Optimized Quantum Compilation for Near-Term Algorithms with OpenPulse

Pranav Gokhale, *University of Chicago*
Ali Javadi-Abhari, *IBM*
Nathan Earnest, *IBM*
Yunong Shi, *University of Chicago*
Frederic T. Chong, *University of Chicago*

***Abstract*—Quantum computers are traditionally operated by programmers at the granularity of a gate-based instruction set. However, the actual device-level control of a quantum computer is performed via analog pulses. We introduce a compiler that exploits direct control at this microarchitectural level to achieve significant improvements for quantum programs. Unlike quantum optimal control, our approach is bootstrapped from existing gate calibrations and the resulting pulses are simple. Our techniques are applicable to any quantum computer and realizable on current devices. We validate our techniques with millions of experimental shots on IBM quantum computers, controlled via the OpenPulse control interface. For representative benchmarks, our pulse control techniques achieve both 1.6x lower error rates and 2x faster execution time, relative to standard gate-based compilation. These improvements are critical in the near-term era of quantum computing, which is bottlenecked by error rates and qubit lifetimes.**

***Index Terms*—quantum computing, pulse control, OpenPulse**

I. Introduction

The present era of quantum computing is characterized by the emergence of quantum computers with dozens of qubits, as well as new algorithms that have innate noise resilience and modest qubit requirements. There are promising indications that near-term devices could be used to accelerate or outright-enable solutions to problems in domains ranging from molecular chemistry [1] to combinatorial optimization [2] to adversarial machine learning [3]. To realize these practical applications on noisy hardware, it is critical to optimize across the full stack, from algorithm to device.

Standard quantum compilers operate at the level of gates. However, the lowest-level of quantum control is through analog pulses. Pulse optimization has shown promise in previous quantum optimal control (QOC) work [4], [5], but we found that noisy experimental systems are not ready for compilation via QOC approaches. This is because QOC requires an extremely accurate model of the both the analog pulses that reach the qubits and the machine itself, i.e. its Hamiltonian. The analog pulses are subject to several errors, most notably sampling errors from the waveform generator [6] and phase offset errors from the temperature difference between classical electronics and the cold qubits [7], [8]. Moreover, Hamiltonians are difficult to measure experimentally and moreover, they drift significantly between daily recalibrations. The few experimental QOC papers address these issues with significant pre-execution calibration overhead such as staggered field calibration [5]. By contrast, we propose a technique that is bootstrapped purely from the daily calibrations that are already performed for the standard set of basis gates. The resulting pulses form our **augmented basis gate set**. These pulses are extremely simple, which reduces control error and also preserves intuition about underlying operations, unlike QOC. This technique leads to optimized programs, with mean 1.6x error reduction and 2x speedup for near-term algorithms.

Email: pranav@super.tech

We emphasize the generality of our approach and our compiler, which can target any underlying quantum hardware. We demonstrate our results via OpenPulse [6], [7], an interface for pulse-level control. In particular, our work is the first experimental demonstration of OpenPulse for optimized compilation of quantum programs (one prior paper used OpenPulse for noise extrapolation [9]). We executed pulse schedules on IBM's 20-qubit Almaden quantum computer, accessible through the cloud via the IBM Q Experience [10]. Our experience-building spanned over 11.4 million experimental shots, 4 million of which are explicitly presented here as concrete research outcomes. Our results indicate that pulse-level control significantly extends the computational capacity of quantum computers. Our techniques are realizable immediately on existing OpenPulse-compatible devices. To this end, all of our code and notebooks are available on Github [11].

We begin with background on quantum computing in Section II. Next, Section III presents an overview of standard quantum compilers and our compiler design (depicted in Figure 1). Sections IV– VII describe four key optimizations in our compiler, all of which are enabled by pulse-level control:

1) **Direct Rotations** (Section IV). Access to pulse-level control allows us to implement any single-qubit operation *directly* with high fidelity, circumventing inefficiencies from standard compilation.
2) **Cross-Gate Pulse Cancellation** (Section V). Although gates have the illusion of atomicity, the true atomic units are pulses. Our compiler creates new cancellation optimizations that are otherwise invisible.
3) **Two-Qubit Operation Decompositions** (Section VI). We recompile important near-term algorithm primitives for two-qubit operations directly down to the two-qubit interactions that hardware actually implements.
4) **Qudit Operations** (Section VII). Quantum systems have infinite energy levels. Pulse control enables d-level *qudit* operations, beyond the 2-level qubit subspace.

978-1-7281-7383-2/20/$31.00 ©2020 IEEE
DOI 10.1109/MICRO50266.2020.00027

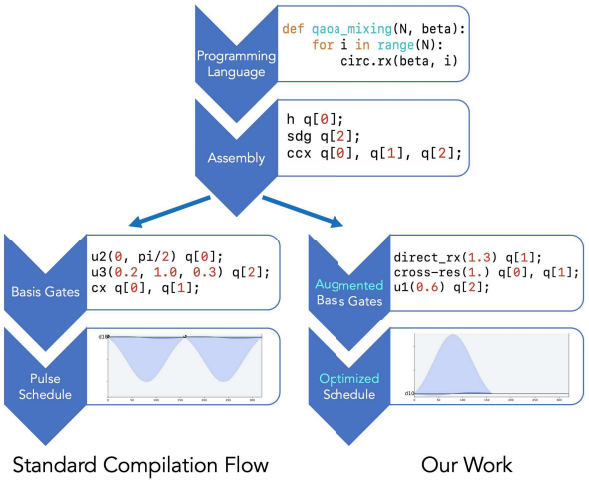


Fig. 1: Like classical programs, quantum programs undergo a compilation process from high-level programming language to assembly. However, unlike the classical setting, quantum hardware is controlled via analog pulses. In our work, we optimize the underlying pulse schedule by augmenting the set *basis gates* to match hardware. Our compiler automatically optimizes user code, which therefore remains hardware-agnostic.

Section VIII presents results from application of these techniques to full algorithms. We conclude in Section IX.

II. Background

We assume some familiarity with the fundamentals of quantum computing. Here we provide a brief review and expand on elements relevant to our work.

A. The Qubit

The core unit involved in quantum computation is the qubit (quantum bit). Unlike a classical bit which is either 0 or 1, a qubit can occupy any *superposition* between the two states, which are now denoted $|0\rangle$ and $|1\rangle$. The Bloch sphere, depicted in Figure 2, is a useful visual representation of the possible states of a qubit. The North Pole is the $|0\rangle$ state, and the South Pole is the $|1\rangle$ state. The state of a qubit can be parametrized by two angles, latitude and longitude. Upon measurement, a qubit *collapses* to either the $|0\rangle$ or $|1\rangle$ state, with probabilities dependent only on the latitude.

B. Quantum Gates

The set of valid single-qubit gates correspond to rotations around the Bloch sphere. Arbitrary such rotations are typically decomposed into $R_x(\theta)$ and $R_z(\theta)$ rotations around the X and Z axes respectively, which are universal for single-qubit rotations [12]. A prominent single-qubit gate is the $X = R_x(180°)$ gate, which in Figure 2 would rotate the green $|0\rangle$ state 180° around the X-axis to the $|1\rangle$ and vice versa. Thus, the X operation implements the NOT gate.

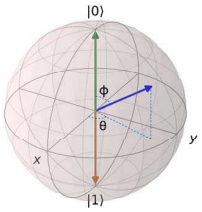


Fig. 2: The points on the Bloch sphere correspond one-to-one with possible qubit states. The green and brown states correspond to $|0\rangle$ and $|1\rangle$ respectively. The blue state is in a superposition described by latitude and longitude angles.

The set of possible multiple-qubit operations is much richer than the set of single-qubit gates, and lacks a clear visualization on the Bloch sphere. Remarkably however, any multiple-qubit operation can be decomposed into single-qubit rotations + an *entangling* gate such as CNOT [12]. The CNOT gate acts on a control and target qubit, and it applies X to the target iff the control is $|1\rangle$. In part because the CNOT gate is easy to understand, most quantum programs are expressed in terms of it. By implementing a small set of gates: single qubit rotations + CNOT, a quantum computer is universal. Accordingly, quantum computers are generally designed with this interface in mind. However, the lowest level of hardware control is performed by microwave pulses. Foreshadowing the main message of our paper: this pulse-backed layer actually provides a richer and "overcomplete" set of gates that outperforms the standard interface for quantum programs.

C. Gate Calibration

To implement this standard interface of universal gates, quantum computers are routinely calibrated to account for continuous drift in the experimental setting [13], [14]. As a concrete example, for superconducting devices, an $R_x(90°)$ gate is calibrated by performing a Rabi experiment [15]–[17] that determines the necessary underlying pulses. An additional DRAG [18]–[20] calibration fine-tunes the $R_x(90°)$ gate by cancelling out stray components. Calibrations in a similar spirit are also performed for the two-qubit gate(s). An interesting feature of the two-qubit gate calibrations is that they have the side effect of also calibrating $R_x(180°)$ pulses on each qubit. We exploit this free calibration in Section IV. Typically, $R_Z(\theta)$ rotation gates do not require calibration because they are implemented in software, as described in Section IV.

D. Experimental Setup

Our experiments were performed on IBM's Almaden, a 20 qubit device [21]. Almaden is the first cloud-accessible OpenPulse device. It comprises 20 transmon qubits, with mean T_1 and T_2 coherence lifetimes of 94 and 88 μs respectively. The mean single-qubit and two-qubit (CNOT) error rates are 0.14% and 1.78%. The mean measurement (readout) error was 3.8%, though we used measurement error mitigation [22], [23] to correct for biased measurement errors.

Stage	Notes	Example
Programming Language	High-level; hardware-unaware; sophisticated control flow	`qft(qc)`
Assembly	Usually 1- or 2- qubit arity gates; minimal control flow	`h q[0]`
Basis Gates	Like an HDL; hardware-aware gate set	`u1(3) q[0]`
Pulse Schedule	Analog waves across channels; ultimate "at the metal" control	

TABLE I: Summary of the four stages of a quantum compiler.

As of December 2019, IBM's publicly cloud-accessible OpenPulse device is the new Armonk device [24], which we used for the most recent results in Figure 13. For both Armonk and Almaden (and for IBM's devices in general), the calibrations described above are performed every 24 hours. Our experiments ran around-the-clock via a cloud job queuing system, with varying elapsed time to the prior calibration.

III. Compiler Flow

As depicted in Figure 1, quantum compilation proceeds through four stages, from high-level to low-level: programming language, assembly, basis gates, and pulse schedule. Also shown is our alternative flow, which creates an augmented set of basis gates and a more optimized pulse schedule. Table I presents a summary of these four stages. We now discuss existing implementations of these stages, why we should augment the standard set of basis gates, our new compiler framework, and the tradeoffs we considered when we designed the framework.

A. Standard Flow

1) Programming Language: Quantum PLs are designed to be user-friendly, with sophisticated control flow, debugging tools, and strong abstraction barriers between target operations and underlying quantum hardware. The most successful languages have been implemented as Python packages, such as IBM's Qiskit [25], Google's Cirq [26], and Rigetti's PyQuil [27]. Others are written as entirely new languages, such as Scaffold [28], [29] which is based on LLVM infrastructure; Quipper [30] which is a functional language embedded in Haskell; and Q# [31] which is Microsoft's quantum domain specific language.

2) Assembly: Quantum assembly languages are closer to hardware, but still aim to be device-agnostic. Generally, the assembly instructions only allow 1- or 2- qubit arity, since hardware primitives act on only 1 or 2 qubits at a time[1]. Quantum assembly is essentially equivalent to the quantum circuit representation of quantum programs. Prominent examples include OpenQASM [6], Rigetti's pyQuil [27], and TUDelft's cQASM [34].

[1]The notable exception is trapped ion quantum computers, which support global entangling operations that simultaneously act on N qubits [32], [33].

3) Basis Gates: Basis gates are similar to assembly, but re-expressed in terms of the gate set that hardware implements. For example, while the well-known Controlled-Z instruction is valid in assembly code, it would be re-written in basis gates as a sequence of H and $CNOT$ gates—which hardware natively implements. The distinction between assembly and basis gates is primarily a conceptual one; in Qiskit, Cirq, and PyQuil, the basis gate and assembly layers are expressed in the same software framework. In some other domains, for example the Blackbird language [35] for continuous-variable quantum computing, the assembly already resembles a hardware-aware basis gate layer. Regardless of the relationship between between assembly and basis gates, our core observation in this paper is that existing implementations of basis gate sets are too far from pulse-level hardware primitives. We will expand on this observation for the rest of the paper.

4) Pulse Schedule: The ultimate lowest-level control of a quantum computer is a schedule of complex-valued analog pulses, across multiple input channels. The image in Table I shows a sample pulse schedule on a single channel. The input channels are controlled by an Arbitrary Waveform Generator (AWG) which outputs a continuous value on each channel at every dt. Modern AWGs, such as the one in our experimental realization using IBM's Almaden system, achieve 4.5 Gigasamples per second, i.e. a new complex number every 0.22 ns.

The pulse schedule on drive channel j is referred to as $d_j(t)$ and is complex-norm constrained by $|d_j(t)| \leq 1$. However, qubits are not directly acted on by $d_j(t)$ or $\mathrm{Re}[d_j(t)]$. Instead, the $d_j(t)$ signal is mixed with a *local oscillator* of frequency f_j, leading to a final signal

$$D_j(t) = \mathrm{Re}[d_j(t)e^{if_jt}] \tag{1}$$

This equation will be relevant when we demonstrate qudit operations in Section VII.

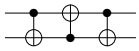
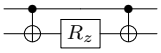
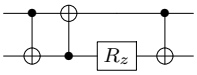
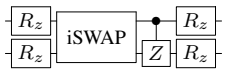
The translations from basis gates to pulse schedules are known analytically. For example, in superconducting quantum hardware, R_z basis gates are implemented in software with zero-duration and perfect-accuracy via the virtual-Z-Gate translation [36], [37]. The X basis gates is transformed into almost-Gaussian "DRAG" pulses [18]–[20]. In the OpenPulse interface, these translations are stored in the `cmd_def` object, and reported by the hardware.

B. Motivation for Different Basis Gates

At a high level, our core observation is that existing basis gates sets are too far from actual hardware primitives at the pulse-level. This leads to missed opportunities for optimization. Sections IV–VII will present optimizations resulting from specific gaps between basis gates and pulse-level hardware primitives. Table II introduces one such gap that we expand upon in Section VI. Each row in the table is a two-qubit operation. The columns express the cost[2] of performing the

[2]Cost here means the number of two-qubit gates needed, since they dominate both error and duration.

TABLE II: Costs of various two qubit operations, by Native gate. Cost reductions at the right indicate optimization opportunities. One $\sqrt{\text{iSWAP}}$ is treated as 0.5 cost, while iSWAP has 1.0 cost.

		Decomposition Cost by Native Gate						
		"Textbook"	Discrete Gates				Half	**Parametrized**
Operation	Standard Circuit Rep.	CNOT	CR(90°)	iSWAP	bSWAP	MAP	$\sqrt{\text{iSWAP}}$	**CR(θ)**
CNOT		1	1	2	2	1	1	**1**
SWAP		3	3	3	3	3	1.5	**3**
ZZ Interaction		2	2	2	2	2	1	**1**
ZZ swap		3	3	3	3	3	1.5	3
Fermionic Simulation		3	3	3	3	3	1.5	**3**
Fermionic Fourier Transformation		2	2	2	2	2	1	2
Bogoliubov Transformation		2	2	2	2	2	1	2

target operation using the given native gate. We computed these costs using Qiskit's `TwoQubitBasisDecomposer` tool, which uses the KAK decomposition [38] described further in [39].

The CNOT column indicates the number of CNOT gates needed to implement the target operation. CNOT is the default "textbook" two-qubit gate, so algorithms are usually written in terms of CNOT. The next group of four columns, Discrete Gates, captures basis gates from

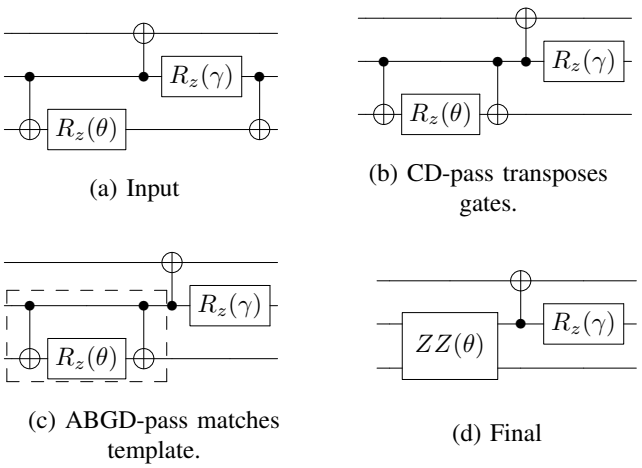
- Fixed-frequency superconducting qubits: 90° Cross-Resonance [40]–[42], bSWAP [43], and MAP [44].
- Frequency-tunable superconducting qubits: iSWAP [45] and also CZ [46], [47] which is omitted because it is equivalent to CNOT.
- Quantum dot spin qubits: iSWAP [48]
- Nuclear spin qubits: iSWAP [49]

All four of these columns have identical costs to the CNOT column. As a result of this parity, the prevailing sentiment in current quantum compilation software is that these basis gates are equivalent. Moreover, since quantum algorithms are usually written in terms of CNOTs, there is not an obvious reason to deviate from these basis gates.

The two rightmost columns, challenge this sentiment. The $\sqrt{\text{iSWAP}}$ reflects the fact that quantum hardware allows one to perform "half" of an iSWAP by damping the pulse shape of a standard iSWAP gate. This Half-gate leads to significant improvements over full iSWAPs–each row's cost is halved. The CNOT decomposition and SWAP decomposition are known [50], [51], but to the best of our knowledge, the ZZ Interaction (ubiquitous operation for quantum chemistry and optimization algorithms) and Fermionic Simulation (ubiquitous for quantum chemistry) decompositions are not previously known. They will have immediate applications on hardware that supports $\sqrt{\text{iSWAP}}$ such as frequency-tunable superconducting qubits, quantum dot spin qubits, and nuclear spin qubits.

The rightmost column, bolded because it was our experimental target, reflects the fact that fixed-frequency superconducting qubits support parametrized Cross-Resonance(θ) via pulse stretching. Since the native gate is parametrized, we used a different approach to compute the decomposition costs in its column. Specifically, we used the COBYLA constrained optimizer [52] in Scipy [53], with the constraint of finding a 99.9+% fidelity decomposition. Subject to this constraint, our decomposer minimizes the cost of the CR(θ) gates needed to perform the target operation. Observe that ZZ Interaction is 2x cheaper with a Parametrized CR(θ) gate than with the standard CR(90°) gate. The ZZ Interaction is in fact the most common two qubit operation in near-term algorithms. This optimization is expanded upon in Section VI.

Our method is extensible to other systems including trapped ions. Some of the trapped ion decompositions have already been studied in recent publications [54]–[56]. In Sections IV–VII, we present experimental realizations on IBM hardware, which corresponds to the CR(θ) column of Table II. However, our compiler framework is general and extends similarly to other native gate decompositions.



(a) Input

(b) CD-pass transposes gates.

(c) ABGD-pass matches template.

(d) Final

Fig. 3: Depiction of our compiler passes for commutativity detection (CD) and augmented basis gate detection (ABGD).

C. Design of Our Compiler

Our compiler is implemented as a fork of Qiskit. While Qiskit has traditionally been used in conjunction with IBM superconducting quantum computers, it is a generic framework that supports any underlying quantum hardware. For example, trapped ion quantum computer vendors have recently integrated with Qiskit [57], and OpenPulse support was recently added [58] to the XACC infrastructure for quantum-classical computing [59]. Thus our framework is general, though we performed our experimental realizations on IBM hardware, which is the first to implement OpenPulse.

Our compiler maintains the overall structure of Qiskit, which is already designed with extensibility in mind. As discussed previously, we augment the set of basis gates to better match pulse-level primitives. To support this augmented basis gate set, we re-write the decomposition rules from assembly instructions to basis gates and add new translations (to `cmd_def`) that convert augmented basis gates to pulse schedules. We expand on the augmented basis gates in Sections IV–VII.

To take advantage of the augmented basis gates, we added Qiskit transpiler passes, which convert input quantum assembly into optimized quantum assembly in the spirit of LLVM Transform passes. Our transpiler passes **automatically** optimize user code by using the augmented basis gates. One transpiler pass traverses a DAG-representation of the quantum assembly and pattern matches for templates that represent sequences of gates (such as the ZZ Interaction) that reduce to an augmented basis gate. We also include a commutativity detection transpiler pass that performs this pattern matching even when obfuscated by false dependencies in intermediate gates; this pass is inspired by techniques described in [60]. Figure 3 shows an example of these two passes. Through these two passes, we maintain the "write-once target-all" behavior of user-written code, which can remain hardware agnostic.

D. Compiler Design Tradeoffs

Another compiler design we considered is Quantum Optimal Control [61], [62], which translates directly from the programming language (specifically from the quantum circuit's overall unitary matrix) down to highly optimized pulses. QOC has been explored extensively in physics communities and more recently from an architectural perspective [60], [63], [64].

QOC is indeed a promising path for future machines, and in fact our original aim was to perform pulse-shaping via optimal control. However, our experience revealed experimental roadblocks. First, QOC requires faithful analog pulse generation. However, the actual pulses that reach qubits are limited by the AWG's finite sampling rate and by the temperature difference between classical electronics and the cold qubits. The former problem can be mitigated by smoothness constraints in the QOC engine, but doing so diminishes both the potential advantage of QOC and the reliable convergence of QOC algorithms [64], [65]. The latter problem is currently a major experimental barrier that requires a phase offset calibration for every candidate pulse [7, Appendix B]. Second, QOC requires a faithful characterization of the device Hamiltonian, since pulses designed from an inaccurate Hamiltonian accumulate substantial error. However, Hamiltonians are difficult to measure experimentally and moreover, they drift significantly between daily recalibrations. In addition, QOC requires evaluation of partial derivatives of a fidelity metric—a task that is easy analytically or in simulation, but extremely difficult with noisy experimental measurements.

Our experience is mirrored by other work on QOC—the vast majority of prior work has been performed via simulation. The few experimental realizations of QOC generally focus on state preparation (easier than unitary synthesis), e.g. [4], [5]. Moreover, these experiments impose significant Hamiltonian tomography or calibration overhead, for example staggered field calibration [5]. Experimental realizations on superconducting qubits, whose Hamiltonians drift over time [13], [14], are even more rare. In fact, the state-of-art for pulse shaping on superconducting qubits has eschewed QOC entirely [66], focusing instead on a closed-loop feedback for tuning pulses. We refer to [66] for further details on the experimental barriers to QOC, particularly in superconducting qubits.

Our approach to pulse-shaping arose from these limitations. In particular, our techniques are bootstrapped from the standard basis gate calibrations, which are already performed daily. By decomposing and then re-scaling the pre-calibrated pulses, we generate an augmented basis gate set, without ever requiring the device Hamiltonian. We emphasize that our technique can be applied on current cloud-accessible quantum devices, as documented in our Github repository [11]. Moreover, while QOC generally leads to convoluted pulses, our pulses are very simple. This simplicity minimizes the possibility of control errors and also leads to greater interpretability.

IV. Optimization 1: Direct Rotations

We now present the first of our four optimizations enabled by pulse control. The gist of this optimization is that pulse-level control enables us to perform single-qubit gates (qubit state rotations on the Bloch sphere) via a *direct* trajectory, saving time and potentially reducing errors.

It can be shown that any arbitrary single-qubit gate, termed U_3 in Qiskit, can be implemented by tuning up a single pulse that rotates the qubit state by 90 degrees around the X axis (the $R_x(90^\circ)$ pulse). This is doable due to the following identity, and due to the fact that rotations about the Z axis can be implemented in software at no cost (implemented by a compiler transformation on all future gates involving the target qubit) [36].

$$U_3(\theta,\phi,\lambda) = R_z(\phi{+}90^\circ)R_x(90^\circ)R_z(\theta{+}180^\circ)R_x(90^\circ)R_z(\lambda) \quad (2)$$

The above is extremely attractive from a hardware calibration perspective, since it suggests that fine tuning one pulse is enough to achieve high-fidelity single-qubit gates. In fact, this is how these gates are implemented on IBM quantum computers. We now present experimental evidence that access to one more calibrated gate, as well as pulse control, gives the compiler the ability to optimize single-qubit gates further.

A. Direct X gates

We first consider the simple X operation, which acts as a NOT by flipping $|0\rangle$ and $|1\rangle$ quantum states. X gates are ubiquitous in algorithms. Our approach relies on access to the $X = R_x(180^\circ)$ rotation, which is already pre-calibrated, as discussed in Section II-C. In our experiments we had access to such a pulse, but one could also be calibrated by the user through OpenPulse. We emphasize that this extra pulse is not strictly necessary for universal computation. However, we use it to demonstrate the power of an *overcomplete* basis for optimizations.

Qiskit's standard compilation flow decomposes an X operation into a U_3 instruction per equation 2. At the pulse level, the U_3 instruction is implemented by two consecutive $R_x(90^\circ)$ pulses. Together these complete an X gate (i.e. 180° rotation).

However, the indirection of implementing X with two $R_x(90^\circ)$ pulses becomes unnecessary in the presence of a pre-calibrated $R_x(180^\circ)$ gate. The procedure for calibrating such a gate is very similar to the $R_x(90^\circ)$, and its direct calibration has benefits beyond our discussion here [15]–[17], [67]. On IBM hardware enabled with OpenPulse, this pulse is readily available in the backend pulse library.

In our compiler, we exploit this simple observation by augmenting the basis gates with a `DirectX` gate, which is linked to the $R_x(180^\circ)$ pulse that is already calibrated on the quantum computer. This gate is twice as fast as Qiskit's standard X gate, and has 2x lower error, as measured through quantum state tomography experiments.

Figure 4 depicts a comparison of pulse schedules used to achieve the X gate in 71.1 ns in the standard framework vs. 35.6 ns in our optimization. It also illustrates why these two pulse schedules are logically equivalent: they have the same (absolute) area-under-curve. To a first approximation—which we will refine below—this area determines how much rotation is applied.

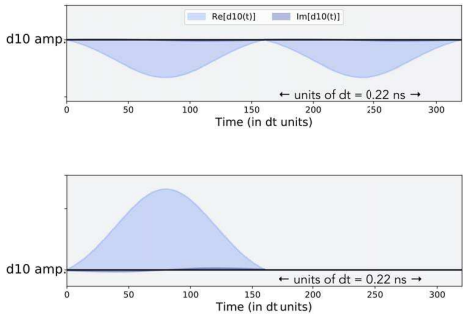


Fig. 4: Pulse schedules for the X gate via standard compilation (top) versus via direct compilation via our approach (bottom). Time is in units of dt = 0.22 ns. Thus, the `DirectX` gate takes 35.6 ns, twice as fast as the 71.1 ns standard X gate.

We next consider more sophisticated direct rotation gates.

B. Direct partial rotation about the X axis

Since OpenPulse gives us access to arbitrary pulse envelopes, it is natural to ask whether "partial" rotations about the X axis ($R_x(\theta)$ gates) can be realized more efficiently without invoking two discrete $R_x(90^\circ)$ pulses (as done by the standard Qiskit decomposition in Equation 2). Our compiler does this by downscaling the amplitude of the pre-calibrated $R_x(180^\circ)$ pulse by $\frac{\theta}{180^\circ}$ to achieve the $R_x(\theta)$ rotation. We represent this as the `DirectRx(`θ`)` augmented basis gate in our compiler. Since we rely on the pre-calibrated $R_x(180^\circ)$ this technique imposes no calibration overhead.

The results of our experiments with the new `DirectRx(`θ`)` are summarized in Figure 5. Bypassing the gate abstraction, our technique speeds up all R_x rotations by 2x and has 16% lower error on average. We discuss the source of the error reduction in Section VIII-C.

In the next subsection, we will note how `DirectRx(`θ`)` generalizes to arbitrary-axis rotations for free.

C. Optimizing generic rotations

Equipped with an augmented gate set that implements arbitrary X axis rotations at reduced cost, we now show that all single-qubit gates can be achieved with one pulse. Recall that in standard Qiskit compilation, general single qubit gates are implemented via two $R_x(90^\circ)$ pulses and three no-cost R_z frame changes. However, we can write the same gate as [12]:

$$U_3(\theta,\phi,\lambda) = R_z(\phi+180^\circ)R_x(\theta)R_z(\lambda-180^\circ) \quad (3)$$

Recall that R_z rotations are implemented by frame changes with perfect fidelity and 0 duration. Thus, this implies that any single-qubit gate can be performed using direct $R_x(\theta)$ rotations, sandwiched by free R_z gates.

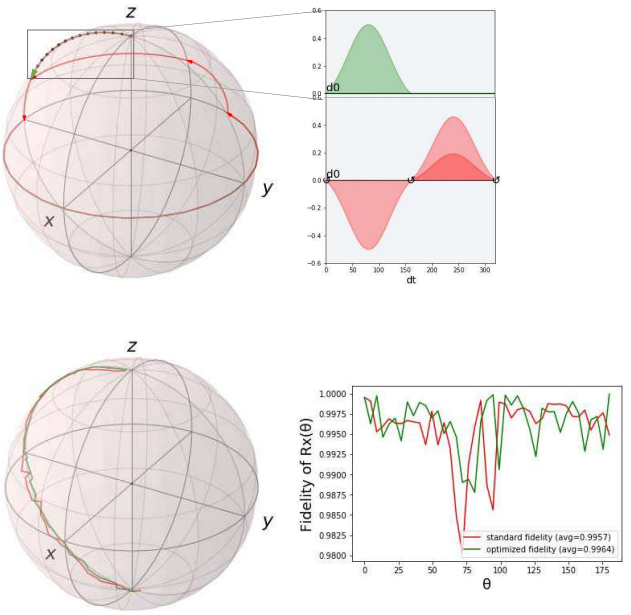
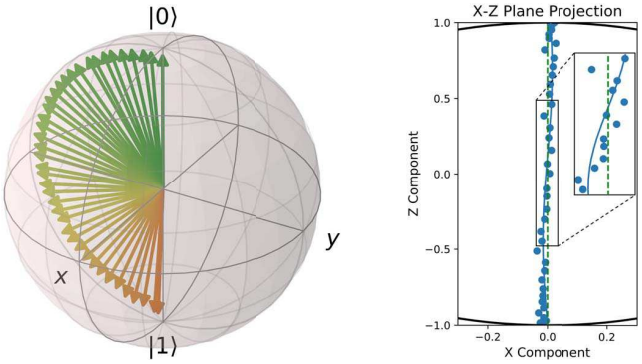


Fig. 5: Illustration of gate-level vs. pulse-level rotation about the X axis. (top) Trajectory of an $R_x(67^\circ)$ rotation, and the pulses that implement them. Standard gate-based compilation (red) includes two applications of the pre-calibrated $R_x(90^\circ)$ pulse (interleaved with R_z (frame changes) which are zero-cost and in software). Optimized pulse-based compilation takes the shortest path from origin to destination, with only one scaled pulse. (bottom) Fidelity of $R_x(\theta)$ rotations. Each data point is obtained using quantum state tomography experiments to rotate around the X axis by θ. Standard gate-compiled rotations (red) show more jitter from ideal, and 16% higher error on average, compared to optimized pulse-compiled rotations (green).



(a) Sweeping 41 angles from $\theta = 0^\circ$ in green to $\theta = 180^\circ$ in orange.

(b) XZ trajectory slightly deviates from $X = 0$.

Fig. 6: Simulated results for Direct $R_x(\theta)$. The inset magnifies the X component.

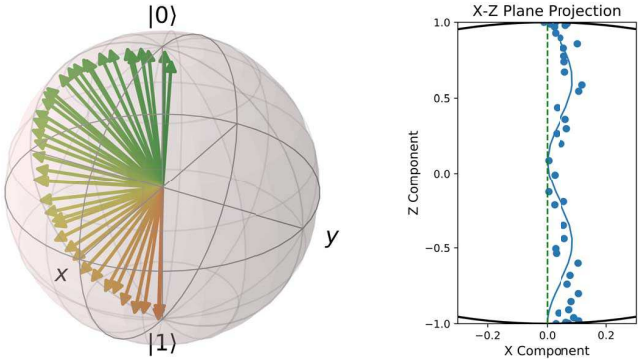


Fig. 7: Experimental results for Direct $R_x(\theta)$ on IBM's Almaden system, based on $3 \times 41 \times 1000 = 123$k shots. This empirical characterization of dephasing from the Meridian can be used to make the gate better at each θ.

D. Compiler implications

In the preceding subsection, we showed how an augmented gate set can be beneficial. However, the compiler now has more than the minimum set of pulses to work with to realize a quantum gate. In order to decide which pulses to use when, we need a deeper understanding and characterization of the errors incurred by $R_x(\theta)$ gates for arbitrary θ. We can use this system characterization to inform the compiler about the best pulse substitution strategy.

We performed pulse simulations and real experiments to gain insight into the errors. Our simulations were done using Qiskit's OpenPulse simulator. We enhanced the simulator to find the Hamiltonian terms for IBM's Almaden system, through a reverse-engineering process and fitting the results to the device-reported pulse library.

Taking Almaden's pre-calibrated direct X pulse (DRAG pulse), we scaled the area-under-curve down by a factor of $\frac{0}{40}, \frac{1}{40}, \frac{2}{40}, ..., 1$. To first order, these should perform $R_x(\theta)$ for $\theta = 0^\circ, 4.5^\circ, 9^\circ, ..., 180^\circ$. For each angle, we performed three simulations and three experiments to measure the X, Y, and Z components of the final quantum state, which allows us to plot on the Bloch sphere.

Figure 6 depicts the results of simulation. Plotting only the X-Z plane, we see that deviations from the Prime Meridian are quite small, but do have a sinusoidal pattern (at exactly 0°, 90°, and 180°, there is no dephasing). These simulation results are in agreement with an independent simulation from [37].

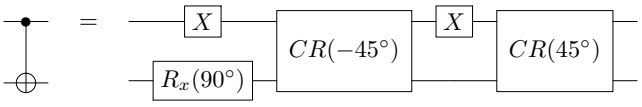
The experimental results are presented in Figure 7. We note two deviations from simulation: (1) the X components are still sinusoidal but now translated to the right and (2) the magnitude of the X-component deviations are larger. However, we can treat these characterization results with an empirical attitude—now that we know the dephasing at each θ value point, we can perform an $R_x(\theta)$ gate by applying a scaled-down X pulse, and then correcting the phase error in accordance with the data in Figure 7.

V. Optimization 2: Cross-Gate Pulse Cancellation

The gist of this optimization is that standard basis gates are not atomic[3], despite conveying this perception. By augmenting basis gates with the true atomic primitives, new gate cancellation opportunities emerge that lead to 24% speedups for common operations.

A. Theory

Generally, two-qubit basis gates are not atomic. For example, in Qiskit, the CNOT basis gate is implemented at the pulse level as a combination of single qubit gates, plus invocations of the hardware primitive Cross-Resonance pulse:



Notice in particular, that even the invocation of the hardware primitive Cross-Resonance pulse is not a clean atomic unit, but is decomposed into two pulses separated by an X gate. This "echoed" Cross-Resonance pulse design is necessary to perform a $CR(90°)$ gate (which is the generator of CNOT) with high fidelity [68].

This analysis reveals there are opportunities for gate cancellation on either side of the CNOT[4]. In fact, such sequences are common. To enable these cancellations, we augment the basis gate set with the hardware primitive $CR(\pm 45°)$ basis gates, which are free from pre-calibrated CNOTs. We replace the assembly instruction for CNOT into this decomposition and invoke Qiskit's optimizer to perform gate cancellations.

B. Application

To demonstrate our technique, we benchmarked using a common operation: the open-Controlled-NOT. The open-CNOT has the "opposite" behavior as a CNOT: it flips the target if the control is $|0\rangle$ and does nothing if the control is $|1\rangle$. Its implementation via the CNOT basis gate is simple: first an X on the control, then a standard CNOT, and then another X to restore the control.

However, by decomposing the CNOT into our augmented basis gates, the first X on the control cancels with the "internal" X in the decomposition of CNOT. Figure 8 depicts the pulse schedules for the open CNOT under standard compilation (top) and via our compilation (bottom). Notice that two X's in the red box cancel out, leading to a 24% reduction in runtime.

We tested the open-CNOT pulse schedules experimentally. To isolate the effect of cross-gate pulse cancellation, we performed the direct X gate from the previous section in both variants. The resulting data indicates a modest increase in success probability from 87.1(9)% to 87.3(9)%, measured over 16k shots (hence the Bernoulli standard deviation of 0.09%). While this is a small improvement in isolation, it is a lower bound that is magnified when we consider larger-width circuits. For example, in Figure 2 of [69], each of the $2m-2$ open-CNOTs is performed while $m-1$ other qubits are idle. Thus, speeding up these open-CNOTs also reduces the accumulated decoherence across the other qubits.

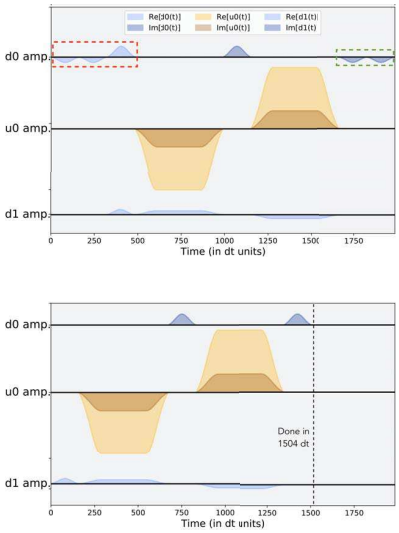


Fig. 8: Pulse schedules for the open-CNOT by standard compilation (top) and our optimized compilation (bottom). Our compiler cancels out the X rotation gates in the red box and combines the two $R_x(-90°)$ pulses in the green box into a single $R_x(180°) = X$ pulse. This reduces the total duration by 24% from 1984 dt to 1504 dt.

We emphasize that the open-CNOT is just one of many typical quantum operations that have R_x rotations next to two-qubit basis gates. Our compiler takes advantage of all such cancellation opportunities, which are otherwise invisible at the granularity of standard, non-atomic basis gates.

VI. Optimization 3: Two Qubit Optimizations

The gist of this optimization is that standard basis gates lead to inefficient decompositions of important two-qubit operations. Instead, we can use pulse-level hardware primitives as new basis gates that lead to operations with 60% lower error.

A. Theory

Recall from Table II that two-qubit operations can be achieved by using a "half" or parametrized basis gate set. For example, data movement (SWAP) is 2x more costly on superconducting qubits with an iSWAP basis gate than on qubits with a $\sqrt{\text{iSWAP}}$ basis gate.

Here, we study basis gate decompositions using the parametrized Cross-Resonance pulse $CR(\theta)$, which is the

[3]We use atomic in the common-usage sense of something that cannot be decomposed into something else more fundamental. This should not be confused with technical meanings of atomicity in computing.

[4]The circuit decomposition clearly depicts gate cancellation opportunities on the left side of the CNOT with the X and $R_x(90°)$ gates; alternatively, the top X can be shifted rightward by commutation identities to create cancellation opportunities on the right side

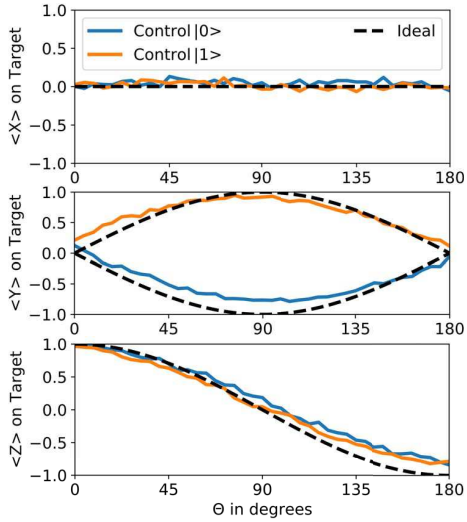


Fig. 9: Tomography on the target qubit in Cross-Resonance(θ) pulse. Results from both experiment and simulation agree with ideal results. $41 \times 3 \times 2 \times 1000$ = 246k shots.

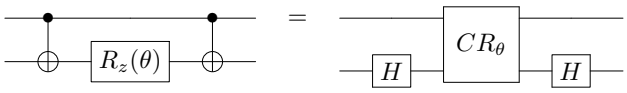
pulse-level hardware primitive on IBM devices. However, we again emphasize that our compiler techniques immediately generalize to any other basis gate decompositions.

As discussed in Section V, neither $CR(\theta)$ nor even $CR(90°)$ are exposed as standard basis gates. Our compiler first extracts the pulse for the $CR(90°)$ gate from the `cmd_def` pulse schedule for the CNOT basis gate. Then, to implement $CR(\theta)$ for arbitrary θ, we horizontally stretch the $CR(90°)$, guided by knowledge of IBM's specific "active cancellation echo" implementation of the Cross-Resonance pulse [67], [70].

Figure 9 shows our experimental results, which closely track with the ideal curve. Given the successful implementation of $CR(\theta)$ at the pulse level, we added it as a new basis gate.

B. Application

As indicated by the last row of Table II, the "ZZ Interaction" two-qubit operation can be implemented using a single CR(θ) gate. By contrast, the "textbook" implementation using standard basis gates requires two CNOTs. The CR(θ) decomposition is depicted below. While this decomposition is fairly simple in hindsight, we discovered it computationally using the optimization procedure mentioned in Section III-B.



To experimentally verify our ZZ Interaction technique, we implemented it using both the standard compiler (i.e. CNOT, $R_z(\theta)$, CNOT) and our optimized compiler (H, CR(θ), H) for θ spanning from $0°$ to $90°$ in $4.5°$ increments. As shown in Figure 10, our compiler achieves better results, with a 60% average reduction in error (state infidelity).

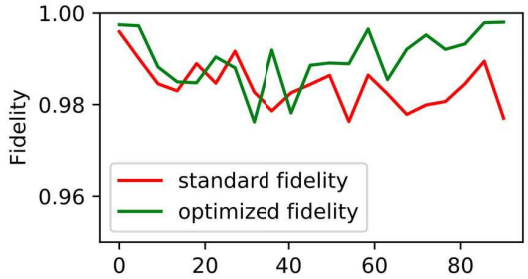


Fig. 10: Experimental results for state fidelity, measured for the ZZ Interaction by standard compilation vs. our optimized compilation. These results reflect $21 \times 2 \times 2000$ = 84k shots. Standard and optimized have fidelities of 98.4% and 99.0% respectively. Thus, our compiler achieves an average 60% reduction in error for the ZZ Interaction.

As we will see in Section VIII's Benchmark Results, the ZZ Interaction is the most frequent two-qubit operation in near-term algorithms. Thus, this optimization is the dominant source of improvements in full benchmarks. Before continuing, we re-iterate that our compiler passes (as discussed in Section III-C) automatically identify ZZ Interactions in user-code, even when obfuscated by false data dependencies. Therefore, programmers may continue to write code using "textbook" CNOT decompositions and do not need to reason about device physics.

VII. Optimization 4: Qudit Operations

The gist of this optimization is that access to quantum hardware at the pulse level enables us to control energy states outside the qubit subspace. In particular, we can instead control our information carriers as d-level *qudits*. We experimentally demonstrate this idea, by introducing three augmented basis gates for $|0\rangle \leftrightarrow |1\rangle$, $|1\rangle \leftrightarrow |2\rangle$, and $|2\rangle \leftrightarrow |0\rangle$, as well as a measurement discriminator for the three qutrit states. This enables us to implement a base-3 counter using a single *qutrit*, a task that would be impossible with a single qubit and enables using these energy states for more efficient gate decompositions [71]. The counter achieves high fidelity, suggesting practical near-term applications.

A. Theory

Many quantum systems used to realize a qubit have other energy levels present, which can be used to construct quantum gates [72]–[74] or, as we demonstrate in this paper, to realize d-level *qudits*. Substantial prior work observed an "information" compression advantage from using 3-level qutrits or higher level qudits [75], which has been further applied to specific algorithms such as Grover search [76]–[79] and Shor factoring [80]. More recent work [71] has even demonstrated exponential gains from using qutrits to implement common operations like the Generalized Toffoli.

However, across nearly all quantum hardware and associated software, standard basis gates are only written to address the qubit subspace of hardware. This is the case in part because the

local oscillator described in Section III-A4 is set to oscillate at the energy gap between the $|0\rangle$ and $|1\rangle$ energy states, f_{01}. Since higher level states are separated by different energy gaps, under normal operation, gates can only address this qubit subspace.

However, we can circumvent this limitation by carefully designing our pulse schedule. For example, suppose we want to address the $|1\rangle$ to $|2\rangle$ transition subspace, whose energy gap we denote as $f_{12} = f_{01} + \alpha$. Per Equation 1, applying a $d_j(t) = e^{-i\alpha t}$ pulse yields a total output of $e^{if_{12}t}$. Thus, by designing a frequency-shifting pulse schedule, we can change the effective frequency of the local oscillator and target subspaces beyond the $|0\rangle$ to $|1\rangle$ regime.

B. Application

By transitioning to these higher energy levels one at a time we can realize a base-d "counter". Not only is this a good benchmark for qudit control, it has potential application in both the near-term era and beyond. In the near-term era, parity checks are commonplace [81] and a counter (modulo d) serves this exact purpose. Qutrit measurement also enables error mitigation by detecting accidental leakages outside the qubit subspace [82], a technique that has immediate application to error mitigation for near-term algorithms. Beyond the near-term era, function evaluation oracles are ubiquitous and can be sped up via a counter. For example, recent work demonstrated that just a single qudit, acting as parity check, can implement an oracle-based quantum algorithm [83].

Here we demonstrate the ability to implement a counter via microwave control of a superconducting qubit, using two transitions previously inaccessible by standard basis gates. Specifically, we target the f_{12} and $f_{02}/2$ transitions (bottom right panel of Figure 11) which act on the $|1\rangle$ to $|2\rangle$ subspace and the $|0\rangle$ to $|2\rangle$ subspace respectively. The required drive strength and duration for these different transitions are dictated by the inherent coupling between each of the levels of interest, which is determined by the physics of the device. In the case of the two photon $f_{02}/2$ transition, the coupling between the $|0\rangle$ and $|2\rangle$ states is suppressed and thereby requires larger drive powers than those needed for an X gate between $|0\rangle \rightarrow |1\rangle$ transitions, with single photon powers around $p_{\text{one}} \approx 0.109$a.u. and two photon powers of $p_{\text{two}} \approx 0.44$a.u., each 35ns in duration. The f_{12} frequency can be measured either by applying an X gate on the $|0\rangle \rightarrow |1\rangle$ transition and subsequently performing qubit spectroscopy, or by driving a two photon $f_{02}/2$ transition and using the prior knowledge of f_{01} to determine f_{12}. Once the transition frequencies are identified, we calibrate the proper amplitude and duration of the pulses to fully switch the qubit to the desired final state.

To gauge the fidelity of our counter, we start off by training a linear discriminator to identify the qutrit state upon readout. In the case of this work, we train a sklearn [84] Linear Discriminant Analysis classifier with the calibrated qutrit $|0\rangle, |1\rangle, |2\rangle$ states and corresponding resonator IQ values (left panel of Figure 11). Once these calibrations are made, we measure the percentage of shots that have the qutrit in the $|0\rangle$ state at the end of the cycle. Due to imperfections in microwave control, our results deviate from the ideal of 1.0 as the number of cycles increase, making this an ideal testbed for further research such as improved microwave control [85], [86] and optimal readout parameters [87]. Nonetheless, the results indicate remarkably high fidelity—we can drive 60 cycles or 180 hops, before "dropout" exceeds 40%.

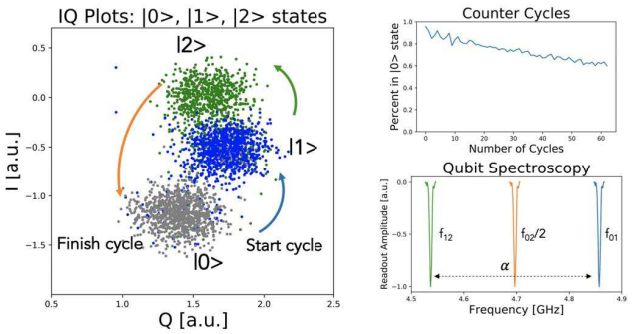


Fig. 11: (Left Panel) IQ Plot of readout resonator for different quantum states, and the specific cycle we follow. (Top Right) Percentage of shots found in the ground state as a function of the number of cycles. (Bottom Right) Different transition frequencies for the first three energy levels of a superconducting qubit with $2\pi f_{01} \sim 5GHz$ and $\alpha \sim$-300MHz and a two photon transition. These results span 150k experimental shots on IBM Almaden.

VIII. Results and Discussion

A. Benchmarks

We applied our compiler towards full quantum algorithms. Before proceeding, we note two thematic differences between our treatment of experimental benchmarks and that done in recent architectural work.

First, we focus exclusively on near-term algorithms. Some recent work [13], [14], [88]–[90] demonstrated impressive compiler optimizations for algorithms like Bernstein-Vazirani [91], Hidden-Shift [92], Adders, and Quantum Fourier Transform [93]. However, we emphasize that these algorithms are not representative of near-term algorithms, which are generally based on a *Hamiltonian simulation* kernel that quantum computers can naturally compute efficiently. Hamiltonian simulation, and thus near-term algorithms broadly, are dominated by the ZZ Interaction optimized in Section VI. We specifically evaluated three types of near-term algorithms: (1) Variational Quantum Eigensolver (VQE) [94], which addresses minimum-eigenvalue problems such as molecular ground state estimation; (2) Quantum Approximate Optimization Algorithm (QAOA) [2], which approximates solutions to NP-Hard combinatorial optimization problems; and (3) Hamiltonian Dynamics, which models molecular dynamics and was recently adapted for near-term applications [95], [96].

Second, we use Hellinger error/distance (or its complement, Hellinger fidelity) as our top-level metric. Intuitively,

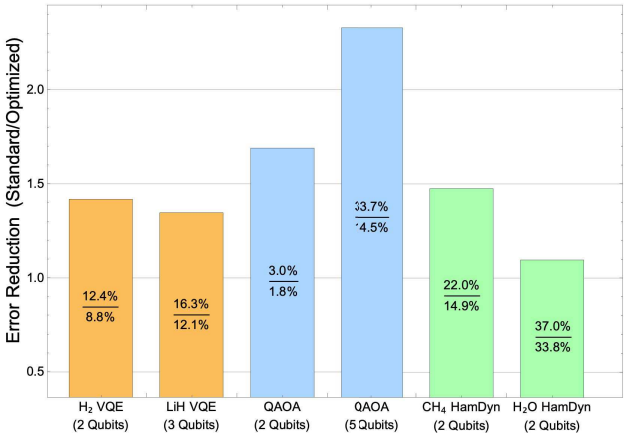


Fig. 12: Reduction in error (Hellinger distance) for benchmarks, due to our optimizations. These results reflect $6 \times 2 \times 8000 = 96$k shots on IBM Almaden.

Hellinger error captures the distance between two probability distributions: two identical distributions have the minimum distance of 0 and two completely antipodal distributions have the maximum distance of 1. Often, it is appealing to use Probability of Success (i.e. of finding the MAXCUT) as the top-level metric for algorithms like QAOA-MAXCUT [97], [98]. However, QAOA is not intended to find the MAXCUT with 100% successful probability (otherwise it would solve NP-hard problems in polynomial time), so a QAOA experiment with 100% "Probability of Success" would actually reflect high error. Instead, QAOA is intended to compute a *distribution* of measurement outcomes, within which bitstrings with large cuts will have boosted probabilities. This motivates our use of Hellinger error, and we urge subsequent experimental work to also evaluate near-term algorithms on the basis of probability-distribution distances.

B. Results

Figure 12 shows the reduction in error due to our optimizations. The H_2 and LiH VQE benchmarks replicate recent experimental work, [99] and [54] respectively. Both experiments are based on the Unitary Coupled Cluster ansatz [100]. The QAOA benchmarks compute MAXCUT on an N-qubit line graph. The Hamiltonian dynamics simulation benchmarks both simulate 6 Trotter steps. The methane and water Hamiltonians were generated with OpenFermion [101], taking advantage of orbital reductions to reduce the problems to two qubits.

For all six benchmarks, our optimized programs run with much lower error (Hellinger distance/infidelity) between the actual and target outcome distributions. The average error reduction factor is 1.55x and the largest benchmark, 5 Qubit QAOA, has a 2.32x reduction in error from 33.7% to 14.5%. The majority of our error reduction stems from our optimization of the ZZ Interaction by augmenting the basis gates with direct access to the Cross-Resonance pulse. We focus on Hellinger error because it is accepted in the quantum community as an (in)fidelity metric and has a "linear" interpretation. The average 1.55x error reduction factor is comparable to a year worth of hardware progress; of course, our method is achievable now and is performed in software. Similar work for QAOA was also recently demonstrated on Rigetti's hardware, using a parametrized XY interaction [102].

In addition to the six qubit benchmarks, we also ran the qutrit incrementer in Section VII and demonstrated 60 cycles, i.e. 180 increment operations, before "dropout" exceeds 40%. This benchmark is unique, because it has no standard qubit comparison—a single qubit cannot model a base-3 counter. This high-fidelity qutrit control confirms that pulse-backed basis gates offer a promising path towards qudit-based optimizations.

C. Source of Fidelity Improvements

The fidelity improvements presented here have three sources:

1) **Shorter pulses**. Our compiler's optimized pulses are shorter: 2x shorter for the single qubit rotations in Optimization 1, 24% shorter for open-CNOTs due to Optimization 2, and ∼2x shorter for ZZ interactions due to Optimization 3. These lower operation latencies are advantageous because qubits have less time to decohere.
2) **Less calibration error susceptibility**. `DirectRx(`θ`)` only applies one pre-calibrated (and then amplitude-downscaled) pulse. By contrast, the standard decomposition applies *two* pre-calibrated pulses, squaring the impact of calibration imperfections.
3) **Smaller pulse amplitudes**. Our pulse shaping techniques either vertically downscale amplitudes (Optimization 1) or horizontally stretch pulses (Optimization 3). As such, our pulse amplitudes are smaller than or equal to those generated by standard compilation. This is beneficial because smaller pulse amplitudes have smaller spectral components, reducing leakage to undesired frequency sidebands—see Figure 14 in [37] for details.

Our experience indicates that all three of these sources have meaningful contribution to the fidelity improvements. To further understand our fidelity improvements and reduce the impact of State Preparation and Measurement errors, we performed a Randomized Benchmarking [103] style experiment. In the experiment, we select $K-1$ random single-qubit unitary operations. We execute these $K-1$ operations, terminated with 1 final single-qubit operation that inverts all of the preceding operations. Therefore, under noise-free execution, the qubit returns to the initial state of $|0\rangle$ with 100% probability. However, due to noise, error accumulates as we increase K from 2 to 25.

Figure 13 presents our results, which ran over several hours on IBM's Armonk device. The *optimized* plot results from compiling with Optimization 1: Direct Rotations. However, to isolate the effect of shorter pulses, we also compiled *optimized-slow*, which inserts NO-OP idling into the optimized pulse schedules, to match the duration of the standard pulse schedules.

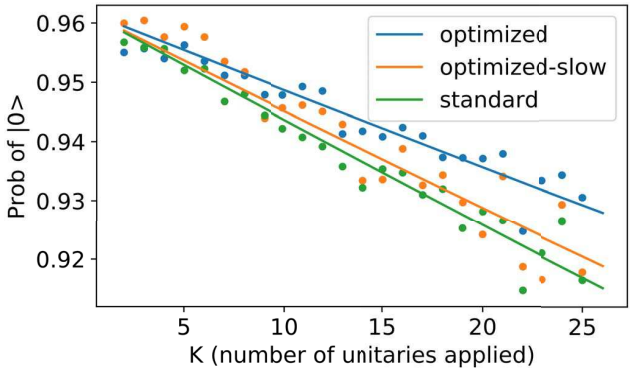


Fig. 13: Randomized Benchmarking style experiment, fit to exponential decay. For each K, we randomized 5 sequences of unitary operations. $5 \times 24 \times 3 \times 8$k = 2.88M total shots.

Each trajectory was fit to the exponential decay, $f^K - b$, where b is a y-intercept term that represents SPAM errors independent of K, and f is interpreted as gate fidelity. The resulting gate fidelities for optimized, optimized-slow, and standard are f = 99.87%, 99.83%, and 99.82%. This implies that shorter pulses (#1) account for 70% of the fidelity improvement, while less susceptibility to calibration imperfection (#2) and smaller pulse amplitudes (#3) account for the remaining 30% improvement. The improvement due to shorter pulses matches theoretical predictions: according to the gate error in coherence limit calculation, the 2x pulse speedup yields a minimum 0.01% fidelity improvement.

IX. Conclusion

Our results demonstrate that augmenting basis gates with pulse backed hardware primitives, bootstrapped from existing calibrations, leads to 1.6x error reductions and 2x speedups for near-term algorithms. Critically, our technique does not rely on knowledge of the system Hamiltonian, thus bypassing the experimental barriers to quantum optimal control. The measured fidelity improvements are equivalent to a year's worth of hardware progress, but our techniques are available immediately, through software.

We hope that our experiences with OpenPulse will encourage more quantum vendors to expose their hardware to pulse-level control. To this end, all of our code is available on Github [11] and can be used to run optimized quantum programs today. More broadly, our research suggests that hardware vendors should revise their basis gates to align more closely to what hardware can natively perform. On IBM systems, we saw significant gains from making the two-qubit basis gate the parametrized CR(θ) instead of the fixed CNOT. Similar gains can be achieved by exposing native two-qubit interactions—especially parametrized ones—as basis gates.

Acknowledgements

This work is funded in part by EPiQC, an NSF Expedition in Computing, under grants CCF-1730449; in part by STAQ under grant NSF Phy-1818914; and in part by DOE grants DE-SC0020289 and DE-SC0020331. This research used resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC05-00OR22725. Pranav Gokhale is supported by the Department of Defense (DoD) through the National Defense Science & Engineering Graduate Fellowship (NDSEG) Program.

References

[1] S. McArdle, S. Endo, A. Aspuru-Guzik, S. Benjamin, and X. Yuan, "Quantum computational chemistry," *arXiv preprint arXiv:1808.10402*, 2018.
[2] E. Farhi, J. Goldstone, and S. Gutmann, "A quantum approximate optimization algorithm," *arXiv preprint arXiv:1411.4028*, 2014.
[3] E. R. Anschuetz and C. Zanoci, "Near-term quantum-classical associative adversarial networks," *arXiv preprint arXiv:1905.13205*, 2019.
[4] C. Song, K. Xu, H. Li, Y.-R. Zhang, X. Zhang, W. Liu, Q. Guo, Z. Wang, W. Ren, J. Hao, *et al.*, "Generation of multicomponent atomic schrodinger cat states of up to 20 qubits," *Science*, vol. 365, no. 6453, pp. 574–577, 2019.
[5] A. Omran, H. Levine, A. Keesling, G. Semeghini, T. T. Wang, S. Ebadi, H. Bernien, A. S. Zibrov, H. Pichler, S. Choi, *et al.*, "Generation and manipulation of schr\" odinger cat states in rydberg atom arrays," *arXiv preprint arXiv:1905.05721*, 2019.
[6] D. C. McKay, T. Alexander, L. Bello, M. J. Biercuk, L. Bishop, J. Chen, J. M. Chow, A. D. Córcoles, D. Egger, S. Filipp, *et al.*, "Qiskit backend specifications for openqasm and openpulse experiments," *arXiv preprint arXiv:1809.03452*, 2018.
[7] 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," 2020.
[8] S. Krinner, S. Storz, P. Kurpiers, P. Magnard, J. Heinsoo, R. Keller, J. Luetolf, C. Eichler, and A. Wallraff, "Engineering cryogenic setups for 100-qubit scale superconducting circuit systems," *EPJ Quantum Technology*, vol. 6, no. 1, p. 2, 2019.
[9] J. Garmon, R. Pooser, and E. Dumitrescu, "Benchmarking noise extrapolation with openpulse," *arXiv preprint arXiv:1909.05219*, 2019.
[10] "Ibm q experience." https://quantum-computing.ibm.com/, 2019.
[11] "Optimizations via openpulse." https://github.com/singular-value/optimizations_via_openpulse, 2020.
[12] M. A. Nielsen and I. Chuang, "Quantum computation and quantum information," 2002.
[13] P. Murali, J. M. Baker, A. Javadi-Abhari, F. T. Chong, and M. Martonosi, "Noise-adaptive compiler mappings for noisy intermediate-scale quantum computers," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 1015–1029, ACM, 2019.
[14] S. S. Tannu and M. K. Qureshi, "Not all qubits are created equal: a case for variability-aware policies for nisq-era quantum computers," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 987–999, ACM, 2019.
[15] J. Majer, J. Chow, J. Gambetta, J. Koch, B. Johnson, J. Schreier, L. Frunzio, D. Schuster, A. A. Houck, A. Wallraff, *et al.*, "Coupling superconducting qubits via a cavity bus," *Nature*, vol. 449, no. 7161, p. 443, 2007.
[16] Y. Galperin, D. Shantsev, J. Bergli, and B. Altshuler, "Rabi oscillations of a qubit coupled to a two-level system," *EPL (Europhysics Letters)*, vol. 71, no. 1, p. 21, 2005.
[17] S. Ashhab, J. Johansson, and F. Nori, "Rabi oscillations in a qubit coupled to a quantum two-level system," *New Journal of Physics*, vol. 8, no. 6, p. 103, 2006.
[18] F. Motzoi, J. M. Gambetta, P. Rebentrost, and F. K. Wilhelm, "Simple pulses for elimination of leakage in weakly nonlinear qubits," *Physical review letters*, vol. 103, no. 11, p. 110501, 2009.
[19] J. M. Gambetta, F. Motzoi, S. Merkel, and F. K. Wilhelm, "Analytic control methods for high-fidelity unitary operations in a weakly nonlinear oscillator," *Physical Review A*, vol. 83, no. 1, p. 012308, 2011.

[20] F. Motzoi and F. K. Wilhelm, "Improving frequency selection of driven pulses using derivative-based transition suppression," *Physical Review A*, vol. 88, no. 6, p. 062318, 2013.

[21] B. Jones and M. Vyushkova, "Quantum computers flip the script on spin chemistry." https://www.ibm.com/blogs/research/2020/02/quantum-spin-chemistry/, 2020.

[22] F. B. Maciejewski, Z. Zimborás, and M. Oszmaniec, "Mitigation of readout noise in near-term quantum devices by classical post-processing based on detector tomography," *arXiv preprint arXiv:1907.08518*, 2019.

[23] Y. Chen, M. Farahzad, S. Yoo, and T.-C. Wei, "Detector tomography on ibm 5-qubit quantum computers and mitigation of imperfect measurement," *arXiv preprint arXiv:1904.11935*, 2019.

[24] A. Asfaw, T. Alexander, P. Nation, and J. Gambetta, "Get to the heart of real quantum hardware." https://www.ibm.com/blogs/research/2019/12/qiskit-openpulse/, 2019.

[25] H. Abraham, I. Y. Akhalwaya, G. Aleksandrowicz, T. Alexander, G. Alexandrowics, E. Arbel, A. Asfaw, C. Azaustre, P. Barkoutsos, G. Barron, L. Bello, Y. Ben-Haim, D. Bevenius, L. S. Bishop, S. Bosch, D. Bucher, CZ, F. Cabrera, P. Calpin, L. Capelluto, J. Carballo, G. Carrascal, A. Chen, C.-F. Chen, R. Chen, J. M. Chow, C. Claus, C. Clauss, A. J. Cross, A. W. Cross, J. Cruz-Benito, Cryoris, C. Culver, A. D. Córcoles-Gonzales, S. Dague, M. Dartiailh, A. R. Davila, D. Ding, E. Dumitrescu, K. Dumon, I. Duran, P. Eendebak, D. Egger, M. Everitt, P. M. Fernández, A. Frisch, A. Fuhrer, I. GOULD, J. Gacon, Gadi, B. G. Gago, J. M. Gambetta, L. Garcia, S. Garion, Gawel-Kus, J. Gomez-Mosquera, S. de la Puente González, D. Greenberg, J. A. Gunnels, I. Haide, I. Hamamura, V. Havlicek, J. Hellmers, Ł. Herok, H. Horii, C. Howington, S. Hu, W. Hu, H. Imai, T. Imamichi, R. Iten, T. Itoko, A. Javadi-Abhari, Jessica, K. Johns, N. Kanazawa, A. Karazeev, P. Kassebaum, A. Kovyrshin, V. Krishnan, K. Krsulich, G. Kus, R. LaRose, R. Lambert, J. Latone, S. Lawrence, D. Liu, P. Liu, P. B. Z. Mac, Y. Maeng, A. Malyshev, J. Marecek, M. Marques, D. Mathews, A. Matsuo, D. T. McClure, C. McGarry, D. McKay, S. Meesala, A. Mezzacapo, R. Midha, Z. Minev, M. D. Mooring, R. Morales, N. Moran, P. Murali, J. Müggenburg, D. Nadlinger, G. Nannicini, P. Nation, Y. Naveh, Nick-Singstock, P. Niroula, H. Norlen, L. J. O'Riordan, P. Ollitrault, S. Oud, D. Padilha, H. Paik, S. Perriello, A. Phan, M. Pistoia, A. Pozas-iKerstjens, V. Prutyanov, J. Pérez, Quintiii, R. Raymond, R. M.-C. Redondo, M. Reuter, D. M. Rodríguez, M. Ryu, M. Sandberg, N. Sathaye, B. Schmitt, C. Schnabel, T. L. Scholten, E. Schoute, I. F. Sertage, N. Shammah, Y. Shi, A. Silva, Y. Siraichi, S. Sivarajah, J. A. Smolin, M. Soeken, D. Steenken, M. Stypulkoski, H. Takahashi, C. Taylor, P. Taylour, S. Thomas, M. Tillet, M. Tod, E. de la Torre, K. Trabing, M. Treinish, TrishaPe, W. Turner, Y. Vaknin, C. R. Valcarce, F. Varchon, D. Vogt-Lee, C. Vuillot, J. Weaver, R. Wieczorek, J. A. Wildstrom, R. Wille, E. Winston, J. J. Woehr, S. Woerner, R. Woo, C. J. Wood, R. Wood, S. Wood, J. Wootton, D. Yeralin, J. Yu, L. Zdanski, Zoufalc, anedumla, azulehner, bcamorrison, brandhsn, dennis-liu 1, drholmie, elfrocampeador, fanizzamarco, gruu, kanejess, klinvill, lerongil, ma5x, merav aharoni, mrossinek, ordmoj, strickroman, tigerjack, yang.luh, and yotamvakninibm, "Qiskit: An open-source framework for quantum computing," 2019.

[26] "Cirq: A python framework for creating, editing, and invoking noisy intermediate scale quantum (NISQ) circuits." https://github.com/quantumlib/Cirq, 2018.

[27] R. S. Smith, M. J. Curtis, and W. J. Zeng, "A practical quantum instruction set architecture," *arXiv preprint arXiv:1608.03355*, 2016.

[28] A. J. Abhari, A. Faruque, M. J. Dousti, L. Svec, O. Catu, A. Chakrabati, C.-F. Chiang, S. Vanderwilt, J. Black, and F. Chong, "Scaffold: Quantum programming language," tech. rep., Princeton University Department of Computer Science, 2012.

[29] A. JavadiAbhari, S. Patil, D. Kudrow, J. Heckey, A. Lvov, F. T. Chong, and M. Martonosi, "Scaffcc: a framework for compilation and analysis of quantum computing programs," in *Proceedings of the 11th ACM Conference on Computing Frontiers*, p. 1, ACM, 2014.

[30] A. S. Green, P. L. Lumsdaine, N. J. Ross, P. Selinger, and B. Valiron, "Quipper: a scalable quantum programming language," in *ACM SIGPLAN Notices*, vol. 48, pp. 333–342, ACM, 2013.

[31] K. M. Svore, A. Geller, M. Troyer, J. Azariah, C. Granade, B. Heim, V. Kliuchnikov, M. Mykhailova, A. Paz, and M. Roetteler, "Q#: Enabling scalable quantum computing and development with a high-level domain-specific language," *arXiv preprint arXiv:1803.00652*, 2018.

[32] A. Sørensen and K. Mølmer, "Entanglement and quantum computation with ions in thermal motion," *Physical Review A*, vol. 62, no. 2, p. 022311, 2000.

[33] D. Maslov and Y. Nam, "Use of global interactions in efficient quantum circuit constructions," *New Journal of Physics*, vol. 20, no. 3, p. 033018, 2018.

[34] N. Khammassi, G. Guerreschi, I. Ashraf, J. Hogaboam, C. Almudever, and K. Bertels, "cqasm v1. 0: Towards a common quantum assembly language," *arXiv preprint arXiv:1805.09607*, 2018.

[35] N. Killoran, J. Izaac, N. Quesada, V. Bergholm, M. Amy, and C. Weedbrook, "Strawberry fields: A software platform for photonic quantum computing," *Quantum*, vol. 3, p. 129, 2019.

[36] D. C. McKay, C. J. Wood, S. Sheldon, J. M. Chow, and J. M. Gambetta, "Efficient z gates for quantum computing," *Physical Review A*, vol. 96, no. 2, p. 022330, 2017.

[37] P. Krantz, M. Kjaergaard, F. Yan, T. P. Orlando, S. Gustavsson, and W. D. Oliver, "A quantum engineer's guide to superconducting qubits," *Applied Physics Reviews*, vol. 6, no. 2, p. 021318, 2019.

[38] N. Khaneja and S. J. Glaser, "Cartan decomposition of su (2n) and control of spin systems," *Chemical Physics*, vol. 267, no. 1-3, pp. 11–23, 2001.

[39] A. W. Cross, L. S. Bishop, S. Sheldon, P. D. Nation, and J. M. Gambetta, "Validating quantum computers using randomized model circuits," *Physical Review A*, vol. 100, no. 3, p. 032328, 2019.

[40] G. Paraoanu, "Microwave-induced coupling of superconducting qubits," *Physical Review B*, vol. 74, no. 14, p. 140504, 2006.

[41] C. Rigetti and M. Devoret, "Fully microwave-tunable universal gates in superconducting qubits with linear couplings and fixed transition frequencies," *Physical Review B*, vol. 81, no. 13, p. 134507, 2010.

[42] J. M. Chow, A. Córcoles, J. M. Gambetta, C. Rigetti, B. Johnson, J. A. Smolin, J. Rozen, G. A. Keefe, M. B. Rothwell, M. B. Ketchen, *et al.*, "Simple all-microwave entangling gate for fixed-frequency superconducting qubits," *Physical review letters*, vol. 107, no. 8, p. 080502, 2011.

[43] S. Poletto, J. M. Gambetta, S. T. Merkel, J. A. Smolin, J. M. Chow, A. Córcoles, G. A. Keefe, M. B. Rothwell, J. Rozen, D. Abraham, *et al.*, "Entanglement of two superconducting qubits in a waveguide cavity via monochromatic two-photon excitation," *Physical review letters*, vol. 109, no. 24, p. 240505, 2012.

[44] J. M. Chow, J. M. Gambetta, A. W. Cross, S. T. Merkel, C. Rigetti, and M. Steffen, "Microwave-activated conditional-phase gate for superconducting qubits," *New Journal of Physics*, vol. 15, no. 11, p. 115012, 2013.

[45] J. M. Chow, *Quantum information processing with superconducting qubits*. Yale University, 2010.

[46] F. W. Strauch, P. R. Johnson, A. J. Dragt, C. Lobb, J. Anderson, and F. Wellstood, "Quantum logic gates for coupled superconducting phase qubits," *Physical review letters*, vol. 91, no. 16, p. 167005, 2003.

[47] L. DiCarlo, J. M. Chow, J. M. Gambetta, L. S. Bishop, B. R. Johnson, D. Schuster, J. Majer, A. Blais, L. Frunzio, S. Girvin, *et al.*, "Demonstration of two-qubit algorithms with a superconducting quantum processor," *Nature*, vol. 460, no. 7252, p. 240, 2009.

[48] D. Loss and D. P. DiVincenzo, "Quantum computation with quantum dots," *Physical Review A*, vol. 57, no. 1, p. 120, 1998.

[49] D. Mozyrsky, V. Privman, and M. L. Glasser, "Indirect interaction of solid-state qubits via two-dimensional electron gas," *Physical review letters*, vol. 86, no. 22, p. 5112, 2001.

[50] P. Echternach, C. P. Williams, S. Dultz, P. Delsing, S. Braunstein, and J. Dowling, "Universal quantum gates for single cooper pair box based quantum computing," *arXiv preprint quant-ph/0112025*, 2001.

[51] A. Lebedev, G. Lesovik, V. Vinokur, and G. Blatter, "Extended quantum maxwell demon acting over macroscopic distances," *Physical Review B*, vol. 98, no. 21, p. 214502, 2018.

[52] M. J. Powell, "A view of algorithms for optimization without derivatives," *Mathematics Today-Bulletin of the Institute of Mathematics and its Applications*, vol. 43, no. 5, pp. 170–174, 2007.

[53] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. Jarrod Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. Carey, İ. Polat, Y. Feng, E. W. Moore, J. Vand erPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and S. . . Contributors, "SciPy

1.0–Fundamental Algorithms for Scientific Computing in Python," *arXiv e-prints*, p. arXiv:1907.10121, Jul 2019.

[54] C. Hempel, C. Maier, J. Romero, J. McClean, T. Monz, H. Shen, P. Jurcevic, B. P. Lanyon, P. Love, R. Babbush, *et al.*, "Quantum chemistry calculations on a trapped-ion quantum simulator," *Physical Review X*, vol. 8, no. 3, p. 031022, 2018.

[55] M. Müller, K. Hammerer, Y. Zhou, C. F. Roos, and P. Zoller, "Simulating open quantum systems: from many-body interactions to stabilizer pumping," *New Journal of Physics*, vol. 13, no. 8, p. 085007, 2011.

[56] Y. Nam, J.-S. Chen, N. C. Pisenti, K. Wright, C. Delaney, D. Maslov, K. R. Brown, S. Allen, J. M. Amini, J. Apisdorf, *et al.*, "Ground-state energy estimation of the water molecule on a trapped ion quantum computer," *arXiv preprint arXiv:1902.10171*, 2019.

[57] A. Javadi-Abhari, P. Nation, and J. Gambetta, "Qiskit – write once, target multiple architectures." https://www.ibm.com/blogs/research/2019/11/qiskit-for-multiple-architectures/, 2019.

[58] T. Nguyen and A. McCaskey, "Enabling pulse-level programming, compilation, and execution in xacc," *arXiv preprint arXiv:2003.11971*, 2020.

[59] A. J. McCaskey, D. Lyakh, E. Dumitrescu, S. Powers, and T. S. Humble, "Xacc: a system-level software infrastructure for heterogeneous quantum-classical computing," *Quantum Science and Technology*, 2020.

[60] Y. Shi, N. Leung, P. Gokhale, Z. Rossi, D. I. Schuster, H. Hoffmann, and F. T. Chong, "Optimized compilation of aggregated instructions for realistic quantum computers," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 1031–1044, ACM, 2019.

[61] J. Werschnik and E. Gross, "Quantum optimal control theory," *Journal of Physics B: Atomic, Molecular and Optical Physics*, vol. 40, no. 18, p. R175, 2007.

[62] J. M. Chow, L. DiCarlo, J. M. Gambetta, F. Motzoi, L. Frunzio, S. M. Girvin, and R. J. Schoelkopf, "Implementing optimal control pulse shaping for improved single-qubit gates," 2010.

[63] J. Cheng, H. Deng, and X. Qian, "Accqoc: Accelerating quantum optimal control based pulse generation," in *Proceedings of the 47th International Symposium on Computer Architecture*, 2020.

[64] P. Gokhale, Y. Ding, T. Propson, C. Winkler, N. Leung, Y. Shi, D. I. Schuster, H. Hoffmann, and F. T. Chong, "Partial compilation of variational algorithms for noisy intermediate-scale quantum machines," in *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 266–278, ACM, 2019.

[65] N. Leung, M. Abdelhafez, J. Koch, and D. Schuster, "Speedup for quantum optimal control from automatic differentiation based on graphics processing units," *Physical Review A*, vol. 95, no. 4, p. 042318, 2017.

[66] Z. Leng, P. Mundada, S. Ghadimi, and A. Houck, "Robust and efficient algorithms for high-dimensional black-box quantum optimization," *arXiv preprint arXiv:1910.03591*, 2019.

[67] S. Sheldon, E. Magesan, J. M. Chow, and J. M. Gambetta, "Procedure for systematically tuning up cross-talk in the cross-resonance gate," *Physical Review A*, vol. 93, no. 6, p. 060302, 2016.

[68] A. D. Córcoles, J. M. Gambetta, J. M. Chow, J. A. Smolin, M. Ware, J. Strand, B. L. Plourde, and M. Steffen, "Process verification of two-qubit quantum gates by randomized benchmarking," *Physical Review A*, vol. 87, no. 3, p. 030301, 2013.

[69] R. Babbush, C. Gidney, D. W. Berry, N. Wiebe, J. McClean, A. Paler, A. Fowler, and H. Neven, "Encoding electronic spectra in quantum circuits with linear t complexity," *Physical Review X*, vol. 8, no. 4, p. 041015, 2018.

[70] E. Magesan and J. M. Gambetta, "Effective hamiltonian models of the cross-resonance gate," *arXiv preprint arXiv:1804.04073*, 2018.

[71] P. Gokhale, J. M. Baker, C. Duckering, N. C. Brown, K. R. Brown, and F. T. Chong, "Asymptotic improvements to quantum circuits via qutrits," *arXiv preprint arXiv:1905.10481*, 2019.

[72] M. J. Peterer, S. J. Bader, X. Jin, F. Yan, A. Kamal, T. J. Gudmundsen, P. J. Leek, T. P. Orlando, W. D. Oliver, and S. Gustavsson, "Coherence and decay of higher energy levels of a superconducting transmon qubit," *Physical review letters*, vol. 114, no. 1, p. 010501, 2015.

[73] H. Ribeiro and A. A. Clerk, "Accelerated adiabatic quantum gates: optimizing speed versus robustness," *arXiv preprint arXiv:1906.06737*, 2019.

[74] N. Earnest, S. Chakram, Y. Lu, N. Irons, R. K. Naik, N. Leung, L. Ocola, D. A. Czaplewski, B. Baker, J. Lawrence, *et al.*, "Realization of a λ system with metastable states of a capacitively shunted fluxonium," *Physical review letters*, vol. 120, no. 15, p. 150504, 2018.

[75] A. Pavlidis and E. Floratos, "Arithmetic circuits for multilevel qudits based on quantum fourier transform," *arXiv preprint arXiv:1707.08834*, 2017.

[76] Y. Fan, "Applications of multi-valued quantum algorithms," *arXiv preprint arXiv:0809.0932*, 2008.

[77] H. Li, C. Wu, W. Liu, P. Chen, and C. Li, "Fast quantum search algorithm for databases of arbitrary size and its implementation in a cavity qed system," *Physics Letters A*, vol. 375, no. 48, pp. 4249–4254, 2011.

[78] Y. Wang and M. Perkowski, "Improved complexity of quantum oracles for ternary grover algorithm for graph coloring," in *2011 41st IEEE International Symposium on Multiple-Valued Logic*, pp. 294–301, IEEE, 2011.

[79] S. Ivanov, H. Tonchev, and N. Vitanov, "Time-efficient implementation of quantum search with qudits," *Physical Review A*, vol. 85, no. 6, p. 062321, 2012.

[80] A. Bocharov, M. Roetteler, and K. M. Svore, "Factoring with qutrits: Shor's algorithm on ternary and metaplectic quantum architectures," *Physical Review A*, vol. 96, no. 1, p. 012306, 2017.

[81] S. McArdle, X. Yuan, and S. Benjamin, "Error-mitigated digital quantum simulation," *Physical review letters*, vol. 122, no. 18, p. 180501, 2019.

[82] S. Rosenblum, P. Reinhold, M. Mirrahimi, L. Jiang, L. Frunzio, and R. Schoelkopf, "Fault-tolerant detection of a quantum error," *Science*, vol. 361, no. 6399, pp. 266–270, 2018.

[83] Z. Gedik, I. A. Silva, B. Çakmak, G. Karpat, E. L. G. Vidoto, D. d. O. Soares-Pinto, E. Deazevedo, and F. F. Fanchini, "Computational speed-up with a single qudit," *Scientific reports*, vol. 5, p. 14671, 2015.

[84] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[85] C. Kabytayev, T. J. Green, K. Khodjasteh, M. J. Biercuk, L. Viola, and K. R. Brown, "Robustness of composite pulses to time-dependent control noise," *Physical Review A*, vol. 90, no. 1, p. 012316, 2014.

[86] C. Edmunds, C. Hempel, R. Harris, V. Frey, T. Stace, and M. Biercuk, "Dynamically corrected gates suppress spatio-temporal error correlations as measured by randomized benchmarking," *arXiv preprint arXiv:1909.10727*, 2019.

[87] T. Walter, P. Kurpiers, S. Gasparinetti, P. Magnard, A. Potočnik, Y. Salathé, M. Pechal, M. Mondal, M. Oppliger, C. Eichler, *et al.*, "Rapid high-fidelity single-shot dispersive readout of superconducting qubits," *Physical Review Applied*, vol. 7, no. 5, p. 054020, 2017.

[88] P. Murali, N. M. Linke, M. Martonosi, A. J. Abhari, N. H. Nguyen, and C. H. Alderete, "Full-stack, real-system quantum computer studies: Architectural comparisons and design insights," *arXiv preprint arXiv:1905.11349*, 2019.

[89] P. Das, S. S. Tannu, P. J. Nair, and M. Qureshi, "A case for multi-programming quantum computers," in *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 291–303, ACM, 2019.

[90] N. M. Linke, D. Maslov, M. Roetteler, S. Debnath, C. Figgatt, K. A. Landsman, K. Wright, and C. Monroe, "Experimental comparison of two quantum computing architectures," *Proceedings of the National Academy of Sciences*, vol. 114, no. 13, pp. 3305–3310, 2017.

[91] E. Bernstein and U. Vazirani, "Quantum complexity theory," *SIAM Journal on computing*, vol. 26, no. 5, pp. 1411–1473, 1997.

[92] A. M. Childs and W. van Dam, "Quantum algorithm for a generalized hidden shift problem," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 1225–1232, Society for Industrial and Applied Mathematics, 2007.

[93] D. Coppersmith, "An approximate fourier transform useful in quantum factoring," *arXiv preprint quant-ph/0201067*, 2002.

[94] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'brien, "A variational eigenvalue solver on a photonic quantum processor," *Nature communications*, vol. 5, p. 4213, 2014.

[95] H. R. Grimsley, S. E. Economou, E. Barnes, and N. J. Mayhall, "An adaptive variational algorithm for exact molecular simulations on a quantum computer," *Nature communications*, vol. 10, no. 1, pp. 1–9, 2019.

[96] M. Otten, C. L. Cortes, and S. K. Gray, "Noise-resilient quantum dynamics using symmetry-preserving ansatzes," *arXiv preprint arXiv:1910.06284*, 2019.
[97] S. S. Tannu and M. Qureshi, "Ensemble of diverse mappings: Improving reliability of quantum computers by orchestrating dissimilar mistakes," in *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 253–265, ACM, 2019.
[98] S. S. Tannu and M. K. Qureshi, "Mitigating measurement errors in quantum computers by exploiting state-dependent bias," in *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 279–290, ACM, 2019.
[99] P. J. O'Malley, R. Babbush, I. D. Kivlichan, J. Romero, J. R. McClean, R. Barends, J. Kelly, P. Roushan, A. Tranter, N. Ding, *et al.*, "Scalable quantum simulation of molecular energies," *Physical Review X*, vol. 6, no. 3, p. 031007, 2016.
[100] J. Romero, R. Babbush, J. R. McClean, C. Hempel, P. J. Love, and A. Aspuru-Guzik, "Strategies for quantum computing molecular energies using the unitary coupled cluster ansatz," *Quantum Science and Technology*, vol. 4, no. 1, p. 014008, 2018.
[101] J. R. McClean, K. J. Sung, I. D. Kivlichan, Y. Cao, C. Dai, E. S. Fried, C. Gidney, B. Gimby, P. Gokhale, T. Häner, *et al.*, "Openfermion: the electronic structure package for quantum computers," *arXiv preprint arXiv:1710.07629*, 2017.
[102] D. M. Abrams, N. Didier, B. R. Johnson, M. P. da Silva, and C. A. Ryan, "Implementation of the xy interaction family with calibration of a single pulse," *arXiv preprint arXiv:1912.04424*, 2019.
[103] E. Knill, D. Leibfried, R. Reichle, J. Britton, R. B. Blakestad, J. D. Jost, C. Langer, R. Ozeri, S. Seidelin, and D. J. Wineland, "Randomized benchmarking of quantum gates," *Physical Review A*, vol. 77, no. 1, p. 012307, 2008.