon qubit eigenfrequency, and calibration details [44]. Applying DD via pulses that anticommute with the ZZ term makes it possible to cancel

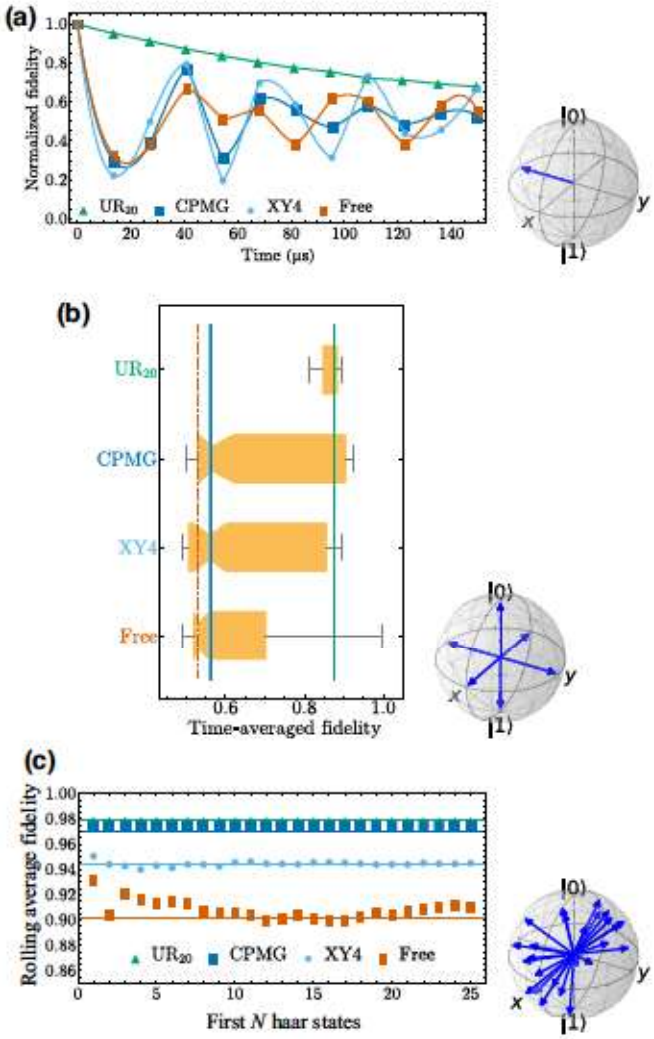


FIG. 1. Representative samples of our results for three DD sequences and free evolution. The Bloch sphere representation of the quantum states used for each plot is shown on the bottom right. (a) Normalized fidelity $f_e(t)/f_e(0)$ under four DD sequences for the initial state $|-i\rangle$ and a fixed calibration cycle on Bogota. (b) We summarize the result of many such fidelity decay curves using a box plot. Each box shows the max (rightmost vertical black lines), inner-quartile range (the orange box), median (the skinny portion of each orange box), and minimum (leftmost vertical black lines) time-averaged fidelities, $F(T = 75\ \mu\mathrm{s})$ in Eq. (44), across the six Pauli states and ten calibration cycles (for a total of 60 data points each) on Bogota. The vertical lines denote the performance of each sequence by its median, colored with the same color as the corresponding sequence. We use this type of box plot to summarize the Pauli demonstration results. (c) We show average fidelity convergence as a function of the number of Haar-random states. In particular, the horizontal lines represent $\mathbb{E}_{100\text{-Haar}}[f_e(T = 3.27\ \mu\mathrm{s})]$, whereas each point represents $\mathbb{E}_{N\text{-Haar}}[f_e(T = 3.27\ \mu\mathrm{s})]$ for increasing N, i.e., the rolling average fidelity. In all cases, we find that 25 states are sufficient for reasonable convergence as $\mathbb{E}_{25\text{-Haar}}[f_e(T)]$ is within 1% of $\mathbb{E}_{100\text{-Haar}}[f_e(T)]$. The data shown are for Jakarta, but a similar result holds across all devices tested.

TABLE II. The three processors used in our cloud-based demonstrations. The total number of qubits n varies from 1 to 7, but in all cases, we applied DD to just one qubit: number 2 for Bogota and 1 for Jakarta (see the insets of Fig. 3 below for the connectivity graph of each device). The choice of the qubit used is motivated by the prediction that $T_1 \gg T_2$ and $T_1 \lesssim T_2$ lead to different DD sequence behavior (see Sec. II E and Ref. [44]). The qubits we have chosen have the highest connectivity on their respective devices and are therefore subject to maximal ZZ crosstalk. The T_1 and T_2 times are the averages of all reported values during data collection for the specified qubit, along with the 2σ sample standard deviation. Data were collected over roughly 20 different calibrations, mostly between August 9–25, 2021 for the Pauli demonstration and January 11–19, 2022 for the Haar-interval demonstration.

Device	ibmq_armonk	ibmq_bogota	ibmq_jakarta
# qubits	1	5	7
Qubit used	q0	q2	q1
T_1 (μs)	140 ± 41	105 ± 41	149 ± 61
T_2 (μs)	227 ± 71	145 ± 63	21 ± 3
Pulse duration (ns)	$71.1\overline{1}$	$35.5\overline{5}$	$35.5\overline{5}$

the corresponding crosstalk term to first order [44,46]. For some sequences—such as UR_{20} in Fig. 1(a)—this first-order cancelation almost entirely dampens the oscillation. One of our goals in this work is to rank DD sequence performance, and ZZ crosstalk cancelation is an important feature of the most performant sequences. The simplest way to cancel ZZ crosstalk is to apply DD to a single qubit (the one we measure) and leave the remaining qubits idle. We choose this simplest strategy in our work since it accomplishes our goal without adding complications to the analysis.

Second, we must choose which qubit to perform our demonstrations on. As we mentioned in our discussion in the previous subsection, we know that the XY4 sequence only approximately cancels certain terms. Naively, we expect these remaining terms to be important when $T_1 \lesssim T_2$ and negligible when $T_1 \gg T_2$. Put differently, a universal sequence such as XY4 should behave similarly to the CPMG sequence when these terms are negligible anyway ($T_1 \gg T_2$) and beat the CPMG sequence when the terms matter ($T_1 \lesssim T_2$). To test this prediction, we pick three qubits where $T_1 > T_2$, $T_2 > T_1$, and $T_1 \lesssim T_2$), as summarized in Table II.

Third, we must contend with the presence of systematic gate errors. When applying DD, these systematic errors can manifest as coherent oscillations in the fidelity decay profile [42,58]. For example, suppose that the error is a systematic over-rotation by ϵ_r within each cycle of DD with N pulses. In that case, we over-rotate by an accumulated error $N\epsilon_r$, which manifests as fidelity oscillations. This explains the oscillations for CPMG and XY4 in Fig. 1(a). To stop this accumulation for a given fixed sequence, one strategy is to apply DD sparsely by increasing the pulse interval τ. However, increasing τ increases the strength of the leading error term obtained by symmetrization, which scales as $\mathcal{O}(\tau^k)$ (where k is a sequence-dependent power).

Thus, there is a tension between packing pulses together to reduce the leading error term and applying pulses sparsely to reduce the build-up of coherent errors. Here, we probe both regimes (dense and sparse pulses) and discuss how this trade-off affects our results. Sequences that are designed to be robust to pulse imperfections play an important role in this study. As the UR_{20} sequence demonstrates in Fig. 1(a), this design strategy can be effective at suppressing oscillations due to both ZZ crosstalk and the accumulation of coherent gate errors. Our work attempts to empirically disentangle the various trade-offs in DD sequence design.

III. METHODS

We performed our cloud-based demonstrations on three IBM Quantum superconducting processors [94] with OpenPulse access [48,50]: ibmq_armonk, ibmq_bogota, and ibmq_jakarta. Key properties of these processors are summarized in Table II.

All our demonstrations follow the same basic structure, summarized in Fig. 2. Namely, we prepare a single-qubit initial state $|\psi\rangle = U|0\rangle$, apply N repetitions of a given DD sequence S lasting total time t, undo U by applying $U^\dagger$, and finally measure in the computational basis. Note that this is a single-qubit protocol. Even on multiqubit devices, we intentionally only apply DD to the single qubit we intend to measure to avoid unwanted ZZ crosstalk effects, as discussed in Sec. II E. The qubit used on each device is listed in Table II.

We empirically estimate the Uhlmann fidelity

$$f_e(t) = |\langle\psi|\rho_{\text{final}}(t)|\psi\rangle|^2 \tag{43}$$

with 95% confidence intervals by counting the number of 0's returned out of 8192 shots and bootstrapping. The results we report below were obtained using OpenPulse [48], which allows for refined control over the exact waveforms sent to the chip instead of the coarser control that the standard Qiskit circuit API gives [49,50]. Appendix B provides a detailed comparison between the two APIs, highlighting the significant advantage of using OpenPulse.

We utilize this simple procedure to perform two types of demonstrations that we refer to as Pauli and Haar, and explain in detail below. Briefly, the Pauli demonstration probes the ability of DD to maintain the fidelity of the six Pauli states (the eigenstates of $\{\sigma^x, \sigma^y, \sigma^z\}$) over long times, while the Haar demonstrations address this question for Haar-random states and short times.

A. Pauli demonstration for long times

For the Pauli demonstration, we keep the pulse interval τ fixed to the smallest possible value allowed by the relevant device, $\tau_{\min} = \Delta \approx 36$ ns (or approximately 71 ns), the width of the X and Y pulses (see Table II for specific Δ values for each device). Practically, this corresponds to placing all the pulses as close to each other as possible, i.e., the peak-to-peak spacing between pulses is just Δ (except for nonuniformly spaced sequences such as QDD). For ideal pulses and uniformly spaced sequences, this is expected to give the best performance [21,95], so unless otherwise stated, all DD is implemented with minimal pulse spacing, $\tau = \Delta$.

Within this setting, we survey the capacity of each sequence to preserve the six Pauli eigenstates $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle, |+i\rangle, |-i\rangle\}$ for long times, which we define as $T \geq 75$ μs. In particular, we generate fidelity decay curves like those shown in Fig. 1(a) by incrementing the number of repetitions of the sequence, N, and thereby sampling $f_e(t)$ [Eq. (43)] for increasingly longer times t. Using the XY4 sequence as an example, we apply $P_1 P_2 \cdots P_n = (XYXY)^N$ for different values of N while keeping the pulse interval fixed. After generating fidelity decay curves over ten or more different calibration cycles across a week, we summarize their performance using a box plot like that shown in Fig. 1(b). For the Pauli demonstrations, the box plot bins the average normalized fidelity,

$$F(T) \equiv \frac{1}{f_e(0)} \langle f_e(t) \rangle_T = \frac{1}{T} \int_0^T dt \frac{f_e(t)}{f_e(0)}, \tag{44}$$

at time T computed using numerical integration with Hermite polynomial interpolation. Note that no DD is applied at $f_e(0)$, so we account for state preparation and measurement errors by normalizing. [A sense of the value of $f_e(0)$ and its variation can be gleaned from Fig. 7 below.] The same holds for Fig. 1(a).

We can estimate the best-performing sequences for a given device by ranking them by the median performance from this data. In Fig. 1(b), for example, this leads to the fidelity ordering UR_{20} > CPMG > XY4 > free evolution on Bogota, which agrees with the impression left by the decay profiles in Fig. 1(a) generated in a single run. We use $F(T)$ because fidelity profiles $f_e(t)$ are generally oscillatory and noisy, so fitting $f_e(t)$ to extract a decay constant (as was done in Ref. [41]) does not return reliable results across the many sequences and different devices we tested. We provide a detailed discussion of these two methods in Appendix C.

B. Haar-interval demonstrations

The Pauli demonstration estimates how well a sequence preserves quantum memory for long times without requiring excessive data, but it leaves several open questions. (1) Does DD preserve quantum memory for an *arbitrary* state? (2) Is $\tau = \tau_{\min}$ the best choice empirically? (3) How effective is DD for short times? In the Haar-interval demonstration, we address all of these questions. This setting—of short times and arbitrary states—is particularly relevant to improving quantum computations with DD [31,35,39,96]. For example, DD pulses have been inserted into idle spaces to improve circuit performance [43,45,47,97,98].

In contrast to the Pauli demonstration, where we fixed the pulse delay $d = 0$ and the symmetry $\zeta = a$ [see Eqs. (35a) and (35b)] and varied t, here we fix $t = T$ and vary d and ζ, writing $f_e(d, \zeta; t)$. Furthermore, we now sample over a fixed set of 25 Haar-random states instead of the six Pauli states. Note that we theoretically expect the empirical Haar average over n states $\mathbb{E}_{n\text{-Haar}}[f_e] \equiv (1/n) \sum_{i=1}^{n} f_e(\psi_i)$ for $|\psi_i\rangle \sim$ Haar to converge to the true Haar average $\langle f_e \rangle_{\text{Haar}}$ for sufficiently large n. As shown by Fig. 1(c), 25 states are enough for a reasonable empirical convergence, while keeping the total number of circuits to submit and run on IBMQ devices manageable in practice.

The Haar-interval demonstration procedure is now as follows. For a given DD sequence and time T, we sample $f_e(d, \zeta; t)$ for $\zeta \in \{a, s\}$ from $d = 0$ to $d = d_{\max}$ for eight equally spaced values across 25 fixed Haar-random states and ten calibration cycles (250 data points for each d value). Here $d = 0$ and $d = d_{\max}$ correspond to the tightest and sparsest pulse placements, respectively. At $d_{\max}$, we consider only a single repetition of the sequence during the time window T. To make contact with DD in algorithms, we first consider a short-time limit $T = T_{5\mathrm{CNOT}} \approx 4$ μs, which is the amount of time it takes to apply five

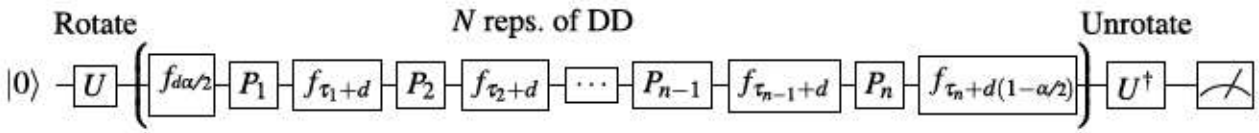


FIG. 2. The "quantum memory" circuit that underlies all of our demonstrations. By sampling the circuit using 8192 shots, we estimate the Uhlmann fidelity, $f_e(t)$ in Eq. (43), between the prepared initial state and the final state under the DD sequence $P_1 f_{\tau_1} P_2 f_{\tau_2} \cdots P_n f_{\tau_n}$. We have included an additional adjustable pulse interval f_d, where we set $d = 0$ for the Pauli demonstration (Sec. III A) and systematically vary d for the Haar demonstration (Sec. III B). The choice of $\alpha = 0$ (1) corresponds to an asymmetric (symmetric) placement of the additional delays.

controlled-NOT (CNOT) gates on the DD qubit. As shown by the example fidelity decay curves of Fig. 1(a), we expect similar results for $T \lesssim 15$ μs before fidelity oscillations begin. To make contact with the Pauli demonstration, we also consider a long-time limit of $T = 75$ μs. Finally, to keep the number of demonstrations manageable, we only optimize the interval of the best-performing UR sequence, the best-performing QDD sequence, as well as CPMG, XY4, and free evolution as constant reference sequences.

IV. RESULTS

We split our results into three subsections. The first two summarize the results of demonstrations aimed at preserving Pauli eigenstates (Sec. III A) and Haar-random states (Sec. III B). In the third subsection, we discuss how theoretical expectations about the saturation of CDD_n [95] and UR_n [59] compare to the demonstration results.

A. The Pauli demonstration result: DD works and advanced DD works even better

In Fig. 3, we summarize the results of the Pauli demonstration. We rank each device's top-ten sequences by median performance across the six Pauli states and ten or more calibration cycles, followed by CPMG, XY4, and free evolution as standard reference sequences. As discussed in Sec. III A, the figure of merit is the normalized, time-averaged fidelity at 75 μs [see Eq. (44)] that is a long-time average.

The first significant observation is that DD is better than free evolution, consistent with numerous previous DD studies. This is evidenced by free evolution (Free) being close to the bottom of the ranking for every device.

Secondly, advanced DD sequences outperform Free and CPMG, XY4 (shown as dark-blue and light-blue vertical lines in Fig. 3). In particular, 29 out of the 30 top sequences across all three devices are advanced—the exception being XY4 on ibmq_armonk. These sequences perform so well that there is a 50% improvement in the median fidelity of these sequences (0.85–0.95) over Free (0.45–0.55). The best sequences also have a much smaller variance in performance, as evidenced by their smaller interquartile range in F. For example, on ibmq_armonk 75% of all demonstration outcomes for $F_{\mathrm{UDDx}_{25}}(75\ \mu\mathrm{s})$ fall between 0.9 and 0.95, whereas for Free, the same range is between 0.55 and 0.8. Similar comparisons show that advanced DD beats CPMG and XY4 for every device to varying degrees.

Among the top advanced sequences shown, $16/29 \sim 55\%$ are UDD or QDD, which use nonuniform pulse intervals. On the one hand, the dominance of advanced

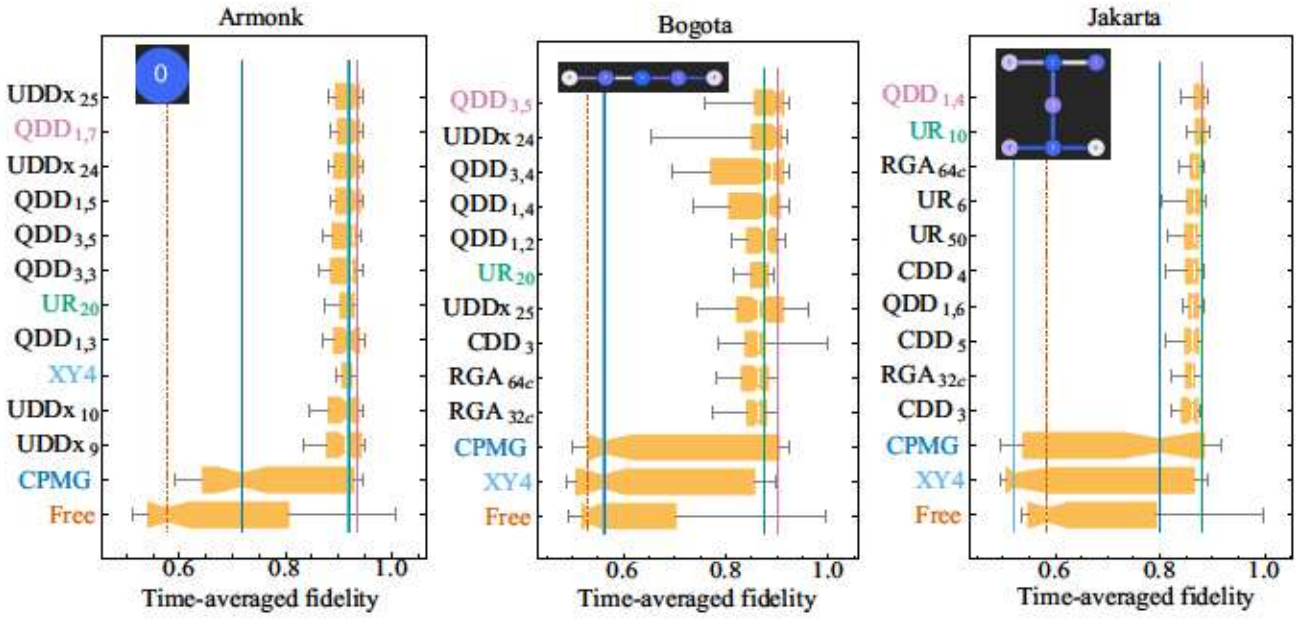


FIG. 3. A summary of the Pauli demonstration results for all three devices. The top-ten sequences are ranked from top to bottom by median average fidelity [Eq. (44)] for the listed time $T = 75$ μs. Also displayed in all cases are CPMG and XY4 along with free evolution (Free). Colored vertical lines indicate the median fidelity of the correspondingly colored sequence (best UR, best QDD, CPMG, XY4, and free evolution). Thin white lines through the orange boxes indicate the median fidelities in all other cases. Otherwise, the conventions of Fig. 1(b) apply. Two main observations emerge. (1) DD systematically outperforms free evolution as indicated by "Free" being at the bottom. The corresponding dash-dot red vertical line denotes the median average fidelity of Free, $F_{\mathrm{Free}}(75\ \mu\mathrm{s})$, which is below 0.6 on every device. (2) Advanced DD sequences, especially the UDD and QDD families, provide a substantial improvement over both CPMG and XY4áźłhe best median performance of the top sequence is indicated by a solid green line, $F_{\mathrm{XY4}}(75\ \mu\mathrm{s})$ by a cyan line, and $F_{\mathrm{CPMG}}(75\ \mu\mathrm{s})$ by a dark blue line.

DD strategies, especially UDD and QDD, is not surprising. After all, these sequences were designed to beat the simple sequences. On the other hand, as reviewed above, many confounding factors affect how well these sequences perform, such as finite pulse-width errors and the effect of the rotating frame. It is remarkable that despite these factors, predictions made based on ideal pulses apply to actual noisy quantum devices.

Finally, we comment more specifically on CPMG and XY4 as these are widely used and well-known sequences. Generally, they do better than free evolution, which is consistent with previous results. On ibmq_armonk XY4 outperforms CPMG, which outperforms free evolution. On ibmq_bogota, both XY4 and CPMG perform comparably and marginally better than free evolution. Finally, On ibmq_jakarta XY4 is worse than CPMG—and even free evolution—but the median performance of CPMG is substantially better than that of free evolution. It is tempting to relate these results to the relative values of T_1 and T_2, as per Table II, in the sense that CPMG is a single-axis-("pure-X") type sequence [Eq. (10)] that does not suppress the system-bath σ^x coupling term responsible for T_1 relaxation, while XY4 does. Nevertheless, a closer look at Fig. 3 shows that such an explanation would be inconsistent with the fact that both single-axis and multiaxis sequences are among the top-ten performers for ibmq_armonk and ibmq_bogota.

The exception is ibmq_jakarta, for which there are no single-axis sequences in the top ten. This processor has a much smaller T_2 than T_1 (for ibmq_armonk and ibmq_bogota, $T_1 > T_2$), so one might expect that a single-axis sequence such as UDD or CPMG would be among the top performers, but this is not the case. In the same vein, the top-performing asymmetric $\mathrm{QDD}_{n,m}$ sequences all have $n < m$, despite this opposite ordering of T_1 and T_2. These results show that the relative value of T_1 with respect to T_2 for a given device is not predictive of DD sequence performance as simply as suggested by theory.

B. Haar-interval demonstration results: DD works on arbitrary states, and increasing the pulse interval can help substantially

We summarize the Haar-interval demonstration results in Fig. 4. Each plot corresponds to $f_e(d, \zeta^*; t)$ as a function of d, the additional pulse-interval spacing. We plot the spacing in relative units of $d/d_{\max}$, i.e., the additional delay fraction, for each sequence. The value $d_{\max}$ corresponds to the largest possible pulse spacing where only a single repetition of a sequence fits within a given unit of time. Hence, $d_{\max}$ depends both on the demonstration time T and the sequence tested, and plotting $d \in [0, d_{\max}]$ directly leads to sequences with different relative scales. Normalizing with respect to $d_{\max}$ therefore makes the comparison between sequences easier to visualize in a single plot.

For each device, we compare CPMG XY4 the best robust sequence from the Pauli demonstration, the best nonuniform sequence from the Pauli demonstration, and Free. The best robust and nonuniform sequences correspond to UR_n and $\mathrm{QDD}_{n,m}$ for each device. We only display the choice of ζ with the better optimum, and the error bars correspond to the inner-quartile range across the 250 data points. These error bars are similar to those reported in the Pauli demonstration.

1. $d = 0$: DD also continues to outperform free evolution over Haar-random states

The $d = 0$ (i.e., additional delay fraction $d/d_{\max} = 0$) limit is identical to those in the Pauli demonstration, i.e., with the minimum possible pulse spacing. The advanced sequences, UR_n and $\mathrm{QDD}_{n,m}$, outperform Free by a large margin for short times and by a moderate margin for long times. In particular, they have higher median fidelity, a smaller interquartile range than Free, and are consistently above the upper quartile in the short-time limit. But, up to error bars, UR_n and $\mathrm{QDD}_{n,m}$ are statistically indistinguishable in terms of their performance, except that UR_n has a better median than $\mathrm{QDD}_{n,m}$ for ibmq_jakarta.

Focusing on CPMG, for short times, it does slightly worse than the other sequences, yet still much better than Free, but for long times, it does about as poorly as Free. On ibmq_bogota and ibmq_jakarta, XY4 performs significantly worse than UR_n and $\mathrm{QDD}_{n,m}$, and is always worse than CPMG. The main exception to this rule is ibmq_armonk where XY4 is comparable to UR_n and $\mathrm{QDD}_{n,m}$ in both the short- and long-time limits.

Overall, while all sequences lose effectiveness in the long-time limit, the advanced sequences perform well when the pulse spacing is $d = 0$.

2. $d > 0$: Increasing the pulse interval can improve DD performance

It is clear from Fig. 4 [most notably from Fig. 4(d)] that increasing the pulse interval can sometimes significantly improve DD performance. For example, considering Fig. 4(a), at $d = 0$ CPMG is worse than the other sequences, but by increasing d, CPMG matches the performance of the sequences. The same qualitative result occurs even more dramatically for long times [Fig. 4(b)]. Here, CPMG goes from a median fidelity around 0.6—as poor as Free—to around 0.9 at $d/d_{\max} \approx 0.55$. The significant improvement of CPMG with increasing d is fairly generic across devices and times, with the only exception being ibmq_jakarta and long times. Thus, even a simple sequence such as CPMG (or XY4, which behaves similarly) can compete with the highest-ranking advanced sequences if the pulse interval is optimized. Unsurprisingly, optimizing the pulse interval can help, but the degree of improvement is surprising, particularly the ability of

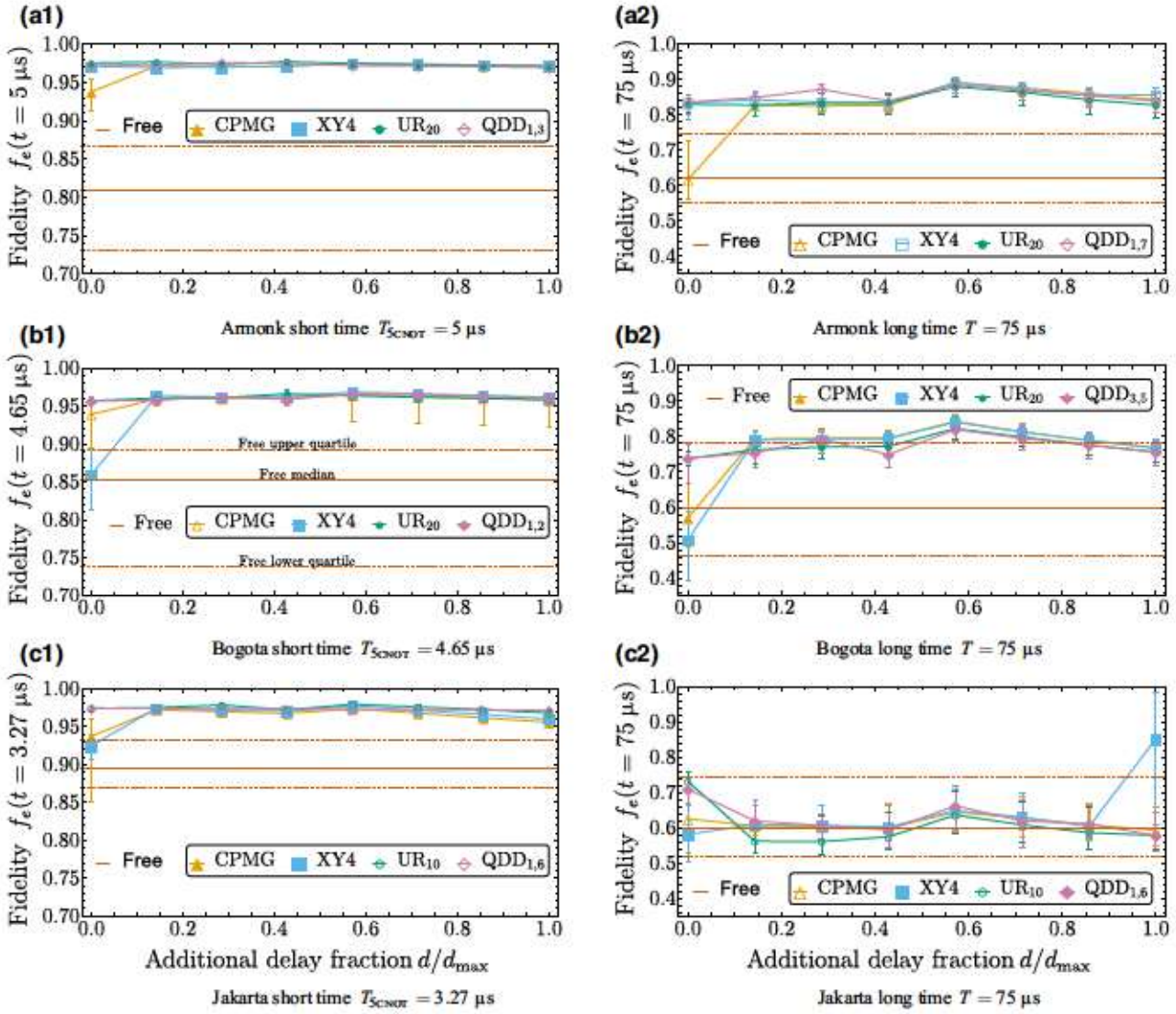


FIG. 4. Summary of the Haar-interval demonstration across the three devices. We explore the relationship between the median fidelity $f_e(d, \zeta^*; t)$ across Haar-random states as a function of the relative spacing of the pulse intervals, d/d_{max}. The value d_{max} corresponds to the largest possible pulse spacing where only a single repetition of a sequence fits. While d_{max} is a sequence-dependent quantity, we sample d/d_{max} evenly regardless of the sequence. For a given device and T, we plot the best robust sequence (UR_n) and the best nonuniform sequence $QDD_{n,m}$ from Fig. 3, as well as CPMG, XY4, and free evolution as reference sequences. The Free curve has a solid line for its median and a dashed line above and below, representing its upper and lower quartiles. The parameters n, m in $QDD_{n,m}$ are chosen so that the sequence fits the time window. The left and right columns correspond to short ($T = T_{5CNOT}$) and long times ($T = 75\ \mu s$), respectively. The confidence intervals are upper and lower quartiles—75% and 25% of all fidelities lie below them, respectively. When the interval is not visible, this is because it is smaller than the marker for the median. Both asymmetric (open symbols) and symmetric (filled symbols) sequences were considered, but we display only the better of the two (for some empirically optimal d^*). (a1) Armonk short time, $T_{5CNOT} = 5\ \mu s$. (a2) Armonk long time, $T = 75\ \mu s$. (b1) Bogota short time, $T_{5CNOT} = 4.65\ \mu s$. (b2) Bogota long time, $T = 75\ \mu s$. (c1) Jakarta short time, $T_{5CNOT} = 3.27\ \mu s$. (c2) Jakarta long time, $T = 75\ \mu s$.

simple sequences to match the performance of advanced ones.

3. $d = d_{max}$: *Performance at the single-cycle limit*

In a similar vein, it is notable how well sequences do in the $d = d_{max}$ limit (at the right of each plot in Fig. 4 where $d/d_{max} = 1$). For CPMG at $T = 75\ \mu s$, this corresponds to applying a pulse on average every 37.5 μs; this certainly does not obey the principle of making τ small, implied from error terms scaling as $\mathcal{O}(\tau^n)$ as in the standard theory. There is always a trade-off between combating noise and packing too many pulses on real devices with finite width and implementation errors. Incoherent errors

result in a well-documented degradation that reduces the cancelation order for most of the sequences we have considered; see Sec. II B. In addition, coherent errors build up as oscillations in the fidelity; see Sec. II E. Low-order sequences such as CPMG and XY4 are particularly susceptible to both incoherent and coherent errors, and are therefore expected to exhibit an optimal pulse interval. Moreover, the circuit structure will restrict the pulse interval when implementing DD on top of a quantum circuit. The general rule gleaned from Fig. 4 is to err toward the latter and apply DD sparsely. In particular, $d = d_{\max}$ results in a comparable performance to $d = 0$ or a potentially significant improvement.

Nevertheless, whether dense DD is better than sparse DD can depend on the specific device characteristics, desired timescale, and relevant metrics. As a case in point, note the Haar demonstration results on ibmq_jakarta in the long-time limit. Here, most sequences—aside from XY4—deteriorate with increasing d. Surprisingly, the best strategy in median performance is XY4, which does come at the cost of a sizeable inner-quartile range.

UR_n for $d = 0$ does substantially better than Free for all six panels, which is an empirical confirmation of its claimed robustness. Even for $d > 0$, UR_n remains a high-fidelity and low-variance sequence. Since other sequences only roughly match UR_n upon interval optimization, using a robust sequence is a safe default choice.

C. Saturation of the CDD and UDD sequences, and an optimum for the UR sequence

In Fig. 5, we display the time-averaged fidelity from ibmq_bogota for CDD_n, UR_n, and UDD_{x_n} as a function of n. Related results for the other DD sequence families are discussed in Appendix E. As discussed in Sec. II A 2, CDD_n performance is expected to saturate at some n_{opt} according to Eq. (20). In Fig. 5(a), we observe evidence of this saturation at $n_{\mathrm{opt}} = 3$ on ibmq_bogota. We can use this to provide an estimate of $\epsilon = \|H_B\| + \|H_{SB}\|$. Substituting n_{opt} into Eq. (20) we find that

$$\overline{c}\epsilon\Delta \in [4^{-5}, 4^{-4}] = [9.77 \times 10^{-4}, 3.91 \times 10^{-3}]. \quad (45)$$

This means that $\epsilon\Delta \ll 1$ (we set $\overline{c} \approx 1$), which confirms the assumption we needed to make for DD to give a reasonable suppression given that XY4 yields $\mathcal{O}(\epsilon\tau^2)$ suppression. This provides a level of empirical support for the validity of our assumptions. In addition, $\Delta \approx 51$ ns on ibmq_bogota, so we conclude that $\epsilon \approx 0.5$ MHz. Since qubit frequencies are roughly $\omega_q \approx 4.5$–5 GHz on IBM devices, this also confirms that $\omega_q \gg \epsilon$, as required for a DD pulse. We observe a similar saturation in CDD_n on Jakarta and Armonk as well (Appendix E).

Likewise, for an ideal demonstration with fixed time T, the performance of UDDx_n should scale as $\mathcal{O}(\tau^n)$, and hence we expect a performance that increases monotonically with n. In practice, this performance should

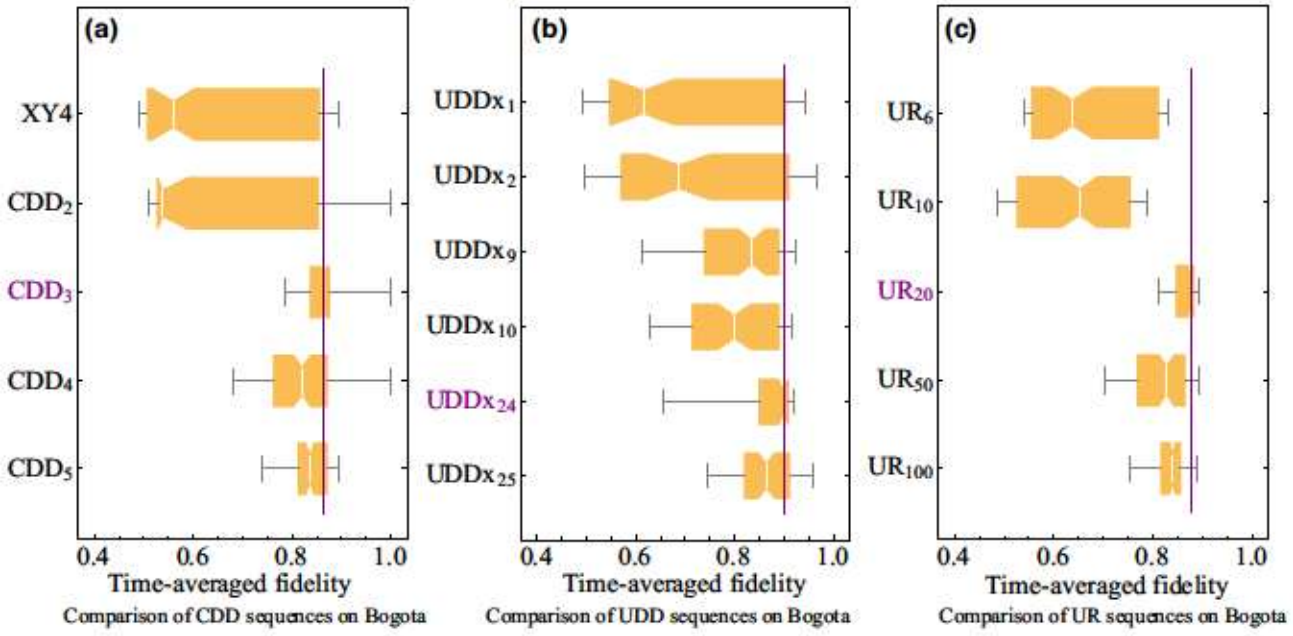


FIG. 5. Average fidelity $f_e(t)$ at 75 μs for (a) CDD_n, $n \in \{1, \ldots, 5\}$, (b) UDD_{x_n}, $n \in \{1, 2, 9, 24, 25\}$, and (c) UR_n, $n \in \{6, 10, 20, 50, 100\}$. All three sequence families exhibit saturation on ibmq_bogota as we increase n, as expected from theory [39,59,76]. The vertical purple line denotes the median performance of the correspondingly colored sequence in each panel, which is also the top-performing sequence by this metric.

saturate once the finite pulse-width error $\mathcal{O}(\Delta)$ is the dominant noise contribution [76]. Once again, the UDD_n sequence performance on ibmq_bogota is consistent with theory. In particular, we expect (and observe) a consistent increase in performance with increasing n until performance saturates. While this saturation is also seen on Armonk, on Jakarta UDDx_n's performance differs significantly from theoretical expectations (see Appendix E).

UR_n also experiences a trade-off between error suppression and noise introduced by pulses. After a certain optimal n, the performance for UR_n is expected to drop [59]. In particular, while UR_n yields $\mathcal{O}(\epsilon_r^{n/2})$ suppression with respect to flip-angle errors (Sec. II B 3), all UR_n provide $\mathcal{O}(\tau^2)$ decoupling, i.e., adding more free evolution periods also means adding more noise. Thus, we expect UR_n to improve with increasing n until performance saturates. On ibmq_bogota, by increasing up to UR_{20}, we gain a large improvement over UR_{10}, but increasing further to UR_{50} or UR_{100} results in a small degradation in performance; see Fig. 5. A similar saturation occurs with ibmq_jakarta and ibmq_armonk (see Appendix E).

V. SUMMARY AND CONCLUSIONS

We performed an extensive survey of ten DD families (a total of 60 sequences) across three superconducting IBM devices. In the first set of demonstrations (the Pauli demonstration, Sec. III A), we tested how well these 60 sequences preserve the six Pauli eigenstates over relatively long times (25–75 μs). Motivated by theory, we used the smallest possible pulse interval for all sequences. We then chose the top-performing QDD and UR sequences from the Pauli demonstration for each device, along with CPMG, XY4, and free evolution as baselines, and studied them extensively. In this second set of demonstrations (the Haar demonstration; Sec. III B), we considered 25 (fixed) Haar-random states for a wide range of pulse intervals, τ.

In the Pauli demonstration (Sec. III A), we ranked sequence performance by the median time-averaged fidelity at $T = 75$ μs. This ranking is consistent with DD theory. The best-performing sequence on each device substantially outperforms free evolution. Moreover, the expected deviation, quantified using the inner-quartile range of the average fidelity, was much smaller for DD than for free evolution. Finally, 29 out of the 30 best-performing sequences were "advanced" DD sequences, explicitly designed to achieve high-order cancelation or robustness to control errors.

We reported pointwise fidelity rather than the coarse-grained time-averaged fidelity in the Haar-interval demonstration (Sec. III B). At $\tau = 0$, the Haar-interval demonstration is identical to the Pauli demonstration except for the expanded set of states. Indeed, we found the same hierarchy of sequence performance between the two demonstrations. For example, on ibmq_jakarta, we found that $\mathrm{XY4} < \mathrm{CPMG} < \mathrm{QDD}_{1,6} < \mathrm{UR}_{10}$ for both Figs. 3 and 4. This suggests that a test over the Pauli states is a good proxy to Haar-random states for our metric.

However, once we allowed the pulse interval (τ) to vary, we found two unexpected results. First, contrary to expectations, advanced sequences, which theoretically provide a better performance, do not retain their performance edge. Second, for most devices and times probed, DD sequence performance improves or stays roughly constant with increasing pulse intervals before decreasing slightly for very long pulse intervals. This effect is particularly significant for the basic CPMG and XY4 sequences. Relating these two results, we found that with pulse-interval optimization, the basic sequences' performance is statistically indistinguishable from that of the advanced UR and QDD sequences. In stark contrast to the theoretical prediction favoring short intervals, choosing the longest possible pulse interval, with one sequence repetition within a given time, is generally better than the minimum interval. The one exception to these observations is the ibmq_jakarta processor for $T = 75$ μs, which is larger than its mean T_2 of 20.7 μs. Here, the advanced sequences significantly outperform the basic sequences at their respective optimal interval values ($\tau = 0$ for the advanced sequences), and DD performance degrades with sufficiently large pulse intervals. The short T_2 for ibmq_jakarta is notable, since in contrast, $T = 75\ \mu\mathrm{s} < \langle T_2 \rangle$ for both ibmq_armonk ($\langle T_2 \rangle = 230\ \mu\mathrm{s}$) and ibmq_bogota ($\langle T_2 \rangle = 146\ \mu\mathrm{s}$). We may thus conclude that, overall, sparse DD is preferred to tightly packed DD, provided decoherence in the free evolution periods between pulses is not too strong.

The UR sequence either matched or nearly matched the best performance of any other tested sequence at any τ for each device. It also achieved near-optimal performance at $\tau = 0$ in four of the six cases shown in Fig. 4. This is a testament to its robustness and suggests that the UR_n family is a good generic choice, provided an optimization to find a suitable n for a given device is performed. In our case, this meant choosing the top-performing UR_n member from the Pauli demonstration. Alternatively, our results suggest that, as long as $T < T_2$, one can choose a basic sequence and likely achieve comparable performance by optimizing the pulse interval. In other words, optimizing the pulse interval for a given basic DD sequence or optimizing the order of an advanced DD sequence at zero pulse interval are equally effective strategies for using DD in practice. However, the preferred strategy depends on hardware constraints. For example, if OpenPulse access (or a comparable control option) is not available so that a faithful UR_n implementation is not possible, one would be constrained to optimizing CPMG or XY4 pulse intervals. Under such circumstances, where the number of DD optimization parameters is restricted, using a variational approach to identify the remaining parameters (e.g., pulse intervals) can be an effective approach [97].

Overall, theoretically promising, advanced DD sequences work well in practice. However, one must fine-tune the sequence to obtain the best DD performance for a given device. A natural and timely extension of our work would be developing a rigorous theoretical understanding of our observations, which do not always conform to previous theoretical expectations. Developing DD sequences for specific hardware derived using the physical model of the system instead of trial-and-error optimization, or using machine learning methods [99], are other interesting directions. A thorough understanding of how to tailor DD sequences to today's programmable quantum computers could be vital in using DD to accelerate the attainment of fault-tolerant quantum computing [39].

The code and data that support the findings of this study are openly available from Zenodo [100]. In more detail, this citable Zenodo repository contains (i) all raw and formatted data in this work (including machine calibration specifications), (ii) PYTHON code to submit identical demonstrations, (iii) the *Mathematica* code used to analyze the data, and (iv) a brief tutorial on installing and using the code as part of the GitHub ReadMe.

ACKNOWLEDGMENTS

N.E. was supported by the U.S. Department of Energy (DOE) Computational Science Graduate Fellowship under Award No. DE-SC0020347. This material is based upon work supported by the National Science Foundation's Quantum Leap Big Idea under Grant No. OMA-1936388. This research was supported by the ARO MURI grant W911NF-22-S-0007. L.T. and G.Q. acknowledge funding from the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Accelerated Research in Quantum Computing under Award No. DE-SC0020316. This material is based upon work supported by the U.S. Department of Energy, Office of Science, under Award No. DE-SC0021661. We acknowledge the use of IBM Quantum services for this work. The views expressed are those of the authors, and do not reflect the official policy or position of IBM or the IBM Quantum team. We acknowledge the access to advanced services provided by the IBM Quantum Researchers Program. We thank Vinay Tripathi for insightful discussions, particularly regarding device-specific effects. D.L. is grateful to Professor Lorenza Viola for insightful discussions and comments.

APPENDIX A: SUMMARY OF THE DD SEQUENCES BENCHMARKED IN THIS WORK

Here we provide definitions for all the DD sequences we tested. To clarify what free evolution periods belong between pulses, we treat uniform and nonuniform pulse-interval sequences separately. When possible, we define a pulse sequence in terms of another sequence (or entire DD sequence) using the notation $[\cdot]$. In addition, several sequences are recursively built from simpler sequences. When this happens, we use the notation $S = s_1([s_2])$, whose meaning is illustrated by the example of CDD_n [Eq. (15)].

1. Uniform pulse-interval sequences

All the uniform pulse-interval sequences are of the form

$$f_\tau - P_1 - f_{2\tau} - P_2 - f_{2\tau} - \cdots - f_{2\tau} - P_n - f_\tau. \quad \text{(A1)}$$

For brevity, we omit the free evolution periods in the following definitions.

We distinguish between single-axis and multiaxis sequences, by which we mean the number of orthogonal interactions in the system-bath interaction (e.g., pure dephasing is a single-axis case), not the number of axes used in the pulse sequences.

First, we list the single-axis DD sequences:

$$\text{Hahn} \equiv X, \quad \text{(A2a)}$$

$$\text{super-Hahn/RGA}_{2x} \equiv X - \overline{X}, \quad \text{(A2b)}$$

$$\text{RGA}_{2y} \equiv Y - \overline{Y}, \quad \text{(A2c)}$$

$$\text{CPMG} \equiv X - X, \quad \text{(A2d)}$$

$$\text{super-CPMG} \equiv X - X - \overline{X} - \overline{X}. \quad \text{(A2e)}$$

Second, the UR_n sequence for $n \geq 4$ and even n is defined as

$$\mathrm{UR}_n = (\pi)_{\phi_1} - (\pi)_{\phi_2} - \cdots - (\pi)_{\phi_n}, \quad \text{(A3a)}$$

$$\phi_k = \frac{(k-1)(k-2)}{2}\Phi^{(n)} + (k-1)\phi_2, \quad \text{(A3b)}$$

$$\Phi^{(4m)} = \frac{\pi}{m}, \qquad \Phi^{(4m+2)} = \frac{2m\pi}{2m+1}, \quad \text{(A3c)}$$

where $(\pi)_\phi$ is a π rotation about an axis that makes an angle ϕ with the x axis, ϕ_1 is a free parameter usually set to 0 by convention, and $\phi_2 = \pi/2$ is a standard choice we use. This is done so that $\mathrm{UR}_4 = \mathrm{XY4}$, as discussed in Ref. [59] (note that despite this UR_n was designed for single-axis decoherence).

Next, we list the multiaxis sequences. We start with the XY4 sequence and all its variations:

$$\text{XY4}\backslash\text{CDD}_1 \equiv Y - X - Y - X, \quad \text{(A4a)}$$

$$\text{CDD}_n \equiv \text{XY4}([\text{CDD}_{n-1}]), \quad \text{(A4b)}$$

$$\text{RGA}_4 \equiv \overline{Y} - X - \overline{Y} - X, \quad \text{(A4c)}$$

$$\text{RGA}_{4p} \equiv \overline{Y} - \overline{X} - \overline{Y} - \overline{X}, \quad \text{(A4d)}$$

$$\text{RGA}_{8c}/\text{XY8} \equiv X - Y - X - Y - Y - X - Y - X, \quad \text{(A4e)}$$

$$\mathrm{RGA}_{8a} \equiv X - \overline{Y} - X - \overline{Y} - Y - \overline{X} - Y - \overline{X}, \tag{A4f}$$

$$\text{super-Euler} \equiv X - Y - X - Y - Y - X - Y - X - \overline{X} - \overline{Y} - \overline{X} - \overline{Y} - \overline{Y} - \overline{X} - \overline{Y} - \overline{X}, \tag{A4g}$$

$$\mathrm{KDD} \equiv [K_{\pi/2}] - [K_0] - [K_{\pi/2}] - [K_0]. \tag{A4h}$$

Here K_ϕ is a composite of five pulses:

$$K_\phi \equiv (\pi)_{\pi/6+\phi} - (\pi)_\phi - (\pi)_{\pi/2+\phi} - (\pi)_\phi - (\pi)_{\pi/6+\phi}. \tag{A5}$$

For example, $(\pi)_0 = X$ and $(\pi)_{\pi/2} = Y$. The series $K(\phi)$ is itself a π rotation about the ϕ axis followed by a $-\pi/3$ rotation about the z axis. To see this, note that a π rotation about the ϕ axis can be written $(\pi)_\phi = R_Z(-\phi)R_Y(-\pi/2)R_Z(\pi)R_Y(\pi/2)R_Z(\phi)$, and one can verify the claim by direct matrix multiplication. KDD (A4g) is the Knill-composite version of XY4 with a total of 20 pulses [58,62]. Note that the alternation of ϕ between 0 and $\pi/2$ means that successive pairs give rise to a $-\pi/3 + \pi/3 = 0$ z rotation at the end.

Next, we list the remaining multiaxis RGA sequences:

$$\mathrm{RGA}_{16b} \equiv \mathrm{RGA}_{4p}([\mathrm{RGA}_{4p}]), \tag{A6a}$$

$$\mathrm{RGA}_{32a} \equiv \mathrm{RGA}_{4}([\mathrm{RGA}_{8a}]), \tag{A6b}$$

$$\mathrm{RGA}_{32c} \equiv \mathrm{RGA}_{8c}([\mathrm{RGA}_{4}]), \tag{A6c}$$

$$\mathrm{RGA}_{64a} \equiv \mathrm{RGA}_{8a}([\mathrm{RGA}_{8a}]), \tag{A6d}$$

$$\mathrm{RGA}_{64c} \equiv \mathrm{RGA}_{8c}([\mathrm{RGA}_{8c}]), \tag{A6e}$$

$$\mathrm{RGA}_{256a} \equiv \mathrm{RGA}_{4}([\mathrm{RGA}_{64a}]). \tag{A6f}$$

2. Nonuniform pulse-interval sequences

The nonuniform sequences are described by a general DD sequence of the form

$$f_{\tau_1} - P_1 - f_{\tau_2} - \cdots - f_{\tau_m} - P_m - f_{\tau_{m+1}} \tag{A7}$$

for pulses P_j applied at times t_j for $1 \le j \le m$. Thus, for ideal, zero-width pulses, the interval times are $\tau_j = t_j - t_{j-1}$ with $t_0 \equiv 0$ and $t_{m+1} \equiv T$, the total desired duration of the sequence.

a. Ideal UDD

For ideal pulses, UDDx_n is defined as follows. For a desired evolution time T, apply X pulses at times t_j given by

$$t_j = T \sin^2\left(\frac{j\pi}{2n+2}\right) \tag{A8}$$

for $j = 1, 2, \ldots, n$ if n is even and $j = 1, 2, \ldots, n+1$ if n is odd. Hence, UDDx_n always uses an even number of pulses—n when n is even and $n+1$ when n is odd—so that, when no noise or errors are present, UDDx_n faithfully implements a net identity operator.

b. Ideal QDD

To define QDD, it is useful to instead define UDDx_n in terms of the pulse intervals, $\tau_j = t_j - t_{j-1}$. By defining the normalized pulse interval,

$$s_j = \frac{t_j - t_{j-1}}{t_1} = \sin\left(\frac{(2j-1)\pi}{2n+2}\right)\csc\left(\frac{\pi}{2n+2}\right) \tag{A9}$$

for $j = 1, 2, \ldots, n+1$, we can define UDDx_n over a total time T,

$$\mathrm{UDDx}_n(T) \equiv f_{s_1 T} - X - \cdots - f_{s_n T} - X - f_{s_{n+1} T} - X^n, \tag{A10}$$

where the notation X^n means that the sequence ends with X (I) for odd (even) n. From this, $\mathrm{QDD}_{n,m}$ has the recursive definition

$$\mathrm{QDD}_{n,m} \equiv \mathrm{UDDx}_m(s_1 T) - Y - \mathrm{UDDx}_m(s_n T) - \cdots - \mathrm{UDDx}_m(s_n T) - Y - \mathrm{UDDx}_m(s_{n+1} T) - Y^m. \tag{A11}$$

This means that we implement UDDy_n (the outer Y pulses) and embed an mth-order UDDx_m sequence within the free evolution periods of this sequence. The inner UDDx_m sequences have a rescaled total evolution time $s_k T$; since the decoupling properties depend only on τ_j (and not the total time), we still obtain the expected inner cancelation. Written in this way, the total evolution time of $\mathrm{QDD}_{n,m}$ is $S_n^2 T$, where $S_n = \sum_{j=1}^{n+1} s_j$.

To match the convention of all other sequences presented, we connect this definition to one in which the total evolution time of $\mathrm{QDD}_{n,m}$ is itself T. First, we implement the outer UDDy_n sequence with Y pulses placed at times t_j according to Eq. (A8). The inner X pulses must now be applied at times

$$t_{j,k} = \tau_j \sin^2\left(\frac{k\pi}{2m+2}\right) + t_{j-1}, \tag{A12}$$

where $j = 1, 2, \ldots, n$ if n is even (or $j = 1, 2, \ldots, n+1$ if n is odd), with a similar condition for k up to m if m is even (or $m+1$ if m is odd). Note that, when m is odd, we end each inner sequence with an X, and then the outer sequence starts where a Y must be placed simultaneously. In these cases, we must apply a $Z = XY$ pulse (ignoring the global phase, which does not affect DD performance). Hence, we must apply rotations about X and Z when m is odd and X and Y when m is even. This can be avoided by instead redefining the inner (or outer) sequence as Z when

m is odd and then combining terms to get X and Y again. In this work, we choose the former approach. It would be interesting to compare this with the latter approach in future work.

c. UDD and QDD sequences with finite width pulses

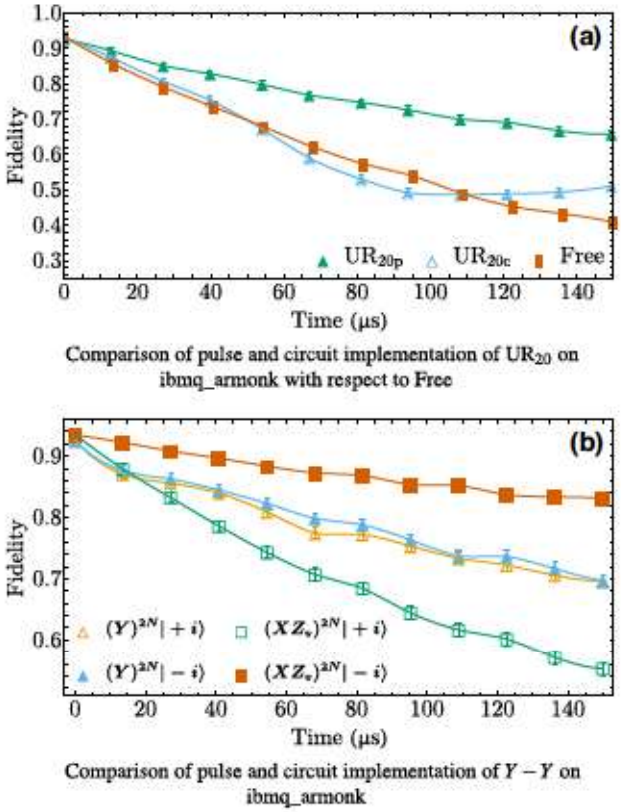
For real pulses with finite width Δ, these formulas must be slightly augmented. First, defining t_j is ambiguous since the pulse application cannot be instantaneous at time t_j. In our implementation, pulses start at time $t = t_j - \Delta$ so that they end when they should be applied in the ideal case. Trying two other strategies—placing the center of the pulse at t_j or the beginning of the pulse at t_j—did not result in a noticeable difference. Furthermore, X and Y have finite width (roughly $\Delta = 50$ ns). When UDDx_n is applied for even n, we must end on an identity, so the identity must last for a duration Δ, i.e., $I = f_\tau$. A similar timing constraint detail appears for $\mathrm{QDD}_{n,m}$ when m is odd. Here, we must apply a Z pulse, but on IBM devices, Z is virtual and instantaneous (see Appendix B). Thus, we apply $Z - f_\Delta$ to obtain the expected timings.

APPENDIX B: CIRCUIT VERSUS OPENPULSE APIS

We first tried to use the standard Qiskit circuit API [49,50]. Given a DD sequence, we transpiled the circuit in Fig. 2 to the respective device's native gates. However, as we illustrate in Fig. 6(a), this can lead to many advanced sequences, such as UR_{20}, behaving worse than expected. Specifically, this figure shows that implementing UR_{20} in the standard circuit way, denoted $\mathrm{UR}_{20\mathrm{c}}$ (where c stands for circuit), is substantially worse than an alternative denoted $\mathrm{UR}_{20\mathrm{p}}$ (where p stands for pulse).

The better alternative is to use OpenPulse [48]. We call this the "pulse" implementation of a DD sequence. The programming specifics are provided in Ref. [100]; here, we focus on the practical difference between the two methods with the illustrative example shown in Fig. 6(b). Specifically, we compare the $Yf_\tau Yf_\tau$ sequence implemented in the circuit API and the OpenPulse API.

Under OpenPulse, the decay profiles for $|+i\rangle$ and $|-i\rangle$ are roughly identical, as expected for the $Yf_\tau Yf_\tau$ sequence. The slight discrepancy can be understood as arising from coherent errors in state preparation and the subsequent Y pulses, which accumulate over many repetitions. On the other hand, the circuit results exhibit a large asymmetry between $|i\rangle$ and $|-i\rangle$. The reason is that Y is compiled into $Z_v - X$, where Z_v denotes a virtual-Z gate [101]. As Fig. 6(b) shows, $Z_v - X$ does *not* behave like Y. The simplest explanation consistent with the results is to interpret Z_v as an instantaneous Z. In this case, $Z|+i\rangle = |-i\rangle$ and the subsequent X rotates the state from $|-i\rangle$ to $|+i\rangle$ by rotating *through the excited state.* The initial state $|-i\rangle$, on the other hand, rotates *through the ground state.* Since the ground state is much more stable than the excited state on IBMQ's transmon devices, this asymmetry in trajectory on the Bloch sphere is sufficient to explain the asymmetry in fidelity [102].



Comparison of pulse and circuit implementation of UR_{20} on ibmq_armonk with respect to Free

Comparison of pulse and circuit implementation of $Y - Y$ on ibmq_armonk

FIG. 6. Comparison of the circuit and OpenPulse API approaches to implementing (a) UR_{20} and (b) $Yf_\tau Yf_\tau$. Panel (a) demonstrates that the OpenPulse version is substantially better. This is partially explained by (b). When using OpenPulse, the $Yf_\tau Yf_\tau$ sequence behaves as expected by symmetrically protecting $|\pm i\rangle$, in stark contrast to the circuit implementation, which uses virtual-Z gates, denoted Z_v.

Taking a step back, every gate that is not a simple rotation about the x axis is compiled by the standard circuit approach into one that is a combination of Z_v, X, and $\sqrt{X}$. These gates can behave unexpectedly, as shown here. In addition, the transpiler—unless explicitly instructed otherwise—also sometimes combines a Z_v into a global phase without implementing it right before an X gate. Consequently, two circuits can be logically equivalent while implementing different DD sequences. Using OpenPulse, we can ensure the proper implementation of $(\pi)_\phi$. This allows the fidelity of $\mathrm{UR}_{20\mathrm{p}}$ to exceed that of $\mathrm{UR}_{20\mathrm{c}}$.

Overall, we found (not shown) that the OpenPulse implementation was almost always better than or comparable to the equivalent circuit implementation, except for the XY4 sequence on ibmq_armonk, where the $\mathrm{XY4}_\mathrm{c}$ sequence was substantially better than the $\mathrm{XY4}_\mathrm{p}$ sequence. However, the $\mathrm{XY4}_\mathrm{p}$ sequence was not the top-performing

sequence. Hence, it seems reasonable to default to using OpenPulse for DD when available.

APPENDIX C: METHODOLOGIES FOR EXTRACTION OF FIDELITY METRICS

In the Pauli demonstration, we compare the performance of different DD sequences on the six Pauli eigenstates for long times. More details of the method are discussed in Sec. III A and, in particular, Figs. 1 and 2. The results are summarized in Fig. 3 and in more detail in Appendix E.

To summarize the fidelity decay profiles, we have chosen to employ an integrated metric in this work. Namely, we consider a time-averaged (normalized) fidelity,

$$F(T) \equiv \frac{1}{f_e(0)}\langle f_e(t)\rangle_T = \frac{1}{T}\int_0^T dt \frac{f_e(t)}{f_e(0)}. \tag{C1}$$

We combine this metric with an interpolation of fidelity curves, which we explain in detail in this appendix. We call the combined approach "interpolation with time averaging" (ITA).

Past work has employed a different method for comparing DD sequences, obtained by fitting the decay profiles to a modified exponential decay function. That is, to assume that $f_e(t) \sim e^{-\lambda t}$, and then perform a fit to determine λ. For example, in previous work [41], some of us chose to fit the fidelity curves with a function of the form [103]

$$f_P(t) = \frac{f_e(T_f) - f_e(0)}{\Gamma(T_f) - 1}[\Gamma(t) - 1] + f_e(0), \tag{C2a}$$

$$\Gamma(t) \equiv \tfrac{1}{2}(e^{-t/\lambda}\cos(t\gamma) + e^{-t/\alpha}), \tag{C2b}$$

where $f_e(0)$ is the empirical fidelity at $T = 0$, $f_e(T_f)$ is the empirical fidelity at the final sampled time, and $\Gamma(t)$ is the decay function that is the subject of the fitting procedure, with the three free parameters λ, γ, and α. This worked well in the context of the small set of sequences studied in Ref. [41].

As we show below, in the context of our present survey of sequences, fitting to Eq. (C2a) results in various technical difficulties, and the resulting fitting parameters are not straightforward to interpret and rank. We avoid these technical complications by using our integrated (or average fidelity) approach, and the interpretation is easier to understand. We devote this appendix to primarily explaining the justification of these statements, culminating in our preference for a methodology based on the use of Eq. (C1).

First, we describe how we bootstrap to compute the empirical fidelities (and their errors) that we use at the beginning of our fitting process.

1. Pointwise fidelity estimate by bootstrapping

We use a standard bootstrapping technique [104] to calculate the 2σ (95% confidence intervals) errors on the empirical Uhlmann fidelities,

$$f_e(t) = |\langle\psi|\rho_{\text{final}}(t)|\psi\rangle|^2. \tag{C3}$$

To be explicit, we generate $N_s = 8192$ binary samples (aka shots) from our demonstration (see Fig. 2) for a given {DD sequence, state preparation unitary U, total DD time T} 3-tuple. From this, we compute the empirical Uhlmann fidelity as the ratio of counted 0's normalized by N_s. We then generate 1000 resamples (i.e., N_s shots generated from the empirical distribution) with replacement, calculating $f_e(T)$ for each resample. From this set of 1000 $f_e(T)$'s, we compute the sample mean, $\langle f_e\rangle_T$, and sample standard deviation, σ_T, where the T subscript serves as a reminder that we perform this bootstrapping for each time point. For example, the errors on the fidelities in Fig. 1(a) are 2σ errors generated from this procedure.

2. A survey of empirical fidelity decay curves

Given a systematic way to compute empirical fidelities through bootstrapping, we can now discuss the qualitatively different fidelity decay curves we encounter in our demonstrations, as illustrated in Fig. 7. At a high level, the curves are characterized by whether they decay and whether oscillations are present. If decay is present, there is an additional nuance about what fidelity the curve decays to and whether there is evidence of saturation, i.e., reaching a steady state with constant fidelity. Finally, it matters whether an oscillation is simple, characterized by a single frequency, or more complicated. All these features can be seen in the eight examples shown in Fig. 7, which we now discuss.

The first four panels in Fig. 7 correspond to curves dominated by decay, but that decay to different final values. For the first two RGA_{8a} plots, the final value seems stable (i.e., a fixed point). For CPMG and Free, the final fidelity reached does not seem to be the projected stable fidelity. The "Free, $|0\rangle$" curve does not decay, consistent with expectations from the stable $|0\rangle$ state on a superconducting device. The last three plots show curves with significant oscillations. For the $\text{QDD}_{1,2}$ plot, the oscillations are strong and only weakly damped. For CPMG, the oscillations are strongly damped. Finally, the KDD plot is a pathological case where the oscillations clearly exhibit more than one frequency and are also only weakly damped.

3. Interpolation versus curve fitting for time-series data

To obtain meaningful DD sequence performance metrics, it is essential to compress the raw time-series data

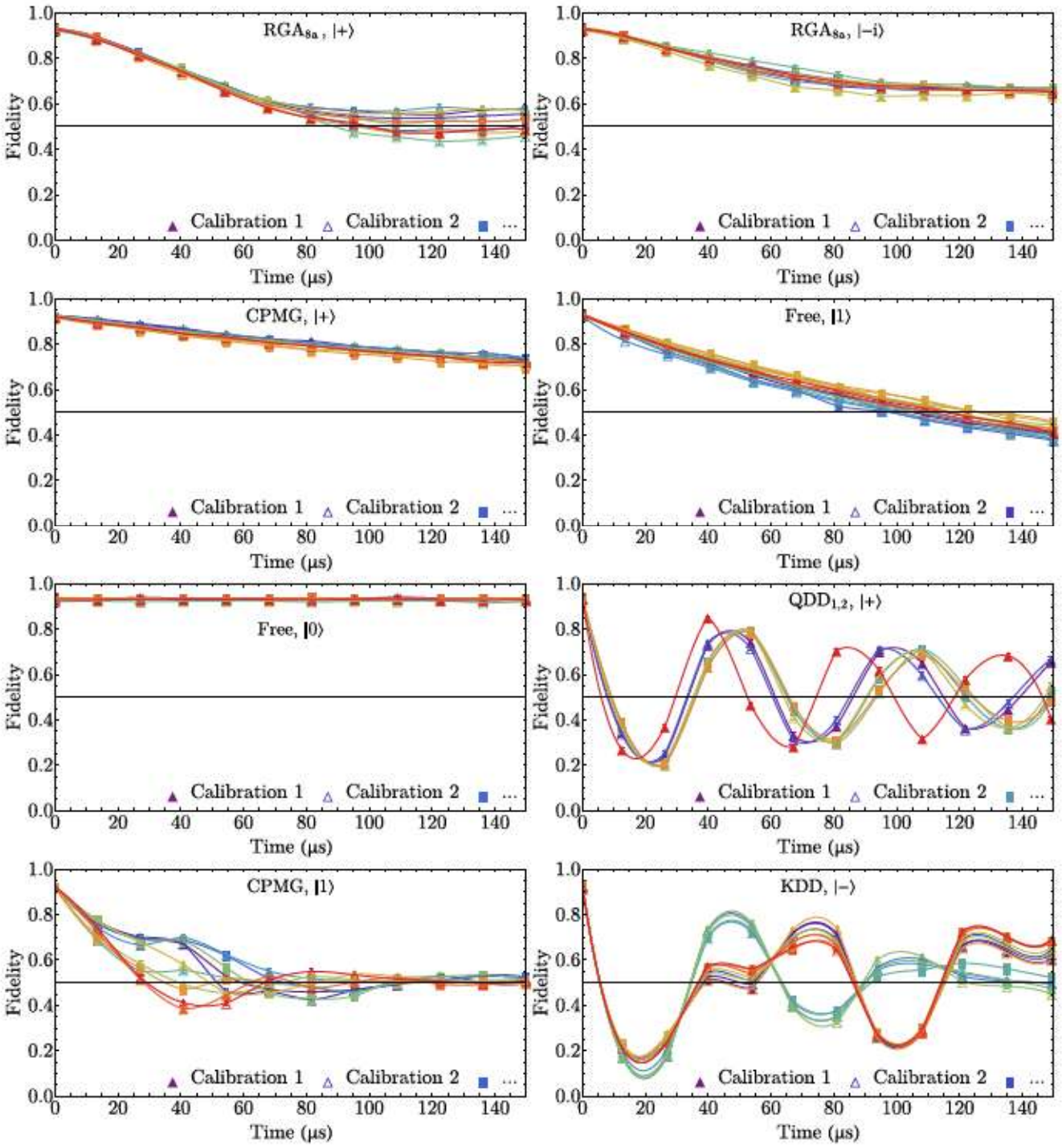


FIG. 7. A sample of fidelity decay curves from ibmq_armonk that showcases the qualitatively different curve types. Each plot shows curves generated from ten different calibration cycles. Note that all Pauli states are sampled within the same job, so data can be compared directly without worrying about system drift across different calibrations.

into a compact form. Given a target protection time T, the most straightforward metric is the empirical fidelity, $f_e(T)$. Given a set of initial states relevant to some demonstration (i.e., those prepared in a specific algorithm), one can then bin the state-dependent empirical fidelities across the states in a box plot.

The box plots we present throughout this work are generated using *Mathematica*'s `BoxWhiskerChart` command. As a reminder, a box plot is a common method to summarize the quartiles of a finite data set. In particular, let Q_x represent the xth quantile defined as the smallest value contained in the data set for which x% of the data lies

below the value Q_x. With this defined, the box plot shows Q_0 as the bottom bar (aka the minimum), $Q_{25} - Q_{75}$ as the orange box (aka the inner-quartile range), Q_{50} as the small white sliver in the middle of the inner-quartile range, and Q_{100} as the top bar (aka the maximum). In our box plots, we have also included the mean as the solid black line. A normal distribution is symmetric, so samples collected from a normal distribution should give rise to a box plot that is symmetric about the median and where the mean is approximately equal to the median.

If there is no predefined set of states of interest, it is reasonable to sample states from the Haar distribution, as we did for Fig. 4. This is because the sample Haar mean is an unbiased estimator for the mean across the entire Bloch sphere. Indeed, for a large enough set of Haar-random states (we found 25 to be empirically sufficient), the distribution about the mean becomes approximately Gaussian. At this point, the mean and median agree, and the inner-quartile range becomes symmetric about its mean, so one may choose to report $\langle f_e(T)\rangle \pm 2\sigma$ instead.

When there is no target time T, we would like a statistic that accurately predicts performance across a broad range of relevant times. The empirical fidelity for a given state at any fixed T is an unreliable predictor of performance due to oscillations in some curves (see, e.g., $\mathrm{QDD}_{1,2}$, CPMG, and KDD in the last three plots in Fig. 7). To be concrete, consider that, for $\mathrm{QDD}_{1,2}$ protecting $|+\rangle$, $\langle f_e(20\ \mu\mathrm{s})\rangle \approx 0.2$, whereas $\langle f_e(50\ \mu\mathrm{s})\rangle \approx 0.8$. The standard statistic, in this case, is a decay constant obtained by a modified exponential fit [e.g., Eq. (C2a)] [41].

So far, our discussion has centered around a statistic that captures the performance of individual curves in Fig. 7, i.e., a curve for a fixed DD sequence, state, and calibration. To summarize further, we can put all found λ values in a box plot. For simplicity, we call any method that fits individual curves and then averages over these curves a "fit-then-average" (FTA) approach. This is the high-level strategy we advocate for and use in our work. Note that we include interpolation as a possible curve-fitting strategy as part of the FTA approach, as well as fitting to a function with a fixed number of fit parameters. In contrast, Ref. [41] utilized an "average-then-fit" (ATF) approach, wherein an averaging of many time series into a single time series was performed before fitting [105]. Only fitting to a function with a fixed number of fit parameters was used in Ref. [41], but not interpolation. We discuss the differences between the FTA and ATF approaches below, but first, we demonstrate what they mean in practice and show how they can give rise to different quantitative predictions. We begin our discussion by considering Fig. 8.

The data in Fig. 8 correspond to the Pauli demonstration for free evolution (Free). On the left, we display the curves corresponding to each of the 96 demonstrations (six Pauli eigenstates and 16 calibration cycles in this case). The top curves are computed by a piecewise cubic spline interpolation between successive points, while the bottom curves are computed using Eq. (C2a) (additional details regarding how the fits were performed are provided in Appendix C 9). For a given state, the qualitative form of the fidelity decay curve is consistent from one calibration to the next. For $|0\rangle$, there is no decay. For $|1\rangle$, there is a slow decay induced from T_1 relaxation back to the ground state (hence the fidelity dropping below $1/2$). For the equatorial states, $|\pm\rangle$ and $|\pm i\rangle$, there is a sharp decay to the fully mixed state (with fidelity $1/2$) accompanied by oscillations with different amplitude and frequencies depending on the particular state and calibration. The $|0\rangle$ and $|1\rangle$ profiles are entirely expected and understood, and the equatorial profiles can be interpreted as being due to T_1 (relaxation) and T_2 (dephasing) alongside coherent pulse-control errors that give rise to the oscillations. On the right, we display the fidelity decay curves computed using the ATF approach, i.e., by averaging the fidelity from all demonstrations at each fixed time. The two curves [top, interpolation; bottom, Eq. (C2a)] display a simple decay behavior, which is qualitatively consistent with the curves in Fig. 2 of Ref. [41].

In Appendix C 4, we go into more detail about what the approaches in Fig. 8 are in practice and, more importantly, how well they summarize the raw data. Before doing so, we comment on an important difference between the two left panels of Fig. 8. Whereas the interpolation method provides consistent and reasonable results for all fidelity decay curves, the fitting procedure can sometimes fail. A failure is not plotted, and this is why some data in the bottom right panel are missing. For example, most fits (15/16) for $|0\rangle$ fail since Eq. (C2a) is not designed to handle flat "decays." State $|1\rangle$ also fails 11/16 times, but interestingly, the equatorial states produce a successful fit all 16 times. The nature of the failure is explained in detail in Appendix C 9, but as a prelude, a fit fails when it predicts fidelities outside the range $[0, 1]$ or when it predicts a decay constant λ with an unreasonable uncertainty. We next address the advantages of the interpolation approach from the perspective of extracting quantitative fidelity metrics.

4. ITA versus curve fitting for fidelity metrics

To extract quantitative fidelity metrics from the fitted data, we compute the time-averaged fidelity [Eq. (C1)] and the decay constant λ [Eq. (C2a)]. The results are shown in Fig. 9. The box plots shown in this figure are obtained from the individual curve fits in Fig. 8. The top panel is the ITA approach we advocate in this work (recall Appendix C): it corresponds to the time-averaged fidelity computed from the interpolated fidelity curves in the top left panel of Fig. 8. The bottom panel corresponds to fits computed using Eq. (C2a), i.e., the bottom left panel of Fig. 8, from which the decay constant λ is extracted.

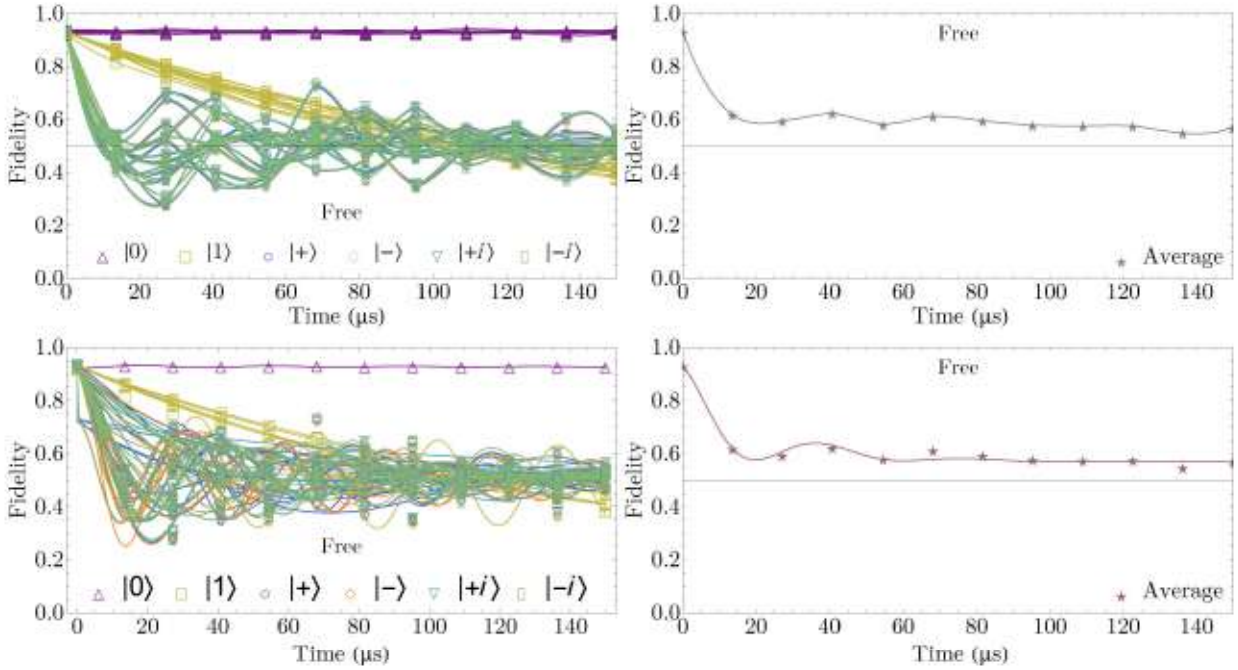


FIG. 8. Illustration of the qualitative difference between different fitting procedures with free evolution (Free) data collected on ibmq_armonk. The empirical data (obtained from six different initial states and 16 calibration cycles, for a total of 96 time series shown) are *identical* in the top left and bottom left panels (the open symbols). We employ two different techniques [interpolation (top) and Eq. (C2a) (bottom)] to fit the data corresponding to individual time series (left) and averaged time series (right). Top left: curves are computed by interpolation, separately for each time series. Bottom left: curves are computed by fitting each time series to Eq. (C2a). The fits here must contend with three different types of behavior: constant (no decay: $|0\rangle$), pure and slow decay ($|1\rangle$), and damped oscillations (the remaining equatorial states). Right: the ATF approach. We first average the fidelity across all the data for each fixed time and then fit it. Top: we utilize a piecewise cubic spline interpolation between points. Bottom: we use the fit given by Eq. (C2a). We remark that fitting to Eq. (C2a) fails for some data sets, and this is why the bottom left panel contains fewer curves than the top left panel. This is explained more in the text and in full detail in Appendix C 9.

Let us first discuss the bottom panel of Fig. 9. First, we again comment on the effect of fit failures. Since $|0\rangle$ does not decay, Eq. (C2a) is not appropriate (i.e., $\lambda \to \infty$, which is not numerically stable), and this results in only 1/16 fits leading to a valid λ prediction. This is insufficient data to generate a box plot; hence, the absence of the $|0\rangle$ data at the bottom. Similar numerical instability issues—though less severe—arise for all of the fits. For example, for $|1\rangle$, only 5/16 fits succeed. Among those that do succeed, the variation in λ is quite large, varying in the range [103, 346]. When compared to the raw data in Fig. 8, such a large variation should not be expected. In contrast, all the fits for the equatorial states succeed, but the variation in λ is again relatively large, in the range [17, 146]. To summarize, fitting Eq. (C2a) to our data has instability issues that manifest as (i) failures to fit some data sets and (ii) large variations in the reported λ that are unphysical. The problem is not specific to Eq. (C2a) and indeed would likely occur for any function that tries to reasonably model the entire set of curves found in practice, as in Fig. 7. Again, this is because (i) not all states decay as an exponential (i.e., the decay if $|0\rangle$ is flat and that of $|1\rangle$ appear roughly linear), (ii) λ and γ are not independent, and (iii) the notion of λ itself depends on the final expected fidelity, which is state dependent.

We argue that instead the approach of interpolating and using time-averaged fidelities, i.e., the approach shown in the top panel of Fig. 9, is preferred. The results shown there demonstrate that this method gives consistent and reasonable results for any fixed state. When significant variations are present (as in the entire data set), it is representative of the clear difference between initial states visible in Fig. 8 (top left). The key is to choose an appropriate value of T to average over. When T is too small, it does not capture the difference between sequences, and when T is too large, the difference between most DD sequences becomes unobservable. As a compromise, we choose T long enough for oscillatory sequences to undergo at least one but up to a few oscillation periods.

Beyond the box-plot variation, it is also worth remarking that the FTA mean F of 75 μs in Fig. 9 (top; blue dashed line) is equal to the ATF F value for the interpolation approach (top; green dash-dot line), i.e., the averaging and fitting operations commute in this sense. Hence, the

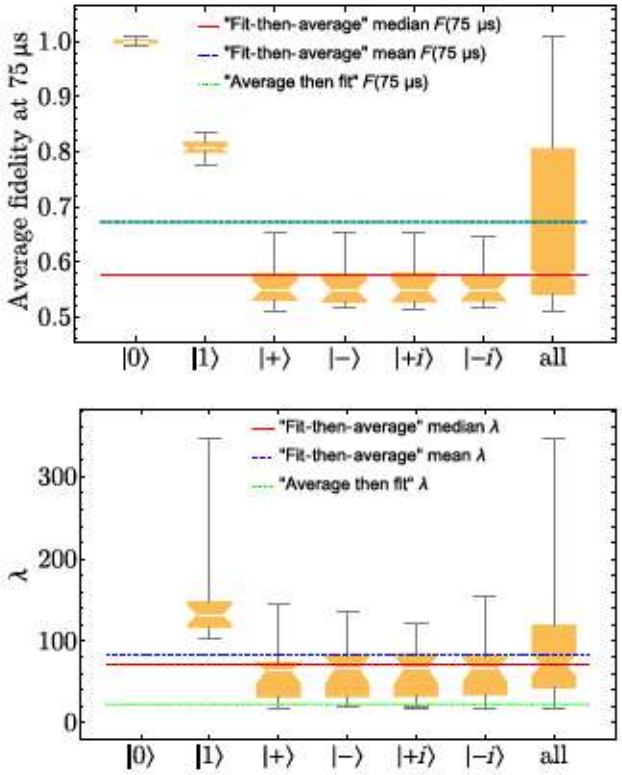


FIG. 9. Summary of the integrated and fitting approaches when applied to the Free curves in Fig. 8. For both of the resulting metrics, we display a state-by-state box plot showing the variation in the reported statistic over the 16 calibration cycles. Top: for each of the 6×16 interpolated curves, we compute a time-averaged fidelity at $T = 75$ μs, using Eq. (C1). These average fidelities are shown in box-plot format, separated by state and also combined together. Bottom: we attempt to extract λ for each of the 6×16 ($|0\rangle$ fails 15/16 times and $|1\rangle$ fails 11/16 times) curves fitted to Eq. (C2a) and shown in Fig. 8. The results are shown in box-plot format, again separated by state and also combined together. Top and bottom: also shown are the median (red line) and mean (blue dashed line) of these FTA results. The green dashed line is the ATF result (in this case, there is just a single number, and hence no variation over calibration cycles or states). The FTA (blue) and ATF (green) curves agree in the top panel, as expected of a reasonable method, whereas they disagree significantly in the bottom panel.

ITA approach has the nice property that we can recover the coarse-grained ATF-integrated fidelity result from the granular box-plot data by taking a mean. The same is not true of fitting where the two mean values of λ differ, as in the bottom panel of Fig. 9 (dashed blue and green lines). For the interpretation issues of fitting explained in Appendices C 5 and C 6 below, alongside the practical reasons explained here, we conclude that the ITA method we use in this work is preferred to methods that attempt to directly extract the decay constant from curve fitting to functions such as $f_P(t)$. The ITA method is more versatile and yields more stable and sensible results. Notably, it is also granular and able to capture the behavior of individual (DD sequence, state) pairs. This granularity allows for finer comparisons to theory (as discussed in Sec. II E) and also leads to practical benefits in our DD design, as seen in Appendix B.

5. The "ambiguous λ" problem and its resolution with ITA

In the previous subsection, we established that the technical difficulties encountered when using FTA as compared to ATF are resolved by interpolating and time averaging the fidelity [ITA, Eq. (C1)], instead of averaging and then fitting to a decay profile [Eq. (C2a)]. In this subsection, we provide further arguments in favor of the ITA approach.

First, we take a step back and discuss what we call the "ambiguous λ" problem when attempting to fit our data using standard models such as Eq. (C2a). Practically, this problem makes λ an unreliable estimator of DD sequence performance without additional information, and this additional information does not lead to a simple ranking. As we discuss below, this problem is generic and persists whether we consider individual decay curves or averaged curves. We argue that our time-averaged fidelity metric is a reliable estimator of performance for the top sequences and resolves this issue. After this practical ranking consideration, we consider the statistical meaning of the FTA and ATF approaches in the context of current noisy quantum devices and again argue in favor of the FTA approach.

The crux of the issue lies in the desired interpretation of the final statistic. In the ATF approach, the final decay constant λ gives an estimate of how well DD does on average over an ensemble of demonstrations. While this might be an appropriate metric for some benchmarking demonstrations, it is not typical of any given memory demonstration or algorithm. Here, we are interested in how well we expect a DD sequence (or free evolution) to preserve an unknown but fixed input state $|\psi\rangle$ in any given demonstration, and as we show below, ITA is better at addressing this "typical behavior" question.

As we mentioned, there is an "ambiguous λ problem" when we try to interpret the meaning of the decay constant in Eq. (C2a). In particular, the notion of decay in this equation is not just a function of λ but also of $f_e(T_f) - f_e(0)$ and γ, and these fit parameters are not mutually independent. To make this concrete, consider the four (artificial example) curves shown in Fig. 10 (top panel). These curves do not preserve fidelity equally, and yet, by construction, they all have decay time $\lambda = 50$ μs. Nevertheless, the time-averaged fidelity, $F(75\ \mu s)$, does distinguish between the four curves sensibly (middle panel). First, the time-averaged fidelity metric predicts that c_1 is the best curve and c_2 is the second best, which indeed seems reasonable from the curves alone. On the other hand, $c_3 > c_4$

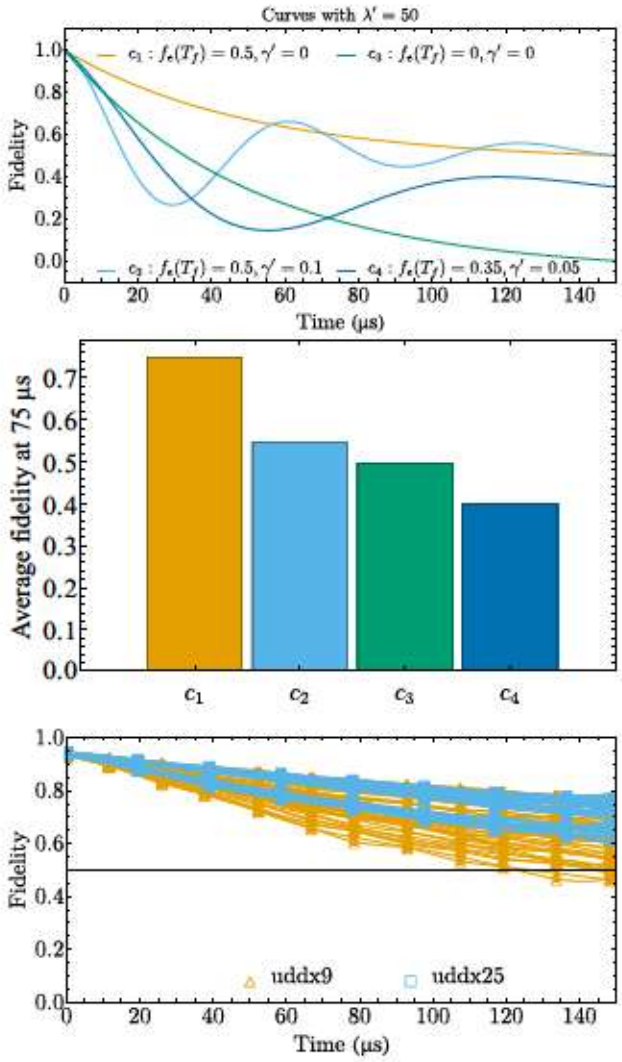


FIG. 10. Examples demonstrating the problem with using λ to rank performance. Top: four different artificially generated fidelity curves from Eq. (C2a) with the same λ value. The legend labels each curve with c_i and lists the value of γ and $f_e(T_f)$ used to generate it. Middle: the time-averaged fidelity evaluated at 75 μs for each curve. This metric differs for each curve. Bottom: the best sequences—such as ranks 1 and 10 in Fig. 3—do not exhibit oscillations across the six Pauli states.

is debatable and depends on the desired time scale. Had we chosen, e.g., $120 \leq T \leq 150$ μs, we would have found that $c_4 > c_3$. Such ambiguities are inevitable when oscillations of different frequencies are present with no preferred target time T. However, the best sequences do not have oscillations across the Pauli states, as we see by example in the bottom panel of Fig. 10. Thus, this time-averaged fidelity ambiguity for oscillatory curves does not affect the ranking of the top sequences, which we present in Fig. 3 as the main result of the Pauli demonstration. We remark that we choose to plot example curves with the same λ to avoid complications with fitting noisy data, but that the real data we have already encountered also faces the "ambiguous λ problem" due to noise.

6. Job-to-job fluctuations complicate comparing data from different jobs

We conclude our arguments in favor of the ITA approach by bringing up a subtle problem in the usage of cloud-based quantum devices. At a high level, users of cloud-based devices often have an implicit assumption that data taken within a fixed calibration cycle all come from roughly the same distribution. Hence, the approach of taking data for many states, averaging, and then fitting (the ATF approach) is sensible. In other words, the average over states is the Haar average computed on the device with the given backend properties. However, this is not generally true for all demonstrations. Some relevant quantum memory demonstrations (such as probing Free fidelities) violate this assumption, and data taken from one job are as if sampled from a different distribution if not appropriately handled. In this sense, we argue against naive averaging of data not taken within the same job. Whenever possible, we prefer to only make direct comparisons within a single job unless stability is empirically observed.

To justify this caution, we carefully test the veracity of standard *a priori* assumptions on a series of identical (procedurewise) Free demonstrations on ibmq_armonk. The Free demonstration procedure follows the standard DD protocol we established in Fig. 2. Namely, we prepare $|+\rangle$, idle for 75 μs by applying approximately 1000 identity gates, unprepare $|+\rangle$, and then measure in the computational basis. To run statistical tests on the result of this demonstration, we repeat it many times. Because IBM quantum computers operate on a queuing system (similar to a high-performance computing scheduler like Slurm), there are actually several different ways to repeat a demonstration, and as we will show, which way is chosen makes a difference. This is the reason for the rather pedantic discussion that follows.

The first notion of repeating a demonstration is to simply sample the same circuit many times by instructing the IBM *job* scheduler (we define "job" below) to use N_s shots. For example, the simplest possible job testing the above demonstration consists of sending ibmq_armonk the tuple (C, N_s), where C is the circuit encoding the above Free demonstration. Upon receiving such a tuple, ibmq_armonk samples C for N_s shots once the job reaches position one in the queue. This repetition is the same as discussed in Appendix C 1, and hence is the basis of estimating the empirical fidelity f_e [Eq. (C3)] by bootstrapping the N_s bitstrings. We are interested in the stability of f_e and hence the stability of repeating this entire procedure. Naively, we could repeatedly send ibmq_armonk the same tuple in M different jobs, i.e., $[J_1 = (C, N_s), J_2 = (C, N_s), \ldots, J_M =$

(C, N_s)], and compute M estimated empirical fidelities, $[f_1, f_2, \ldots, f_M]$, where we have dropped the e subscript for simplicity of notation. However, submitting demonstrations this way wastes substantial time waiting in the queue since J_1 and J_2 are not generally run contiguously due to other users also requesting jobs.

More generally, a *job* consists of a list of circuits that are sampled contiguously, i.e., without interruption by other users. For our purposes, each job J_k shall consist of L_k identical demonstration tuples, $J_k = [E_{k,1} \equiv (C_1, N_s)_{k,1}, E_{k,2} \equiv (C_2, N_s)_{k,2}, \ldots, E_{k,L_k} \equiv (C_k, N_s)_{k,L_k}]$, which allows us to compute L_k empirical fidelities, $R_k = [f_{k,1}, f_{k,2}, \ldots, f_{k,L_k}]$ (R_k standing for result k). Since $C_j = C$ for all j, the index on C_j is technically redundant, but keeping it helps us make our final point on the intricacies of collecting data within a job. In particular, ibmq_armonk samples the $L_k \times N_s$ bitstrings by running the circuits of J_k in the order $[C_1, C_2, \ldots, C_L]^{N_s}$ as opposed to $[C_1]^{N_s} \cdots [C_L]^{N_s}$, as one might naively expect. The goal of this strategy is to try and prevent device drift from biasing the results of $E_{k,1}$ compared to, e.g., E_{k,L_k}. With the notation defined, we can state exactly what tests we ran.

To generate our data set, we constructed 67 jobs $J_1, \ldots, J_{67}$ with a uniformly random value of $L_k \in [10, 75]$ (75 is the maximum demonstration allotment on ibmq_armonk), but an otherwise identical set of demonstrations $E_{k,l}$ and submitted them to the ibmq_armonk queue [106]. The value in choosing a random L_k is to check whether the duration of the job itself has any effect on results. We coarsely summarize the set of all job results in Fig. 11, which we call the "stats test data set." The entire data set was taken over a 10-h period over a fixed calibration cycle. In fact, 22 of the 67 jobs were run contiguously (i.e., one after the other without other user's jobs being run in between) [107]. Despite the jobs being localized in time, there are large jumps in average f_k values from job to job, which violates our *a priori* assumption of how sampling from a quantum computer should work within a fixed calibration cycle. We make this more precise in a series of assumptions and results that support or reject the given assumption.

Assumption 1.—Fidelities collected within a fixed job are normally distributed. Using the above notation, samples within $R_k = [f_{k,1}, \ldots, f_{k,L}]$ are drawn from a fixed normal distribution, i.e., $f_{k,l} \sim \mathcal{N}(\mu_k, \sigma_k)$ for some unknown mean and variance.

Result 1.—Assumption 1 is well supported by hypothesis tests (Jarque-Bera, Shapiro-Wilk tests) and visual tests (box plot, quantile-quantile plot).

The Jarque-Bera test is a means to test whether a sample of points has the skewness and kurtosis matching that expected of a normal distribution [108]. We implement it in *Mathematica* using the `JarqueBeraALMTest` command. The Shapiro-Wilk test checks whether a set of sample points has the correct order statistics matching that expected of a normal distribution [109]. We implement it using the *Mathematica* command `ShapiroWilkTest`. Both tests are hypothesis tests where the null hypothesis H_0 is that the data were drawn from a normal distribution and an alternative hypothesis H_a that it was not. Like most hypothesis tests, they are rejection-based methods. Namely, the tests return a p-value corresponding to the probability of H_0. If $p < 0.05$, we reject the null hypothesis with 95% confidence (this is the default setting for the *Mathematica* command that we employ). Otherwise, we say that the data do not support rejecting H_0—namely, it is reasonable that a normal distribution could have produced such a sample. We summarize the results of both tests in Table III.

From Table III, we see that among the 67 jobs, only three give us cause for concern. Namely, jobs {17, 22, 46} reject the null hypothesis under the Shapiro-Wilk test. From the

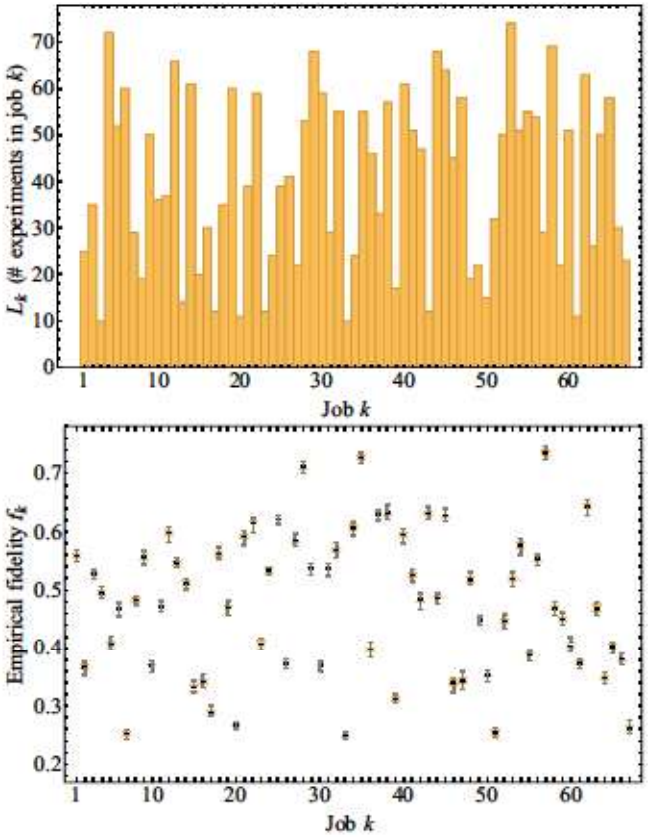


FIG. 11. A coarse view of the entire "stats test data set" taken on ibmq_armonk. Top: the random value of demonstrations, $L_k \in [10, 75]$, chosen for each job. Bottom: the spread of fidelities for each job summarized in 67 box plots. In our earlier notation, this is a box plot over the empirical fidelities, R_k. All data were taken within a fixed calibration cycle over a 10-h period, and many jobs (22/67) are contiguous, e.g., jobs 2 and 3 are run back to back with no interruption from other users' jobs. Despite this, there is a large variation in fidelity from job to job, which does not seem correlated with the job duration.

TABLE III. Summary of jobs where the null hypothesis that the data are sampled from a normal distribution is rejected with 95% confidence.

Test used	Job's k where H_0 rejected	p-values of rejected jobs
Jarque-Bera test	{}	{}
Shapiro-Wilk test	{17, 22, 46}	{0.028, 0.033, 0.048}

bar chart in Fig. 11, these jobs have {12, 59, 45} demonstrations, respectively. Since there are several other jobs with similar lengths whose data do not reject H_0, there does not seem to be a correlation between job length and rejection of normality.

We can also check the normality of the data visually using box plots or quantile-quantile (QQ) plots, as we do in Fig. 12. Recall that Q_x represent the xth quantile, i.e., the smallest value contained in the data set for which $x\%$ of the data lie below the value Q_x. The QQ plot is a way to compare the quantiles of two data sets. In our case, we compare the quantiles of the sample data set to that of a normal distribution with the same mean and standard deviation of the data. Namely, given samples $R_k = [f_1, \ldots, f_{L_k}]$, we compare the quantiles of R_k itself to $\mathcal{N}(\overline{R}_k, s_{R_k})$, where $\overline{R}_k$ is the mean of the data and σ_{R_k}

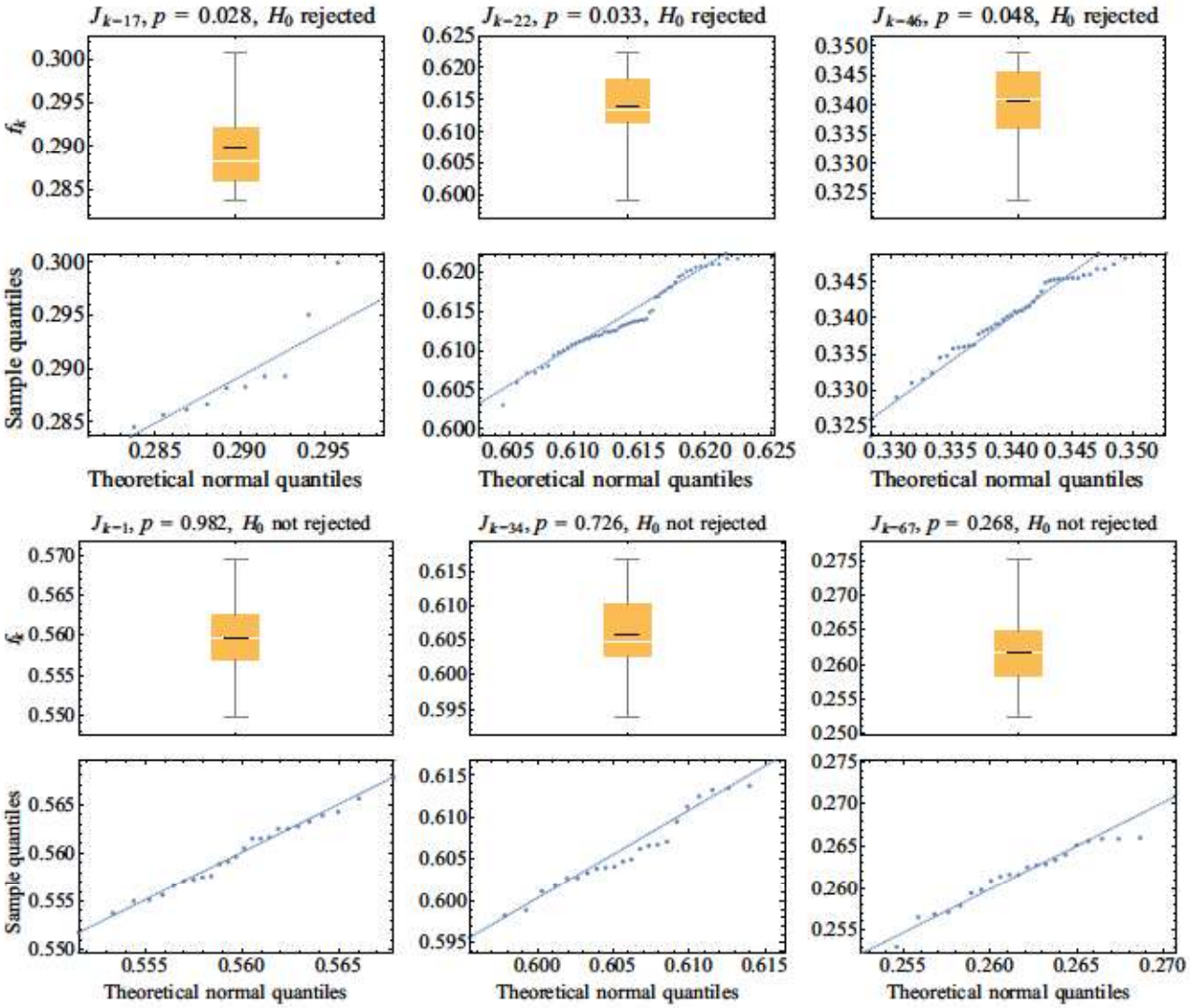


FIG. 12. Box plots and quantile-quantile (QQ) plots corresponding to different job data from Fig. 11. The top two rows show the three jobs that failed the Shapiro-Wilk test for normality. The bottom two rows show three jobs that did not fail the test and are otherwise chosen to be the first, middle, and last jobs in the set of data. Normality is characterized visually as (i) a symmetric box plot whose mean and median agree or (ii) a QQ plot whose data fall approximately along the diagonal. This is best exhibited by the $k = 1$ data in the third row. A deviation from normality is the negation of any of the above properties. For example, the $k = 17$ job in the first row departs from normality in all three respects.

is the uncorrected sample standard deviation, i.e., $s_{R_k} = ((1/N)\sum_t (f_t - \overline{R}_k)^2)^{1/2}$. Letting Q_x be the quantiles of our raw data and $\hat{Q}_x$ the quantiles of the corresponding normal distribution, then points on our QQ plots correspond to $(Q_x, \hat{Q}_x)$. Hence, if the two distributions are the same (or close) then the data should fall on the diagonal $Q_x = \hat{Q}_x$, which indicates normality.

We illustrate the use of these plots in Fig. 12. In the top two rows we exhibit the three example jobs that failed the Shapiro-Wilk test for normality. For $J_{k=17}$, the corresponding box plot is (i) not symmetric and (ii) has a mean that deviates significantly from its median. The QQ plot also deviates from a diagonal line, especially for larger fidelities. As we move from $k = 17$ to $k = 22$ and then $k = 46$, the data appear increasingly more normal, but still fail the Shapiro-Wilk test. The extent to which they fail the visual tests is best seen by considering a normal-looking example, which we can glean from the bottom two rows of plots, which were not rejected by either hypothesis test.

For the data shown, $J_{k=1}$ is the closest data set to normal. Here, $p = 0.982$ for the Shapiro-Wilk test, the box plot is symmetric, the mean and median agree, and the QQ plot data hardly deviate from the diagonal. Given this almost ideally normal data set, it is easier to see why the other data sets fail to appear as normal. For example, even those that pass hypothesis tests exhibit some non-normal features. But among those rejected, we see deviations from the diagonal line, skewness (a lack of symmetry in the box plot), and a discrepancy between the mean and the media.

Since most jobs (64/67) have data that pass hypothesis and visual normality tests, we conclude that Assumption 1 is reasonable.

Assumption 2.—Any demonstration within a fixed job is representative of any other identical demonstration within the same job. Hence, it is not necessary to repeat an identical demonstration within a fixed job. In other words, $f_{k,1} \pm \sigma_{k,1}$, where $\sigma_{k,1}$ is obtained by bootstrapping as in Appendix C 1, is a sufficient summary of the estimates μ_k and σ_k in $f_{k,j} \sim \mathcal{N}(\mu_k, \sigma_k)$.

Result 2.—Assumption 2 is justifying by a simple direct comparison of mean fidelity and standard deviation estimates. In particular, we compare (i) directly computing the sample mean and sample standard deviation and assuming normality, (ii) bootstrapping over the fidelities in R_k, and (iii) bootstrapping over demonstration output counts used to compute $f_{k,1}$. For all jobs (i.e., for all k), the three methods are consistent with each other since the 2σ error bars have significant overlap in all cases. See Fig. 13.

The basic idea is that all three methods are consistent with each other. By consistency, we mean the intuitive notion that the error bars have significant overlap with each other. Further rigor, in this case, is not necessary due to the degree of matching with respect to job-to-job fluctuations we discuss below. This consistency is significant because

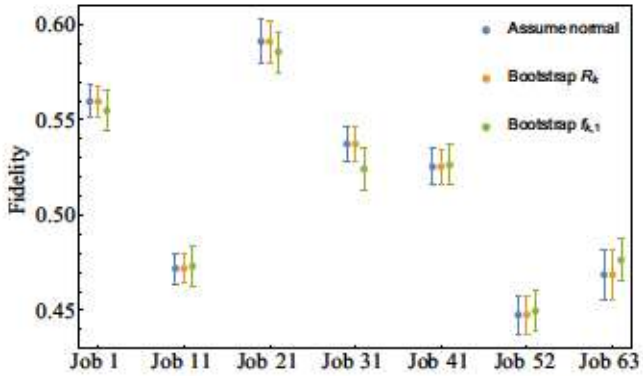


FIG. 13. We compare different approaches of summarizing the fidelities from running repeated identical demonstrations within a fixed job. In other words, we compare three ways to summarize $R_k = [f_1, \ldots, f_{L_k}]$. The first method, "assume normal," is to simply compute the sample average $\overline{f_k}$ and sample standard deviation s_k and then report $\overline{f_k} \pm 2s_k$ as the average fidelity. The second method, "bootstrap R_k," is to bootstrap over the fidelities in R_k. To correctly estimate the population standard deviation, note that we must rescale the bootstrapped mean error by a factor of $\sqrt{L_k}$ (see the text for more details). Finally, we can instead bootstrap over the counts used to compute $f_{k,1}$ directly, which we call "bootstrap $f_{k,1}$." This does not use the information for $f_{k,j}$ for $j > 2$ at all. Nevertheless, all three methods yield self-consistent results, as seen visually by the significant overlap in error bars between the three estimates. Note that we show only a subset of the results for all jobs to ensure that the error bars are large enough to be visible. For a sense of the consistency among the approaches, note that job 31 has the largest deviation. Hence, it is sufficient to use the bootstrap $f_{k,1}$ method as purported in Result 2.

the final method shown in Fig. 13 bootstraps only over the counts used to compute $f_{k,1}$. Hence, it is sufficient to run a demonstration only once within a single job and use the results as being representative of repeating the identical demonstration, thereby showing the claimed result.

We next detail the methods leading to the three summaries in Fig. 13. The first is to summarize R_k by the sample mean and sample standard deviation, $\overline{f_k} \pm 2s_k$, where $\overline{f_k} = (1/L_k)\sum_{t=1}^{L_k} f_t$ and $s_k = \sqrt{[1/(L_k - 1)]\sum_{t=1}^{n} (f_t - \overline{f_k})^2}$. By Result 1, it is appropriate to then characterize the sample with the first two moments—hence the name "assume normal." The second and third methods, called "bootstrap R_k" and "bootstrap $f_{k,1}$," both utilize the bootstrapping procedure discussed in Appendix C 1. The difference is in what is treated as the samples. In the R_k case, we bootstrap over the $f_{k,j}$ values using $n_s \equiv 10\,000$ L_k-sized samples. From these n_s estimates of $\overline{f_{kl}}$, we can estimate $\langle R_k \rangle = (1/n_s)\sum_{t=1}^{n_s} \overline{f_{kl}}$. We can then estimate the sample standard deviation $s_{R_k} = \sqrt{[1/(n_s - 1)]\sum_{t=1}^{n_s} (\overline{f_{kl}} - \langle R_k \rangle)^2}$. It is important to note that s_{R_k} here has the meaning of the standard deviation of

our estimate of the mean. As $L_k \to 0$, $s_{R_k} \to 0$, so s_{R_k} is *not* an estimate of σ_k as in some parametric distribution $\mathcal{N}(\mu_k, \sigma_k)$. To match the same meaning as the "assume normal" estimate, we report $\langle R_k \rangle \pm 2\sqrt{L_k} s_{R_k}$ as the "bootstrap R_k" estimate. In the $f_{k,1}$ case, we bootstrap over the counts used to compute $f_{k,1}$ in exactly the same way as discussed in Appendix C 1. Hence, this method does not utilize the demonstrations $f_{k,j}$ for $j > 2$, and the sample standard deviation is an estimate of σ_k and is consistent with "assume normal."

Assumption 3.—Given that R_k can be modeled as samples from $\mathcal{N}(\mu_k, \sigma_k)$, data in a different job R_p for $p \neq k$ but in the same calibration cycle can also be modeled as samples from $\mathcal{N}(\mu_k, \sigma_k)$.

Result 3.—Assumption 3 is not supported by the data we have already seen in Figs. 11–13. It is worth reiterating that this assumption does not hold even when the calibration is fixed or even when the jobs are contiguous.

In Fig. 11, we see box plots scattered wildly with median fidelities ranging from 0.24 to 0.75 and whose ranges do not overlap. Put more precisely, recall that, when "assuming normality" as in Fig. 13, we can simplify each job's results to a single estimate $\overline{f}_k \pm 2s_k$. Once we have done that, we find that $\min_k \overline{f}_k = 0.250$ and $\max_k \overline{f}_k = 0.734$ and yet $\max_k s_k = 0.013$, so it is not possible for all the data to be consistent as described in Assumption 3. This discrepancy is not resolved when considering jobs that are time proximate or contiguous, as we have discussed before. To see this, note that all jobs were taken within a 10-h window for a fixed calibration: the first nine jobs were taken within the first hour, the first three of which within the first 10 min. Despite this, there are large variations that violate Assumption 3 even within the first three jobs.

This result is the first major departure from expectations; it tells us that a demonstration must be repeated many times across different jobs to estimate the variance in performance from run to run. One consequence is that if we test the preservation of $|\psi\rangle$ in job 1 and $|\phi\rangle$ in job 2, it is not reliable to compare the fidelities $f(|\psi\rangle, T)$ and $f(|\phi\rangle, T)$ and draw conclusions about the results of a generic demonstration testing both $f(|\psi\rangle, T)$ and $f(|\phi\rangle, T)$. Alternatively, if we test the efficacy of two sequences s_1 and s_2 given the same state, but where the data are taken from different jobs, this is not a reliable indicator of their individual relative performance either. Instead, many demonstrations must be taken where the variance can be estimated, or direct comparisons should only be made within a fixed job and then repeated to check for the stability of the result.

The next two assumptions and results address to what extent data taken across different jobs, or even calibrations, can be modeled as being normally distributed.

Assumption 4.—It is appropriate to model $f_{k,1} \sim \mathcal{N}(\mu_C, \sigma_C)$ for some fixed calibration cycle C provided all jobs k are in C. In other words, demonstrations taken from different jobs can be viewed as sampled from a fixed normal distribution.

Result 4.—This assumption is justified since it is not rejected by the Jarque-Bera and Shapiro-Wilk hypothesis tests and is supported by visual indicators (box plot, QQ plot, and error bar comparison plot).

Let $A = \{f_{k,1}\}_{k=1}^{67}$ be the set comprising the first empirical fidelity from each job. Given this data set, neither the Jarque-Bera nor the Shapiro-Wilk test support rejecting the hypothesis that $f_{k,1} \sim \mathcal{N}(\mu_C, \sigma_C)$. More precisely, the p-values are 0.348 and 0.192, respectively, so with 95% confidence, we cannot reject the hypothesis.

In addition, we show the standard visual tests—box plot and QQ plot—in Fig. 14 alongside the error bar comparison plot similar to Fig. 13. The standard visual tests also support assuming normality since the box plot is symmetric, has a mean and median that almost agree, and the QQ plot data lies mostly along the diagonal. However, perhaps the most important evidence is the error bar comparison (bottom panel). By "assume normal," we compute the sample mean ($\overline{A}$) and standard deviation of σ_A of A as the estimates of μ_C and σ_C. We find that $\overline{A} = 0.48$ and $\sigma_A = 0.12$. Hence, we would report 0.48 ± 24 as the average fidelity value across the fixed calibration. When instead bootstrapping the samples of A, we again find 0.48 ± 0.24, which remarkably has symmetric error bars.

Hence, in all, it is reasonable to model the fidelity samples from different jobs as samples of a fixed normal distribution within a fixed calibration cycle. Note that we were careful not to try and model the entire data set as normal; instead, we held the number of samples from each job constant. Had we modeled the entirety of the data set, the Jarque-Bera and Shapiro-Wilk tests would reject normality. This is because some samples are unfairly given more weight, which is no longer normal [110].

Assumption 5.—Let $\hat{f}_c$ be empirical fidelity samples similar to $f_{k,1}$ from before but not necessarily sampled from a fixed calibration. Then $\hat{f}_c \sim \mathcal{N}(\mu, \sigma)$ even across different calibrations.

Result 5.—This assumption is justified for the free evolution (Free) data but is violated for some DD sequences such as KDD. Nevertheless, for average pointwise fidelity estimates of the form $\overline{f}_c \pm 2\sigma_c$, the difference between assuming normality or bootstrapping is negligible.

Here, we need to consider a different data set for the first time in this series of assumptions and tests. Instead of considering data from a fixed time $T = 75$ μs, we consider fidelity decay data as in Figs. 7 and 8. Namely, we consider the preservation of $|+\rangle$ and $|1\rangle$ under Free and KDD as a function of time. In each case, we consider the fidelity at 12 roughly equidistant time steps $f_c^{(s)}(|l\rangle, T_j)$; $T_j \approx \frac{150}{13}(j - 1)$ μs for $j = 1, \ldots, 12$. Here, s is a placeholder for the sequence (Free or KDD), c stands for the calibration at

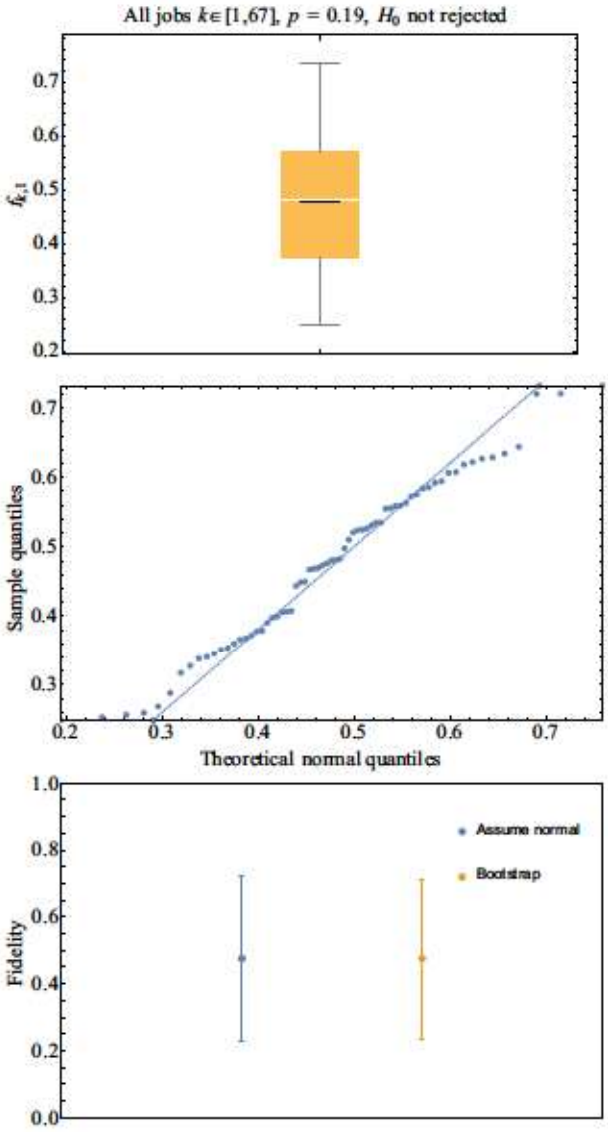


FIG. 14. We graph the visual normality tests for the data set consisting of the first fidelity obtained from each job, $A = \{f_{k,1}\}_{k=1}^{67}$. Alongside the fact that both the Jarque-Bera and Shapiro-Wilk tests do not reject normality, we can conclude that data set A does indeed appear to have been sampled from a normal distribution $\mathcal{N}(\mu_C, \sigma_C)$ by the same aforementioned box-plot and QQ-plot criteria. In the bottom panel, we show the difference between the "assume normal" procedure for data set A versus bootstrapping. Since both agree, it is not only reasonable to assume normality but also to use either method to make a consistent prediction for curve fitting.

which the fidelity was sampled, and l denotes the state. We test the normality for each set of fixed (s, l, j), which amounts to a set such as $\hat{f}_c$ in Assumption 4.

We begin by applying both the Jarque-Bera and Shapiro-Wilk hypothesis tests. We tabulate the cases where the hypothesis of normality fails with 95% confidence in Table IV. According to these tests, we can confidently claim that the set of KDD fidelity decay curves for $|+\rangle$

TABLE IV. Summary of Free and KDD decay curve data sets that violate the null hypothesis of normality with 95% confidence according to the Jarque-Bera (JB) or Shapiro-Wilk (SW) test. The data set consists of a set of fidelities $\{f_c^{(s)}(|l\rangle, T_j)\}_{c=1,\ldots,14}$, where s denotes a DD sequence, c denotes the calibration, $|l\rangle$ denotes the state, and j denotes the time point for $j = 1, \ldots, 12$. Note that we test normality where samples are drawn over different calibrations c with the other parameters fixed.

s	l	Rejected j by JB	Rejected j by SW
Free	$\|+\rangle$	{6}	{6}
Free	$\|1\rangle$	{2}	{2}
KDD	$\|+\rangle$	{}	{3,4,5,6,7,8,9}
KDD	$\|1\rangle$	{6}	{6,7,8}

does not obey all the properties we would expect for samples of a series of normal distributions. Namely, the times T_j for $j \in [3, 9]$ do not have the right order statistics since they fail the Shapiro-Wilk test. This seems reasonable given Fig. 7, in which the KDD data appear to be bimodal for the middle times.

We can also consider box plots for the data that support the claims made in Table IV. As an example, in Fig. 15 we show the box plot for (Free, $|1\rangle$), which agrees with normality at each time point, and (KDD, $|+\rangle$), which fails to appear normal at most time points.

Despite not passing the tests for normality, we can still formally pretend that the data are normal and compute the sample mean and variance as estimates of μ and σ. When we do this and compare the result to bootstrapping the mean and rescaling the confidence interval, the results are almost identical for both sets of data, as we can see in Fig. 16. By rescaling the confidence interval (CI), we mean that we compute the 95% CI of the mean estimate by bootstrapping, (Δ_l, Δ_u), and report $(\sqrt{C}\Delta_l, \sqrt{C}\Delta_u)$, where $\sqrt{C}$ is the number of calibration cycles (i.e., the number of fidelities) we bootstrap over. The rescaling is done to obtain an estimate of the spread of the underlying distribution and not of the sample mean. In other words, we observe that averaging via the bootstrap or by the simple sample mean and standard deviation agree even for pathological data of the form we obtain for KDD.

This observation seems to suggest that we can compare DD sequences by first averaging fidelities across calibrations. This would be in line with preview work such as Refs. [41,111] that did not report on the subtleties of job-to-job variation. As outlined in the main text and implied by the analysis presented in this appendix, we choose not to do this; the next subsection clarifies why and what we do instead.

7. Effect of job-to-job fluctuations

Let $f_c^{(s)}$ be a fidelity sample for state s and calibration c for some fixed time and sequence. We saw in Result 5

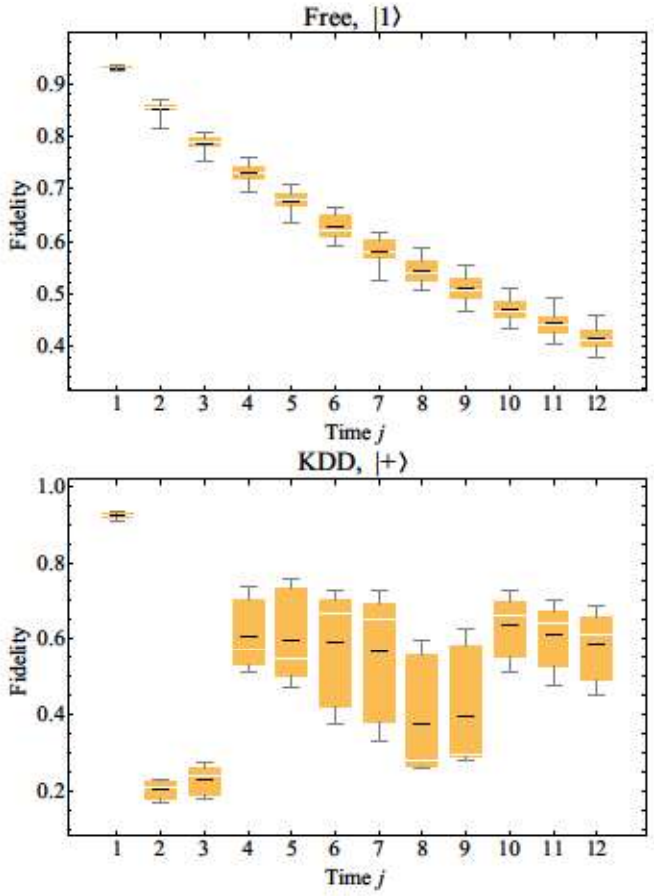


FIG. 15. We plot the fidelity decay of $|1\rangle$ under Free and $|+\rangle$ under KDD. The variation in the box plots is obtained by running identical demonstrations for that time across different calibrations. The Free data appear mostly normal at each time, which is corroborated by the hypothesis tests in Table IV. The KDD data do not appear normal for the intermediate times, in agreement with the hypothesis tests.

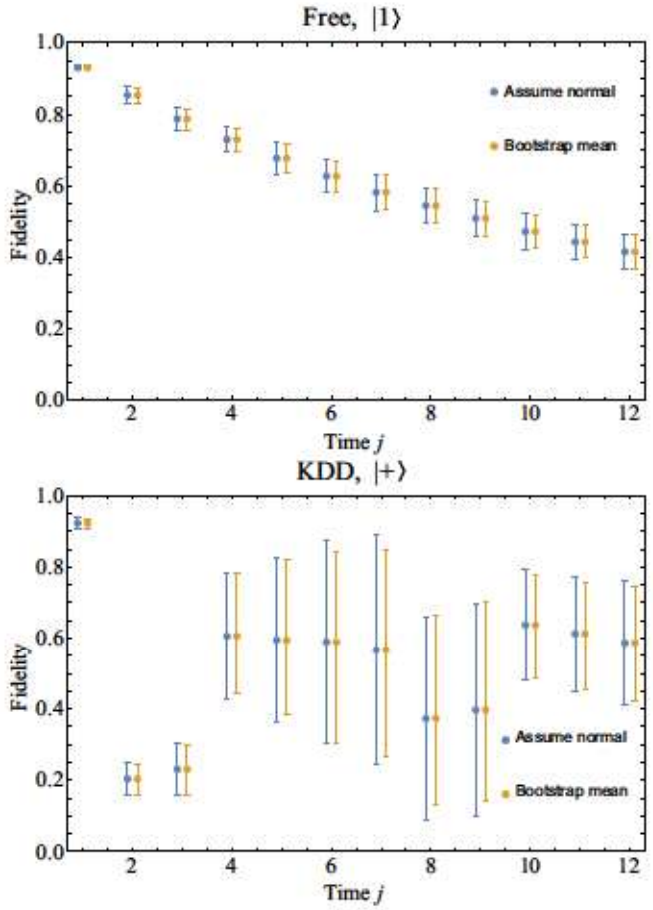


FIG. 16. We show that, regardless of whether the data formally appear normal, the reported mean and confidence interval are almost identical under this assumption or under bootstrapping. In particular, the top panel appears normal to various normality tests, whereas the bottom data do not. Despite this, reporting the sample mean and the standard deviation is almost identical to reporting the bootstrapped mean and confidence interval.

that assuming that $f_c^{(s)} \sim \mathcal{N}(\mu_s, \sigma_s)$ is not formally justified for all sequences and states. In particular, the $|+\rangle$ state decay for KDD does not pass hypothesis tests for normality at intermediate times. Nevertheless, if we are only concerned with the average fidelity of a fixed state s' across calibrations,

$$\langle f^{(s')} \rangle_N \equiv \frac{1}{N} \sum_{c=1}^{N} f_c^{(s')}, \tag{C4}$$

assuming normality or bootstrapping leads to consistent predictions. But related work—such as Refs. [41, 111]—was concerned with the average fidelity across states,

$$\overline{f}_{c'} \equiv \frac{1}{|S|} \sum_{s \in S} f_{c'}^{(s)}, \tag{C5}$$

for some set of states S and fixed calibration c'. If $f_c^{(s)} \sim \mathcal{N}(\mu_s, \sigma_s)$ and independent and identically distributed (IID) then $\overline{f}_c \sim \mathcal{N}(\mu, \sigma)$, where $\mu = (1/|S|) \sum_{s \in S} \mu_s$ and $\sigma = (1/|S|^2) \sum_{s \in S} \sigma_s$. Thus, we would expect the state-average fidelity to be consistent from job to job. In practice, however, this does not happen. For example, the Pauli averaged fidelity (i.e., choosing $S = \{|0\rangle, |1\rangle, |+\rangle, |-\rangle, |+i\rangle, |-i\rangle\}$) for different calibrations for the KDD sequence at time T_6 is given in Fig. 17.

The inconsistency of the average state fidelity as exhibited in Fig. 17 has several consequences. First, data from a single calibration are generally not enough to characterize typical behavior. For example, if we only sampled a single calibration and happened to sample $c = 6$, we would be left with the wrong impression of KDD's performance. Second, data from a single calibration are also generally not representative of average behavior. Indeed, the red dot

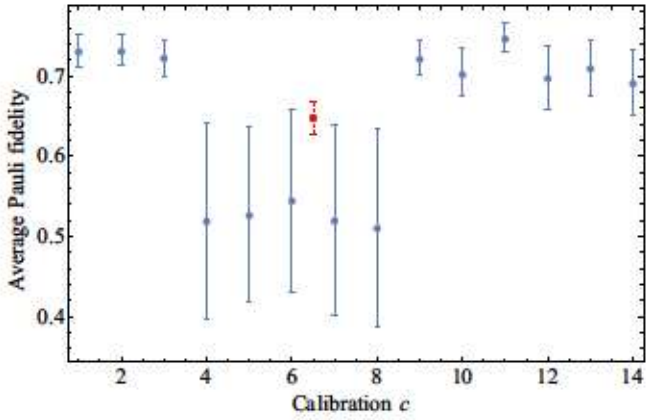


FIG. 17. The average fidelity over the six Pauli eigenstates across 14 calibrations under protection by the KDD sequence. Time is fixed to T_6, as defined in Fig. 15. The red point is the mean fidelity of the entire data set. This total average is not representative of any calibration average.

in the middle corresponds to the average over the 14 calibration averages, and no single calibration has a mean consistent with this average. Finally, averaging over states sampled in different jobs can lead to misleading results. For example, suppose that we queued up several runs of $|0\rangle$ in calibration $c = 1$, of $|1\rangle$ in $c = 2$, etc. Then, according to the average fidelities in Fig. 17, it is likely that a $|+\rangle$ run in $c = 3$ would have substantially higher fidelity than a $|-\rangle$ run in $c = 4$. Yet, if we compared $|+\rangle$ to $|-\rangle$ within the same calibration, they would have roughly the same fidelity.

To clarify this averaging subtlety, we present the raw data for each state as a box plot in Fig. 18. The variation arises from the fidelity fluctuations of each state across calibrations. Interestingly, the polar states, $|0\rangle$ and $|1\rangle$, give

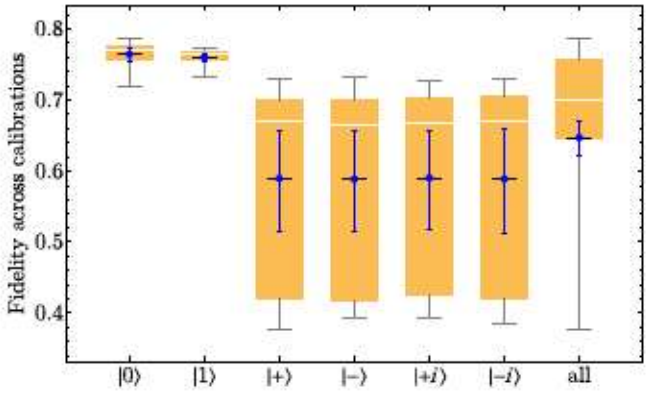


FIG. 18. The variation in fidelity across 14 calibrations for each Pauli state protected under KDD for a fixed time T_6. The final box compiles the data for all six Pauli states. The blue point shows the average (with 95% bootstrapped confidence intervals) across the data in each box plot.

surprisingly consistent results, while the remaining equatorial states have large variations in performance. This suggests that in some calibrations, the equatorial states were adversely affected, while the polar states were mostly unaffected. By checking the data calibration by calibration, we find this to indeed be true. Namely, the equatorial states collectively have a similar performance that is worse in some calibrations and better in others. This could happen, e.g., if, in some calibrations, T_2 dropped while T_1 stayed roughly the same. Thus, to have a direct comparison of state performance, it is imperative to confine the comparison to within a single job. The same can be said about comparing the fidelities of a fixed state generated at different times. Without confining comparisons to a single job, it is unknown if the difference in fidelities is due to drift or other causes.

On the other hand, it is not possible to test all DD sequences, states, and decay times within a short enough window to avoid significant drift. Even if it were, fair queuing enforces a maximum number of 75 circuits per job, so this sets a hard cutoff on what can be compared reliably in a time-proximate way. As a compromise between time-proximate comparisons and the hard 75 circuit-per-job limit, we collected data as described in Sec. III. Namely, within a single job, we collect the fidelity decay curves for the six Pauli states. Each curve consists of 12 time points. In total, this takes up 72 circuits, and we pad this with an additional set of two measurement error mitigation circuits. In this sense, we guarantee that the fidelities for different states and times can be compared reliably within this data set, i.e., they are not different due to drift. We then repeat this demonstration over different calibration cycles to obtain an estimate of fidelity differences due to drift. By reporting the median across this data set, we provide an estimate of *typical* performance—the performance estimate of a hypothetical next demonstration. We emphasize that the typical performance is different from the average performance.

To further support this point, we have included average fidelities (with bootstrapped 95% confidence intervals) superimposed on the box plots in Fig. 18. For all data sets, the mean and median do not agree. For the polar states ($|0\rangle$ and $|1\rangle$), the difference is slight, and their respective median performance is included in the error bar of the mean. We remark that averages and medians generally agree, as they do here for the best-performing DD sequences (see Fig. 3). The mean is heavily biased downward for the remaining equatorial states, and the confidence intervals do not include the mean. Once averaging over the entire data set, the discrepancy is even worse. Here, the mean falls on top of the 25% quantile, and the top of the confidence interval differs from the median by a significant amount. Hence, an average method is inappropriate for DD sequences whose performance is highly sensitive to drift effects, and a median

method is more appropriate. Again, we remark that this should not affect the top-performing sequences shown in Fig. 3, but it gives us a consistent way to sift through all the sequences at once to even get to a top ranking in the first place.

To summarize, we have shown that averaging fidelities over states is a delicate issue that depends on how the data are averaged. When averaging within a fixed job, the results appear reasonable and simple averaging of the fidelities over states gives an average state fidelity that behaves as expected. When comparing state data across different jobs, a straightforward average can be biased due to a particularly bad calibration, resulting in an average sensitive to drift. A nonparametric comparison in which fidelities are assessed via a box plot, on the other hand, does not suffer from this issue. When confined to a fixed job, the median over states gives a measure of typical performance over the set of states. When fidelities are collected over both states and calibrations, the median is still a measure of typical performance, but now over both states and calibrations. Hence, a measure of typical performance is more robust than an average performance metric when considering drift. In practice, typical performance is more relevant for any given fixed demonstration. These facts, alongside the practical difficulties of averaging and fitting, are why we opted for the FTA approach.

8. Summary of fitting fidelity

The standard approach to comparing DD sequences is to generate average fidelity decay curves, fit them, and report a summary statistic like a decay constant. For this reason, we call this an "average-then-fit" approach. In Ref. [41], for example, the authors generated fidelity decay curves for roughly 40 equally spaced times and 36 states (30 Haar-random and six Pauli eigenstates). For each time, they computed a state-average fidelity constituting a single average fidelity decay curve. They then fit this curve to a modified exponential decay [Eq. (C2a)] and reported the decay constant for the XY4 sequence compared to free evolution. This was sufficient to show that, on average, applying the XY4 sequence was better than free evolution.

In this work, we aim to go beyond this approach and identify any pitfalls. To do so, we set out to first compare a large collection of DD sequences and not just the XY4 sequence and Free. Second, we are interested in whether the XY4 sequence retains universality for superconducting devices (see Sec. II D). This question requires us to know precisely whether, e.g., $|+\rangle$ or $|1\rangle$ is better protected under the XY4 sequence. This led us to two methodological observations. (i) Using a standard fit such as Eq. (C2a) to individual state decay curves results in several problems: it does not always work in practice, leading to an ambiguous interpretation of the decay constant, and it requires a complicated fitting procedure. (ii) The fidelity of a given state protected with DD for a fixed time is unstable from job to job due to device drift. This means that one must be careful when comparing fidelities not collected within the same job.

We resolved both issues by (i) focusing on typical performance instead of average and (ii) using an FTA approach where we interpolate with time averaging. Along the way, the insistence on not first averaging results in the identification of significant and unexpected asymmetries in performance between states (see Appendix B and especially Fig. 6), which led to better DD sequence design.

9. Details of fitting using *Mathematica*

a. Standard fit to exponential details

At a high level, we fit the fidelity decay curves (see Fig. 7 for example curves and Fig. 8 for example curves with fits) to Eq. (C2a) using *Mathematica*'s `NonlinearModelFit` (NLM) [112,113] that performs weighted least-squares regression. In more detail, NLM finds the parameters $\{\hat{\lambda}, \hat{\gamma}, \hat{\alpha}\}$ that minimize the weighted sum of squares

$$S = \sum_{t=1}^{N} \frac{1}{\sigma_{T_i}^2} (\langle f_e \rangle_{T_i} - f_P(T_i; \lambda, \gamma, \alpha)) \tag{C6}$$

with the assumption that

$$\langle f_e \rangle_{T_i} = f_P(T_i; \lambda^*, \gamma^*, \alpha^*) + \epsilon_{T_i}, \qquad \epsilon_{T_i} \sim \mathcal{N}(0, \sigma_{T_i}), \tag{C7}$$

for some unknown parameters λ^*, γ^*, and α^*. Given a good model and fitting procedure, the NLM procedure yields $\hat{\lambda} \approx \lambda^*, \hat{\gamma} \approx \gamma^*$, and $\hat{\alpha} \approx \alpha^*$ [114]. Furthermore, NLM computes the covariance matrix C between the estimated parameters,

$$C = X^T W X, \tag{C8a}$$

$$X = \begin{bmatrix} \partial_\lambda F(T_1) & \partial_\gamma F(T_1) & \partial_\alpha F(T_1) \\ \vdots & \vdots & \vdots \\ \partial_\lambda F(T_N) & \partial_\gamma F(T_N) & \partial_\alpha F(T_N) \end{bmatrix}, \tag{C8b}$$

$$W = \mathrm{diag}(1/\sigma_{T_1}^2, \ldots, 1/\sigma_{T_N}^2), \tag{C8c}$$

where $F(T)$ is the integrated fidelity [Eq. (C1)] and this particular form of C and W results from setting the `Weights` and `VarianceEstimatorFunction` NLM settings appropriately for experimental data [115]. As usual, we compute the parameter standard errors

(aka the standard deviations of the statistics) from the covariance matrix as

$$\mathrm{SE}_\lambda = \sqrt{C_{11}}, \qquad \mathrm{SE}_\gamma = \sqrt{C_{22}}, \qquad \mathrm{SE}_\alpha = \sqrt{C_{33}}. \tag{C9}$$

From this, we can estimate the parameter 95% confidence intervals by calculating a t score. To do this, we first find the number of degrees of freedom we have, $\nu = N - \#$ parameters. The t score is then the $t^*_{0.95}$ such that

$$P(-t^*_{0.95} \le T \le t^*_{0.95}) = 0.95, \qquad T \sim \mathrm{StudentTDist}(\nu). \tag{C10}$$

In the end, we report our best-fit parameters with 95% confidence using

$$\lambda = \hat{\lambda} \pm \mathrm{SE}_\lambda \times t^*_{0.95} \equiv \hat{\lambda} \pm \delta_\lambda; \tag{C11}$$

with an exactly analogous formula for γ and α (i.e., same $t^*_{0.95}$ throughout) [116].

b. Consistency checks and selecting appropriate fitting options

The previous section outlined the general methodology and output of *Mathematica*'s `NonlinearModelFit` function. One detail we glossed over is the method by which `NonlinearModelFit` minimizes the weighted sum of squares, S. By default, S is minimized using the `LevenbergMarquardt` method—a particular implementation of the Gauss-Newton algorithm. Many of the details and optimization settings are outlined in *Mathematica*'s "Unconstrained Optimization: Methods for Local Minimization" [117] documentation. An important feature of this method is that it is a local search method, and hence the quality of the fit depends on the particular input settings and randomness in each run. Coupled with the fact that λ, γ, and α of Eq. (C2a) are not independent parameters, it is important in practice to try many different settings to obtain a good fit. A full enumeration of all settings we attempted can be found in the *fittingFreeData.nb Mathematica* notebook contained in our code repository [100]. For example, we consider fits for which we reduce the number of parameters (e.g., setting $e^{-t/\alpha} = 1$) or where $\gamma = 0$ is either seeded or unseeded.

Upon trying many possible fits, we then need an objective way to compare their relative quality. To do so from a fit obtained using `NonlinearModelFit`, one can query its associated (corrected) Akaike information criterion (AIC) [118] via *Mathematica*'s `Fit["AICc"]`. The AIC estimates the relative quality of each model in a collection of models for a given data set. In our case, the sample size is relatively small: 12 points for each fit. To remedy this, we employ the corrected AIC, which corrects the tendency of AIC to favor overfitting for small sample sizes. See Ref. [118] for a summary of the formulas for AIC and corrected AIC. A more detailed introduction and derivations of the formulas can be found in Ref. [119, pp. 51–74].

The lower the corrected AIC, the better the model is relative to others. However, we must also ensure that the final fit satisfies consistency conditions. Namely, a fit is considered unreasonable if it (a) predicts a fidelity less than zero or greater than one, (b) predicts a parameter whose error is larger than the value itself, or (c) violates the Nyquist-Shannon sampling theorem [120], i.e., has frequency larger than the sample rate of the data itself. That is, γ in Eq. (C2a) must satisfy $0 \le \gamma \le 2\pi/2\Delta t \equiv B$, where $\Delta t = (150/12)$ μs is the spacing between data points in our fidelity decay curves. After fitting, we reject fits that violate properties (a) and (b) and correct those that violate (c). We next illustrate what this postselection or correction looks like for our data.

First, we perform many fits without regard to whether (a)–(c) are satisfied since enforcing constraints makes the nonlinear fit results unstable, and moreover, the constraints are not always enforced even when specified inside `NonlinearModelFit`. We then postselect the fits that respect (a) and (b), and among these, we select the fit with the lowest corrected AIC. If no fit respecting (a) and (b) is found, we drop that data set. Finally, we rescale the frequency via $\gamma \to \mathrm{mod}(\gamma, B)$. This procedure—along with the fits along the way—is summarized in Fig. 19. Note that we are forced to postselect fits that are well modeled by Eq. (C2a) in order to report a reasonable value of λ for the given data. For this reason, the fitting procedure thus described is biased toward data that are appropriately modeled by Eq. (C2a).

c. Time-averaged fidelity approach

We construct a polynomial interpolation of the fidelity decay data using *Mathematica*'s `Interpolation` function [112,121]. Then we integrate over the interpolation to compute a time-averaged fidelity according to Eq. (C1). By default, the interpolation uses a third-order Hermite method [122] that we found to be sufficient to adequately model our data. For example, the curves in Fig. 7 consist of Hermite interpolations of the triangular raw data, which appear reasonable. To compute Eq. (C1) from the interpolation, we used *Mathematica*'s `NIntegrate`.

For a sense of stability, we also tried the popular cubic spline interpolation method [123], but this does not affect our results outside of the interpolation error bound itself. For example, consider the complicated interpolation necessary for KDD using the Hermite or cubic spline methods in Fig. 20. Hence, either interpolation is equally valid when using the time-averaged fidelity metric.

We remark that polynomial interpolation is much easier to do than fitting to a model such as Eq. (C2a) since we

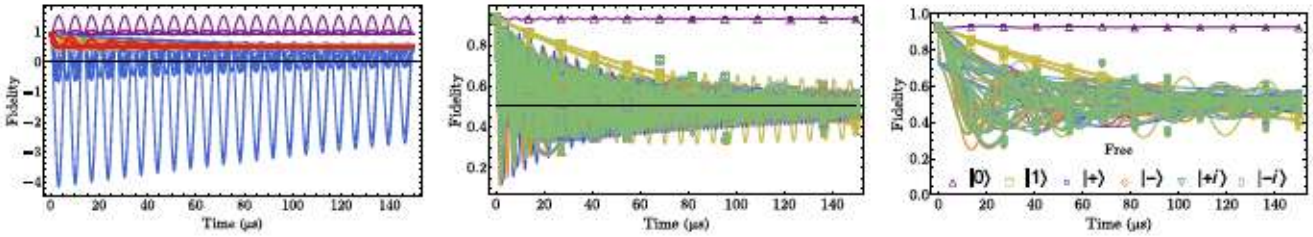


FIG. 19. A summary of the postselection process necessary when fitting to Eq. (C2a) with *Mathematica*'s NonlinearModelFit. The data are the same free evolution data displayed in Fig. 8. Left: for each curve, we select the fit with the lowest corrected AIC without regard to whether the fit meets consistency conditions such as having positive fidelity. Middle: the result of first postselecting those fits that satisfy consistency conditions and then choosing that fit with the lowest corrected AIC. Some data sets (especially the flat $|0\rangle$ state decay) have no reasonable fits, so they are dropped. Right: finally, we rescale the frequency in accordance with the Nyquist-Shannon sampling theorem (this does not affect the predicted decay constant λ). This final plot is displayed as the result in Fig. 8 and the curves for which the λ's are reported in Fig. 9. Namely, data that do not have a reasonable fit are not included in the final summary, which biases towards data that do fit Eq. (C2a).

need not (i) select reasonable fitting equations, (ii) fit them using nonlinear least-squares regression, (iii) postselect fits that satisfy consistency conditions and the low corrected AIC. Additionally, the predictions are less susceptible to ambiguities, as described in Appendix C 5. Finally, we are not forced to discard any data sets that fail the postselection criteria, so we do not bias the final summary results in any particular way.

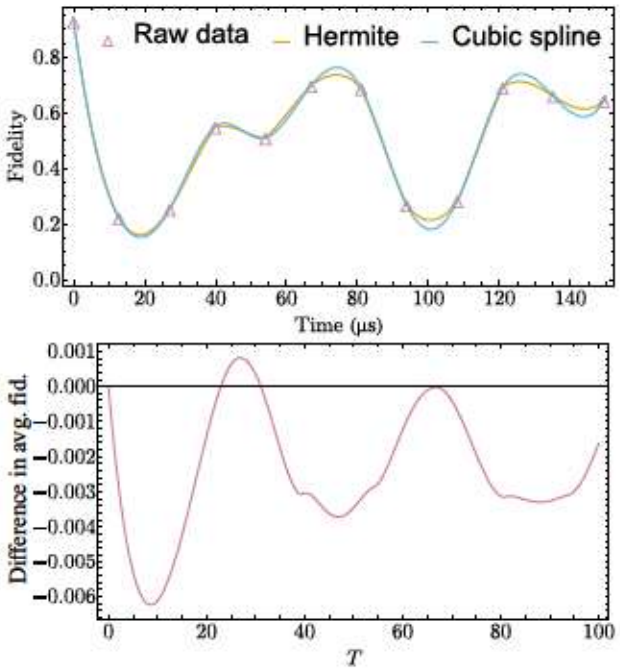


FIG. 20. We compare third-order Hermite and cubic spline interpolations on the fidelity decay curve generated by KDD with the initial state $|-\rangle$. The resulting interpolations are qualitatively similar, and the resulting average fidelities are within 6×10^{-3} of each other for all $0 \leq T \leq 100$ μs. Hence, they are equally valid interpolations when using time-averaged fidelity as a performance metric.

APPENDIX D: TOGGLING FRAME

We assume that without any external control, the system and bath evolve under the time-independent noise Hamiltonian H. A DD pulse sequence is realized via a time-dependent control Hamiltonian $H_c(t)$ acting only on the system, so that the system and bath evolve according to $H + H_c(t)$ (we refer the reader to Sec. II A for definitions of the various Hamiltonian terms).

For understanding the effects of the control Hamiltonian, it is convenient to use the interaction picture defined by $H_c(t)$, also known as the *toggling frame* [21,31,34,124,125]. The toggling-frame density operator $\tilde{\rho}_{SB}(t)$ is related to the Schrödinger-picture density operator $\rho_{SB}(t)$ by

$$\begin{aligned}\rho_{SB}(t) &= U(t,0)\rho_{SB}(0)U^{\dagger}(t,0)\\ &\equiv U_c(t)\tilde{\rho}_{SB}(t)U_c^{\dagger}(t),\end{aligned} \tag{D1}$$

where $U(t,0)$ is the evolution operator generated by the full Hamiltonian $H + H_c(t)$. Therefore, the toggling-frame state evolves according to

$$\tilde{\rho}_{SB}(t) = \tilde{U}(t,0)\tilde{\rho}_{SB}(0)\tilde{U}^{\dagger}(t,0), \tag{D2}$$

where the toggling-frame time evolution operator

$$\tilde{U}(t,0) \equiv U_c^{\dagger}(t)U(t,0) \tag{D3}$$

is generated by the *toggling-frame Hamiltonian*

$$\tilde{H}(t) \equiv U_c^{\dagger}(t)HU_c(t). \tag{D4}$$

Since $U_c(t)$ acts nontrivially only on the system, $\tilde{H}(t)$ can be written as

$$\tilde{H}(t) = H_B + \tilde{H}_{\rm err}(t), \tag{D5}$$

where $\tilde{H}_{\rm err}(t) \equiv U_c^{\dagger}(t)H_{\rm err}U_c(t)$ is the toggling-frame version of $H_{\rm err}$. Because the operator norm is unitarily

invariant, we have $\|\tilde{H}(t)\| = \|H\| \leq \epsilon$ and $\|\tilde{H}_{\rm err}(t)\| = \|H_{\rm err}\| \leq J$.

Throughout, we consider *cyclic DD*, where $U_c(t)$ returns to the identity (up to a possible irrelevant overall phase) at the end of a cycle taking time $t_{\rm DD}$:

$$U_c(t_{\rm DD}) = U_c(0) = I. \tag{D6}$$

Therefore, at the end of the cycle, the toggling-frame and Schrödinger-picture states coincide.

APPENDIX E: RESULTS OF ALL PAULI DEMONSTRATIONS

In Fig. 3, we summarized the results of the Pauli demonstration for the top-ten sequences on each device and the baseline of CPMG, XY4, and free evolution. This appendix presents the data for all 60 tested sequences. In particular, we split the data for each device into a 3×3 grid of plots shown in Figs. 21–23, separating by family as in Table I when possible. For convenience, CPMG, XY4, and Free are still included in all plots along with colored reference lines matching the convention in Fig. 3. The purple reference line denotes the best sequence in the given family plot excluding the baseline sequences placed at the end. For example, in Fig. 21(b) RGA_{64a} is the best RGA sequence even though it does not outperform XY4; it is marked with a purple reference line.

We plot the same families for each device in a 3×3 grid, labeled (a)–(i). In each caption, we make a few general comments on DD performance overall and then comment on observations specific to each sequence family. Recall that a specific definition of each sequence is given in Appendix A, and a summary of their properties along with references in Table I. To avoid excessive repetition in each caption, we first provide a brief description of each of the cases (a)–(i) shown in Fig. 21–23.

(a) This "family" serves as a catch-all for the basic sequences Hahn [51], CPMG [52,53], and XY4 [54], along with sequences born from their modifications. Namely, we also plot the Eulerian supercycle versions [56,83] (denoted by S), KDD [58,62], which is a composite pulse XY4, and CDD_n [55,95], which is a recursive embedding of XY4.

(b) The RGA family [57].

(c) The UR family [59].

(d) The UDD family using X pulses, UDDx [60].

(e)–(i) The QDD family with the same inner and outer orders, $\mathrm{QDD}_{n,n}$; with fixed outer order 1, $\mathrm{QDD}_{1,m}$; with fixed outer order 2, $\mathrm{QDD}_{2,m}$; with fixed outer order 3, $\mathrm{QDD}_{3,m}$; and with fixed outer order 4, $\mathrm{QDD}_{4,m}$, respectively [61].

For each family, we expect the empirical hierarchy of sequence performance to be a complicated function of device-specific properties. Specifically, actual performance is a competition between (i) error cancelation order, (ii) number of free evolution periods, and (iii) systematic pulse errors due to finite width and miscalibrations, among other factors discussed in Sec. II. For each family, we summarize our expectations regarding these factors.

(a) For ideal pulses, we expect that $\mathrm{CDD}_{n+1} > \mathrm{CDD}_n \geq \mathrm{KDD} = \text{S-XY4} = \mathrm{XY4} > \text{S-CPMG} = \mathrm{CPMG} = \text{S-Hahn} > \mathrm{Hahn}$. With a finite bandwidth constraint, we expect $\mathrm{CDD}_{n+1} > \mathrm{CDD}_n$ to only hold up until some optimal concatenation level $n_{\rm opt}$ after which performance saturates. Using finite width pulses with systematic errors, we expect that S-Hahn > Hahn (and similarly for other S sequences) and KDD > XY4, provided the additional robustness is helpful. That is, if the pulses are extremely well calibrated and errors are dominated by latent bath-induced errors then we should instead see that Hahn > S-Hahn.

(b) The expected performance hierarchy for RGA is rather complicated, as indicated by the labeling, and is best summarized in depth using Table II of Ref. [57]. A quick summary is that if we have strong pulses dominated by miscalibration errors ($\epsilon\Delta \ll \epsilon_r$) then we expect RGA_{8a} and RGA_{64a} to do well. In the opposite limit, we expect $\mathrm{RGA}_4, \mathrm{RGA}_{8c}, \mathrm{RGA}_{16b}, \mathrm{RGA}_{64c}$ to do well. The increasing number indicates the number of pulses, and as this increases, the decoupling order $\mathcal{O}(\tau^n)$ increases, and the same competition between order-cancelation and free evolution periods as in CDD_n also applies.

(c) The UR_n sequence provides $\mathcal{O}(\epsilon_r^{n/2})$ suppression of flip-angle errors at the expense of using n free evolution periods. The relationship of n to $\mathcal{O}(\tau)$ decoupling is not well established in Ref. [59], but, by construction, seems to be $\mathcal{O}(\tau^2)$ for all n. Thus, our expectation is that UR_n improves with increasing n until performance saturates and the $\mathcal{O}(\tau^2)$ contribution dominates the $\mathcal{O}(\epsilon_r^{n/2})$ contribution. To see this, note that, for a fixed time, the number of free evolution periods will be roughly the same regardless of n.

(d) Ideally, for a fixed demonstration duration T, the performance of UDDx_n should scale as $\mathcal{O}(\tau^n)$, and hence improves monotonically with increasing n. In practice, this performance should saturate once the finite pulse-width error $\mathcal{O}(\Delta)$ is the dominant noise contribution.

(e)–(i) An extensive numerical study of $\mathrm{QDD}_{n,m}$ performance is discussed in Ref. [75] with corresponding rigorous proofs in Ref. [77]. For ideal pules, the decoupling order is expected to be at least $\mathcal{O}(t_s^{\min\{m,n\}})$, where t_s is the total evolution time of implementing a single repetition. Since we are instead interested in a fixed total time T consisting of multiple sequence repetitions with a minimal pulse interval, the interplay of competing factors is quite complicated. Furthermore, we are forced to apply rotations

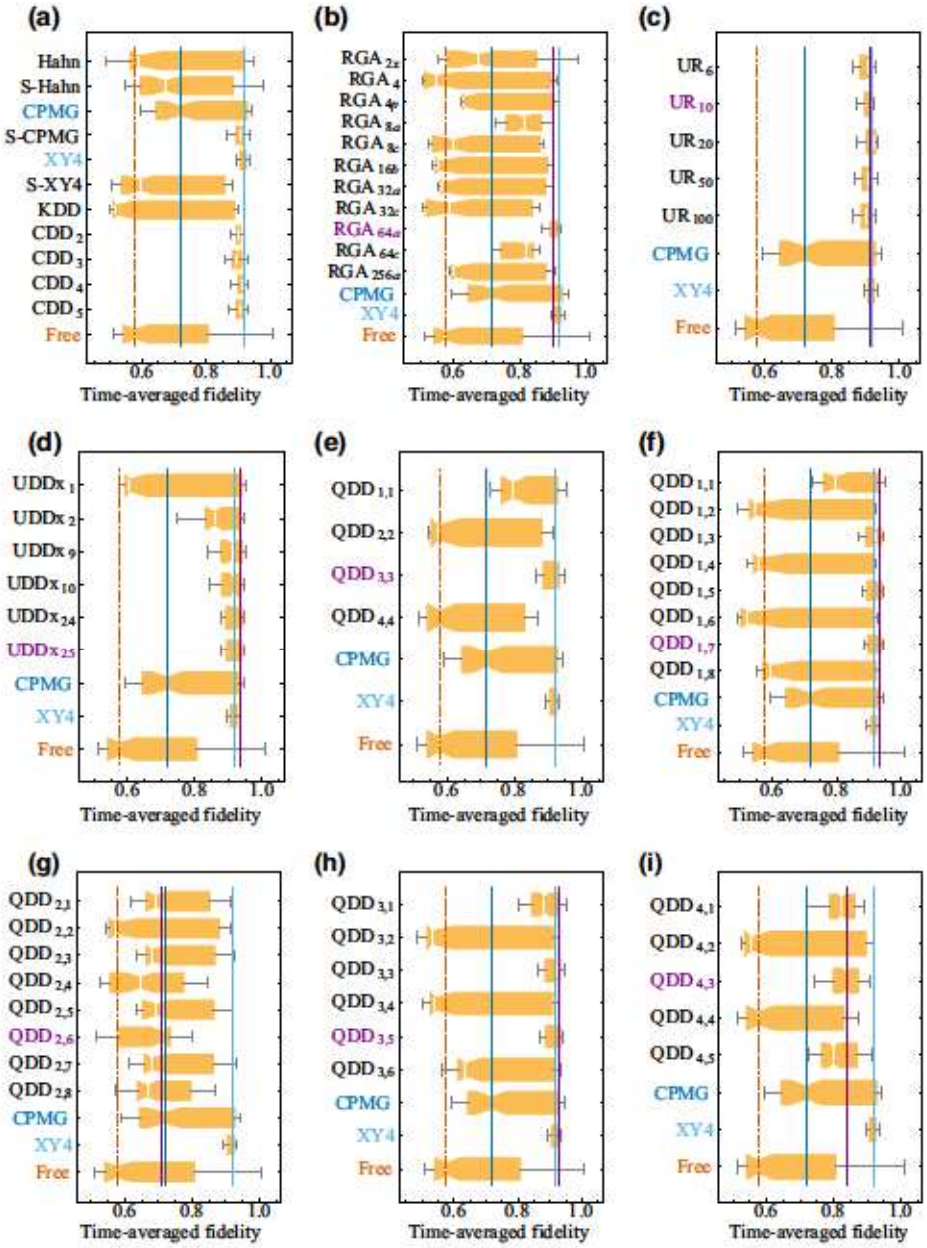


FIG. 21. Collection of all Pauli-demonstration results for ibmq_armonk. The best-performing sequence for each family (solid purple line) substantially outperforms Free (orange) and CPMG (blue), but only marginally differs from the performance of XY4 (cyan). (a) Results are not consistent with ideal-pulse theory, but are sensible when considering realistic competing factors. First, S-Hahn > Hahn and S-CPMG > CPMG are consistent with the large pulse widths on ibmq_armonk, for which $\Delta \approx 142$ ns. That XY4 works very well is also consistent with its expected approximate universality given that $T_1 \approx \frac{1}{2}T_2$ on ibmq_armonk. Given that XY4 already performs well, it is not surprising that its robust versions, S-XY4 and KDD, do worse. In particular, they have little room for improvement, but also add extra free evolution periods that accumulate additional error. Finally, CDD_n is roughly flat for all n tested, which is consistent with an expected saturation that happens to occur at $n_{\mathrm{opt}} = 1$. (b) The performance of RGA does not match theoretical expectations. To see this, note that RGA_4 is itself a four-pulse sequence with error scaling $\mathcal{O}(\tau^2)$ that is the same as XY4. Hence, it should perform comparably to XY4, but it does significantly worse. Furthermore, it is also unexpected that the best RGA sequences are 8*a*, 64*a*, and 64*c*. Indeed, 8*a* and 64*a* are expected to work best in a flip-angle error-dominated regime, but in this regime, 64*c* has a scaling of $\mathcal{O}(\epsilon_r^2)$, so it is not expected to do well. (c) The UR_n sequence performance is consistent with theory. First note that $\mathrm{UR}_4 = \mathrm{XY4}$. Thus, we expect UR_n for $n > 4$ to improve upon or saturate at the performance of XY4, and the latter is what happens. (d) The UDD_n sequence performance is consistent with theory. In particular, we expect (and observe) a consistent increase in performance with increasing n until performance saturates. (e)–(i) The $\mathrm{QDD}_{n,m}$ results in part match theoretical expectations, since they exhibit a strong even-odd effect, as predicted in Refs. [75,77,80]. Nevertheless, the optimal choice of n and m has to be fine-tuned for ibmq_armonk. We note that five out of ten of the top-ten sequences on ibmq_armonk are from the QDD family.

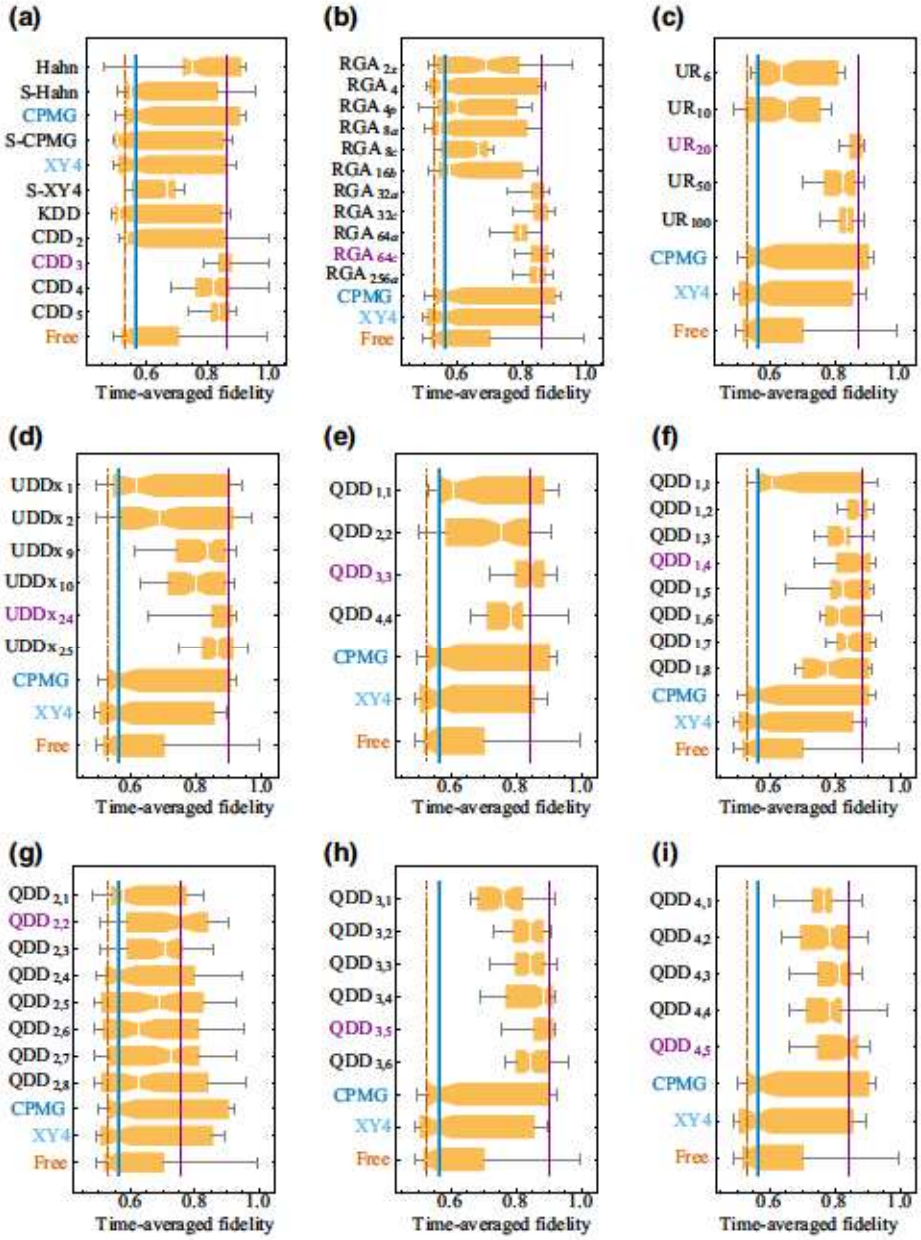


FIG. 22. Collection of all Pauli-demonstration results for ibmq_bogota. The best-performing sequence for each family (solid purple line) substantially outperforms Free (orange), CPMG (blue), and XY4 (cyan). (a) Results are not fully consistent with ideal-pulse theory. For example, it is unexpected that Hahn > CPMG and that Hahn < S-Hahn, while at the same time S-XY4 > XY4. Nevertheless, some trends are expected such as $\mathrm{CDD}_1 < \mathrm{CDD}_2 < \mathrm{CDD}_3$ and then saturating. (b) Results are somewhat consistent with ideal-pulse theory. First of all, $\mathrm{RGA}_4 \approx \mathrm{XY4}$ in performance, which is sensible. The first large improvement comes from RGA_{8c} and then from numbers 32 and greater. This trend is similar to CDD_n increasing, which is expected since larger RGA sequences are also recursively defined (e.g., RGA_{64c} is a recursive embedding of RGA_{8c} into itself). However, it is again unexpected that both "a" and "c" sequences should work well at the same time. For example, $\mathrm{RGA}_{8c} > \mathrm{RGA}_{8a}$ theoretically means that ibmq_bogota has negligible flip-angle error. In such a regime, the decoupling order of RGA_{8a} is the same as RGA_{64a} since they are designed to cancel flip-angle errors. But, we find that $\mathrm{RGA}_{64a} > \mathrm{RGA}_{8a}$ in practice. (c) The UR_n results are mostly consistent with theory. First, UR_6 is an improvement over XY4, and though UR_n does increase with larger n, it is not simply monotonic as one would expect in theory. Instead, we find a more general trend with an empirical optimum at $n = 20$. (d) The UDDx_n results are mostly consistent with expectations. Again, performance mostly increases with increasing n, but the increase is not fully monotonic. (e)–(i) The $\mathrm{QDD}_{n,m}$ results are fairly consistent with theory. In (e), performance of $\mathrm{QDD}_{n,n}$ increases with n until $n = 3$. The degradation for $n = 4$ is consistent with expectations in the bandwidth-limited setting [78]. In (f)–(i) the results are again fairly expected: aside from parity effects (odd or even m), for $\mathrm{QDD}_{n,m}$, we expect monotonic improvement with increasing m until $n = m$, after which performance should saturate or even slightly improve; this is the general empirical trend.

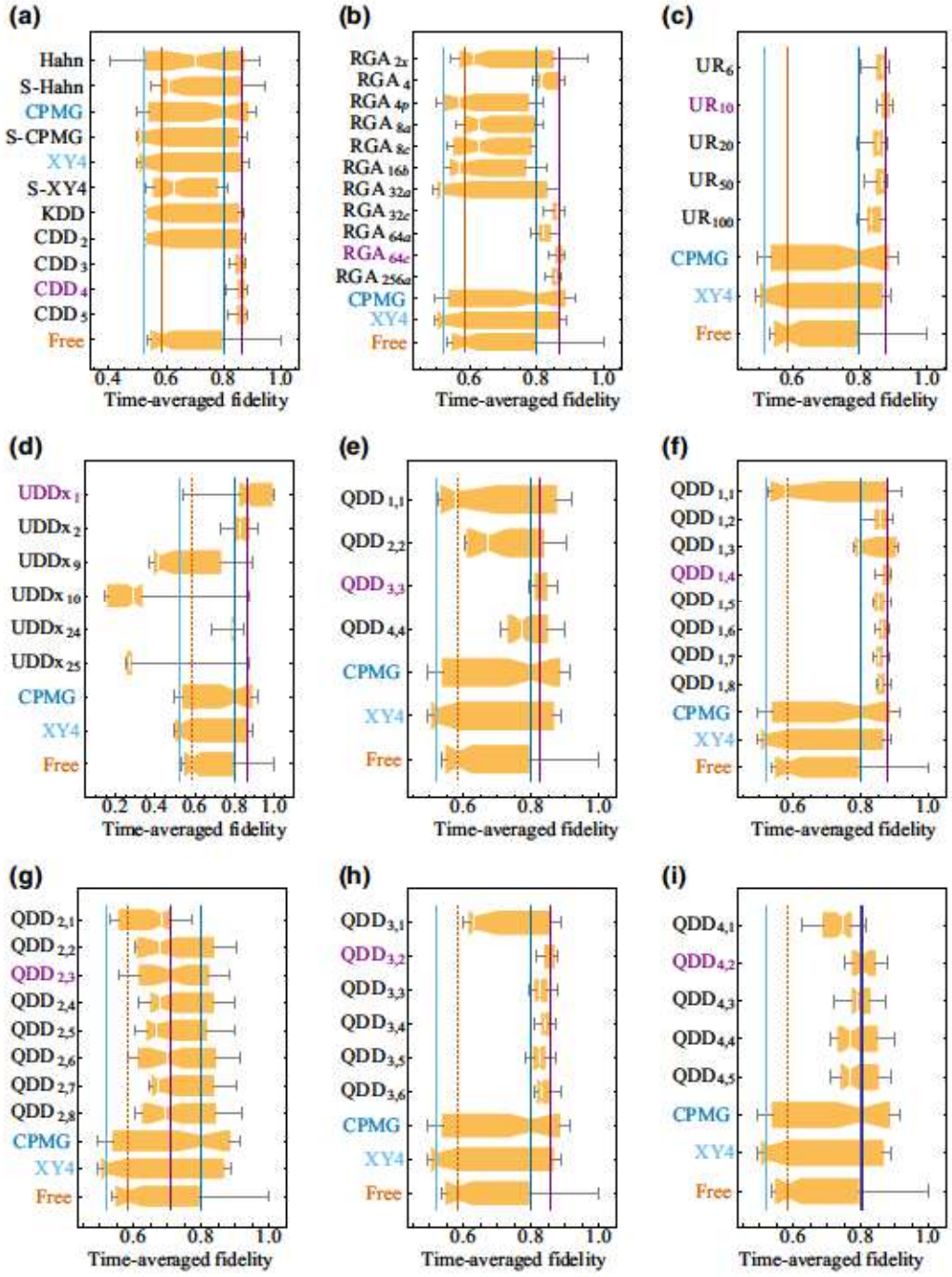


FIG. 23. Collection of all Pauli-demonstration results for ibmq_jakarta. The best-performing sequence for each family (solid purple line) substantially outperforms Free (orange) and CPMG (blue) but only marginally differs from the performance of XY4 (cyan). (a) Results are not fully consistent with ideal-pulse theory. For example, it is unexpected that Hahn > CPMG and that Hahn > S-Hahn, while at the same time S-XY4 > XY4. Nevertheless, some trends are expected, such as $CDD_1 < CDD_2 < CDD_3 > CDD_4$ and then saturating. (b) Results are somewhat consistent with the theory. First of all, $RGA_4 > XY4$ is sensible and implies that we are in a flip-angle error-dominated regime. However, we would then expect $RGA_4 > RGA_{8a}$, which does not occur. Nevertheless, the recursively defined sequences (number 32 and above) generally outperform their shorter counterparts, which is similar to CDD_n, as expected. (c) The UR_n results are consistent with theory. First, UR_6 is a large improvement over XY4, and from there $UR_{10} > UR_6$. After this, the improvement plateaus. (d) The performance of $UDDx_n$ greatly differs from the theoretical expectation of monotonic improvement with n. In fact, the behavior is so erratic that we suspect device calibration errors dominated the demonstration. Nevertheless, the performance of $UDDx_1$ was excellent in this case. (e)–(i) The $QDD_{n,m}$ results are fairly consistent with theory, and quite similar to Fig. 22. The same comments as made there apply here.

about X, Y, and Z to implement $\mathrm{QDD}_{n,m}$ when m is odd, but as noted in Appendix B, Z is virtual without OpenPulse. In summary, the naive theoretical expectation is that $\mathrm{QDD}_{n,m}$ should improve with increasing $\min\{n,m\}$, eventually saturating for the same reasons as UDDx_n. However, we expect the fixed T and virtual-Z set-up to complicate the actual results.

[1] A. Montanaro, Quantum algorithms: An overview, npj Quantum Inf. **2**, 15023 (2016).

[2] J. Preskill, Quantum Computing in the NISQ era and beyond, Quantum **2**, 79 (2018).

[3] A. J. Daley, I. Bloch, C. Kokail, S. Flannigan, N. Pearson, M. Troyer, and P. Zoller, Practical quantum advantage in quantum simulation, Nature **607**, 667 (2022).

[4] E. T. Campbell, B. M. Terhal, and C. Vuillot, Roads towards fault-tolerant universal quantum computation, Nature **549**, 172 EP (2017).

[5] D. Gottesman, Opportunities and challenges in fault-tolerant quantum computation, ArXiv:2210.15844 (2022).

[6] K. L. Pudenz, T. Albash, and D. A. Lidar, Error-corrected quantum annealing with hundreds of qubits, Nat. Commun. **5**, 3243 (2014).

[7] S. J. Devitt, Performing quantum computing experiments in the cloud, Phys. Rev. A **94**, 032329 (2016).

[8] W. Vinci, T. Albash, and D. A. Lidar, Nested quantum annealing correction, npj Quantum Inf. **2**, 16017 (2016).

[9] J. R. Wootton and D. Loss, Repetition code of 15 qubits, Phys. Rev. A **97**, 052313 (2018).

[10] C. Vuillot, Is error detection helpful on IBM 5Q chips?, Quantum Inf. Comput. **18**, 0949 (2018).

[11] J. Roffe, D. Headley, N. Chancellor, D. Horsman, and V. Kendon, Protecting quantum memories using coherent parity check codes, Quantum Sci. Technol. **3**, 035010 (2018).

[12] D. Willsch, M. Willsch, F. Jin, H. De Raedt, and K. Michielsen, Testing quantum fault tolerance on small systems, Phys. Rev. A **98**, 052348 (2018).

[13] A. Pearson, A. Mishra, I. Hen, and D. A. Lidar, Analog errors in quantum annealing: Doom and hope, npj Quantum Inf. **5**, 107 (2019).

[14] R. Harper and S. T. Flammia, Fault-tolerant logical gates in the IBM quantum experience, Phys. Rev. Lett. **122**, 080504 (2019).

[15] C. Ryan-Anderson, J. Bohnet, K. Lee, D. Gresh, A. Hankin, J. Gaebler, D. Francois, A. Chernoguzov, D. Lucchetti, N. Brown, *et al.*, Realization of real-time fault-tolerant quantum error correction, Phys. Rev. X **11**, 041058 (2021).

[16] Z. Chen, *et al.*, Exponential suppression of bit or phase errors with cyclic error correction, Nature **595**, 383 (2021).

[17] L. Viola and S. Lloyd, Dynamical suppression of decoherence in two-state quantum systems, Phys. Rev. A **58**, 2733 (1998).

[18] P. Zanardi, Symmetrizing evolutions, Phys. Lett. A **258**, 77 (1999).

[19] D. Vitali and P. Tombesi, Using parity kicks for decoherence control, Phys. Rev. A **59**, 4178 (1999).

[20] L.-M. Duan and G.-C. Guo, Suppressing environmental noise in quantum computation through pulse control, Phys. Lett. A **261**, 139 (1999).

[21] L. Viola, E. Knill, and S. Lloyd, Dynamical decoupling of open quantum systems, Phys. Rev. Lett. **82**, 2417 (1999).

[22] M. J. Biercuk, H. Uys, A. P. VanDevender, N. Shiga, W. M. Itano, and J. J. Bollinger, Optimized dynamical decoupling in a model quantum memory, Nature **458**, 996 (2009).

[23] M. J. Biercuk, H. Uys, A. P. VanDevender, N. Shiga, W. M. Itano, and J. J. Bollinger, Experimental Uhrig dynamical decoupling using trapped ions, Phys. Rev. A **79**, 062324 (2009).

[24] S. Damodarakurup, M. Lucamarini, G. D. Giuseppe, D. Vitali, and P. Tombesi, Experimental inhibition of decoherence on flying qubits via "bang-bang" control, Phys. Rev. Lett. **103**, 040502 (2009).

[25] J. Du, X. Rong, N. Zhao, Y. Wang, J. Yang, and R. B. Liu, Preserving electron spin coherence in solids by optimal dynamical decoupling, Nature **461**, 1265 (2009).

[26] E. R. Jenista, A. M. Stokes, R. T. Branca, and W. S. Warren, Optimized, unequal pulse spacing in multiple echo sequences improves refocusing in magnetic resonance, J. Chem. Phys. **131**, 204510 (2009).

[27] G. A. Álvarez, A. Ajoy, X. Peng, and D. Suter, Performance comparison of dynamical decoupling sequences for a qubit in a rapidly fluctuating spin bath, Phys. Rev. A **82**, 042306 (2010).

[28] X. Peng, D. Suter, and D. A. Lidar, High fidelity quantum memory via dynamical decoupling: Theory and experiment, J. Phys. B: At., Mol. Opt. Phys. **44**, 154003 (2011).

[29] Y. Sagi, I. Almog, and N. Davidson, in *CLEO/QELS: 2010 laser science to photonic applications* (Optica Publishing Group, San Jose, California United States, 2010), p. 1.

[30] Z.-H. Wang, G. de Lange, D. Ristè, R. Hanson, and V. V. Dobrovitski, Comparison of dynamical decoupling protocols for a nitrogen-vacancy center in diamond, Phys. Rev. B **85**, 155204 (2012).

[31] L. Viola, S. Lloyd, and E. Knill, Universal control of decoupled quantum systems, Phys. Rev. Lett. **83**, 4888 (1999).

[32] L. Viola, Quantum control via encoded dynamical decoupling, Phys. Rev. A **66**, 012307 (2002).

[33] K. Khodjasteh and D. A. Lidar, Universal fault-tolerant quantum computation in the presence of spontaneous emission and collective dephasing, Phys. Rev. Lett. **89**, 197904 (2002).

[34] L. Viola and E. Knill, Random decoupling schemes for quantum dynamical control and error suppression, Phys. Rev. Lett. **94**, 060502 (2005).

[35] K. Khodjasteh and D. A. Lidar, Rigorous bounds on the performance of a hybrid dynamical-decoupling quantum-computing scheme, Phys. Rev. A **78**, 012355 (2008).

[36] D. A. Lidar, Towards fault tolerant adiabatic quantum computation, Phys. Rev. Lett. **100**, 160506 (2008).

[37] J. R. West, D. A. Lidar, B. H. Fong, and M. F. Gyure, High fidelity quantum gates via dynamical decoupling, Phys. Rev. Lett. **105**, 230503 (2010).

[38] T. van der Sar, Z. H. Wang, M. S. Blok, H. Bernien, T. H. Taminiau, D. M. Toyli, D. A. Lidar, D. D. Awschalom, R. Hanson, and V. V. Dobrovitski, Decoherence-protected quantum gates for a hybrid solid-state spin register, Nature **484**, 82 (2012).

[39] H. K. Ng, D. A. Lidar, and J. Preskill, Combining dynamical decoupling with fault-tolerant quantum computation, Phys. Rev. A **84**, 012305 (2011).

[40] G. A. Paz-Silva and D. A. Lidar, Optimally combining dynamical decoupling and quantum error correction, Sci. Rep. **3**, 1530 (2013).

[41] B. Pokharel, N. Anand, B. Fortman, and D. A. Lidar, Demonstration of fidelity improvement using dynamical decoupling with superconducting qubits, Phys. Rev. Lett. **121**, 220502 (2018).

[42] A. M. Souza, Process tomography of robust dynamical decoupling with superconducting qubits, Quantum Inf. Process. **20**, 237 (2021).

[43] P. Jurcevic, *et al.*, Demonstration of quantum volume 64 on a superconducting quantum computing system, Quantum Sci. Technol. **6**, 025020 (2021).

[44] V. Tripathi, H. Chen, M. Khezri, K.-W. Yip, E. M. Levenson-Falk, and D. A. Lidar, Suppression of crosstalk in superconducting qubits using dynamical decoupling, Phys. Rev. Appl. **18**, 024068 (2022).

[45] B. Pokharel and D. Lidar, Better-than-classical Grover search via quantum error detection and suppression, ArXiv:2211.04543 (2022).

[46] Z. Zhou, R. Sitler, Y. Oda, K. Schultz, and G. Quiroz, Quantum crosstalk robust quantum control, ArXiv:2208.05978 (2022).

[47] B. Pokharel and D. A. Lidar, Demonstration of algorithmic quantum speedup, ArXiv:2207.07647 (2022).

[48] T. Alexander, N. Kanazawa, D. J. Egger, L. Capelluto, C. J. Wood, A. Javadi-Abhari, and D. McKay, Qiskit pulse: Programming quantum computers through the cloud with pulses, ArXiv:2004.06755 (2020).

[49] H. Abraham, *et al.*, Qiskit: An open-source framework for quantum computing (2019).

[50] D. C. McKay, T. Alexander, L. Bello, M. J. Biercuk, L. Bishop, J. Chen, J. M. Chow, A. D. Córcoles, D. Egger, S. Filipp, J. Gomez, M. Hush, A. Javadi-Abhari, D. Moreda, P. Nation, B. Paulovicks, E. Winston, C. J. Wood, J. Wootton, and J. M. Gambetta, Qiskit backend specifications for OpenQASM and OpenPulse experiments, ArXiv:1809.03452 (2018).

[51] E. L. Hahn, Spin Echoes, Phys. Rev. **80**, 580 (1950).

[52] H. Carr and E. Purcell, Effects of diffusion on free precession in nuclear magnetic resonance experiments, Phys. Rev. **94**, 630 (1954).

[53] S. Meiboom and D. Gill, Modified spin-echo method for measuring nuclear relaxation times, Rev. Sci. Instrum. **29**, 688 (1958).

[54] A. A. Maudsley, Modified Carr-Purcell-Meiboom-Gill sequence for NMR Fourier imaging applications, J. Magn. Reson. **69**, 488 (1986).

[55] K. Khodjasteh and D. A. Lidar, Fault-tolerant quantum dynamical decoupling, Phys. Rev. Lett. **95**, 180501 (2005).

[56] L. Viola and E. Knill, Robust dynamical decoupling of quantum systems with bounded controls, Phys. Rev. Lett. **90**, 037901 (2003).

[57] G. Quiroz and D. A. Lidar, Optimized dynamical decoupling via genetic algorithms, Phys. Rev. A **88**, 052306 (2013).

[58] A. M. Souza, G. A. Álvarez, and D. Suter, Robust dynamical decoupling for quantum computing and quantum memory, Phys. Rev. Lett. **106**, 240501 (2011).

[59] G. T. Genov, D. Schraft, N. V. Vitanov, and T. Halfmann, Arbitrarily accurate pulse sequences for robust dynamical decoupling, Phys. Rev. Lett. **118**, 133202 (2017).

[60] G. S. Uhrig, Keeping a quantum bit alive by optimized π-pulse sequences, Phys. Rev. Lett. **98**, 100504 (2007).

[61] J. R. West, B. H. Fong, and D. A. Lidar, Near-optimal dynamical decoupling of a qubit, Phys. Rev. Lett. **104**, 130501 (2010).

[62] A. M. Souza, G. A. Álvarez, and D. Suter, Robust dynamical decoupling, Philos. Trans. R. Soc. A: Math. Phys. Eng. Sci. **370**, 4748 (2012).

[63] L. Capelluto and T. Alexander, OpenPulse: Software for experimental physicists in quantum computing (2021), https://www.research.ibm.com/publications/openpulse-software-for-experimental-physicists-in-quantum-computing.

[64] Z.-Y. Wang and R.-B. Liu, Protection of quantum systems by nested dynamical decoupling, Phys. Rev. A **83**, 022306 (2011).

[65] We use $X = \sigma^x$ interchangeably, and likewise for Y and Z, where σ^α denotes the αth Pauli matrix, with $\alpha \in \{x, y, z\}$. See Appendix A for a precise definition of all sequences.

[66] S. Meiboom and D. Gill, Modified spin-echo method for measuring nuclear relaxation times, Rev. Sci. Instrum. **29**, 688 (1958).

[67] M. Suzuki, On the convergence of exponential operators—the Zassenhaus formula, BCH formula and systematic approximants, Commun. Math. Phys. **57**, 193 (1977).

[68] P. Zanardi, Symmetrizing evolutions, Phys. Lett. A **258**, 77 (1999).

[69] M. S. Byrd and D. A. Lidar, Bang–bang operations from a geometric perspective, Quantum Inf. Process. **1**, 19 (2002).

[70] The construction works for any base sequence, but we specify the XY4 sequence here for ease of presentation since our results labeled with CDD_n always assume that the XY4 sequence is the base sequence.

[71] We use the sup operator norm (the largest singular value of A): $\|A\| \equiv \sup_{\{|v\rangle\}} \|A|v\rangle\|/\||v\rangle\| = \sup_{\{|v\rangle \text{ such that } \||v\rangle\|=1\}} \|A|v\rangle\|$.

[72] L. Cywiński, R. M. Lutchyn, C. P. Nave, and S. Das Sarma, How to enhance dephasing time in superconducting qubits, Phys. Rev. B **77**, 174509 (2008).

[73] G. A. Álvarez and D. Suter, Measuring the spectrum of colored noise by dynamical decoupling, Phys. Rev. Lett. **107**, 230501 (2011).

[74] H. Uys, M. J. Biercuk, and J. J. Bollinger, Optimized noise filtration through dynamical decoupling, Phys. Rev. Lett. **103**, 040501 (2009).

[75] G. Quiroz and D. A. Lidar, Quadratic dynamical decoupling with nonuniform error suppression, Phys. Rev. A **84**, 042328 (2011).

[76] G. S. Uhrig and D. A. Lidar, Rigorous bounds for optimal dynamical decoupling, Phys. Rev. A **82**, 012301 (2010).

[77] W.-J. Kuo and D. A. Lidar, Quadratic dynamical decoupling: Universality proof and error analysis, Phys. Rev. A **84**, 042329 (2011).

[78] Y. Xia, G. S. Uhrig, and D. A. Lidar, Rigorous performance bounds for quadratic and nested dynamical decoupling, Phys. Rev. A **84**, 062332 (2011).

[79] L. Jiang and A. Imambekov, Universal dynamical decoupling of multiqubit states from environment, Phys. Rev. A **84**, 060302 (2011).

[80] W.-J. Kuo, G. Quiroz, G. A. Paz-Silva, and D. A. Lidar, Universality proof and analysis of generalized nested Uhrig dynamical decoupling, J. Math. Phys. **53**, 122207 (2012).

[81] The terminology arises from Euler cycles traversed by the control unitary in the Cayley graph of the group generated by the DD pulses.

[82] L. Viola, Advances in decoherence control, J. Mod. Opt. **51**, 2357 (2004).

[83] S. T. Smith, M.Sc. Thesis (2007), https://cpb-us-e1.wpmucdn.com/sites.dartmouth.edu/dist/9/931/files/2016/10/TaylorSmithThesis-27vgr7u.pdf.

[84] To clarify their relationship, suppose that we take the ideal $\Delta \to 0$ limit. Here, the XY4 sequence is strictly better since EDD uses twice as many pulses (and therefore free periods) to accomplish the same $\mathcal{O}(\tau)$ decoupling.

[85] K. Khodjasteh, D. A. Lidar, and L. Viola, Arbitrarily accurate dynamical control in open quantum systems, Phys. Rev. Lett. **104**, 090501 (2010).

[86] J.-M. Cai, B. Naydenov, R. Pfeiffer, L. P. McGuinness, K. D. Jahnke, F. Jelezko, M. B. Plenio, and A. Retzker, Robust dynamical decoupling with concatenated continuous driving, New J. Phys. **14**, 113023 (2012).

[87] Caveats include the following. (1) The analytical CDCG constructions given in Ref. [85] do not accommodate for the setting where always-on terms in the system's Hamiltonian are needed for universal control (so that they cannot just be included in H_{eff}). (2) Control-induced (often multiplicative) noise can be simultaneously present along with bath-induced noise; multiplicative noise requires modifications to the formalism of Ref. [85].

[88] C. A. Ryan, J. S. Hodges, and D. G. Cory, Robust decoupling techniques to extend quantum coherence in diamond, Phys. Rev. Lett. **105**, 200402 (2010).

[89] R. Freeman, *Spin Choreography: Basic Steps in High Resolution NMR* (Oxford University Press, Oxford, 1998).

[90] K. R. Brown, A. W. Harrow, and I. L. Chuang, Arbitrarily accurate composite pulse sequences, Phys. Rev. A **70**, 052318 (2004).

[91] K. R. Brown, A. W. Harrow, and I. L. Chuang, Erratum: Arbitrarily accurate composite pulse sequences [Phys. Rev. A 70, 052318 (2004)], Phys. Rev. A **72**, 039905 (2005).

[92] This frame choice is motivated by the observation that in IBMQ devices, the drive frequency is calibrated to be resonant with a particular eigenfrequency of the devices' qubits, which depends on the state of the neighboring qubits. Transforming into a frame that rotates with one of these frequencies gets one as close as possible to describing the gates and pulses as static in the rotating frame.

[93] Intuitively, the Z_1Z_2 rotation at frequency ω_d is already self-averaging the effects of noise around the z axis, so in a sense, is acting like a DD sequence.

[94] IBM Quantum, https://quantum-computing.ibm.com/ (2021).

[95] K. Khodjasteh and D. A. Lidar, Performance of deterministic dynamical decoupling schemes: Concatenated and periodic pulse sequences, Phys. Rev. A **75**, 062310 (2007).

[96] K. Khodjasteh and D. A. Lidar, Quantum computing in the presence of spontaneous emission by a combined dynamical decoupling and quantum-error-correction strategy, Phys. Rev. A **68**, 022322 (2003), erratum: *ibid*, Phys. Rev. A **72**, 029905 (2005),

[97] G. S. Ravi, K. N. Smith, P. Gokhale, A. Mari, N. Earnest, A. Javadi-Abhari, and F. T. Chong, in *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)* (IEEE, Seoul, Korea Republic of, 2022), p. 288.

[98] M. Amico, H. Zhang, P. Jurcevic, L. S. Bishop, P. Nation, A. Wack, and D. C. McKay, Defining standard strategies for quantum benchmarks, ArXiv:2303.02108 (2023).

[99] A. Zlokapa and A. Gheorghiu, A deep learning model for noise prediction on near-term quantum devices, ArXiv:2005.10811 (2020).

[100] N. Ezzell, Naezzell/edd: edd arxiv v2 release (2023).

[101] D. C. McKay, C. J. Wood, S. Sheldon, J. M. Chow, and J. M. Gambetta, Efficient Z-gates for quantum computing, Phys. Rev. A **96**, 022330 (2017).

[102] This observation and explanation is due to Vinay Tripathi.

[103] We have added a factor of $1/2$ to $\Gamma(t)$ that was unfortunately omitted in Ref. [41].

[104] B. Efron, in *Breakthroughs in statistics: methodology and distribution*, edited by S. Kotz and N. L. Johnson (Springer New York, New York, NY, 1992), p. 569.

[105] We caution that we use the term "averaging" to indicate both *time* averaging [as in Eq. (C1)] and *data* averaging, i.e., averaging fidelity data from different calibration cycles and states.

[106] We actually sent 100 jobs, but after 67 successful completions, our program crashed due to an internet connection issue.

[107] We claim that two jobs are contiguous if they began running on the device within 1 min of each other. This is because the full circuit takes approximately 80 μs to execute, so 8192 shots takes around 0.657 s to execute. Hence, a job with 10 to 75 demonstrations takes between 6.57 and 49.3 s to execute, excluding any latency time to communicate between our laptop and the ibmq_armonk server.

[108] Wikipedia, Jarque–Bera test, https://en.wikipedia.org/wiki/Jarque-Bera_test (2023).

[109] Wikipedia, Shaprio-Wilk test, https://en.wikipedia.org/wiki/Shapiro-Wilk_test (2023).

[110] By taking a different number of samples, it is as if we roll a die but write down the answer a random number

of times. This sampling strategy takes longer to converge to the expected result.

[111] A. M. Souza, Process tomography of robust dynamical decoupling in superconducting qubits, ArXiv:2006.10585 (2020).

[112] S. Wolfram, Mathematica: A System for Doing Mathematics by Computer, version 3.0 for SGI.

[113] Wolfram Research, NonlinearModelFit, https://reference.wolfram.com/language/ref/NonlinearModelFit.html (2008).

[114] J. H. Williams, *Quantifying Measurement* (Morgan & Claypool Publishers, San Rafael, CA, 2016), p. 2053.

[115] The choice of weights is exactly as stated in Eq. (C8c), but the variance estimator setting is more subtle. We set the variance estimator to the constant 1, which prevents the covariance matrix from being rescaled by a variance estimate of σ_{T_i} in Eq. (C7). We do this since we do not need to estimate σ_{T_i}—we already did this by performing our demonstration, and the weight matrix W reflects this knowledge. These settings are discussed in *Mathematica*'s Fit Models with Measurement Errors tutorial.

[116] We spell out the details for completeness, but the estimate for $\delta_{\hat{\lambda}}$ can be immediately returned from NLM using the "ParameterConfidenceIntervals" option without needing to compute the covariance matrix directly at all.

[117] https://reference.wolfram.com/language/tutorial/UnconstrainedOptimizationMethodsForLocalMinimization.html#216554821.

[118] Wikipedia, Akaike information criterion, https://en.wikipedia.org/wiki/Akaike_information_criterion (2023).

[119] S. Konishi and G. Kitagawa, *Information Criteria and Statistical Modeling* (Springer, New York, NY, 2008).

[120] Wikipedia, Nyquist–Shannon sampling theorem, https://en.wikipedia.org/wiki/Nyquist-Shannon_sampling_theorem (2023).

[121] W. Research, Interpolation, https://reference.wolfram.com/language/ref/Interpolation.html (2008).

[122] Wikipedia, https://en.wikipedia.org/wiki/Cubic_Hermite_spline Cubic Hermite spline (2022).

[123] Wikipedia, https://en.wikipedia.org/wiki/Spline_interpolation Spline interpolation (2022).

[124] U. Haeberlen, *High Resolution NMR in Solids*, Advances in Magnetic Resonance Series, Supplement 1 (Academic Press, New York, 1976).

[125] L. Viola and E. Knill, Verification procedures for quantum noiseless subsystems, Phys. Rev. A **68**, 032311 (2003).